LNCS 7236

Sergio Di Martino
Adriano Peron
Taro Tezuka (Eds.)

Web and Wireless Geographical Information Systems

11th International Symposium, W2GIS 2012
Naples, Italy, April 2012
Proceedings

Springer

Sergio Di Martino   Adriano Peron
Taro Tezuka (Eds.)

Web and Wireless Geographical Information Systems

11th International Symposium, W2GIS 2012
Naples, Italy, April 12-13, 2012
Proceedings

Springer

Volume Editors

Sergio Di Martino
Adriano Peron
University of Napoli "Federico II"
Dipartimento di Scienze Fisiche
Via Cintia, 80127 Naples, Italy
E-mail: sergio.dimartino@unina.it; peron@na.infn.it

Taro Tezuka
University of Tsukuba
Graduate School of Library, Information and Media Studies
1-2 Kasuga, 305-8550 Tsukuba, Japan
E-mail: tezuka@slis.tsukuba.ac.jp

ISSN 0302-9743 e-ISSN 1611-3349
ISBN 978-3-642-29246-0 e-ISBN 978-3-642-29247-7
DOI 10.1007/978-3-642-29247-7
Springer Heidelberg Dordrecht London New York

Library of Congress Control Number: 2012934279

CR Subject Classification (1998): H.2.8, H.2-5, H.5, D.2, C.2

LNCS Sublibrary: SL 3 – Information Systems and Application, incl. Internet/Web and HCI

Typesetting: Camera-ready by author, data conversion by Scientific Publishing Services, Chennai, India

Printed on acid-free paper

Springer is part of Springer Science+Business Media (www.springer.com)

Preface

This volume contains the papers selected for presentation at the 11th edition of the International Symposium on Web and Wireless Geographical Information Systems (W2GIS 2012), hosted by the University of Naples "Federico II" in the dazzling city of Naples, Italy, in April 2012.

W2GIS is a series of events alternating between Europe and East Asia. It aims at providing a forum for discussing advances in theoretical, technical, and practical issues in the field of wireless and Internet technologies suited for the spreading, usage, and processing of geo-referenced data. W2GIS now represents a prestigious event within the research community that continues to develop and expand.

For the 2012 edition, we received 32 submissions from 12 countries in 4 continents. Each paper received three reviews and based on these reviews, 13 full papers and 4 short papers were selected for presentation at the symposium and inclusion in Springer LNCS Volume 7236. The accepted papers are all of excellent quality and cover topics that range from mobile GIS and location-based services to spatial information retrieval and wireless sensor networks.

We had the privilege of having a distinguished invited talk by Christopher B. Jones from the School of Computer Science at Cardiff University (UK). The best paper of the symposium was selected by the Steering Committee and invited to submit an extended version for publication in the *Journal of Spatial Information Science*.

We wish to thank all authors that contributed to this symposium for the high quality of their papers and presentations. Our sincere thanks go to Springer's LNCS team. We would also like to acknowledge and thank the Program Committee members for the quality and timeliness of their reviews. Finally, many thanks to Christophe Claramunt, Michela Bertolotto, and the entire Steering Committee members for providing continuous support and advice.

April 2012

Sergio Di Martino
Adriano Peron
Taro Tezuka

Organization

Symposium Chairs

Sergio Di Martino	University of Naples Federico II, Italy
Adriano Peron	University of Naples Federico II, Italy
Taro Tezuka	University of Tsukuba, Japan

Steering Committee

Michela Bertolotto	University College Dublin, Ireland
James D. Carswell	Dublin Institute of Technology, Ireland
Christophe Claramunt	Naval Academy Research Institute, France
Max J. Egenhofer	NCGIA, University of Maine, USA
Ki-Joune Li	Pusan National University, Korea
Kazutoshi Sumiya	University of Hyogo, Japan
Taro Tezuka	University of Tsukuba, Japan
Christelle Vangenot	Université de Genève, Switzerland

Local Organizing Committee

Francesco Cutugno	University of Naples Federico II, Italy
Vincenza Anna Leano	University of Naples Federico II, Italy
Valerio Maggio	University of Naples Federico II, Italy

Program Committee

Masatoshi Arikawa	University of Tokyo, Japan
Michela Bertolotto	University College Dublin, Ireland
Alain Bouju	University of La Rochelle, France
Tomas Brinkhoff	IAPG, Germany
Elena Camossi	JRC ISPRA, Ireland
James D. Carswell	Dublin Institute of Technology, Ireland
Christophe Claramunt	Naval Academy Research Institute, France
Rolf De By	ITC, The Netherlands
Sergio Di Martino	University of Naples Federico II, Italy
Matt Duckham	University of Melbourne, Australia
Max J. Egenhofer	NCGIA, University of Maine, USA
Filomena Ferrucci	University of Salerno, Italy
Peter Frohlich	FTW, Austria
Keith Gardiner	Dublin Institute of Technology, Ireland
Jerome Gensel	University of Grenoble, France
Yoshiharu Ishikawa	Nagoya University, Japan

Bin Jiang	University of Gävle, Sweden
Hanssan Karimi	University of Pittsburgh, USA
Yongjin Kwon	Korea Aerospace University, Korea
Ki-Joune Li	Pusan National University, Korea
Songnian Li	Ryerson University, Canada
Steve Liang	University of Calgary, Canada
Miguel R. Luaces	Universidade da Coruña, Spain
Eoin Macaoidh	JRC ISPRA, Italy
Herve Martin	University of Grenoble, France
Gavin Mcardle	National University of Ireland Maynooth, Ireland
Pedro R. Muro-Medrano	Universidad de Zaragoza, Spain
Adriano Peron	University of Naples Federico II, Italy
Dieter Pfoser	ATHENA Research Center, Greece
Cyril Ray	Naval Academy Research Institute, USA
Markus Schneider	University of Florida, USA
Kazutoshi Sumiya	University of Hyogo, Japan
Taro Tezuka	University of Tsukuba, Japan
Genny Tortora	University of Salerno, Italy
Taketoshi Ushiama	Kyushu University, Japan
Christelle Vangenot	Université de Genève, Switzerland
Stephan Winter	University of Melbourne, Australia

Additional Reviewers

Florczyk, Aneta Jadwiga
Guan, Lin-Jie
Kim, Kyoung-Sook
Kitayama, Daisuke
Lacasta, Javier
Renteria-Agualimpia, Walter

Sponsors

Table of Contents

Effects of Variations in 3D Spatial Search Techniques on Mobile Query Speed vs. Accuracy

Junjun Yin and James D. Carswell

Digital Media Centre, Dublin Institute of Technology, Ireland
yinjunjun@gmail.com, jcarswell@dit.ie

Abstract. This paper presents three Spatial Search Algorithms for determining the three dimensional visibility shape (threat dome) at a user's current location in a built environment. Users then utilize this multifaceted 3D shape as their query "window" to retrieve information on only those objects visible and stored in a spatial database. Such visibility shape searching addresses the information overload problem by providing "Hidden Query Removal" functionality for mobile LBS applications. This functionality will be especially useful in the *Web 4.0* "Future Internet of Things" era when trillions of micro-sensors placed throughout our built environment become available for discovery based on their geo-referenced IP address. In this paper we present and evaluate the effects that variations in mobile 3D query algorithms have on query speed vs. accuracy.

Keywords: MSI, Mobile LBS, Spatial Databases, Isovist 3D, Threat Dome.

1 Introduction

Visualisation of 3D built environment datasets on commercially available Smart-Phones (e.g. Google Maps 5 for Android 2.0+) is now reality. This is made possible by rendering the mobile map from a single set of vector data tiles instead of multiple sets of raster image tiles, and allows for smooth and continuous map viewing and scaling from different perspectives using the same set of vector data. Although Google Maps 5 is not yet photo realistic, the resulting 3D models are close to being geometrically accurate as they are derived from extruding building footprints to known heights for different parts of a building (Figure 1).

Within such a 3D vector dataset of Dublin, we have attached attributes (meta-data) to the various floors, windows, doors of buildings, plus affixed a range of (simulated) environmental sensor data streams to other scattered locations on a building's façade. Together this provides the beginnings of an "Internet of Things" type environment for testing our developed 2D/3D visibility-based spatial querying algorithms and techniques for addressing the "information overload" problem on mobile devices.

It is recognized that analyzing enormous volumes of data on mobile devices requires reducing "information overload" to eliminate display clutter. Allied research into the information overload problem is ongoing, where map personalisation and other semantic based filtering mechanisms are essential to de-clutter and adapt the exploration of the real world to the processing/display limitations of a mobile device [2, 3, 4, 5]. In this paper, we propose that another way to filter this information is to intelligently refine the search space by applying *hidden query removal* (HQR) functionality in three dimensions.

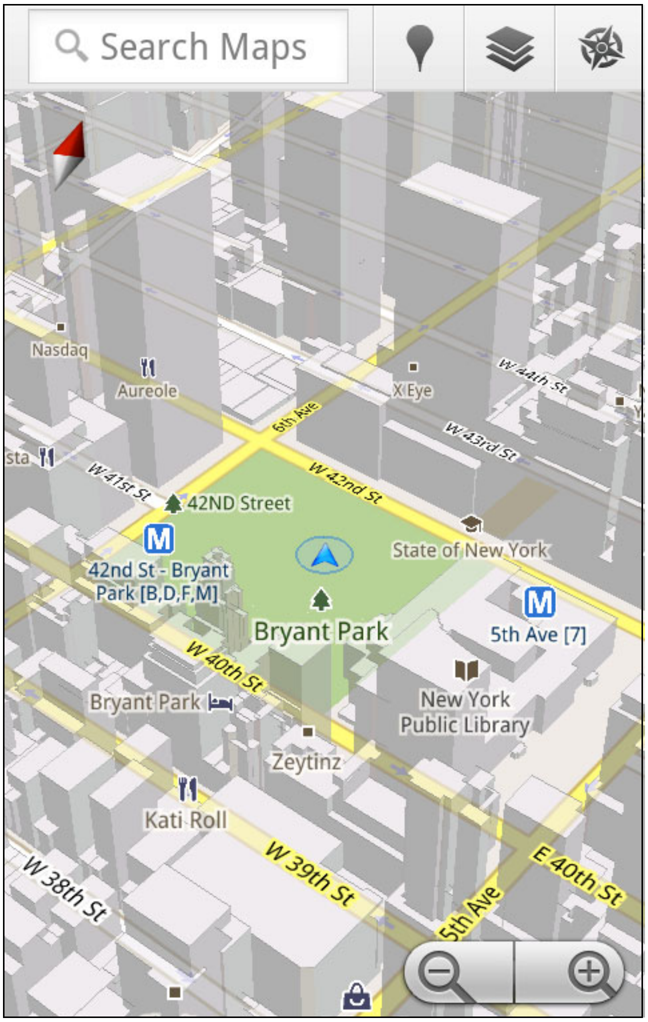


Fig. 1. 3D tilt, zoom, and rotation enabled mobile map displayed by *Google Maps 5 for Android* [1]

The combined effect gives a more accurate and expected query (search) result for Location-Based Services (LBS) applications by returning information on only those objects/sensor enabled "things" visible within a user's 3D field-of-view (FOV) as they move through a built environment. For example, visitors can now explore both their horizontal and vertical surroundings by pointing their smartphones at stores, offices, POIs, or any space to retrieve from the web any recorded information about these objects - answering specific questions such as: “Whose office window is that up there?” or more generally; “What are the air pollution readings along this street?” or perhaps more interestingly; “Can I see any CCTV cameras from where I’m sitting?” or indeed; “Are they seeing me?”.

However, to make our 3DQ (Three Dimensional Query) prototype function effectively in real-time requires mobile spatial query techniques that extend today’s spatial database technology both on the server and on the mobile device itself. This paper describes the various algorithms developed and results of tests carried out on COTS (commercial off-the-shelf) mobile devices querying our sample 3D vector dataset. The ultimate goal is to dynamically visualize on a SmartPhone the 3D query space, or threat dome, overlaid in real-time on a 3D mobile map, together with any returned search results (Figure 2).

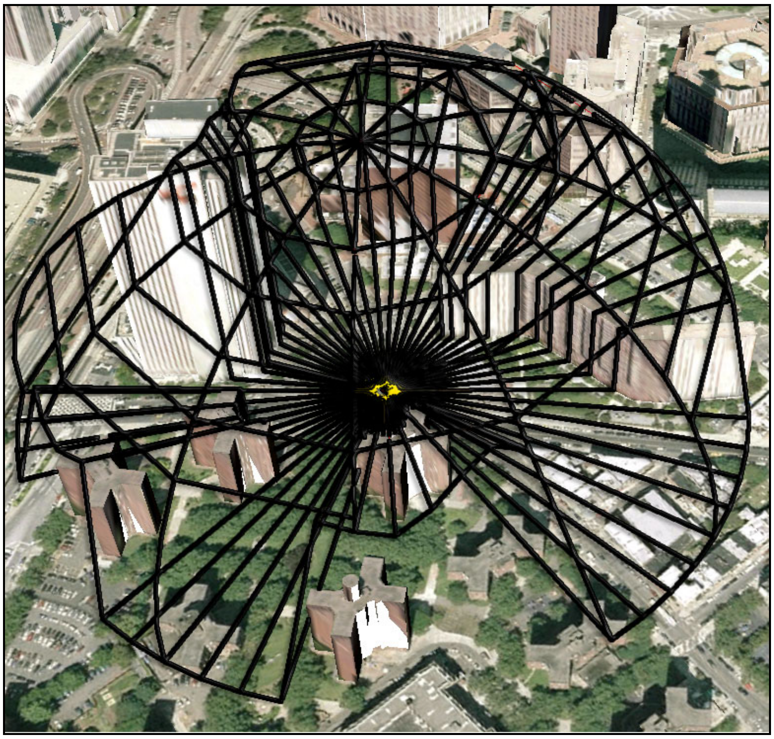


Fig. 2. Threat Dome search space interacting with a 3D cityscape model; only *things* intersecting the solid dome shape get returned by the query

Since 3DQ is intended to be deployed as a web-based service for mobile users, multiple users can connect and perform location based searches at the same time. Therefore, one significant research challenge was to efficiently and rapidly calculate the underlying visibility query shapes for each user. At the same time, maintaining the accuracy of the retrieved results is essential to providing a better user experience while reducing the information overload risk.

The remainder of this paper is organized as follows: Section 2 introduces the vector datasets that we utilize in our work. Section 3 describes our main contributions by presenting a comprehensive discussion of the algorithms and implementation of our 3DQ prototype. This is followed in Section 4 by some evaluations of the performance of 3DQ in terms of speed vs. accuracy when using different search algorithms and parameters, and Conclusions and plans for Future Work can be found in Section 5.

2 Vector Datasets Background

Geospatial information is increasingly recognized as the common denominator in both today's *web 2.0* peer-to-peer social network era and tomorrow's *web 4.0* – where it is envisioned that the Internet becomes connected to trillions of micro-sensors placed into real-world objects of all types (i.e. mechanical and non-mechanical), all with their own 128 bit IP address [6]. In other words, an "Internet of Things" that collects and sends time-stamped data to the cloud every second about their location, movement, plus any number of other measureable phenomena – e.g. environmental data such as air/water quality, ambient light/noise data, energy consumption, etc.

It is in this "Big Data" realm where we envisage 3DQ operating most effectively. When the potential of the sensor-web becomes realised, every brick of every building, every cobblestone of every street, every road sign, traffic light, street light, water bottle, beer can, garbage can and flower pot could conceivably be individually communicating their whereabouts and local conditions to the world. In such a world, we believe the ability to filter out, both semantically and spatially, unwanted/unnecessary/unsolicited information while at the same time retrieving task-relevant data for making informed decisions will be paramount.

Three dimensional indexing is a requirement for storing and querying 3D vector objects, which at time of writing limits our spatial database options to *Oracle Spatial 11gR2*, although *PostGreSQL* with their anticipated *PostGIS 2.0* extension for 3D indexing will be, once available, a useful open source addition to this very short list. However, we've discovered clear limitations of Oracle's spatial query operators when trying to determine the spatial relationships among 3D geometries. These include creating 3D R-Tree indexes on 3D geometries using a minimum-bounding cube. Oracle only considers if these cubes intersect with one another as a method for determining whether their underlying 3D geometries actually intersect. Retrieving the actual 3D location where two geometries (vector objects) intersect in 3D is not yet supported.

For example, to derive the true shape of any particular 3D search space, it is important to detect exactly where the intersection between a generated ray (simulating the 3D pointing direction of a SmartPhone) and a 3D building occurs. In this case, Oracle derives the intersection point using the 2D spatial operator *SDO_INTERSECTION* by first projecting the query shape (3D ray in this case) and the target (3D building) onto the ground plane and then only returning the 2D position of this 3D ray/building intersection. Using this information and combining it with the tilt angle and 2D distance to the nadir of the actual intersection point, we are left to compute the actual 3D intersection point of this query ourselves.

In an accurately computed threat dome, the generated dome shape will usually have a large number of surface elements relative to the complexity of the surrounding environment. If we want to consider this dome as a single 3D query shape (surface) in the form of an Oracle *SDO_GEOMETRY* (in order to make use of the *SDO_INTERSECTION* query operator), we find that its total number of surface elements will typically exceed the number of elements allowed in the *SDO_ELEM_INFO_ARRAY* - where it seems an arbitrary maximum of 999 coordinates (i.e. 333 3D points) are permitted.

In our case, where each surface element contains 12 coordinates defining its shape (four 3D vertices), a maximum of only 27 surface elements are then allowed in one *SDO_ORDINATE* list. As it happens, this is typically far fewer than what is required to accurately describe the boundary of a complete 3D threat dome shape. To get round this limitation, we must first split the complete threat dome into 3 or more sections and then query them individually against the database - instead of creating a single 3D volume as the threat-dome query shape. The returned query results are then a sum of all object/section intersections after first removing any duplication. Ironically, one beneficial consequence of this extra processing is that it encouraged us to mirror the spatial database across multiple servers and then send each individual 3D dome section query to a separate database. The end result is a much faster (~2sec.) query process which potentially allows for near real-time threat dome visualisations and searching on 3D mobile maps – our ultimate goal for this work.

Since the introduction of the "Isovist" concept in [7] for describing the 2D visibility shape or 3D visibility volume at a given position, there have been a number of developments that employ Isovist-like approaches for urban environment analysis. The notion of a "Spatial Openness Index" (SOI) developed in [8, 9] measures the volume of visual perception within a surrounding sphere from a given point of view, but without defining its shape. Other techniques to measure 2D and 3D visibility in an urban environment are shown in [10], which calculate the visibility of pixel coordinates on Digital Elevation Models (DEMs). Their proposed "iso-visi-matrix" claims to be a very useful from a visual perception viewpoint. Different to these approaches, visibility modelling algorithms developed in [11, 12], calculate the visibility of local landmarks in an urban context. They determine the visibility of a "Feature of Interest" (FOI) for location based services (LBS) that notify users when they are in a position that can actually see those landmarks. 3DQ acknowledges the importance and usefulness of carrying out 2D/3D visibility based analysis in the urban environment and aims to extend this idea by exploiting the actual 3D visibility shape as a query "window" to retrieve only those spatial objects that a user can physically see from a given viewpoint.

When calculating 3D Isovists, a user's visual perception is usually simulated and interpreted as a collection of "sight lines" or "line-of-sight" collisions with spatial objects in the environment [13]. In this regard, the technique of ray casting is a common approach to determine sight-line/object intersections where the collective intersection points of collisions eventually form the visibility shape used as the query window.

In most modern computer gaming applications, collision detection techniques are very well developed to determine and render only those visible objects in a game scene to optimize display speed. However, those type of calculations are normally Boolean value based operations, which means if the ray hits an object the returned value is “true” and vice versa. For the purpose of computer graphics layered rendering, this technique (without further calculation of the intersection point) are proven to be quite efficient [14, 15, 16]. However, our 3DQ prototype requires more than just determining which objects constitute a scene from a user’s viewpoint, we also need the 3D Isovist shape to determine where objects (e.g., built environment, sensors in the Sensor Web) are intersected. Therefore, accurate vertices (intersection points) of the visibility shape are necessarily required to form the corresponding search space in a spatial database for subsequent query processing operations.

3 3DQ Search Space Algorithms

The 3DQ system adopts a “client-server” architecture to deliver spatial searching as web-services for mobile devices. The services are in *RESTful* format style, where mobile device as client collects readings from its integrated sensors (e.g. GPS, compass, accelerometer), constructs them into a standard URL, and sends them to the server. Once the server finishes with the query calculations, the responses are organized and sent back in *GEO-JSON* format, which is OGC standard compatible and completely text based. A more detailed description of the 3DQ system architecture can be found in [17].

In a 2D scenario, the vertical dimension of a built environment is ignored in favour of the geometry of building footprints on the horizontal plane. Ideally, the length of a ray, which simulates a sight-line from a user’s view point, shall be infinite unless it hits an object along its path. However, to speed up the query calculation, we default the search length (user’s perception distance) to 200 meters. In other words, the footprints used to load the built environment around a user’s vicinity are limited to a 200 meters radius, thus speeding up considerably the query calculation. Options for users to adjust this search distance are also provided.

An example of a visibility search in a 2D environment (i.e. 2D Isovist) at a given location is shown in Figure 3 (a). The black square represents the user’s current location which is picked up from GPS on the mobile device. The surrounding built environment is constructed from the footprints of all building blocks within 200 meters. Benefiting from R-Tree indexing in *Oracle Spatial 11g*, the retrieval of all buildings from a given location is quite efficient [19]. The filled polygon is the user’s 360º visibility shape at that location. The 2D Isovist shape is then utilized as the query window to retrieve all database objects that intersect it. The Isovist construction process is based on the method developed in [18], which is an open source library for fast 2D floating-point visibility algorithms.

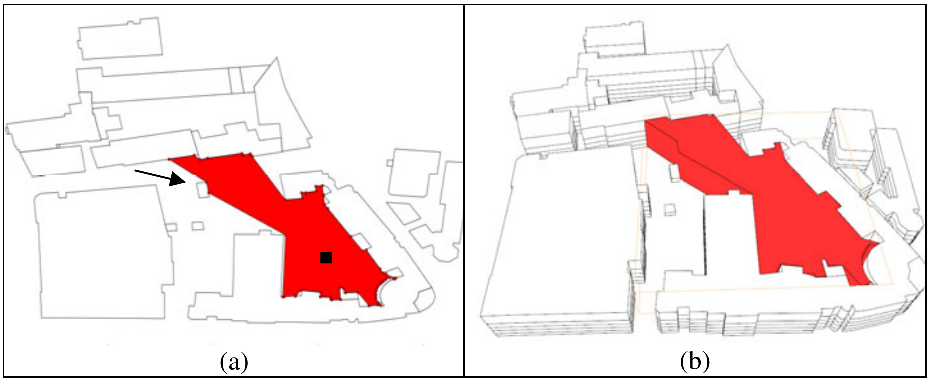


Fig. 3. (a) 2D Isovist view (b) Extruded 2.5D Isovist in a 3D environment

The 2D Isovist query is especially useful for conventional 2D mobile map searches, where it can serve as the preliminary filter to reduce the sometimes overwhelming amount of information available at a given location. However, as Google Maps 5 has progressed 2D mobile maps into 3D, a more realistic look and feel of a built environment is now available. Although, with this added vertical dimension come 3D visibility calculations that are much more complicated than in 2D. For instance, the arrow in Figure 3(a) points to a small building block where a 2D ray gets truncated, but the building's height is much lower than the surrounding buildings so a user's sight-line can in reality see over top of this block.

Although fast, to simply extrude a 2.5D Isovist (Figure 3(b)) would be incomplete as it does not pick up on this height difference. In fact, the results retrieved from an extruded 2.5D Isovist would be no different than those returned from a 2D Isovist, as the 3D coordinates for each returned object have the same x and y. Yet, to derive an accurate 3D Isovist, as shown in [20], is far too calculation intensive for real-time searching and therefore not optimal for serving multiple users as a mobile web-service. Therefore, we utilize ray casting techniques on vector datasets using a predefined length (radius) for each ray to save on computation effort. Thus the final query shape of the 3D Isovist appears as the "dome" shape shown in Figure 2 where the vertices that form the dome are the intersection points between each ray and any objects the ray hits out to 200m, or some other pre-specified distance.

An example demonstrating how vertices are detected in *Search Algorithm 1* is shown in Figure 4. In this illustration, assume building blocks with different heights are along the path a ray travels. On the horizontal plane, the interval between each ray is predefined at 6° by default (horizontal ray spacing). While on the vertical plane, we first detect what objects the ray hits and then calculate the corresponding 3D intersection point. The next ray along this same direction is initialized with a tilt angle of 15° and so on (vertical ray spacing).

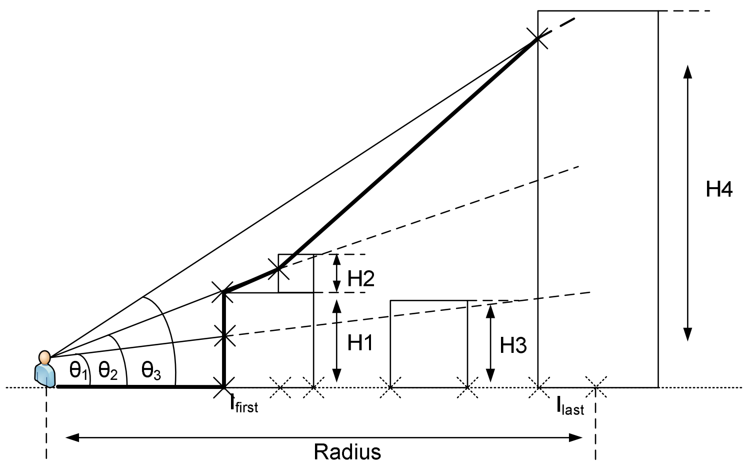


Fig. 4. Search *Algorithm 1* for determining sight-line intersections in a 3D environment. The thick black line outlines the final boundary of the threat dome in this direction.

The pseudo code for *Search Algorithm 1* follows:

Algorithm I. Tilting Ray Approach

```
-----------------------------------------

Input:  radius, horizontal ray spacing, tilt angle,
        current location
Output: A 3D threat dome visibility shape

Function RayTilting3D (radius, raySpacing, initalLo-
cation, tiltAngle):
Initialize ray generator from initialLocation
  Initialize final shape list: ShapePtCollect
  For each ray start with an initial tiltAngle:
    Initialize list: IntersectionPtCollect
    Get all the intersections and add to list
    Determine the first intersection: intersectionPt
    tiltAngle += AngleInterval (15° default)
    ShapePtCollect.append(intersectionPt)
  Return ShapePtCollect
```

As mentioned, Oracle does not return the exact 3D intersection point when a ray hits the building blocks. Instead, it projects the ray and building blocks onto a horizontal plane and returns a collection of 2D line segments. The intersection point is then determined by getting the first intersection point from all the segments, together with the tilting angle θ. The ray then continues to detect the next intersection point and so on until it stops once titling angle θ reaches 90°. This approach takes advantage of the 3D spatial query operators provided by *Oracle Spatial* as well as 3D spatial indexing and serves as a good approximation of the true dome shape. However, it can be seen in Figure 4 that building a threat dome using the tilting angle approach may miss certain intersection points vertically as the ray may overpass a building block because of the gap between any two tilting angles.

To improve on this approach, *Search Algorithm 2* acts like “reverse water-flow”, where the ray does not stop at the intersection point but instead continues on to determine the next intersection until it finally stops at the distance specified by the search radius (Figure 5). The process starts by determining the intersections on the horizontal plane between each ray and the projected footprints of the 3D building blocks. The actual intersections are a list of line segments and each of them has a pair intersection point <I_{in}, I_{out}>. The collection of I_{in} points will be picked up and ordered by their distance from the user’s location. We then determine the first intersection point of all I_{first}, which represents the first hit between a ray and the objects. The ray restarts from I_{first} and a tilting angle θ_1 is initialized once it reaches the top of the building. The next calculation happens at the next I_{in} point in the list, where if the height of the ray at that point is higher than the height of the building, the process carries on to the next I_{in} point in the list, otherwise, a new tilting angle is established and the same process iterates to the next I_{in} point.

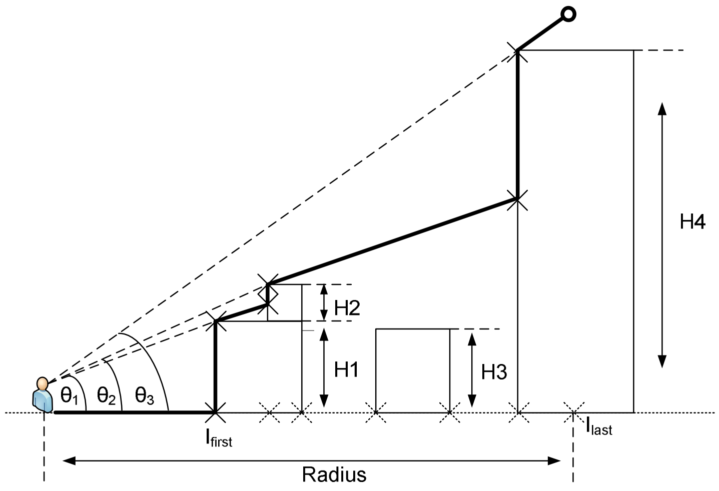


Fig. 5. *Search Algorithm 2* for determining sight-line intersections in a 3D environment. The thick black line outlines the final boundary of the threat dome in this direction.

Another advantage of *Search Algorithm 2* is that instead of using the tilting angle to generate multiple rays vertically, it only needs to process one ray on the horizontal plane and collect height information of all building blocks along its path. In *Search Algorithm 2*, the 3D building objects, which are represented as solids in *Oracle Spatial*, are replaced as 2.5D data structures with a height value attached as an attribute to each footprint and therefore uses 2D spatial indexing when deriving the initial intersection list, which has a simpler and faster data structure than 3D spatial indexing.

The pseudo code for *Search Algorithm 2* follows:

Algorithm II. Reverse Water-Flow Approach

```
----------------------------------------------------

Input:  radius, horizontal ray spacing, current location
Output: A 3D threat dome visibility shape

Function RaySweeping3D (radius, raySpacing, initialLocation):
    retrieve all building block geometries that are
    within the radius of a user's current location
  For each ray in the horizontal plane:
      initialize lists: HeightsCollect, DistanceCollect,
      IntersectionPtCollect
      derive all the intersection points from the ray:
        determine first intersection of each collision
      fill the lists
  sort the IntersectionPtCollect according to their
      distances from the initial location
  tgθ = Height(I_first)/Distance(I_first)
  initial final shape list: ShapePtCollect
  For each point in the list:
    If Height(I_next)< Dist(I_next)* tgθ = newHeight:
      pass
    Else:
      ShapePtCollect.append(I_next, Height= newHeight)
      ShapePtCollect.append(I_next, Height= Height(I_next))
      tgθ = Height(I_next)/Distance(I_next)
  Return ShapePtCollect
```

A common feature found in both Search Algorithms 1 and 2 is defined ray spacing when scanning for objects in the horizontal plane. However, no matter how small the interval is, there is still a risk of missing certain intersection points, which reduces the ultimate accuracy of the final query shape. As noticed in the 2D isovist calculation shown in Figure 3(a), the 2D visibility shape is continuous at filling every visible corner/gap between building geometries. Our third search approach therefore applies *Search Algorithm 2* on top of a 2D Isovist shape. Once a 2D Isovist is calculated, each vertex of the Isovist polygon together with the user's location defines a ray direction with the length equal to the predefined radius. An example is shown in Figure 6, where the rays travel through each of the vertices in a 2D Isovist polygon instead of being evenly specified by a horizontal ray spacing value. Although it involves two steps of calculation, it provides a more accurate visibility shape.

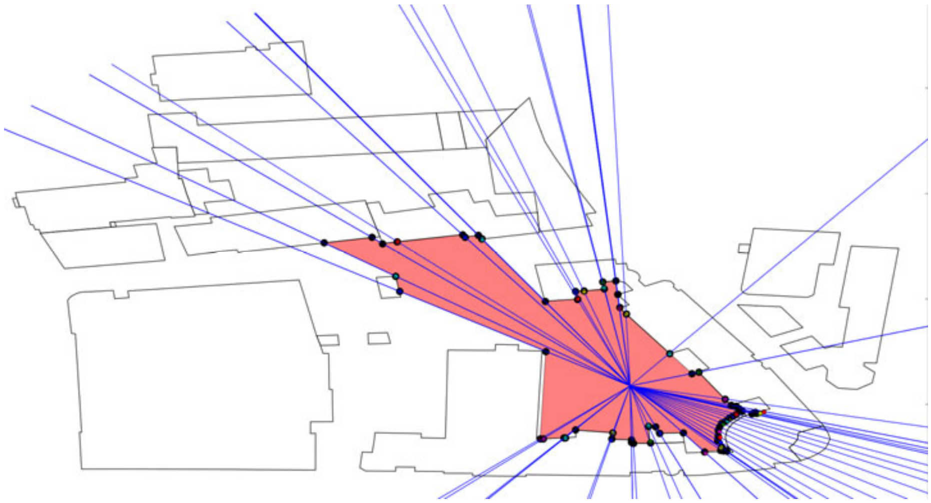


Fig. 6. An example of generating object intersection rays through the vertices of a 2D Isovist

The pseudo code for *Search Algorithm 3* is follows:

Algorithm III. 2D Isovist Based Ray Sweeping Approach

```
------------------------------------------------------------------

Input:  radius, user' current location
Output: A 3D threat dome visibility shape

Function RayIsovist3D (radius, initialLocation):
  retrieving all the 2D footprints within the radius
   from current location
  calculating 2D isovist
  initial final shape list: ShapePtCollect
  For each vertices of the Isovist generate a ray
    through the initialLocation:
     applying Algorithm II
  Return ShapePtCollect
```

As mentioned previously, in an accurately computed threat dome, the generated dome will usually have a large number of surfaces relative to the complexity of the surrounding environment. We must therefore split the complete threat dome into 3 sections and then query them individually against the database - instead of creating a single 3D volume for the threat-dome query shape. This led us to mirror the spatial database across three servers and then send each individual 3D dome section query to a separate database. In today's cloud computing era, it may be possible to deploy this approach as a cloud-based service with multiple spatial databases running in different virtual machines at the same time. The result is potentially a much faster query process that allows for near real-time threat dome visualisations on 3D mobile maps.

4 Evaluation of Query Speed vs. Accuracy

In this section, we present our performance evaluation of the 3 Searching Algorithms implemented in terms of speed vs. accuracy. The datasets used include 2D footprints of DIT campus with actual heights stored as a non-spatial attribute, 3D wireframe building outlines stored as 3D multi-polygons, and extruded 3D solids from the 2D footprints up to the stored heights. Plus 100 3D points simulating environmental sensor "things", which are attached to the surfaces of the building blocks (e.g., windows, walls, and doors, etc.) and other objects (e.g. light posts) in the database. A screenshot of the combined dataset is shown in Figure 7.

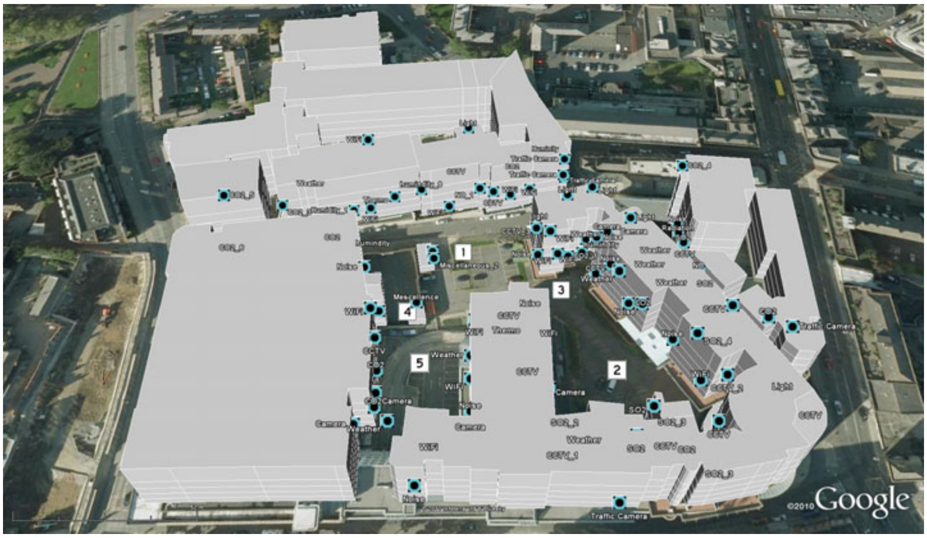


Fig. 7. 3D model of university campus affixed with 100 synthesized environmental sensors

The evaluation of each search algorithm was carried out at five different locations in the campus complex. To test our approaches on large datasets, as when deployed in a real-world application, the 2D footprints consists of 345,316 polygons with height values attached, and cover most of Dublin city. The datasets are stored in 3 *Oracle Spatial* databases mirrored on three different machines. More specifically, machine one is running Windows 7 (32 bit) with 4G RAM and Core2Duo 2.8 GHz CPU, the other two machines are running virtual machines under Windows XP (32 bit) with 2G RAM and Core2Duo 2.2 GHz CPU. The programming language is Python 2.7 with the capability of providing OGC standard compatible *Geo-JSON* output to mobile devices.

Figure 8 shows a comparison of query speed for the three different search algorithms. *Search Algorithm 1* was applied to a limited 3D solid dataset of the nearby campus area while *Search Algorithms 2 & 3* were applied against a complete 2D polygon map of Dublin City with building heights stored as a non-spatial attribute. Notice how 3D querying on 2D datasets proves to be usefully quicker - even though

Oracle Spatial R-tree indexing limits the search space to only those database objects that are within a specified 2D radius in either case.

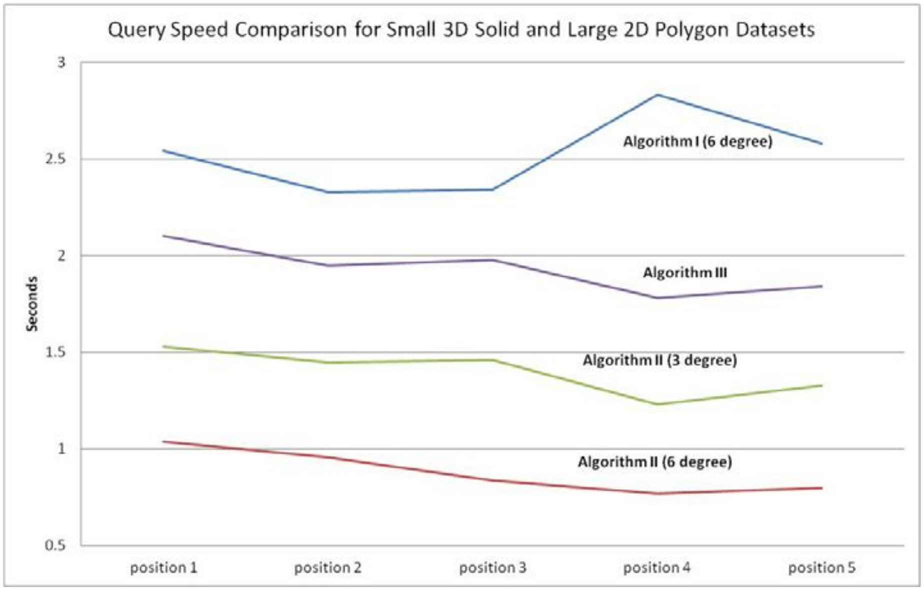


Fig. 8. Comparisons of query speed for different search algorithms at 5 positions on 3D solid and 2D polygon datasets of Dublin

Table 1 shows a comparison of accuracy for all visibility query approaches. Compared to a standard range query, where returning information on all 100 sensored *things* would overload the mobile display at every query position, *Search Algorithm 3* is shown to return the most actual visible sensors in every query position.

Table 1. Comparisons of accuracy of 3 Search Algorithms at 5 query positions together with the maximum number of sensors actually visible at each position

	Position 1	Position 2	Position 3	Position 4	Position 5
Algorithm I (6^o ray spacing)	19/33	14/32	20/34	5/12	12/26
Algorithm II (6^o ray spacing)	20/33	18/32	20/34	9/12	13/26
Algorithm II (3^o ray spacing)	25/33	26/32	29/34	10/12	20/26
Algorithm III	30/33	29/32	32/34	11/12	24/26

The above query speed experiment was run again on a complete Dublin City 3D solid dataset using all search algorithms. The results of this test are shown in Figure 9. It can be seen that for each search algorithm, the time taken to complete a 3D query on 3D solid data is noticeably longer than when performed on 2D data of the same area with height stored as a non-spatial attribute. The reason for such a large dip in Algorithm 3 timings at Position 4 is because there are only 12 sensors actually visible due to the restricted visible search space at this location.

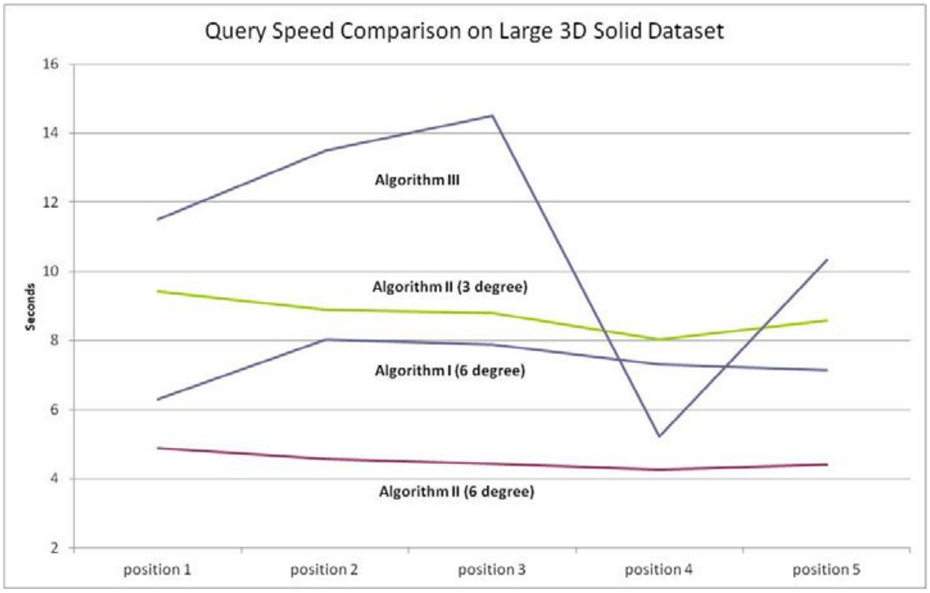


Fig. 9. Comparisons of query speed for different search algorithms at 5 positions on 3D solid dataset of entire Dublin City

5 Conclusions and Future Work

This paper presented three Spatial Search Algorithms developed in our 3DQ prototype for determining the 3D visibility shape (threat dome) at a user's current location in a built environment. Users then utilize this 3D shape as their query "window" to retrieve information on only those objects visible within a spatial database. Visibility shape searching addresses the information overload problem by providing "Hidden Query Removal" functionality for mobile LBS, whereas conventional spatial queries apply either a bounding box or circle for their search space with no concern for actual visibility of returned results.

We believe that this functionality will be especially useful in the impending *Future Internet* era when trillions of micro-sensors become available for query through IP access. Those sensors will be deployed throughout the 3D built environment, such as on different floors, windows, walls, street furniture, plants, people, etc. Therefore, it becomes increasingly essential to know accurately the 3D visibility search space over contemporary 2D bounding boxes at any given location. Analogous to scenes from *Star-Trek*, with Spock scanning his *tri-corder* for readings of life and atmospheric conditions on some strange world, such “situation awareness” type queries would be very interesting to bikers, joggers, walkers, city workers, and all concerned parents and citizens alike on this world who want to know, for example, the health of their immediate environment at any point in time.

Speed and accuracy are two very important requirements of any mobile LBS application. Among the three search algorithms tested, *Search Algorithm 3* shows the most accuracy while *Search Algorithm 2* has the fastest speed. Further study into the trade-offs between speed vs. accuracy are underway to find the most suitable approach for deploying 3DQ in a real-world mobile eCampus application for real student/staff users.

Our ultimate goal is to make 3DQ work in real-time, or near real-time, where the display of the visibility dome shape changes dynamically on top of the 3D map display as a user walks around their city. Currently, the Google Maps 5 visualisation API is not yet publically available for this; however, there are alternatives that can be used for testing. For instance, VisioDevKit [21] provides 3D rendering capability such that any 3DQ visibility shape can be overlayed.

To achieve our real-time threat dome calculation/visualisation goal, improvements to the 3 search algorithms will also be further investigated. As our prototype currently adopts a “client-server” architecture, performance is mainly constrained to the latency of mobile networks. As mobile devices become more powerful as computing platforms, together with some open source mobile spatial database options, implementations of Search Algorithms 2 and 3 entirely on the mobile device will be investigated.

Acknowledgements. Research presented in this paper was funded by a Strategic Research Cluster Grant (07/SRC/I1168) by Science Foundation Ireland under the National Development Plan. The authors gratefully acknowledge this support.

References

1. Google Maps 5 (2011), `http://googlemobile.blogspot.com/2010/12/next-generation-of-mobile-maps.html` (retrieved) (accessed October 15, 2011)
2. Di Martino, S., Ferrucci, F., McArdle, G., Petillo, G.: Automatic Generation of an Adaptive WebGIS. In: Carswell, J.D., Fotheringham, A.S., McArdle, G. (eds.) W2GIS 2009. LNCS, vol. 5886, pp. 171–186. Springer, Heidelberg (2009)
3. Ceri, S., Fraternali, P., Bongio, A., Brambilla, M., Comai, S., Matera, M.: Designing Data-Intensive Web Applications. The Morgan Kaufmann Series in Data Management Systems. Morgan-Kaufmann Publishers, San Francisco (2002) ISBN-13: 978-1558608436
4. Aoidh, E.M., Wilson, D.C., Bertolotto, M.: A Study of Spatial Interaction Behaviour for Improved Delivery of Web-Based Maps. In: Carswell, J.D., Fotheringham, A.S., McArdle, G. (eds.) W2GIS 2009. LNCS, vol. 5886, pp. 120–134. Springer, Heidelberg (2009)

5. Mountain, D.M.: Spatial Filters for Mobile Information Retrieval. In: 4th ACM Workshop on Geographical Information Retrieval (GIR 2007), pp. 61–62. ACM Press, Lisbon (2007)
6. Ball, M.: How do crowd sourcing, the Internet of Things and Big Data converge on geospatial technology? V1 Magazine 5(41) (2011), `http://www.vector1media.com/-dialog/perspectives/23362-how-do-crowdsourcing-the-internet-of-things-and-big-data-converge-on-geospatial-technology.html` (accessed October 11, 2011)
7. Benedikt, M.L.: To take hold of space: isovists and isovist fields. Environment and Planning B 6, 47–65 (1979)
8. Fisher-Gewirtzman, D., Wagner, I.A.: Spatial openness as a practical metric for evaluating built-up environments. Environment and Planning B: Planning and Design 30(1), 37–49 (2003)
9. Fisher-Gewirtzman, D., Burt, M., Tzamir, Y.: A 3-D visual method for comparative evaluation of dense built-up environments. Environment and Planning B: Planning and Design 30(4), 575–587 (2003)
10. Morello, E., Carlo, R.: A digital image of the city: 3D isovists in Lynch's urban analysis. Environment and Planning B: Planning and Design 36(5), 837–853 (2009)
11. Bartie, P., Mills, S., Kingham, S.: An Egocentric Urban Viewshed: A Method for Landmark Visibility Mapping for Pedestrian Location Based Services. In: Moore, A., Drecki, I. (eds.) Geospatial Vision – New Dimensions in Cartography, New Zealand, pp. 61–85. Springer, Heidelberg (2008)
12. Bartie, P., Reitsma, F., Kingham, S., Mills, S.: Advanced visibility modelling algorithms for urban environments. Computers, Environment and Urban Systems 34, 518–531 (2010)
13. SkylineGlobe, TerraExplorer viewer for 3D earth (2011), `http://www.-skylinesoft.com/` (accessed October 11, 2011)
14. Ericson, C.: Real-time collision detection (2005) ISBN 1-55860-732-3
15. Bergen, G.: Collision detection in Interactive 3D environment (2004) ISBN: 1-555860-801-X
16. Watt, A., Policarpo, F.: 3D Games: Real-time rendering and Software Technology (2011) ISBN: 0201-61921-0
17. Carswell, J.D.: 3DQ: Threat Dome Visibility Querying on Mobile Devices. GIM International 24(8) (August 2010)
18. Obermeyer, K.J.: The VisiLibity library, (2008), `http://www.VisiLibity.org` (accessed October 11, 2011)
19. Ravada, S., Kazar, B.M., Kothuri, R.: Query processing in 3D spatial databases: Experience with Oracle Spatial 11g. 3D Geo-Information Sciences, 153–173 (2009), doi:10.1007/978-3-540-87395-2
20. 3D isovist. An Grasshopper extension for calculating 3D isovist (2011), `http://parametricmodel.com/3DIsovist/32.html` (accessed October 11, 2011)
21. VisioDevKit, A real-time 2D/3D rendering engine for mobile devices (2011), `http://www.nn4d.com/site/global/developer_resources/apis_sdks` (accessed October 11, 2011)

ARCAMA-3D – A Context-Aware Augmented Reality Mobile Platform for Environmental Discovery

Betül Aydın[1], Jérôme Gensel[1], Sylvie Calabretto[2], and Bruno Tellez[2]

[1] Laboratoire d'Informatique de Grenoble (LIG), STEAMER Team, 681, Rue de la Passerelle 38402 Saint Martin d'Hères, France
[2] INSA de Lyon, LIRIS, UMR5205, 20, Avenue Albert Einstein, 69621, Villeurbanne Cedex, France
{betul.aydin,jerome.gensel}@imag.fr,
{sylvie.calabretto,bruno.tellez}@liris.cnrs.fr

Abstract. In this paper, we present ARCAMA-3D, a platform for 3D map-based visualization on mobile devices powered by augmented reality. The platform offers context-aware interactions related to the concept of ubiquitous computing. The general purpose of the project is to enable users to navigate in an area with their mobile devices and interactively discover their surroundings. The system integrates real-time sensing technologies (GPS and other embedded sensors) and exploits user's context and preferences in order to provide her with the necessary information. In return, the user consults the information (text, photo, audio or video files, etc.) that is published on the 3D model with the help of augmented reality. The innovative aspect of our approach lies in a light-weight 3D visualization system which is superimposed on the real scene. This approach facilitates the discovery of surroundings without preventing the visualization of real entities. We also alleviate the cognitive load of the user by avoiding the presentation of excessive information.

Keywords: 3D visualization, augmented reality, cognitive load, location-based services, mobile devices, ubiquitous computing, user interaction.

1 Introduction

Recent parallel improvements in two research areas, namely 3D representation of geographic data and mobile computing devices, have contributed to the creation of new research domains. Considering the mobile device capabilities, the 3D graphics, bandwidth and memory, battery lifetime and processor speed have been upgraded tremendously during the last decade. These developments have extended the potentials of digital maps and carried them to mobile device platforms. Real-time monitoring of geographic data has embedded the use of location-based services [1] and context-awareness [3] within mobile GIS (Geographic Information Systems). Moreover, it was not so later that the use of realistic 3D models on mobile devices has emerged [12].

The improvements in mobile graphics and processing speed also encouraged the use of augmented reality, since the urban and landscape geometry can be exploited to superimpose the virtual objects on the real world view. Some of these applications concern displaying information tags aligned with the physical background, or projecting the 3D historical view of a building [14,19]. The vision-based feature tracking techniques and mobile sensing technology (accelerometer, gyroscope, etc.) are used to maintain the accuracy of the superimposition [9,18].

Mobile 3D graphics applications suffer from severe limitations, such as low computational power, limited screen space, memory, battery and user interface limitations. Therefore, storage of large data, sending large documents, or providing high frame rate videos is difficult on mobile devices. In order to develop a smooth 3D navigation application on a mobile device, these limitations have to be considered. While the current technology allows reasonable use of 3D graphics for the discovery of the environment on mobile devices, there are no standardized solutions that correspond to the current limitations. The research projects conducted for this purpose are mostly achieved by adding on-board sensors and PC-equipped platforms to the mobile devices, in order to get processing power and accurate sensor data. Also, in some projects, the 3D rendered models are too heavy to use on mobile platforms for the real-time discovery of the environment.

The purpose of our research is to develop a 3D mobile platform that enables users to discover their surroundings and access information that is stored in location-based databases. The overall system, called ARCAMA-3D (Augmented Reality for Context Aware Mobile Applications with 3D), includes a platform for interactive information exchange using location-based services. It contributes to the design aspect of mobile 3D map visualization by providing perceptually optimized solutions.

The paper is organized as follows. In Section 2, we describe the related works concerning the 3D navigation maps on mobile devices, the current methods and the limitations. Section 3 describes our solution, ARCAMA-3D system, with a scenario and a typical sequence of interaction with the system. In Section 4, we present concluding remarks and future works.

2 Related Work

Unlike 2D maps, 3D mobile GIS provide an easily recognizable environment, especially for unfamiliar users, with a realistic representation of the surroundings. While, 2D maps require skills and experience from the user, such as experience with the maps, knowledge about the environment, signs, map features, understanding the scale and directions, etc. [5], 3D maps can be easily read and understood by taking advantage of the visual similarity with the environment.

Providing navigational support with 3D maps has led to the emergence of car navigation applications. The degree of freedom of pedestrians is not limited to roads; therefore the maps had to be re-modeled for pedestrian navigation [2]. This requires modeling additional objects such as buildings, parks or squares, rather than only roads and hills. Since then, several mobile map-based approaches are proposed for different contexts, such as for the exploration of cities, museums, exhibitions, mountain tracks, and sea routes [4].

The design of a 3D map is an interdisciplinary work. The modeling, symbolization and visualization of the map should be considered during the design process [8]. Even though there are widely accepted definitions for 2D maps, it is not the case for 3D maps. Shortcomings are mainly due to the ignorance of user needs, the use of different map symbolization, frequent overloaded representations of the scene or insufficient level of details for object interpretation.

3D map visualization has also encouraged the use of augmented reality. Augmented reality consists in superimposing a virtual object on a video of a real scene, possibly offering some interaction with the user. The urban and landscape objects are exploited to align the virtual objects within the scene. This enables the virtual objects to be projected on the scene at the right place. In several mobile applications, augmented reality representations are used to provide additional information to the user, or to augment the view of a site that is not possible to observe with bare eyes (the views of historical sites in previous eras, underground water and electricity lines, etc.) [11,18]. In mobile augmented reality systems, the superimposition of the virtual object with the scene depends on the use of an accurate camera tracking algorithm. Camera position and direction tracking are achieved via vision-based tracking (tracking the visual features of the scene) [7,17]; or using a combination of GPS receiver and inertial sensors (accelerometer, gyroscope, etc.) [16,17]; or using a fusion of these two techniques (i.e. hybrid tracking) [10,18].

However, in the literature, to our knowledge, the 3D maps are rarely used as an interactive augmented reality tool that users can interact with, in order to get more information about their surroundings. Instead, augmented reality is simply used for adding some virtual objects superimposed on the real object in order to present extra information. Some augmented reality applications are developed with this idea; for example, showing the historical representation of a building by superimposing its old appearance on the video stream [11,14,15,19]. However, there are few researchers that mentioned the use of interaction mechanisms with the augmented 3D maps.

Touring Machine is an early research project that uses augmented reality systems for outdoor environment discovery [6]. It mentioned the interaction with the 3D mobile augmented reality system, and posing queries about objects. However, the hardware consists of a backpack computer with high-performance 3D graphics and additional handheld computer. In order to apply the same application on a mobile device, several limitations should be considered and overcome, since computational resources of a mobile device are too limited. In applications, where extra on-board sensors (such as GPS antennas, mounted accelerometer and gyroscope sensors, or wider touch screens for interaction, etc.) are added to the hardware, as in [11,18,19], the possibility execution of such an application on mobile device platforms should be reconsidered due to these hardware limitations.

3 The ARCAMA-3D System

In the related work presented in Section 2, most of the methods are based on adding high accurate sensors to mobile platforms, since the location and orientation based mobile augmented reality applications may be erroneous due to the inaccuracy of sensor data. However, we hold on the idea that, since a 3D location-based map resembles to the real scene, the user can relate the 3D objects with the real scene.

Therefore, when a 3D object superimposed to the real scene is given to the user, she is allowed to correct the misalignment occurred due to the erroneous sensor data. User accomplishes this by simply interacting with the virtual 3D object using the touch screen of the mobile. She drags the virtual object and locks the alignment when the 3D object superimposed on the real scene at the right place.

Accordingly, we use a 3D model as an augmented reality tool to better understand the environment, and to facilitate the interaction with the user. We also exploit a 3D model (by highlighting the objects) to indicate the user that a particular nearby object might be of interest to her. If the user decides to get more information about that highlighted object, she interacts with it. Then, the information is filtered with the user's context and displayed on the 3D model.

Figure 1 is a representation of our system, where a user points her mobile phone towards a landmark, Notre Dame Cathedral in Paris. The user, at this point, observes a light-weight 3D model of the Cathedral aligned with the real scene. Then, she interacts with the 3D model in order to get more information about the Cathedral. This information will be filtered according to the context of the user and displayed on the 3D model using augmented reality.

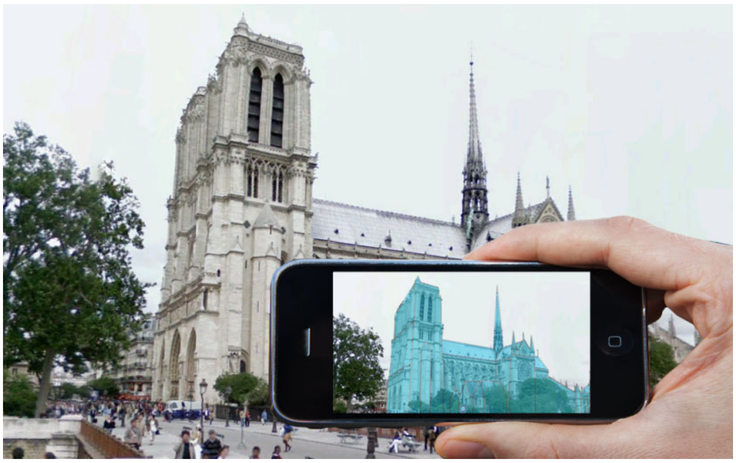


Fig. 1. A representation of ARCAMA-3D system

The primary objectives of ARCAMA-3D system are:

1) Advocating for the design of a perceptually optimized 3D model which can be used for various usage scenarios,
2) Proposing a context-aware client-server architecture for real-time augmented reality visualization of location-based data on a 3D mobile map,
3) Developing a solution for filtering information according to the context of the user.

To reach these objectives, our method proposes a light-weight model that provides an adequate level of detail without overloading the user's cognitive abilities and the computational resources of the device. Also, we offer a framework that adapts to the direction pointed by the user with her mobile device, which encourages her to focus on the environment and discover her surroundings using augmented reality.

3.1 Scenario

In order to explain our approach, let us consider a simple scenario of a mobile user, called Anna. Anna visits to a city where she has never been before. She turns on the ARCAMA-3D application. She prefers to login using her social network profile (Facebook, Twitter, etc.). The application gathers her user's profile information: age, gender, schools where she made her studies, her topics of interest etc. She can also modify her profile data and select some topics of interest using the dropdown menu of ARCAMA-3D. Then, she turns off the mobile screen and puts the device on her pocket.

According to the context of Anna (preferences and profile), it appears that she is a history student and interested in movies. The application uses the GPS module of her mobile phone in order to detect her coordinates. When she walks around in the city, the mobile device beeps to draw her attention in order to inform her that a movie store or a cinema is nearby. She, then, turns on the screen of the device and a 3D model is sent to her. This 3D model is composed of transparent white buildings of her surroundings, with no textures and no architectural details. By using the gyroscope and accelerometer sensors of her mobile device, her orientation and direction is extracted by the ARCAMA-3D application. Then, the 3D model is superimposed on the real view of her surroundings. However, since sensors are not precise enough, the superimposition is somewhat misaligned. Then, Anna is invited by the application to correct it by dragging the 3D model on the real view, so that the 3D objects are superimposed correctly on their real counterparts. She, then, locks the 3D model and uses the 3D objects as an interactive augmented reality tool.

While most of the objects are white, some of the 3D objects are highlighted with a different color. This indicates that these 3D objects have selected by the applications as Surrounding Objects of Interest (SOIs) for Anna, which indicates that they have interesting information according to her context. Anna can interact with these objects by using the touch screen, to access the information. She clicks on the 3D objects and related information appears on the 3D model (such as a photo, a video, a wiki page, etc.). According to her context, these information might be about a cinema that she is passing by, or a store selling movies. She can check the movies on the program, or the movies that are being sold in the store, or the opening and closing hours of the store. These information is displayed attached to the 3D object.

3.2 Architecture

As can be seen in Figure 2, the ARCAMA-3D architecture is a 3-tier architecture with a middleware between the client and two databases (the 3D Map Database and the Information Database). Data is controlled by a database management system which facilitates the creation, organization, storage, management and retrieval of the data.

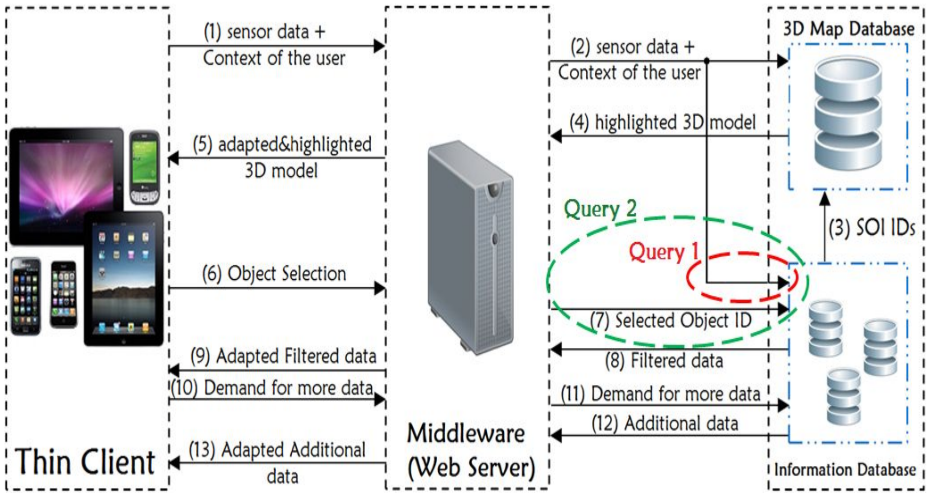


Fig. 2. ARCAMA-3D Architecture

In Figure 2 above, the Thin Client represents the mobile devices (smartphones, personal digital assistants (PDAs), tablet computers, etc.). A Graphical User Interface is deployed on the Client and data are provided by the servers. The reason of using a client-server approach is due to the limitations of mobile devices, such as computational power and storage capacity. Mobile devices have also limited bandwidths. Therefore, data provided by the servers should be kept simple for the capacity of the mobile device and enough to meet the user's needs.

The Web Server serves as a Middleware between the Databases and the Client. Since the mobile devices have various different characteristics (different operating systems, screen sizes, processing power, connection speed, etc.) and the data formats can be diverse (photos, texts, video, audio files, etc.), the Web Server acts as an adaptation layer. The data is formatted according to the characteristics of the mobile.

The user is provided with *information* about *SOIs* (Surrounding Object of Interest), which corresponds to the first ranked information for each data type. SOIs are determined according to the context (the preferences and the profile information) of the user. In addition to the location data, the context of the user is used to filter the data in the Information Database. We enable user to declare her preferences by choosing the preference options from the menu of the application. Currently, the context of the user is composed of these preferences. This context information is used to filter the data. The query (Query 2 in Figure 2) sent to the Information Database consists of the Location data, the Context of the user (her preferences) and the Selected Object ID. The Client, after consulting the highest ranked data, can demand more data (the subsequent data) by interacting with the object.

A Typical Sequence Diagram Interaction with ARCAMA-3D. In this part, we will detail the Figure 2.

(1) *Sensor data + Context of the user*: The application starts with the initialization, which corresponds to acquisition of both the sensor data and the context of the user. Sensor data are obtained from the embedded sensor modules of the mobile device. The context of the user is interactively learnt from the user by asking her preferences and her profile.

(2) *Sensor data*: By sensor data, we refer to the accelerometer, gyroscope sensors and GPS module of the mobile device (as well as the time, which will be used for Information Database). These data will give the direction, orientation and the geo-location of the user. They are necessary to get the corresponding 3D model view of the real scene that the user points her mobile. Also, they are used to update this view according to the user's location and orientation, when she moves around. Therefore, sensor data are sent to the 3D Map Database and acquired constantly throughout the application while the user is moving around. It should be noted that, there might be errors in the sensor data. Therefore the 3D model sent to the mobile phone corresponds to a wider area of the user's surroundings.

Location data + Context of the user: At the same time, the location data and the Context of the user are sent to the Information Database for filtering information according to the context. The user is basically asking a question: here are my preferences, and here is my location; is there any SOI around? To answer this question, the Information Database filters the information with the location data first, and then it filters the result with the context of the user (Query 1 in Figure 2). Each of the data (document) belongs to at least one object. Therefore, each of them has 3D Object IDs in their structure.

(3) SOI IDs: Then, these 3D Object IDs (SOI IDs) acquired from the Information Database are sent to the 3D Map Database. These 3D objects are highlighted (with a different color) on the 3D model to indicate that the user may gather useful information if she interacts with these objects.

(4) Highlighted 3D models: Highlighted 3D model is sent to the Web Server.

(5) Adapted&Highlighted model: The Web Server sends the model to the user considering the device capabilities.

(6) Object Selection: Now, the user can see the 3D model of her surroundings adapted to the view that she points her mobile at. She can interact with the highlighted objects by clicking them through the touch screen. With this action, the ID of the object is sent to the Web Server for further information retrieval.

(7) Selected Object ID: The ID of the selected 3D object is sent to the Information Database, which still keeps the filtered documents according to the location and context of the user, as a result of Query 1 in step (2) above. Now, these documents will be filtered with the Selected Object ID, which constitutes the Query 2.

(8) Filtered Data: Information Database sends back the filtered information, which belongs to the selected object, to the Web Server. Filtered data corresponds to the highest ranked data for each data type (photo, video, audio, text, etc.).

(9) Adapted Filtered Data: Such data is adapted to the mobile phone characteristics via Web Server as in step (5) above.
(10) Demand for more data: User, who can now observe the highest ranked data, may demand more data. For example, after looking to the highest ranked video about Eiffel Tower, she can ask for the next highest ranked video by interacting with the 3D model. This interaction, as done in object selection in step (6), is achieved by interacting with the touch screen, as well. Then, this demand is sent to the Web Server.
(11) Demand for more data: Web Server transmits this demand to the Information Database, which holds the next ranked data.
(12) Additional data: Additional data is retrieved and sent back to the Web Server.
(13) Additional data: Web Server adapts the data to the mobile device characteristics and these data are presented on the 3D object as in step (9).

Information Database. All the information that is displayed on the 3D model is held on the location-based services with object coordinates (x, y, z). Time and orientation (t, α) information might be useful for a photo or a video, since the user might want to retrieve some photos of a monument taken from her current point of view and at the same time of the year or the day (winter, afternoon, etc.). Therefore, the information database holds documents annotated with the (x, y, z, t, α) information, and as well as the semantic tags. These semantic tags are provided by the owner of the application. If the owner allows users to add documents to the Information Database, these tags can be also provided by the owner of document. During the information retrieval, if these tags match with the user's preferences, these documents are filtered and sent to the user, since it corresponds to her interests.

3D Map Database. The 3D model is held on a database. The 3D query is sent by the client through the web server. This query includes the sensor data (geo-location, orientation and direction of movement) that is extracted using the embedded sensors of the mobile device. Then, the 3D Map Database returns the corresponding 3D data set (model) of user's location (covering some 100 meters of range). The sensor data will be used to update user's 3D view, to correspond to the view that she points her mobile device. For this reason, the interchange of sensor data and 3D data set between the client and the 3D Map Database will take place throughout the application.

In this interchange, sensor data are used to detect the orientation and location of the user. However, the sensor data may not always be very accurate. For example, nowadays, outdoor positioning mainly relies on the satellite infrastructure of the GPS, which is known to have an accuracy of 10-50 meters [13]. This precision can be improved with the GSM (Global System for Mobile Communications), which is based on triangulation of cellular networks. Considering the developments in the mobile technology during the last decade, the precision of the sensor data will be improved in the following years, therefore, we assume that it provides a certain level of accuracy for our application.

For this reason, we have included an interaction mechanism in order to overcome possible problems. The interaction mechanism relies on the user interaction. In ARCAMA-3D application, the environment is discovered through a 3D model that is superimposed on the real scene that the user points her mobile device at. As we explained in the scenario (Section 3.1), if this superimposition is not correct, the user can interact with the 3D model and superpose the 3D model on the real view using the touch screen of the device.

4 Conclusion and Future Work

In this work, we have described ARCAMA-3D, a 3D map-based visualization platform for the environmental discovery on mobile devices. ARCAMA-3D contributes to the design aspects of mobile 3D maps by providing perceptually optimized solutions for the user. It also provides a light-weight 3D model considering the limitations of the mobile devices.

Mobile navigation systems are sensitive applications considering the shortcomings of mobile devices and also the usage scenarios in which they are used. Users would generally consult these applications when they are navigating in an unknown environment. Therefore, they do not have enough time to interpret the objects or focus all their attention to the user interface of the application. The main objective of these applications should be enabling the users to focus on the environment as much as possible, instead of keeping them busy looking at the mobile screen. ARCAMA-3D proposes an environmental discovery architecture using an interaction mechanism with an augmented 3D map, which avoids surcharging the cognitive load of the user.

We are currently improving our approach on time-based filtering of the information and presenting techniques on the 3D model. Also, we are working on the user profile management which will help to filter the corresponding information from the Information Database.

References

1. Baus, J., Krüger, A., Wahlster, W.: A resource-adaptive mobile navigation system. In: Proceedings of the 7th International Conference on Intelligent User Interfaces, San Francisco, California, USA, pp. 15–22 (2002)
2. Bogdahn, J., Coors, V.: Using 3D Urban Models for Pedestrian Navigation Support. In: GeoWeb 2010 (2010)
3. Cheverst, K., Davies, N., Mitchell, K., Friday, A., Efstratiou, C.: Developing a Context-aware Electronic Tourist Guide: Some Issues and Experiences. In: Proc. of the 2000 Conf. on Human Factors in Computing Systems (CHI 2000), pp. 17–24. ACM Press, New York (2000)
4. Chittaro, L.: Visualizing Information on Mobile Devices. IEEE Computer 39(3), 34–39 (2006)
5. Elias, B., Hampe, M., Sester, M.: Adaptive Visualisation of Landmarks Using an MRDB. In: Map-based Mobile Services - Theories, Methods and Implementations, pp. 73–86. Springer, Heidelberg (2005)

6. Feiner, S., MacIntyre, B., Höllerer, T., Webster, T.: A touring machine: Prototyping 3D mobile augmented reality systems for exploring the urban environment. In: Proc. ISWC 1997 1st IEEE International Symposium on Wearable Computers, pp. 208–217 (1997)
7. Fua, P., Lepetit, V.: Vision based 3D tracking and pose estimation for mixed reality. In: Haller, M., Billinghurst, M., Thomas, B.H. (eds.) Emerging Technologies of Augmented Reality Interfaces and Design, pp. 43–63. Idea Group, Hershey (2007)
8. Haeberling, C.: Cartographic Design Principles for 3D Maps - A Contribution to Cartographic Theory. In: 22nd International Cartographic Conference, Caruña, Spain, July 9-16 (2005)
9. Honkamaa, P., Siltanen, S., Jäppinen, J., Woodward, C., Korkalo, O.: Interactive outdoor mobile augmentation using markerless tracking and GPS. In: Proc. Virtual Reality International Conference (VRIC), Laval, France, pp. 285–288 (April 2007)
10. Kim, S., DiVerdi, S., Chang, J.S., Kang, T., Iltis, R.A., Höllerer, T.: Implicit 3D modeling and tracking for anywhere augmentation. In: ACM Symposium on Virtual Reality Software and Technology (VRST 2007), pp. 19–28 (2007)
11. King, G.R., Piekarski, W., Thomas, B.H.: ARVino - Outdoor Augmented Reality visualization of viticulture GIS data. In: Proceedings of the 4th IEEE/ACM International Symposium on Mixed and Augmented Reality, ISMAR 2005 (2005)
12. Kray, C., Elting, C., Laakso, K., Coors, V.: Presenting Route Instructions on Mobile Devices. In: IUI 2003, pp. 117–124 (2003)
13. Krüger, A., Baus, J., Heckmann, D., Kruppa, M., Wasinger, R.: Adaptive Mobile Guides. In: Brusilovsky, P., Kobsa, A., Nejdl, W. (eds.) Adaptive Web 2007. LNCS, vol. 4321, pp. 521–549. Springer, Heidelberg (2007)
14. Noh, Z., Sunar, M.S., Pan, Z.: A Review on Augmented Reality for Virtual Heritage System. In: Chang, M., Kuo, R., Kinshuk, Chen, G.-D., Hirose, M. (eds.) Edutainment 2009. LNCS, vol. 5670, pp. 50–61. Springer, Heidelberg (2009)
15. Nurminen, A., Kruijff, E., Veas, E.: HYDROSYS – A mixed reality platform for on-site visualization of environmental data. In: Tanaka, K., Fröhlich, P., Kim, K.-S. (eds.) W2GIS 2011. LNCS, vol. 6574, pp. 159–175. Springer, Heidelberg (2010)
16. Piekarski, W., Thomas, B.H.: An Object-Oriented Software Architecture for 3D Mixed Reality Applications. In: Proceedings of the 2nd IEEE/ACM International Symposium on Mixed and Augmented Reality, ISMAR 2003 (2003)
17. Reitmayr, G., Drummond, T.W.: Going out: Robust model-based tracking for outdoor augmented reality. In: Proceedings of 5th IEEE/ACM International Symposium on Mixed and Augmented Reality (ISMAR 2006), pp. 109–118 (2006)
18. Schall, G., Wagner, D., Reitmayr, G., Taichmann, E., Wieser, M., Schmalstieg, D., Hofmann-Wellenhof, B.: Global Pose Estimation using Multi-Sensor Fusion for Outdoor Augmented Reality. In: Proceedings of the 8th IEEE/ACM International Symposium on Mixed and Augmented Reality (ISMAR 2009), pp. 153–162 (2009)
19. Schnädelbach, H., Koleva, B., Flintham, M., Fraser, M., Izadi, S., Foster, M., Benford, S., Greenhalgh, C., Rodden, T.: The Augurscope: A Mixed Reality Interface for Outdoors. In: ACM Conference on Computer-Human Interaction (CHI 2002), pp. 9–16 (2002)

Speech and Multimodal Interaction in Mobile GIS Search: A Case of Study

Francesco Cutugno, Vincenza Anna Leano, Gianluca Mignini, and Roberto Rinaldi

University of Naples "Federico II", Complesso Universitario Monte Sant'Angelo, Via Cinthia 80100 Naples, Italy
{cutugno,vincenzaanna.leano}@unina.it,
g.mignini@cedat85.com,
rober.rinaldi@studenti.unina.it

Abstract. The In this short paper we will present an Android mobile application making use of a multimodal interface. We base our application on a proprietary architecture based on the W3C recommendation MM-Framework. App-users can talk and use natural gestures on a mobile device touch screen in order to formulate a complex interrogation to a geo-referenced web service. The interaction produce as result a query whose origin is a semantically incomplete audio sentence complete by a deictic gesture (i.e.: "please, find all bus stops in this area - - while tapping or making a circle on the screen where a map is showed - -").

Keywords: mobile GIS, multimodal mobile interaction, mobilepublic transport GIS.

1 Case Study: GIS Multimodal Platform for Neapolitan Public Transportation

The case study here chosen to show the advantage of the utilization of a multi-modal approach to mobile user interface for GIS system queries, is the creation of an informative map-based system for Neapolitan public transportation (henceforth ANM). The system provides the user with the ability to make multi-modal queries to obtain information regarding public transportation, such as:

- seeing a map that shows all of the bus stops made by a chosen line of transportation
- seeing the arrival times of transportation lines at a chosen bus stop
- seeing additional information relative to a bus stops such as a list of transportation lines that transit at the chosen stop and all of the temporary detours on the line
- seeing a map with the nearest bus stops to the current location
- seeing a map with the bus stops on a given street or area of interest.

2 GIS System Architecture

Getting results from spacial queries implies the elaboration of a great quantity of data. Taking into consideration the physical limitations and the computational power of modern mobile devices, it's not possible to implement these operations on the device itself, and therefore the chosen architecture for the service is of the client-server type, so that the calculations (spacial queries) can be made on the server, limiting the mobile device to sending requests and processing the replies so that they can be displayed on the map. In order to handle the great quantities of geographic data on the server, a Spacial Database is absolutely necessary in order to index the data and allow queries to be made on it (for example coverage or intersection between two geometries). Multi-line geometries are present in the database in order to represent the roads in the area of interest and points to represent the bus stops. As far as the functions implemented by the server, only one spacial operator is provided: coverage. Specifically the operator handles the shape drawn by the user and returns the objects present within.

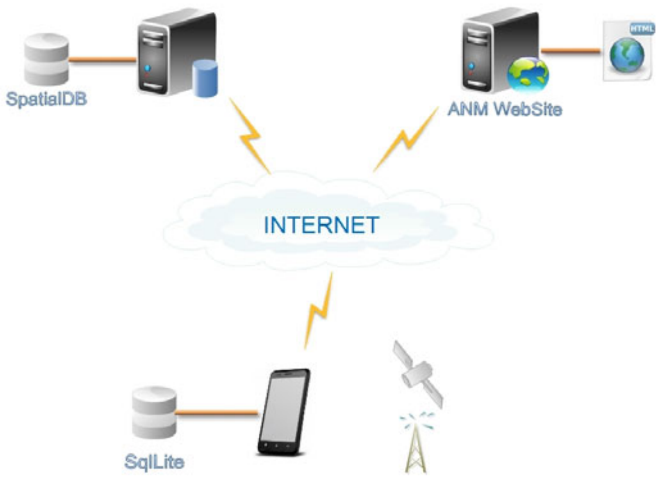


Fig. 1. The GIS system architecture. A mobile database (SqlLite), a spatial database (GeoServer) and a web-service.

A database like SqlLite is present in the mobile application and contains all of the information relative to the spatial position of the bus stops, the active public transportation lines, possible detours, user preferences etc... The information regarding weather predictions for the following 30 minutes at a given bus stop is retrieved from the public transportation company website by making an http post request through a web-service offered by ANM and parsing the html response received.

3 Architecture for the Management of the Multi-modal Interface

In order to handle the system's multi-modal interface (in the proposed case of voice and gestures on a map), an approach that conforms with W3c Multi-modal Framework [1] was adopted and the proprieties of the relationship between different modes of interaction specified in [2] we considered. A framework was created that allows a detailed description of the multi-modal interaction between the user and the GIS system using an XML file. In order to model the multi-modal commands according to the CARE proprieties an approach that uses NFA (Non-deterministic Finite State Automa) was chosen, just as it was specified in [3], where they were utilized to implement a toolkit to evaluate the usability of multi-modal interfaces.

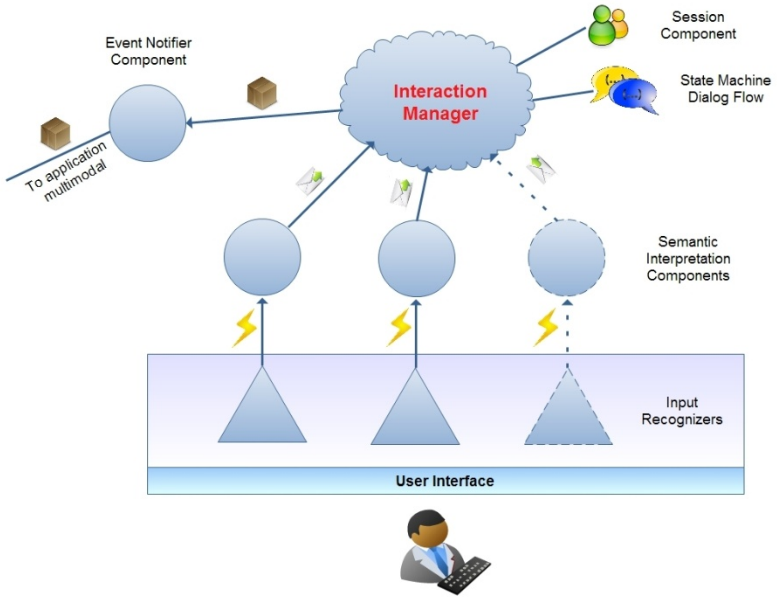


Fig. 2. Multimodal Interface Architecture

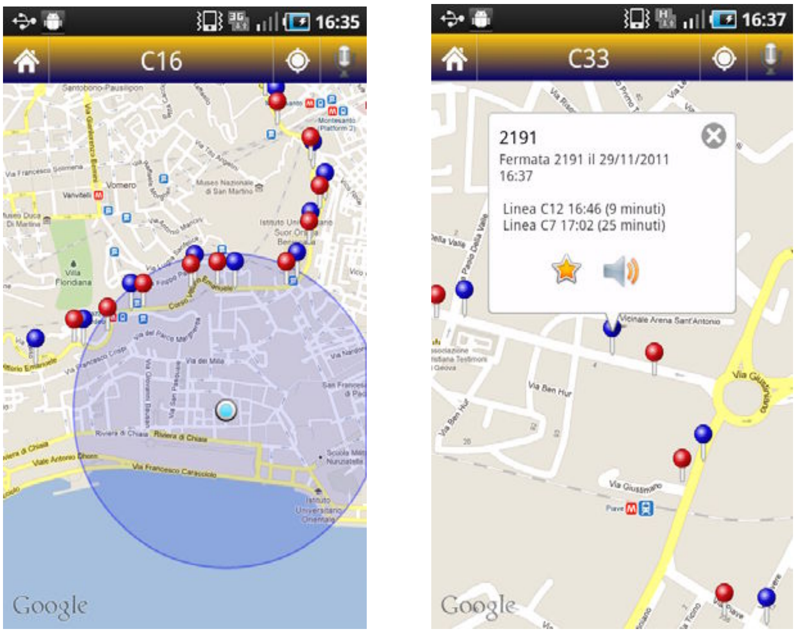
The architecture for the multi-modal interface management of the examined case study is in figure 2. The architecture provides a communication between the various interfaces modes and the Interaction Manager based on asynchronous events. Every mode of interaction is composed of two subsections:

- *Input Recognizer*: which focuses its attention on a single mode source of information like voice and gesture on touch-screen. Therefore it captures the input from the user and transforms it in a machine-readable form.
- *Semantic Interpretation Component:* which manages and keeps constantly observes its relative Input Recognizer in order to verify its progress and performs a semantic extraction of what it receives. Its task is also to forward the extracted data to the Interaction Manager.

The events sent by the Semantic Interpretation Component are received by the *Interaction Manager* that, thanks to an XML file which describes how the interaction needs to happen [4], notifies the Application Logic of the mobile GIS client of the behavior to carry out with the data it needs. All of this happens with the help of the *Event Notifier* Component.

4 Case Study in Action on Android Platform

In this section we describe how the prototype of the proposed study was implemented on Android platform. A Google Maps [5] integrated service was used in an Android environment and the touch screen of the smartphone was used to specify spatial queries.



2191
Fermata 2191 il 29/11/2011
16:37

Linea C12 16:46 (9 minuti)
Linea C7 17:02 (25 minuti)

Fig. 3. (a) the blue circle represents the user's current location. The markers represent stops bus of the route C16 – Fig. 3(b) balloon with information about the expected arrival of public transports to the bus stop 2191.

The application which allows the following operations:

- 'Search by line': allows the user to see all of the bus stops of a public transportation line on a map.
- 'Search by bus stop number': allows the user to know the arrival time predictions of the various bus lines by specifying the number of the bus stop of interest
- 'Favorite stops': allows the user to see predicted arrival times for his favorite bus stops which were previously specified
- 'Browse stops': allows a user to visualize a list of the routes and bus stops of the carious public transportation lines.
- 'Search by street name': allows the user to visualize on a map the bus stops on a specified street.

- 'Settings': allows the user to configure some of the application's settings like the default transportation line to be displayed on the map
- 'Vocal commands': by clicking on the microphone icon present on the right of the title bar, it's possible to interact with the application through vocal commands.

Let us go into the details of the functionality of 'Search by line'. As shown in figure 3a-3b, the user sees a local map relative to his current position with markers which represent the stops made by various lines of the service.

Once the stops of interest are displayed on the map, the user can get various information for a bus stop simply by tapping on the corresponding marker and a balloon can be recalled containing expected arrival times.

4.1 Vocal and Multi-modal Commands in the Application

The mobile application allows the fruition of some content by vocal and multimodal command. A vocal interaction is started by a click on the microphone icon.

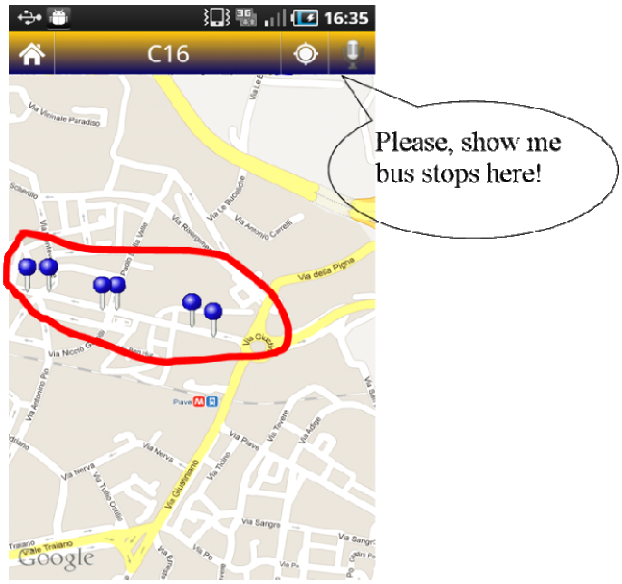


Fig. 4. Multimodal Interaction - the red line is drawn by the user; blue dots are produced as output of the multimodal query. The sentence in the comics represents the speech action said by the user while drawing the red poly-line on the map.

There are three voice-activated functions:

1. show a transportation line on the map
2. load the arrival time predictions for a chosen stop
3. know the wait time of a chosen line at a chosen stop.

User can perform also a multi-modal query on the map drawing a location on the map and uttering the action to be done in that location by voice. The implemented drawing functions support point and poly-lines. To be more specific: a point is when the user touches a specific object (e.g. the bus stop) on the mobile map; a poly-line is when the user swipes the screen over the map. The implementation of these query interfaces is relatively straight forward. We attach two layers to the touch screen: one at the bottom is the map layer and one at the top is an empty layer, which is used to capture any touch gestures from the user. The top layer is the most important as it converts the touch points into a latitude/longitude coordinates set transmitted to the underlying base map.

Audio is processed by the speech platform which first performs speech recognition and then extract some semantic knowledge to understand which action to do, the gesture is recognized by the specific recognizer and performs a geo-coding of the touch points on the map. These data then is processed by the Multimodal Framework (cfr. 3).

The system sends the latitude/longitude on the line drawn on the maps to the GIS server and then return the bus stops closest to the route drawn.

References

1. Larson, J.A., Raman, T.V., Raggett, D., Bodell, M., Johnston, M., Kumar, S., Potter, S., Waters, K.: W3C multimodal interaction framework. W3C Note (May 2003), `http://www.w3.org/TR/mmi-framework/`
2. Coutaz, J., Nigay, L., Salber, D., Blandford, A., May, J., Young, R.M.: Four easy pieces for assessing the usability of multimodal interaction: The CARE properties. In: Proceedings of INTERACT 1995, Lillehammer, pp. 115–120 (June 1995)
3. Bourguet, M.-L.: Designing and Prototyping Multimodal Commands. In: Human-Computer Interaction 2003, pp. 717–720 (2003)
4. Dumas, B., Lalanne, D., Ingold, R.: Description languages for multimodal interaction: a set of guidelines and its illustration with SMUIML. Multimodal User Interfaces 3, 237–247 (2010)
5. `http://code.google.com/intl/it/android/add-ons/google-apis/reference/index.html`

Asynchronous Ultrasonic Trilateration for Indoor Positioning of Mobile Phones

Viacheslav Filonenko, Charlie Cullen, and James D. Carswell

Digital Media Centre, Dublin Institute of Technology, Ireland
{viacheslav.filonenko,charlie.cullen,jcarswell}@dit.ie

Abstract. In this paper we discuss how the innate ability of mobile phone speakers to produce ultrasound can be used for accurate indoor positioning. The frequencies in question are in a range between 20 and 22 KHz, which is high enough to be inaudible by humans but still low enough to be generated by today's mobile phone sound hardware. Our tests indicate that it is possible to generate the given range of frequencies without significant distortions, provided the signal volume is not turned excessively high. In this paper we present and evaluate the accuracy of our asynchronous trilateration method (Lok8) for mobile positioning <u>without</u> requiring knowledge of the time the ultrasonic signal was sent. This approach shows that only the differences in time of arrival to multiple microphones (control points) placed throughout the indoor environment is sufficient. Consequently, any timing issues with client and server synchronization are avoided.

Keywords: Indoor Mobile Positioning, Ultrasonic Trilateration, LBS.

1 Introduction

The role of mobile phones in society has changed dramatically in the past few years as for many people their SmartPhone is an omnipresent gateway to information. The mobile nature of the device is of key importance. Users have come to expect constant access to the phone's information facilities in many different circumstances and environments that take into account location and personal preference when providing useful and timely decision support services.

Currently outdoor Location Based Services (LBS) have the advantage of relatively reliable positioning via GPS (also Wi-Fi and GSM) and a defined business model for the delivery of content to the user. This has led to applications of outdoor LBS greatly expanding in recent years, leaving indoor locationing technologies and services on mobile devices to yet fully mature. The current state-of-the-art of merging accurate (i.e. sub-metre) indoor positioning and context-sensitive services for indoor LBS therefore is still an open problem.

The following factors make indoor positioning challenging:

1. Generally indoor environments require higher accuracy to be useful for practical LBS purposes. This is because when indoors we are dealing with objects and distances on a smaller scale. While accuracy of +/- 10 meters may be good enough

to direct someone to a cafe or a bus stop, indoors it could mean we are unsure in which room the user currently is located.

2. Locationing services that rely on satellite signals such as GPS for positioning do not work indoors at all because these signals require a direct line-of-sight to the receiver.

3. When used indoors, electromagnetic fields and sound signals can suffer from fading and multipath propagation when they encounter walls, windows, and other structures. This requires implementations of a robust solution that can effectively overcome the positioning difficulties typically found in cluttered, complex indoor environments.

Under these circumstances it is understandable that very specialized hardware may be required unless we are willing to sacrifice accuracy. However, given the role of commercial off-the-shelf (COTS) SmartPhones in today's society, they have by default become the platform of choice for implementations of indoor positioning and therefore the standard hw platform we have developed our Lok8 (locate) indoor positioning solution to work on.

While many mobile positioning approaches are erroneously described in the press as *triangulation*, where angles between mobile devices to various receivers (control points) would be required, what is in fact being described is *trilateration*, where distances to known control points or beacons are instead used in the positioning calculation. Significantly, what is often common among these solutions is some sort of timing synchronisation requirement between transmitter and receiver to provide a full measure of distance as inputs to the trilateration process.

The main contribution of our Lok8 approach is that we remove this often delicate synchronisation problem between transmitter and receiver by instead requiring that receivers (i.e. 3 or more microphones) be connected to a central server that starts a timer once an ultrasonic signal is detected by any of the mics. By making the time the original signal was sent irrelevant, only the differences in time between when the signal reaches each of the remaining microphones is needed in our solution. The result is a more robust mobile trilateration method.

As comprehensive explanation of mobile trilateration procedure is all too rare in indoor LBS literature, another worthwhile contribution of this paper is to describe in detail our subtle but significant modification to standard least squares trilateration in Section 3. Where standard trilateration assumes that distances from an unknown position to all control points are known a-priori, instead we only know the differences between these distances - not the distances themselves. So while our asynchronous trilateration derivation is similar to the standard case, the initial conditions are different and therefore the standard trilateration solution requires modification. Before this we first discuss some background work in Section 2, and follow this with a summary of our principal contributions to the field of indoor mobile positioning and plans for future work in Section 4.

2 Indoor Positioning Background

There are many different methods and reported accuracies for locating a mobile device indoors (see Table 1). Methods that use propagation of Radio Frequency (RF) signals are prevalent in this field, with the exception of computer vision, where simultaneous localization and mapping (SLAM) appears to be the most promising but considered by many an operational technology still in its infancy [1]. Computer vision techniques, while potentially very accurate, is characterized by high computational load, complicated procedures of recovery from tracking failures and susceptibility to camera shake and motion blur. These problems are addressed in the studies done by Williams et al. [2] and Wagner et al. [3]. Another difficulty associated with computer vision is that the user is supposed to be looking through the display screen when using the device.

Table 1. Comparison of Indoor Positioning Implementations

	Best Accuracy	**Underlying Technology**	**Available on SmartPhones**
Wide Signal Strength Fingerprinting	2.48m	GSM	no
Skyhook(GSM)	200m	GSM	yes
Navizon(GSM)	50m	GSM	yes
Skyhook(Wi-Fi)	10m	Wi-Fi	yes
Navizon(Wi-Fi)	20m	Wi-Fi	yes
RADAR	2m	WaveLan	no
GP for Signal Strength-Based Location Estimation	2m	Wi-Fi	yes
Ekahau	1m	Wi-Fi	no
The Bat	3cm	Ultrasound	no
The Cricket	3cm	Ultrasound	no
Lok8	**Sub-metre**	**Ultrasound**	**yes**

RF-based transceivers such as GSM, Wi-Fi and Bluetooth can be found on every modern SmartPhone. Five meter accuracy, one of the best results for indoor GSM positioning, was displayed by Otsason et al. with the help of wide signal-strength fingerprinting [4]. Unfortunately wide signal-strength fingerprinting is impossible on many modern phones due to OS restrictions. Other GSM positioning methods are generally impractical for indoor use due to poor accuracy. Wi-Fi positioning is on average better than twice as accurate as GSM. A method proposed by Ferris et al. where Gaussian processes are used to mathematically predict signal strength in areas outside the exact spots where fingerprints were taken seems to promising [5]. The best accuracy among commercial solutions using this approach was shown by Ekahau: 1-3 meters [6].

Bluetooth has the shortest range among the three wireless technologies but there are two major problems that make Bluetooth positioning particularly difficult. First of all it is designed to adjust signal strength when signals become too strong or too weak making any subsequent distance measurements based on signal strength unreliable. Disabling this feedback loop is discussed by Zhou et al. [7]. Another problem is that it takes a lot of time for a new device to be fully discovered. Very often it means that the user has already left the area [8]. This makes Bluetooth trilateration impractical; however coarser room-level positioning can be done relatively quickly as device pairing is not required.

Notably, it is not reported possible to achieve accuracy below one meter [9] using RF-based technologies present in mobile phones [4, 5, 10]. However, sound travels at significantly slower speeds than radio waves and can therefore be easily localised to a few centimetres due to this much longer time of flight. Other useful features of sound show that it is possible to emit a 21 KHz (just above the human hearing range) signal from a mobile phone speaker and successfully receive it with a conventional microphone [11]. In a separate study, Peng et al. [12] showed that it is possible to utilize sound in order to measure the distance between two mobile phones using synchronized time-of-arrival techniques.

In previous work [16], we tested the useable range of SmartPhone ultrasound to find that these signals can indeed be successfully detected up to distances of 20m or more (Figure 1). In this experiment, two values below 10 dB were registered but this is still well above the 21.5 KHz component of background noise, which is around 1 dB. However there is no guarantee that the maximum value belongs to a signal that arrived by direct path and not via a longer deflected path. In any case, it can be seen from the shape of the graph that even with speaker and microphone pointing directly at each other, signal strength can't be relied on alone to accurately measure distance.

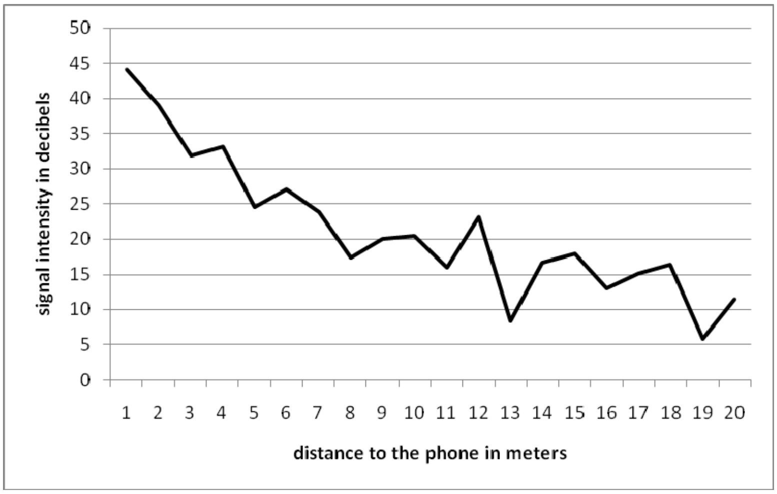


Fig. 1. Relationship between signal strength and distance for conditions where SmartPhone speaker and microphone point at each other

Therefore, in our Lok8 trilateration method we endeavour to make use of the useful characteristics of inaudible mobile ultrasound by exploiting the *differences* in signal time-of-arrival at a static microphone array for accurate mobile positioning. An accuracy comparison of our method compared to other reported indoor positioning methods, together with their availability for implementation on today's SmartPhones, is also given in Table 1.

3 Time Difference of Arrival (TDOA) Trilateration

Sound is a mechanical wave which travels at speeds much slower than the speed of light. In dry air at a temperature of 25°C the speed of sound is only 346 m/s. At such propagation speeds, one sample of a standard 44.1 KHz stream (44100 cycles/second) accounts for 0.8cm of distance [4, 13]. In other words a signal will travel only 0.8 centimeters in the duration of the smallest time grain. Technically it is possible to work with sound even at 384 KHz, which can give much finer accuracy.

As discussed previously, by using trilateration it is possible to calculate one's position based on the distance to several other (control) points with known positions [14, 15]. To find one's position in 2 dimensions the number of required known points is 3; for position in 3 dimensions the number of known points is 4. Given that the speed of sound propagation is constant under the same temperature and humidity conditions, the time it takes a signal to travel between the phone to each known microphone control point can be directly converted into distance between the phone and microphones. This is the TOA (Time of Arrival) approach. In general, the main problem with this approach is that both the time the signal was sent and the time it was received are required in order to get the time of flight.

In our scenario of quickly and accurately locating a mobile phone indoors, TOA requires that times from two separate systems with two separate clocks will have to be synchronised - a major source of error. As such it is desirable to compare only the time of arrival at each of the microphones and ignore completely the time the signal was originally sent from the phone, making Lok8 a TDOA (Time Difference of Arrival) approach. The problem is illustrated in Figure 2 and the detailed solution follows.

Problem

- Mobile phone (P) has unknown position (X_P,Y_P).
- 4 microphones (M1, M2, M3, M4) have known positions (X_{M1},Y_{M1}), (X_{M2},Y_{M2}), (X_{M3},Y_{M3}), (X_{M4},Y_{M4})
- 4 distances (d_1, d_2, d_3, d_4) from P to M1, M2, M3, M4 are unknown but the differences between them (m_2, m_3, m_4) are measured ultrasonically; these are the *observations*.
- Find coordinates of P=(X_P,Y_P) by solving a system of equations (mathematical model) that relates the *m = 3 observations* (m_2, m_3, m_4) to the *n = 2 unknown parameters* (X_P,Y_P).

Solution

Although the coordinates of P could be found using readings from only 3 microphones (2 observations), 4 or more readings can be effectively used in the method of *Least Squares* to determine the *Most Probable Value* (MPV) for the coordinates of P, plus a *Standard Deviation* for the MPV.

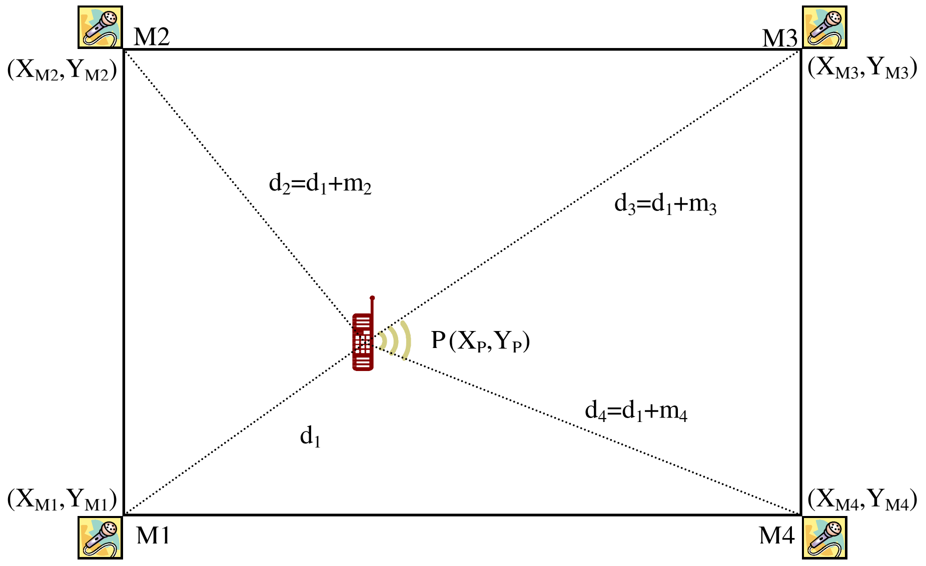


Fig. 2. Time Difference of Arrival. Control points M1, M2, M3 and M4 are known microphone positions. Point P is the unknown mobile phone's position, coordinates of which we are trying to find. Lines d1, d2, d3 and d4 are unknown distances between the phone and each microphone. However, what are known are the differences between the three measurements: m2, m3 and m4.

Least Squares Method for TDOA Trilateration

From Pythagoras we derive the following *mathematical model* to describe the ultrasonic relationships between phone P and microphones M1, M2, M3, M4:

$$d_1^2 = (X_P - X_{M1})^2 + (Y_P - Y_{M1})^2 \quad \text{or} \quad d_1 = \sqrt{(X_P - X_{M1})^2 + (Y_P - Y_{M1})^2}$$

$$d_2^2 = (X_P - X_{M2})^2 + (Y_P - Y_{M2})^2 \quad \text{or} \quad d_2 = \sqrt{(X_P - X_{M2})^2 + (Y_P - Y_{M2})^2}$$

$$d_3^2 = (X_P - X_{M3})^2 + (Y_P - Y_{M3})^2 \quad \text{or} \quad d_3 = \sqrt{(X_P - X_{M3})^2 + (Y_P - Y_{M3})^2}$$

$$d_4^2 = (X_P - X_{M4})^2 + (Y_P - Y_{M4})^2 \quad \text{or} \quad d_4 = \sqrt{(X_P - X_{M4})^2 + (Y_P - Y_{M4})^2}$$

However, we can re-write d_2, d_3, d_4 in terms of d_1:

$$\begin{gathered} d_2 = d_1 + m_2 \\ d_3 = d_1 + m_3 \\ d_4 = d_1 + m_4 \end{gathered}$$

And then substitute above d_1 expressions back into the mathematical model:

$$\begin{aligned} d_1 + m_2 &= \sqrt{(X_P - X_{M2})^2 + (Y_P - Y_{M2})^2} \text{ or } m_2 = \sqrt{(X_P - X_{M2})^2 + (Y_P - Y_{M2})^2} - d_1 \\ d_1 + m_3 &= \sqrt{(X_P - X_{M3})^2 + (Y_P - Y_{M3})^2} \text{ or } m_3 = \sqrt{(X_P - X_{M3})^2 + (Y_P - Y_{M3})^2} - d_1 \\ d_1 + m_4 &= \sqrt{(X_P - X_{M4})^2 + (Y_P - Y_{M4})^2} \text{ or } m_4 = \sqrt{(X_P - X_{M4})^2 + (Y_P - Y_{M4})^2} - d_1 \end{aligned}$$

Then replace d_1 in m_2, m_3, m_4 equations above with equivalent d_1 expression from mathematical model to give:

$$\begin{aligned} m_2 &= \sqrt{(X_P - X_{M2})^2 + (Y_P - Y_{M2})^2} - \sqrt{(X_P - X_{M1})^2 + (Y_P - Y_{M1})^2} \\ m_3 &= \sqrt{(X_P - X_{M3})^2 + (Y_P - Y_{M3})^2} - \sqrt{(X_P - X_{M1})^2 + (Y_P - Y_{M1})^2} \\ m_4 &= \sqrt{(X_P - X_{M4})^2 + (Y_P - Y_{M4})^2} - \sqrt{(X_P - X_{M1})^2 + (Y_P - Y_{M1})^2} \end{aligned}$$

Re-write above three mathematical model equations as *observation equations* by adding a residual v_m to each measurement:

$$\begin{aligned} \text{F:} \quad m_2 + v_{m2} &= \sqrt{(X_P - X_{M2})^2 + (Y_P - Y_{M2})^2} - \sqrt{(X_P - X_{M1})^2 + (Y_P - Y_{M1})^2} \\ \text{G:} \quad m_3 + v_{m3} &= \sqrt{(X_P - X_{M3})^2 + (Y_P - Y_{M3})^2} - \sqrt{(X_P - X_{M1})^2 + (Y_P - Y_{M1})^2} \\ \text{H:} \quad m_4 + v_{m4} &= \sqrt{(X_P - X_{M4})^2 + (Y_P - Y_{M4})^2} - \sqrt{(X_P - X_{M1})^2 + (Y_P - Y_{M1})^2} \end{aligned}$$

Because number of measurements (m = 3) is greater than number of unknowns (n = 2), use Least Squares to determine the MPV of the unknowns (X_P,Y_P). Since the observation equations are non-linear in the unknowns (X_P,Y_P), a first-order *Taylor Series* is needed to approximate a set of linear observation equations before taking partial derivatives.

Considering function F above (describing ultrasonic relationship between M2 and P):

$$\text{F:} \quad m_2 + v_{m2} = \sqrt{(X_P - X_{M2})^2 + (Y_P - Y_{M2})^2} - \sqrt{(X_P - X_{M1})^2 + (Y_P - Y_{M1})^2}$$

This non-linear function can be written as:

$$F(X_P, Y_P) = m_2 + v_{m2}$$

Where

$$F(X_P, Y_P) = \sqrt{(X_P - X_{M2})^2 + (Y_P - Y_{M2})^2} - \sqrt{(X_P - X_{M1})^2 + (Y_P - Y_{M1})^2}$$

The above function is *linearized* using a first-order Taylor Series approximation:

$$F(X_P, Y_P) = F(X_{Po}, Y_{Po}) + \left(\frac{\partial F}{\partial X_P}\right)_o dX_P + \left(\frac{\partial F}{\partial Y_P}\right)_o dY_P$$

Where

- X_{Po} and Y_{Po} are initial estimates of SmartPhone position in the environment calculated by taking average of all known microphone positions.
- $F(X_{Po}, Y_{Po})$ is the non-linear function evaluated with these estimates.
- dX_p and dY_p are corrections to the initial estimates such that $X_p = X_{p_o} + dX_p$ and $Y_p = Y_{p_o} + dY_p$.

The partial derivatives $\left(\frac{\partial F}{\partial X_P}\right)$ and $\left(\frac{\partial F}{\partial Y_P}\right)$ are found by first re-writing function F:

F: $$F(X_P, Y_P) = \left((X_P - X_{M2})^2 + (Y_P - Y_{M2})^2\right)^{\frac{1}{2}} - \left((X_P - X_{M1})^2 + (Y_P - Y_{M1})^2\right)^{\frac{1}{2}}$$

and then take partial derivative with respect to X_P:

$$\begin{aligned}
\frac{\partial F}{\partial X_P} &= \frac{1}{2}\left((X_P - X_{M2})^2 + (Y_P - Y_{M2})^2\right)^{-\frac{1}{2}} \bullet 2(X_P - X_{M2}) \\
&\quad - \frac{1}{2}\left((X_P - X_{M1})^2 + (Y_P - Y_{M1})^2\right)^{-\frac{1}{2}} \bullet 2(X_P - X_{M1}) \\
&= \frac{(X_P - X_{M2})}{\sqrt{(X_P - X_{M2})^2 + (Y_P - Y_{M2})^2}} - \frac{(X_P - X_{M1})}{\sqrt{(X_P - X_{M1})^2 + (Y_P - Y_{M1})^2}} \\
&= \frac{(X_P - X_{M2})}{d_1 + m_2} - \frac{(X_P - X_{M1})}{d_1}
\end{aligned}$$

and then with respect to Y_P:

$$\frac{\partial F}{\partial Y_P} = \frac{1}{2}\left((X_P - X_{M2})^2 + (Y_P - Y_{M2})^2\right)^{-\frac{1}{2}} \bullet 2(Y_P - Y_{M2})$$
$$-\frac{1}{2}\left((X_P - X_{M1})^2 + (Y_P - Y_{M1})^2\right)^{-\frac{1}{2}} \bullet 2(Y_P - Y_{M1})$$
$$= \frac{(Y_P - Y_{M2})}{\sqrt{(X_P - X_{M2})^2 + (Y_P - Y_{M2})^2}} - \frac{(Y_P - Y_{M1})}{\sqrt{(X_P - X_{M1})^2 + (Y_P - Y_{M1})^2}}$$
$$= \frac{(Y_P - Y_{M2})}{d_1 + m_2} - \frac{(Y_P - Y_{M1})}{d_1}$$

Where d_1 is always (re)evaluated using Pythagoras at current estimates for (X_P, Y_P).
Therefore:

$$F(X_P, Y_P) = F(X_{Po}, Y_{Po}) + \left(\frac{(X_P - X_{M2})}{d_1 + m_2} - \frac{(X_P - X_{M1})}{d_1}\right)_o dX_P$$
$$+\left(\frac{(Y_P - Y_{M2})}{d_1 + m_2} - \frac{(Y_P - Y_{M1})}{d_1}\right)_o dY_P$$

So the linearized observation equation for m_2, describing the ultrasonic relationship between microphone M2 and phone P becomes:

$$\left(\frac{(X_P - X_{M2})}{d_1 + m_2} - \frac{(X_P - X_{M1})}{d_1}\right)_o dX_P + \left(\frac{(Y_P - Y_{M2})}{d_1 + m_2} - \frac{(Y_P - Y_{M1})}{d_1}\right)_o dY_P$$
$$= (m_2 - m_{2_o}) + v_{m2}$$

Likewise for function G (between M3 and P):

$$\left(\frac{(X_P - X_{M3})}{d_1 + m_3} - \frac{(X_P - X_{M1})}{d_1}\right)_o dX_P + \left(\frac{(Y_P - Y_{M3})}{d_1 + m_3} - \frac{(Y_P - Y_{M1})}{d_1}\right)_o dY_P$$
$$= (m_3 - m_{3_o}) + v_{m3}$$

and function H (between M4 and P):

$$\left(\frac{(X_P - X_{M4})}{d_1 + m_4} - \frac{(X_P - X_{M1})}{d_1}\right)_o dX_P + \left(\frac{(Y_P - Y_{M4})}{d_1 + m_4} - \frac{(Y_P - Y_{M1})}{d_1}\right)_o dY_P$$
$$= (m_4 - m_{4_o}) + v_{m4}$$

When using Matrix Methods for Least Squares, the observation equations are represented in matrix form as:

$${}_mA_n\,{}_nX_1 = {}_mL_1 + {}_mV_1$$

Where in our case:

- $m = 3, n = 2$
- ${}_mA_n$ contains the coefficients of the unknowns (X_P, Y_P)
- ${}_nX_1$ contains the corrections to be applied to the initial estimates for the unknowns (dX_P, dY_P)
- ${}_mL_1$ contains the measurements (m_2, m_3, m_4)
- ${}_mV_1$ contains the residuals (one for each measurement).

Solving for X gives the solution:

$$X = \left(A^T A\right)^{-1} A^T L \quad \text{where:}$$

$$A = \begin{bmatrix} \frac{(X_P - X_{M2})}{d_1 + m_2} - \frac{(X_P - X_{M1})}{d_1} & \frac{(Y_P - Y_{M2})}{d_1 + m_2} - \frac{(Y_P - Y_{M1})}{d_1} \\ \frac{(X_P - X_{M3})}{d_1 + m_3} - \frac{(X_P - X_{M1})}{d_1} & \frac{(Y_P - Y_{M3})}{d_1 + m_3} - \frac{(Y_P - Y_{M1})}{d_1} \\ \frac{(X_P - X_{M4})}{d_1 + m_4} - \frac{(X_P - X_{M1})}{d_1} & \frac{(Y_P - Y_{M4})}{d_1 + m_4} - \frac{(Y_P - Y_{M1})}{d_1} \end{bmatrix}$$

$$X = \begin{bmatrix} dX_P \\ dY_P \end{bmatrix} \quad L = \begin{bmatrix} m_2 - m_{2_0} \\ m_3 - m_{3_0} \\ m_4 - m_{4_0} \end{bmatrix} \quad V = \begin{bmatrix} v_{m2} \\ v_{m3} \\ v_{m4} \end{bmatrix}$$

Matrix X contains the corrections to be applied to the original estimates for (X_P, Y_P). These new (X_P, Y_P) coordinates are then used to recalculate updated distances for $(d_1, m_{2_0}, m_{3_0}, m_{4_0})$. The process is repeated until coordinates of (X_P, Y_P) don't change significantly (e.g. in the 3rd decimal place for mm precision).

After a solution has been reached, the residuals V for each measurement and Standard Deviation of *unit weight* σ_o for the overall least squares adjustment can be calculated with:

$$V = AX - L \qquad \text{and} \qquad \sigma_o = \pm\sqrt{\frac{\left(V^T V\right)}{r}}$$

Where *degrees of freedom* r = m–n and the Standard Deviation of each adjusted unknown is then given by:

$$\sigma_{Xi} = \pm\sigma_o\sqrt{(Q_{XiXi})}$$

In our case $\boldsymbol{\sigma}_{X_1}$ is the Standard Deviation for X_P, and $\boldsymbol{\sigma}_{X_2}$ is the Standard Deviation for Y_P. These standard deviations imply that there is a 68% probability that the adjusted values for X_P and Y_P are within $\pm\sigma$ of this amount.

$\left(A^T A\right)^{-1}$ is called the *variance-covariance* matrix or $\left(Q_{XX}\right)$ matrix and $\left(Q_{XiXi}\right)$ is the *variance* of unknown *i*, or the element in the i^{th} row and i^{th} column of the $\left(A^T A\right)^{-1}$ matrix.

Practical Example

To test the accuracy of our TDOA Trilateration method, we used it to calculate the position of several random SmartPhone locations and compare the results to their actual positions in Figure 3. We used four control points (microphones) arranged in the corners of a rectangular room to locate the phone's position at 6 different locations within the room.

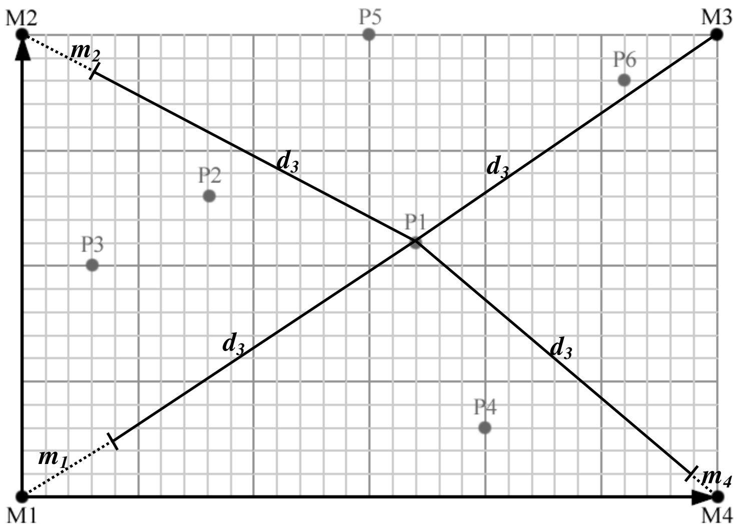


Fig. 3. TDOA Trilateration experiment with four microphones and six different smartphone positions. Control points M1, M2, M3 and M4 are microphones. Points P1, P2, P3, P4, P5 and P6 are actual SmartPhone locations. Each square of the grid represents 1 unit in length.

Regarding input data for testing the Lok8 trilateration algorithm, the locations of M1(0,0), M2(0,20), M3(30,20), M4(30,0) were used and the initial distances between the mics and the various phone positions were measured manually. Although we could have used Pythagoras in Figure 3 to calculate exactly the measurements representing the ultrasonic distances between the microphones and various phone positions, we wanted to introduce some error in the measurements so chose instead to simply use a ruler to measure these distances on paper to one decimal point precision. After that we subtracted the shortest measured distance for any given phone position from each of the remaining three mic distances. The resulting 3 distance differences were then used as “ultrasonic” input to the asynchronous trilateration procedure in addition to the known microphone locations.

For example, for phone position P1 the measured distance to M1 was 20.2, to M2 19.2, M3 15.8, and M4 17.0. The shortest distance is to M3, therefore it is subtracted from the other 3 distances to leave; m_1= 4.4, m_2= 3.4, m_4= 1.2. These values simulate time measurements translated to distance for the ultrasonic signal to reach these 3 mics after first triggering the server clock at M3. The input data is summarised in Table 2 and the trilateration results for the phone’s position relative to the 4 microphones are compiled in Table 3. Notice that if we assumed metres for units in this example, the standard deviations for the phone positions are of sub-metre accuracy.

Table 2. Sample TDOA Trilateration input. Second and third columns contain coordinates of a microphone and fourth column contains differences between distance to mic and closest mic. In this example microphone M3 is closest to phone position P1 so its corresponding distance difference equals zero.

Mic	X	Y	Distance Difference (m_i)
M1	0	0	4.4
M2	0	20	3.4
M3	30	20	0
M4	30	0	1.2

Table 3. Comparison of TDOA output and expected results. Second column contains X and Y coordinates of a given phone position, third column contains coordinates of the phone as calculated by our TDOA trilateration procedure. Fourth and fifth columns contain the Standard Deviations (σ_X, σ_Y) for each trilaterated phone positon and number of iterations to get there.

Phone Point	Actual Location	TDOA Trilateration	Standard Deviation	Number of Iterations
P1	17 , 11	16.987, 10.986	0.0002, 0.0003	3
P2	8 , 13	7.978, 12.966	0.0158, 0.019	3
P3	3 , 10	2.96, 10.0	0 , 0	4
P4	20 , 3	20.002, 2.996	0.011, 0.0195	3
P5	15 , 20	15.0, 20.0	0 , 0	4
P6	26 , 18	25.999, 18.031	0.0144, 0.0214	4

4 Conclusions and Future Work

In this paper we demonstrated an asynchronous trilateration method that can be reliably used to accurately locate an ultrasonic signal source without knowing the time the signal was sent. This eliminates the need to synchronize clocks between signal source and receivers.

An advantage of using a Least Squares approach for trilateration is its ability to tolerate errors in measurements; with more measurements provided, less is the impact from a single erroneous measurement. Also, due to the iterative nature of this approach allowing for a large *pull-in range*, initial approximations for a phone's position in a room can be simply taken as the average of all microphone (control point) positions. While the algorithm can work with only three receivers (mics), at least four or more are recommended for scenarios where measurements are likely to be contaminated with signal noise caused by multipath propagation.

For future work we plan to implement our TDOA Trilateration method in a real-time indoor positioning system on COTS SmartPhones and interconnected mics. We will then evaluate how well Lok8 manages with unavoidable measurement errors due to background noise, obstructions, and uncertainty due to the presence of multiple ultrasonic source devices.

Acknowledgments. The authors wish to thank the Higher Education Authority (HEA) in Ireland and their Technological Sector Research Strand III for funding the ultrasound work in the Lok8 project. Preparation of this publication and the asynchronous trilateration work was funded by a Strategic Research Cluster Grant (07/SRC/I1168) by Science Foundation Ireland under the National Development Plan. The authors also gratefully acknowledge this support.

References

1. Siciliano, B., Khatib, O.: Springer Handbook of Robotics. Springer, Heidelberg (2008)
2. Williams, B., Klein, G., Reid, I.: Real-Time SLAM Relocalisation. In: Computer Vision (2007)
3. Wagner, D., Schmalstieg, D.: First Steps Towards Handheld Augmented Reality. In: 7th IEEE International Symposium on Wearable Computers. IEEE Computer Society (2003)
4. Otsason, V., Varshavsky, A., LaMarca, A., De Lara, E.: Accurate GSM Indoor Localization. In: Pervasive and Mobile Computing (2007)
5. Ferris, B., Hähnel, D., Fox, D.: Gaussian Processes for Signal Strength-Based Location Estimation. In: Robotics Science and Systems (2006)
6. Ekahau RTLS Overview, `http://www.ekahau.com/products/real-time-location-system/overview.html` (cited May 21, 2009)
7. Zhou, S., Pollard, J.: Position measurement using Bluetooth. IEEE Transactions on Consumer Electronics 52(2), 555–558 (2006)
8. Kolodziej, K., Hjelm, J.: Local positioning systems: LBS applications and services (2006)
9. Addlesee, M., Curwen, R., Hodges, S., Newman, J., Steggles, P., Ward, A., Hopper, A.: Implementing a Sentient Computing System. IEEE Computer 34(8), 50–56 (2001)
10. Hallberg, J., Nilsson, M., Synnes, K.: Positioning with Bluetooth. In: ICT (2003)
11. Borriello, G., Liu, A., Offer, T., Palistrant, C., Sharp, R.: WALRUS: Wireless Acoustic Location with Room-Level Resolution using Ultrasound. In: Mobisys (2005)
12. Peng, C., Shen, G., Zhang, Y., Li, Y., Tan, K.: BeepBeep: A High Accuracy Acoustic Ranging System using COTS Mobile Devices. In: SenSys (2007)
13. Navizon Technical Paper (2007), `http://www.navizon.com/Navizon_wifi_gps_and_cell_tower_positioning.pdf` (cited January 11, 2009)

14. Bossler, J., Jensen, J., McMaster, R., Rizos, C.: Manual of Geospatial Science and Technology, 1st edn. Taylor & Francis (2002)
15. Ghilani, C., Wolf, P.: Adjustment Computations: Spatial Data Analysis, 4th edn. John Wiley & Sons, Inc. (2006)
16. Filonenko, V., Cullen, C., Carswell, J.D.: Investigating Ultrasonic Positioning on Mobile Phones. In: Proc. of International Conference on Indoor Positioning and Indoor Navigation (IPIN). IEEE Xplore (2010)

A Semantic Resolver for Coordinate Reference Systems

Dimitar Misev, Mihaela Rusu, and Peter Baumann

Jacobs University Bremen,
Campus Ring 1, 28759 Bremen, Germany
{d.misev,m.rusu,p.baumann}@jacobs-university.de

Abstract. Coordinate Reference Systems (CRSs) are at the heart of geo services. A large number of very different CRSs is in active use worldwide. EPSG maintains a well-known inventory of geodetic CRSs upon which most service implementations and also the standards of the Open Geospatial Consortium rely. For recent data and services, however, this set turns out insufficient - time series, climate and ocean data, etc. require time axes and non-spatio-temporal axes in addition. Further, when retrieving slices from geo data cubes, new, unforeseen CRSs need to be constructed on the fly, such as lat/t or lon/z CRSs.

To achieve a uniform handling of all CRSs we propose a URL-based identification scheme. This scheme naturally extends OGC's EPSG identification URLs with parametrized CRSs and ad-hoc CRS and axis combination. Our approach currently is under discussion within OGC for adoption as a standard. We have implemented a CRS resolver based on this concept which accepts identification URLs and returns corresponding GML definitions. The resolver is available publicly to study and demonstrate feasibility of URL-based CRS naming schemes.

In this paper we summarize the concept and implementation of the resolver as a service following the described name type specification.

Keywords: Coordinate Reference Systems, CRS identifiers, OGC.

1 Introduction

Geo services make heavy use of Coordinate Reference Systems (CRSs) to define the semantics of coordinates relating object locations in space and time to some reference position. A vast amount of different CRSs is in active use today, each of which fits best a particular requirement. Most common are

– Predefined CRSs, such as WGS84;
– Families of predefined CRSs, such as the EPSG set;
– Compound CRSs, such as the result of combining a geodetic 2-D coordinate reference system like WGS84 with a vertical 1-D CRS like elevation or bathymetry;

– Parameterized CRSs, such as the AUTO CRS family defined in the OGC Web Map Service (WMS) standard [16]; these CRSs represent templates which require, e.g., instantiation with a particular point of origin. Each such CRS thereby describes an infinite number of concrete CRSs.

The Open Geospatial Consortium (OGC) geo service standardization body and ISO have established an unambiguous description for CRSs [6] as part of the Geography Markup language (GML) [4]. While these can be used to describe a wide range - though not all - CRSs required, geo applications typically prefer dealing with CRS identifiers, such as those provided by EPSG [13]. OGC has standardized URL identifiers for these well-known CRSs, thereby supporting interoperability across agents handling georeferenced data. However, the new, complex nature of coverages [5] general as space/time varying phenomena necessitates reconsideration and extension of CRS concepts.

While GML provides a solid basis for CRS description, it is not always adequate for objects with more than two dimensions. Among others, time is handled separately from space, thereby complicating handling, and non-spatio-temporal axes are missing. Other shortcomings do not relate so much to GML, but rather to the CRS sets available. For example, unorthodox axis combinations like lat/t and lon/z are not available; array indexes (misleadingly called "Image CRS") are not convincingly integrated; definition of domain-specific axes like pressure (heavily used in climate modeling as a replacement for height) is not possible.

Hence, while establishing the OGC Web Coverage Service (WCS) 2.0 suite of standards which distinctly generalize coverages beyond the traditional 2-D horizontal lat/long imagery to data and service standards for the full spectrum of coverage data structures as defined by the abstract ISO 19123 [8] and the concrete, 19123-based GML Application Schema for Coverages [5], abbreviated GMLCOV. The need arose while defining CRS handling in the WCS standard to establish a more general scheme for identifying CRSs.

A simple, expressive, and HTTP compatible mechanism is required, allowing Web services to create, identify, and understand CRSs and their definitions. We address this with a Name Type Specification (NTS) which allows to identify predefined, combined, and parametrized CRS definitions via URLs. We have implemented these concepts in a **Se**mantic **Co**ordinate Reference System **Re**solver (SECORE), representing a registry service able to resolve conforming URLs to CRS definitions expressed in GML. Concepts and service are currently under discussion within OGC for general adoption, starting with a trial phase of our implementation in spring 2012.

The remainder of this paper is structured as follows. First, we inspect the state of the art. In Sections 2 and 3 we introduce the core concepts of CRS Name Types. The resolver service concept is presented in Section 4, while we outline its publicly available implementation in Section 5. Section 6 concludes the paper.

1.1 Related Work

SECORE is based on (but not limited to) the EPSG Geodetic Parameter Dataset [12], a repository of parameters required to define Coordinate Reference

Systems, along with transformations and conversions (coordinate operations) that allow coordinates from one CRS to be translated to another. The EPSG Geodetic Parameter Registry [13] is an ebXML repository also operating on the EPSG Dataset in GML [9], but implemented according to ebRIM 3.0 [3]. For example, the following request will return the WGS84 definition:

```
http://epsg-registry.org/indicio/query?request=getRepositoryItem&
  id=urn:ogc:def:crs:EPSG::4610
```

The EPSG Registry differs from SECORE in that it's simply a repository for the EPSG Dataset. It allows to retrieve entities from the dataset, but the URLs used do not identify the resolved resources.

Further, as part of work to unify Web service handling, OGC is preferring URLs over URNs, which leads to a syntactic (although not semantic) change of identifiers. In such a URL representation, the above CRS name appears as

```
http://www.opengis.net/def/EPSG/0/4610
```

and is accompanied, for example, by the time CRS as defined by ISO:

```
http://www.opengis.net/def/ISO/0/8601
```

2 Basic Concepts

A coordinate is a sequence of n values describing an n-dimensional position. When expressing the location of some object in space and time through coordinates, these coordinates need to have an unambiguous reference, usually given by a CRS. CRSs consist of a coordinate system (an abstract mathematical concept defined by a sequence of axes with specified units of measure) which is related to a specific reference object (Earth, ship, etc.) through a datum. Typically, however, it is inconvenient for both producers and consumers of such information to communicate the complete definition of the CRS used; rather, a unique identifier is preferred – even more as the vast majority of applications rely on some commonly accepted standard CRSs.

The OGC Web Service (OWS) Common standard [14] specifies that an OWS shall always reference a CRS by using an XML attribute or element of type `anyURI`. Such an `anyURI` value can be used to reference a CRS whether the definition of that CRS is or is not included in the same document transferred, can or can not be electronically accessed.

The name type specification defines the syntax of such `anyURI` items. In its basic form, a URL uniquely identifies a CRS. Through a parametrization mechanism, the mere identification is extended beyond single CRS identifiers to also express ad-hoc combination of CRS components. OGC specifications may allow parametrized URLs in some places, and sometimes require unique CRS URLs. A typical situation occurs with the OGC Web Coverage Service (WCS) [15]. Its CRS Extension supports not completely specified CRS URLs to be computed, thereby enabling a compact presentation of all coordinate reference systems supported by a particular service. When accessing a coverage (e.g., by subsetting it) a unique CRS URL is passed to the server to concretely specify the coordinate transformation to be applied.

2.1 General Identifier Syntax

A CRS Identifier is a URL which has the common URL format:

```
"http:" hierarchical-part [ "?" query ] [ "#" fragment ]
```

In RESTful terminology, the resolver accepts a URL and returns a GML document as the representation of the resource, the CRS definition addressed. Some examples may illustrate these concepts.

1. A coverage document (such as a climate data set or a satellite image) may contain a local CRS definition referred to by its srsName attribute:
   `srsName="#my_local_CRS"`
2. Company ACME may offer CRS definitions (i.e., employ a CRS Identifier resolver) understanding CRS URLs like:
   `http://www.acme.com/def/this-is-EPSG-4326`
3. The following URL resolves to a predefined CRS, namely N2000 height:
   `http://www.opengis.net/def/crs/EPSG/0/3900`

Generally, not all possible parameter values will identify some existing (or meaningful) CRS. The authoritative decision about validity of OGC CRS URLs is with the OGC CRS Name Resolver with service endpoint *www.opengis.net/def.*

Query format. In the query format variant, a CRS Identifier shall be constructed as an HTTP GET query, following this syntax:

```
"http:" hierarchical-part "?" par1 "=" val1 "&" par2 "=" val2 "&" ...
```

where `par1`, `par2` etc. represent parameter names and `val1`, `val2`, etc. the values passed for the corresponding parameter.

For the CRS identifiers, the authority, version and code are required, and there can't be duplicates.

Path Format. In the special case that the CRS identifier consists of an authority, a version and a code, the path variant, i.e., a RESTful syntax, can be used alternatively to the query syntax. A URL in path format shall adhere to the syntax variant:

```
"http:" hierarchical-part "/" authority "/" version "/" code
```

where the placeholders `authority`, `version`, and `code` denote admissible values for the corresponding lower-case variable parts defined above. This syntax establishes backward compatibility to pre-existing OGC CRS definitions.

Table 1. Identifier parameters

Name	Definition	Data type
authority	the OGC-specified abbreviation for the authority organization that specified the referenced definition. As such, it identifies an authority recognized by the OGC, for example "EPSG" or "ISO"	NCName
version	the version of the authority or code for the referenced definition. When the referenced definition does not have a version use the string "0" (without quotes).	String
code	unique identifier of the referenced CRS definition, as specified by the referenced authority, for example "4326" is the EPSG code for WGS84.	NCName

3 Identifier Types

3.1 Predefined Definitions

Referencing predefined CRSs is the simplest case for handling – a CRS identifier identifies a CRS definition as it is stored on the server. Such an identifier consists of three components: the authority that maintains the CRS, the version and the CRS code. Consider for example the identifier for the WGS84 coordinate reference system:

```
http://www.opengis.net/def/crs/EPSG/0/4326
```

The same identifier can be represented in a key-value pair (KVP) query format which allows for more flexibility:

```
http://www.opengis.net/def/crs?authority=EPSG&version=0&code=4326
```

Besides CRS definitions, further entities like datums, meridians, elipsoids, operations, etc. are also referenceable. An example identifier for the Greenwich prime meridian:

```
http://www.opengis.net/def/meridian/EPSG/0/8901
```

3.2 Parameterized CRSs

In WMS 1.3 [16] a special class of "automatic" coordinate reference systems that include a user-selected centre of projection was defined. Here we extend this concept to generic *parameterized* CRSs, where it is possible to instantiate an abstract, or *template* CRS definition to a concrete and unique CRS based on user-supplied parameter values, and optionally parameters derived from these via some mathematical formula.

We propose a simple XML syntax for defining parameterized CRSs, and exemplify it by looking at the auto universal transverse mercator layer CRS (AUTO2:42001).

```
<ParameterizedCRS>
  <parameters>
    <parameter name="lon"/>
    <parameter name="lat">
      <value>0.0</value>
    </parameter>
    <parameter name="zone">
      <value>min( floor( (${lon} + 180.0) / 6.0 ) + 1, 60 )</value>
    </parameter>
    <parameter name="central_meridian">
      <value>-183.0 + ${zone} * 6.0</value>
      <target>//greenwichLongitude</target>
    </parameter>
    <parameter name="false_northing">
      <value>(${lat} >= 0.0) ? 0.0 : 10000000.0</value>
      <target>//falseNorthing</target>
    </parameter>
  </parameters>
  <identifier>http://www.opengis.net/def/crs/AUTO/1.3/42001</identifier>
  <targetCRS xlink:href="http://www.opengis.net/def/crs/EPSG/0/4326"/>
</ParameterizedCRS>
```

As can be noticed, the definition of `ParameterizedCRS` is very similar to the other GML CRS types, like GeodeticCRS, ProjectedCRS, etc. The main difference is that it references the particular CRS it parameterizes (via `targetCRS`), instead of referencing a coordinate system.

The main part of a parameterized CRS definition is the parameters list. At instantiation time, the parameter `value`s for which `target`s have been specified will be substituted in the respective target elements[1]. A parameter value is allowed to reference values of other parameters by enclosing the corresponding parameter names in an `${...}`. Dereferencing a parameter value means recursively substituting all parameter references with the actual referenced values, so that in the end the value contains no references.

The parameter values themselves have to be either provided by the user via the CRS identifier, or specified in the parameterized CRS definition. A parameter value is valid if it

– can be successfully dereferenced;
– is a valid Java Script expression [10] when dereferenced;
– does not participate in a circular reference, e.g. Φ referencing Ψ, and vice versa, Ψ referencing Φ.

A CRS identifier for parameterized CRSs is an extension of the identifiers for predefined CRS. In addition to the authority, version and code, parameter values can be specified for the corresponding parameter names. For example, the following URL instantiates the above CRS with a value of -100 for longitude:

```
http://www.opengis.net/def/crs/AUTO/0/42001?lon=-100
```

[1] Selected via XPath expressions.

Since we have omitted the value for `lat`, the default of 0.0 will be used in this case. The `zone` parameter will be then evaluated to 14 based on the given `lon` value, then `central_meridian` to -99.0 and `false_northing` to 0.0.

3.3 Composing CRSs

Traditionally coordinate reference systems are either horizontal (2D) or vertical (1D). A 3D or higher dimensional point is usually described by combining horizontal coordinates from one coordinate reference system with height or depth coordinates from another for example. Such a point composed of coordinates from different coordinate reference systems is referenced to a new *compound* CRS, which is composed of the respective, non-repeating, single (not compound) component CRSs [6]. The coordinate order in the compound CRS follows the order of the coordinates in the component coordinate reference systems.

In the context of these compound CRSs we have a special type of a *combined* CRS URL in which the parameter value provided itself is a CRS URL. A combined CRS URL is a CRS identifier with authority OGC and code `crs-compound`, together with further parameters describing a combination of complete CRSs into a new compound CRS.

For example, there is no pre-existing coordinate reference system for referencing 4D latitude/longitude/height/time data, so we get a compound CRS on the fly with the following identifier:

```
http://www.opengis.net/def/crs-combine?
   1=http://www.opengis.net/def/crs/EPSG/0/4326&
   2=http://www.opengis.net/def/crs/EPSG/0/4440&
   3=http://www.opengis.net/def/crs/ISO/2004/8601
```

Here we combine the geodetic WGS84 CRS, with the NZVD2009 vertical CRS and the ISO 8601 temporal CRS. The parameter order determines the CRS order in the resulting compound CRS.

In another case we may want to obtain pressure/time slices from a 4D latitude/longitude/time/pressure atmospheric data cube, which requires a pressure/time CRS. There is no such predefined CRS, so we again find combining of single coordinate reference systems necessary.

3.4 Axes

Coordinate system axes can be spatial (example: latitude, elevation), temporal, or none of both (example: pressure). While these concepts are uniquely defined, they frequently have several synonyms, for example, elevation is synonymous to altitude, height, and z. Likewise, CRS Axis URLs uniquely define axes for use in CRS definitions, but allow synonyms. Therefore, while axes URLs can contain identifiers in KVP and REST formats like

```
http://www.opengis.net/def/axis-name/EPSG/0/9906
http://www.opengis.net/def/axis-name?authority=EPSG&version=0&code=9906
```

they can also be referenced by name:

`http://www.opengis.net/def/axis/Easting`

Table 2 lists axis names which shall be synonymous. Each element in a row shall be substitutable by any other element in the same row. For example, the geodetic latitude axis can be identified by `http://www.opengis.net/def/axis/y` or, synonymously, by `http://www.opengis.net/def/axis/Geodetic%20latitude`.

While non-spatiotemporal axes like pressure or language translations will remain individual, synonyms in Table 2 are expected to get standardized.

Table 2. Axis synonyms

Axis synonyms	Default
Y, y, Longitude, longitude, Lon, lon, Long, long	Geodetic longitude
X, x, latitude, Latitude, Lat, lat	Geodetic latitude
Z, z, Height, height, Elevation, elevation, Altitude, altitude	Gravity-related height
Depth, depth	Gravity-related depth
T, t, Time, time	Temporal

4 Implementation

SECORE is a classical three-tier system (Fig. 1). This architecture makes the system flexible, as any tier can be developed, tested, and modified/replaced independently from the other tiers.

The front-end accepts requests in GET-KVP and RESTful syntax, decodes and parses them, and then forwards them to the application logic residing in the middle layer if they are syntactically well formed, or returns an error otherwise.

The middle tier contains the application logic that does the core work, constructing the response given a parsed request.

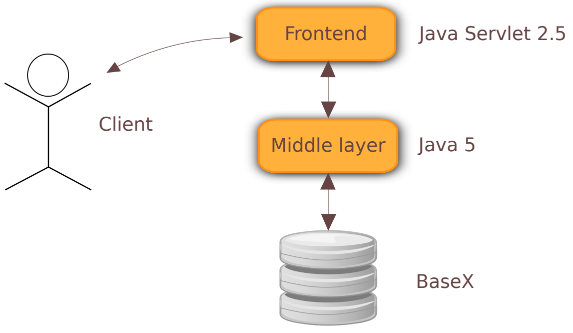


Fig. 1. SECORE architecture

Our service returns XML results in GML format, so all data is stored in an XML database (BaseX [1]). The database is mostly static (all data loaded once at startup), so our usage will focus on querying. For querying we use the XQuery API for Java (XQJ), which to XML databases is same as the JDBC API is to relational database. By using a standard interface we are not committing to any particular XML database, as most have support for the XQJ API.

Apart from the above, our resolver provides a hierarchical browser interface, which allows to browse through the predefined definitions and to perform real-time changes to the loaded data. This also permits to modify the axis synonyms accordingly. For a more extended usage it supports the direct execution of XQuery statements on the database.

5 Demonstrator

The web service conforming to the above specifications supports requests following both the compact RESTful syntax and the more intuitive KVP syntax. The format of the requests can be generally changed without the result being affected. A demonstration of the service that can be readily tested is available on EarthLook [2], an interactive online demonstration of open geo services.

For the following we anticipate the URL domain *www.opengis.net*, although this is not mandatory; the EarthLook sample service is available at *kahlua.eecs.jacobs-university.de:8080*.

KVP Syntax. The KVP syntax allows for more flexibility in the request, such as shuffling parameters. Both URLs below are equivalent:

```
http://www.opengis.net/def/crs?authority=EPSG&version=0&code=4327

http://www.opengis.net/def/crs?version=0&code=4327&authority=EPSG
```

RESTful Syntax. While being more compact, the RESTful syntax is very restrictive in terms of parameters. Therefore, the identifier can be queried only in the predefined order described in 2.1: `http://www.opengis.net/def/crs/EPSG/0/4327`

Axis Synonyms. In case of axis names, the resolver returns the same definition for the synonyms from Table 2.

Response. The resolver response is an XML file consisting of the CRS definition identified by the request URL, or an exception in case the request was badly formed or it did not resolve to any definition.

6 Conclusion

The proposed standard for specifying CRS resources via URLs facilitates the identification and exchange of CRS definitions. The adaptive, RESTful URL syntax smoothly embeds itself into existing service standards and CRS identification mechanisms. The implementation of the resolver web service is available as a fully usable software component to be released in open source. Moreover, being composed of lightweight components that are grouped together in loosely connected layers and employing current web standards and technologies, the resolver is flexible in terms of possible changes and improvements, both to the standard and to the underlying components.

Currently, the resolver is being established as a beta service by OGC with the goal of making it the official reference resolver for OGC CRS standards, to be used by all services which make use of CRSs. That said, the freely available resolver code allows any entity to set up its private CRS resolution realm for use as either addition to or replacement of the OGC service.

Next steps include specifying URL identifiers for deriving a projected CRS definition from a specific geographic CRS via coordinate conversion; extending concepts to include combined references for concatenated operations; extending the list of synonyms for axes.

References

1. Gruen, C.: Storing and Querying Large XML Instances. PhD Thesis. University of Konstanz (January 2011) ISBN: 978-3838124599
2. EarthLook, `http://www.earthlook.org` (accessed November 24, 2011)
3. ebXML Registry Information Model (RIM) v3.0. OASIS ebXML Registry TC (May 2005)
4. Geography Markup Language (GML) Encoding Standard, version 3.2.1. OGC document 07-036
5. GML 3.2.1 Application Schema - Coverages, version 1.0.0. OGC document 09-164r1
6. ISO 19111:2007: Geographic information – Spatial referencing by coordinates
7. ISO 19111-2:2009: Geographic information – Spatial referencing by coordinates – Part 2: Extension for parametric values
8. ISO 19123:2005: Geographic Information – Schema for Coverage Geometry and Functions
9. ISO 19136:2007: Geographic information – Geography Markup Language (GML)
10. JSR-231 – Scripting for the JavaTM Platform. Final Draft Specification, version 1.0
11. Name type specification – definitions – part 1 – basic name. OGC document OGC 09-048r3
12. OGP Publication 373-7-1 – Geomatics Guidance Note number 7, part 1 (June 2011)
13. OGP Publication 373-7-3 – Geomatics Guidance Note number 7, part 3 (March 2010)
14. OGC Web Service Common Implementation Specification. OGC document 06-121r9
15. Web Coverage Service (WCS) 2.0 Core Interface Standard. OGC document 09-146r1
16. WMS 1.3.0 – OpenGIS Web Map Service (WMS) Implementation Specification. OGC document 06-042

Visualization with a New Visual Metaphor for Hierarchical and Stratified Temporal Domain

Francesco Cerasuolo[1,2], Francesco Cutugno[1], and Vincenza Anna Leano[1]

[1] LUSI-Lab, University of Naples “Federico II”, Italy
[2] Istituto di Scienza e Tecnologie dell'Informazione "Alessandro Faedo", CNR Pisa, Italy
{cutugno,vincenzaanna.leano}@unina.it,
francesco.cerasuolo@isti.cnr.it

Abstract. Studies have proved that 80% of data has a spatial reference [1] and it would be reasonable to assume that the temporal reference has a similar relevance. Spatio-temporal data visualization assumes an important role in the data presentation to users. Offering a synchronized view on three dimensions of data (i. e. descriptive, temporal and spatial) helps users in their knowledge discovery process. In some fields, like Cultural Heritage, time assumes an important role to explore data. The same timeline could be view in different thematic context, temporal domain could be stratified and the time reference could be qualitative and imprecise. Managing this kind of features improves the ability of the users to recognize pattern in data. In this paper is presented a spatio-temporal data visualization application that shows a new method to explore hierarchical and stratified temporal domains with imprecise temporal reference. The application interface offers a high level of personalization. Users can choose between different type of views and visual metaphors to compare objects on geobrowser activating or disabling layers of their interest. Interface is open and extendible to new kinds of visualization.

Keywords: Spatio-Temporal Visualization, Spatio-Temporal Exploration, Cultural Heritage, Temporal Exploration.

1 Introduction

Several different forms of spatio-temporal data types are available in real world and the current state of art offers many ways to classify them (e.g. see[2], [3], [4]). For each spatio-temporal dataset several visualization metaphors have been proposed [5] in order to improve users knowledge discovering process. Many of the currently available spatio-temporal visualization systems focus on the spatial and descriptive representation of the spatio-temporal phenomena, limiting the representation of the temporal dimension to a numeric value or to a one-dimensional time-bar (timeline).

In some fields, like Cultural Heritage, the temporal domain can assume a complex structure. Experts in this field would partition the temporal domain in different manners depending on the context (i.e. “History of Campi Flegrei” or “History of Literature”). Each context might have different granularity levels (for example “History of

Naples" could have levels: "Epoch" "Ages", "Empire", "Battle", "Important Events"). The elements of this timeline might not have a precise quantitative temporal dating and can eventually limit themselves to topological relation with other events in the same context. Mapping this complex temporal structure into numeric values or to a one-dimensional time-bar representation would cause a loss of information and could compromise the data exploration process. There is a need of an adequate temporal visualization, integrated and synchronized with spatial and descriptive ones.

In this paper we present a new interaction metaphor to visualize spatio-temporal data with a hierarchical and stratified temporal domain. The interaction interface allows to independently interact with the three dimension of data (spatial, temporal and descriptive) [6]. A web application based on the proposed interaction metaphor is presented. The application interface offers spatiotemporal visualization with a high level of personalization. It incorporates and synchronizes spatial, temporal, and descriptive views in an integrated and extensible way. Users can choose between different types of views and visual metaphors to see and compare objects into the geobrowser, activating or disabling information layers they are interested in. By interacting with a map, users can perform spatial queries to obtain information about the referencing objects and their content. Users can visualize the complex temporal structure in an intuitive way and they can perform complex temporal queries by clicking on the temporal component. The web application relies on a flexible three tiers architecture that shows low coupling about tiers and uses standard exchange data formats like WFS, KML, GML, in order to guarantee the independence from storage and visualization tools.

The rest of the paper is structured as follows: in Section 2 we describe some related works on spatio-temporal visualization, Section 3 describes the proposed interaction metaphor, in Section 4 we present the spatio-temporal visualization metaphor we designed running on a sample case study. Finally we will present some conclusions and future works.

2 Related Work

Adequate spatio-temporal data visualization techniques improve human's mental analysis capabilities allowing users to "see" knowledge inside data [7].

In a spatio-temporal analysis, users interact with information in a different manner, discovering, analyzing and exploring information into a unique interoperable environment ([8], [9]). To allow this kind of knowledge acquisition, user needs tools to support data exploration to validate and/or reformulate hypothesis. These tools are fundamental for the analysis and exploration of data in the assisted process using adequate spatio-temporal visualization techniques, providing a high level of interactivity, a fundamental feature for the analysis and exploration of data [10].

In [5] authors provide a classification of existing visualization techniques for exploratory analysis of spatio-temporal data. They extend the Pequet's classification scheme of data components users are interested in (Where, When, What) [6] using the notion of reading levels individually applied to spatial, temporal and spatio-temporal data [11]. This approach generates four categories of spatio-temporal queries, and for each of them the authors provide with the best visualization technique, like Map iteration, Map animation, time windowing and so on.

In [12] the authors illustrate a tool based on ArcGis to investigate Historic site changes. In their interface animation maps, time-life bar and 3D model view of the History site are used. Users can navigate only by time, stopping, forwarding or rewarding the animation map. The maps, the time life bar and the 3D model are not interactive.

Stefanakis in [13] presents a prototype educational framework for modeling, analyzing and visualizing the Ancient Greek Mythology. He deals with events without any absolute time reference, but are related each other by topological relations. He proposes a web interface that shows Myths on a map. In this interface temporal navigation is missing, furthermore granularity of space and time is missing in the model.

With regard to the temporal dimension, managing quantitative and qualitative temporal reference is a well-known problem in the temporal database literature ([14], [15]) and in Cultural Heritage field [16]. Theoretical works on temporal domains [17] introduce the concept of temporal granularity, with the aim to study the expressiveness power and the decidability properties of navigational operators on stratified temporal domains. Nowadays available temporal data visualization techniques know in the literature do not take fully into account the above temporal domain feature.

3 An Interaction Metaphor

Many of the existing spatio-temporal visualization systems often focus on one of the peculiarity of temporal domain (e.g. linear time versus cyclic time), or process the temporal feature as a numeric one, without taking into account the full complexity of the temporal domain.

As happen for the space, users might partition the domain in thematic contexts, and for each context they would define different levels of granularity. This organization has to be efficiently visualized in order to make evident the difference between temporal thematic context and the recognition of both granularity levels and temporal relationships [18] among temporal events. The temporal visualization, integrated and synchronized with spatial and descriptive ones, enhances the spatio-temporal exploration.

In the following subsections we will describe the designed temporal interface that exposes the proposed temporal visual metaphor based on the hierarchical and stratified temporal model.

3.1 Spatio-Temporal Visualization Interface

The Spatio-Temporal Visualization Interface design was inspired by two basic concepts:

1. *The triad model* [6]. The object can be seen under three dimension: spatial, temporal and descriptive. The underlying model and the visualization metaphor reflect the threefold nature of data.
2. *Shneiderman's visual information seeking mantra* [19], that is an (iterative) three step approach: overview first, zoom and/or filter, and details on demands. The proposed interface offers an overview of the data in a spatial domain and gives the opportunity to users to filter them by temporal queries using the timeline, or by applying filter on the descriptive dimension. Requested details of a selected object are showed in a separate panel.

The developed user interface is composed of four basic components, integrated in a web page (Fig. 1):

- *Geobrowser*: it is responsible of rendering data on a map, allowing users to explore the spatial dimension. Users can navigate through the map by selecting areas of interest and activating the (spatial) thematic layers existing into the dataset.
- *Timebar*: it is the distinctive element of the proposed interface, allowing users to interact with the temporal dimension implementing the visual metaphor described in section 3.3
- *Active Filters Panel*: it helps users to focus on the descriptive dimension by selecting the information layer to be visualized on the map. They can choose and personalize the type of visualization, by activating and disabling layers, allowing for an easily data crossing.
- *Spatial Query Panel*: it is responsible of interacting with the geobrowser in order to allow users to perform spatial queries (i.e. visualizing objects nearest to a point in a circle area defined by a chosen radius) or by drawing the interested area on the map.

This interface allows users to navigate among the three dimensions: spatial, temporal and descriptive. The components are fully synchronized, they use standard protocols and data format to communicate.

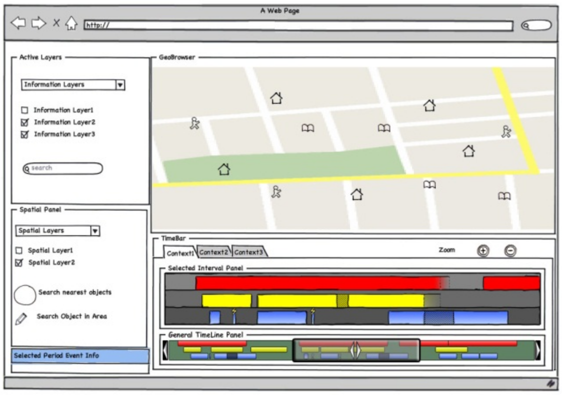


Fig. 1. A Mock-up of the User Interface

3.2 Temporal Model

It can happen that some kind of data do not always present precise purely quantitative time reference (see for example the case study on Cultural Heritage). Events could be qualitatively referenced to others by ordering relations (before, after etc.) or an Event may hinge on others. Frequently, events have a duration (period). Moreover experts would make different partitions on time based on a context of interest and a notion of temporal granularity.

The temporal model presented in this paper merges the qualitative and quantitative aspect of time references using the de facto standard model [16] with the granularity concepts studied in theoretical works [17], adding the possibility to partition the temporal domain in thematic contexts, each of them with its granularity and the possibility of making quantitative and qualitative temporal references. In the following, the formal definition of thematic context and granularity is proposed, and then, the model managing quantitative and qualitative reference is presented.

Thematic Contexts and Time Granularity. The temporal domain can be partitioned in *thematic context*. Let us call $Cont=\{C_1, C_2, ..., C_n\}$ the set of thematic context defined by users. Each context has a finite number of layers, that represents the temporal granularity, let us call $Layers_C=\{L_1, L_2, ..., L_m\}$ with $C \in Cont$ the set of the layers defined for the context *C*.

Formally time domain is given by the set of pairwise disjoint pairs $(T_{C,L}, \leq_L)$, where $T_{C,L}$ is a non-empty set of *Temporal-Entities* in a context *C*, *L* is a layer in the context *C* and $\leq_L$ is a partial linear order relation on $T_{C,L}$. The ordering relation holds on *Temporal-Entities* belonging to the same Layer. The definition of time granularity is inspired by [17]. For a context *C*:

- L_1: contains *Temporal-Entities* of finer granularity such as atomic *Temporal Instant* and atomic *Temporal Interval.*
- L_i with $i \geq 2$: contains the remaining not atomic *Temporal-Entities.*

In the model of time considered in this work, the temporal entities are *Period-Event* objects. A *Period-Event* is an event or a time interval in a specific thematic context and it is qualified by a given level of granularity.

Quantitative and Qualitative Temporal Reference. In the assumed model the temporal reference of an object is given by the abstract concept Period-Event.

If the temporal reference is quantitative, then the *Period-Event* has a *temporal_extension* property that refers to an abstract object called *Temporal-Entity*. A *Temporal-Entity* could have:

1. a point temporal extension, called *Temporal Instant*, indicated by *t*;
2. an interval temporal extension, called *Temporal Interval*. We have two kind of *Temporal Intervals*. The first one is defined by a start temporal reference *s* and a final temporal reference *t* where *s* and *t* are *Temporal-Entities* (indicated as $i=[s,t]$). The second one models the temporal events that hinge on other events and

is defined by the triple *i=(off_ante, e, off_post)* where *e* is a *Temporal-Entity*, *off_ante* is the number of instants *i* preceding *e* and *off_post is* the number of instants *i* following *e*.

If the temporal reference is qualitative, then the *Period-Event* object has a *temporal_relationship* property that refers to another *Period-Event* object in the same *Context.* The structure of the adopted time model is presented in Fig. 2.

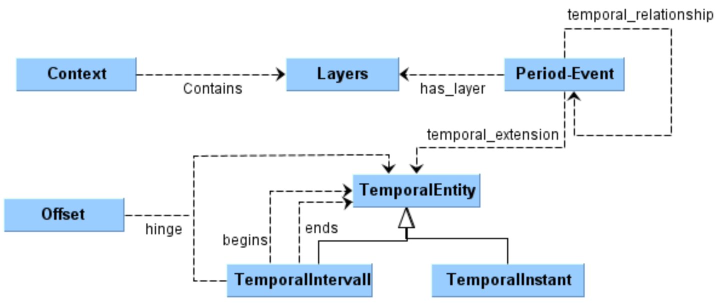


Fig. 2. Temporal Model

3.3 Temporal Visualization Metaphor

The temporal model defined in the previous section introduces some new features, allowing users to navigate information by thematic Context, granularities Layers and Period-Events. So an ad hoc temporal visualization metaphor has to be created.

The proposed Temporal Visualization Component (TVC), provides scalable temporal information, allowing users to dynamically choose the temporal detail level that best matches theirs search criteria. Through the TVC it is possible to make a targeted selection of temporal data, reducing the information overload by filtering data onto the temporal dimension.

A mockup of the proposed temporal visualization metaphor is shown in Fig. 3.

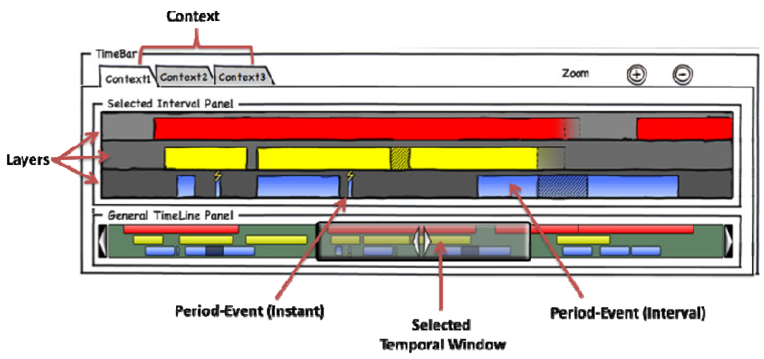


Fig. 3. Temporal Visualization Component Mock-up

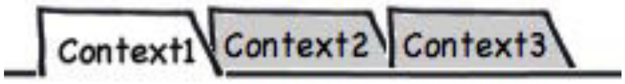


Fig. 4. Contexts Tab

The TVC has a selectable tab (Fig. 4) for each thematic temporal context defined in the database: by choosing one of them, only the corresponding temporal data are retrieved.

The temporal navigation interface is structured in two panels (described in the following subsections):

1. *the General TimeLine Panel*: it offers a global temporal overview and allows users to select a temporal interval of interest;
2. the *Selected Interval Panel*: it gives a detailed view of the temporal interval selected into the General Time Panel.

General TimeLine Panel. The aim of this panel is to offer the same functionality of an *Overview Map*, i.e. to show the location of the current view respect to a wider and more general context. Typically this interaction metaphor is used in bi-dimensional spaces, such as maps or images, to give an immediate insight of the rendered portion of space, given a wider domain. The General TimeLine Panel has the same goal, in the temporal domain, which is mono-dimensional. We choose a representation based on a scrollbar-like interaction (Fig. 5), where the whole time span is displayed, and a semitransparent selector is used to indicate the temporal frame currently considered.

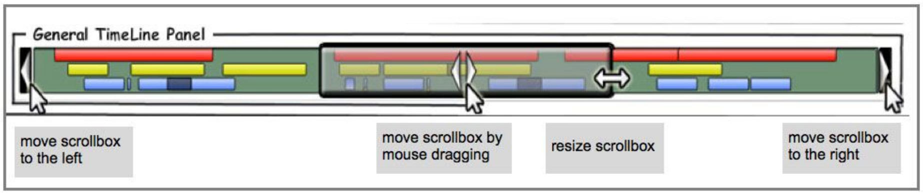

General TimeLine Panel
move scrollbox to the left
move scrollbox by mouse dragging
resize scrollbox
move scrollbox to the right

Fig. 5. The General TimeLine Panel, and the allowed interactions

This timeline should be the main temporal selector for users, allowing them to obtain a fast temporal navigation. Besides the usual interactions, other than the typical ones (moving the scrollbox and clicking the arrow buttons), we have envisioned also the possibility to directly resize the scrollbox itself, to change the covered time span. This can be achieved by dragging the borders of the scroll box, till the desired size.

Selected Interval Panel. The Selected Interval Panel offers a detailed view of the temporal area selected by the General TimeLine Panel. This panel has a twofold goal: the first one is to display all temporal information available in the database for the selected time span, and the second one is to allow users to select only some temporal elements of interest (i.e. period-events, intervals or specific instants), in order to further filter the data shown into the geobrowser.

The panel has a bar for each Layer belonging to the chosen context and shows the Period-Events distribution over the selected temporal window. Period-Events are differently shaped based on their temporal extension (Fig. 6):

– the temporal interval ones are represented by rectangles having a width directly proportional to the temporal length;
– the temporal instant ones are represented by stylized vertical bars with a flash on top.

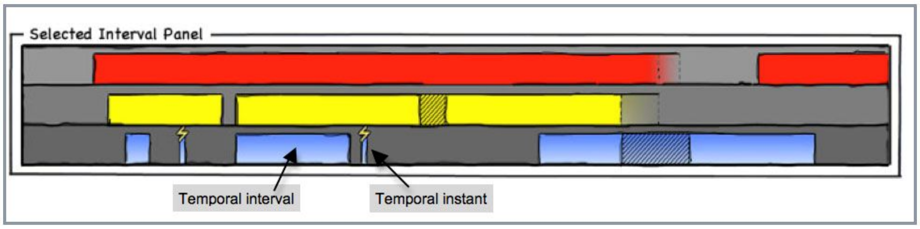


Fig. 6. Period-Events Representation

Users can interact with this panel by selecting a Period-Event just clicking on it, or by selecting an interval of interest, drawing a box onto the time span of interest (Fig. 7). The action of selecting a span causes a complex query on DB: this retrieves not only the data that is strictly related to the selected span, but also all the data that are related to the span by overlapping and inclusion relationship.

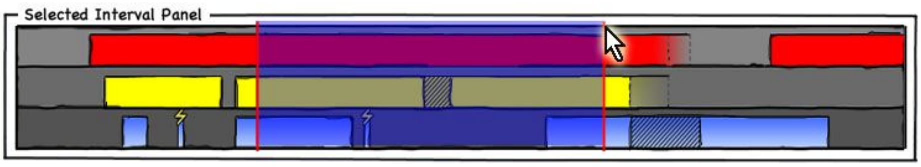


Fig. 7. Temporal Interval Selection

4 Web Based Prototype

The here proposed framework has been implemented as a web-based prototype, to assess its feasibility. The web application manages literary sources and related spatial and temporal aspect. The web architecture and the developed web prototype will be described in the following section as applied to a specific case study in the field of Cultural Heritage.

4.1 System Architecture

In this section the system web architecture is described. Our main goal is to realize a flexible architecture, where each module is characterized by a loose coupling, in order to achieve the independence from data storage and visualization tools.

In particular, the proposed architecture relies on a three-tier architecture, composed by a data storage at the back end, a business logic layer, and a visualization layer at the front end, arranged as shown in Fig. 8. As suggested by the OGC, to achieve modularization all the communication among or intra modules are carried out through standard open protocols and exchanging data formats. The three tiers of the proposed architecture are detailed in the following subsections.

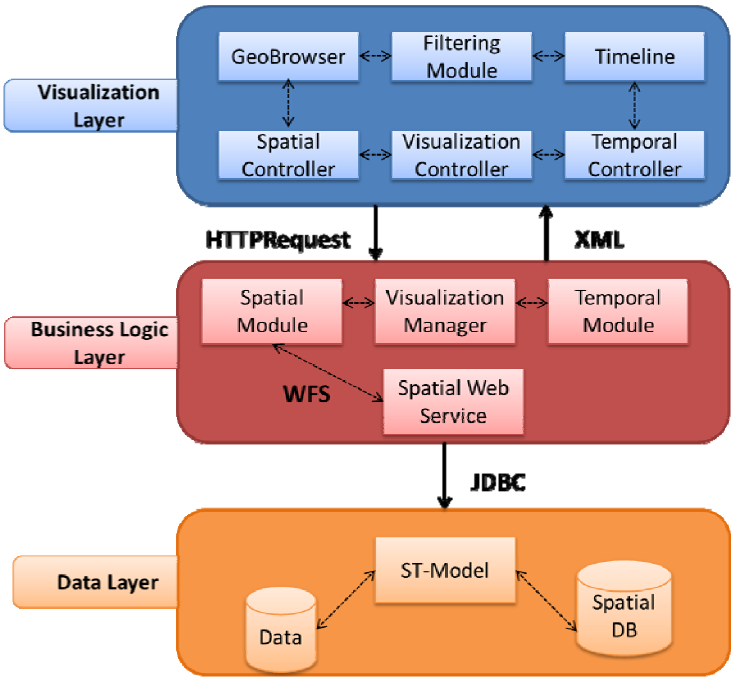


Fig. 8. System Architecture

Data Layer (DL). This component is responsible of storing and making persistent the application data. It is composed by a standard DMBS and a Spatial DBMS (in our case Oracle Spatial [20]) implementing the DB view of the model. This layer communicates with the upper one by means of a standard protocol like JDBC.

Business Logic Layer (BL). The layer is responsible for performing the operations required by the user and integrating data belonging to the heterogeneous data sources in a single data structure, to be fed to the *Visualization Layer*. This tier is composed of three main modules (*SpatialModule, TimelineModule* and V*isualizationManager)* and make use of OGC-compliant web-services for spatial operation (*SpatialWebService*).

- *VisualizationManager*. This component is responsible for interpreting and managing the operations received by the *Visualization Tier.* It dispatches the spatial and temporal request to the respective module and merges the results, completing them with the descriptive information produced by the heterogeneous DB queries that feeds the upper layer.
- *SpatialModule*. This module performs the spatial query requested by the user invoking a *SpatialWebService* through the standard protocol (WFS).
- *TimelineModule*. This component implements all the temporal operator defined in [18]. It is responsible of performing the temporal query requested and returns the results to the *VisualizationManager*.
- *SpatialWebService*. This component implements the OCG compliant web services (WFS, WCS, WMS), and it works with standard protocol like HTTP, SOAP etc. We adopt Geoserver [21] as our *SpatialWebService*.

Visualization Layer (VL). The main goal of this layer is to present spatial, temporal, textual and multimedia information merged together and offered to the users, furthermore we aim at notifying the underlying layers with the query they performed. *VL* uses a Geobrowser and a Timeline visual component in order to allow the spatial and temporal exploration and to make them independent. The descriptive data dimension is represented on the Geobrowser or/and in the descriptive panel (a more detailed description of the interface will be presented in the next section).

This Layer is composed by four components:

- *VisualizationController (VC).* It is responsible of handling events on the user interface and of notifying the Logic Tier about the operation the user wants to do. The extra-tier communication uses the HTTP/XML protocol, the intra-tier one uses an XML data format: as a response the *VC* receives XML data to refresh its visual components.
- *SpatialController*. It is able both to render user selected information onto a map and to interpret and notify the spatial query to the *VC*.
- *TimelineController*. The component implements the visual metaphor that allows an easily navigation and interaction with the hierarchal and stratified time dimension.
- *FilteringModule*. The module is responsible of applying the descriptive filters given by the user, to notify the user operations to the underlying layer and to display the descriptive dimension of data.
- The *SpatialController* was implemented using OpenLayers [22], the *TimelineController* was implemented extending the Simile Timeline [23], and all the components are managed and synchronized with JQuery [24] and AJAX technologies.

Exchange Protocol. The architecture here proposed uses standard files and protocols (WFS, GML) for the communication intra and extra module, in order to be independent from storage and visualization tools.

The user interaction is handled by the *Visualization Layer (VL)*, it interprets and intercepts the user query and notifies the *Business Logic Layer (BL)* performing an *HTTPRequest*. At this point the *BL* switches the request to the specific component. In particular, the spatial manager invokes a *Spatial Web Service* using the WMS standard protocol receiving a KML file as response. The *BL* invokes the data layer too in order to retrieve the descriptive data information from the DBMS using a JDBC connection. This response is merged with the KML from the spatial component and with the response from the temporal component building a new XML file. This XML is given as response to the *VL*. Filtering operations on the descriptive dimension are made using XQuery [25] on the merged XML file. An example of interaction is shown in Fig. 9.

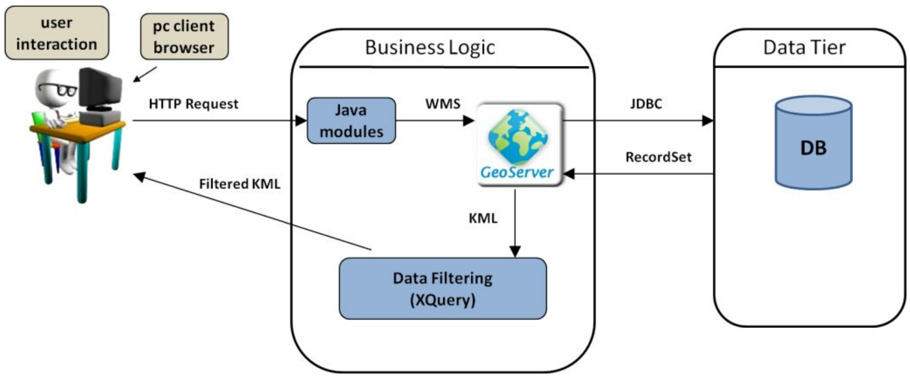


Fig. 9. Exchange protocol example

4.2 Web Application

The implemented case of study is a web application whose goal is to present some typical agricultural Products as referenced by Literary Sources from ancient Latin and Greek Authors produced in ancient times in the area named Campi Flegrei, a rich archeological site located nearby Naples (Italy) needing promotion in the spirit of increasing the international attention on Italian Cultural Heritage Patrimony. In our application Products, Literary Sources and Authors are represented in the same framework in their temporal and spatial dimensions. The application itself also manages the Biographic Notes on Authors.

The web interface consists of three synchronized areas (see Fig. 10): the Timeline that implements the visualization metaphor proposed in 3.3, the GeoBrowser that allows user to make spatial queries and visualize the spatial referenced object and the Active Layers Panel that allows the selection of the information layer to be visualized on the map. This interface allows users to navigate among the three dimensions: spatial, temporal and descriptive.

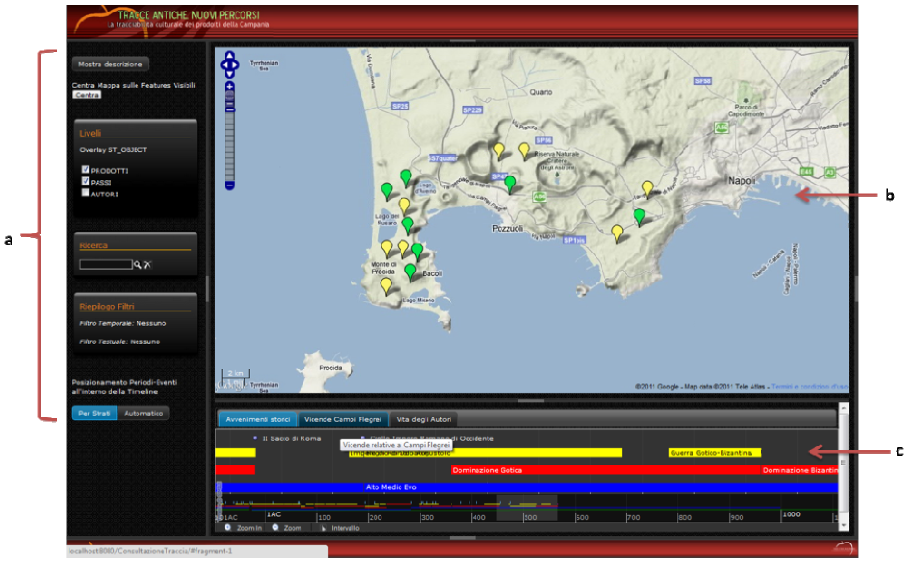


Fig. 10. Three panel web interface: a) Active Layers Panel b) Geobrowser c) Timeline

Clicking on a Period-Event or selecting an interval on the Timeline, users can filter information visualized on the map (Fig. 11). Timeline structure allows an easy understanding of temporal relationship [18] like before, after, etc., making also immediate to recognize the overlaps among Period-Event belonging to different layers.

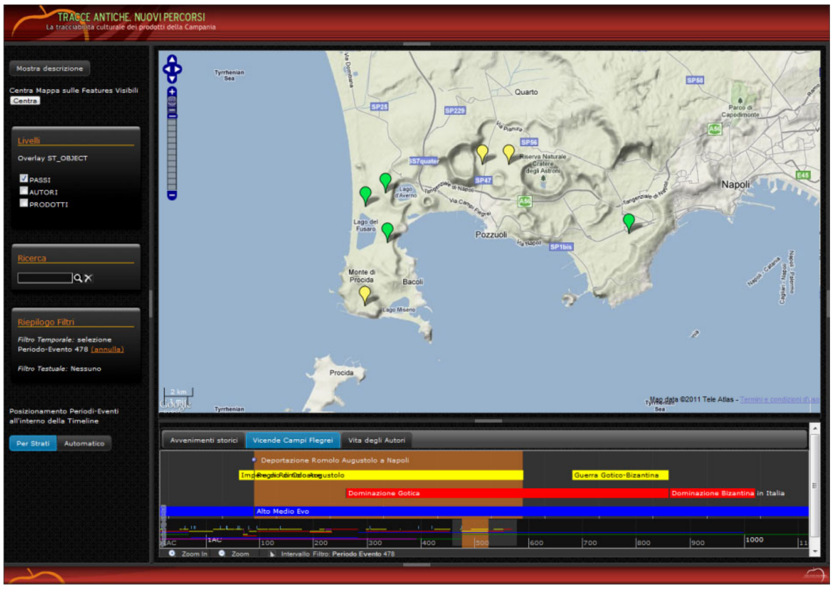
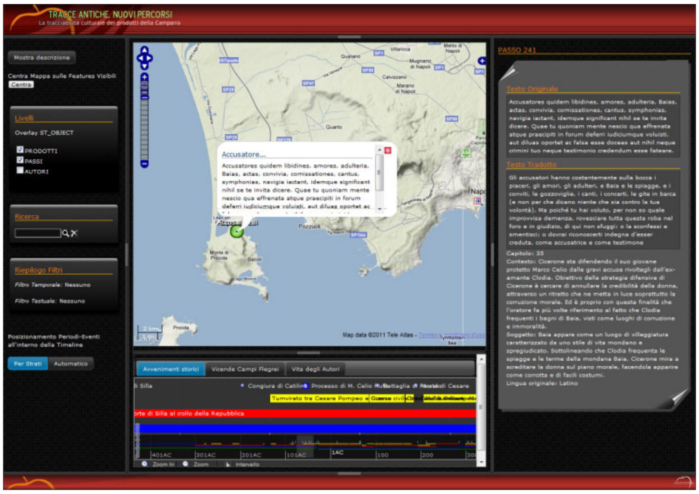

TRACCE ANTICHE. NUOVI PERCORSI
La tracciabilità culturale dei prodotti della Campania
Mostra descrizione
Centra Mappa sulle Features Visibili
Centra
Livelli
Overlay ST_OBJECT
PASSI
AUTORI
PRODOTTI
Ricerca
Riepilogo Filtri
Filtro Temporale: selezione Periodo-Evento 478 (annulla)
Filtro Testuale: Nessuno
Posizionamento Periodi-Eventi all'interno della Timeline
Per Strati
Automatico
Avvenimenti storici
Vicende Campi Flegrei
Vita degli Autori
Deportazione Romolo Augustolo a Napoli
Guerra Gotico-Bizantina
Dominazione Gotica
Dominazione Bizantina in Italia
Alto Medio Evo
Zoom In
Zoom
Intervallo Filtro: Periodo Evento 478
Quarto
Napoli
Pozzuoli
Bacoli
Procida

Fig. 11. Temporal Filter

Clicking on an object on the map, a balloon displaying some details shows up; clicking on a link in the balloon a further panel will appear, showing more detailed information (Fig. 12).

The *Active Layer Panel* helps users to focus on the descriptive dimension. In this case this given by "*Literary Sources*", "*Authors*" and "*Product*". In this panel users can perform textual search and can filter any kind of data they want to be visualizes on the map. User can also switch among visual metaphors, for example choosing to visualize the path of an author over his lifetime (Fig. 13).



Accusatore...

Accusatores quidem libidines, amores, adulteria, Baias, actas, convivia, comissationes, cantus, symphonias, navigia iactant, idemque significant nihil se te invita dicere. Quae tu quoniam mente nescio qua effrenata atque praecipiti in forum deferri iudiciumque voluisti, aut diluas oportet ac

PASSO 241

Testo Originale

Accusatores quidem libidines, amores, adulteria, Baias, actas, convivia, comissationes, cantus, symphonias, navigia iactant, idemque significant nihil se te invita dicere. Quae tu quoniam mente nescio qua effrenata atque praecipiti in forum deferri iudiciumque voluisti, aut diluas oportet ac falsa esse doceas aut nihil neque crimini tuo neque testimonio credendum esse fateare.

Testo Tradotto

Gli accusatori hanno costantemente sulla bocca i piaceri, gli amori, gli adulteri, e Baia e le spiagge, e i conviti, le gozzoviglie, i canti, i concerti, le gite in barca (e non par che dicano niente che sia contro la tua volontà). Ma poiché tu hai voluto, per non so quale improvvisa demenza, rovesciare tutta questa roba nel foro e in giudizio, di qui non sfuggi: o la sconfessi e smentisci; o dovrai riconoscerti indegna d'esser creduta, come accusatrice e come testimone

Capitolo: 35
Contesto: Cicerone sta difendendo il suo giovane protetto Marco Celio dalle gravi accuse rivoltegli dall'ex-amante Clodia. Obiettivo della strategia difensiva di Cicerone è cercare di annullare la credibilità della donna, attraverso un ritratto che ne metta in luce soprattutto la corruzione morale. Ed è proprio con questa finalità che l'oratore fa più volte riferimento al fatto che Clodia frequenti i bagni di Baia, visti come luoghi di corruzione e immoralità.
Soggetto: Baia appare come un luogo di villeggiatura caratterizzato da uno stile di vita mondano e spregiudicato. Sottolineando che Clodia frequenta le spiagge e le terme della mondana Baia, Cicerone mira a screditare la donna sul piano morale, facendola apparire come corrotta e di facili costumi.
Lingua originale: Latino

Fig. 12. Balloon and Details Panel

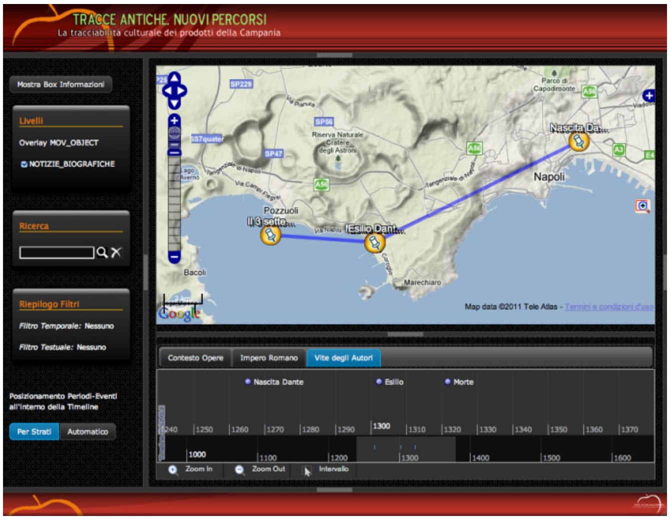


Fig. 13. Authors Life (moving object) metaphor

5 Conclusion and Future Works

In this paper we have proposed a new interaction metaphor to explore hierarchical and stratified temporal domain synchronized with a spatial and descriptive visualization.

The metaphor relies on web-architecture compliant with existing standards and that aims to be independent from data storage and visualization tools.

The spatial query panel presented in the mock-up is not yet fully implemented. We are planning to add other spatial and spatio-temporal visualization metaphor, like tag-maps and tag-cloud [26] treemaps [27], thematic maps etc. and to provide a hierarchical and stratified model and representation also for the spatial domain.

The web application representing the case of study we implemented to test our model is fully deployed and is currently under testing.

Acknowledgments. Authors wish to thank Francesco Mangiacrapa for his support in the design and development of the prototype proposed in this paper.

References

1. Franklin, C.: An introduction to geographic information systems: linking maps to databases. Database 15(2), 12–21 (1992)
2. Asproth, V., Hakansson, A., Revay, P.: Dynamic Information in GIS Systems. Computers Environment and Urban Systems 19(2), 107–115 (1995)
3. Kisilevich: Spatio-Temporal Clustering: a Survey. ISTI - CNR (2005)
4. Nadi, S., Mahmoud, R.: Spatio-Temporal Modeling of Dynamic Phenomena in GIS. In: Proceedings ScanGIS 2003 - The 9th Scandinavian Research Conference on Geographical Information Science, Espoo, Finland, June 4-6, pp. 215–225 (2003)
5. Andrienko, N.: Exploratory spatio-temporal visualization: an analytical review. Journal of Visual Languages & Computing 14, 503–541 (2003)
6. Peuquet, D.: Its About Time - a Conceptual-Framework for the Representation of Temporal Dynamics in Geographic Information-Systems. Annals of the Association of American Geographers 84(3), 441–461 (1994)
7. Camossi, E., Bertolotto, M., Bertino, E., Guerrini, G.: A multigranular spatiotemporal data model. In: Proceedings of the 11th ACM International Symposium on Advances in Geographic Information Systems, New Orleans, Louisiana, USA, pp. 94–101 (2003)
8. MacEachren, A., Kraak, M.: Research challenges in geovisualization. Cartography and Geographic Information Science 28, 3–12 (2001)
9. Andrienko, G., Andrienko, N., Jankowski, P., Keim, D., Kraak, M., MacEachren, A., Wrobel, S.: Geovisual analytics for spatial decision support: Setting the research agenda. Int. J. Geogr. Inf. Sci. 21(8), 839–857 (2007)
10. MacEachren, A., Gahegan, M., Pike, W., Brewer, I., Cai, G., Lengerich, E., Hardistry, F.: Geovisualization for knowledge construction and decision support. IEEE Computer Graphics and Applications 24, 13–17 (2004)
11. Koussoulakou, A., Kraak, M.: Spatio-temporal maps and cartographic communication. The Cartographic Journal 29, 101–108 (1992)
12. Lee, J., Chiu, H., Koshak, H.: Visualization System of Spatial-Temporal information for Historic sites based on GIS. In: Proceedings of Computers in Urban Planning and Urban Management (CUPUM 2005) Conference, London, UK (2005)

13. Stefanakis, E.: A journey to the ancient greek myths - An enhanced educational framework to story-telling with geo-visualization capabilities. In: Proceedings of the First International Workshop on Story-Telling and Educational Games (STEG 2008), Maastricht, The Netherlands (2008)
14. Koubarakis, M.: Database models for infinite and indefinite temporal information. Information Systems 19, 141–173 (1994)
15. Chaudhuri, S.: Temporal Relationships in DataBase. In: Proceedings of the 14th VLDB Conference, pp. 68–73 (1988)
16. Doerr, M., Kritsotaki, A., Stead, S.: Which Period is it? A Methodology to Create Thesauri of Historical Periods. In: Computer Applications and Quantitative Methods in Archaeology Conference, CAA 2004, Prato, Italy (2004)
17. Bettini, C., Wang, X., Jajodia, S.: A general framework and reasoning models for time granularity. In: Proceedings of Third International Workshop on Temporal Representation and Reasoning (TIME 1996), pp. 104–111 (1996)
18. Allen, J.: Time and time again: The many ways to represent time. International Journal of Intelligent Systems 6, 341–355 (1991)
19. Shneiderman, B.: The eyes have it: a task by data type taxonomy for information visualizations. In: IEEE Symposium on Visual Languages Proceedings, pp. 336–343 (1996)
20. Oracle: Oracle Spatial and Oracle Locator, `http://www.oracle.com/it/products/database/options/spatial/index.html` (accessed 2011)
21. OSGeo: Geoserver, `http://geoserver.org` (accessed 2011)
22. OpenLayers: OpenLayers, `http://openlayers.org/` (accessed 2011)
23. Huynh, D.: SimileTimeline. In: Simile Projects, `http://www.simile-widgets.org/timeline/` (accessed 2011)
24. JQuery: JQuery, `http://jquery.com/` (accessed 2011)
25. W3C: XQuery 1.0: An XML Query Language, 2nd edn., `http://www.w3.org/TR/xquery/` (accessed 2010)
26. Slingsby, A., Dykes, J., Wood, J., Clarke, K.: Interactive Tag Maps and Tag Clouds for the Multiscale Exploration of Large Spatio-temporal Datasets, pp. 497–504 (2007)
27. Slingsby, A., Dykes, J., Wood, J.: Using treemaps for variable selection in spatio-temporal visualisation. Information Visualization 7(3), 210–224 (2008)

Tag@Map: A Web-Based Application for Visually Analyzing Geographic Information through Georeferenced Tag Clouds

Davide De Chiara, Vincenzo Del Fatto, Monica Sebillo,
Genoveffa Tortora, and Giuliana Vitiello

Dipartimento di Matematica e Informatica
University of Salerno,
Via Ponte don Melillo, Fisciano (Salerno), Italy
{ddechiara,vdelfatt,msebillo,tortora,gvitiello}@unisa.it

Abstract. In their activities of monitoring a territory and its spatio-temporal phenomena, decision makers often face problems which require rapid solutions in spite of the complexity of scenarios under investigation. Researchers from the Geographic Information domain support their activities by providing them with highly interactive visualization tools able both to synthesize information from large datasets and perform complex analytical tasks. The goal of this paper is to present a Web-based application for on-line GeoVisual Analytics which exploits the visualization capability of the Tag Cloud technique, the interaction ability of visual environments, and the Web-based data querying and accessing. The application elaborates an interactive simplified map containing a georeferenced cloud of tags, placed where the associated information is appropriate and significant. By applying semantic and geographic operators on the map tags, users are allowed to acquire information about the underlying data, useful for a better comprehension of the described phenomena. A system prototype has been realized which builds an overview of data distribution and classification expressed as a georeferenced tag cloud. It also allows users to navigate and analyze maps through zooming and filtering operations defined in agreement with Visual Analytics guidelines.

Keywords: Geographic Information Visualization, GeoVisual Analytics, Advanced GeoVisualization Techniques for Web Applications, Interactive Visual Environments for Accessing and Analyzing Geospatial Information.

1 Introduction

In their activities of monitoring and prevention, decision makers often face problems which require rapid solutions in spite of the complexity of scenarios under investigation. Tasks such as risk discovery, simulations of *what-if?* scenarios, detection of cause/effect relationships represent daily activities that expert users have to perform in order to better respond to sudden and/or relevant events.

Researches in the domain of Geographic Information (GI, for short) aim at giving support to those activities with highly interactive tools by which expert users can both visually synthesize information from large datasets and perform complex analytical tasks. In particular, recently efforts from the GI community have been devoted to understand how to share expertise from the different key disciplines in order to reach the above goals.

GeoVisualization and GeoVisual Analytics represent a solution. They play a relevant role for the achievement of this aim, because they exploit results from several disciplines, such as exploratory data analysis and GIScience, to provide advanced tools able to semantically integrate quantitative, qualitative and cognitive aspects of a domain of interest.

The research we are carrying out along this line is meant to support expert users' work through software applications capable both to build an immediate overview of a scenario and to explore elements featuring it [1-3]. To this aim, we are defining methodologies and techniques which embed key aspects from different disciplines, such as augmented reality and location-based services. Their integration is targeted to realize advanced tools where the geographic component role is primary and is meant to contribute to a human-information discourse.

In the present paper we propose a GeoVisual Analytics method which assembles results from an initial phase of our research. It combines advanced GeoVis techniques for visualizing geographic data and the capability of Web-based visual environments to query a dataset. In particular, the proposed method adopts the Tag Cloud rationale and extends it by exploiting techniques for summarizing datasets and simplifying their geographic representation, such as Chorems and Cartograms. Starting from a geographic dataset, the proposed method extracts relevant information about a geographic area by counting and/or summarizing data, and generates a simplified map containing tags placed within the geographic area which original data are related to. The resulting map is interactive, that is to say, its tags along with the georeferenced cloud can be manipulated in order to answer specific users' requests. In particular, semantic and geographic operations have been specified whose aim is to both capture visual and semantic details, and select areas and data of interest.

In order to apply this method, a Web application, named Tag@Map, has been developed. It allows both to build a map containing the tag cloud, representing summarized data extracted from the underlying dataset, and to support expert users' interaction to analysis phenomena of interest.

The paper is organized as follows. Sections 2 and 3 introduce the proposed method. In Section 4 the system architecture and a prototype are described. An example of usage of the prototype is given in Section 5. Conclusions and future work are drawn in Section 6.

2 Visualizing Data Summaries by Georeferenced Clouds of Tags

Chorems [4], cartograms [5] and Tag Clouds [6] share the same approach for representing geographic information. They all aim to synthesize a large amount of data and translate the quantitative parameters in a qualitative representation.

In this paper we aim to increase the role that a map containing a cloud of tags may play in geographic domains, by assigning it an active role also when analysis tasks are required.

The idea underlying our proposal is based on the Visual Information-Seeking mantra stated by Ben Shneiderman [7] and on Keim’s adaptation to the Visual

Analytics domain, presented in [8]. Shneiderman's formulation "*Overview first, zoom and filter, then details on demand*", is a well-known visualization paradigm which encompasses several visual design guidelines and provides a general framework for designing information visualization applications. Shneiderman argues that by using the overview task, a user can gain a general idea of the entire collection of data and, at the same time, such an overview can help users control the content of the detail view [7]. Keim's adaptation of the mantra, "*Analyze First - Show the Important - Zoom, Filter and Analyze Further - Details on Demand*", points out that when dealing with the analysis of huge amounts of data, it is reasonable to apply first some analysis computations and then provide an overview of the resulting relevant contents. In that way the user may interact with a synthesis of important components of data, thus avoiding the risk to get lost inside the original data collection.

The goal of this section is to describe the visualization technique underlying the proposed Web application. Although this technique is an integral part of the whole process, it can be invoked separately to produce a visual synthesis of a scenario useful for the expert users' comprehension of a phenomenon under investigation.

Starting from a territory and a geographic dataset referring to it, the goal of the visualization technique is twofold, namely it both elaborates a simplified map of the territory and performs extraction and aggregation of data of interest. In particular, it counts and summarizes data and expresses them through tags placed within the geographic area which data are related to.

The method for visualizing geographic information combines Tag Cloud and advanced GeoVis techniques, and revises the Tag Map approach [9]. In particular, it exploits the GeoVis capability of summarizing datasets and simplifying their geographic representation also by altering their original shape. Moreover, it adopts tags to express concepts extracted from data sources, and finally, it adapts the georeferencing concept by associating it with the whole cloud.

Figure 1 depicts a map produced according to the proposed method. It illustrates the distribution of the most popular 60 Italian surnames. The prominence of each surname is expressed by its size which is proportionally depicted.

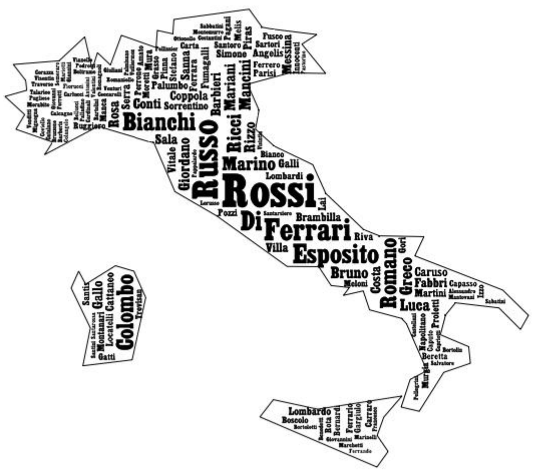


Fig. 1. A georeferenced tag cloud for the most popular Italian surnames

It is worth to notice that the position of each single tag is meaningless, because all of them refer to the geographic boundary of the Italian territory, that is, the tag placement is "area based", thus implying a graphic approach for the map layout. This tag independency from its specific position also guarantees the applicability of the best placement algorithm, which may elaborates different solutions. As a consequence, two subsequent applications of the method on the same dataset may produce two different clouds in terms of tag position, provided that their number and their size are identical, because both represent the same scenario / phenomena.

In [10] an initial prototype has been proposed meant to build a georeferenced cloud of tags and experiment its capability of conveying the expected information. In this paper, we modify and improve the cloud generation by applying an *ad hoc* algorithm which frees Tag@Map from an external web service, thus providing an application available for further customization. Moreover, we extend the role that the cloud of tags may play in geographic domains. As a matter of fact, conventional tag clouds are expressed as static maps meant to visually represent data distribution and frequency. In this paper, we refer to an application which generates an interactive simplified map where the cloud and its content can be directly manipulated. This function allows expert users to perform analytical tasks by querying data determining a frequency and understanding relationships among them. The GeoVisual Analytics approach based on the above visualization technique is described in the following Section.

3 Performing GeoVisual Analytics by Georeferenced Clouds of Tags

The synthetic global view offered by a map containing a tag cloud exactly results from the application of the *analize first – show the important* step of the extended mantra. Such maps are indeed conceived to provide a schematized representation of the distribution of a spatio-temporal phenomenon, built upon large sets of source data. In agreement with Shneiderman's theory, the overview represented by an initial map may be then used as a means to locate which part of it to analyze. It is always available during users' navigation both to orient and help them control the focus of the analysis.

As for the *zoom-and-filter* step of the visual information-seeking mantra, the goal is to focus attention on a reduced portion of the whole space. Zooming and Filtering represent the basic techniques used in information visualization to achieve this goal. They require a more complex interaction with the users by means of tools which allow them to control the zoom focus and the zoom factor. In our proposal, by zooming and filtering an interactive map containing tag clouds, users may gradually reduce the search space and select a subset of data in agreement with Shneiderman's and Keim's interpretation.

Once not necessary information has been visually discarded, the final step of the paradigm, namely *details on demand*, is meant to obtain detailed information about a particular element. In our proposal, users may easily browse a portion of a tag cloud and obtain descriptive information related to a selected tag.

In order to build the whole analysis path, we adopt the visual approach defined in [3] for analytical tasks, and customize the four operations to perform the aforementioned user interactive tasks.

Although chorems and tag clouds differ for the spatio-temporal phenomena they represent, sharing a similar approach for the information visual representation allow us to inherit the meaning of the operators and adapt it to the expected behavior when applied to tag clouds. Moreover, distinguishing geographic and semantic aspects of both tags and clouds also lets us obtain an exact relationship between operators and operands, thus providing users with a well-defined functionality to navigate an interactive map from an initial overview to a particular detail. In the following more details about these operators are given.

A Geographic Zoom corresponds to the traditional GIS zoom operator, also known as graphical or geometric zoom. It acts exclusively on the visual aspect of the map, by changing the size of the visible details of the involved tags, and allows the user to focus on a specific area. The Semantic Zoom originates from the well-known technique in Information Visualization research field useful to change the type and meaning of information displayed by an object. When a Semantic Zoom is applied on a selected area of a tag cloud, a different level of abstraction is applied on that area without affecting the initial map scale. In particular, invoking a semantic zoom-in (resp., -out) implies that the initial cloud is disaggregated (resp., aggregated) and a smaller area is centered and then expressed as a new cloud. Tags related to the new territory are calculated accordingly, thus allowing the access to a different level of information.

In order to select predefined areas, the Geographic Filter can be used to apply a spatial filter to a wider territory. In particular, an existing geometry can be used to cut a region of interest, thus requiring only tags belonging to that region. Both scale and level of information detail are unchanged.

Finally, the Semantic Filter operation allows users to filter tags which satisfy a particular condition, by directly operating on their semantics. Users can select the tags they need to analyze, and identify the descriptive element on which a threshold or a condition should be applied in order to reduce the set of visualized data.

In the following section, we describe the components of the system architecture underlying the prototype of the Tag@Map Web-based application, which embeds all the described operators.

4 The System Architecture and the *Tag@Map* Prototype

A system prototype, named TaG@Map, has been developed in order to implement the proposed method and to check its applicability.

In [10], the authors proposed a preliminary architecture consisting of the four following modules, namely *Data Selection*, *Data Aggregation and Generalization*, *Cloud Creation* and *Data Representation*. The architecture is focused on data extraction and aggregation, and output visualization in order to obtain an overview of data distribution and classification.

In this paper we adopt such an architecture by extending and adapting it to a conventional client-server architecture. The server side embeds a spatial database and an application which elaborates and publishes data in a map format, while the client visualizes the resulting map and allows the interaction through a browser. In order to allow the interaction with different abstraction levels of the same information, data underlying the map have to be hierarchically structured.

The aforementioned four modules have been then integrated in the application server side. In particular, the Data Selection module is meant to read and elaborate data from a geographic dataset by allowing expert users to choose the geographic map boundary and the alphanumeric data to represent within the map as tag cloud. The Data Aggregation and Generalization module is meant both to allow expert users to choose the aggregation method and generalize the map boundaries used for the cloud generation. The Cloud Creation module has been designed to create the tag cloud, which has to fit within the simplified map boundaries. Finally, the Data Representation module is able to merge the simplified boundaries and the generated cloud in a hierarchical structure in order to visualize the resulting interactive map.

TaG@Map has been implemented by using various web technologies, such as PHP as programming language, XML and SVG as exchange and storage means, CartoWeb[1] as a comprehensive and ready-to-use Web-GIS, based on the UNM MapServer[2] engine. In particular, the Data Selection module has been implemented as a PHP application which takes as input a geographic dataset, such as an ESRI Shapefile or a Postgres/Postgis table, and allows users to select both the geographic map boundary and the alphanumeric data to elaborate and visualize. The Data Aggregation and Generalization module has been implemented as a PHP application on the server side in order to allow expert users to choose the aggregation method and some configuration parameters. Moreover, it allows to simplify the map boundary by using the well-known Ramer-Douglas-Peucker (RDP) algorithm. The Cloud Creation module has been implemented as a PHP application to generate tag clouds. Differently from [10], where a web service is used to create the tag clouds, an *ad hoc* basic algorithm has been developed to produce them for this prototype. Finally, the Data Representation module has been implemented as a basic CartoWeb client, able both to visualize the resulting map and allow interactive analysis tasks.

In the following Section, the above four operators are exemplified in order to show how they works when embedded into Tag@Map. In particular, some examples of interaction tasks are described by which users are able to navigate data from a general overview to a specific detailed information.

5 The Analysis of Italian Surnames through a Map of Tag Clouds

The following screenshots are taken from the specific scenario illustrated in Figure 1. It shows the distribution of surnames in Italy, where the prominence of each surname is expressed by the font size. By interacting with the interactive map, a philology or a

[1] `http://www.cartoweb.org`
[2] `http://mapserver.org`

linguistic expert user could speed and simplify his/her research activities about the origins of the population who inhabits the territory under investigation.

As previously stated, the creation of this map results from a process which starts from a geographic database containing demographic data related to the Italian territory. In particular, for each Italian Province the dataset contains the associated surnames, the geographic data, and the data about the Region to which it belongs. Such a user-controlled process aggregates the alphanumeric and geographic data for both Region and the whole Italian territory. The result consists of a value which expresses the frequency of surnames at the national level. In particular, a threshold determined by the user (60 for this map) sets the number of surnames to be displayed, which can be tuned in order to obtain a map with a more or less crowded cloud.

In order to refine the analysis, the interface allows the user to zoom-in the map and obtain data about a small portion of territory by performing two different zoom operations, namely geographic and semantic zoom. In particular, the former allows the user to better visualize tags related to a portion of territory in agreement with the common GIS zoom without creating a new cloud. Differently, a Semantic Zoom invokes a new cloud generation referring to the portion of territory on which the user focuses. Figure 2 shows the Northern Italy on which a geographic zoom-in operator is applied.

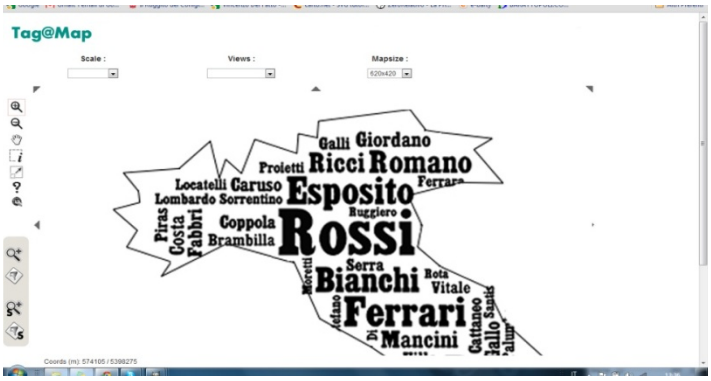


Fig. 2. A map resulting from a geographic zoom-in operation

Figure 3 shows a map resulting from a semantic zoom-in operator applied on four Italian regions selected by the user, namely *Emilia Romagna, Toscana, Marche, Umbria*. Data referring to the selected territory have been aggregated and the corresponding tags have been calculated and embedded within the boundary resulting from the merge of the underlying regions.

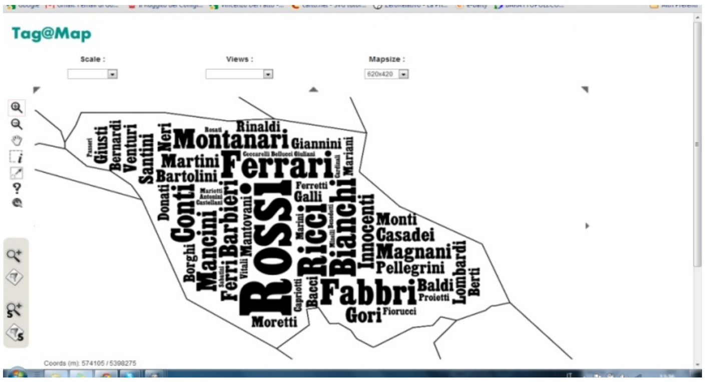


Fig. 3. The map resulting from a semantic zoom-in operation on four Italian regions

A different selection can be performed through a geographic filter. The user can directly choose a given territory among those visualized, and the corresponding tag cloud is generated, accordingly. In this case, by either dragging a selection rectangle on the map or simply pointing on the area of interest, the interface invokes the creation of the cloud corresponding to the selected area. Figure 4 shows the application of a geographic filter which generates the tag cloud of the *Emilia Romagna* region.

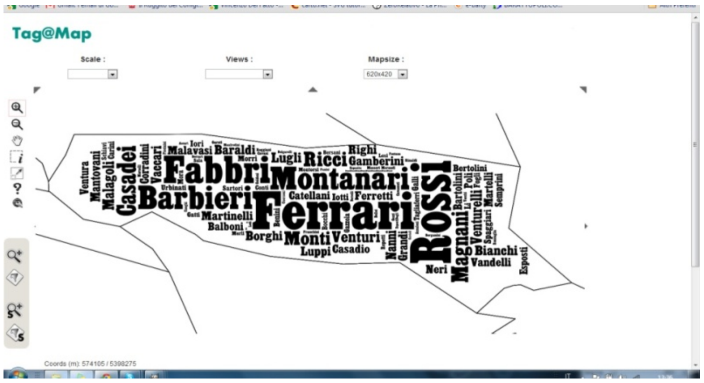
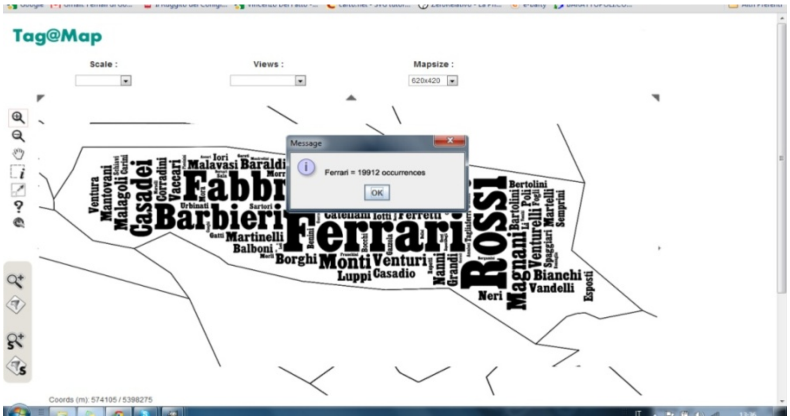


Fig. 4. The map resulting from a geographic filter operation

Finally, the interface allows the user to query the tag cloud in order to obtain details about the selected tag. Figure 5 illustrates the popup window specifying the number of occurrences of the selected tag. It results from the application of the Semantic Filter on the *Ferrari* tag, the most frequent surname in *Emilia Romagna*. By pointing at this tag, the interface invokes a query which performs the requested computation.



Ferrari = 19912 occurrences

Fig. 5. A popup window resulting from a Semantic Filter on the map

6 Final Remarks

In this paper we proposed a GeoVisual Analitycs method meant to support expert users in representing and investigating phenomena of interest. It consists of two main parts, the former is targeted to produce a simplified map containing a georeferenced cloud of tags, in order to provide users with an overview of data distribution and classification, relevant for a territory in a specific domain. The latter is aimed to support users' interaction tasks to detect possible relationships among data underlying the represented scenario. To this aim four operators have been specified capable to focus on, select and query areas of interest. These operators have been implemented within the Tag@Map prototype which satisfies basic characteristics of cloud generation and interaction functionality.

Improvements are currently under investigation. They are targeted to enhance the algorithm underlying the tag cloud generation, in terms of graphics and layout. Visual hints related to the qualitative aspect of tags should be also added to the tag cloud, thus providing further details.

Finally, we aim to provide the interface with wizards in order to customize the available aggregation methods thus supporting unskilled users' interaction.

As future work, we plan to test effectiveness of the system by designing and performing a usability study with potential users, which will provide us with a proper evaluation of the application.

References

1. Paolino, L., Sebillo, M., Tortora, G., Vitiello, G.: Framy – Visualising geographic data on mobile interfaces. J. of Location Based Services 2(3), 236–252 (2008)
2. De Chiara, D., Paolino, L., Romano, M., Sebillo, M., Tortora, G., Vitiello, G.: LINK2U: Connecting Social Network Users through Mobile Interfaces. In: Qiu, G., Lam, K.M., Kiya, H., Xue, X.-Y., Kuo, C.-C.J., Lew, M.S. (eds.) PCM 2010. LNCS, vol. 6298, pp. 583–594. Springer, Heidelberg (2010)

3. De Chiara, D., Del Fatto, V., Laurini, R., Sebillo, M., Vitiello, G.: A Chorem-based Approach for Visually Analyzing Spatial Data. Journal of Visual Languages & Computing (JVLC) 22(3), 173–193 (2011); Special Issue on Visual Analytics and Visual Semantics
4. Brunet, R.: La Carte-modele et les Choremes. Mappemonde 4, 2–6 (1986)
5. Tobler, W.: Geographic Area and Map Projections. The Geographical Review LIII(1), 59–78 (1963)
6. Viegas, F.B., Wattenberg, M., Feinberg, J.: Participatory Visualization with Wordle. IEEE Transactions on Visualization and Computer Graphics 15(6), 1137–1144 (2009)
7. Shneiderman, B.: The Eyes Have It: A Task by Data Type Taxonomy for Information Visualizations. In: Proceedings of the 1996 IEEE Symposium on Visual Languages, pp. 336–343 (1996)
8. Keim, D.A.: Information Visualization and Visual Data Mining. IEEE Transactions on Visualization and Computer Graphics 7(1) (2002)
9. Jaffe, A., Naaman, M., Tassa, T., Davis, M.: Generating Summaries and Visualization for Large Collections of Geo-referenced Photographs. In: Proceedings of the 8th ACM International Workshop on Multimedia Information Retrieval (MIR 2006), pp. 89–98. ACM, New York (2006)
10. De Chiara, D., Del Fatto, V., Sebillo, M.: Visualizing Geographical Information through Tag Clouds. In: De Marco, M., Te'eni, D., Albano, V., Za, S. (eds.) Information Systems: Crossroads for Organization, Management, Accounting and Engineering, ITAIS 2011 (2011) (in press) ISBN: 978-3-7908-2788-0

Clustering User Trajectories to Find Patterns for Social Interaction Applications*

Reinaldo Bezerra Braga[1], Ali Tahir[2], Michela Bertolotto[2], and Hervé Martin[1]

[1] LIG UMR 5217, UJF-Grenoble 1, Grenoble-INP, UPMF-Grenoble 2, CNRS
38400, Grenoble, France
{braga,herve.martin}@imag.fr

[2] School of Computer Science and Informatics, University College Dublin (UCD)
Dublin, Ireland
{ali.tahir,michela.bertolotto}@ucd.ie

Abstract. Sharing of user data has substantially increased over the past few years facilitated by sophisticated Web and mobile applications, including social networks. For instance, users can easily register their trajectories over time based on their daily trips captured with GPS receivers as well as share and relate them with trajectories of other users. Analyzing user trajectories over time can reveal habits and preferences. This information can be used to recommend content to single users or to group users together based on similar trajectories and/or preferences. Recording GPS tracks generates very large amounts of data. Therefore clustering algorithms are required to efficiently analyze such data. In this paper, we focus on investigating ways of efficiently analyzing user trajectories and extracting user preferences from them. We demonstrate an algorithm for clustering user GPS trajectories. In addition, we propose an algorithm to correlate trajectories based on near points between two or more users. The obtained results provided interesting avenues for exploring Location-based Social Network (LBSN) applications.

1 Introduction

Social network platforms have emerged as a collaborative solution to provide social connectivity, giving people the capability to create virtual communities and share interests, opinions, and personal information with other users. However, while there has been an increase in virtual communities, a reduction of social interactions in real communities is evident. We have noticed that social network platforms do not make use of correct context-aware mechanisms in order to improve social contacts in real communities. Therefore, we argue that these

* Research presented in this paper was funded by a Strategic Research Cluster grant (07/SRC/I1168) by Science Foundation Ireland (SFI) under the National Development Plan, the IRCSET Ulysses program, French Ministry of Higher Education and Research, ÉGIDE program and European Cooperation in Science and Technology (COST). The authors gratefully acknowledge this support.

platforms should be based on users' daily routines to increase social interactions among mobile users in real communities.

Nowadays, we have observed a large adoption of smart phones and social networks. As a consequence several mobile social applications have been developed to register social behaviors of mobile users [1] including Ipoki[1], Google Latitude[2], Carticipate[3] and Daily Places[4]. Despite the availability of these mobile social applications to register and share users' daily routines, we face a rapid increase of diverse kinds of space-associated data, such as measurements from mobile sensors, GPS tracks, or georeferenced multimedia. As prospective sources of useful knowledge and information, these data require scalable methods of analysis, which need to consider the particular attributes of the geographical space, such as heterogeneity, diversity of characteristics and relationships, spatio-temporal autocorrelation, and multiple map scales.

Furthermore, recording GPS tracks generates a large amount of data. This data holds spatio-temporal information about a moving object (such as pedestrians, cars, buses, etc.). In order to analyze such data there exists several exploratory as well as data mining techniques. Clustering and aggregation (data mining) techniques have generally been adopted to explore and analyze movement data when visualization (exploratory) techniques are not enough to explore large spatio-temporal datasets. This scenario is also pertinent in case of LBSN applications we have developed.

The purpose of this paper is to explore the capabilities provided by clustering algorithms to analyze user trajectories and extract relevant information from them. We have focused on clustering and aggregating multiples trajectories generated by the same user in order to identify his/her preferences. Once each user preference is identified we apply trajectory correlation algorithm in order to find similarities between multiple user trajectories and near points of interests between two or more users.

To validate our approach, we considered a dataset of trajectories representing a user daily routine (i.e. to go from home to work). We implemented and tested the clustering and trajectory correlation algorithm to understand similarities between users. The results show that our technique is effective in analyzing trajectories datasets and extracting the user preferences. Besides that, the correlation trajectory algorithm is able to effectively find similar PoI between two or more users. Based on the results we envision interesting avenues for social interactions between users.

The rest of this article is organized as follows. To provide the necessary context for our work, we start with the related work in the next section. The proposed architecture, clustering and correlation algorithms are described in Section 3. Section 4 shows experimental results and evaluation we have conducted. Finally, Section 5 presents the conclusions and some directions for future work.

[1] `ipoki.com`
[2] `google.com/latitude`
[3] `carticipate.com`
[4] `dailyplaces.com`

2 Related Work

In general, mobile social applications that implement Mobile Trajectory Based Service (MTBS) consider information about time and space to represent users' trajectories in transportation networks. In [2], the authors present a new strategy to find the fastest route in dynamic transportation networks, making use of previous trajectory information and real-time traffic conditions. Other strategies use the Dijkstra algorithm to solve the same problem in dynamic networks [3]. An important work was proposed in [4], in which the authors introduce a mechanism to model the intelligence of taxi drivers and the properties of dynamic networks to find the fastest route. All these strategies allow the sharing of mobile traces or trajectories to provide a large number of mobile social applications, ranging from a simple navigation mechanism to a robust context-aware and trust-based recommendation system [5].

In spite of the large number of mobile social applications based on context aware information and the adoption of several social networks, some studies show that virtual communities do not increase significantly the amount of social interactions in real communities [6] [7]. Social interactions in the form of user trajectories can generate a huge amount of spatio-temporal data. This can be roughly categorized into a single as well as multiple users trajectories. The former relates to users generating their trajectories over a certain time period, while the latter focuses on group of users interacting socially with their friends and generating their trajectories. In both cases the amount of trajectories produced could be enormous and therefore challenging to interpret for the analysts. Many techniques exist in the literature, however clustering and aggregation techniques are found to be the most suitable for such analysis.

Clustering is a data-mining technique to identify similar and dissimilar groups in a given dataset. The clustering methods however can be classified broadly into partitioning, hierarchical, density-based, grid-based, model-based, constrain-based methods and clustering high-dimensional data [8]. While the overall objective of clustering is the same, they differ based on how they analyze additional parameters such as outliers, noise analysis and dimensions of a given dataset. Each technique can be described in detail with their merits and de-merits. One such study evaluated clustering techniques with focus on trajectory clustering [9].

In our scenario of social interaction application the focus is to find groups with varying density and concentration. For this purpose, density-based clusters are found to be suitable. The main idea is to enlarge a cluster as long as the density of data objects in the neighborhood exceeds a certain threshold value. A typical condition is that for each data point within a cluster, the neighborhood of a given radius has to contain at least a minimum number of points. These methods are quite efficient to find noise and outliers as well as to discover clusters of arbitrary shape. When trajectories are collected in real time, they usually suffer low resolutions of measurements, which make noise tolerance a highly considerable feature [10]. Density-Based Spatial Clustering of Applications with Noise (DBSCAN) [11] and Ordering Points To Identify the Clustering Structure (OPTICS) [12] are widely used density-based clustering methods.

OPTICS has proven effective when offered to trajectory data in some applications [13]. The approach successfully clustered mouse trajectories and obtained good results. An important input to clustering algorithm is an appropriate distance metric. Morris and Trivedi [9] performed an evaluation and discussed distance similarity measures based on fixed length measures (such as Hu Euclidean and PCA subspace) as well as time-normalized distances (also suitable for unequal trajectories length) such as Dynamic Time Warping (DTW), Longest Common Subsequence (LCS) and Modified Hausdorff (MH) (see [9] for an overview of these measures). In case of trajectory data few distance measures have been provided by CommonGIS, a stand-alone visualization tool [14] developed for analysis of movement datasets. They have defined two simple distance methods namely common start and common destination to group trajectories based on their starting and ending points respectively. They also defined two more complex functions called route similarity and route similarity and dynamics. These methods compare two trajectories of unequal length and find the spatial as well as spatio-temporal distance between two trajectories.

3 Our Approach

We present a novel solution in order to increase social interactions by relating daily routines and points of interest based on trajectories of mobile users. For instance, a mobile social application jointly with a social network can answer the following questions: Which of my friends stop in my preferred bakery at the same period of the day? Do any of my friends pass near my apartment to get from their home to their work? Which of my contacts will be passing close to me during the week?[5]

In relation to this we introduce the following 3 definitions to support our discussion.

1. **Road Segment (S)** is defined as a directed link between two extreme points (s_a) and (s_b), composed by a list of intermediate points by using a polyline.
2. **User Trajectory (U_T)** is defined as a set of road segments. Thus, $U_T = \{S_1, S_2, S_3...S_n\}$, where the end point of S_k is the point just before the start point of S_{k+1}, and $(1 \leq k < n)$.
3. **Trajectory (T)** is defined as a set of consecutive points captured through a Global Positioning System (GPS) to one travel performed by the user. Each position (p) is composed of a set of information (latitude, longitude, altitude, direction, time stamp for each registered point (t_p) and an approximate speed provided by the GPS). Since $T = \{p_1, p_2, p_3, ..., p_n\}$, the time interval between two points is computed by the subtraction of $t_{p(k+1)} - t_{p(k)}$, where $(1 \leq k < n)$. Although the points are characterized by latitude, longitude and altitude, we focus on points in 2D space (latitude and longitude) to represent the position of each user.

[5] The user defines the contacts to share his/her daily routine.

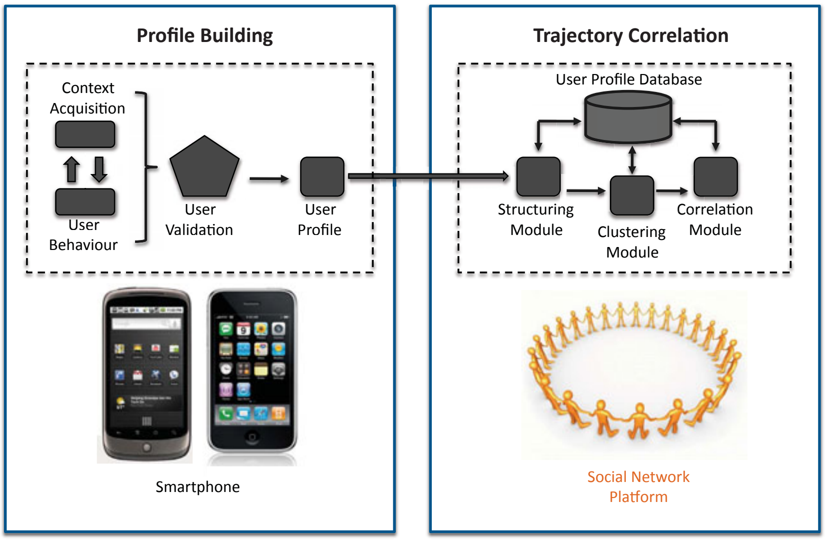


Fig. 1. Architecture overview

Two major components compose our architecture: Profile Building and Trajectory Correlation. Figure 1 presents this architecture. The profile building component should operate in offline mode, but the trajectory correlation components works in online mode. The offline part only needs to be performed once unless the trajectory archive is updated.

As we can observe in the profile building process, users can use a mobile social application to register their trajectory in order to describe their daily routine. After visualizing and validating the trajectory that represents users' daily routine, the user profile is created and the trajectory information is sent to the next step, the structuring module. At this moment, the structuring module verifies if there is a previous trajectory for the same user stored in the database. If there is no trajectory, it creates a new user's daily routine. On the other hand, if multiple trajectories are found, clustering and aggregation techniques can support the analysis to identify the aggregated trajectory (a best representative of user's daily routine). The user daily routine then is enriched with additional information about Points of Interest. Finally, the structuring module exports the enriched information to update the user profile database. These two components are detailed in the next sections.

3.1 Profile Building

The user profile can be designed taking into account two basic types of data that are used for constructing and enriching the profile model. These two basic types are defined as *personal* and *contextual* data. Personal data describes the main details of an entity and the contextual data characterizes the situation.

An entity can be a person, place, physical or computational object. For example, in a personal tracking application for mobile users, the personal data would be the information about the user, such as name, birthday, gender, etc. On the other hand, contextual data would be composed of movement records that the user performed over a period of time. A movement record can include such characteristics as the initial point, speed, direction, and time, as well as weather information. We define an entity as a mobile user using a smart phone equipped with GPS, digital camera and Internet connection (e.g. 3G or Edge).

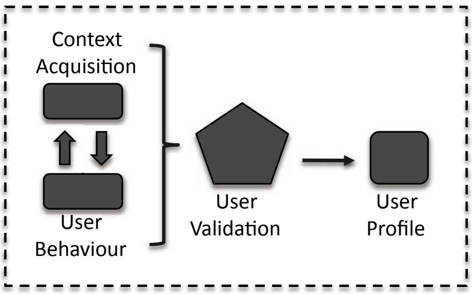


Fig. 2. The profile building process

In addition, we use a third type of data, which is named *behavioral* data. Behavioral data is defined according to specifications for representing Points of Interest (POI) of the users, which has been developed by the W3C Points of Interest Working Group Charter [15]. This Working Group has defined specifications for Points of Interest data that can be used in a large number of applications, such as augmented reality browsers, geo-caching and games, mapping and navigation systems, and many others. The behavioral data describes the behavior of the users learned from their daily routines. One way to define user behavior is with a set of conjunctive rules, such as classification or association rules. Some examples of rules describing user behavior are: “When user Hervé goes from work to his residence, he usually stops at the bakery”, “Every Monday Carina goes from her work to the tennis court at 13:00 and comes back to her work at 14:30”, “Whenever user Reinaldo goes from his residence to his office, he stops in the Residence Matisse at 08:00 to take his friends to work”. The use of rules in profiles offers a perceptive, descriptive and modular way to characterize user behavior and was presented in[16].

The rules can be either determined by specialists or derived from transactional data of a user, making use of clustering algorithms or machine learning techniques. Since we consider mobile social applications in the profile building process, our rule discovery method is used individually to the transactional data of each user, capturing and comparing personal behaviors. Hence, the rules are discovered using a clustering algorithm in multiple user trajectories.

3.2 Clustering Algorithm

We have adapted OPTICS [12] clustering algorithm, which produces an ordering of a dataset while storing the core distance and a suitable reachability distance of each user trajectory. OPTICS provides information about the overall clustering structure unlike other method that computes a flat partitioning of data (such as K-means [17]). A brief overview of OPTICS is presented with the help of underlying terminologies. Assume ρ = object from a dataset D, ε = distance threshold, $N\varepsilon$ (ρ) = ε-neighborhood of object ρ, *minPts* = natural number, *minPts-distance*(ρ) = distance from ρ to its *minPts* neighbor. The core distance (CD) is defined as:

$$CD = \begin{cases} Undefined, & \text{if } Card(N\varepsilon(\rho)) < minPts \\ minPts\text{-}distance(\rho), & \text{otherwise} \end{cases}$$

Thus, the *core distance* is the smallest distance ε between ρ and an object in its ε-neighborhood such that ρ would be a core object. The *core distance* is *Undefined*, otherwise. For reachability distance, assume ρ and o = objects from a dataset D, $N\varepsilon$ (o) = ε-neighborhood of object o, *minPts* = natural number. The reachability distance (RD) of ρ with respect to o is defined as:

$$RD = \begin{cases} \textit{Undefined}, & \text{if } |(N\varepsilon(o))| < minPts \\ \textit{max(core-distance}(o), \textit{distance}(o,\rho)), & \text{otherwise} \end{cases}$$

Thus, the reachability distance of ρ is the smallest distance such that ρ is directly density-reachable from a core object o. Otherwise, if o is not a core object, even at the generating distance ε, the reachability distance of ρ with respect to o is *Undefined*.

OPTICS produces a reachability plot that shows the cluster ordering and the reachability values. The reachability plot gives a graphical view of the structure of the data by providing data independent visualization. From the output plot, clustering can be obtained by choosing an appropriate threshold value of reachability distances. There are automatic techniques available to identify clusters from this plot, which is applicable when the dataset is very large. Figure 3 illustrates cluster ordering with the help of a reachability plot showing valleys to identify potential clusters. Two additional parameters are of significant importance in OPTICS algorithm (maximum distance threshold and minimum number of neighbors). As Ankerst et al.[12] suggest the distance threshold influences the number of clustering levels, which can be seen in a reachability plot. The smaller the distance, the more objects may have undefined reachability distances. Therefore, the clusters with lower density might be less visible and hence this situation should be prevented. Similarly, the larger minimum neighbor value will yield better results.

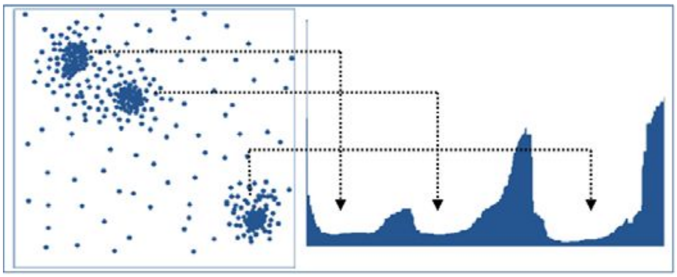


Fig. 3. A reachability plot showing data densities and respective clusters [12]

3.3 Trajectory Correlation

Taking into account the idea to analyze user's daily routines in order to increase the number of social interactions between users, we propose an optimized algorithm based on Minimum Bounding Rectangles (MBR) [18] and the Hausdorff distance [19]. Firstly, we identify four extreme points of each trajectory (the northernmost, the southernmost, the westernmost and the easternmost). With these points, we create the MBR for the users' trajectories.

The Hausdorff distance is often used to determine the similarity of two shapes [20] and to measure errors for approximating a surface in generating a triangular mesh [21]. In our approach, we are interested to use Hausdorff distance computation in two different cases. Basically, the first case is applied when the algorithm finds a correlated area between two MBRs. It uses Hausdorff distance to compute the distance between the points that are in the correlated area. On the other hand, if there is no correlated area, the Hausdorff distance computation is used to compute the distance of near points between two MBRs. When the distance of two MBRs is found, the algorithm allows the expansion of both MBRs in order to find one or more points of social interactions, taking into account a threshold (D_{max}) for the expansion. The trajectory correlation is executed according to the algorithm as follows.

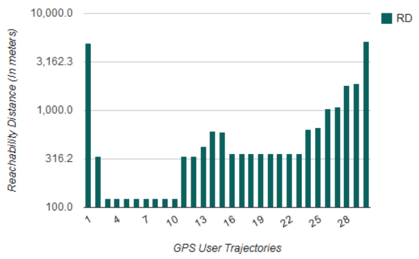
Algorithm 1. Main algorithm

```
if ($Lat_{max(A)} < Lat_{min(B)}$) or ($Lat_{max(B)} < Lat_{min(A)}$) or ($Lon_{max(A)} < Lon_{min(B)}$) or ($Lon_{max(B)} < Lon_{min(A)}$) then
    Execute HausDist of MBR(A) and MBR(B);
    if HausDist $< D_{max}$ then
        Expand MBRs;
    else
        There is no correlated area;
        Stop main algorithm;
    end if
end if
Select correlated area;
Execute HausDist;
```

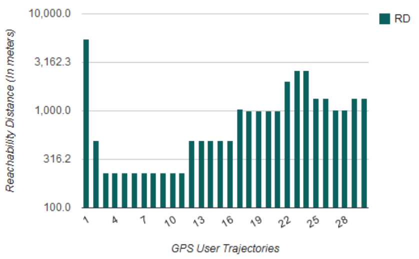
The Hausdorff distance from MBR(A) to MBR(B) can be determined by exploiting the characteristic that for each MBR face, there has to be at least one object that touches it. Therefore, we identify the face in MBR(A) closest to a face in MBR(B). After that, the algorithm computes the Hausdorff distance of these two faces and compares the result with D_{max}. If Hausdorff distance is less than D_{max}, then both MBRs expand their related faces from the current distance to the result of D_{max}. Once the correlated area of MBRs is found, the main algorithm executes the Hausdorff distance computation of the points.

Our approach is able to identify a correlated area of near points. In addition, it optimizes the Hausdorff distance computation owing to selection of points in the correlated area. This avoids the execution of the distance computation for all points in the trajectory.

Making use of context information, our approach allows the identification of segments S, which can be represented by landmark graphs. This information could be used to increase social interaction. For example, we can capture context information in order to send a message to users, alerting that a friend passes in front of a specific number of the street X all the weekdays between 10:00 AM and 10:30 AM. This message can also contain accurate information of distance, which is acquired by the Hausdorff distance algorithm.



(a) User 1 (ϵ = 1000 & minNbs = 3).



(b) User 2 (ϵ = 1000 & minNbs = 3).

Fig. 4. Reachability plots showing clustering structure

As an additional feature, the trajectory correlation module enables the generation of a message based on the context information. It reads all the fields related to a correlated point in order to automatically create the message that will be sent to one or both users.

4 Results and Discussion

To demonstrate our concept we have applied our approach to two separate users based on their registered trajectories. The overall approach can be summarized in three steps. First of all clustering is applied to individual user trajectories over a period of one month. A typical user route is a trajectory from home to work. After obtaining distinct groups an aggregated trajectory has to be chosen.

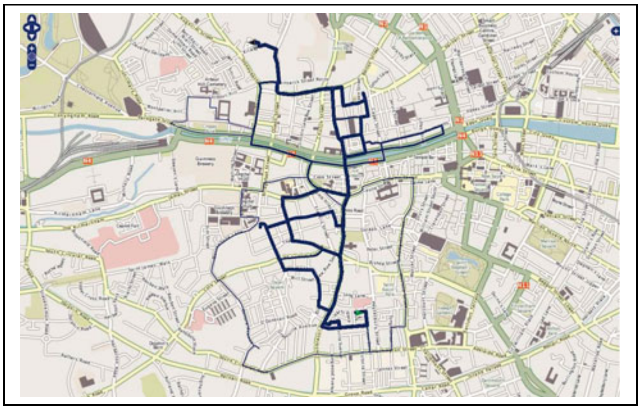


Fig. 5. Three clusters showing distinct routes of User 1 (overlay on map)

With the help of visualization and aggregation techniques, a best representative trajectory for each user is obtained. This aggregated trajectory obtained from several user trajectories is then compared to other users by applying our trajectory correlation algorithm. This will enable groups of users to share similar routes to increase geospatial social interaction. We now explain the different input parameters we have used in order to verify the results.

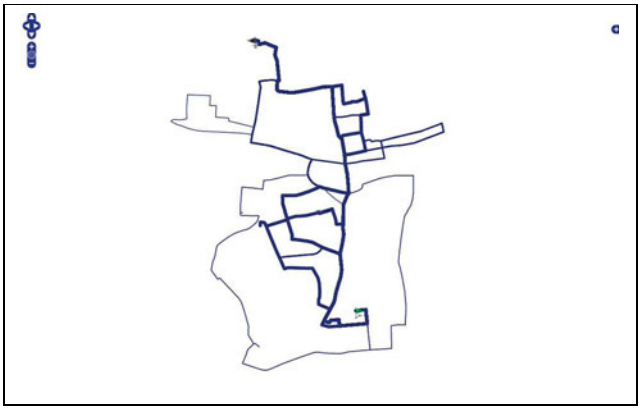


Fig. 6. Three clusters showing distinct routes of User 1 (without overlay)

OPTICS clustering algorithm requires two input parameters: distance threshold (ϵ) and minimum neighbors *(minNbs)*. The authors of OPTICS [12] suggest that the value of these two parameters have to be large enough to yield good results. We structured our experiment in a way that we choose a range of distance threshold values as well as minimum neighbors. For our scenario, we defined the distance threshold between 1000 meters and 15000 meters $\Rightarrow$ *($1000 \leq \epsilon \leq 15000$).*

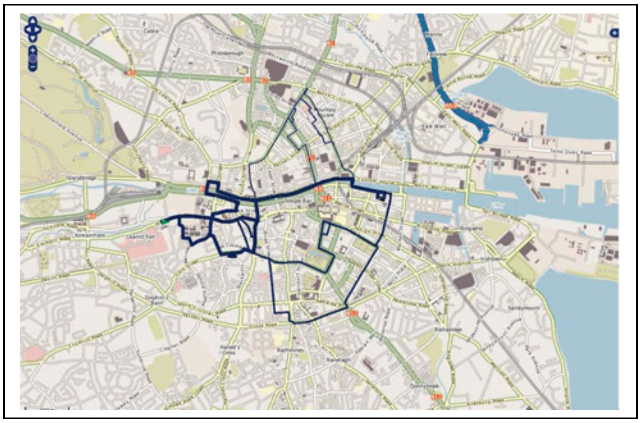


Fig. 7. Three clusters showing distinct routes of User 2 (overlay on map)

Similarly, for minimum neighbors we selected a value of 1 up to 10 $\Rightarrow$ *(1 $\leq$ minNbs $\leq$ 10)*. The experiment was run with a combination of values for both parameters. Based on the statistics and a range of reachability plots we obtained, we found the best combination of values $\Rightarrow$ *(ϵ = 1000 & minNbs = 3)*. This condition revealed a satisfactory result in terms of the clustering structure from the reachability plots.

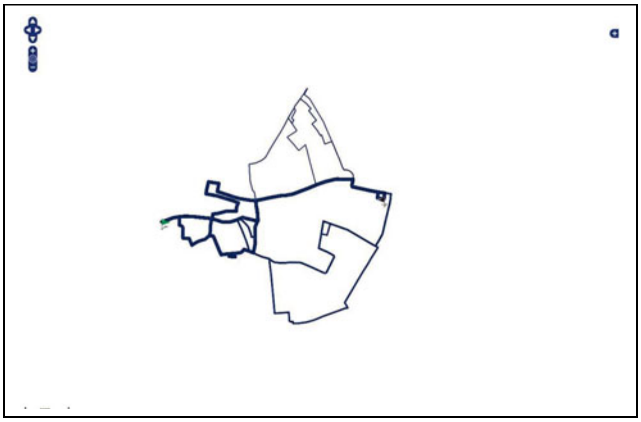


Fig. 8. Three clusters showing distinct routes of User 2 (without overlay)

The reachability plots obtained are illustrated in Figures 4(a) and 4(b). The plots show re-ordering of objects (trajectories in the dataset) on x-axis while y-axis demonstrates the reachability distances between trajectories. Automatic cluster extraction techniques from a graph were presented in [12][22]. This data independent visualization provides analysts a high-level understanding of clustering structure. From these graphs clusters can be identified based on Gaussian-bumps or valleys. As a general rule the cluster starts from a steep-down area and ends at a steep-up area.

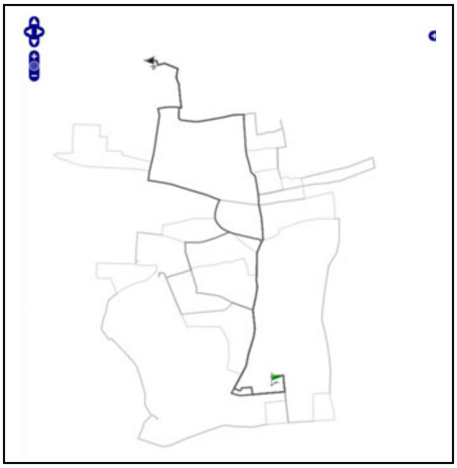


Fig. 9. Best representative aggregated user trajectories (user 1)

Based on the first plot in Figure 4(a), we can clearly see that there are two dominant clusters in user trajectories (trajectory 2 to 13 and trajectory 14 to 25) shown by the valleys in the plot. The other cluster is a group of trajectories, which does not specifically form a valley however they are grouped together into one cluster. The second graph (see Figure 4(b)) also shows three clusters with varying cardinalities (trajectory 2 to 16, 17 to 22 and 23 to 30). In both the graphs, the first trajectory is considered as noise (see OPTICS algorithm [12]). In Figures 5, 6, 7 and 8, the three clusters (from both graphs) are drawn in different styles. The representative routes for each cluster are drawn with different thickness for visualisation purposes.

The clusters show three distinct routes both users adopted over a period of one month to travel from home to work. On average each user trajectory contains almost 100 points. The clustering structure also forms distinct groups based on a specific route on a specific day of the month. For example in Figures 5 and 6, cluster 2 holds trajectories starting from trajectory *14* to trajectory *25* that include *11* days routes. For this specific case we can acquire knowledge about the patterns related with a particular day of a week or a month. For example, if we observe the order in which the trajectories were recorded in case of cluster 2 we obtain *(1,2,3,4,7,8,9,12,13,14,15)*. We can apply heuristics and visualization techniques such as heat maps in order to gain more insights into user behaviors. As apparent from the above sequence user 1 always follows a similar or close

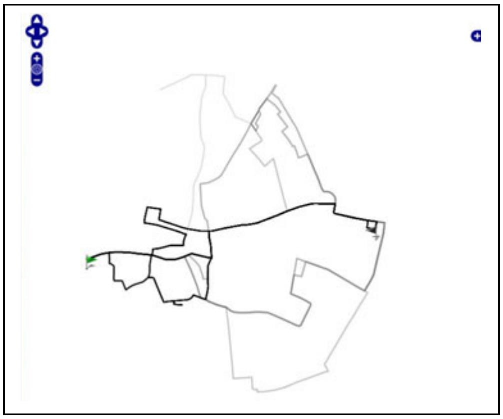


Fig. 10. Best representative aggregated user trajectories (user 2)

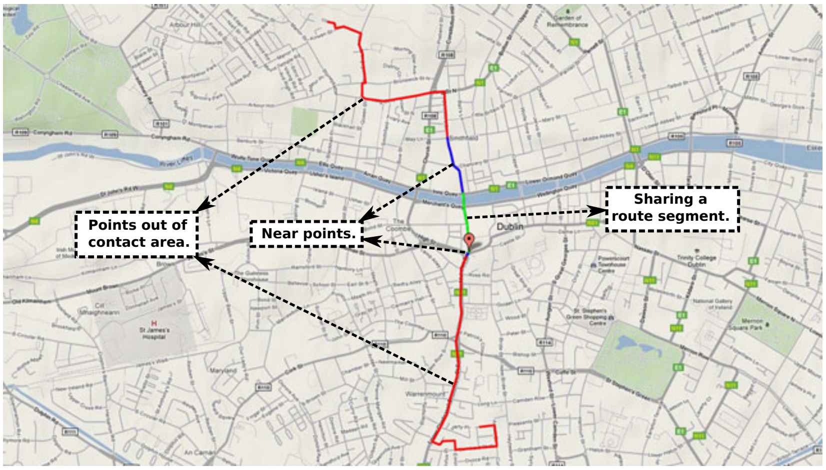


Fig. 11. Best representative trajectory of user 1 in comparison to user 2

route during at least three consecutive days of a month such as *(1,2,3)*, *(7,8,9)* and *(13,14,15)*.

After analyzing the clustering structure the next step is to find an aggregated trajectory or a best representative of a particular user route. For this purpose we have applied a simple yet interesting visualization technique. When all three clusters from both users are visualized using a single grey scale color scheme, it reveals the most frequent route adopted. The color has to be selected in a way that it must be transparent enough to visualize these changes. The phenomenon is illustrated in Figures 9 and 10, where user 1 and user 2 best representatives can be visualized and extracted respectively for further analysis.

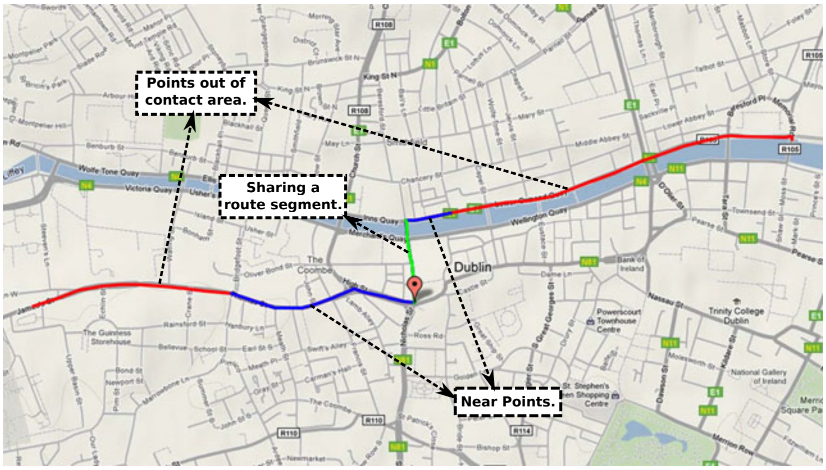


Fig. 12. Best representative trajectory of user 2 in comparison to user 1

Once the clustering algorithm recognizes the best representative trajectory for each user, the trajectory correlation algorithm is executed. For this example, the algorithm firstly generates the MBRs for each best representative user trajectory and identifies the correlation between both MBRs. After that, it computes the Hausdorff distance of the points in the correlated area.

In order to present the accuracy and efficiency of our system we used a color-based scheme to represent the points in the same road segment, the near points and the points out of the correlated area. Figures 11 and 12 show the trajectory of the users 1 and 2 respectively with the colors representing the near points between them. The green color represents the same segment that is used by both users for their daily routines. The blue color denotes the possible points of interaction, which is in the correlated area among the MBRs. Finally, the red color indicates the points that are out of the correlated area. Additionally, the system allows the generation of messages making use of the context information.

Making use of the presented results and taking into account the use of context information to describe Points of Interests (PoI), we conclude that our approach can be applied to a large number of applications, for instance: to offer a system that increases social interactions in real communities; to develop a system that encourages rides among friends (car pooling).

5 Conclusion and Future Work

Virtual community platforms provide solutions to social connectivity, giving people the capability to share interests, opinions, and personal information with other users. Nevertheless, due to the reduction of social connections in real communities and the absence of context-aware mechanisms in virtual communities, social interactions are frequently missed. As a solution, the users' daily routines, can be captured by mobile social applications and shared in virtual communities in order to improve the social connections in real communities.

This paper presents an approach to explore the capabilities provided by clustering algorithms to analyze user trajectories and extract relevant information from them. We focused on clustering and aggregating multiples trajectories generated by the same user in order to identify habits or preferences. We introduced our trajectory correlation algorithm to find similarities between multiple user trajectories based on each user preference and PoI. Consequently, the near points of interests between two or more users are identified. Taking into account the obtained results, we conclude that our research provided interesting avenues for exploring Location-based Social Network (LBSN) applications.

As future work, we intend to evaluate our algorithm with the MBR expansion process. Besides that, we also aim to use a data-mining algorithm implemented in mobile devices. Therefore, the device allows the trajectory analysis, comparing the current rule with previous rules in order to propose a new personal rule. By using a suitable data-mining algorithm, we can infer the time estimation for a specific segment. Finally, we intend to offer a framework for the development of context-aware systems based on trajectory correlation, focusing on the impact of sharing trajectory information in online social networks as well as their privacy implications [23]. This framework will provide a collection of procedures to acquire, store, increase and infer contextual metadata related to the near points in the correlated area. Additionally, we aim to reuse our techniques in different types of scenarios (for example car pooling and tourism related applications). Finally, in this paper we did not take privacy issues into account; however, these will have to be considered if the application is deployed commercially.

References

[1] Braga, R.B., Martin, H.: Captain: A context-aware system based on personal tracking. In: The 17th International Conference on Distributed Multimedia Systems / DMS 2011. KSI, Florence (2011)

[2] Wu, Q., Huang, B., Tay, R.: Adaptive Path Finding for Moving Objects. In: Li, K.-J., Vangenot, C. (eds.) W2GIS 2005. LNCS, vol. 3833, pp. 155–167. Springer, Heidelberg (2005)

[3] Pfoser, D., Brakatsoulas, S., Brosch, P., Umlauft, M., Tryfona, N., Tsironis, G.: Dynamic travel time provision for road networks. In: The 16th ACM SIGSPATIAL International Conference on Advances in Geographic Information Systems, GIS 2008, pp. 68:1–68:4. ACM, New York (2008)

[4] Yuan, J., Zheng, Y., Zhang, C., Xie, W., Xie, X., Sun, G., Huang, Y.: T-drive: driving directions based on taxi trajectories. In: The 18th SIGSPATIAL International Conference on Advances in Geographic Information Systems, GIS 2010, pp. 99–108. ACM, New York (2010)

[5] Andersen, R., Borgs, C., Chayes, J., Feige, U., Flaxman, A., Kalai, A., Mirrokni, V., Tennenholtz, M.: Trust-based recommendation systems: an axiomatic approach. In: The 17th International Conference on World Wide Web, WWW 2008, pp. 199–208. ACM, New York (2008)

[6] Cavanagh, A.: From culture to connection: Internet community studies. Sociology Compass 3, 1–15 (2009)
[7] Online Conference on Networks and Communities: Are virtual communities a good thing socially? (2010), `http://networkconference.netstudies.org` (last access: October 27, 2011)
[8] Han, J.: Data Mining: Concepts and Techniques. Morgan Kaufmann Publishers Inc., San Francisco (2005)
[9] Morris, B., Trivedi, M.: Learning trajectory patterns by clustering: Experimental studies and comparative evaluation. In: IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2009, pp. 312–319 (2009)
[10] Rinzivillo, S., Pedreschi, D., Nanni, M., Giannotti, F., Andrienko, N., Andrienko, G.: Visually driven analysis of movement data by progressive clustering. Information Visualization 7, 225–239 (2008)
[11] Ester, M., Kriegel, H.P., Sander, J., Xu, X.: A density-based algorithm for discovering clusters in large spatial databases with noise. In: The 2nd International Conference on Knowledge Discovery and, pp. 226–231 (1996)
[12] Ankerst, M., Breunig, M.M., Kriegel, H.P., Sander, J.: OPTICS: Ordering Points to Identify the Clustering Structure. SIGMOD Rec. 28, 49–60 (1999)
[13] Tahir, G., McArdle, M.B.: Visualising user interaction history to identify web map usage patterns. In: 14th AGILE International Conference on Geographic Information Science, Advancing Geoinformation Science for a Changing World, Utrecht, The Netherlands (2011)
[14] Andrienko, G., Andrienko, N., Wrobel, S.: Visual analytics tools for analysis of movement data. SIGKDD Explorations Newsletter - Special Issue on Visual Analytics 9, 38–46 (2007)
[15] Points of Interest Working Group: W3c points of interest working group charter (2011), `http://www.w3.org/2010/POI/charter/` (last access: October 27, 2011)
[16] Benevenuto, F., Rodrigues, T., Cha, M., Almeida, V.: Characterizing user behavior in online social networks. In: The 9th ACM SIGCOMM Conference on Internet Measurement, IMC 2009, pp. 49–62. ACM, New York (2009)
[17] MacQueen, J.B.: Some methods for classification and analysis of multivariate observations. In: Cam, L.M.L., Neyman, J. (eds.) The 5th Berkeley Symposium on Mathematical Statistics and Probability, vol. 1, pp. 281–297. University of California Press (1967)
[18] Papadias, D., Sellis, T., Theodoridis, Y., Egenhofer, M.J.: Topological relations in the world of minimum bounding rectangles: a study with r-trees. In: ACM SIGMOD International Conference on Management of Data, vol. 24, pp. 92–103 (1995)
[19] Atallah, M.J.: A linear time algorithm for the hausdorff distance between convex polygons. Informatics Processing Letters 17, 207–209 (1983)
[20] Jacox, E.H., Samet, H.: Metric space similarity joins. ACM Transaction on Database Systems 33, 7:1–7:38 (2008)
[21] Bischoff, S., Pavic, D., Kobbelt, L.: Automatic restoration of polygon models. ACM Transactions on Graphics 24, 1332–1352 (2005)
[22] Brecheisen, S., Kriegel, H., Kröger, P., Pfeifle, M.: Visually mining through cluster hierarchies. In: International Conference on Data Mining, Citeseer, Orlando, FL (2004)
[23] Gross, R., Acquisti, A.: Information revelation and privacy in online social networks. In: Proceedings of the 2005 ACM Workshop on Privacy in the Electronic Society, WPES 2005, pp. 71–80. ACM, New York (2005)

Making a Pictorial and Verbal Travel Trace from a GPS Trace

Pablo Martinez Lerin, Daisuke Yamamoto, and Naohisa Takahashi

Dept. of Computer Science and Engineering,
Nagoya Institute of Technology,
Gokisocho, Showaku, Nagoya 466-8555, Japan
{pablo,daisuke,naohisa}@moss.elcom.nitech.ac.jp

Abstract. This paper presents a system for making a pictorial and verbal representation of a travel trace from a GPS trace with the following functions:

(1) A function to make a travel trace consisting of following paths and off-road paths, which indicate the turns and the reached districts.
(2) A function that generates a partonomic tree that contains the sections of the travel trace, i.e. portions of the paths of the travel trace associated with verbal information, such as toponyms, route names, and district names.
(3) A function that generates clickable labels that represent the sections of a travel trace in a map view. By using the labels, users can obtain detailed information while using a small map scale, understand the part-of relations among the administrative regions represented, and intuitively navigate the representation based on the verbalized sections.

We have developed a prototype of the proposed system for a web-based interactive map service, which receives either a KML or a GPX file with a GPS trace, and provides a pictorial and verbal travel trace representation on a focus+glue+context map developed in our previous work. Using the prototype, we have confirmed the feasibility of the proposed system.

Keywords: GIS, GPS, Web mapping, Focus+Glue+Context, travel trace, route labeling.

1 Introduction

The widespread use of GPS-enabled mobile devices and attractive GIS applications encourages travelers to log their GPS traces while travelling. A GPS trace is a sequence of logged GPS points that forms the travel trace of a trip. Travelers use GPS traces to store and share information about a trip, because GPS traces are easy to log and can be loaded in many GIS applications using standard file formats [1][2]. Users load and examine in GIS applications the GPS trace of a past trip for several purposes: recall a past trip, understand someone else's trip, plan a future trip based on a past trip, and collect statistics about one or several trips (e.g. how popular is a route when travelling from city A to city B?).

Several GIS applications, such as Google Earth and Global Mapper [3][4], draw a GPS trace as a line that follows GPS points over a map view, and display it on a screen. The map view is composed of raster map images, which are built beforehand for each map scale. In order to understand a trip in more detail, users can retrieve a map view by changing the map scale and position (zooming and panning), and relate the line of the travel trace with images, labels, and shapes of the map.

The representations of the map image and the line of the travel trace are useful, but often cause the following problems:

- Because the label of a region (e.g. city) is often in the center of the region, it may fall outside the map view, while the trace within the region is represented inside the map view. This problem occurs especially in mobile devices with small screens.
- When the labels of small regions, like wards, are placed in a map image of a large scale, the labels of larger regions, like prefectures, might not be placed in the map image. On the other hand, when the labels of the larger regions are placed, the small roads and parks within a smaller region are not represented.

As a result, users are required to use many map operations like zooming and panning. Obtaining detailed information in large trips is a hard and time-consuming task, because users must examine each part of the trip using a large map scale.

Moreover, the points of a GPS trace are often not enough to provide a rich interaction with users, because they are not structured and are associated only with numerical values, like latitude and longitude. As a result, users cannot easily specify a portion of the trip, for example the portion within a park, or the portion within a city. Users might want to select portions of the map, in order to apply visualization operations (e.g. adjust the map view to a portion and highlight a portion), and editing operations (e.g. copy, replace, and remove a portion).

To solve the above-mentioned problems, we propose a system that generates a pictorial and verbal representation of a travel trace that matches a GPS trace. The proposed system performs the following three main tasks:

- It divides a travel trace into multiple smaller portions, called sections, and associates them with road networks and districts in a map, by finding off-road paths and following paths in a travel trace from a GPS trace. As a result, it enables users to specify sections of a travel trace using map data, such as addresses, route names, district names, and others, i.e. to verbalize the sections of a travel trace.
- It constructs a partonomic tree that represents part-of relations among administrative regions and the sections of the travel trace. Using the tree makes it easy and fast to find the sections within the administrative regions.
- It generates clickable labels representing sections that are determined according to the scale and bounds of the map view. Labels are organized in piles that clearly represent (1) labels that describe the whole map view, and (2) part-of relations among the administrative regions reached. The labels describe all the sections within the map view, even very small sections, and the overlapping of labels is reduced by compounding and replacing related labels by representative icons. Clicking

labels allows users to intuitively navigate through the representation based on the verbalized sections, and clicking on icons allows users to decompose and see the replaced labels.

2 Related Work

Several research papers focus on the verbal representation of maps [5][6]. However, they do not address the following subjects: the locations in a travel trace that require verbal representation, and the kinds of verbal data that should be related to locations.

Map matching has been investigated in numerous studies [7][8]. Our proposed map-matching algorithm differs from others because it identifies elements of a travel trace, such as turns and off-road paths, which is required to verbalize later sections of the trace.

Several systems use labels and icons over maps and implement methods to avoid overlapping. In GoogleMaps, the verbal information (driving instructions) associated to a travel route is represented in a panel next to the map. When users click an instruction, a label containing the instruction is placed on the map [9]. A method labels public transit lines (paths) and avoids overlapping by changing the size of the labels [10]. Another method creates a sketch-route map where the roads are scaled and simplified, so that they can be easily labeled [11]. Our proposed method is different, because it displays all the information on the map, it organizes the labels, and it represents several kinds of information (routes, districts, regions, and turns), as opposed to labeling only roads or driving instructions.

In previous studies, we have proposed a method to generate icons that represent compound landmarks [12]. The method presented in this paper is different, because it compounds labeled sections and provides icons with interactions.

3 Pictorial and Verbal Representation

A pictorial and verbal travel-trace representation consists of (1) a line that represents the shape of the travel trace, and (2) a set of clickable labels that represent verbal information of the trace; both displayed on a map view.

In addition, the labels are associated with portions of the travel trace, called sections. For example, the label “Tokyo” is associated with the sections within Tokyo. The association is used to provide the following interaction: when users click on a label, the system increases or decreases the map scale to adjust the map view to the associated sections, i.e. the associated sections are displayed as big as possible within the map view. As a result, clickable labels are used as follows:

- By reading the labels, users find and understand information about the travel trace.
- By clicking the labels, users navigate through the representation.

Labels represent the following verbal information of a travel trace: name of the administrative regions reached (countries, prefectures, cities, and wards), name of the routes used, name of the districts reached (parks, gardens, squares, etc), and name of the intersections where the trace turns (reorientation point). The following paths and off-road paths of a travel trace are useful to identify the turns and the districts reached.

The proposed system produces a clear and organized representation fitted for the map view, defined by its map scale and bounds. To provide a fast response when users modify the map view, the proposed system generates the pictorial and verbal representations from a loaded GPS trace in two main steps described below.

Main step 1. Each time a new GPS trace is loaded:

- o The following paths and off-road paths of the travel trace are computed from the GPS trace.
- o The paths are divided in sections and the sections are organized in a tree.

Main step 2. Each time the map view is modified:

- o Labels are generated by using the tree created in Main step 1.

The following three chapters describe in detail the three points of the two main steps described above.

4 Finding Following Paths

4.1 Overview

A travel trace is composed of following paths and off-road paths as shown in Fig. 1. The intersections between following paths indicate the turns of the trace and the off-road paths indicate the districts (parks, gardens, squares, etc) reached by the trace. This chapter describes the method for finding the following paths and off-road paths of a travel trace represented by a GPS trace. The important data structures used in this method are defined below.

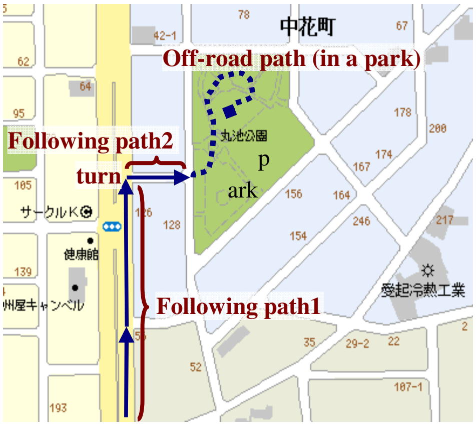


Fig. 1. Example of a travel trace

GPS Trace. A GPS trace *GT*, is a sequence of time-stamped points, i.e., $GT = (p_1, p_2, \ldots, p_n)$. A point p_i, for $i = 1, 2, \ldots$, n, is a GPS point that has three fields, namely latitude (p_i.lat), longitude (p_i.lng,), and time (p_i.t), where $\forall\ 1 \le i < n$, $p_{i+1}.t > p_i.t$.

Road Network. A road network is a graph *R(V,E),* where *V* is a set of vertices representing the intersections and terminal points of the roads, and *E* is a set of directed edges representing the links of the roads, i.e. the sections of the roads between intersections and terminal points.

Link. A link *L* is a section of road between two intersections or terminal points, composed by its id and shape points, i.e. *L* = (*id*, *shapePoints*). *shapePoints* is a set of points (latitude, longitude) that define the shape of the link, i.e. *shapePoints* = ($sp_1, sp_2, \ldots, sp_n$). Two connected links share a single shape point, which is the intersection of the links, as shown in Fig. 2.

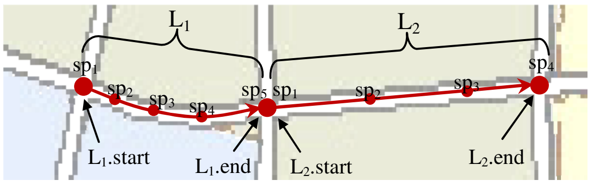


Fig. 2. Representation of two connected links, L_1 and L_2

Following Path. A following path *FPath*, is a sequence of connected links in a road network that do not form turns, i.e. $FPath = (L_1, L_2 \ldots, L_n)\ \forall\ 1 \le i < n$, and L_{i+1}.start = L_i.end. In other words, a following path is a path that a human would follow by travelling straight and not turning at intersections.

Off-Road Path. An off-road path *OFFPath,* is a sequence of consecutive points from a GPS trace that are off-road points, i.e. $OFFPath = (p_1, p_2 \ldots, p_n)$. An off-road point is a point that does not lie on any link in a road network.

Following paths and off-road paths are computed as follows:

Step1 [Refine GPS Trace]. The points of the GPS trace are refined according to a distance-based algorithm as follows. Incrementally consider each point in the GPS trace, and remove a point whose distance to its predecessor point is smaller than a predefined threshold value. The prototype system uses a threshold value of 10 meters.

Step2 [Find Start of Following Path]. Rank the refined GPS points in temporal order and for each point find the start of a following path, i.e., the GPS point that lies on a link of a road network. The path of consecutive GPS points that do not lie on links of the road network is considered an off-road path.

Step3 [MakeFollowingPaths]. For each refined GPS point ranked in temporal order and not yet considered, find following paths from the start found in step2 that contain the GPS points. When the path cannot be followed by links, end this step.

Step4 [Repeat Step2-3]. Repeat Step2 and Step3 until all refined GPS points have been considered.

Steps 2 and 3 are described in more detail in the following subchapters.

4.2 Finding the Start of a Following Path

Two measures, Proximity and Orientation, are used to decide whether a GPS point of a GPS trace lies on a link of a road network. The method to combine both measures is a simplification of the method proposed by Greenfeld [13].

Proximity(*p*, *L*): Given a point *p* and a link *L*, when *L* and the perpendicular line from *p* to *L* intersect with *L* as shown in Fig. 3(a), Proximity(*p,L*) is defined as the Euclidean distance between *p* and *L*. If the perpendicular line from *p* to *L* does not intersect with *L*, Proximity(*p,L*) is defined as the Euclidean distance between *p* and the closest endpoint of *L*, as shown in Fig. 3(b).

Orientation(*p,L*): Given a point *p* and a link *L*, Orientation(*p,L*) is defined as the angle between *L* and a vector formed by *p* and the successor point of *p*, denoted as succ(*p*), as shown in Fig. 4.

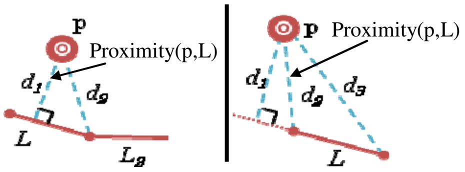


Fig. 3. Two examples of the Proximity measure

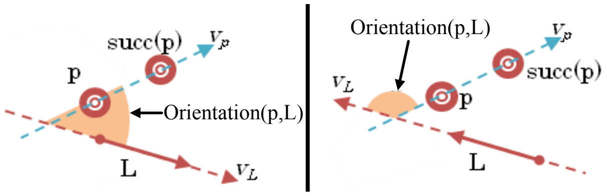


Fig. 4. Two examples of the Orientation measure

The algorithm to find the start of a following path is described below. The algorithm uses the following system parameters: (1) a threshold distance, *dmax*, (2) a threshold of the Proximity, *dt*, and (3) a threshold of the Orientation, *at*.

Input: a road network *R*, and a sequence of GPS points of a GPS trace not yet considered, *sPoints* = $(p_1, p_2, \ldots, p_n)$.
Output: a link, an off-road path

```
    OFFPath = create a new off-road path
    Add OFFPath to the solution.
    For each point $p_i$ in sPoints perform the following steps:
        Step1. Load a set of links, $S_1$, within a rectangle R1 from a database server for
               the road network R, where R1 is a minimum bounding rectangle of the
               circle whose center and radius are $p_i$ and $d_{max}$, respectively.
        Step2. Set $S_2 = \{L \mid \text{Orientation}(p_i, p_{i+1}, L) \leq \alpha_t, L \in S_1\}$.
        Step3. Set $S_3 = \{L \mid \text{Proximity}(p, L) \leq dt, L \in S_2\}$.
        Step4. If $S_3$ is empty, add $p_i$ to OFFPath and skip steps 5 and 6.
        Step5. Set $L = \text{argmin Proximity}(p, L_i)$, where $L_i \in S_3$.
        **Step6.** Return $L$.
    EndFor
```

Our prototype system uses the following values for the system parameters: d_{max} = 100 meters, α_t = 30°, d_t = 15 meters. The parameter d_{max} must be higher than the parameter d_t. It has not impact on the matching result and is used to avoid using all available road network data, which may be too large. We have chosen 100 meters so we can use the same road network data for several consecutive iterations. The parameter α_t is relevant for the result as follows. Using a value too high, for example 80°, the algorithm would find many starts of following path that are wrong (false positive). Using a value too small, for example 5°, the algorithm would not find many starts of following path that are right (false negative). The parameter d_t is the parameter most relevant for the result of the algorithm. Using a value too high, the algorithm would match off-road points to roads (false positives). Using a value too small, the algorithm would not be able to match on-road points (false negatives). The impact of the parameters is reduced considerably by preprocessing the input GPS points, as described in the step1 of the overview subchapter.

4.3 Computing Following Paths

The algorithm used to compute the following paths uses the **Proximity** measure, which was defined in the previous subchapter, and the functions **SelectFollowingLink**, **SelectTurningLink**, and **IsAhead** defined below.

Given a link *L*, the **following link** of *L* is the link connected to the end of *L* that a human would follow if travelling on *L* and continuing straight without turning at the end of *L*. The links connected to *L* that are not the following link of *L* are referred to as the **turning-links** of *L*. Fig. 5 shows a representation of *L* and its four connected links, where L_3 is its following link, and L_1, L_2, and L_4 are its turning-links.

The following path algorithm defined in our previous work [14] is used to find the following link of a link. Below, we define a simplified function that does not consider misalignments in the road network data.

SelectFollowingLink(*L, S*): Given a link *L* and a set of links *S*, which is a subset of a road network, if no link from *S* is connected to *L*, return NULL. If only one link from *S* is connected to *L*, return that link. Otherwise, return the connected link *Lc* that minimizes the function Angle(*L*, L_c). The function Angle(*La, Lb*) returns the angle between links *La* and *Lb*, as shown in Fig. 5.

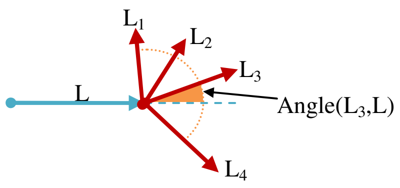


Fig. 5. Links connected to link *L*, and the angle formed between links *L* and L_3

Next, we define a function used to find the most suitable turning-link that passes through a GPS point.

SelectTurningLink(*p, L, S*): Given a GPS point *p*, a link *L*, and a set of links *S*, which is a subset of a road network, the most suitable turning-link is found as follows. If any link of *S* is connected to *L*, return NULL. Otherwise, return the connected link *Lc* that minimizes the function Proximity(*p*, L_c) defined in the previous chapter.

Below, we define the function IsAhead(*p, L*). Given a point *p* and a link *L*, IsAhead(*p, L*) returns TRUE when *p* lies on a following path from *L*. IsAhead(*p, L*) returns FALSE when *p* lies either on *L*, or a link of another following path that follows a turning-link starting from *L*.

IsAhead(*p, L*): Given a point p and a directed link *L*, IsAhead(*p, L*) determines whether *p* is ahead of *L*. Point p is considered ahead of L and IsAhead(*p, L*) returns TRUE, if and only if the angle between *L* and a vector formed by *p* and the end point of *L* is greater than 90°, as shown in Fig. 6.

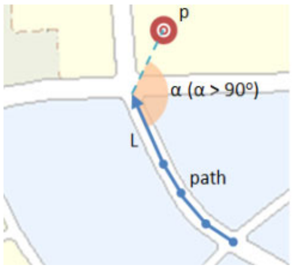


Fig. 6. An Example of IsAhead(*p, L*) returning TRUE

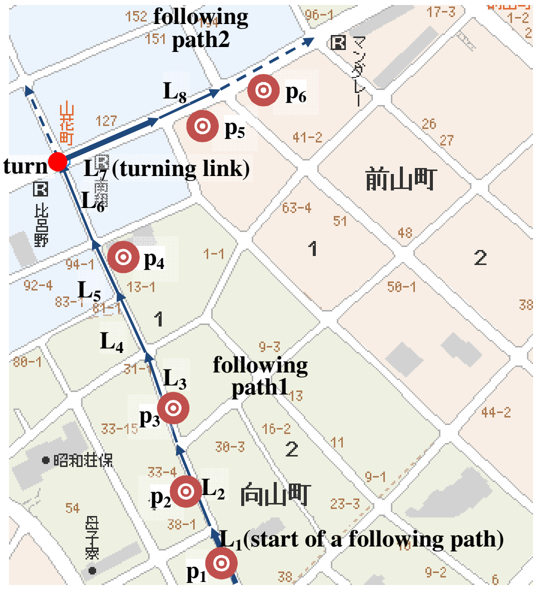


Fig. 7. Following paths that pass through the GPS points

Below we define the algorithm used to compute the following paths. Given the start point of a following path, the algorithm constructs following paths by finding following links and turning links that pass through the points of the GPS trace, as shown in Fig. 7. In this example, the algorithm generates five following links, L_2 to L_6, which follow L_1 until the path reaches turning link L_7, and then it generates the following link L_8, which follows L_7.

The step **MakeFollowingPaths** is defined as follows. Given a following path *FP*, the last link of the path is accessed by *FP*.last.

Input: a road network *R*, a start link *L*, and a sequence of GPS points of a GPS trace not yet considered *sPoints* = $(p_1, p_2,\ldots, p_n)$.
Output: a set of following paths.

System Parameters: a threshold distance d_{max}, and a threshold of the Proximity d_t, discussed in the previous subchapter.

```
FPath = create a new following path that contains L.
Add FPath to the solution.
For each point $p_i$ in sPoints perform the following steps:
    Step1. [Load] Load a set of links $S_1$ within a rectangle R1, from a database
    server for the road network R, where R1 is a minimum bounding rectangle that
    includes two circles of radius $d_{max}$, whose centers are $p_i$ and the last shape point
    of FPath.last.

    Step2. [Follow]
    If IsAhead(FPath.last, $p_i$) = TRUE,
        NextL = SelectFollowingLink(FPath.last, $S_1$)
        If nextL = NULL,
            Terminate
        Else
            Append nextL to the end of FPath
            Repeat Step2
        EndIf
    EndIf

    Step3. [Turn]
    If Proximity($p_i$, FPath.last) > dt,
        nextL = SelectTurningLink($p_i$, FPath.last, $S_1$).
        If nextL = NULL,
            Terminate
        EndIf
        If nextL has been already selected in the iteration of $p_i$,
            Terminate.
        EndIf
        FPath = create a new following path that contains nextL.
        Add FPath to the solution.
        Go to Step2.
    EndIf

EndFor
```

5 Generating a Travel Tree

5.1 Overview and Data Definitions

This chapter describes the partonomic tree of a travel trace, hereinafter called travel tree for short, and the method to generate it. Important concepts and data structures used are defined below.

Address: A set of four names: country, prefecture, city, and ward name.

(Administrative) region: Country, prefecture, city, or ward.

District: A traversable area outside public roads, i.e. off-road, for example parks, gardens and squares.

Turn: An intersection where a travel trace turns.

Section: Sections are portions of a travel trace associated with regions, districts, and road networks. We define the following four kinds of sections of a travel trace:

- Turn section: a portion that is a link whose last intersection is a turn of the trace.
- Route section: a portion that is a set of links that belongs to the same *address* and route, and does not contain turns.
- District section: a portion that is a set of GPS points that belong to the same *address* and district.
- Straight section: a portion of a link or GPS points that belongs to the same *address*, and does not belong to a route or district, and does not contain a turn.

Travel Tree: A partonomic tree of a travel trace consists of a root, a set of nodes (*pNode*), and a set of leaves (*pLeaf*). The partonomic tree manages all sections of the travel trace according to the part-of-relations among the regions where the sections belong. The nodes of the tree are grouped in four levels, as shown in Fig. 8.

pLeaf: A *pLeaf* represents a section of the travel trace and has the following attributes.

seq: the sequence numbers of the sections in a travel trace.

shape: a set of off-road points or a set of links.

address: the *address* of the section.

districtName: the name of the district where the section belongs (can be NULL).

routeName: the name of the route where the section belongs (can be NULL).

turnName: the name of the intersection at the end of the link, if the section is a link, and after the link the trace turns (can be NULL).

pNode: A *pNode* represents a region reached in the travel trace and has the following attributes.

category: a kind of an administrative region, which is one of country, prefecture, city, or ward.

label: the name (toponym) of the region.

area: the minimal rectangle that includes all sections in the leaves that are the descendants of this *pNode*.

children: an association to a set of child nodes or a set of leaves. In a travel tree, the descendants of *pNode* are contained in *pNode*.

5.2 Generating a Travel Tree

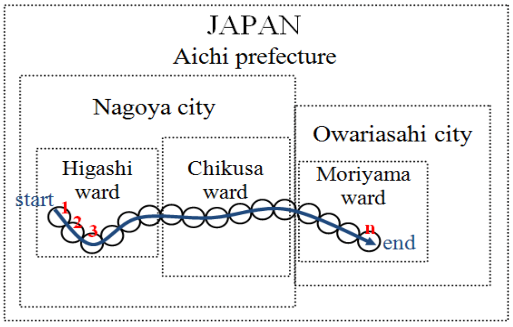
A partonomic tree of a travel trace is generated from the sequence of following paths and off-road paths of a travel trace by the below steps.

Step1. For each link and off-road point of the sequence of paths, obtain the following information:

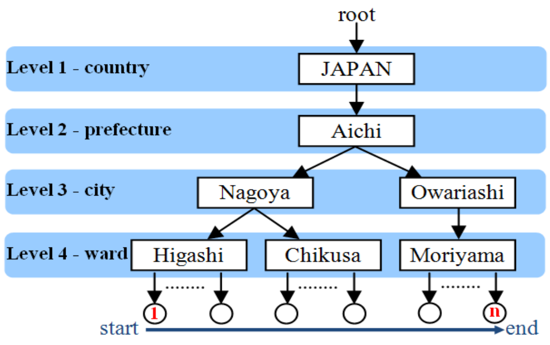
- Address: get the address by reverse geocoding.
- District: if it is an off-road point and belongs to a district, get the name of the district it belongs from a GIS database. Otherwise set to NULL.
- Route: if it is a link and belongs to a route, get the name of the route it belongs from a GIS database. Otherwise set to NULL.
- Turn: if it is the last link of a following path, get the name of the last intersection of the link. Otherwise set to NULL.

Step2. For each sequence of consecutive links, and each sequence of consecutive off-road points that share the same values for all attributes, create a *pLeaf* using their attributes.

Step3. For each different region reached, create a *pNode* using the address of the sections.



(a) travel trace on a map



(b) partonomic tree of a travel trace

Fig. 8. Two representations of the regions reached (squares) and the sections (circles) of a travel trace

6 Travel Trace Labelling

6.1 Overview

This chapter describes the methods used to generate (and place) clickable labels from the partonomic tree of a travel trace for a map view. When multiscale map views are used, for example focus+glue+context maps, areas with different map scales are considered different map views and are labeled independently.

Clickable label: A clickable label is displayed as a box that contains text (a name), and a pictogram representing its category. A label has the following attributes.

text: the text written in the label.
category: the category of the label, defined below.
location: the location where the label is placed.
scope: the set of sections represented by the label. The scope can be also defined by a node, meaning that the scope is composed by all the leaf descendants of the node in the travel tree.

Labels can belong to any one of the following seven categories: (1) country, (2) prefecture, (3) city, (4) ward,(5) district, (6) route, and (7) turn. Hereinafter, labels of the first four categories are referred as region labels and labels of the last three categories are referred as path labels.

The strategy of the map labeling methods is as follows:

(1) All district, route, and turn sections that are visible are represented by path labels, regardless of the scale. Overlapping of labels is reduced by replacing labels by representative icons. As a result, users can easily find information, even when using small map scales.

(2) Region labels are organized in the view so that they do not overlap, and the part-of relations among the reached regions is clear. As a result, users can intuitively navigate through the representation by clicking on the region labels.

The labels are generated by the following steps:

Step1. Generate the visible travel tree.
Step2. Generate the path labels.
Step3. Compound the path labels.
Step4. Generate the region labels.

The first and second steps are described below, while the third and forth steps are explained in detail in the following subchapters.

The first step generates a visible travel tree, hereinafter referred to as *vT*, which contains the leaves and nodes of the original travel tree that are visible, i.e. at least a part of the leaves and nodes is represented within the map view. Given a travel tree *T*, its *vT* is generated efficiently by using the attribute area of the nodes, as follows. For each node child of the root of *T*, if its area overlaps with the map view, it is added to *vT*, and the process is repeated for its children. Otherwise, the node and its descendants are not added to *vT*. The sections added to *vT* keep the same id value than in the original travel tree.

The second step generates path labels by using the visible travel tree. The district, route, and turn sections in *vT* are labeled by taking into consideration their visible parts, so that labels do not fall outside the view. A label is placed in the middle of each visible part of each section, as shown in Fig. 9.

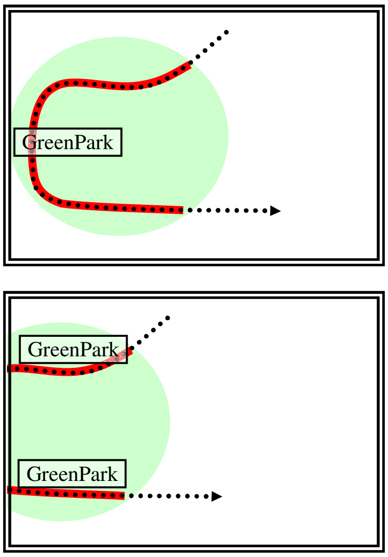


Fig. 9. Two representations of a travel trace where district sections are labeled. The map view bounds are represented by a double frame and the labeled sections are represented by a thick line.

This chapter presents two methods that, in addition to providing an organized representation, considerably reduce the overlap among labels. After applying the two methods, labels of different categories may overlap. Resolving these overlaps is beyond of the scope of this research, because many approaches available in the field of map labeling address this problem [15].

6.2 Compounding Method

This method solves the problem of overlapping labels of the same category, and provides a useful interaction for the user. The following steps are applied in this method.

Step1. A set of placed labels of the same category that overlap is replaced by a clickable icon that represents their category.

Step2. The scopes of the replaced labels are compounded into one scope that becomes the scope of the icon.

Step3. The location of the icon is set equal to the average of the locations of the replaced labels.

Step4. When users click the icon,

Step4.1. If a single-scale map is used,

- o the system adjusts the scale of the map to the compound scope represented by the icon.

Step4.2. If a multi-scale map is used,

- o the system places a new map area of a larger scale, centered at the location of the icon.
- o the system adjusts the scale and size of the new area to the compound scope represented by the icon, so that its labels do not overlap.

Fig. 10 shows four representations that illustrate the compounding method. When users click on the icon in representation (b), representations (c) and (d) are generated.

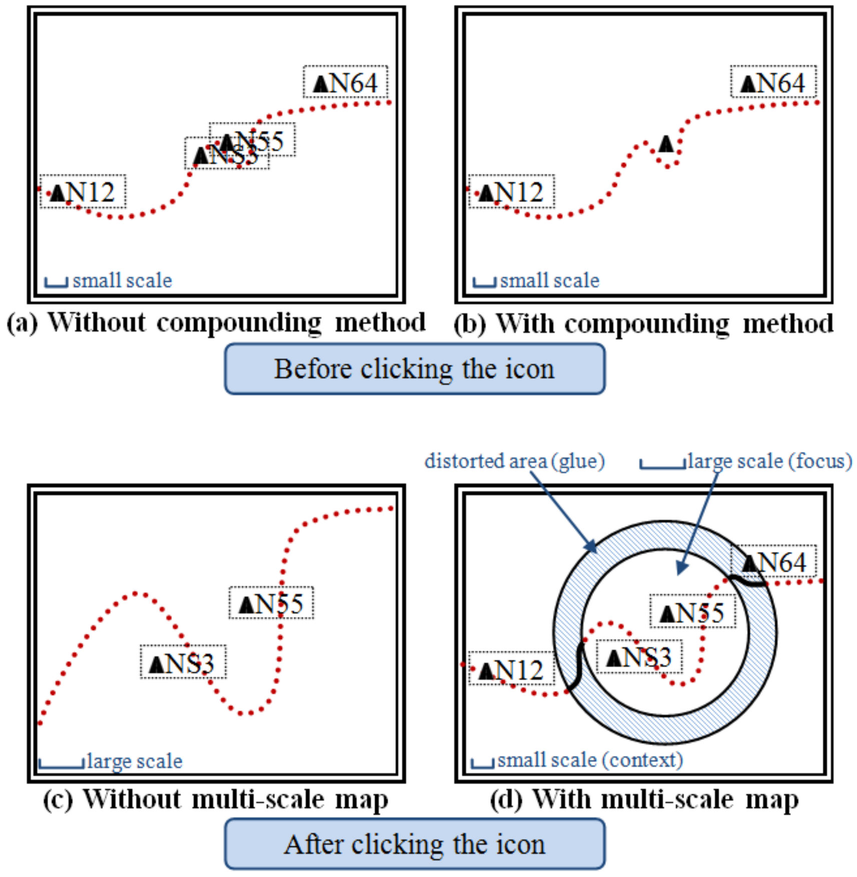


Fig. 10. Four representations of the same travel trace where only the routes are labeled. The map view is represented by a double square.

6.3 Piling Method

This method generates a clear and organized representation of the reached regions in a map view. In addition, the part-of relations among the regions are represented. Let us consider the following concepts.

Pile of labels. A pile of labels is a set of region labels arranged vertically (one above the other), according to the following relation: a region represented by a label is part of the regions represented by the labels above. For example, a portion of a travel trace in a city called Nagoya that is within the prefecture Aichi in Japan can be represented by using a pile of three labels: "Japan", "Aichi" and "Nagoya", being Japan at the top of the pile and Nagoya at the bottom. Fig. 11 shows several piles of labels.

View label. A view label is a label of a region that includes the whole travel trace represented within a map view.

The method is based in three points as follows: (1) the regions too small for the map scale are not labelled, (2) the view labels are arranged in one pile and placed in the corner of the screen, and (3) the remaining region labels are arranged in piles and place over the map view avoiding label overlaps.

First, the piling method selects one level of the tree vT regarding the current map scale. The method selects the lowest level of the tree such that the area of every node in the level is big enough when represented in the current map scale to place a pile of four labels. The regions represented by nodes in the visible tree vT below the selected level are considered too small and not labelled.

Then, the method identifies and places the view labels in the corner of the screen. View labels are easily identified by using the tree vT. A view label represents a node that is the only node in its level in vT. Fig. 11 shows two example of vT where the nodes represented by a view label are dashed boxes.

Finally, the method arranges the region labels that are not view labels by following the below steps.

Step1. Place the labels that represent the nodes of the selected level in the average location of the represented portion of trace. For example, the label of the city Nagoya is placed in the average location of the trace points within Nagoya.

Step2. Place the labels that represent the nodes of the upper level above the label of the largest child node. For example, in Fig. 11 the label "Nagoya" is placed above the label "Chikusa" because the trace within the ward Chikusa is larger than the region within the ward Higashi.

Step3. Repeat Step2 using the upper levels until the top of vT is reached.

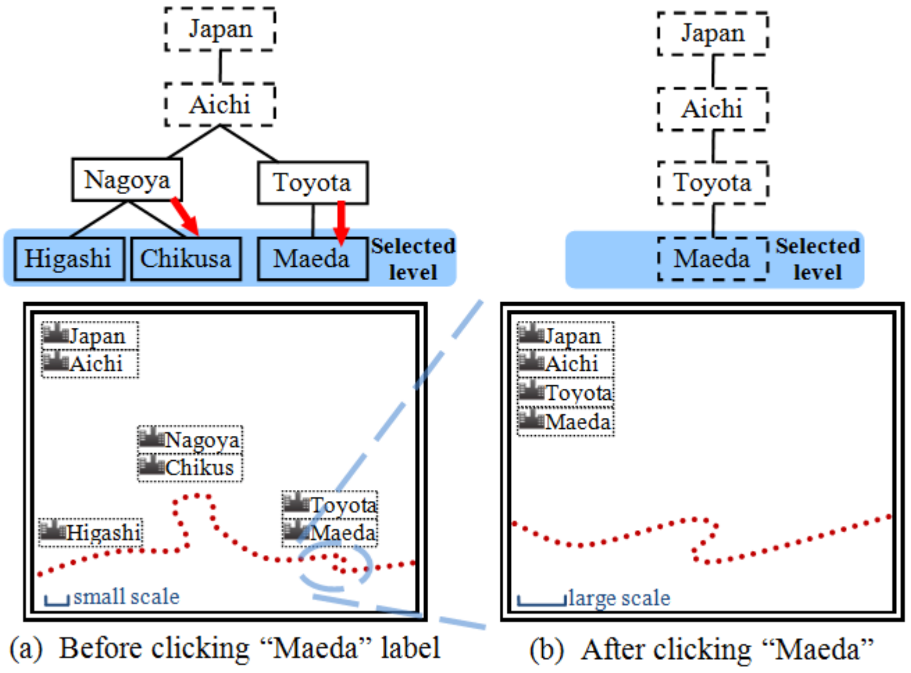


Fig. 11. Two representations of a travel trace where only the regions are labeled. Map view bounds are represented by a double frame. The corresponding visible travel tree is represented above each representation.

7 Prototype System

We have developed a prototype of the proposed system for a web-based interactive map service, which receives either a KML or a GPX file with a GPS trace, and provides a pictorial and verbal travel trace representation on a focus+glue+context map [14][16]. A demonstration movie of the prototype system is provided on the internet[1].

The focus+glue+context map system, which is a fisheye-type map developed by our previous work, displays an area of the map, the focus area, in a detailed map scale. The rest of the map view, the context area, is displayed in a smaller map scale. A glue area is placed between the two areas to absorb the scale difference.

We used the prototype system to evaluate the feasibility of the proposed system. Fig. 12 shows a screenshot of the proposed system, where a pictorial and verbal travel trace is represented over a focus+glue+context map. In the figure, there is one view pile (Japan-Aichi) for the context map view, and one view pile for the focus (Showa) map view. An icon represents several compound turn labels.

The prototype system obtains GIS data via the web service of GeoNames that allows reverse geocoding [17].

[1] `http://tk-www.elcom.nitech.ac.jp/demo/fisheye.html`

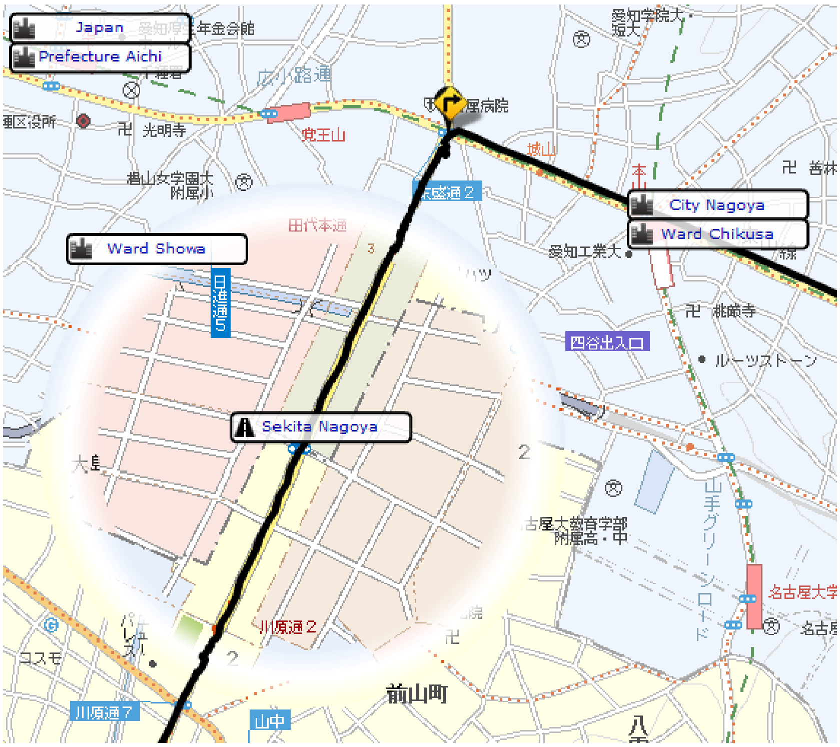


Fig. 12. Screenshot obtained from the developed prototype, depicting a pictorial and verbal representation

8 Conclusions and Future Work

In this paper, we presented a novel and effective system for generating a pictorial and verbal travel trace representation from a GPS trace. The representation allows users to understand details of a travel trace and navigate it on the basis of verbalized portions of the trace. The prototype of the proposed system showed that the system is useful and feasible.

In the future, we plan to evaluate the proposed system in terms of performance. In particular, we will evaluate real users using a traditional representation (line over map images) and the proposed pictorial and verbal trace representation. Moreover, we plan to develop a system that not only retrieves elements related to the travel trace but also those related to user behavior; these elements will be combined to provide a complete travel plan [18].

Acknowledgments. In the development of the prototype system, we were supported by Yahoo Japan Corporation. We would like to thank the Yahoo Japan Corporation. This work was also supported by JSPS KAKENHI 20509003 and 23500084.

References

1. KML, `http://code.google.com/apis/kml/`
2. GPX, `http://www.topografix.com/gpx.asp`
3. Google Earth, `http://earth.google.com`
4. Global Mapper, `http://www.globalmapper.com/`
5. Tversky, B., Lee, P.U.: Pictorial and Verbal Tools for Conveying Routes. In: Freksa, C., Mark, D.M. (eds.) COSIT 1999. LNCS, vol. 1661, pp. 51–64. Springer, Heidelberg (1999)
6. Tappe, H., Habel, C.: Verbalization of Dynamic Sketch Maps: Layers of Representation and their Interaction. In: Proc. the Cognitive Science Conference (1998)
7. Brakatsoulas, S., Pfoser, D., Salas, R., Wenk, C.: On map-matching vehicle tracking data. In: Proceedings of the 31st International Conference on Very Large Data Bases (VLDB 2005), pp. 853–864. VLDB Endowment (2005)
8. Yuan, J., Zheng, Y., Zhang, C., Xie, X., Sun, G.-Z.: An interactive-voting based map matching algorithm. In: Proc. 11th Int'l Conf. on Mobile Data Management (MDM), pp. 43–52 (2010)
9. Google Maps, `http://maps.google.com/`
10. Polishchuk, V., Vihavainen, A.: Periodic Multi-labeling of Public Transit Lines. In: Fabrikant, S.I., Reichenbacher, T., van Kreveld, M., Schlieder, C. (eds.) GIScience 2010. LNCS, vol. 6292, pp. 175–188. Springer, Heidelberg (2010)
11. Agrawala, M., Stolte, C.: A Design and Implementation for Effective Computer-Generated Route Maps. In: AAAI Symposium on Smart Graphics, pp. 5–46 (2000)
12. Yamamoto, D., Taniguchi, K., Ozeki, S., Takahashi, N.: Ontology-based Abstraction on Elastic Mobile Maps. In: Third International Workshop on Ubiquitous, Pervasive and Internet Mapping, Shepherdstown, West Virginia (2008)
13. Greenfeld, J.: Matching GPS observations to locations on a digital map. In: Proc. 81th Annual Meeting of the Transportation Research Board (2002)
14. Yamamoto, D., Ozeki, S., Takahashi, N.: Focus+Glue+Context: An Improved Fisheye Approach for Web Map Services. In: Proc. the 17th ACM SIGSPATIAL Int'l Conf. on Advances in Geographic Information Systems, pp. 101–110 (2009)
15. Kern, J.R., Brewer, C.A.: Automation and the Map Label Placement Problem. Cartographic Perspectives 60, 22–45 (2008)
16. Takahashi, N.: An Elastic Map System with Cognitive Map-based Operations. In: International Perspectives on Maps and Internet. Lecture Notes in Geoinformation and Cartography, pp. 73–87 (2008)
17. GeoNames, `http://www.geonames.org/`
18. Lerin, P.M., Yamamoto, D., Takahashi, N.: Inferring and Focusing Areas of Interest from GPS Traces. In: Kim, K.-S. (ed.) W2GIS 2011. LNCS, vol. 6574, pp. 176–187. Springer, Heidelberg (2010)

Event Processing and Real-Time Monitoring over Streaming Traffic Data

Kostas Patroumpas[1] and Timos Sellis[1,2]

[1] School of Electrical and Computer Engineering
National Technical University of Athens, Hellas
[2] Institute for the Management of Information Systems, R.C. "Athena", Hellas
{kpatro,timos}@dbnet.ece.ntua.gr

Abstract. Tracking mobility of humans, animals or merchandise has recently given rise to a wide variety of location-based services and monitoring applications. In this paper, we particularly focus on real-time traffic surveillance over densely congested road networks in large metropolitan areas. In such a setting, streaming positional updates are being frequently relayed into a central server from numerous moving vehicles (buses, taxis, passenger cars etc.). Our analysis concerns two important aspects. First, we outline characteristics of a robust processing engine that is capable to efficiently manage such massive, transient, and perhaps noisy geospatial data. Our objective is to provide online aggregates and reliable estimates regarding the current traffic situation at multiple levels of resolution. At a second step, we design a framework for effective multi-modal dissemination of derived information to the end users, in the form of interactive maps for intuitive visualization as well as instant notifications via message feeds. As a proof of concept, we also report on our ongoing development of EPOPS; in its current version, this functional prototype of the proposed scheme is able to deliver cross-platform geographic, textual, and even multimedia content through web and smartphone interfaces.

Keywords: cross-platform dissemination, events, geostreaming, multi-resolution, traffic analytics.

1 Introduction

Widespread deployment of sensor networks and cost-effective communication technologies have recently led to a surge in Intelligent Transportation Systems (ITS). Along highways, vehicle counts get collected via stationary detectors or inductive loops placed on the road surface. Surveillance cameras monitor traffic conditions and can provide data for speed and travel times from post-processed video images. But installation of such systems over dense networks in urban areas with millions of inhabitants seems neither feasible nor affordable, because expensive equipment is set up at fixed positions on major roads.

Instead, modern positioning devices (like GPS, RFID, GSM) can actually turn moving vehicles into active contributors of high-quality, real-time traffic information. Effectively, such floating car data (FCD) tracks successive locations for a fleet of registered vehicles at distinct time instants. At a relatively low cost, FCD is then transmitted into a central server, where it gets analyzed for extracting traffic statistics [6,22,24]. Apart from control centers, this data might further assist to travel planning for commuters, support coordination of public transport, and also integrate with car navigators or traffic telematics.

In this paper, we advocate for a real-time evaluation of vehicle traces over road networks and their collective representation as *traffic data streams*. Similar platforms have emerged lately [1,7,10,11,18], as vehicle positions and derived statistics are inherently fluctuating, potentially intermittent, and ever more voluminous to be hosted by a traditional DBMS. In a spirit close to ours, the objective is to turn quantitative samples (raw positions) into qualitative estimates (e.g., average speed, expected travel times). Such approximate, yet reliable analytics is derived online, employing stages of filtering, aggregation and export.

Nevertheless, our framework is mostly geared towards a *multi-resolution representation* of traffic streams and props up *multi-modal diffusion* of custom, succinct analytics to potential users. We stress that our goal is not to suggest novel practices to traffic engineers; unsurprisingly, sophisticated algorithms already abound. What we really attempt is a *geostreaming* approach to road traffic data, enhanced with methods for extracting privacy-preserving driving patterns and evolving phenomena network-wide. Interesting events may not just be issued from external sources (e.g., reported incidents), but could be promptly detected by observing trends in traffic conditions (e.g., increasing delays).

Therefore, we introduce an efficient and robust combination of: (i) suitable *traffic semantics* that can capture spatiotemporal phenomena and evolving events along road networks; (ii) the advanced data management capabilities offered by *stream processing engines*; (iii) the skillful dissemination power of *web and mobile technologies*; and (iv) rich visualization tools from modern *geospatial infrastructure*. Implementation of our prototype platform (EPOPS) confirms that this methodology can offer up-to-date traffic information at varying levels of detail and greater range, and also achieve its versatile portrayal through interactive maps. Our contribution can be summarized as follows:

- We outline an abstract model for representing traffic data streams.
- We suggest a solid methodology for online processing of floating car data.
- We combine on-the-fly traffic measurements with contextual information in order to provide timely, reliable, and multi-faceted analytics.
- We develop interfaces for visualizing, inspecting and diffusing results through web and smartphone applications.

The remainder of this paper is organized as follows. In Section 2, we discuss fundamental notions and modeling of traffic data streams. In Section 3, we develop a processing framework for translating vehicle positions into online traffic analytics. Section 4 presents methods for cross-platform delivery of results to the end-users. In Section 5, we report our experience from implementing EPOPS, a functional prototype of the proposed scheme. In Section 6, we survey related work. Section 7 offers conclusions and indicates possible future extensions.

2 Fundamentals of Traffic Data Streams

In this section, we analyze design considerations and modeling guidelines concerning how streaming positional data is acquired, represented, and eventually transformed into meaningful traffic aggregates.

2.1 Data Acquisition

Although we do not rule out accepting traffic surveillance data feeds from public authorities (Traffic Police, Transportation Offices, etc.), we mainly focus on processing positional information issued from the moving vehicles. Indeed, monitoring a few thousand (e.g., 5000) location-aware cars moving in a city can provide a fairly accurate FCD sample for assessing actual traffic conditions. Assuming a proper combination of vehicle types (buses, taxis, trucks, passenger cars etc.), not only can we capture various driving patterns (fixed itineraries, ad-hoc routes, detours etc.), but also get traffic indications network-wide. Such raw tracking data could be acquired through diverse means: GPS-equipped vehicles, mobile phones of the drivers, wireless networks etc.

Positional updates are reported to a central server, but not necessarily at fixed periods, as sampling rates may not be standard for the entire fleet (e.g., buses with known itinerary may report less frequently than taxis). Reporting frequency might also be varying even for a single vehicle so as to reduce communication cost; for instance, it could relay a new location when making a turn or upon significant change in its speed. On the server side, positional information is periodically correlated with the geographic representation of the underlying network and certain rules in order to derive online traffic analytics. In brief:

Guideline 1. *Timestamped positional updates from moving vehicles are being transmitted to a centralized processor in a streaming fashion.*

2.2 Data Semantics

We adopt an object-relational schema with spatiotemporal extensions for representing static and dynamic data. Static tables store information about entities that rarely change (e.g., road network and its classification, vehicle types), whereas dynamic tables retain streaming data (e.g., vehicle positions, speed profiles for roads) always associated with timestamp values. A rich combination of semantics is employed to capture interesting traffic phenomena:

Spatial Semantics. It goes without saying that a detailed *road network* would provide a solid basis for the entire framework. Typically:

Guideline 2. *The road network is abstracted as a 2-d graph with links and nodes.*

Links represent centerlines of road segments as 2-dimensional polyline vectors, seamlessly interconnected at nodes that denote crossroads, junctions, or dead-ends. On the other hand, vehicle positions are abstracted as 2-dimensional *point* *locations*, ignoring other parameters such as the volume of each car or the altitude. Other geographic entities are clearly optional, since they do not directly interfere with traffic analysis; for example, points of interest (terminal stations, sporting venues, shopping malls etc.) or zone boundaries (limited-access areas, neighborhoods, or districts) might prove useful for specific calculations (e.g., access to a station), but they are not deemed indispensable features.

Temporal Semantics. Each incoming positional update, as well as every resulting aggregate must hold time indications [15]. Accordingly:

Guideline 3. *Timestamping of stream items achieves ordering and simultaneity.*

The former property guarantees the sequential nature of positional feeds and traffic analytics, and also allows reconstruction of the historical trace for each vehicle. The latter enables correlation of events occurring at the same time, provided that they carry identical timestamps. Not only does timestamping come for free because location-aware devices always emit *valid time* (i.e., when a position was actually measured), but it should be retained throughout evaluation stages for properly signaling aggregates (estimated speeds, travel times etc.). In case of unsynchronized clocks, *transaction time* of data admission to the system could be alternatively used, instead of temporarily buffering delayed messages.

Motion Semantics. Spatiotemporal notions such as displacement, travel time, speed, heading etc. are important when dealing with moving objects [9]. Derived values per vehicle may be either instantaneous (based on its two most recent recordings) or averages (over larger intervals). Yet, it is crucial to realize that:

Guideline 4. Atomic measurements *per moving vehicle should eventually translate into* collective anonymized aggregates *along roads.*

In our case, the current speed or direction of each individual vehicle are entirely insignificant, no matter their accuracy. On the contrary, averaging speed measurements for vehicles moving along a specific road gives an approximate, but still reliable indication about traffic conditions there. From a privacy perspective, such aggregates can also effectively hide the identity of each driver among similarly moving others, and thus cannot disclose its whereabouts.

Traffic Semantics. Apart from their geographic representation, roads are key features for traffic estimation. Therefore, they should include properties such as length, national and international codes, speed limits, number of lanes, direction of traffic flow, hierarchy (highways, primary and secondary arterials, or collector roads) etc. Furthermore, roads can be dynamically classified according to their observed level of congestion, thus distinguishing currently saturated links from those with normal flow. Most importantly:

Guideline 5. *The road network may be organized in tiers of gradual resolution.*

At the finest tier, consecutive geometric *segments* belonging to the same road can form a unified *section* (e.g., between signaled crossroads); further, a series of

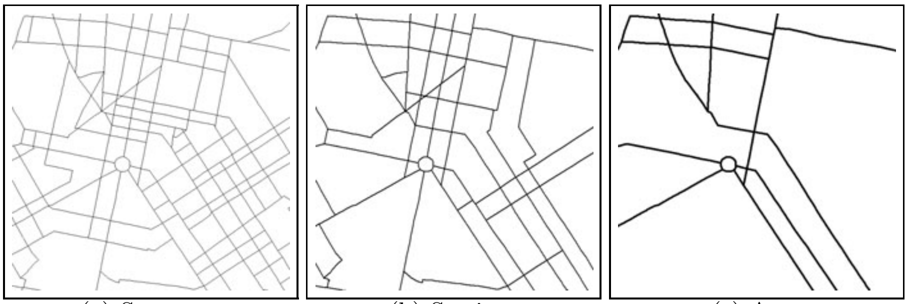


Fig. 1. Bottom-up hierarchical organization of road network in tiers

sections with similar characteristics can represent an entire *axis* at the coarser tier. As shown in Fig. 1, smaller and less important roads may be suppressed from higher levels. Optionally, this logical grouping could further lead to multi-scale geographic representations of the network: at smaller scales the map could only display geometrically simplified axes, whereas zooming into a particular area should render all segments at the finest detail. As we discuss later, such modeling is advantageous not only for its richer semantics, but also for its decisive impact on data processing and map visualization of traffic phenomena.

Event Semantics. As vehicles circulate across the network, several events may occur, such as road accidents, demonstrations, bad weather conditions etc., which may have local or more severe implications to current traffic flow. Hence:

Guideline 6. *Support for event detection and online notification is essential.*

Event handling is two-fold: (i) for *declaring certain incidents*, like announcements issued from the Traffic Police for accidents or urgent warnings; and (ii) for *discovering driving patterns online* (e.g., a sharply reduced speed well below normal levels along a road may indicate an accident; a long-observed deterioration of traffic flow on a highway may cause queues stretching over several kilometers etc.). As we explain in Section 3.5, this latter class of events requires special-purpose algorithms or complex continuous queries against the traffic streams.

2.3 Data Manipulation

As each vehicle reports its position at distinct time instants, the server is able to trace its movement as an evolving *trajectory* [9]. But in our case, neither is it affordable nor even necessary to maintain the historical trace per vehicle as a 3-d ”polyline” vector. This representation uses vertices at successive timestamped locations, and interpolates to approximate positions in between sampled readings. Since our objective is to monitor network traffic and not individual vehicles, we must identify those road segments traversed by each car across time. Therefore:

Guideline 7. *Vehicle positions incrementally create* trajectory feeds *as sequences of road segments, judiciously selected from the underlying network.*

This way, the route of each vehicle is constructed as a series of road segments derived from a "map-matching" process, as we discuss in Section 3.1. Note that geometric accuracy and connectivity are essential properties of the network, since error-prone GPS readings should be mapped into suitable road segments (i.e., where a car actually moves on) minimizing spatial mismatches. In terms of achieving smooth traffic estimates, such modeling of trajectories is certainly beneficial, as it can effectively cope with diverse reporting frequencies and possible communication delays, while it also avoids double-counting of vehicles that keep moving along the same segment. Admittedly, some trajectory feeds may appear occasionally disconnected with gaps between selected segments (Fig. 3a). However, this is not a serious flaw, as vehicle traces are eventually turned into approximate traffic estimates. These traffic analytics are computed at coarser resolutions of the network, usually against sections or axes, each stretching over several segments. Hence, estimates for speed, link saturation, expected travel times etc. are actually aggregates from values contributed by trajectories of vehicles that have recently "charged" a given road.

Guideline 8. *Traffic aggregates are computed from trajectory feeds for selected tiers of the road network.*

We stress that all derived data is implicitly georeferenced against the road network. Every trajectory measurement or traffic aggregate is always associated with a particular road entity, so the identifier of the respective linear feature suffices and must be attached to that data item. There is absolutely no need to copy geometry features into any derived dataset, as these can be readily obtained from a simple join operation thanks to common identifiers. Of course, results must also be timestamped with respect to time indications of the input.

3 Traffic Stream Processing

Figure 2 illustrates the processing components of the suggested framework. Rectangles stand for domain data; each domain may be physically implemented with multiple relational tables of static records or streaming items. Diamonds represent processing tasks applied against incoming data according to specifications, rules, and parametrization depicted with ovals. Thick arrows indicate data flow from raw positional input to output traffic information passing through intermediate modules. Next, we discuss each component in detail.

3.1 Map-Matching

This pre-processing stage attempts to associate vehicle locations with road segments of the underlying network. As already mentioned, tracking data from moving vehicles has limited positional accuracy. Apart from the sampling error

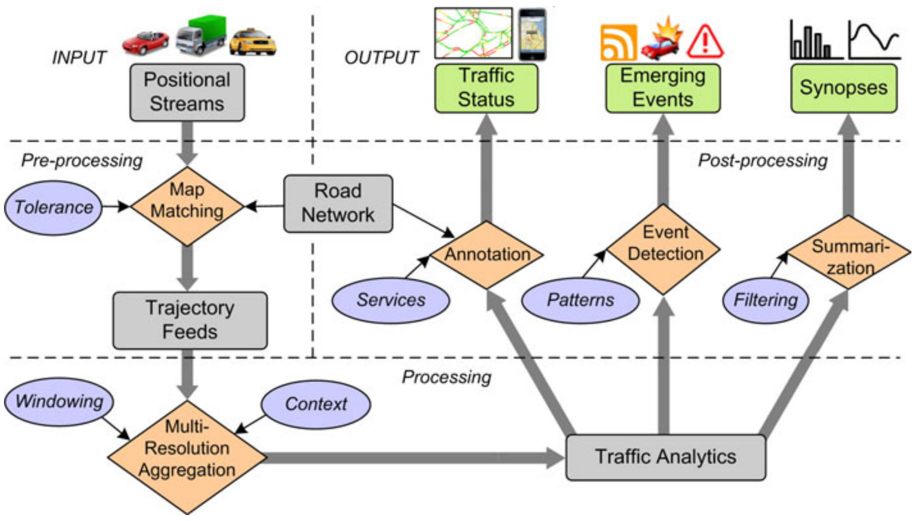


Fig. 2. Data flow diagram

caused by the reporting frequency, there is also a measurement error from the positioning device itself (e.g., GPS) [3]. Besides, the vector representation of the road network can never be perfectly accurate, due to digitization errors, scale of the source map or imagery resolution, lack of recent updates etc. As a result, it is not always straightforward to snap a positional item to the nearest node or link of the network. It may occur that locations could fall several meters off the road centerlines, and thus might be wrongly assigned (dotted segments in Fig. 3a). In addition, raw positional data are inherently noisy and must be properly cleansed before attempting any map-matching. For instance, vehicle positions should be ignored in case they might indicate manoeuvres for on-street parking followed by immobility, entrance into a private garage or parking space etc.

This problem is definitely crucial, but orthogonal to our approach. Thus, we can make use of any state-of-the-art algorithm like [3,5,19,20] that minimizes erroneous assignments under proper tolerance metrics (such as Fréchet or Hausdorff distance), also taking into account the recent vehicular movement as well as network features (e.g., direction of traffic flow, road hierarchy etc.). Depending on the technique, apart from timestamped points, other features like bearing or speed could be used for more accuracy. We integrate spatial access methods for indexing and fast retrieval of road entities, since identification of relevant segments must be performed for every incoming position. The result of map-matching is a sequence of road segments which the vehicle has supposedly traversed. Yet, we need something more: we also have to know when this vehicle entered each particular road segment, assuming that it kept moving at an estimated speed along each segment. As shown in Fig. 3b, interpolation of timestamp values from the original samples along involved road segments can provide a fair approximation of such entrance times. Note that when a car turns

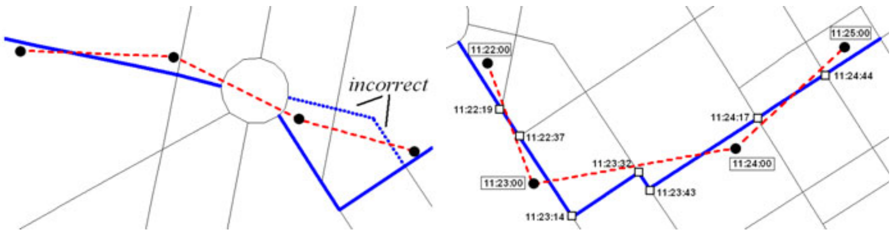


Fig. 3. (a) Mismatching and occasional gaps. (b) Map-matching of positional samples.

into another road, intermediate segments should get included into the sequence, so we utilize a shortest-path algorithm between successive samples.

Consequently, such incremental trajectory feeds contain tuples that denote when a given vehicle entered a particular road segment and what was its actual velocity. This latter feature involves simple manipulations with the latest displacement of each moving vehicle, by comparing its current and previously reported position. Apart from the speed, we also need to know the direction of movement, which is quite significant for bidirectional roads.

3.2 Multi-resolution Traffic Analytics

The main processing task deals with online aggregation against map-matched trajectory feeds. Although such spatiotemporal computations are not particularly sophisticated, they must be repetitively applied across an extended network with thousands of links, and also keep track of the most recent traces for numerous vehicles. Such a task cannot be accomplished by a traditional DBMS, not only because of the unsustainable burden of frequent transactions, but also due to the necessity for incremental response to continuous traffic analytics.

We advocate for a lightweight, special-purpose stream processing mechanism that works in main memory and can offer qualitative, real-time estimates for:

– *Average speed.* Since every trajectory tuple for a given vehicle always states which road segment it currently traverses and at what speed, this aggregation is just a grouping of items by their road identifier.
– *Expected travel time* can be simply expressed as quotient of the geometric length of each road entity over its estimated average speed.
– *Vehicle counts.* This measure of traffic load is actually based on samples, since monitoring all vehicles moving in a city is not realistic by today's standards. Still, such counts might be useful as a rough indication of flow saturation, but primarily for assessing the accuracy of other traffic statistics.

Preferably, traffic estimates make sense for larger sections or axes (i.e., top tiers of road network) in order to attain sufficient samples. If only a few vehicles are currently moving along a large stretch of a road, then this sample may not be representative, so doubtful measurements should not get publicized. As a means of coping with such inherent deficiencies and also offering succinct traffic information, we opt for computing *multi-resolution analytics* according to:

– *Spatial context.* Data can be progressively aggregated at multiple network tiers (mostly axes and sections) so as to offer more insight into actual traffic conditions. For instance, flow may appear almost regular along an axis, but at a certain section the situation may be worsening. Similar roll-ups or drill-downs at varying levels of detail can help explaining certain traffic phenomena. Depending on application requirements, more options are possible: travel times may be estimated for several frequent routes (e.g., from/to the airport); low speed in designated zones may indicate congestion (around major junctions, near terminal stations or commercial districts) etc.
– *Temporal context.* Aggregating over the latest measurements only (e.g., instantaneous speed of each vehicle) can hardly give a truthful glimpse of traffic status. In most cases, data should be examined at varying time horizons (during past 5, 10, 30 minutes or more) for smoothing fluctuations caused by waiting times at traffic lights, impulsive reactions from drivers etc.
– *Traffic context.* Analyzing data by vehicle type can give precious clues about driving patterns for buses, taxis, trucks, passenger cars, etc. Besides, for selected arterials or road classes, this breakdown can also provide the presumed traffic composition (i.e., % of each vehicle type over the total traffic).

With respect to time frames, it matters not merely the period of interest for the latest streaming data, but also how frequently analytics should get refreshed. As in most stream processing engines, we stipulate *sliding windows* [15] for specifying the *range* ω (e.g., samples received during past 10 minutes), the *sliding step* β (e.g., results get reevaluated every minute) and the *initiation time* τ_0 of this computation. Note that parametrization of time frames is based on timestamps, in order to accomplish incremental production of results at regular intervals. Certainly, the exact window specifications are application-dependent, but they can be fixed in case of periodic evaluations (e.g., rush hours), or ad-hoc for capturing specific events (such as traffic near a sporting venue).

3.3 Derived Annotations

As soon as a new batch of traffic analytics is produced, this information should be forwarded to users. But is it worth returning the exact statistics for average speed or travel time per road, since these are just presumed estimates? We suggest that such approximate data should better be translated into inferred, *qualitative* indications. Indeed, knowing that the observed road speed is 12 km/h may seem too detailed and hardly precise due to limited samples, whereas an indication of "slow speed" is fairly reasonable, succinct and clear.

We employ a post-processing stage that interprets traffic analytics into *annotated* information according to its intentional use, particularly in web and wireless platforms. Determined by the type of services to be offered (e.g., maps, charts, statistics etc.), a variety of lookup tables may be utilized to transform numerical into categorical data. For instance, distinguishing speed values as "slow", "moderate" or "fast" is easily comprehensible, especially when depicted on maps with suitable symbology. Deviations of observed travel times from historical figures could simply characterize the congestion level of each road feature as "heavy", "normal" or "light". Note that such classification could be done dynamically according to time-dependent lookups (e.g., ranges may vary for rush hours, holidays etc.) and domain expert knowledge of local traffic conditions.

3.4 Updateable Synopses

Although online manipulation of incoming updates is our main focus, it should be noted that certain historical data may also prove valuable to traffic analysis. Raw positional data should not be maintained, not only because of privacy concerns but also due to their accumulating bulk. Nevertheless, properly summarized information about speed profiles per axis, vehicle counts or travel time estimates for traversing each road segment, can all be permanently stored in tables for both online use and offline historical comparisons.

Such concise synopses come from further aggregation of traffic analytics at *multiple granularities* [14]. Taking into account some expert knowledge, applied windows may span time periods varying from a few minutes to an entire day. Such post-processed data are incrementally appended into summaries and thus essentially maintain several evolving timeseries for critical parameters (speed, flow level, travel time etc. per road). In addition, the most fresh batch of such aggregates may also be used to offer online services, particularly for finding time-dependent shortest paths according to fluctuating traffic conditions.

3.5 Event Detection

Timely recognition of emerging incidents across the road network is of great importance for traffic surveillance. Such interesting phenomena include:

Link saturation: When the observed speed along a road steadily approaches or drops below a threshold (deduced from statistics over this time period), there are high chances that this link might soon become overloaded. As we discussed in [14], computing average speed values against nested time frames (say, spanning 5, 10 and 20 minutes) could instantly give a sign for unsatisfactory flow.

Deteriorating traffic flow: We can make use of a multi-level linear regression algorithm for discovering trends in traffic patterns across selected roads. The method we introduced in [14] involves multi-granular windows for online estimation of linear fits over varying time ranges in the recent past. When the current slope of these trendlines exacerbates or deviates substantially from the historical pattern of vehicle circulation on a road, traffic controllers can be notified to investigate the possible causes of that unusual situation (e.g., an accident, an inundated underpass after heavy rain, lengthy queues towards the stadium etc.).

4 Cross-Platform Content Delivery

The wealth of traffic data gathered and analyzed through the processing engine makes its availability to the public a challenging opportunity for versatile, customized, and timely dissemination of information through web and mobile platforms. Offering online traffic analytics and event warnings can be accomplished with a user-friendly combination of rich visualization tools, interoperable data formats and modern technological outlets, as we explain next.

Advanced Visualization. Interactive maps are the perfect means for conveying traffic information. World-scale mapping infrastructure (e.g., GoogleMaps, OpenStreetMap etc.) or commercially available cartographic data could be used as the backdrop for real-time traffic monitoring. Properly annotated analytics for average speed, expected delays, and incident locations across the network can be exported in XML, RSS etc. Furthermore, correlating results with the underlying road network can also dynamically produce geographic features in GML/KML formats. Taking advantage of a multitude of APIs and open-source libraries, such dynamic layers can then be easily superimposed over the basic map with standardized symbology (colors, symbols, legends). On the other hand, continuously maintained speed profiles and traffic composition for roads could be illustrated with diagrams, histograms or charts. Last but not least, geographic annotation could integrate multimedia content (e.g., images linked to event locations, or streaming video from surveillance cameras), as well as eye-catching animation (e.g., flashing heavily congested links).

Customized Presentation. Traffic controllers, public authorities, and drivers often have differing perceptions of the traffic status. A common driver is primarily concerned about congestion along her way, while the Police usually want to avert bottlenecks and keep a regular flow mostly throughout arterial roads. Thus, results should be made available via *localized views* at various levels of detail (city, district, neighborhood) or depending on proximity to the current location of the driver.

Apart from standard information available for free, subscribers of the service may be offered a *personalized view* of the map with more detailed content. For example, user preferences could affect the desired level of resolution, refresh periods, filters for particular types of events, specific zones of interest etc. In addition, registered users may be allowed to post their remarks or hints to other drivers (e.g., a tweet to avoid a congested route).

Multi-modal Availability. Nowadays, the Web is the primary medium for making data easily accessible to the public. Hence, a user-friendly portal is indispensable for presenting and searching traffic information, combining maps, text, charts or multimedia. In addition, web services can be offered to public authorities or subscribed companies, such as fleet management agencies, taxi associations, delivery firms, advertising, radio/TV stations, etc. Such services can provide custom information including privacy-preserving statistics, traffic composition, speed profiles for roads etc.

Development of mobile applications for smartphones, tablets and PDAs would prove by far the most popular solution, as drivers can become aware of the traffic conditions while on the move. From instant notification about incidents, to interactive maps, to alternative route planning or a suite of online facilities (e.g., route recalculation, message tweets, image posts), the potential is enormous.

Broadcasting notifications can also be achieved in various ways. Depending on their preference settings, subscribers could receive urgent alerts with SMS texts or monitor periodically refreshed RSS feeds for incidents of their particular interest. Traffic message boards along major roads can display warnings to the drivers, whereas touch screens in parking lots or gas stations around the city could be used to depict interactive maps and news feeds. Finally, traffic message channel technology (TMC) via radio frequencies is currently operational worldwide for delivering traffic and travel information.

5 The EPOPS Prototype

EPOPS[1] is the acronym of our framework for **E**vent **P**rocessing and **O**nline monitoring of **P**ositional **S**treams. We have begun implementing a traffic monitoring platform using the open-source TelegraphCQ stream engine [17] and publicly available APIs. A central processor is continuously "listening" for streaming elements, whereas annotated analytics and events are periodically materialized into XML formats, thereafter available through web and mobile interfaces.

5.1 Data Management

Due to lack of real-time traffic information, EPOPS is currently tested against synthetic datasets that represent random itineraries of 10 000 vehicles circulating at the road network of greater Athens. We simulate online updates, enforcing a very frequent arrival rate of a new location per moving vehicle issued every 15 seconds. We assume no stream imperfections (e.g., missing or delayed data), so all positional items can be ordered by their original timestamp values.

Although TelegraphCQ is a research prototype with certain limitations (e.g., no support for nested subqueries or stream self-joins), it has been built on top of PostgreSQL. Hence, it comes readily equipped with built-in spatial operators, functions and data types (point, path, polygon, etc.), offering a great benefit for expressing continuous queries over geospatial streams as we demonstrated in [13]. Using SQL-like commands, we have defined a schema for maintaining static data (spatial and non-spatial tables) as well as properly organized streaming data (concerning incoming locations and derived features). At the least,

[1] $\pi o \psi$ is the ancient Greek name of the Hoopoe (*Upupa epops*), a colorful bird with a distinctive "crown" of feathers. In Aristophanes' comedy "The Birds" ($\rho\nu\iota\theta\epsilon\varsigma$, 414 BC), two Athenians are in search of prince Tereus, whom the Olympian gods have turned into a hoopoe and charged as all-seeing ruler ($\pi\alpha\nu\tau\epsilon\pi\acute{o}\pi\tau\eta\varsigma$) over the birds.

each moving vehicle relays tuples ⟨vehicleID, x, y, ts⟩ with its geographic position in lon/lat coordinates at timestamp ts. This data resides at stream table Positions in main memory. The road network of Athens is stored in spatial table RoadSegments and its polyline features are indexed with R-tree. Thanks to a three-part coding scheme, higher tiers are defined as external views (RoadSections, RoadAxes) over the detailed road segments.

We developed stored procedures in PL/pgSQL for coping with map-matching of observed locations into road network features. As positional items are turned into trajectory sequences that feed a stream table Trajectories, windows are employed in order to periodically derive traffic analytics at varying levels of resolution (sections and axes). Streamlined results are always emitted with the latest timestamp value, thanks to the wtime(*) function of TelegraphCQ.

Specifically, a tumbling window [15] fetches the most recently received items and assists in computing an indicative traffic load as *vehicle counts*. Note that the cutoff threshold of 10 samples could be varying for each road section:

```
SELECT T.SectionID, COUNT(T.VehicleID) AS Num_vehicles, wtime(*) AS ts
FROM Trajectories AS T [RANGE BY '15 SECONDS' SLIDE BY '15 SECONDS'
                        START AT '2011-11-28 12:00:00']
GROUP BY T.SectionID
HAVING COUNT(T.VehicleID) > 10;
```

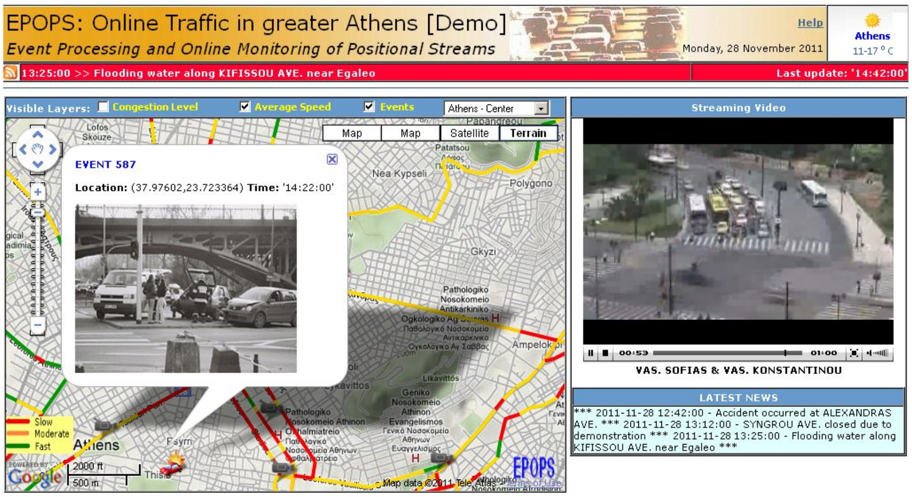
Instead, for computing the *average speed* per axis, a sliding window is employed to aggregate instantaneous speed values over the past 5 minutes. Results are updated every 15 seconds, in pace with newly arriving measurements:

```
SELECT T.AxisID, AVG(T.Speed) AS Avg_speed, wtime(*) AS ts
FROM Trajectories AS T [RANGE BY '5 MINUTES' SLIDE BY '15 SECONDS'
                        START AT '2011-11-28 12:00:00']
GROUP BY T.AxisID;
```

Expected *travel times* over road features can be also expressed with an adequate sliding window. Correlating the geometric length of each axis (as obtained from view RoadAxes) with the respective speed estimate during the past 10 minutes, easily provides such time-dependent figures every minute:

```
SELECT T.AxisID, R.Length/AVG(T.Speed) AS Travel_time, wtime(*) AS ts
FROM Trajectories AS T [RANGE BY '10 MINUTES' SLIDE BY '1 MINUTE'
                        START AT '2011-11-28 12:00:00'], RoadAxes AS R
WHERE T.AxisID = R.AxisID
GROUP BY T.AxisID;
```

Incremental results from this continuous query could be appended into a synopsis that may serve online routing requests from users. Indeed, by updating network cost values with these evolving travel times and making use of the pgRouting module [16] for PostgreSQL, we managed to provide a *shortest path facility* for resolving routes between arbitrary origin/destination points. We have also successfully experimented with a scenario for online toll charging against drivers that habitually aggravate congestion levels during rush hours.


EPOPS: Online Traffic in greater Athens [Demo]
Event Processing and Online Monitoring of Positional Streams
Help
Monday, 28 November 2011
Athens 11-17 °C
Last update: '14:42:00'
Visible Layers: Congestion Level Average Speed Events
Athens - Center
Map Map Satellite Terrain
Slow Moderate Fast
Streaming Video
VAS. SOFIAS & VAS. KONSTANTINOU
LATEST NEWS
EPOPS

13:25:00 >> Flooding water along KIFISSOU AVE. near Egaleo

EVENT 587

Location: (37.97602,23.723364) Time: '14:22:00'

*** 2011-11-28 12:42:00 - Accident occurred at ALEXANDRAS AVE. *** 2011-11-28 13:12:00 - SYNGROU AVE. closed due to demonstration *** 2011-11-28 13:25:00 - Flooding water along KIFISSOU AVE. near Egaleo ***

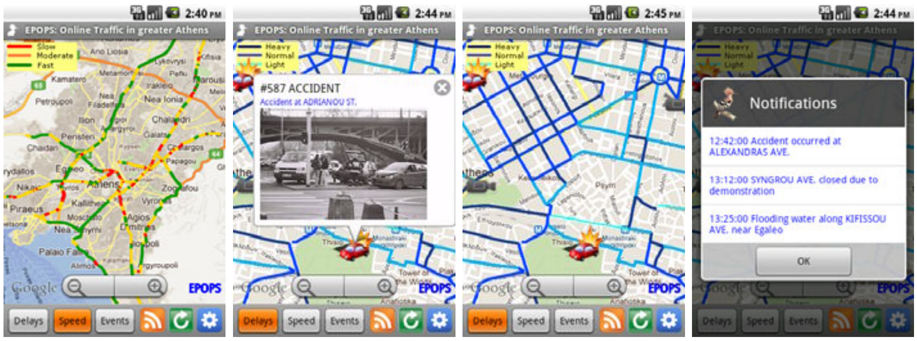
Fig. 4. Web interface for the EPOPS prototype

In terms of event handling, we currently support issuing of incidents from authorized users only. Tuples like ⟨`eventID, x, y, ts, type, message, media`⟩ get inserted into a stream table `Events`, stating occurrence of an incident (car crash, road works, etc.), along with a textual message and multimedia content (image or video). We are working towards integration of our standalone algorithms [14] for detecting events directly from traffic streams (cf. Section 3.5).

5.2 Multi-modal User Interfaces

In terms of content delivery, EPOPS intends to facilitate user interaction with traffic information in an intuitive fashion. Implemented interfaces for *web* (with JavaScript and php scripts) and *smartphone* platforms (in Android SDK) offer functionalities that can handle maps, notifications and multimedia content.

More specifically, both interfaces are centered around geographic maps using GoogleMaps API. Typical cartographic operations are built-in, so our effort focused on proper rendering of dynamic layers (speed, delays, events) with suitable symbology. Reported events and annotated aggregates from online analysis in TelegraphCQ get periodically posted at the server in various formats (XML, KML, RSS). Derived elements become available at multiple resolutions, expressed in different map scales and event rankings. For example, zooming out the map to the entire city only displays speed levels along major axes and a handful of the most important incidents. But zooming into a neighborhood retrieves more detailed information for road sections and all events occurred recently in that area. To help reduce communication costs and response times for rendering, we have chosen to provide lightweight KML files with properly generalized shapes (simplified polylines) depending on the actual map scale on screen.


#587 ACCIDENT
Accident at ADRIANOU ST.

Notifications
12:42:00 Accident occurred at ALEXANDRAS AVE.
13:12:00 SYNGROU AVE. closed due to demonstration
13:25:00 Flooding water along KIFISSOU AVE. near Egaleo

Fig. 5. Screens of Android application for the EPOPS prototype

Aside from visual inspection of traffic phenomena, the implemented web interface (Fig. 4) also offers: localized views when the user chooses an area of interest, a widget for actual weather conditions, as well as scrolling messages for events as they occur (accidents, road works, demonstrations, etc.). Furthermore, users may display photos for chosen incidents and even watch videos from surveillance cameras at major junctions. Because access to streaming video is not yet authorized for our tests, we currently show archived footage only.

The Android application for EPOPS (Fig. 5) offers a basic screen for map rendering of traffic layers with custom symbols, always at a spatial resolution implicitly controlled by the actual scale. The user is able to personalize reception of results according to her preferences; configuration is now limited to vehicle types and time horizons, but more options are possible. Finally, the mobile application connects to the server and accepts XML/RSS feeds with event notifications.

6 Related Work

Stream-based processing of geospatial data, also known as *geostreaming*, offers a novel paradigm for spatiotemporal management, particularly for online monitoring of location-aware vehicles. Spatially-enabled stream engines have emerged, covering a wide range of topics related to ITS, such as data acquisition, map-matching, aggregation, and prediction. CarTel [10] is a mobile sensor computing platform designed to collect, process, deliver, and visualize data from sensors located on automobiles. IBM InfoSphere Streams [1] shares a lot of features with our approach, as it aims at scalability, timeliness and versatility of derived information in ITS. This component-based distributed platform utilizes a declarative language SPADE for specifying data flow graphs as well as for deploying applications at runtime. Accordingly, a prototype ITS application was developed for traffic monitoring in the city of Stockholm, using real vehicle GPS readings and road network maps. Besides, an evaluation framework equipped with data stream mining algorithms is proposed in [7] specifically for traffic applications. In a case study, cooperative cars exchange messages through cellular infrastructure and the proposed techniques may detect road segments with queue-ends. Finally, GeoInsight [11] is based on Microsoft SQL Server StreamInsight platform and leverages continuous query execution with spatial and temporal capabilities, as well as ad hoc analytical extensions for online refinement and prediction. Their demo scenario handles streaming data from traffic sensors in Los Angeles county, and correlates it with historical statistics in order to predict future trends. As opposed to such sophisticated, full-fledged commercial engines, our prototype takes advantage of open source processing software and endorses mobile technology features for cross-platform dissemination of traffic statistics.

Regarding map-matching of sensed vehicle locations, one class of algorithms is mostly geared towards maximizing *accuracy* of the resulting road identifications. Having a traffic engineering flavor, such approaches (e.g. [19,20]) attempt to minimize any mismatched links and reduce error propagation. A second class primarily focuses on maximizing *throughput* without compromising quality, thus taking a data management perspective. Throughput-oriented techniques can be further characterized either as: (a) *incremental* based solely on positional samples [8,5], when they examine edge distances and orientation of movement so as to greedily expand the existing path with an additional edge for fast response, or (b) *global* [2,3,4,12,21,23], in case that they check against all possible trajectories to find the most similar to the actual movement with better accuracy. More specifically, [4] proposed an ϵ-road-snapped trajectory construction algorithm in a weighted graph representation that returns a path, whose Hausdorff distance to the vehicle trajectory does not exceed location-sensor error ϵ. Based on a combination of spatial, topological and temporal features, a global ST-Matching algorithm in [12] considers GPS trajectories of low sampling rate. Exploiting the evolving trajectories of vehicles, algorithms introduced in [3] consider the entire path of a vehicle instead of its current position only. In essence, they attempt to minimize the Fréchet distance between the trajectory and the constructed sequence of road links; the computed distance also serves as a quality guarantee for the result. Such an approach, possibly enhanced with error estimates and output sensitivity [23], seems suitable for real-time tracking. Without excluding global techniques, an incremental one suits better to our scenario, as the cost of checking with potentially large trajectories would be prohibitive for increased stream rates of GPS readings. Since we presently handle synthetic datasets, for simplicity we take the shortest path between consecutive GPS measurements along the network as a close estimation of the vehicle trajectory, as in [1].

Aside from popular web sources (e.g., Google Maps, Yahoo Maps, Bing Maps, ESRI ArcGIS Online) that mostly visualize digested data, various methods have been proposed for travel time and shortest path estimation specifically for floating car data. Indicatively, algorithms in [6] are based on neural networks and pattern matching and offer short-term predictions. Characterizations of unique traffic patterns per road are addressed in [24], whereas [22] correlates GPS points with neighboring ones in space and time, in order to assess traffic status.

For travel time estimation, [18] combines archived data with live traffic feeds in a streaming-oriented mode. In contrast, we opt for multi-resolution, succinct traffic statistics intended for easily perceptible portrayal and quick notifications.

7 Conclusions and Further Extensions

Overall experience from our functional prototype is more than encouraging. Implementation of EPOPS is still a work-in-progress, but already integrated modules offer concrete evidence for its robustness and low latency. The stream model for processing positional updates, as well as the hierarchical organization of network features have proven decisive factors for achieving timely, concise and multi-grained results. However, the most promising prospect comes from swift integration of modern geospatial, mobile, and web technologies for online delivery of traffic analytics. We anticipate that similar platforms will soon become the chief providers of real-time information for road traffic and safety.

In terms of scalability, it appears that even a full-fledged stream engine shows certain limitations when faced with increasing data volumes and arrival rates. In our extensive tests, TelegraphCQ was proven able to emit results promptly, but we believe that monitoring applications may need to accept information from tens of thousands of vehicles or more. A successful system should adequately fuse complementary tracking data that emanates from diverse devices (inductive loops, cameras etc.) and flows through heterogeneous networks. Not only could such a deployment provide better estimates, but it also opens prospects for custom web services to partners in related domains (logistics, transport, car navigation systems etc.) and social networking (user feedback, quick suggestions etc.). In that respect, traffic stream computing in the cloud would provide a flexible, highly-distributed solution.

We further envisage a probabilistic treatment for addressing the inherent uncertainty, gradual ageing, and transmission delays of positional streams. Traffic forecasts at short-term horizons (like 15, 30, or 60 minutes ahead) could be issued, gracefully weighing online analytics with offline statistics. Developing approximation algorithms in order to get traffic estimates with error guarantees seems a promising and quite challenging topic for research.

Acknowledgements. We would like to thank M.Sc. students Kyriakos Georgoussis and Chrysaida Papadopoulou for their collaboration in developing demo applications for online toll charges and shortest paths over this framework.

References

1. Biem, A., Bouillet, E., Feng, H., Ranganathan, A., Riabov, A., Verscheure, O., Koutsopoulos, H., Moran, C.: IBM InfoSphere Streams for Scalable, Real-time, Intelligent Transportation Services. In: SIGMOD, pp. 1093–1104 (2010)
2. Bouillet, E., Ranganathan, A.: Scalable, Real-Time Map-Matching Using IBM's System S. In: MDM, pp. 249–257 (2010)

3. Brakatsoulas, S., Pfoser, D., Salas, R., Wenk, C.: On Map-Matching Vehicle Tracking Data. In: VLDB, pp. 853–864 (2005)
4. Cao, H., Wolfson, O.: Nonmaterialized Motion Information in Transport Networks. In: Eiter, T., Libkin, L. (eds.) ICDT 2005. LNCS, vol. 3363, pp. 173–188. Springer, Heidelberg (2005)
5. Civilis, A., Jensen, C.S., Pakalnis, S.: Techniques for Efficient Road-network-based Tracking of Moving Objects. IEEE TKDE 17(5), 698–712 (2005)
6. de Fabritiis, C., Ragona, R., Valenti, G.: Traffic Estimation and Prediction Based on Real Time Floating Car Data. In: IEEE ITSC, pp. 197–203 (2008)
7. Geisler, S., Quix, C., Schiffer, S.: A Data Stream-based Evaluation Framework for Traffic Information Systems. In: IWGS, pp. 11–18 (2010)
8. Greenfeld, J.: Matching GPS Observations to Locations on a Digital Map. In: 81th Annual Meeting of the Transportation Research Board, Washington, DC (2002)
9. Güting, R.H., Böhlen, M.H., Erwig, M., Jensen, C.S., Lorentzos, N.A., Schneider, M., Vazirgiannis, M.: A Foundation for Representing and Querying Moving Objects. ACM Transactions on Database Systems 25(1), 1–42 (2000)
10. Hull, B., Bychkovsky, V., Zhang, Y., Chen, K., Goraczko, M., Miu, A., Shih, E., Balakrishnan, H., Madden, S.: CarTel: A Distributed Mobile Sensor Computing System. In: SenSys, pp. 125–138 (2006)
11. Kazemitabar, S.J., Demiryurek, U., Ali, M., Akdogan, A., Shahabi, C.: Geospatial Stream Query Processing using Microsoft SQL Server StreamInsight. In: VLDB, pp. 1537–1540 (2010)
12. Lou, Y., Zhang, C., Zheng, Y., Xie, X., Wang, W., Huang, Y.: Map-Matching for Low-Sampling-Rate GPS Trajectories. In: ACM GIS, pp. 352–361 (2009)
13. Patroumpas, K., Kefallinou, E., Sellis, T.: Monitoring Continuous Queries over Streaming Locations. In: ACM GIS, pp. 521–522 (2008)
14. Patroumpas, K., Sellis, T.: Multi-granular Time-based Sliding Windows over Data Streams. In: IEEE TIME, pp. 146–153 (2010)
15. Patroumpas, K., Sellis, T.: Maintaining consistent Results of Continuous Queries under Diverse Window Specifications. Information Systems 36(1), 42–61 (2011)
16. pgRouting Project, `http://www.pgrouting.org/`
17. TelegraphCQ Project, `http://telegraph.cs.berkeley.edu/telegraphcq/v2.1/`
18. Tufte, K., Li, J., Maier, D., Papadimos, V., Bertini, R., Rucker, J.: Travel Time Estimation using NiagaraST and latte. In: SIGMOD, pp. 1091–1093 (2007)
19. Velaga, N., Quddus, M., Bristow, A.: Developing an Enhanced Weight-based Topological Map-matching Algorithm for Intelligent Transport Systems. Transportation Research, Part C 17, 672–683 (2009)
20. Wang, W., Jin, J., Ran, B., Guo, X.: Large-Scale Freeway Network Traffic Monitoring: A Map-Matching Algorithm Based on Low-Logging Frequency GPS Probe Data. Journal of Intelligent Transportation Systems 15(2), 63–74 (2011)
21. Weber, M., Liu, L., Jones, K., Covington, M.J., Nachman, L., Pesti, P.: On Map Matching of Wireless Positioning Data: A Selective Look-ahead Approach. In: ACM GIS, pp. 290–299 (2010)
22. Wei, L., Peng, W., Lin, C., Jung, C.: Exploring Spatio-Temporal Features for Traffic Estimation on Road Networks. In: Mamoulis, N., Seidl, T., Pedersen, T.B., Torp, K., Assent, I. (eds.) SSTD 2009. LNCS, vol. 5644, pp. 399–404. Springer, Heidelberg (2009)
23. Wenk, C., Salas, R., Pfoser, D.: Addressing the Need for Map-Matching Speed: Localizing Global Curve-Matching Algorithms. In: SSDBM, pp. 379–388 (2006)
24. Yoon, J., Noble, B., Liu, M.: Surface Street Traffic Estimation. In: MobiSys, pp. 220–232 (2007)

A Context-Aware Web Content Generator Based on Personal Tracking

Reinaldo Bezerra Braga[1], Sócrates de Moraes Medeiros da Costa[1], Windson Viana de Carvalho[2], Rossana Maria de Castro Andrade[2,⋆], and Hervé Martin[1]

[1] LIG UMR 5217, UJF-Grenoble 1, Grenoble-INP, UPMF-Grenoble 2, CNRS
38400, Grenoble, France
{braga,herve.martin}@imag.fr, socrates.mm.costa@gmail.com

[2] Group of Computer Networks, Software Engineering and Systems (GREat),
Federal University of Ceara (UFC)
Fortaleza, Ceara, Brazil
windson@virtual.ufc.br, rossana@great.ufc.br

Abstract. Context-awareness has been successfully included in the mobile phone applications due mainly to the presence of numerous sensors and the access to several communication networks. Therefore, we present a Context-Aware Web Content Generator Based on Personal Tracking, which uses the user context information obtained by mobile devices to generate content for a large number of web applications. While registering the trajectory followed by the mobile device, it allows users to create multimedia documents (e.g. photo, audio, video), which are connected to an enriched description of the user context (e.g. weather, location, date). Finally, all this data and documents are combined to produce a new content, which is published on the Web. We also show results of tests performed in a real scenario and describe our strategy to avoid battery overconsumption and memory overflow in mobile phones. Moreover, a user evaluation is presented in order to measure the system performance, in terms of precision and system overall usability.

1 Introduction

Mobile phones, nowadays, are not simple call-making devices anymore. They have already become real information centers. With all the embedded features like GPS, accelerometer, Internet connection, digital camera, among others, a user easily creates and publishes personal multimedia content. For instance, any user can quickly take a picture and put it in his/her web-based photo album. In addition, multimedia content can be enriched and organized with context information collected by smartphones, such as date, geographical position and current weather.

There are several applications that use context information to enrich and organize multimedia documents. This information might be proximity of people or

⋆ Research Scholarship funded by CNPq (process number 314021/2009-4).

objects in the photo, current temperature, date, etc. This type of metadata can be obtained from sensors of mobile devices or from the web. With this information associated, context-aware applications can better organize the multimedia, providing user-friendly visualization of the content, and suggesting annotations for document indexation[1][2][3].

In this paper, we go a step forward proposing the use of context information to generate new multimedia content. First of all, the user trajectory is registered by using the GPS sensor of the device. While registering the trajectory, the user can produce multimedia documents, such as: photos, audio or video. Likewise, context information can be associated to each multimedia created, as geographic position, date, and temperature. These data will be easily shared to the Internet, presented as a microblog, for example. In short, our system works in three steps: i) collecting context and user-added data; ii) processing and organizing them; iii) publishing the composed content on a web-based application (e.g., blogs/microblogs, web albums).

It is also important to mention that context-aware systems have some dependencies that may not be satisfied in some situations. The Internet connection, for example, can be limited or even not available at certain moments. Another problem is related to the mobile device battery. For example, all these features (GPS, Bluetooth, Internet access, etc.) are necessary to the context data acquisition, but they spend too much electric power. In order to minimize these dependencies, we propose some design decisions that have impact in trajectory and context gathering mechanisms.

In the interest of evaluate the system usability and the performance of our gathering mechanisms, we apply it in a challenging scenario. It was used by three of the crewmembers of a boat as a digital logbook. The system registered the boat trajectory, allowed the insertion of photos, suggested annotations using context information and published the content in the blog of the project ZeroCO2[1].

The organization of this paper is presented as follows. Section 2 presents related works and introduces an overview about context-awareness. Section 3 presents our proposed system. Section 4 discusses a case study of our system tested in a real situation. Section 5 presents results of the system performance and user evaluation. Finally, Section 6 concludes this work and gives some perspectives.

2 Context-Awareness Is More Than System Adaptation

Several research areas use the notion of context with distinct meanings.

In the field of information systems, the concept of context refers primarily to the user status and the surrounding environment at the moment he/she is accessing a system. Frequently, the knowledge of the user location is a prerequisite for the success of this kind of system. According to Dey *et al.* [4], the context is constructed from all information elements that can be used to characterize the situation of an entity. An entity is defined as any person, place or thing (including users and the own applications) considered relevant to the interaction

[1] `www.zeroco2sailing.com/blog/`

between the user and the application. Consequently, the term "context-aware" is associated with systems that guide their behavior according to their context of use. Most authors in this field consider context awareness as the ability to perceive the situation of the user in several aspects, and adapt the system behavior accordingly [5].

On the other hand, in the multimedia domain, the notion of context, and mainly, its exploitation is slightly different. Context-awareness is more than simple adaptation mechanisms. This distinction is studied in some works, such as Naaman et al. [6], which presented the behavior of users to organize and find photos. In fact, most of the information referred by people about their image memories consists of aspects related to their context at the moment the photo is taken (when, where, with who, etc.). These authors argue that the information about the context creation of a photo facilitates the search of a specific photo in a set of multimedia documents.

The popularization of mobile devices equipped with location sensors and GIS (Geographical Information Systems) have provided the technology and data necessary to develop multimedia systems able to gather the desired context information. Nowadays, we can categorize these context-aware multimedia systems in three groups: multimedia organization and annotation tools; multimedia sharing systems; and context sharing systems.

2.1 Multimedia Organization and Annotation Tools

Following the aforementioned concepts in Naaman *et al.* [6], some research projects and commercial applications propose automatic photo annotation by using context metadata. In fact, nowadays, the use of photo geotagging is not unusual for mobile users since most of the smartphones contain geotagging applications. For example, in Kennedy *et al.* [7], the authors identified local markers from 110,000 Flickr images of the San Francisco Bay Area. Most of the photos were taken from mobile phones and were georeferenced. Hence, image data with views that best represent a marker according to visual similarity were retrieved by means of a marker or location search.

Research projects, such as PhotoGeo [3], PhotoCopain [8], MediAssist [2], and PhotoMap [9] gather a larger set of contextual metadata, which includes user location, identity of nearby objects and people, date, season and temperature. They exploit these contextual metadata for photo organization, publication and visualization. For instance, PhotoMap provides automatic annotation about spatial, temporal and social contexts of a photo (i.e., where, when, and who was nearby). PhotoMap also offers a Web interface for spatial and temporal navigation in photo collections. The system exploits spatial Web 2.0 services to show where a user took the photos and the itinerary followed when taking them.

2.2 Multimedia Sharing Systems

The modern capabilities of mobile devices and the success of Web 2.0 sites stimulate a new kind of multimedia phenomenon: the create-to-share behavior

[10]. Mobile users create multimedia using their devices and with the purpose of sharing the information almost instantly.

Some context-aware systems try to exploit context metadata to increase this experience of multimedia sharing [10] [11]. For instance, Zonetag projects [10] use the position information to suggest the photo annotation before sharing it. Other approaches aim to refine the multimedia content taking into account the user context. For example, the Aware project [11] replaces the MMS application in Nokia mobile phones by a context-aware application, which adds automatically the position information to each MMS sent by the user, such as an address derived from the combination of a GSM Cell-ID and an address database.

2.3 Context Sharing Systems

A large number of messages shared on social networks, such as FourSquare and Twitter microblogs, refers to the information of user context. Hence, this information can be derived automatically by mobile phones equipped with sensors [12] and published on these Web sites. For instance, ContextWatcher [13] is a mobile application to capture and share the most common context information. The main objective is to acquire and describe accurately the current status of the user. The context information of a user is composed of position (e.g., geographic coordinates, altitude and address), speed, humor, heartbeats and weather. All this information is combined and published over a map-based site that shows the current context of all users.

Other approaches, such as Melog [14] and SnapToTell [15], propose the generation of more complex multimedia documents from a set of pictures created by users and the context information associated to them. For instance, Melog tries to recognize events by using clustering techniques. The identified events are used to structure a micro-blog about the user travels.

3 Our Approach

Taking into account the classification of context-aware groups, our proposal can be classified into groups one and three. Figure 1 presents an overview of our system, which is divided in three main parts: Data Acquisition, Data Processing and Publishing. Data Acquisition concerns the sensor application, note writing, data capturing and every other data collection process. After that, all acquired data will be processed in the Data Processing. In this part, the system uses the raw data in order to capture inferred information and to suggest a textual annotation. When the user context is properly collected and inferred, the Publishing part initiates its process. Finally, a new content is formerly produced and can be shared in the Internet, taking into account the association of each context information.

For instance, a user is registering the trajectory of his/her boat trip using our system. While he/she is arriving in the harbor, he/she decides to take a photo of another boat. At this moment, besides the photo, the sensors acquires context

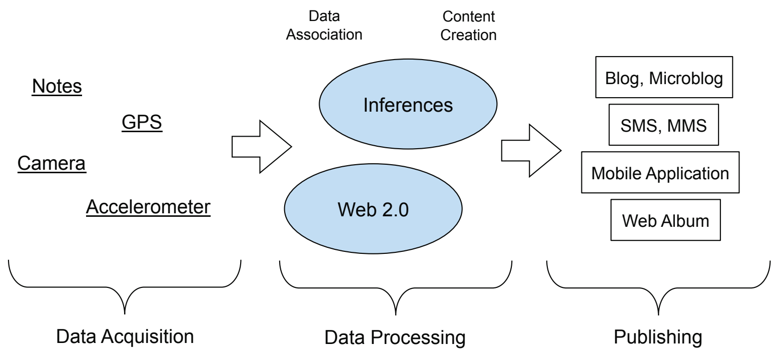


Fig. 1. System Overview

information, such as position, direction, speed and weather[2]. The collected data is manipulated by the second part in order to acquire inferred information and to suggest textual annotations to the user. After validating the annotations and association with the photos, the user can visualize his/her augmented trajectory and publish the content on the web.

Nowadays, one of the main features of context-aware systems is the location tracking. Our system also relies in this feature. It gets the mobile device position periodically by GPS and derives the trajectory followed by the mobile. In addition, GPS collects the geographic location of a taken picture to add this information in the metadata. This action is important to help the content generation as well as to acquire new information (e.g., weather) of a photo that was previously taken. These three parts of our system are detailed in the next sections.

3.1 Data Acquisition

One of the most important parts of our system is the data acquisition. It uses the sensors in the mobile device to get information about localization, device orientation, speed, time, etc. In addition, some initial notes made by the user are also considered as *Data Acquisition.*

During the data acquisition process, we have to do the relations between each information collected, as presented in Figure 2. According to the Figure, the user starts the data acquisition process in the mobile device. The tracking mechanism, then, begins to register the user trajectory. While the tracking mechanism is running, the user decides to take a photo, creating a new event. At the moment, the parameters of the digital camera are defined. Besides that, the context information is gathered using sensors. Other kind of information can be obtained if an Internet connection is available, such as the location name and weather conditions, both using the position information acquired by the GPS.

[2] Weather will be acquired if an Internet connection is available.

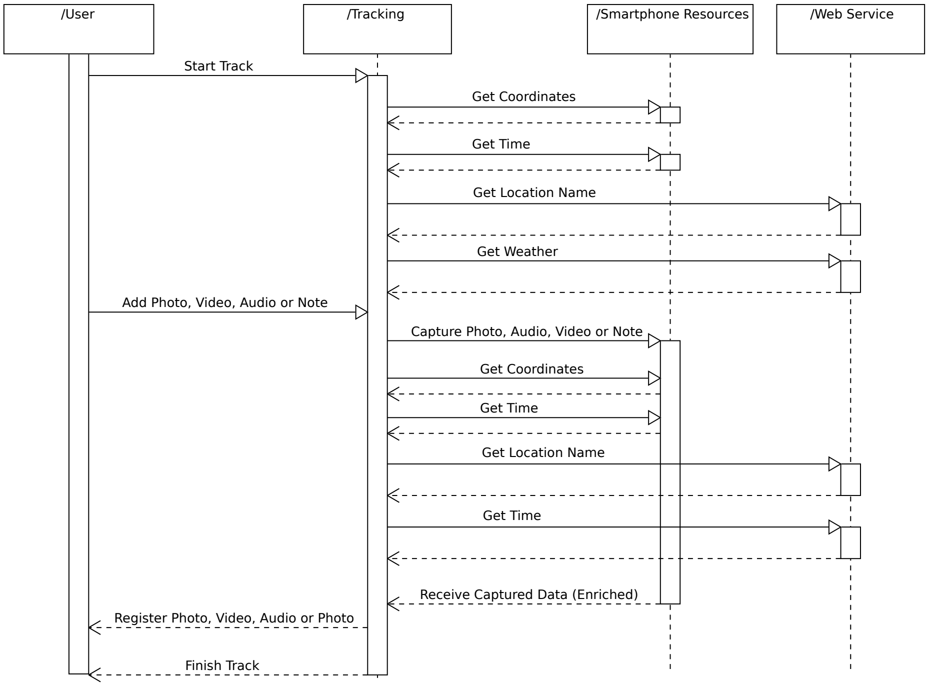


Fig. 2. Sequence Diagram of Data Acquisition

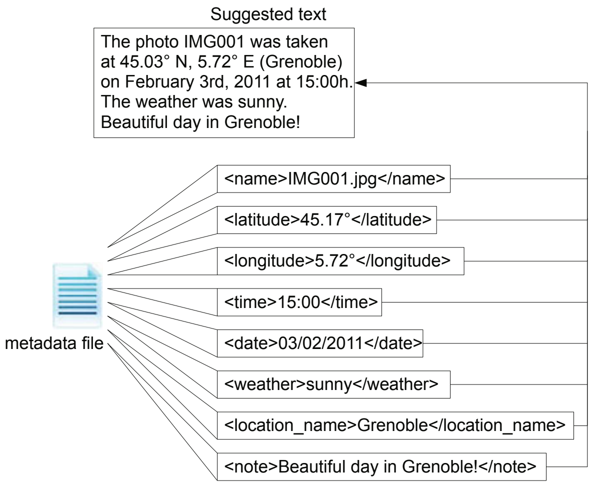
Since we are working with mobile devices, the efficiency of our system is directly related with the battery consumption. To reduce the overconsumption of battery, we propose the insertion of a distance filter in the tracking mechanism. The key idea is to avoid registering coordinates for short distances. Consequently, we have to observe what is the best distance filter value to register the coordinates. For example, we define the distance filter equal to 10 meters, the mobile device will register the current position in the metadata if it is higher than 10 meters. Otherwise, it will be dropped. The distance filter is an important feature of our system because it is responsible for the relation between battery consumption and trajectory construction.

In order to improve the data processing step, it is important to organize the acquired data into the metadata. Therefore, we used *tags* to arrange each information in the metadata.

3.2 Data Processing

The *Data Processing* is the real core of the system. It is responsible to increase the robustness of our system by offering more than a context-aware data collector, as follows. It associates, suggests and organizes the information in order to provide a comprehensive structure to be published.

Making use of the acquired data organized by tags, the data processing part is started. It has to provide an interface for users to facilitate the content generation. The key idea is to use the context information of a data to suggest the text that will be published. Our system provides an initial recommended text based on the information acquired by the mobile application. For example, if a user is registering his/her trajectory and takes a photo in a specific position, the system will generate a new photo with the name *IMG0001* and will register the coordinates *45º10'0"N, 5º43'0"E* at *15:00* on *03/02/2010*. Besides that, the user adds a note describing some characteristics of the photo. According to Figure 3, a text is suggested for each acquired data. Following the previously example, if the user selects the photo $IMG0001$ in our system interface, then a new text might be suggested: *"The photo IMG0001 was taken on the location* $< location\_name >$ *at 15:00 on 03/02/2010 and the weather was* $< weather\_status >$*.* $< additional\_note >$*"*.



Suggested text
The photo IMG001 was taken at 45.03° N, 5.72° E (Grenoble) on February 3rd, 2011 at 15:00h. The weather was sunny. Beautiful day in Grenoble!
metadata file
<name>IMG001.jpg</name>
<latitude>45.17°</latitude>
<longitude>5.72°</longitude>
<time>15:00</time>
<date>03/02/2011</date>
<weather>sunny</weather>
<location_name>Grenoble</location_name>
<note>Beautiful day in Grenoble!</note>

Fig. 3. Data Processing

If the mobile device due to an absence of connection does not acquire the information of location name and weather, our system interface has to be able to obtain this information based on context information. Nevertheless, the specialized web services provide the weather status for present and future times. To solve this problem, we propose a mechanism to capture this information using a HTML parser in order to get the location name and weather status for the past time. This parser reads the web page $DailyHistory$ of the $WeatherUnderground$ and obtains the weather status related to the context information provided by the acquired data. When the user generates the content to describe all events registered during his/her trajectory, the third step of our system can be started.

3.3 Publishing

The last part of the system is responsible for publishing the content produced. In Figure 1, we have proposed some applications to publish the content, such as microblog, SMS, mobile application, etc. In spite of the existence of a large number of applications to publish the content generated by our system, we choose the web content publication on blogs/microblogs because of their natural manner to publish the web content. Their structure, based on individual posts, is perfect to publish a data with context information. We can use the natural content organization to sort the posts in terms of the context information. Moreover, the user could view the content organized by day or by place, for example.

In addition, we propose a map-based interface and pop up windows in order to present the content (annotations, photos, audio, video and context information related to a position) in the trajectory. We intend to use map-based interfaces taking into account the usability studies presented in the literature [16][2]. These works show that map interfaces demonstrate more interactivity advantages than browsing information with hierarchical links. Moreover, with a map-based interface, we can easily illustrate the trajectories generated by the mobile users together with the context information.

4 Using the Proposed System in a Real Situation

To evaluate the efficiency of our system in a real situation, we implemented our proposal for the ZeroCO2 project [17]. We designed our system to be a digital logbook during a boat expedition around the Mediterranean Sea. The logbook, which was created as a book to record readings from the ship log [18], is an essential instrument to the navigation and has to be used daily. In general, the crew uses paper-based logbooks to register all information and, frequently, the information is collected from distinct equipments. Hence, we concluded that our system was able to create a complete logbook for this boat expedition. In addition, the challenging scenario of the sea added some problems involving the recurrent absence of Internet connection and the lack of battery charging.

Our system was responsible to track the trajectory followed by the boat, adding all context information to each registered coordinates. Although our system proposes the use of audio, video and photo as data, we used only photos for this first experiment in the project ZeroCO2. Taking into account this scenario, we face new challenges that have motivated us to improve the context-aware system proposed in the previous section.

4.1 Challenges and System Improvements

When using any context-aware application in the sea, we have to handle new challenges in order to avoid problems in the application and information loss. The main difficulties that we consider in this work are detailed as follows.

- **Lack of Continuous Internet Connection.** In a sea expedition, frequently, there is an absence of Internet connection on mobile phones. 3G or 4G signal is perceived only when the ship is near the coast. Some high-level context information, like a location address, can be computed in a future moment (i.e., when an Internet connection is available). However, some other data cannot be easily recovered. For instance, Weather Forecast services only provide real-time information. For this reason, a special context "cache" system should be designed for providing past context information.
- **Robustness.** the absence of Internet connection and the movable nature of a ship expedition make the remote repair of the mobile application impossible or in situ. Thus, the mobile application has to be reliable. Previously, our research team has also developed context-aware multimedia systems following the architectures of PhotoMap [9] and CoMMediA (Context-Aware Mobile Multimedia Architecture) [1]. Therefore, we tested both projects that adopt Java Mobile Edition as mobile platform. In the user tests of these systems, some memory overflow incidences occurred caused by simultaneous access to the GPS sensor and the camera phone. This problem occurs even when using synchronized threads, and, sometimes, requires redeployment of the mobile application.
- **Energy Limitation to Recharge Mobile Devices.** Another critical problem found in the PhotoMap and CoMMediA projects was heavy energy consumption during the use of the mobile application. For instance, in forty minutes, the battery of a Nokia N95 was fully discharged since GPS, photo camera and Bluetooth sensors are greedy in energy consumption. In some ship expeditions, electric energy restrictions are present and the mobile application should be designed to overcome this issue.

Based on the system overview presented in Figure 1, we decided to divide the digital logbook in three parts: a mobile application to register the boat trajectory; a desktop application to receive the acquired data and generate the web content; and a blog to publish the content.

An overview of the digital logbook is presented in Figure 4. The *Data Acquisition* process was developed in the iOS platform, since the user-friendly interaction is well known in this mobile platform. The mobile application performs the boat tracking, take the photos, and carries out the relation among each data and its context information. It is important to note that some of *Data Processing* features were also implemented in the mobile phone, such as the inference mechanism to get the weather status and location name.

The *Data Processing* step was implemented as a desktop application to offer an interface of creation and publication of web content. It implements the module to get the context information that was not acquired by the mobile application, using the HTML parser. Another important feature in the desktop application is the function of text suggestion for each photo. We tried to develop a robustness and intuitive interface user interface to improve the usability.

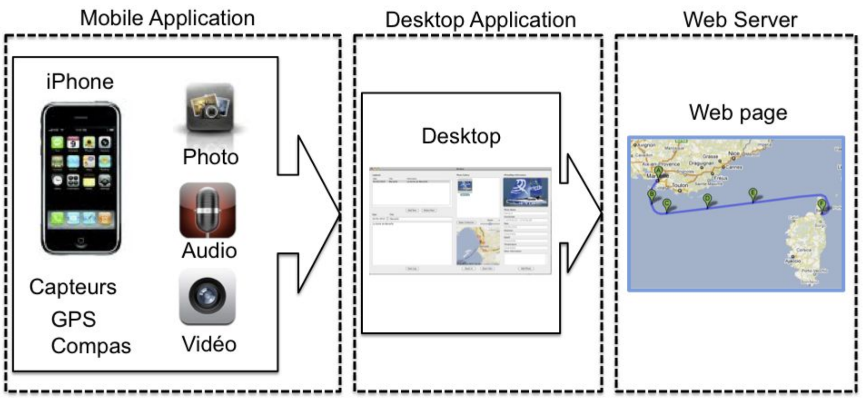


Fig. 4. Digital logbook parts

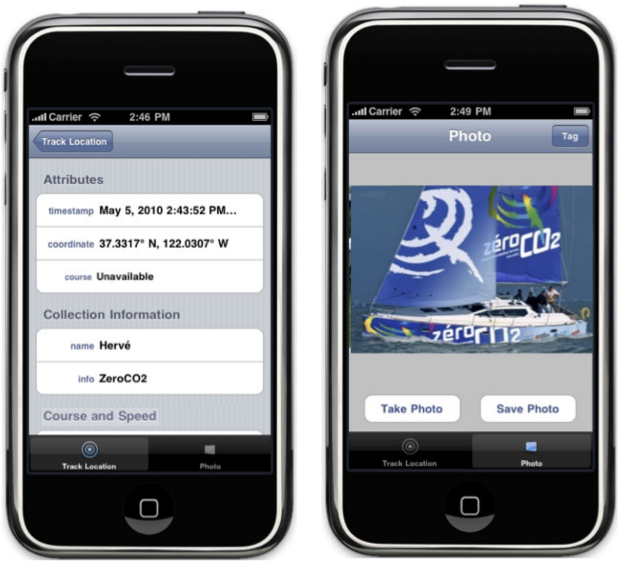
The *Publishing* step was developed using an open source blog solution, which is available on the web page of the ZeroCO2 project[3]. It shows the complete digital logbook, containing the content generated by the crewmembers and the map with the boat trajectory.

4.2 Mobile Application

The mobile application interface is shown in Figure 5. As we can observe, there are two main functions: the tracking mechanism and the digital camera. The tracking mechanism is responsible for registering the geographic coordinates to construct the trajectory. The interface shows the position, speed, date and, if Internet connection is available, wind speed and humidity. The digital camera takes a picture and, automatically, adds the context information to it. There is also the option "Tag" with which you can add the information manually.

An important result discovered during our tests is related to the use of metadata following some standard, such as Web Ontology Language OWL [19]. Several solutions adopt this language to obtain inferred information about a context. However, it needs to add a large number of information in the metadata file to perform this task. Consequently, the mobile application generates several unused information into the metadata file, causing some problems of memory overflow in the mobile application. Therefore, we optimized the content of our metadata files registering only the relevant information. Besides that, we developed our own local parser to get the information of each tag and to infer about context information using the HTML parser.

[3] www.zeroco2sailing.com/blog/



Attributes

timestamp	May 5, 2010 2:43:52 PM...
coordinate	37.3317° N, 122.0307° W
course	Unavailable

Collection Information

name	Hervé
info	ZeroCO2

Course and Speed

(a) Tracking mechanism. (b) Digital camera.

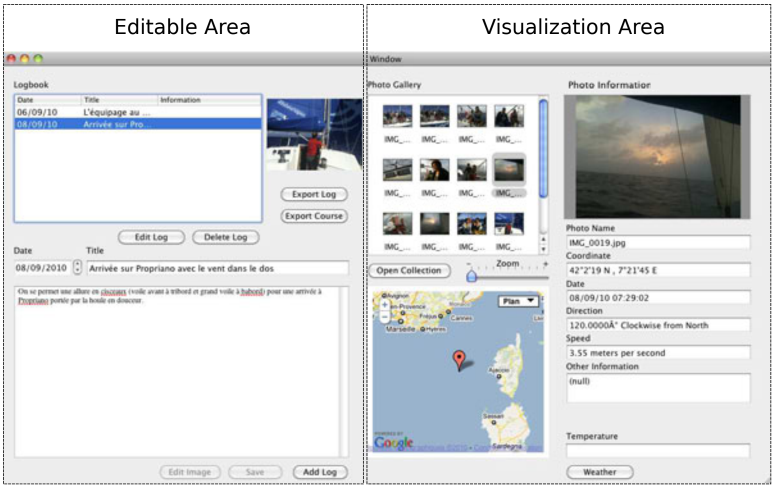
Fig. 5. Tracking mechanism and digital camera

4.3 Desktop Application

The desktop application interface is shown in Figure 6. As stated, the desktop application has two segments: the editing area and the visualization area. In the editing area, the user can add an annotation based on the text suggestion function of our system (Figure 3). Besides that, the user is able to edit previously annotations. In the visualization area, the application shows the photo album jointly with all context information about each selected photo. A small map shows the position where the selected photo was taken. In addition, this interface permits that the user captures weather status of a photo, in case this information was not captured at the moment the photo was taken, due to an absence of Internet connection. This is only possible due to our proposed HTML parser (Section 3.2).

4.4 Web Application

The web application is a microblog solution, in which each annotation created by the crewmember is posted. Some parts of the blog are shown in Figure 7. All content generated by the user in the desktop application is stored in a MySQL database and consulted by Java and PHP scripts. In addition to the illustrated posts, it also presents a map showing the trajectory of the boat. This map was developed with the Google Maps API [20]. We developed our map-based interface taking into account the usability studies presented in the literature [2] [16].



Date	Title	Information
06/09/10	L'équipage au ...	
08/09/10	Arrivée sur Pro...	

Date: 08/09/2010
Title: Arrivée sur Propriano avec le vent dans le dos

On se permet une allure en ciseaux (voile avant à tribord et grand voile à babord) pour une arrivée à Propriano portée par la houle en douceur.

Photo Name: IMG_0019.jpg
Coordinate: 42°2'19 N , 7°21'45 E
Date: 08/09/10 07:29:02
Direction: 120.0000Â° Clockwise from North
Speed: 3.55 meters per second
Other Information: (null)
Temperature:

Fig. 6. Desktop Application Interface

The web application also performs an indexation process to improve browsing and interaction procedures. The amount of multimedia and context information increments quickly in our system. Then, to avoid future performance difficulties related to the large number of access, spatial and temporal indexes are associated with each annotation in the MySQL database.

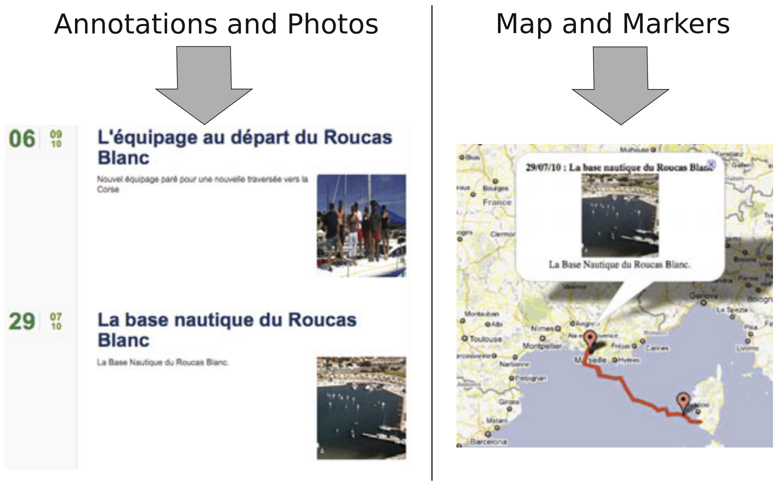
5 Results

In this section we describe the performance and user evaluation of our system.

5.1 Performance Evaluation

The first evaluation was done during a travel around the Marseille coast. We ran this first test to calibrate the distance filter option and to execute the performance evaluation in the mobile phone. As explained before, this option is responsible to define the detail level of the trajectory. We assigned the value fifty meters to the distance filter, which means that a position will be registered if it is higher than fifty meters in comparison with the last position registered. With the first results, we refined our system to the second test: a travel from Marseille to Ajaccio (Corsica Island).

Figure 8 shows the performance evaluation of our application in the mobile phone during the interval from 26 to 29 minutes. The evaluation was conducted during the first tests, using the XCode Instruments [21] version 2.7. We observed that the *Total Load* (i.e., System and User) and the *Physical Memory Free* followed the same behavior while the mobile application functions were in operation. According to the results, the tracking mechanism requires approximately 10% of the memory and 25% of the processing to capture and register the positions. Likewise, while the iPhone digital camera is working, the memory used is approximately equal to 80% and the total load did not change. After taking the photo, the function *Save Photo* can be selected. When the *Save Photo* function is activated, the maximum load is used to associate and register all data and context information in the hard disk. Finally, the memory is cleaned when all data and information are associated and saved and the total load returns to follow the tracking mechanism. These results were important to guarantee that the user can use the application for a long time without stopping it due to memory or processing overhead problems.



06 09 10 L'équipage au départ du Roucas Blanc

Nouvel équipage paré pour une nouvelle traversée vers la Corse

29 07 10 La base nautique du Roucas Blanc

La Base Nautique du Roucas Blanc.

29/07/10 : La base nautique du Roucas Blanc

La Base Nautique du Roucas Blanc.

Fig. 7. Web Application

Other important results are related to the mobile phone battery consumption during the trajectory registration. In the first test, when the distance filter had been configured to register each movement of the user, the iPhone battery level was down to 10% after 2 hours. After setting the distance filter to fifty meters, the iPhone battery level was down to 10% after 3 hours. Another factor that can affect this result is the frequency that photos are captured.

5.2 User Evaluation

Once the first distance filter adjustments were performed, our system was used by three ZeroCO2 crewmembers. After a one-week expedition, the users filled in a general usability survey. Despite the small number of users, we have tried, with this questionnaire, to measure the main benefits and issues of our context-aware annotation proposal. We also wanted to know if using a mobile phone in an "adverse environment" could disturb the real ZeroCO2 missions. The following survey questions were asked:

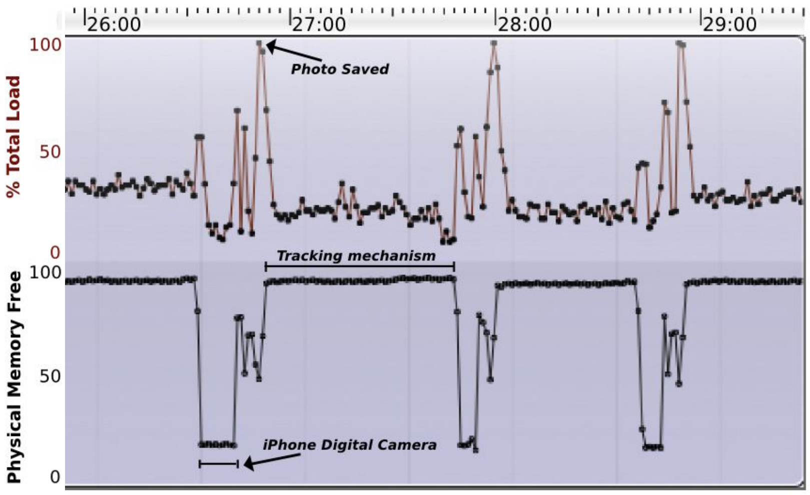


Fig. 8. Physical Memory Free and Total Load in the iPhone

– Rate how easy it was to create a digital logbook without and with our digital logbook.
– Rate how fast it was to create a digital logbook without and with our digital logbook.
– How do you qualify the accuracy of the digital logbook generated annotations?
– Could you describe the main digital logbook advantages and shortcomings?

For the first three questions, a five-scale graph was provided in which the number one corresponds to a very bad rate, and five corresponds to very a good rate. For instance, for question 1, the number one corresponds to very difficult, and five corresponds to very easy. Figures 9 and 10 show the experiments results for the first two questions.

Without our system, the crewmembers have to synchronize all information collected by a digital camera with a desktop application (e.g., a word processor) in order to create a digital logbook. Additionally, another step has to be performed for publishing the logbook information on the web (e.g., using a blog authoring tool). With a mobile device and the digital logbook, most of the processes of logbook creation, edition, and publishing are automated by our proposal. The survey results presented in Figures 9 and 10 reflect the differences between these two approaches. Interestingly, two users have given a greater difference in scores concerning the time question (Figure 10), which shows how fast it is to publish information with our digital logbook.

Regarding the accuracy question, two users have scored "precise", and the other one has scored "very precise". Despite the use of distance filter option, one can see that the generated annotation is still very satisfactory for the users.

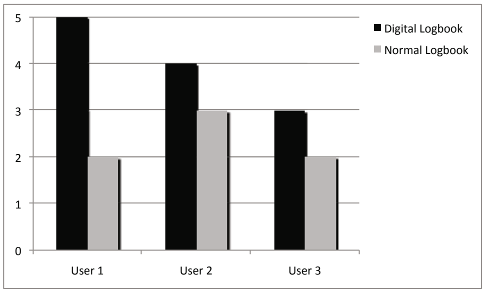


Fig. 9. Easiness comparison between our digital logbook and normal logbook tools

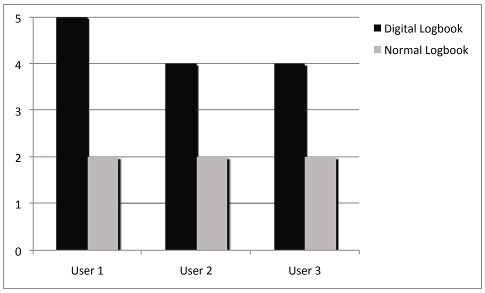


Fig. 10. Annotation time comparison between our digital logbook and normal logbook tools

For question 4, the users have highlighted the advantages of intuitive interface on the iPhone application, and the simplicity and speed for logbook creation. None of the users have mentioned disruption on their daily missions. However, the synchronization between the iPhone and the Mac book was pointed out as the main drawback. Two users have even suggested skipping this step by editing the information on the iPhone and publishing them directly on the Web.

With these results in mind, the generation of context-aware annotation is, as we expected, a useful way to automate multimedia edition and publishing even in an "adverse environment".

6 Conclusion

In this paper, we presented a new context-aware web content generator based on personal tracking. It is a context-aware system for the creation, annotation and sharing of multimedia content. We showed that our solution was used as a wizard editor for the generation of a real digital logbook. By designing a practical and efficient strategy, our system provided a user-friendly interface and offered a mechanism for context acquisition that avoids battery overconsumption and memory overflow.

Usability and performance tests were also performed in collaboration with ZeroCO2 project. The user evaluation results demonstrated that the crewmembers were comfortable using our system and found it an excellent tool to accurately publish context information according to the geographical position. Beyond our approach for context-aware systems, another important contribution is associated with the development of a context-aware photo management tool on smartphones.

As future work, we aim to offer a framework for the development of context-aware systems. This framework will provide a collection of procedures able to acquire, store, increase and infer contextual metadata related to multimedia document. The key idea is to reuse our proposal in several types of scenarios, for example: tracking an excursion in forests and mountains; studying the behavior pattern of a vehicle based on its speed, course, and position; mapping the course of runners and other athletes; and applying that for mobile learning lectures such as Geology courses that are usually taken in the field.

Ackowledgements. This work is partially funded by both French Ministry of Higher Education and Brazilian Ministry of Education under the Program of Scientific Cooperation called STIC-AmSud. The sponsored project is entitled Learning While Moving (LWM) and the authors gratefully acknowledge this support.

References

[1] Viana, W., Miron, A., Moisuc, B., Gensel, J., Villanova-Oliver, M., Martin, H.: Towards the semantic and context-aware management of mobile multimedia. Multimedia Tools and Applications, 1–39 (2010), doi:10.1007/s11042-010-0502-6

[2] O'Hare, N., Smeaton, A.F.: Context-aware person identification in personal photo collections. Trans. Multi. 11, 220–228 (2009)

[3] de Figueirêdo, H., Lacerda, Y., de Paiva, A., Casanova, M., de Souza Baptista, C.: Photogeo: a photo digital library with spatial-temporal support and self-annotation. Multimedia Tools and Applications, 1–27 (2011), doi:10.1007/s11042-011-0745-x

[4] Abowd, G.D., Dey, A.K., Brown, P.J., Davies, N., Smith, M., Steggles, P.: Towards a Better Understanding of Context and Context-Awareness. In: Gellersen, H.W. (ed.) HUC 1999. LNCS, vol. 1707, pp. 304–307. Springer, Heidelberg (1999)

[5] Marinho, F.G., Lima, F., Ferreira Filho, J.B., Rocha, L., Maia, M.E.F., de Aguiar, S.B., Dantas, V.L.L., Viana, W., Andrade, R.M.C., Teixeira, E., Werner, C.: A Software Product Line for the Mobile and Context-Aware Applications Domain. In: Bosch, J., Lee, J. (eds.) SPLC 2010. LNCS, vol. 6287, pp. 346–360. Springer, Heidelberg (2010)

[6] Naaman, M., Harada, S., Wang, Q., Garcia-Molina, H., Paepcke, A.: Context data in geo-referenced digital photo collections. In: Schulzrinne, H., Dimitrova, N., Sasse, M.A., Moon, S.B., Lienhart, R. (eds.) ACM Multimedia, pp. 196–203. ACM (2004)

[7] Kennedy, L.S., Naaman, M.: Generating diverse and representative image search results for landmarks. In: WWW 2008: Proceeding of the 17th International Conference on World Wide Web, pp. 297–306. ACM, New York (2008)

[8] Tuffield, M.M., Harris, S., Brewster, C., Gibbins, N., Ciravegna, F., Sleeman, D., Shadbolt, N.R., Wilks, Y.: Image annotation with photocopain. In: Proceedings of Semantic Web Annotation of Multimedia (SWAMM 2006) Workshop at the World Wide Web Conference, WWW 2006, pp. 22–26 (2006)

[9] Viana, W., Filho, J.B., Gensel, J., Villanova Oliver, M., Martin, H.: PhotoMap – Automatic Spatiotemporal Annotation for Mobile Photos. In: Ware, J.M., Taylor, G.E. (eds.) W2GIS 2007. LNCS, vol. 4857, pp. 187–201. Springer, Heidelberg (2007)

[10] Ames, M.: Why we tag: motivations for annotation in mobile and online media. In: CHI 2007: Proceedings of the SIGCHI Conference on Human Factors in Computing Systems, pp. 971–980. ACM Press (2007)

[11] Raento, M., Oulasvirta, A., Petit, R., Toivonen, H.: Contextphone: A prototyping platform for context-aware mobile applications. IEEE Pervasive Computing 4, 51–59 (2005)

[12] Barkhuus, L., Brown, B., Bell, M., Sherwood, S., Hall, M., Chalmers, M.: From awareness to repartee: sharing location within social groups. In: Proceeding of the Twenty-sixth Annual SIGCHI Conference on Human Factors in Computing Systems, CHI 2008, pp. 497–506. ACM, New York (2008)

[13] Koolwaaij, J., Tarlano, A., Luther, M., Nurmi, P., Mrohs, B., Battestini, A., Vaidya, R.: Context Watcher-Sharing context information in everyday life, Calgary, Canada (2006)

[14] Li, H., Hua, X.S.: Melog: mobile experience sharing through automatic multimedia blogging. In: Proceedings of the 2010 ACM Multimedia Workshop on Mobile Cloud Media Computing, MCMC 2010, pp. 19–24. ACM, New York (2010)

[15] Cemerlang, P., Lim, J.H., You, Y., Zhang, J., Chevallet, J.P.: Towards automatic mobile blogging. In: 2006 IEEE International Conference on Multimedia and Expo, pp. 2033–2036 (2006)

[16] Ryu, D.S., Chung, W.K., Cho, H.G.: Photoland: a new image layout system using spatio-temporal information in digital photos. In: SAC 2010: Proceedings of the 2010 ACM Symposium on Applied Computing, pp. 1884–1891. ACM, New York (2010)

[17] Joseph Fourier University, Atomic Energie Commission (CEA), Floraris and RM Fora Marine: Zeroco2 project (2010)

[18] May, W.E., Holder, L.: A history of marine navigation, by W. E. May; with a chapter on modern developments by Leonard Holder. Foulis, Henley on Thames (1973)

[19] Dean, M., Schreiber, G.: OWL web ontology language reference. W3C recommendation, W3C (2004)

[20] Google Inc.: The Google Maps/Google Earth APIs (2010)

[21] Apple Inc.: Instruments for Performance Analysis - Version 2.7 (2010)

A Holistic Semantic Similarity Measure for Viewports in Interactive Maps*

Andrea Ballatore[1], David C. Wilson[2], and Michela Bertolotto[1]

[1] School of Computer Science and Informatics
University College Dublin, Belfield, Dublin 4, Ireland
{andrea.ballatore,michela.bertolotto}@ucd.ie
[2] Department of Software and Information Systems
University of North Carolina
9201 University City Boulevard, Charlotte
NC 28223-0001, USA
davils@uncc.edu

Abstract. In recent years, geographic information has entered the mainstream, deeply altering the pre-existing patterns of its production, distribution, and consumption. Through web mapping, millions of online users utilise spatial data in interactive digital maps. The typical unit of visualisation of geo-data is a viewport, defined as a bi-dimensional image of a map, fixed at a given scale, in a rectangular frame. In a viewport, the user performs analytical tasks, observing individual map features, or drawing high-level judgements about the objects in the viewport as a whole. Current geographic information retrieval (GIR) systems aim at facilitating analytical tasks, and little emphasis is put on the retrieval and indexing of visualised units, i.e. viewports. In this paper we outline a holistic, viewport-based GIR system, offering an alternative approach to feature-based GIR. Such a system indexes viewports, rather than individual map features, extracting descriptors of their high-level, overall semantics in a vector space model. This approach allows for efficient comparison, classification, clustering, and indexing of viewports. A case study describes in detail how our GIR system models viewports representing geographical locations in Ireland. The results indicate advantages and limitations of the viewport-based approach, which allows for a novel exploration of geographic data, using holistic semantics.

Keywords: Geographic Information Retrieval, Viewport, Holistic semantics, Geo-semantics, OpenStreetMap, Vector space model.

1 Introduction

The term *neogeography* aptly describes the recent explosion of novel geographic practices, involving the mass production and consumption of geographic information over the Internet [31]. One of the most striking aspects of this nexus

* Research presented in this paper was funded by a Strategic Research Cluster grant (07/SRC/I1168) by Science Foundation Ireland under the National Development Plan. The authors gratefully acknowledge this support.

of phenomena is the rapid diffusion of interactive web maps, enabled by enhancements in web technologies in the early 2000s [6]. In parallel, the amount of spatially-referenced information available online has kept increasing at an explosive rate, resulting in spatial information overload.

In order to satisfy the users' spatial information need, the development of effective geographic information retrieval techniques has become a core effort for both academia and industry. The discipline of text information retrieval emerged to find documents matching desired criteria in large collections [17]. Similarly, geographic information retrieval (GIR) aims at identifying relevant geographic objects in vast dataset, indexing locations, toponyms, and minimum bounding rectangles [12,22]. Several efforts have been made to enrich GIR systems with geographic knowledge, linking geographic data to ontologies [10,15,2,14]. However, as Leveling puts it, large-scale evaluations indicate that "more geographic knowledge typically had little or no effect on performance of GIR systems or that it even decreases performance compared to traditional (textual) information retrieval baselines" [12, p.29].

Popular GIR systems, such as Google Maps, Bing Maps, and Yahoo! Maps,[1] focus on the indexing of aspects of the geographic features, to allow efficient retrieval of individual features based on their textual meta-data, such as place name and street address [21]. The user is presented with a rectangular frame often called *viewport*, containing a pre-rendered image of a map that displays features based on their visibility at the current scale. Several actions can be applied on the web map, including text searches, panning - changing the bounding box location - and zooming - changing the map scale level in discrete, predefined steps. Beyond the details of each system, geographic data is most typically visualised and consumed in viewports, rendered at a specific map scale. In a trial-and-error process, users perform actions in order to satisfy their spatial information need [22].

Such web GIR systems can be utilised to examine aspects of objects, for example to observe the structure of a large building or a lake. This type of cognitive activity is generally considered to be an analytical process: complex objects are divided into their constituent parts and mutual relationships, in order to reach the desired piece of information. It has been argued that analytical thinking dominates Western culture [20,19]. This predominance notwithstanding, various forms of holism have emerged in psychology, cognitive science, and geography as an important mode of perception, learning, and thinking [33,25,1]. In particular, the field of landscape ecology strongly claims that landscape, as Antrop and Van Eetvelde put it, "should be considered a complex whole that is more than the sum of its composing parts" [1, p. 43].

In a holistic process, the emphasis is not on individual objects but on the set of objects considered as a unified whole. In GIR, users often judge the *overall semantic content* of a large area to evaluate it against their information need. Holistic judgements on geographic areas are done in several instances. For

[1] `http://maps.google.ie,http://www.bing.com/maps,http://maps.yahoo.com` (accessed on 24/1/2012).

example, a user might want to evaluate possible areas when looking for houses on sale. If the user is interested in seaside towns, they want to retrieve areas matching an overall semantic content, such as a small urban settlement located on the coast, with a high density of amenities, beaches, etc, without focusing on specific, individual features. Similarly, a geographer might compare several viewports to study large-scale phenomena affecting the landscape. As in these use cases the focus is on entire viewports rather than on specific features, users can benefit from a holistic semantic query tool.

For all the aforementioned reasons, we think that a computable measure of semantic similarity *between viewports*, taken as holistic units of geographic information, and not between individual features, offers a different approach to GIR. However, this approach does not aim at superseding text-based retrieval, but rather at offering an additional tool that can be integrated with existing GIR methods. In this paper we describe a novel technique to extract holistic semantic descriptions of viewports, and the computation of their similarity, as a foundation of a holistic, viewport-based GIR system.

The remainder of this paper is organised as follows: Section 2 surveys related work in the area of GIR, viewports, and holistic cognition. Section 3 reports the core of the proposed approach, while Section 4 illustrates a case study walk-through in the computation of holistic semantic descriptors in a typical web map. Finally section 5 presents concluding remarks, and outlines directions for future research.

2 Related Work

Web mapping is one of driving technologies that brought geographic information into the mainstream, enabling the explosion of neogeography since 2005 [31]. Haklay et al give an account of the recent developments in Internet web mapping, including map mash-ups, crowdsourcing, geostack, and folksonomies, under the umbrella-term 'Web Mapping 2.0' [6]. A major innovation is that map users are not only consumers of geographic information, but also producers of the so-called 'Volunteered Geographic Information' [5].

In such web mapping services, geographic data is distributed through a viewport, a rectangular viewing frames that represent a geographic area at a given scale. Typically, users can zoom and pan, updating the viewport. The concept of viewport is inscribed in the long-standing representational tradition of the *screen*. Manovich traces a compelling genealogy of the screen, seen as a flat, rectangular surface "acting as a window into another space" [16, p. 115].

While interacting with map viewports, users aim at fulfilling their spatial information need. This process is often focused on specific individual map features, decomposing the represented landscape analytically. However, the field of landscape ecology strongly argues that landscape is perceived holistically, as a complex whole. Antrop states that the holistic approach was stimulated by aerial photography, which represents the landscape in its holistic complexity [1]. Naveh, in his broad discussion on landscape ecology and system theory, identifies holism as “perceiving all parts in their full context”, and criticises analytical, reductionist approaches “focused on single, isolated parts of the system” [18, p .13]. Moreover, in cognitive science and psychology, holistic cognition is believed to play a major role in perception [29,20].

To be interpreted by humans, the geographic information represented in viewports has to convey some intelligible meaning. The semantics of geographic data has been discussed extensively by Kuhn, who points out the difficulties of grounding meaning in symbolic systems [11]. It is a tautology to state that meaning is crucial in geographic information retrieval (GIR), which aims at identifying relevant features in large datasets [22]. To date, most GIR systems focus on individual map features, with particular emphasis on text-based retrieval [4].

In order to compare, classify, index and cluster geographic objects by their semantics, several analytical approaches have been devised [23,8]. Schwering surveys and classifies the similarity measures for geographic data [26]. In these approaches, the similarity of individual semantic geographic concepts are compared based on their commonalities, differences, positions in taxonomies, and so on. While such models can compute plausible similarities between specific features or feature types, they do not consider the computation of holistic similarity of the map fragments that are, ultimately, displayed to and manipulated by the users in rectangular viewports.

When presented to users, viewports are bi-dimensional images. For this reason, our approach can be seen as analogous to techniques used in image processing systems [30] to compare raster images. However, while such techniques are based on the analysis of low-level image features, such as colour, to compute the similarity of raster viewports, our focus is on the semantics of specific objects. Therefore, our GIR system focuses exclusively on vector data, regardless of the particular visual display of the rendered viewport.

Moreover, viewports show geo-information at a specific map scale. In a typical web map, a viewport is associated with a scale and includes different types of features depending on specific visibility rules. None of the traditional semantic similarity measures discussed above take scale into account, as they focus on abstract psychological classes rather than on viewports. Our system, on the other hand, captures the scale of a viewport by building a semantic descriptor that includes only features that are present (i.e. represented) at the viewport scale – but independently of how they are represented.

To the best of our knowledge, no GIR system focuses on the holistic semantics of map viewports. To explore this concept, the next Section outlines a holistic information retrieval system, based on viewport semantic descriptors.

3 A Viewport-Based, Holistic GIR System

In this Section we detail our proposal of a viewport-based GIR system, by examining the structure of a typical web map, and constructing vector-based semantic descriptors for viewports. In a session in our GIR system, the user can retrieve viewports that are similar to a query viewport, indicated as fulfilling the users’

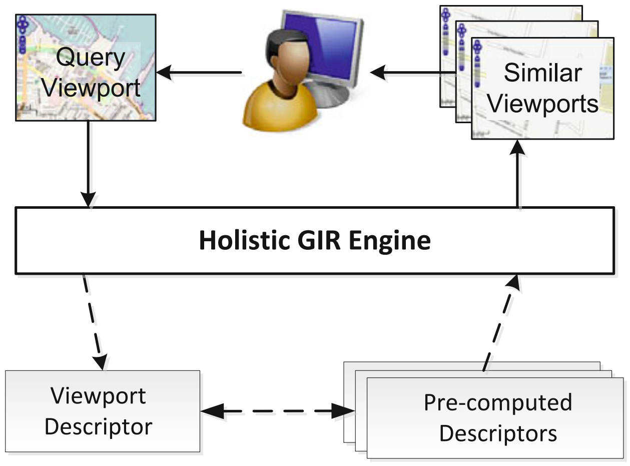


Fig. 1. The architecture of a viewport-based, holistic GIR system. The user submits a query viewport to the system, and the system retrieves the most similar viewports from a set of precomputed semantic descriptors.

spatial information need. The system compares the query viewport with pre-computed viewports, and returns to the user the most similar viewports it has found. This GIR architecture is schematised in Figure 1.

3.1 Viewports

When using GIR systems, users are presented with geographic data displayed in *viewports*, defined as a rectangular, bi-dimensional images rendered on a screen. To capture the semantic content of a viewport, it is useful to start from the visualisation structure of a typical web map. In order to test our approach, we utilised the OpenStreetMap vector dataset, released under a Creative Commons license [7]. As a representative of typical web mapping, we selected the Cloud-Made service, which renders OpenStreetMap data as interactive online maps.[2] As opposed to other commercial geo-services, this service enables exploration of the internal structure of a viewport, and its underlying geographic content.

In the CloudMade maps, the scale can be set to 19 discrete zoom levels, ranging from scale 1:446M (zoom level 0) to 1:1700 (zoom level 18). The map scale is controlled by Equation 1, where y is either the distance in meters or the map scale, and z is the zoom level. The constant C is $78,271$ in the case of meters per pixels, and $223 \cdot 10^6$ in the case of map scale. This equation allows the conversion between map scale, screen pixels, and zoom levels:

$$y = C\ 2^{1-z} \qquad z \in [0, 18] \tag{1}$$

[2] `http://maps.cloudmade.com` (accessed on 24/1/2012).

Table 1. Overview of the zoom levels of a CloudMade web map. Total number of feature types: 101.

Zoom Level	Meters per pixel	Scale	Visible Types	Description
1	78271	$1:223M$	2	Region
3	19568	$1:55M$	2	–
5	4892	$1:14M$	3	Country
7	1123	$1:3.5M$	5	–
9	306	$1:871K$	10	County
11	76	$1:217K$	23	–
13	19	$1:54K$	44	Neighbourhood
15	5	$1:13K$	64	–
17	1	$1:3400$	93	Building

At each zoom level, the map displays certain types of features, e.g. at the region level, only countries and seas are shown. For the purpose of our study, the geographic dataset has been subdivided into 101 feature types, closely modelled on the visualisation rules of the CloudMadeMap.[3] For example, types include *restaurant*, *stadium*, and *prison*. The visibility of each type per zoom level is defined as a range, e.g. restaurants are visible when $z \in [16, 18]$, while stadiums, being generally larger objects, in range $z \in [14, 18]$. For the sake of clarity, all the notations used in this paper are displayed in Table 2. Intuitively, the number of visible types increases as the scale decreases. The characteristics of each zoom level are summarised in Table 1.

In this context, a viewport $v_{bb,z}$ is defined by a bounding box bb, specified by the latitude/longitude coordinates of its bottom-left and top-right corners, and a zoom level $z \in [0, 18]$ as defined in Table 1. As stated in Section 1, a viewport can be seen as the visualisation unit of geographic data in a web map. A user session on a web map consists of a sequence of manipulative actions on the map, such as panning and zooming, resulting in the visualisation of a sequence of viewports $\{v_1 \dots v_n\}$. In our GIR system, the user can select a viewport by drawing a bounding box on the map, and the selected viewport v_s is used as a query, described in the next Section.

3.2 Holistic Viewport Descriptors

In order to retrieve semantically similar viewports, our GIR system constructs a holistic semantic descriptor for each viewport v in a vector space model. To extract the overall semantic content from a viewport, the system performs spatial queries on the OpenStreetMap dataset. Given the input viewport v, the system will perform spatial queries to retrieve F_v, all the visible features in that viewport (Equation 2):

$$\forall t \in S_z, \quad q(bb, z, t) \rightarrow F_t, \qquad F_v = \{F_{t_1} \dots F_{t_{|S_z|}}\} \tag{2}$$

[3] The visibility rules are defined at `http://maps.cloudmade.com/editor` (accessed on 24/1/2012).

Table 2. Notations

z	Zoom level $\in [0, 18]$ (see Table 1 for details).
bb	Bounding box, specified by the latitude/longitude coordinates of its bottom-left and top-right corners.
v	A viewport on bounding box bb at zoom level z. h_v and w_v is the viewport height and width in screen pixels.
t	A type of map feature (e.g. *restaurant*, *prison*, etc). In this work 101 types were defined.
$\sigma(z)$	Function mapping the visibility of feature types at zoom level z. $S_z \leftarrow \sigma(z)$.
S_z	Set of visible t at zoom level z. $S_z = \{t_0 \dots t_n\}$, where $\forall t$ is visible at zoom level z.
D_v	Semantic descriptor of viewport v.
$q(bb, z, t)$	Spatial query on bounding box bb, zoom level z, and feature type t. $q(bb, z, t) \rightarrow F_t$
F	Set of all existing map features.
F_t	Set of features of type t, $F_t = \{f_0 \dots f_n\}$
F_v	Set of features visible in viewport v. $F_v = \{F_{t_1} \dots F_{t_n}\}$
$I(t)$	Self-information of type t, assuming a random distribution of types in the map.
$a(f)$	Area of feature f.
V_g	Set of viewports extracted from a geographic area g.

The service can now compute a holistic descriptor D_v, combining all the visible types S_z in a multidimensional vector as in Equation 3, where n is the cardinality $|S_z|$, $t \in S_z$, and w are non-negative normalised weights.

$$D_v = w_1 t_1 + w_2 t_2 + \dots + w_{n-1} t_{n-1} + w_n t_n, \qquad w \in [0, 1], \ \sum_{i=1}^{n} w_i = 1 \quad (3)$$

In order to characterise D_v, we propose four ways to compute the weights w: linear, logarithmic, information-theoretic, and surface-based approaches.

(a) Linear weights. The simplest approach consists of assigning them proportionally to the cardinality of sets $F_t \in F_v$, using the normalised cardinality:

$$w_i = \frac{|F_{t_i}|}{\sum_{j=1}^{n} |F_{t_j}|} \quad (4)$$

The main limitation of this approach lies in the fact that the statistical distribution of types t is heavily skewed in favour of very frequent features, such as *road.* In a viewport on an urban area, the number of features *road* is often greater than other types by several orders of magnitude, such as *restaurants*, which matches our intuition on the fact that roads are very common map objects, while restaurants are less frequent. For example, it is uncommon to find 2 restaurants, and

200 roads in a viewport. In this case, the weighting function in Equation 4 would assign a very low weight to the type *restaurant*, and an extremely high weight to *secondary*.

(b) Logarithmic weights. A variant that focuses on magnitude rather than number of features is the following, where δ is a positive quantity that prevents the nullification of a term if $|F_{t_i}| = 1$:

$$w_i = \frac{log(|F_{t_i}| + \delta)}{\sum_{j=1}^{n} log(|F_{t_j}| + \delta)}, \qquad \delta = 1 \tag{5}$$

This logarithmic version is less sensitive to small changes in the statistical distribution of types t, and tends to preserve the importance of infrequent features.

(c) Information theoretical weights. A second variant to compute weights w_i taking into account the statistical occurrence of feature types, is based on the information theoretical approach [27]. Given a set of spatial features F, the probability p and the self-information I of feature type t randomly from the dataset are defined as in Equation 6. The self-information of type t can then be used to weight its importance in the vector, by combining it with the number of features:

$$p(t) = \frac{|F_t|}{|F|} \qquad I(t) = -log(p(t)) \qquad w_i = \frac{I(t_i)|F_{t_i}|}{\sum_{j=1}^{n} I(t_j)|F_{t_j}|} \tag{6}$$

In this case, the importance of a type t in the descriptor D_v is increased or reduced depending on its frequency in the dataset. Therefore features of type *secondary* carry low self-information, while features *restaurant* are emphasised. Although this weighting approach intuitively seems the most promising among the three we have presented (Equations 4, 5, and 6), it can assign very high weights to rare features. While in some cases this might be a desirable behaviour (for example to detect landmarks), in general it risks skewing the descriptor towards unusual features, regardless of their actual semantic weight in the viewport.

(d) Area weights. With features modelled as a polygon, it is possible to attribute a weight proportionally to the feature area, on the assumption that large features should have higher semantic importance in the descriptor. Defining the feature area as $a(f)$, and $a(F_t)$ as the sum of all the areas of the features in the set, the surface weights are computed as:

$$w_i = \frac{a(F_{t_i})}{\sum_{j=1}^{n} a(F_{t_j})} \qquad a(F_t) = \sum a(f), \quad \forall f \in F_t \tag{7}$$

In this approach, the feature area is weighted against the area of the other features, and not of the viewport. Thus, the weight can account for overlapping features.

Table 3. Weighting approaches in semantic viewport descriptor D_v

Approach	Key Parameter	Description
(a) Linear	Number of features	When types have different magnitude, large magnitudes take a large section of the descriptor.
(b) Logarithmic	Log of number of features	Represent the magnitude of types. Low sensitivity when types have the same magnitude.
(c) Self-Information	Self-Information of feature type	Common feature have low weight, while unusual feature types have very high weight.
(d) Area	Sum of feature areas	Large features have high weight. Not computable when feature is not a polygon (e.g. for points of interest)

The effectiveness of these four approaches to weight the semantic types in the viewport descriptor, summarised in Table 3, largely depends on the specific application context. As each approach captures different aspects of the holistic semantics of a viewport, and presents specific limitations, the weights can be computed by averaging different approaches. Without doubt, one of the main advantages of such vector-based viewport semantic descriptors is the wide range of techniques to compare, cluster, and classify them, discussed in the next Section.

3.3 Sampling the Viewport Space

In order to describe the semantics of a web map viewport, we have defined a vector-based descriptor D_v, in four variants (linear, algorithmic, self-information, and surface). Given a web map covering a certain geographic area g, e.g. Ireland, we aim at extracting a number of descriptors that represent its semantics. The viewport space is the set of all possible viewports in g at zoom level z. The area g can be sampled in a number of viewports $V_g = \{v_1 \dots v_n\}$, where n is the desired number of viewports. The theoretical number of viewports that can be extracted in a geographic area delimited by a bounding box bb_g at zoom level z, where h_g, w_g are the height and width of the geographic area converted into pixels with Equation 1. h_v and w_v are the height and width of the viewport in pixels:

$$|V_g| = (h_g - h_v)(w_g - w_v) \tag{8}$$

For example, a geographic area g delimited by a bounding box of size $\approx 300 \times 230\ km^2$ corresponds at $z = 9$ (county level) to a screen of 4096×3072 pixels. Sampling g with 1024×768 pixel viewports, a common resolution for web maps, the possible viewports amount to ≈ 7.6 million. With higher zoom levels, the number of possible viewports increase rapidly following the power law in Equation 1. It is therefore evident that a sampling technique has to be utilised to extract a computable number of viewports, in particular for high zoom levels.

The most intuitive way of choosing viewports is based on user interests. Viewports in which user activity is performed are automatically included in the sample. However, to overcome the cold start problem that arises in this case, a general sampling mechanism is necessary to index g. A possible technique is that of random sampling, based on the law of large numbers. Even though it is difficult to compute all the possible viewports, it is possible to extract a sufficient number of random viewports to represent accurately the whole set of viewports, setting the sample size to a limit β, as shown in Equation 9. In order to determine β, a confidence level and a confidence interval have to be chosen.

$$|V_g|_\beta = \frac{\beta}{|V_g|}(h_g - h_v)(w_g - w_v) \qquad 0 < \beta < |V_g| \tag{9}$$

Similarly, it is possible to sample g by defining an arbitrary grid γ of pixels h_γ and w_γ, which reduces the number of viewports as follows:

$$|V_g|_\gamma = \frac{(h_g - h_v)(w_g - w_v)}{h_\gamma w_\gamma} \qquad 0 < h_\gamma < h_g, \quad 0 < w_\gamma < w_g \tag{10}$$

A third possibility is a combination of grid and random sampling. The geographic area g is divided into an arbitrary number of grid cells, and each cell is sampled randomly. The choice of the sampling technique (random, grid-based or both) has to be done on an empirical basis, depending on the specific application context. Once a sample V_g has been selected, the corresponding descriptors D_v can be computed. Subsequently, the system can compare, cluster, and retrieve viewports (see Figure 1). The next Section describes comparison techniques for the semantic descriptors.

3.4 Comparing Viewport Descriptors

A geographic area g can sampled as a set of viewports V_g, with the techniques described in the previous Section. The corresponding descriptors D_v can then be pre-computed through one of the approaches presented in Section 3.2. The similarity of two viewports is therefore the similarity of their descriptors (Equation 11).

$$s(v_a, v_b) = s(D_{v_a}, D_{v_b}) \qquad s(v_a, v_b) = s(v_b, v_a) \qquad 0 \leq s(v_a, v_b) \leq 1 \tag{11}$$

Given that these descriptors are multidimensional vectors, encoding semantic aspects of the viewports, it is possible to compare them with well-known techniques in a vector space [24,9,32]. In the viewport semantic vector space, every viewport can be modelled as a row in a multidimensional matrix $|V_g| \times |t|$, having a column for each feature type t. Vector similarity is traditionally computed using linear algebra techniques, such as the Euclidean, cosine, Chebyshev, and Manhattan distances [3,13].

In a semantic information retrieval system in which the user submits a viewport v_q as a query, the most similar k viewports must be retrieved and displayed to them. To do so, these distance measures can be efficiently computed between the descriptor of the query viewport D_{v_q} and all the pre-computed descriptors D_v, where $v \in V_g$. Additionally, the similarity computation can be constrained in several ways to match specific user information needs. Among others, common constraints are the maximum or minimum distance from the query viewport v_q, a zoom level range ($z \in [z_{min}, z_{max}]$), and a weight constraint on a feature type ($w_{min} < w_t < w_{max}$).

The next Section presents a case study, in which the descriptors are used to capture the holistic semantics of viewports taken from the Dublin area in Ireland, and are used to retrieve semantically similar viewports.

4 A Case Study Walkthrough

To capture a holistic impression of semantics in a web map, we have defined semantic descriptors as vectors D_v that represent the overall semantic content of the viewport v in which the map is displayed. This Section illustrates an interaction with our viewport information retrieval system (see Figure 1), suggesting possible applications, strengths and weaknesses of the approach.

We noted certain tasks that users perform on web maps are not only analytical, i.e. focused on the decomposition of large objects into simpler parts, but are also holistic, treating a geographic area as a unified entity. Analytical tasks involve the examination of specific target objects, and so on. On the other hands, examples of spatial holistic tasks are the classification of an urban area versus a rural area, in which the user needs not to focus on individual objects, but classify the area as a whole. These holistic tasks should not be considered in opposition to analytical tasks, but they are intertwined in the complex cognitive interplay that occur in the interaction with information retrieval systems.

In a holistic geographic information retrieval, the user can retrieve, instead of specific geographic features, viewports that represent visually geographic areas rendered at a given zoom level. In this case, the user's information need is not a specific spatial information, e.g. where is the target object, or what is the area of the target object, etc, but is an implicit semantic judgement on viewports displayed on the screen. A viewport representing a geographic area that fulfills the user's information need, e.g. a seaside town or a commercial port, is used as a query viewport to retrieve viewports conveying similar semantic content. To achieve this, the system has to be able to model this *implicit judgement* on geographic content displayed in a viewport v.

Sample Viewports. In this case study we consider a small set of viewports selected from a bounding box bb, corresponding to the surroundings of Dublin. This geographic area g contains a total of $\approx 20,000$ features. Five sample viewports were extracted at zoom level $z = 15$, including a Dublin suburb, a park, a port, and two seaside towns. Five semantic descriptors D_v were then computed for each of the six viewports v, linear, logarithmic, information-theoretic, surface, and a mean of the first four, limiting for the sake of illustration the number of feature type to 10 out of the 64 visible types. The self-information of each

Table 4. Viewport semantic descriptors D_v for 5 sample viewports, with weights computed using four approaches, and their mean. Symbol '–' corresponds to 0.

Viewport	Feat Type	Linear	Log	Area	Self-Info	Mean
v_1	building	.495	.321	.162	.388	.341
	coastline	–	–	–	–	–
Milltown: suburb of Dublin, with hospital, college, and residential estates.	commercial	.029	.113	.034	.058	.058
	hospital	.01	.056	.111	.021	.05
	industrial	–	–	–	–	–
	park	.01	.056	.043	.025	.033
	port	–	–	–	–	–
	road	.427	.309	–	.462	.3
	town	.01	.056	.59	.021	.169
	wood	.019	.089	.06	.025	.048
v_2	building	.229	.272	.035	.156	.173
	coastline	–	–	–	–	–
Phoenix Park: Large urban park with zoo, polo grounds, and American embassy.	commercial	–	–	–	–	–
	hospital	–	–	–	–	–
	industrial	–	–	–	–	–
	park	.029	.086	.789	.064	.242
	port	–	–	–	–	–
	road	.257	.285	–	.242	.196
	town	–	–	–	–	–
	wood	.486	.358	.175	.539	.389
v_3	building	.087	.16	.089	.047	.096
	coastline	.442	.268	–	.542	.313
	commercial	–	–	–	–	–
Howth: seaside town with tourist attractions, cliffs, and trekking trails.	hospital	–	–	–	–	–
	industrial	–	–	–	–	–
	park	.029	.096	.276	.052	.113
	port	–	–	–	–	–
	road	.308	.243	–	.233	.196
	town	.01	.048	.309	.015	.095
	wood	.125	.184	.325	.111	.186
v_4	building	.296	.207	.158	.175	.209
	coastline	.194	.183	–	.257	.158
Dun Laoghaire: seaside town with a small port, a private school, and a hospital.	commercial	.143	.165	.175	.214	.174
	hospital	.01	.042	.088	.017	.039
	industrial	.02	.067	.026	.028	.035
	park	.01	.042	.018	.02	.022
	port	.01	.042	.184	.021	.064
	road	.306	.209	–	.251	.191
	town	.01	.042	.351	.017	.105
	wood	–	–	–	–	–
v_5	building	.17	.19	.12	.08	.14
	coastline	.136	.176	–	.144	.114
Dublin Port: docks of the Dublin port, where large ships load and unload containers.	commercial	.386	.244	.326	.461	.354
	hospital	–	–	–	–	–
	industrial	.227	.209	.174	.251	.215
	park	–	–	–	–	–
	port	.011	.048	.38	.019	.115
	road	.068	.133	–	.044	.062
	town	–	–	–	–	–
	wood	–	–	–	–	–

Table 5. Similarity of sample viewports. The matrices are symmetrical, and their diagonal values are equal to 1. The euclidean similarity has been computed as $1 - d$, where d is the euclidean distance.

Viewport Name	$\star$	Cosine v_1	v_2	v_3	v_4	v_5	Euclidean v_1	v_2	v_3	v_4	v_5
Milltown	v_1	–	–	–	–	–	–	–	–	–	–
Phoenix Park	v_2	.55	–	–	–	–	.52	–	–	–	–
Howth	v_3	.54	.65	–	–	–	.55	.59	–	–	–
Dun Laoghaire	v_4	.82	.38	.68	–	–	.72	.48	.66	–	–
Dublin Port	v_5	.37	.15	.29	.73	–	.46	.35	.45	.68	–

feature type was computed on g, and not on the entire OpenStreetMap dataset. The viewports and corresponding descriptors are reported in Table 4.

Looking at the weights of the descriptors, it is possible to trace behaviour, pros and cons of each of the four weighting mechanism proposed in Section 3.2. The linear weights reflect the number of features visible in the viewport. For this reason, important feature such as the Phoenix Park in v_2 rank very low, and roads rank very high in most viewports. This is because roads are represented in numerous small chunks, while large objects such as a park consist of one large polygon, and the linear weights do not take this aspect into account.

This problem is partly addressed by the logarithmic weights, which smooth the results by increasing the importance of types with few features and by decreasing that of types with many occurrences in the viewport. In v_3, the logarithmic weight of *building* has been doubled, while that of *coastline*, another type of feature modelled in small chunks, has been reduced. However, despite the smoothing, the resulting weights are still strongly biased towards types such as *road*, and *building*, while large and infrequent features are squeezed into small weights.

The area-based technique tends to correct this bias. In v_2, the type *park*, which is almost ignored by the previous weighting mechanisms, is the most important in the viewport. Thanks to its large area, this type gains a lot of influence in the descriptor. As it is possible to notice by the frequent 0 values in the area column, only a subset of features are polygons and can be included in this descriptor, resulting in a limited information problem.

The behaviour of the self-information weights is more difficult to interpret. For types that occur very frequently in g, such as *road* and *building*, $I(t)$ is low (3.51 and 4.85), while less frequent types have higher $I(t)$ (12.21 for *port*, and 11.56 for *park*). This is correct, but when the self-information values are multiplied by the number of features, they smooth the results to a limited extent. In the case of viewport v_2, according to the self information weights, buildings are more important than parks, maintaining the bias of the linear and logarithmic weights. As it is expected, the mean weights are heavily smoothed, but maintain some of the bias of the linear, logarithmic, and self-information weights.

Viewport Similarity. The semantic similarity of the viewports can be computed via the cosine distance between their descriptors, as shown in Table 5.

In this case, the two vector similarity measures rank the pairs in the same way. The pairs with highest similarities are $\langle v_1, v_4\rangle$, and $\langle v_4, v_5\rangle$. Viewports v_1 and v_4 are semantically very similar, and this result is satisfactory. The pair $\langle v_4, v_5\rangle$ is surprising, because a tourist seaside town appears very different to a commercial port. This result is easily explained with the omission of feature types that would increase the distance between the two viewports, such as restaurants, tourist attractions, amenities, which are strongly present in v_4 but not in v_5. The two viewports share the fact of including the seaside, the presence of a port, many buildings and commercial activities, all aspects that are captured by the 10 feature types considered in this case study.

On the other hand, the least similar pairs are $\langle v_2, v_5\rangle$, and $\langle v_3, v_5\rangle$. This seems to be a valid result, as these pairs represent very different areas, sharing very few feature types. Based on this case study, it can be concluded that the holistic semantic descriptors proposed in this paper are a promising approach to compute semantic similarity of viewports.

5 Conclusions and Future Work

In this paper, we have proposed a technique to extract semantic descriptors for viewports, which can be used in a viewport-based, holistic GIR system (see Section 3). Instead of focusing on specific geographic features as in traditional GIR systems, our system aims at capturing the overall semantic content in a viewport. Thus, viewports are treated in a manner similar to documents in text-based IR. Based on the work presented in this paper, the following conclusions can be drawn:

- The vector-based semantic descriptors capture the overall semantic content of a viewport in a holistic mode, without focusing on specific individual features. The user retrieves viewports that present similar characteristics to the query viewport that is submitted to the system.
- The viewports display a map at a given zoom level. The corresponding descriptor captures the semantic content displayed at the specific zoom level. This enables the semantic analysis of the viewports, taking the map scale into account.
- Four weighting mechanisms are proposed to extract semantic content from a viewport. Such weights can be combined to achieve different representation of the same viewport. Being based on the well-known vector space model, our approach can benefit from a wide range of techniques to index, compare, classify, and cluster large sets of vectors. Descriptors can be used to perform collaborative filtering, and user profiling.
- Our holistic GIR system offers an additional technique to retrieve relevant geographic information from a large dataset. It is not conceived as antagonistic to traditional GIR, but rather as a complementary approach that can be combined with existing analytical techniques to enable retrieval through holistic semantics.

The case study outlined in Section 4 indicates that the system is able to capture the holistic semantic of a viewport. However, further work is necessary to assess its accuracy and recall on a big spatial dataset, in the context of realistic information needs. The limitations of the presented approach can be overcome by incorporating more sophisticated semantic techniques for vector space models, such as latent semantic analysis [32,28]. Besides, other holistic metrics be added to the descriptors, such as heterogeneity, entropy, and fractal dimension [1].

The holistic GIR system presented in this paper provides a different approach to traditional geographic information retrieval, modelling the overall semantic content of a viewport in a vector space model. Treating viewports as documents enables the exploration of digital maps from a holistic perspective, stressing the need for reconsidering the undisputed centrality of analytic approaches.

References

1. Antrop, M., Van Eetvelde, V.: Holistic aspects of suburban landscapes: visual image interpretation and landscape metrics. Landscape and Urban Planning 50(1-3), 43–58 (2000)
2. Ballatore, A., Bertolotto, M.: Semantically Enriching VGI in Support of Implicit Feedback Analysis. In: Kim, K.-S. (ed.) W2GIS 2011. LNCS, vol. 6574, pp. 78–93. Springer, Heidelberg (2010)
3. Berry, M.W., Drmac, Z., Jessup, E.R.: Matrices, vector spaces, and information retrieval. SIAM Review, 335–362 (1999)
4. Gey, F., Larson, R., Sanderson, M., Joho, H., Clough, P., Petras, V.: GeoCLEF: The CLEF 2005 Cross-Language Geographic Information Retrieval Track Overview. In: Peters, C., Gey, F.C., Gonzalo, J., Müller, H., Jones, G.J.F., Kluck, M., Magnini, B., de Rijke, M., Giampiccolo, D. (eds.) CLEF 2005. LNCS, vol. 4022, pp. 908–919. Springer, Heidelberg (2006)
5. Goodchild, M.F.: Citizens as Sensors: the world of volunteered geography. GeoJournal 69(4), 211–221 (2007)
6. Haklay, M., Singleton, A., Parker, C.: Web Mapping 2.0: The Neogeography of the GeoWeb. Geography Compass 2(6), 2011–2039 (2008)
7. Haklay, M., Weber, P.: OpenStreetMap: User-Generated Street Maps. IEEE Pervasive Computing 7(4), 12–18 (2008)
8. Janowicz, K., Keßler, C., Panov, I., Wilkes, M., Espeter, M., Schwarz, M.: A study on the cognitive plausibility of SIM-DL similarity rankings for geographic feature types. In: Fabrikant, S., Wachowicz, M. (eds.) The European Information Society, pp. 115–134. Springer, Heidelberg (2008)
9. Jing, L., Ng, M.K., Huang, J.Z.: Knowledge-based vector space model for text clustering. Knowledge and Information Systems, 1–21 (2010)
10. Jones, C., Alani, H., Tudhope, D.: Geographical Information Retrieval with Ontologies of Place. In: Montello, D.R. (ed.) COSIT 2001. LNCS, vol. 2205, pp. 322–335. Springer, Heidelberg (2001)
11. Kuhn, W.: Geospatial Semantics: Why, of What, and How? In: Spaccapietra, S., Zimányi, E. (eds.) Journal on Data Semantics III. LNCS, vol. 3534, pp. 1–24. Springer, Heidelberg (2005)
12. Leveling, J.: Challenges for Indexing in GIR. SIGSPATIAL Special 3(2), 29–32 (2011)

13. Li, H., Shi, R., Chen, W., Shen, I.F.: Image tangent space for image retrieval. Pattern Recognition 2, 1126–1130 (2006)
14. Liu, W., Gu, H., Peng, C., Cheng, D.: Ontology-based retrieval of geographic information. In: 2010 18th International Conference on Geoinformatics, pp. 1–6 (June 2010)
15. Lutz, M., Klien, E.: Ontology-based retrieval of geographic information. International Journal of Geographical Information Science 20(3), 233–260 (2006)
16. Manovich, L.: The Language of New Media. MIT Press, Cambridge (2001)
17. Meadow, C.T., Boyce, B.R., Kraft, D.H.: Text information retrieval systems. Academic Press (2007)
18. Naveh, Z.: What is holistic landscape ecology? A conceptual introduction. Landscape and Urban Planning 50(1-3), 7–26 (2000)
19. Nisbett, R.E., Miyamoto, Y.: The influence of culture: holistic versus analytic perception. Trends in Cognitive Sciences 9(10), 467–473 (2005)
20. Nisbett, R.E., Peng, K., Choi, I., Norenzayan, A.: Culture and systems of thought: Holistic versus analytic cognition. Psychological Review 108(2), 291 (2001)
21. Nivala, A.M., Brewster, S., Sarjakoski, L.T.: Usability Evaluation of Web Mapping Sites. The Cartographic Journal 45(2), 129–138 (2008)
22. Purves, R., Jones, C.: Geographic information retrieval. SIGSPATIAL Special 3(2), 2–4 (2011)
23. Rodríguez, M.A., Egenhofer, M.: Comparing Geospatial Entity Classes: An Asymmetric and Context-Dependent Similarity Measure. International Journal of Geographical Information Science 18(3), 229–256 (2004)
24. Salton, G., Wong, A., Yang, C.S.: A vector space model for automatic indexing. Communications of the ACM 18(11), 613–620 (1975)
25. Schwarzer, G., Huber, S., Dümmler, T.: Gaze behavior in analytical and holistic face processing. Memory & Cognition 33(2), 344–354 (2005)
26. Schwering, A.: Approaches to Semantic Similarity Measurement for Geo-Spatial Data: A Survey. Transactions in GIS 12(1), 5–29 (2008)
27. Shannon, C.E.: A mathematical theory of communication. Bell System Technical Journal 27, 379–423, 623–656 (1948)
28. Sizov, S.: GeoFolk: latent spatial semantics in web 2.0 social media. In: Proceedings of the Third ACM International Conference on Web Search and Data Mining, pp. 281–290. ACM (2010)
29. Smith, J.D., Shapiro, J.H.: The occurrence of holistic categorization. Journal of Memory and Language 28(4), 386–399 (1989)
30. Tan, H., Yu, P., Li, X., Yang, Y.: Digital image similarity metrics and their performances. In: Artificial Intelligence, Management Science and Electronic Commerce (AIMSEC), pp. 3922–3925. IEEE (2011)
31. Turner, A.: Introduction to Neogeography. O'Reilly Media, Inc., Sebastopol (2006)
32. Turney, P.D., Pantel, P., et al.: From frequency to meaning: Vector space models of semantics. Journal of Artificial Intelligence Research 37(1), 141–188 (2010)
33. Ward, T.B., Scott, J.: Analytic and holistic modes of learning family-resemblance concepts. Memory & Cognition 15(1), 42–54 (1987)

Recommendations Based on Region and Spatial Profiles

Gavin McArdle[1], Mathieu Petit[2], Cyril Ray[3], and Christophe Claramunt[3]

[1] National Centre for Geocomputation, National University of Ireland Maynooth, Maynooth, Co. kildare, Ireland
gavin.mcardle@ucd.ie
[2] Matiasat System R&D, Levallois-Perret, France
mpetit@matiasat.com
[3] Naval Academy Research Institute, Brest, France
{cyril.ray,christophe.claramunt}@ecole-navale.fr

Abstract. Fuelled by the quantity of available online spatial data that continues to grow, the requirement for filtering spatial content to match mobile users' context becomes increasingly important. This paper introduces a flexible algorithm to derive users' preferences in a mobile and distributed system. Such preferences are implicitly computed from users' virtual and physical interactions with spatial features. Using this concept, region profiles for specific spatial contexts can be generated and used to recommend content to those visiting that region. Our approach provides a set of profiles (personal and region-based) which are combined to adapt the presentation of a given service to suit users' immediate needs and interests. A proposed college campus navigation assistant illustrates the benefits of such an unobtrusive recommender system.

Keywords: Location-based services, Contextual adaptation, Implicit profiling, Multi-user recommendations.

1 Introduction

Due to the increase in the availability of online services which permit users to tag and edit spatial data as well as share location information, the quantity of available geo-information continues to grow. While there are many positive aspects to the availability of this information, including greater access to free spatial data and up-to-date information, there are also an increasing number of opportunities emerging in this domain. For example, information overload which is a well-known issue in the Web domain is now becoming prevalent among spatial data as the two are merged through Location-Based Services (LBS). In the Web domain a single search query can return millions of matching Web pages. Although most search engines order the returned results, it is still an ominous task for the user to search through these results as the semantics that emerge are not always those which are of interest. Similarly, the amount of information available via a LBS can be so voluminous that it makes finding relevant information difficult. This effect is particularly serious for LBS in which client tiers operate on mobile devices and therefore have reduced processing capabilities coupled with a smaller screen size on which to display information [4,5]. Therefore, it is advantageous to only recommend a subset of data in this case. Of course this recommendation needs to be configured, so that the subset of data match the user preferences and interests while taking locational context into account.

This paper introduces a novel technique for generating user profiles within a LBS. By segmenting an environment into physical regions based on the underlying infrastructure topology, profiles for each region can be generated. This is achieved by amalgamating the individual profiles of those visiting such regions into a common region profile. LBS users are given the opportunity to determine which form of personalisation and recommendation (personal, collaborative or regional) suits their current needs. This technique is described by applying it to a case study of a college campus LBS assistant and highlights how this hybrid approach to profile generation can effectively resolve issues with comparable approaches.

The next section describes related work in the area of user profiling and shows how our work builds on this through the development of region profiles which resolve common problems with existing recommender systems. Section 3 presents the campus assistant LBS and highlights how profiling permits client-side adaptations and recommendations. Section 4 details clusters and group dynamic derivation, while section 5 describes the proposed profiling algorithm. A discussion of the limitations and on-going development of the approach are presented in section 6, while a summary of the work is provided in section 7.

2 User Profiling Methodologies

This section introduces user profiling techniques and describes their strengths and weaknesses as well as contexts where they have been applied. Our proposed technique and details of how it resolves inherent issues with existing approaches is also presented.

2.1 Profiling Techniques

Determining user preferences and defining profiles can be achieved using explicit or implicit techniques. The former involves directly querying the user for input regarding their interests. While this approach has the merit of an immediate set of interests, it can be time consuming, suffer from distortion and subjectivity while also distracting the user from the task at hand [18,19]. Alternatively, implicit profiling monitors user behaviour and infers user interests. There are two main approaches used to implicitly derive a user profile and produce recommendations: content-based and collaborative.

The content-based approach for recommendation considers past actions of individuals as indicators of future behaviour. Individual analytical user profiling in the spatial domain has received attention lately; in particular such implicit profiling techniques have been employed to determine user interests when interacting with spatial content [11,2]. In all cases, the information gleaned through such profiling is used to recommend a spatial content to the user. Amongst its advantages, this process has no additional overheads for the user. This method is favoured in many Web-based situations where interactions such as link clicking, bookmarking and printing are seen as an indication that the user is interested in the associated content [7]. Recently this methodology has been extended to the spatial context where interactions with map data act as interest indicators, permitting map personalisation [11]. While content-based recommendation is effective, there are several issues with this approach. In particular, the well-known *cold start period problem* is prevalent among such recommender systems. This period occurs when a new user profile is not fully defined and does not contain enough information to reliably infer user interests [15,1]. Similarly, profiles generated by the content-based approach can also suffer from an *inertia of the content* related to the difficulty of measuring changes in rapidly changing behaviour [8].

Collaborative profiling approaches consider current actions and preferences of similar users as an indicator of one's own interests. Nowadays, this technique is a topic of interest in the spatial domain. As the use of LBS and the number of mobile users equipped with smart phones continues to grow, the ability to provide relevant group preferences to individuals is appealing. For example, [16] builds group profiles based on user interaction with map objects as well as geographic proximity to objects. Similarly, [6] use a location bias as the first step in performing collaborative recommendation. Such approaches are far less sensitive to the *inertia* of profile generation while also eliminating the *cold start* problem by associating a specific group profile to new users. However, using group profiles to define personal profiles can introduce *stereotypes of users* that satisfy users in general, but none of them in particular [13]. Furthermore it is difficult to formalise the notion of contextual proximity so that associations of users are neither too loose nor too restrictive, referred to as *grouping criterion.*

Both content-based and collaborative approaches have their merits and determining which one to use is not always clear. Hybrid methods can take the respective advantages of both techniques [1,3]. The approach proposed in this paper extends a user profiling technique, which combines user mobility and interface interaction to infer interests [11]. Especially, these profiles are used to personalise services on a collaborative basis. This is achieved using implicit interest indicators, described in [10], combined with collaborative filtering and case-based reasoning [17], along with user location and context [12].

2.2 Proposed Approach

By introducing a region profile whereby all users contribute to, and take part of, a common and shared profile for specific geographical regions in which they interact, implicit profiling is improved. Unlike current group profiling techniques, different geographical regions assume a profile which is derived from the commonalities in the profiles of people visiting a specific spatial region. Accordingly, region profiles can be considered as a type of a group profile with a spatial context. It is however necessary to firstly ascertain individual user interests by monitoring their interactions with the device, information systems and physical locations. When visiting an area or spatial region, the contents of this profile, such as preferences, contribute to the profile for that spatial area. Simultaneously, recommendations are made to individual users considering their own profile and that of their current spatial context. Newcomers to the system, who have no profile can then be assigned the profile of the current spatial region that they are in before their own profile matures. Our algorithm envisions a multi-tiered LBS and provides different types of profile which resolve the problems of traditional implicit profiling as highlighted below:

- *Grouping criterion* : LBS is about getting the right information at the right place and time. That is how the execution space can be clustered along with the platform components giving rise to an almost natural "functionality-guided" grouping criterion.
- *Inertia of content* : as space is clustered, profiles can be broken apart and regularly updated whilst users move from one region to another.
- *Cold start* : group profiles can be derived locally for each LBS cluster, as long as different users join during the execution (but not necessarily at the same time). These profiles can be suggested to newcomers.
- *Stereotyped users* : as a user's experience in a cluster matures, their personal profile enriches and the initial suggested group profile has less and less impact.

3 Guiding Scenario: The Campus Assistant

The proposed profiling methodology is exemplified by an illustrative case study involving students as they interact with facilities located on a college campus. This scenario involves students with wide-ranging interests. For example *Jim* is a new student discovering the campus, *John* is already registered in Computer Science, and is interested in mathematics and engineering. *Jane* is more interested in geography or history. Each would like to receive information that matches with their interests without demanding too much of their attention. These students' diverse interests are also reflected at the geographic level. The campus is divided into groups of buildings, each of them hosting one or several departments. For example, the humanities departments are co-located in the same building complex.

Our profiling methodology takes advantage of both user and geographic specialisations. First, the students are compared according to their location and nearby resources. Then their experience of the system is personalised according to their own preferences and to the preferences inferred by their current location in the campus. Profiles are therefore likely to differ from one place to another, and from one user to another.

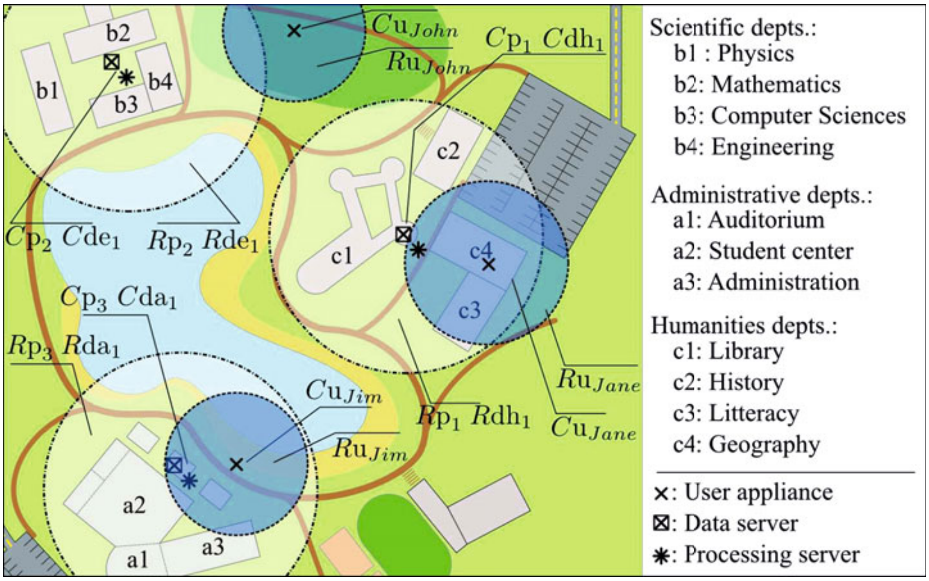
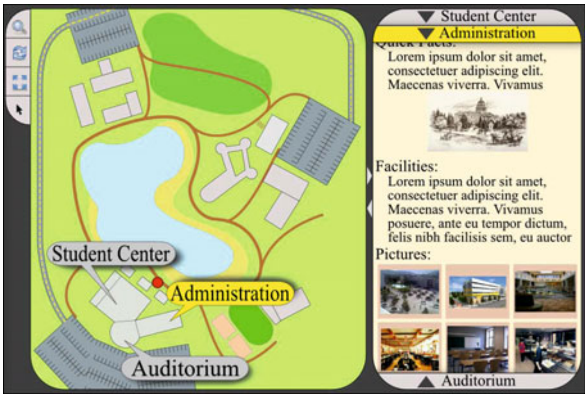


Fig. 1. Envisioned campus and service areas of the assistant information system at a given time instant t_1 of the execution

The campus assistant is designed so that several buildings and faculties provide service areas through WiFi hotspots (Fig. 1). These hotspots reflect the geographic clustering of the campus departments. Humanities, administrative and science areas have been defined. All service areas allow users to receive information regarding buildings in the surrounding area so that when a user is close to a particular building, they can obtain information about the departments located there. This approach, using geographical proximity to objects, is an initial step in a personalisation technique which contributes to the overall goal of reducing information overload.

Figure 2 displays examples of the proposed campus assistant on a mobile phone. On the left, the map panel highlights a user's current location and provides GIS functionality. The panel on the right details the currently selected element. Users can click on specific buildings and objects on the map to obtain additional information. Similarly, a user can click on the tabs within the panel to obtain information about other elements of the map. For example, as the user *Jim* (referred as Cu_{Jim} in figure 1) walks through the "Administrative" area of service, he has access to information about the department and consults the central administration panel.

In the campus assistant, preference profiles are generated through interactions with content displayed and places visited. They are used to recommend spatial and non-spatial content to users [11]. As *Jim* is a new student he has

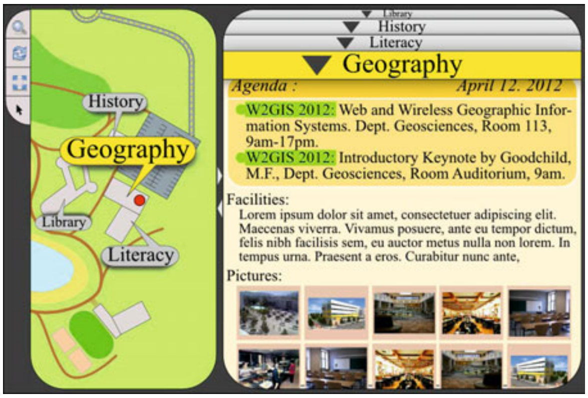


Quick Facts:
Lorem ipsum dolor sit amet, consectetuer adipiscing elit. Maecenas viverra. Vivamus

Facilities:
Lorem ipsum dolor sit amet, consectetuer adipiscing elit. Maecenas viverra. Vivamus posuere, ante eu tempor dictum, felis nibh facilisis sem, eu auctor

Pictures:

(a) *Jim*'s: Administrative Depts. interface, with no profile adaptation



Agenda : April 12. 2012
W2GIS 2012: Web and Wireless Geographic Information Systems. Dept. Geosciences, Room 113, 9am-17pm.
W2GIS 2012: Introductory Keynote by Goodchild, M.F., Dept. Geosciences, Room Auditorium, 9am.

Facilities:
Lorem ipsum dolor sit amet, consectetuer adipiscing elit. Maecenas viverra. Vivamus posuere, ante eu tempor dictum, felis nibh facilisis sem, eu auctor metus nulla non lorem. In tempus urna. Praesent a eros. Curabitur nunc ante,

Pictures:

(b) *Jane*'s: Humanities Depts. interface, with slight profile adaptations

Fig. 2. Sketches of the proposed campus assistant users interfaces at t_1

no profile and no preferences for content yet. While he waits at the university registration desk, he accesses the administrative service area, and all information he receives is displayed on an equal basis (Fig.2a). From previous use of the system, *Jane* already derived preferences regarding humanities related content. Her profile, among other things, indicates that she likes geography and that she usually displays information in a large frame. When *Jane* moves into the geography building based in the Humanities area, her display is adapted to emphasise geography-related information, and she receives notification of an upcoming conference (Fig. 2b). As she walks around the department, her profile is incorporated with that which exists for the humanities region.

At t_2, *Jim* also enters this spatial region, and his display adapts to the spatial profile for that region. In this case, it is derived from *Jane*'s profile as she is the only other individual that visited the humanities region. As $Jane's$ profile heavily recommends geography, this department appears high in the profile of the region and so Jim receives the same suggestion as $Jane$ regarding the conference. From now on, both $Jane$ and Jim contribute to the region profile. As $Jim's$ own interest for literacy begins to grow, their shared profile at the region level balances between literacy and geography.

At t_3, on her way to the science departments, $Jane$ leaves Jim to meet $John$, whose profile favours computer sciences and literacy. As they chat, their appliances share profiles. Specifically, $John's$ smart phone embeds $Jane's$ and $Jim's$ commonly derived humanities preferences, and $Jane's$ client downloads $John's$ science area profile.

Finally, at t_4, $John$ meets Jim in the humanities area. Although the profile he received from $Jane$ might have provided him with adapted content, $John$ prefers literacy. Therefore he rejects the profile he receives by cancelling the modification on his client and favours his own. By doing so, the conference event is not recommended to him. $John$ also strengthens the significance of literacy at the region profile level.

The above scenario effectively highlights the principles of how individual and region profiles operate in conjunction with each other to provide personalised recommendations for users of the system. The next sections formalise regions and profiles definitions in the context of the campus system. Details of how they are combined to provide different levels of personalisation are also outlined.

4 Clustering Components of a Location Based System

This section describes a LBS as a set of hardware components. At any time of execution, the spatial union of these components (communication range) gives rise to region-based clusters defined by spatial, functional and related context.

4.1 Dynamic Component and Regions Distribution

Multi-tiered systems such as the college campus assistant can be built upon a fluctuating set of active pieces of hardware. These components define the physical platform and assume several *functional roles* in the system. For example, multi-tier systems usually contain a user-interaction provider role, a data manager role, and a processing server role [9,14]. These roles are implemented in one or several supporting components. For example, the user interaction components provide user-oriented views and interaction facilities; the data components import and export information subsets; and the processing components host data analysis and transformation functionality. The nature and number of roles are not limited but rather depend on how the system is modelled, and on the designers own choices. As a general rule, any components implementing identical tasks and/or hosting the same information belong to a same role.

Notation 1. Role and ID of a component: In the following, let $C\text{<role>}_{id}$ denote a hardware component identified by "*id*" with respect to this component role, identified by "<role>". When, at a given time instant t_i, this component is active within the system execution space, it belongs to the set $Platform(t_i) = \{C\text{rla}_{id1}, C\text{rla}_{id2}, \ldots, C\text{rlx}_{idy}, \ldots, C\text{rln}_{idm}\}$ of active components. Components belonging to this platform at t_i support distinct functional roles labelled as"rla", "rlx", "rln", and at least two components, $C\text{rl1}_{id1}$ and $C\text{rl1}_{id2}$, implement the role rla.

In the campus assistant, raw information about groups of buildings are managed by dedicated data servers. Their roles "de", "da" and "dh" have been chosen according to the content of the data they are hosting. For example, humanities information is broadcast by components associated to the role "dh" (Fig. 1($C\text{dh}_1$)). Processing components are provided with raw data to generate tiled map views and layout the department information panels. As the processing and functionalities offered at their levels are identical, a unique role "p" has been defined and encompasses all three processing components (Fig. 1($C\text{p}_{1\to 3}$)). At the infrastructure level, hardware components combine data management and processing facilities (Fig. 1(⊠✱). Users have been assigned role "u". Their clients, like $C\text{u}_{Jim}$, constitute the uncertain piece of the platform, as a user's walk through the campus might make them available (or not) to the other components of the system.

Each component of $Platform(t_i)$ corresponds to a region that represents their accessibility range. Depending on the roles of their supporting components, several types of regions are distinguished. For example, 3-tier systems usually host: user-region(s) $R\text{u}_i$, where the user(s) is/are located and interacts with the system; broadcasting region(s) $R\text{d}_j$, where the information and data are available to the system; and processing region(s) $R\text{p}_k$, where the tools and functionality for completing given tasks are available to the user.

Notation 2. Region of interest of a component: In the following, let $R\text{<role>}_{id}$ denote the region of interest generated by component $C\text{<role>}_{id}$. Such a region is an element of the set $Regions(t_i) = \{R\text{rla}_{id1}, R\text{rla}_{id2}, \ldots, R\text{rlx}_{idy}, \ldots, R\text{rln}_{idm}\}$. This set represents at t_i, the spatial extension of the multi-tiered architecture.

At the geographic level, a region is defined by the spatial boundaries in which components can interact with each other. Such limits can be derived from the communication and range capabilities of the hardware. For example, a standard WiFi transmission limits regions to within $\approx$50m from their supporting components. In another region definition, system designers can limit the access to hardware components to nearby locations, so that only the closest components can share resources. Such definitions induce a bi-directional connection between two hardware components when both components are included in the spatial range of the other. Components $C\text{rla}_x$ and $C\text{rlb}_y$ are said to be *related* and verify the equality $Related(C\text{rla}_x, C\text{rlb}_y, t_i) = 1$ when such two-way communication is possible. Inter-component relation based on communication capabilities induces that a given piece of hardware $C\text{rla}_x$ is related to himself, and thus for all t_i, $\mathit{Related}(C\text{rla}_x, C\text{rla}_x, t_i) = 1$.

The set of active components gives rise to several spatial regions within the campus information system space. For example, at t_1, $\mathit{Regions}(t_1) = \{R\text{u}_{Jim}, R\text{u}_{John}, R\text{u}_{Jane}, R\text{p}_{1\rightarrow 3}, R\text{dh}_1, R\text{da}_1, R\text{de}_1\}$. In the campus assistant, the boundaries of the users' regions depend on the respective wireless capabilities of their mobile device. For example *Jim*'s, client's wireless access generates a region centred on $C\text{u}_{Jim}$, and within which he interacts with the system. Processing and data handling components are paired as they are accessible through the same hotspots. At a spatial level, $R\text{p}_{1\rightarrow 3}$ and $R\text{da}_1$, $R\text{dh}_1$, $R\text{de}_1$ respectively overlap. In contrast to the user regions, their boundaries have been purposely assigned so that the information about a group of buildings can only be accessed and processed nearby. For example, from his current location, *Jim* can only access information from the administrative departments. (Fig. 2a).

4.2 Clustering the Components and Grouping the Users

In typical group-based recommender systems, components that share similar functionality, content and context, derive common preferences. The grouping criterion however depends on the system and might be difficult to model. With the proposed modelling approach of a LBS, the set of role-assigned components along with their regions provide an immediate criterion to a grouping process. In the following, a *cluster* is defined as a group of communicating components. More specifically, the *related* components at t_i form clusters, and a component $C\text{rla}_x$ belongs to a cluster only when a component $C\text{rlb}_y$ exists in this cluster so that $\mathit{relate}(C\text{rla}_x, C\text{rlb}_y, t_i) = 1$ is verified. As a component always relates to itself, the isolated components of the system derive single-element clusters. More formally, let $\mathit{Cluster}(\ldots)$ return at t_i the set of clustered components[1]:

$$\mathit{Cluster}(t_k) = \bigcup_{\substack{C\text{rla}_i \in \\ \mathit{Platform}(t_k)}} \big(\mathit{Group}(C\text{rla}_i, \emptyset) - \emptyset\big)$$

with

$$\mathit{Group}(C\text{rla}_i, A) = \left\{ \begin{array}{l} C\text{rlb}_j \in \mathit{Platform}(t_k) \;\cup\; \mathit{Group}\big(C\text{rlb}_j, A \cup \{C\text{rlb}_j\}\big) \\ \mid\; \mathit{Relate}(C\text{rla}_i, C\text{rlb}_j, t_k) = 1 \;\wedge\; C\text{rlb}_j \notin A \end{array} \right\}$$

The individual boundaries of accessible components in the same cluster can be unified to highlight the spatial boundaries of a cluster. Figure 3 illustrates a configuration where cluster boundaries change as users move. For example, at time instant t_2, three clusters co-exists in the campus guide system, and $\mathit{Cluster}(t_2) = \big\{\{C\text{u}_{John}, C\text{p}_2, C\text{de}_1\}, \{C\text{u}_{Jane}, C\text{u}_{Jim}, C\text{p}_1, C\text{dh}_1\}, \{C\text{p}_3, C\text{da}_1\}\big\}$. In the configuration depicted, *Jane* and *Jim* belong to a same cluster as they are both able to send and receive information from the humanities hotspot (Fig.3a).

[1] $\mathit{Cluster}(\ldots)$ relies on a recursively defined $\mathit{Group}(\ldots)$ function. Given a component $C\text{rla}_i$ and the empty set $A = \emptyset$ as an input, this function completes and returns A with the tree of related components accessible to $C\text{rla}_i$.

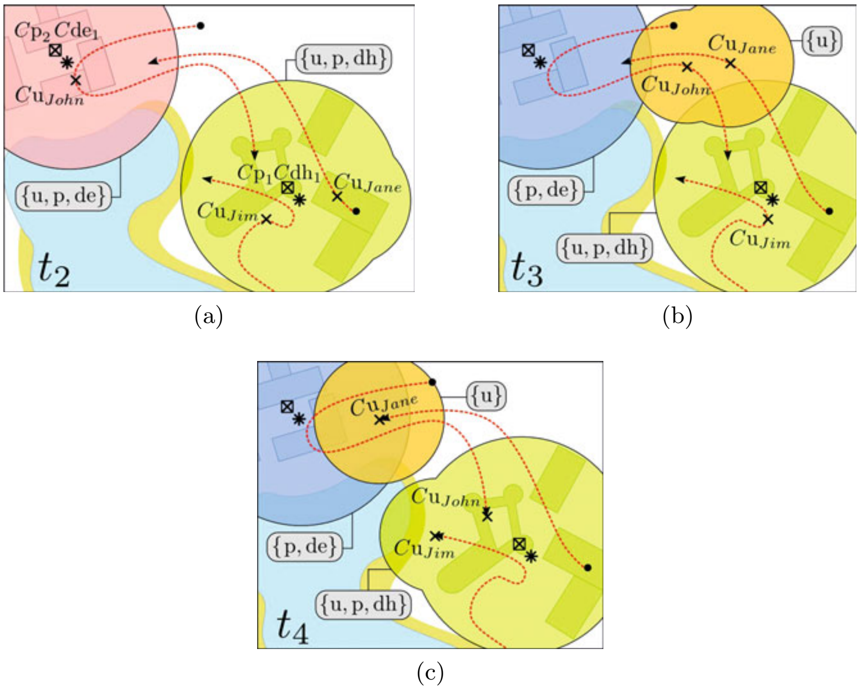


Fig. 3. Footprints of the changing set of clusters from t_2 to t_4 (only the moving components are labelled; Administrative depts. cluster is not displayed)

When *Jane* moves towards the science departments, at some time between t_2 and t_3, $Relate(C\mathrm{u}_{Jim}, C\mathrm{u}_{Jane}, t_i \in]t_2, t_3[) = 0$, and her component opens a fourth, self-contained, cluster. At t_3, *John* meets *Jane*, their components exchange information, and $Relate(C\mathrm{u}_{John}, C\mathrm{u}_{Jane}, t_3) = 1$. At the cluster level, *John* joins $Jane's$ newly created cluster (Fig. 3b). Again, the configuration changes at t_4 when *John* accesses the humanities hotspot and so now belongs to the cluster in green. Conversely, *Jim* loses his relationship with $C\mathrm{p}_1$ and $C\mathrm{dh}_1$ servers. However, thanks to a bridged connection through $C\mathrm{u}_{John}$, $Jim's$ component is still part of the cluster in green (Fig. 3c).

Clusters need to be uniquely identified in order to be paired with preference profiles. A signature ID is given by the set of roles that the components in a cluster assume. For example, in figure 3, the green cluster signature is $\{\mathrm{u}, \mathrm{p}, \mathrm{dh}\}$ (i.e.: roles "user", "processing" and "data Humanities"), the users-only clusters are identified by the singleton $\{\mathrm{u}\}$ and $\{\mathrm{de}, \mathrm{p}\}$ label a cluster made of science department hotspot components without users.

Properties of Clusters. With the objective of profiling user preferences and recommending content, the proposed spatial clustering induces the following properties:

– the components and the environmental context are shared among the users in a cluster;
– at the functional level, the data and tools offered to users are relative to each cluster.

Both properties by-pass the implicit profiling difficulty for grouping users, or components together in a recommender system (i.e.: *Grouping criterion* flaw). When users in a cluster contribute to a profile, they share a physical context and obtain access to the same subset of information and tools. Furthermore, the derivation of this grouping criterion occurs at no cost as users only need to identify neighbouring components corresponding to the cluster they belong to.

5 Profiling in a Clustered System

This section details the content and make-up of spatial and user profiles. An algorithmic approach to profile derivation is also introduced. In our attempt to address the drawbacks of recommender systems, the definitions of profiles and the suggested algorithm complement cluster properties.

5.1 Content and Types of Profiles[2]

The proposed profiles combine the preferences of one or several users with respect to the content delivered and the functionality offered in a given cluster. To be more specific, a profile gathers an n-element set of ('key', scr) pairs, associated with a cluster signature [Clust. ID] and a component $C\mathrm{rla}_{idx}$ which this profile applies to. For example, the contents of an individual profile (IP) is given by:

$$IP^{\text{[Clust. ID]}}_{C\mathrm{rla}_{idx}} \rightarrow \{(\text{'key}_1\text{'}, scr_1), (\text{'key}_2\text{'}, scr_2), \ldots, (\text{'key}_i\text{'}, scr_i), \ldots, (\text{'key}_n\text{'}, scr_n)\}$$

The 'key' parts identify pieces of information or functionality that are available in a cluster with signature [Clust. ID], while scr scores quantify a user's (or group of users) interest towards the associated 'key' elements. In a profile, a pair of scores scr_a and scr_b verifying $scr_a > scr_b$ acknowledges a user's or group preferences for the information or functionality 'key$_a$' over 'key$_b$'. Such scores are float numbers in $[0, 1]$, so that their sum in a profile equals 1 (i.e.: $\sum_{i=1}^{n} scr_i = 1$).

Ordering user elements of interest by attributing scores, makes personalising the geospatial services possible. Personalisation in this case involves highlighting content relevant to the current users and hiding content which is not of interest to them. Similarly, the interface and functionality used to display and interact with the content can be adapted according to such preference profiles.

In the campus assistant *Jane's* individual profile at t_1 highlights her interest for geography (Tab. 1a). *Jane's* client consequently emphasises geography

[2] This section summarises the authors previous work, without detailing much of the actual profile content derivation. A complete description of personal profile derived from user's actions is given in [11].

Table 1. Contents of *Jane* personal profiles

(a) at t_1, within Humanities hostpot area	(b) at $t_i \in]t_2, t_3[$
$IP^{\{\text{u,p,dh}\}}_{Cu_{Jane}} \rightarrow \{(\text{'Geog.'}, .4,), (\text{'Libr.'}, .1), (\text{'Hist.}, .2)$ $(\text{'Liter.'}, .2), (\text{'InfoPane'}, .07), (\text{'MapPane'}, .03)\}$	$IP^{\{\text{u}\}}_{Cu_{Jane}} \rightarrow \{(\text{'InfoPane'}, .2),$ $(\text{'MapPane'}, .8)\}$

related elements: the labels have been adapted, and the content panel automatically loads information about this department (Fig. 2b). The layout of the user interface also adapts to $Jane's$ current preference for descriptive content rather than a map. Conversely, $Jane's$ client infers her preference for campus mapping when she is alone between t_2 and t_3 (Tab. 1b). Accordingly, her client emphasises the map during this period.

The definition of profiles allows their contents to be easily mixed and average or historical profiles to be derived. This approach for constructing profiles discriminates several levels locally to each cluster:

– *individual profiles (IP)* gather the preferences and scores of a single user. At every time during execution, a client adapts its content and display to the individual profile of the current cluster. Table 1 provides examples of the content of such personal profiles;
– *immediate group profiles (IGP)* account for the averaged preferences of the users currently grouped in a cluster. Such profiles continually combine the individual profiles of users in a cluster on a per-value basis. The preferences depicted in group profiles reflect the most favoured content and functionality among the users;
– *strengthened group profiles (SGP)* balance the immediate group profiles with previously derived scores. For example, SGP at t_i averages the current immediate group profile and the previously derived SGP at t_{x-1}. These profiles are less sensitive to sudden variations of user preferences than immediate profiles.

For example, when Jim enters the humanities cluster at t_2, his component computes an immediate group profile $IGP^{\{\text{u,p,dh}\}}_{Cu_{Jim}}$ and subsequently adapts his user interface and content. Such profile derivation averages the individual profiles of all components in the cluster and can be summarised by the operation in fig. 4. Although the standard deviation of scores have been reduced by $Jim's$ calibrated profile, the derived group profile still reflects $Jane's$ interest for geography. The server components also take part in the immediate group profile

$$\begin{aligned}
&\big(IP^{\{\text{u,p,dh}\}}_{Cu_{Jane}} \rightarrow \{(\text{'Geog.'}, .4,), (\text{'Libr.'}, .1), (\text{'Hist.}, .2), (\text{'Liter.'}, .2), (\text{'InfoPane'}, .07), (\text{'MapPane'}, .03)\} \; + \\
&\; IP^{\{\text{u,p,dh}\}}_{Cu_{Jim}} \rightarrow \{(\text{'Geog.'}, .16,), (\text{'Libr.'}, .16), (\text{'Hist.}, .16), (\text{'Liter.'}, .16), (\text{'InfoPane'}, .16), (\text{'MapPane'}, .16)\} \\
&+ IP^{\{\text{u,p,dh}\}}_{Cdh_1} \rightarrow \{\dots\} + IP^{\{\text{u,p,dh}\}}_{Cp_1} \rightarrow \{\dots\}\big)/4 \\
&= IGP^{\{\text{u,p,dh}\}}_{Cp_{jim}} \rightarrow \{(\text{'Geog.'}, .28,), (\text{'Libr.'}, .13), (\text{'Hist.}, .18), (\text{'Liter.'}, .18), (\text{'InfoPane'}, .12), (\text{'MapPane'}, .09)\}
\end{aligned}$$

Fig. 4. *Jim* constitution of an immediate group profile at t_2

derivation. Their contribution depends on the sharing algorithm, detailed in the next section.

Profiles are stored by each component belonging to a cluster. Immediate and strengthened group profiles are identical for all components of the cluster. Individual profile scores differ on each client device, while they are equal at the server component level (i.e.: every score of a n-elements profile equals $1/n$). In the following $(IP|IGP|SGP)^{\text{[Clust. ID]}}_{C\text{rla}_{idx}}$ denotes the individual, immediate group, and strengthened group profiles of the component $C\text{rla}_{idx}$, locally in the cluster identified by [Clust. ID].

Properties of profiles. With regards to the usual recommender system drawbacks, different profile levels induce beneficial properties:

– users of the system are able to switch their active individual profile to either an immediate or strengthened group profile that are available in the cluster they belong to, whatever their adaptation policy is;
– profile choices can also be made automatically by the recommender system on a client device. For example, group profiles can be favoured only when a minimum number of users is reached;
– even when no other users currently belong to a cluster, a previous group profile should exist on one of the components when a newcomer enters a cluster.

Applying an individual profile is advantageous for a user who does not feel at ease with group preferences, and wants to avoid being *stereotyped*. New users are invited to adapt their content to a group profile; they therefore preserve themselves from a *cold start* period of undetermined preferences. Differences between IGP and SGP also help the recommender system to find the appropriate level of *content inertia*.

5.2 Maturing the Profiles in a Mobile System

The mobile nature of a LBS favours an additional solution to spread the preferences and avoid too much inertia in user profiles. Moving clients and servers can carry profiles constructed in distant clusters and share them in a peer-to-peer way. A set of components gathering in a cluster share not only profiles identified by this cluster but also any other profiles of different clusters they might have in common. Algorithm 1 summarises the proposed peer sharing methodology. This pseudo-code runs in a loop on each component of the system.

At first, a given component and their neighbouring devices exchange roles to identify the cluster they belong to (l.2). If necessary, IP, IGP and SGP profiles are allocated with all scores valued equally (l.3$\rightarrow$5). Sharing and averaging scores occurs for all existing cluster IDs defined among the profiles allocated (l. 6). The sharing and merging procedure collects the personal profiles with ID [AnyClust] of the components in the current cluster and derives an averaged immediate profile as an output (l.7). In turn, the produced output combines to

Algorithm 1. Iterative profiles derivation at a given component $C\text{rla}_{idx}$ level

```
1: loop
2:   compute current cluster ID as [MyClust]
3:   if ! ($IP^{[\text{MyClust}]}_{C\text{rla}_{idx}}$ allocated) then
4:     allocate equal profiles: $IP^{[\text{MyClust}]}_{C\text{rla}_{idx}}$, $IGP^{[\text{MyClust}]}_{C\text{rla}_{idx}}$, and $SGP^{[\text{MyClust}]}_{C\text{rla}_{idx}}$
5:   end if
6:   for all (individual profile $IP^{[\text{AnyClust}]}_{C\text{rla}_{idx}}$ allocated) do
7:     $IGP^{[\text{AnyClust}]}_{C\text{rla}_{idx}} \leftarrow ShareAndMerge([\text{AnyClust}])$
8:     $SGP^{[\text{AnyClust}]}_{C\text{rla}_{idx}} \leftarrow Merge(IGP^{[\text{AnyClust}]}_{C\text{rla}_{idx}}, SGP^{[\text{AnyClust}]}_{C\text{rla}_{idx}})$
9:     if (Favour *Immediate group profile* policy) then
10:      $IP^{[\text{AnyClust}]}_{C\text{rla}_{idx}} \leftarrow IGP^{[\text{AnyClust}]}_{C\text{rla}_{idx}}$
11:    else if (Favour *Strengthened group profile* policy) then
12:      $IP^{[\text{AnyClust}]}_{C\text{rla}_{idx}} \leftarrow SGP^{[\text{AnyClust}]}_{C\text{rla}_{idx}}$
13:    end if
14:  end for
15:  $adapt()$ $C\text{rla}_{idx}$ to $IP^{[\text{MyClust}]}_{C\text{rla}_{idx}}$
16:  $infer()$ preferences and scores until $t_{x+1}$
17:  $update()$ $IP^{[\text{MyClust}]}_{C\text{rla}_{idx}}$ with the infered scores
18: end loop
```

the last SGP profile and is stored as a newer version (l.8). Depending on the current component policy, the individual profile scores are updated to either IGP or SGP values; or are left unchanged (l.9→13). Lines 15 to 17 are built on existing work and is where the adaptation of the content presented, and the implicit perception of user preferences occur. The procedures *adapt*(), $infer()$ and *update*() have been defined in [11]. Overall, these functions act together as a recommender system on the client device: they infer preferences, complete the individual profile scores, and update the layout accordingly.

The profile sharing behaviour described in the case study are derived from the proposed algorithm. For example, at t_3, sharing profiles between *Jane* and *John* do not solely focus on their current user-only cluster (i.e.: with ID {u}), but also encompass the previously derived profiles they carry. Thanks to $Jane's$ experience of the system, $John's$ client embeds a humanities related IGP even if he stands outside of the boundaries of the humanities cluster. $John's$ decision not to adapt his content to this immediate group profile reflects a switch in his policy that favours personal instead of group profiles.

Properties of the Sharing Algorithm. The profile derivation algorithm encompasses typical features to improve preference sharing and recommendations in location-based services:

- profiles assigned to clusters are also shared outside of the cluster's context. The profiles of users can therefore be updated before they actually belong to a cluster;

– servers and user components are treated alike. Although preferences are not inferred and content is not adapted on servers, all the components of $Platform(t_i)$ share the profiles in a peer-to-peer approach.

Since the scores of any profile can be updated in every cluster, having a substantial number of components running the profile derivation algorithm is advisable to settle *inertia of profile content*. Due to the movement of components and interactions of devices, the content of profiles does not remain static for long.

6 Discussion

Recommender systems have been introduced as a solution to the increasing difficulty to sort and present relevant information to users of location-based services. In such systems, inferring preferences and adapting the content and interface is valuable to the user. Implicit as well as collaborative profilers sometimes fail to achieve this objective due to several technical and/or design drawbacks as mentioned in section 2. This paper highlights the properties of LBS that can resolve some profiling challenges for mobile systems. The concept has been implemented through a prototype simulator. This tool allows different recommender system behaviours to be tested. This includes the modelling of different user mobility (with pauses) as well as different interaction modes based on mouse tracking. The simulator generates group profiles that can produce adapted interfaces and allows for comparison between variations of different profiles.

Table 2. Summary of the proposed improvement sources with regard to the usual challenges a recommender system faces

	Grouping criterion	*Inertia of content*	*Cold start*	*Stereotyped users*
Regional clustering	components undergo a same functional and environmental context	-	-	-
	low-resources signatures derivation			
Profile nature and content	-	profiles types *IGP* and *SGP* differ by their content inertia	ready to use profiles when entering a cluster	user or system policy to adapt to the most appropriate type.
Sharing algorithm	-	clusters profiles are shared outside of their boundaries	-	-
		each component produces and shares profiles		

The proposed approach models a LBS as a set of regions whose relationships give rise to clusters of active users and components. Profiles are built locally for each cluster and rank the data and functionality offered within a cluster. Several types of profiles, either personal or group-based, have been introduced. Depending on their experience of the system, users can switch from one type to another and find the profile that best suits their preferences. An algorithm to share and merge profiles across the system space benefits from user and component mobility and enriches profiles with up-to-date content scores. Clusters, profile descriptions, and sharing algorithms constitute distinct layers that together improve traditional recommender systems. Several properties have been explored in the paper and are summarised in table 2.

The potential positive properties of our hybrid approach can be discussed and raise additional open questions. While one of the main benefits of the approach, is the ability to provide different types of profiles, which may improve the *inertia* and *cold start* issues, these are not terms or concepts familiar to average users. As a result, inexperienced users may not alternate between profiles, choosing to remain with the default one. A challenge is therefore, to inform users on when to switch profiles and the benefits this can bring to their interaction with the system. This can be achieved at the interface level by providing information about the contents of other types of profile.

In the system described here, the cluster arrangement is preconfigured based on the underlying computer network infrastructure. In the case of the campus navigation assistant, this corresponds to the physical layout of the buildings. Generating such clusters creates an overhead for the developer and system designer. It would be more advantageous for the cluster arrangement to form naturally based on the underlying infrastructure. The implementation of this would enable a generic system to be developed and used in various contexts.

Many of the limitations discussed above can be alleviated through further development of the approach and refinement of the algorithms. For example, fine tuning of the algorithm for assigning weights to preferences within profiles can improve the recommendations returned to users. Generally, such details are dependent on the specific context of use which must be taken into consideration. Technical details on how devices share profiles also needs to be investigated. The approach described here is robust and can be broadened to not only include places or regions visited within the context of a small geographical area, like a campus for example, but also for wider interactions. As people share more and more location-based information through social interaction and networks, this data can be used to produce region profiles on a national and international level.

7 Conclusion

This paper has introduced a new type of profile, the region profile, which offers an innovative technique for personalising location-based services. Individual user profiles are generated by monitoring user interactions with the physical environment and online content. These robust profiling techniques can be used to produce relevant recommendations at the interface level. To solve issues with recommender systems, such as the well-known *cold start* and *inertia* problems, region profiles are introduced. Regions of interest are generated by the components of an underlying distributed infrastructure. The individual profiles of users entering such regions get combined to create a profile for that region.

This emerging region profile then contributes to individual recommendations and profiles. By empowering the user with the possibility of using their personal profile, that of the region or a hybrid, appropriate recommendations can be delivered from a user's first interaction with the system. A case-study of a college campus is provided to illustrate the approach. Work on refining the algorithms and the technical details of how profiles are shared is ongoing.

Acknowledgement. Research presented in this paper was funded by a Strategic Research Cluster grant (07/SRC/I1168) by Science Foundation Ireland under the National Development Plan and by a grant (Ulysses Programme) from the Irish Research Council for Science, Engineering and Technology (IRCSET) and the French ministry of foreign affairs. The authors gratefully acknowledge this support.

References

1. Balabanović, M., Shoham, Y.: Fab: content-based, collaborative recommendation. Communications of the ACM 40(3), 66–72 (1997)
2. Ballatore, A., McArdle, G., Kelly, C., Bertolotto, M.: Recomap: An interactive and adaptive map-based recommender. In: Proceedings of the 25th ACM Symposium on Applied Computing (SAC), pp. 887–891. ACM (2010)
3. Burke, R.: Hybrid recommender systems: Survey and experiments. User Modeling and User-Adapted Interaction 12(4), 331–370 (2002)
4. Gartner, G., Cartwright, W.E., Peterson, M.P. (eds.): Location-Based Services and TeleCartography. Lecture Notes in Geoinformation and Cartography. Springer, Heidelberg (2007)
5. Gartner, G., Rehrl, K. (eds.): Location-Based Services and TeleCartography II. Lecture Notes in Geoinformation and Cartography. Springer, Heidelberg (2009)
6. Gupta, G., Lee, W.-C.: Collaborative spatial object recommendation in location based services. In: International Conference on Parallel Processing Workshops, pp. 24–33 (2010)
7. Kelly, D., Teevan, J.: Implicit feedback for inferring user preference: a bibliography. ACM SIGIR Forum 37(2), 18–28 (2003)
8. Lam, W., Mukhopadhyay, S., Mostafa, J., Palakal, M.: Detection of shifts in user interests for personalized information filtering. In: SIGIR 1996: Proc. of the 19th International Conference on Research and Development in Information Retrieval, pp. 317–325. ACM (1996)
9. Longley, P.A., Goodchild, M.F., Maguire, D.J., Rhind, D.W.: Geographical Information Systems and Sciences, 2nd edn., 517 pages. John Wiley and Sons (2005)
10. Mac Aoidh, E., Bertolotto, M., Wilson, D.C.: Understanding geospatial interests by visualising map interaction behaviour. Information Visualization 7(3-4), 257–286 (2008)
11. Mac Aoidh, E., McArdle, G., Petit, M., Ray, C., Bertolotto, M., Claramunt, C., Wilson, D.: Personalization in adaptive and interactive gis. Annals of GIS 11(1), 23–33 (2009)
12. Petit, M., Ray, C., Claramunt, C.: A user context approach for adaptive and distributed GIS. In: Proceedings of the 10th International Conference on Geographic Information Science (AGILE 2007), Aalborg, Denmark. Lecture Notes in Geoinformation and Cartography, pp. 121–133. Springer, Heidelberg (2007)

13. Rich, E.: Users are individuals: individualizing user models. Interntational Journal of Man-Machine Studies 18(3), 199–214 (1983)
14. Satyanarayanan, M.: Pervasive computing: Vision and challenges. IEEE Personal Communications 8(4), 10–17 (2001)
15. Schein, A.I., Popescul, A., Ungar, L.H., Pennock, D.M.: Methods and metrics for cold-start recommendations. In: Proceedings of the 25th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval, pp. 253–260. ACM, New York (2002)
16. Tahir, A., McArdle, G., Ballatore, A., Bertolotto, M.: Collaborative filtering - a group profiling algorithm for personalisation in a spatial recommender system. In: Geoinformatik 2010 (2010)
17. Wilson, D., Doyle, J., Weakliam, J., Bertolotto, M., Lynch, D.: Personalized maps in multimodal GIS. International Journal of Web Emerging Technology 3(2), 196–216 (2007)
18. Wu, D., Zhao, D., Zhang, X.: An adaptive user profile based on memory model. In: Web-Age Information Management, pp. 461–468 (2008)
19. Yang, Y., Claramunt, C.: A Hybrid Approach for Spatial Web Personalization. In: Li, K.-J., Vangenot, C. (eds.) W2GIS 2005. LNCS, vol. 3833, pp. 206–221. Springer, Heidelberg (2005)

Enforcing Protection Mechanisms for Geographic Data

Alban Gabillon and Patrick Capolsini

Université de la Polynésie Française,
BP 6570, 98702 FAA'A, French Polynesia
{Alban.Gabillon,Patrick.Capolsini}@upf.pf

Abstract. In the framework of a geographic application displaying maps, there are several solutions for protecting a sensitive object. Sensitive objects can be hidden, masked, blurred or even replaced by fake objects. In this paper we suggest a framework to specify protection mechanisms to enforce whenever a prohibition is derived from the security policy. This framework includes (i) logical rules allowing us to derive protection mechanisms from prohibitions, and (ii) an algorithm which builds the map to display, according to the derived protection mechanisms.

Keywords: Access Control, Geo-spatial Data visualization, Map service, Policy Enforcement Point.

1 Introduction

Given a query, the security policy of a database application specifies which objects are authorized and which objects are unauthorized. In a traditional database approach, enforcement of the security policy is simply done by removing the unauthorized objects from the final answer to the query. In a geographic database application, things are more complicated. Let us consider a map service building maps from various spatial objects. In such an application, there are several methods for protecting unauthorized objects. Some unauthorized objects are simply removed from the final map as it is the case in a traditional database application, but some other sensitive objects are protected by using methods which are specific to geographic applications. Examples of such specific methods are referred to as *blurring*, *masking*, *pixelization* or "*cut and paste*" (i.e. overlay a sensitive object with another fake objects). Figure 1 shows some examples of sensitive objects which were protected by using such methods in well known geographic applications. These examples are all taken from [1]. The top left map shows a masked area at the state border between Yukon and Alaska. The top right map shows a pixelizated factory at Toulouse in southwestern France. On the bottom left map, part of the Michael Army Airfield (Utah) is blurred. On the bottom right map a fake landscape object overlays a sensitive military area of Xinshe in Taiwan.

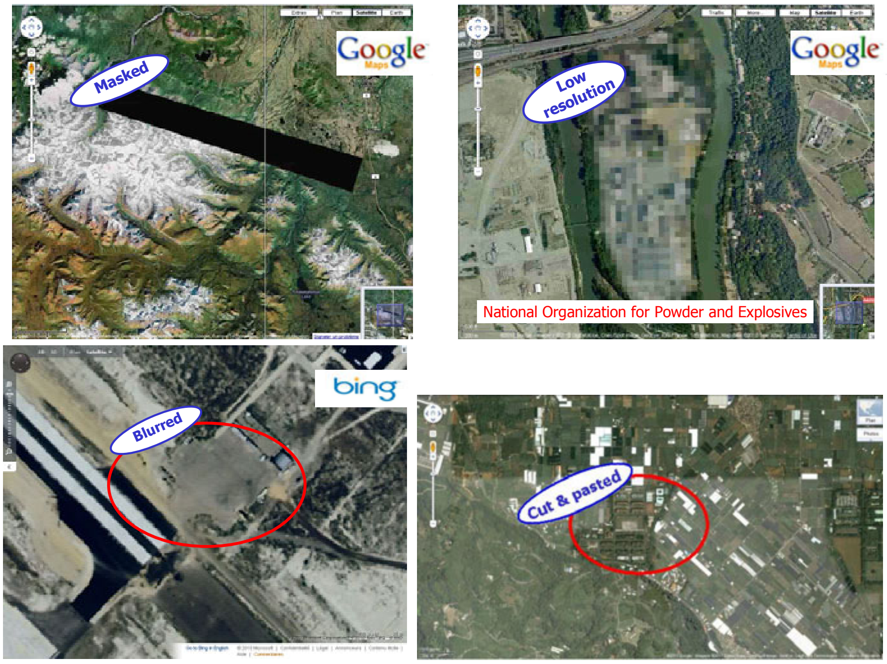


Fig. 1. Sensitive spatial objects protected with various methods

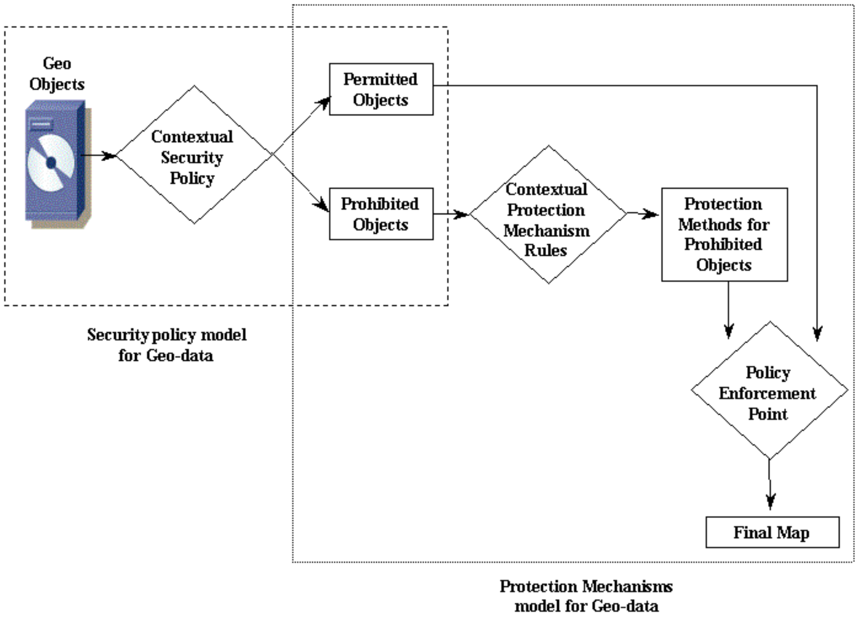

Geo Objects
Contextual Security Policy
Permitted Objects
Prohibited Objects
Contextual Protection Mechanism Rules
Protection Methods for Prohibited Objects
Policy Enforcement Point
Final Map
Security policy model for Geo-data
Protection Mechanisms model for Geo-data

Fig. 2. Protection Mechanisms Model

Existing security models [2-5] for geo-spatial applications focus on how to express security policies. Figure 2 shows that the work presented in this paper is located downstream of these models. The studies cited above deal with what is represented inside the dashed rectangle. They provide models for expressing contextual security policies for geographic data. Given a query that addresses a set of spatial objects, the security policy determines which objects are authorized and which objects are not authorized. The work presented in this paper deals with what is inside the dotted rectangle. It breaks down as follows:

- It defines a formal framework for deriving security mechanisms to be enforced on unauthorized objects.
- It proposes a Policy Enforcement Point (PEP) algorithm to display the final map taking into account the protection mechanisms that should be enforced on unauthorized objects.

To our knowledge, we are the first to propose a complete formal framework for specifying the mechanisms that should be enforced on prohibited objects. Let us mention, that we published a preliminary version of this work as a position paper in [6].

In section 2 of this paper, we felt the need to recall the basics of a security model we already defined in [4] and [5]. This security model allows us to express contextual security policies for geographic applications. In section 3 we define our logical framework for specifying *protection rules*. In section 4, we define the PEP algorithm which enforces *protection mechanisms* and builds the map to display. In section 5 we illustrate our proposal with a complete application example. In section 6, we give a sketch of the implementation of our proposal within the framework of the OpenGIS® Styled Layer Descriptor (SLD) Profile [7] of the OpenGIS® Web Map Service (WMS) Encoding Standard [8]. A prototype showing the feasibility of our approach can be found at the following url:

http://pages.upf.pf/Patrick.Capolsini/rech/protect/index.htm.

In section 7, we review related works. Finally section 8 concludes this paper.

2 Security Policy Model

In [4] and [5], we proposed a security policy model for geographic applications, based on the OrBAC model [9] and the ABAC model [10]. Our model considers dynamic spatial security rules. A spatial dynamic security rule can be activated or deactivated depending on some *spatial context*. Generally, a spatial context is considered to be a spatial condition that holds on the subject and/or the object. In [4], we identified and modelled various types of spatial contexts based on the user location and/or the spatial object location. We also showed how to model geo-temporal contexts and contexts related to movement. In [5], we focused on visualization of geo-data i.e. we showed how to model various types of *visualization contexts* (such as zoom-in factor, layers transparency, brightness …etc) for geo-data and how to express dynamic security rules based on such contexts.

Since this paper focuses on policy implementation and not on the security policy itself, we present here a simplified version of our model. It mainly defines a logical language for expressing security policies for geographic applications. Note, however, that we also use the predicates and functions defined in this section to express our protection rules in section 3.

In section 2.1 we define the objects of our model. In sections 2.2 and 2.3 we define elements of our language for writing security policies. In section 2.4, we define the concept of spatial query. In section 2.5 we define authorization rules.

2.1 Geometric Entities

A *georeferenced (geometric) object* is a granule of information that is relevant to an identifiable subset of the Earth's surface [11]. Any geometric object has the following two components [12] : a *description*: the entity is described by a set of descriptive attributes (e.g. the name of a city) and a *geometry* which indicates the entity's location and its shape. The geometry model we consider is the OpenGIS Geometry Model [13].

2.2 Spatial Analysis Functions

Spatial analysis functions take one or more geometric objects as input and return either a number or another geometric object. We consider the following functions. Let *a* and *b* be two geometric objects and *x* a scalar:

- *distance(a,b)* – Returns the shortest distance (a scalar) between any two points in the two geometric objects *a* and *b*
- *buffer(a,x)* – Returns a geometric object that represents all points whose distance from geometric object *a* is less then or equal to *x*
- *convexHull(a)* – Returns a geometric object that represents the convex hull (mathematical definition) of geometric object *a*
- $a \cap b$, $a \cup b$, $a \setminus b$, $a \Delta b$, – Respectively returns a geometric object that represents the point set intersection (resp. union, resp. difference, resp. symmetric difference) of object *a* with object *b*
- *I(a)*, *B(a)*, *E(a)* and *dim(a)* respectively returns the interior, boundary, exterior and dimension (-1 for the empty geometry Ø, 0 for *Point*, 1 for *Linestring* and 2 for *Polygon*) of *a*.
- *speed(a)* – Returns the speed of the object. The speed is a scalar value greater than or equal to 0.
- *direction(a)* – Returns the direction taken by the object. The direction is an angle value between 0 and 360 degrees. It is equal to N/A (Not Applicable) if the speed is equal to 0.

2.3 Spatial Predicates

Spatial predicates are used to test for the existence of a specified topological relationship between two geometric entities. Using functions *I(g)* and *dim(g)* returning respectively the interior and dimension of geographic object *g*, [13] defines eight spatial predicates namely, *Equals*, *Disjoint*, *Intersects*, *Touches*, *Crosses*, *Within*, *Contains* and *Overlaps*.

$$\forall g_1, \forall g_2, \mathit{Equals}(g_1, g_2) \leftrightarrow (g_1 \cap g_2) = (g_1 \cup g_2) \tag{1}$$

$$\forall g_1, \forall g_2, \mathit{Disjoints}(g_1, g_2) \leftrightarrow g_1 \cap g_2 = \varnothing \tag{2}$$

$$\forall g_1, \forall g_2, \mathit{Touches}(g_1, g_2) \leftrightarrow (I(g_1) \cap I(g_2) = \varnothing) \wedge (g_1 \cap g_2 \neq \varnothing) \tag{3}$$

$$\begin{aligned} &\forall g_1, \forall g_2, \mathit{Crosses}(g_1, g_2) \leftrightarrow \\ &(\mathit{dim}(I(g_1) \cap I(g_2)) < \mathit{max}(\mathit{dim}(g_1), \mathit{dim}(g_2))) \wedge (g_1 \cap g_2 \neq g_1) \wedge (g_1 \cap g_2 \neq g_2) \end{aligned} \tag{4}$$

$$\forall g_1, \forall g_2, \mathit{Within}(g_1, g_2) \leftrightarrow (g_1 \cap g_2 = g_1) \wedge (I(g_1) \cap E(g_2) \neq \varnothing) \tag{5}$$

$$\forall g_1, \forall g_2, \mathit{Contains}(g_1, g_2) \leftrightarrow \mathit{Within}(g_2, g_1) \tag{6}$$

$$\begin{aligned} &\forall g_1, \forall g_2, \mathit{Overlaps}(g_1, g_2) \leftrightarrow (\mathit{dim}(I(g_1)) = \mathit{dim}(I(g_2)) = \mathit{dim}(I(g_1) \cap I(g_2))) \\ &\qquad \wedge (g_1 \cap g_2 \neq g_1) \wedge (g_1 \cap g_2 \neq g_2) \end{aligned} \tag{7}$$

$$\forall g_1, \forall g_2, \mathit{Intersects}(g_1, g_2) \leftrightarrow \neg \mathit{Disjoint}(g_2, g_1) \tag{8}$$

2.4 Spatial Query

In the framework of a map service a spatial query outputs a map. This map is constructed from a set of geo-referenced objects which are all displayed at the same zoom-in factor. This zoom-in factor is a parameter of the query.

Let *q* be a spatial query. We denote *O(q)*, the set of objects addressed by query *q* and *zf(o)* the zoom-in factor of object *o*. This zoom-in factor is inherited from query *q* and is the same for all objects addressed by query *q*.

2.5 Contextual Authorization Rules

Security rules specify how subjects can execute actions on objects. Our model includes permissions (positive rules) and prohibitions (negative rules). Given a query, authorized objects addressed by the query are used to build up the map.

We define a positive authorization rule as a logical rule having the following form:

$$\forall s \forall o (\mathit{Condition} \rightarrow \mathit{Permit}(s, o)) \tag{9}$$

Permit(s,o) reads “s is permitted to view object o.”

We define a negative authorization rule as a logical rule having the following form:

$$\forall s \forall o (Condition \rightarrow Deny(s,o)) \tag{10}$$

Deny(s, o) reads “s is forbidden to view object o.”

In both rules *Condition* is a logical expression used to express some properties regarding the subject, the object and the context.

Let us consider the following example of security policy which consists of the four following rules:

The first security rule says that civilians are forbidden to view tanks.

$$\forall s \forall o (Civilian(s) \wedge Tank(o) \rightarrow Deny(s,o)) \tag{11}$$

The second security rule says that civilians are forbidden to view barracks at a zoom-in factor greater than 1.

$$\forall s \forall o (Civilian(s) \wedge Barrack(o) \wedge zf(o) > 1 \rightarrow Deny(s,o)) \tag{12}$$

The third security rule says that soldiers have the permission to view tanks:

$$\forall s \forall o (Soldier(s) \wedge Tank(o) \rightarrow Permit(s,o)) \tag{13}$$

The fourth security rule says that soldiers do not have the permission to view tanks which are not within the military zone:

$$\forall s \forall o (Soldier(s) \wedge Tank(o) \wedge \neg Within(o, MilitaryZone) \rightarrow Deny(s,o)) \tag{14}$$

Note that there is a conflict between the last two rules regarding tanks which are not within *MilitaryZone* area. It is not the purpose of this paper to discuss this issue. The reader can refer to [4], where we use the conflict resolution strategy defined in the OrBAC model. This conflict resolution strategy is based on separation constraints and priorities assigned to rules. Our first aim, in this paper, is to devise a logical framework to specify the security mechanisms which are to be enforced whenever we derive an instance of the *Deny* predicate from the security policy, regardless of the conflict resolution strategy which is used. We define this framework in the next section.

3 Contextual Protection Rules

3.1 Definition

In this section, we define a complete framework for specifying the protection mechanism which should be enforced in case a user is denied to view a given object. The logical language we use is based on the language we defined in the previous section.

A *Protection Rule* is a rule of the form:

$$\forall s \forall o \big(Condition \wedge Deny(s,o) \rightarrow Protect(o,M) \big) \tag{15}$$

As it is the case with authorization rules, *Condition* is used to express some properties that should hold on the subject, the object and the context. This means in particular that given an unauthorized object, the protection mechanism that should be enforced may vary from one context to another.

Protect(o,M) reads "*o* should be protected with mechanism *M*". *M* is a protection mechanism function which is one of the followings:

Let *g* be a geometric object and *i* a scalar:

- *reject_query*: reads "reject the query which requires *o* to be displayed" i.e. empty map is returned even if the query addressed some authorized objects.
- *blur* : reads "blur *o*"
- *mask*: reads "mask *o*"
- *pixelizate*: reads "lower *o*'s resolution".
- *hide*: reads "remove *o*"
- *paste(g)*: reads "cut and paste g" i.e. "overlay *o* with *g*". . *paste(g)* is very often used to build what is referred to as a *cover story* i.e. a lie. It can be a fake object which does not exists in the real world. It can be an existing object from which some visual details have been hidden. It can be an existing object, but shown at an incorrect location etc.
- *zoom_in(i)*: reads "forces the zoom-in factor of object *o* to a value which is less than or equal to *i*". As we will see in section 4, this protection method would also decrease the zoom-in factor of other objects, even authorized ones, since given a map the zoom-in factor is the same for all objects.

Note that, to define our model, we do not need to enter into the details of the *blur*, *mask* and *pixelizate* functions. However, in a real implementation, these protection mechanisms would require some parameters such as the intensity for the *blur* function, the shape and the position of the mask for the *mask* function and the resolution for the *pixelizate* function.

Let us consider the following two examples of protection rules:

$$\forall s \forall o \big(Tank\ (o) \wedge Deny\ (s,o) \rightarrow Protect\ (o, hide\ ) \big) \tag{16}$$

$$\forall s \forall o \big(Barrack\ (o) \wedge Deny\ (s,o) \rightarrow Protect\ (o, zoom\ \_in(1)) \big) \tag{17}$$

The first rule says that if someone who is forbidden to see tanks request to see them then tanks should be removed from the returned map. The second rule says that someone who is forbidden to see barracks can in fact see them but at a zoom-in factor equal to 1.

If for a given prohibition there is no specific protection rule then a default mechanism applies. This default mechanism depends on the application. For example the following default rule says that the default mechanism is *blur*.

$$\forall s \forall o \big(Deny\,(s,o) \rightarrow Protect\,(o, blur\,) \big) \tag{18}$$

If for a given prohibition, several mechanisms can be derived then only one of them should be selected. Such selection should be done on a priority basis. However, we have two options for assigning priorities:

- either we assign priorities to the mechanism themselves. For example, the following list could represent the hierarchy of mechanisms (from the lowest priority to the highest priority): {*zoom-in, pixelizate, blur, mask, paste, hide, reject_query*},
- or we assign priorities to protection rules (with the default rule having the lowest priority).

In section 4, we design a Policy Enforcement Point (PEP) algorithm which works with both approaches.

3.2 Merging Prohibitions and Protection Rules

In our model, we distinguish between the prohibitions and the protection rules. However, we could envisage merging the two types of rules as follows:

$$\forall s \forall o \big(Condition \rightarrow Deny(s,o) \wedge Protect(o, M\,) \big) \tag{19}$$

In the above definition, we directly specify in the prohibition rule the protection mechanism that should be enforced. If we apply this principle to the example described in section 3.1, then rules 11, 12 and 14 are merged with rules 16 and 17 as follows:

$$\forall s \forall o \big(Civilian(s) \wedge Tank(o) \rightarrow Deny(s,o) \wedge Protect(o, hide) \big) \tag{20}$$

$$\forall s \forall o \begin{pmatrix} Civilian(s) \wedge Barrack(o) \wedge zf\,(o) > 1 \\ \rightarrow Deny(s,o) \wedge Protect(o, zoom\_in(1)) \end{pmatrix} \tag{21}$$

$$\forall s \forall o \begin{pmatrix} Soldier(s) \wedge Tank(o) \wedge \neg Within(o, MilitaryZone) \\ \rightarrow Deny(s,o) \wedge Protect(o, hide) \end{pmatrix} \tag{22}$$

The obvious advantage of merging prohibitions and protection rules is that we end up with managing only one set of rules. However, this approach has the following disadvantages:

- If we need to enforce the same protection mechanism for several different prohibitions then we have to specify this mechanism in each of the prohibition rules. For example, we had to specify that the protection mechanism should be *hide* in rules 20 and 22.
- We reduce the expressive power of our model. In rule 19, the same condition triggers both the prohibition and the protection mechanism, whereas in rules 10 and 15, the condition triggering the prohibition can be different from the condition triggering the protection mechanism.

In the remainder of this paper we will not consider any more the possibility of merging the two types of rules since we want our model to have the highest possible expressive power. However, from a practical point of view, we are perfectly aware that a single set of rules might be easier to manage than two separate sets of rules.

4 Policy Enforcement Point Algorithm

In this section we define an algorithm for (i) enforcing protection mechanisms and (ii) construct the map to display. *O(q)* denotes the set of objects addressed by query *q*. *zf(q)* denotes the zoom-in factor of query *q* (see section 2.4). *map* denotes the map to construct. *empty_map* denotes the empty map. *minzf* denotes the zoom-in factor at which the final map is going to be displayed. *insert(o,map)* denotes a procedure which inserts geo-referenced object *o* into map *map*. *overlay(o,g)* denotes a procedure which overlays geo-referenced object *o* with geo-referenced object *g*. *mask(o)* denotes a function which overlays *o* with a mask*, blur(o)* denotes a function which blurs *o*, *pixelizate(o)* denotes a function which lowers *o*'s resolution. *applyzf(i,map)* is a function which applies zoom-in factor *i* on map *map*.

```
/* Protect(o,M) should be read "Protect(o,M) can be derived from
the Protection Rules" */
map ← empty_map
minzf ← zf(q)
For o in O(q)
   If Protect(o,reject_query) then
      Return(empty_map)
   Else
      If NOT Protect(o,hide) then
         insert(o,map)
         If  Protect(o,paste(g)) then
            overlay(o,g)
         Else
            If Protect(o,mask) then
               mask(o)
            Else
               If Protect(o,blur) then
                  blur(o)
               Else
                  If Protect(o,pixelizate) then
                     pixelizate(o)
                  Else
                     If Protect(o,zoom_in(i)) then
                        minzf ←min(i,minzf)
Return(applyzf(minzf,map))
```

This algorithm works in the following two cases:

- Mechanisms have different priorities and the following mechanism hierarchy is used (from the lowest priority to the highest priority): *{zoom-in, pixelizate, blur, mask, paste, hide, reject_query}*. The algorithm is designed to select the highest priority mechanism in case more than one mechanism can be derived from a single prohibition.
- Priorities are assigned to protection rules (with the default rule having the lowest priority). The algorithm selects the mechanism derived from the highest priority rule in case more than one mechanism can be derived from a single prohibition. However, it might happen that several mechanisms are selected. This can be the case if some protection rules have the same priority. If this occurs, then the algorithm selects the mechanism to enforce on the basis of the mechanisms hierarchy.

Regarding this algorithm we can make the following comments:

- If, for any object, mechanism, *reject_query* should be enforced then the algorithm terminates (line 5) and an empty map is returned, even if the query addressed some authorized objects.
- Prohibited objects that should be removed from the final map are ignored (line 7).
- Line 8 inserts authorized objects. It also inserts prohibited objects on which a protection mechanism should be applied.
- The returned map is displayed at the lowest zoom-in factor imposed by protection rules (line 22). For example, let the zoom-in-factor of the query be equal to 5. Assume there are two objects addressed by the query which are protected and should be displayed respectively at zoom-in factor equal to 4 and zoom-in factor equal to 3. The lowest zoom-in factor imposed by protection rules is selected and the map is displayed at zoom-in factor equal to 3.
- Lines 10, 13, 16 and 19 apply various protection methods.
- Line 23 applies the zoom-in factor and returns the final map.
- This algorithm is linear with the number of objects addressed by the query.

Of course this algorithm could be written differently. We could consider another mechanism hierarchy or we could consider a partial order on the set of mechanisms. In this latter case, if two mechanisms which cannot be compared can be derived from the same prohibition then priorities on rules should be used to select one of these mechanisms.

5 Application Example

5.1 Contextual Security Policy

We consider an organization simultaneously managing a fleet of taxis and a fleet of ambulances. While driving, drivers from this company use a spatial application displaying surrounding objects. Fig 3 shows that subjects are drivers who can be either taxi drivers or ambulance drivers. Objects are buildings, roads and military areas (including military hospitals). Basically, the security policy expresses the fact that drivers can view spatial data which are within a radius of 40 km around their position. However, there are some restrictions to this general rule.

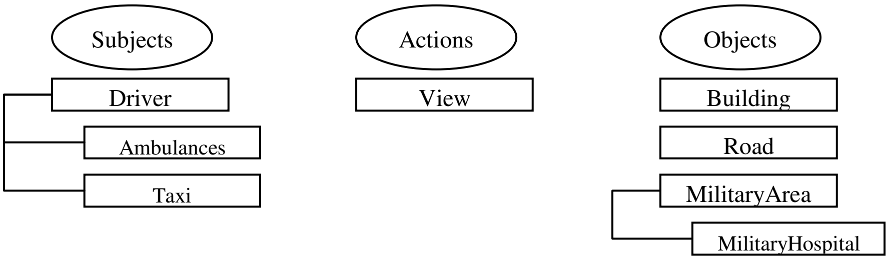


Fig. 3. Synopsis of our example

Default Policy: The default policy is closed i.e. given a subject *s* and an object *o*, if *Permit(s,o)* cannot be derived from the security policy then *Deny(s,o)* should be derived.

Drivers have the permission to view at a maximum zoom-in factor of 10 any object that is within a radius of 40km around their position.

$$\forall s \forall o \left(Driver(s) \wedge distance(s,o) \leq 40 \wedge zf(o) \leq 10 \rightarrow Permit(s,o) \right) \tag{23}$$

Drivers have the permission to view roads at a maximum zoom-in factor of 10 (even those which are not within a radius of 40km).

$$\forall s \forall o \left(Driver(s) \wedge Road(o) \wedge zf(o) \leq 10 \rightarrow Permit(s,o) \right) \tag{24}$$

Taxis driving at a speed greater than 100km per hour are forbidden to view any object. This rule does not apply to ambulances since they are emergency vehicles.

$$\forall s \forall o \left(Taxi(s) \wedge speed(s) \geq 100 \rightarrow Deny(s,o) \right) \tag{25}$$

Drivers are prohibited to view military areas .

$$\forall s \forall o \left(Driver(s) \wedge MilitaryArea(o) \rightarrow Deny(s,o) \right) \tag{26}$$

Drivers are prohibited to view buildings which are contiguous to military areas

$$\forall s \forall o \forall m \left(Driver(s) \wedge Building(o) \wedge MilitaryArea(m) \wedge Touches(o,m) \rightarrow Deny(s,o) \right) \tag{27}$$

Ambulances are permitted to view military hospitals at a maximum zoom-in factor of 5.

$$\forall s \forall o \left(Ambulance(s) \wedge MilitaryHospital(o) \wedge zf(o) \leq 5 \rightarrow Permit(s,o) \right) \tag{28}$$

The above security policy may lead to conflicts. Rule 23 and rule 24 conflict with the default policy. For a taxi driving at more than 100 km per hour, rule 25 conflicts with rules 23 and 24. For military areas, rule 26 conflicts with rule 23. For buildings contiguous to military areas, rule 27 conflicts with rule 23. For ambulances, military hospitals and a zoom-in factor lower than 5, rule 28 conflicts with rule 26. As we said before, it is not the purpose of this paper to discuss conflict resolution. In this example, we simply assume that rules 23 and 24 override the default policy, rule 25 overrides rules 23 and 24, rule 26 overrides rule 23, rule 27 overrides rule 23 and rule 28 overrides rule 26.

5.2 Protection Rules

Default Mechanism: We define the default mechanism as *hide*:

$$\forall s \forall o \big(Deny(s,o) \rightarrow Protect(o, hide)\big) \tag{29}$$

Rule 30 says that if a taxi driving at 100km is forbidden to view an object then his query should be rejected

$$\forall s \forall o \big(Taxi(s) \wedge speed(s) \geq 100 \wedge Deny(s,o) \rightarrow Protect(o, reject\_query)\big) \tag{30}$$

Rule 31 says that if a subject is forbidden to view a building within a radius of 40km then the resolution of this building should be lowered.

$$\forall s \forall o \big(Building(o) \wedge distance(s,o) \leq 40 \wedge Deny(s,o) \rightarrow Protect(o, pixelizate)\big) \tag{31}$$

Rule 32 says that if a subject is forbidden to view a military area within a radius of 40km then this military area should be masked.

$$\forall s \forall o \big(MilitaryArea(o) \wedge distance(s,o) \leq 40 \wedge Deny(s,o) \rightarrow Protect(o, mask)\big) \tag{32}$$

Rule 33 says that if an ambulance is forbidden to view a military hospital within a radius of 40km then the zoom-in factor should be lowered to 5.

$$\forall s \forall o \begin{pmatrix} MilitaryHospital(o) \wedge Ambulance(s) \wedge distance(s,o) \leq 40 \wedge Deny(s,o) \\ \rightarrow \text{Protect}(o, zoom\_in(5)) \end{pmatrix} \tag{33}$$

We assign priorities to protection rules. The default rule 29 has the lowest priority. Rule 30 has the highest priority. Rules 31 and 32 have the same priority. Rule 33 has a higher priority than rule 32.

The default mechanism applies whenever it is not possible to derive any mechanism for a given prohibition. Therefore we can easily see that the default mechanism (rule 29) applies to instances of the *Deny* predicate which are derived from the default (closed) policy. These instances address objects which are not the roads and which are outside of a 40km radius. Rule 30 applies to instances of the *Deny* predicate which are derived from rule 25. Taxis driving too fast should see their query rejected i.e. taxis are in fact forbidden to use the application as long as they drive too fast. Rule 31 applies to the instances of the *Deny* predicate which are derived from rule 27, i.e. buildings touching military areas should be pixelizated. Rule 33 applies to *some* instances of the *Deny* predicate which are derived from rule 26. These instances address ambulances requesting to view military hospital at a zoom-in factor greater than 5 (recall that rule 28 say that ambulances are permitted to view military hospitals at a zoom-in factor lower than 5). Rule 32 applies to all the other instances of the *Deny* predicate which are derived from rule 26.

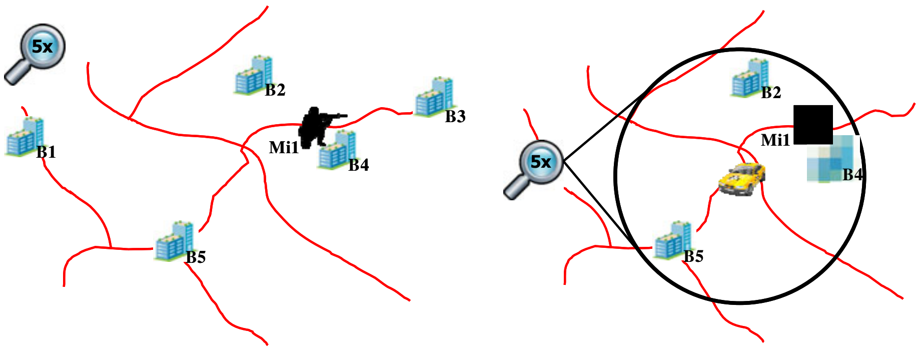


Fig. 4. Original map and Taxi driver view

The left picture of figure 4 shows an example of an original road map. The right picture shows the taxi driver view of the same map. The zoom-in factor is 5, the circle represents the 40km radius, objects outside this radius are hidden (except roads), the military area is masked and the building near the military area is pixelizated. Note that under the assumption that the military area is not a military hospital, the ambulance view is the same as the taxi view.

6 Sketch of Implementation

In this section, we sketch the implementation of our model within the framework of the OpenGIS® Styled Layer Descriptor (SLD) [7] Profile of the OpenGIS® Web Map Service (WMS) [8] specification.

6.1 The OpenGIS® Web Map Service Specification

Among all the specifications published by the OGC [14] regarding Geographic Databases and data exchange protocols, the most popular and widely used is undoubtedly the OpenGIS® Web Map Service (WMS). WMS servers support the creation and display of registered and superimposed map-like views of information coming from multiple heterogeneous sources including other WMS servers. The underlying protocol for WMS is the Hypertext Transfer Protocol (HTTP). Most WMS servers implement a common gateway interface (cgi-bin) and may be requested via an

URL issued from a standard web-browser or any WMS-enabled software. The main parameters of a basic *GetMap* request to a WMS server include an ordered (bottom to top) list of layers (spatial objects), an ordered list of styles in which each layer is to be rendered (with a one-to-one correspondence with the list of layers), a Bounding Box specifying the geographical extent of the region to map and two parameters (*width* and *height*) specifying the final size of the requested image. The response of the server to a valid WMS *GetMap* request is an image file in the specified format (MIME type such as PNG, GIF or JPEG) having the dimension *width* by *height* pixels. Two points are fundamental in WMS requests: (i) conjunction of the requested image size (*width* and *height* parameters) with the *in situ* geographical extent (Bounding Box parameter) leads to the definition of the zoom-in factor for the final map and (ii) applying a specific style to a specific layer (geographical object) leads to the concept of Styled Layers developed in the next subsection.

6.2 The OpenGIS® Styled Layer Descriptor Profile

A styled layer represents a particular combination of a 'layer' and a 'style' in which that layer can be symbolized. Conceptually, the layer defines a stream of features and the style defines how those features are symbolized. Defined by OGC in 2007, the Styled Layer Descriptor (SLD) is an XML-based description format for formatting data from a WMS flow. It plays the same role as a CSS file to an HTML page, the goal being to completely separate the style from the data. For example the same geographic object of type point may be symbolized as a small blue dot, a large red cross or a medium green square. A polygonal object may be drawn as a light blue transparent hatch, a fully opaque black polygon as well as a green grass-looking textured object. Named-styles are predefined using SLD files and used within the *Styles* parameter of the WMS request.

6.3 Rewriting WMS Queries

Our prototype acts as a front-end engine rewriting WMS queries issued from an authenticated user on the basis of the outcome produced by the PEP algorithm presented in section 4. User queries are rewritten as follows:

- Each object that should be hidden (line 7 of the PEP algorithm) is simply removed from the list of requested layers.
- Each object *o* that should be overlaid by another object *g* (line 11 of the algorithm) is replaced by object *g* in the list of requested layers.
- Reducing the zoom-in factor (line 23 of the algorithm) is achieved by modifying the width and height of the final image so that the ratio between the bounding box and the size of the image respects the zoom-in factor imposed by the PEP algorithm.
- We wrote three SLD *protection styles* simulating respectively the three protection mechanisms *blur*, *mask* and *pixelizate* (line 14, 17 and 20 of the PEP algorithm). For each object that should be blurred, masked or pixelizated, the corresponding protection style replaces the style of the original query.

Let us assume for example that the following query is issued by a taxi driver. It requests all layers (objects) with the default style for each layer (see Figure 4).

```
http://yourWmsServer.com/wms?SERVICE=wms&VERSION=1.3.0&REQUEST=G
etMap&LAYERS=Roads,B1,B2,B3,B4,B5,Mi1&STYLES=&BBOX=x1,y1,x2,y2&W
IDTH=600&HEIGHT=600&FORMAT=image/png&SRS=epsg:4326
```

This query would be rewritten as follows by our front-end engine:

```
http://yourWmsServer.com/wms?SERVICE=wms&VERSION=1.3.0&REQUEST=G
etMap&LAYERS=Roads,B2,B4,B5,Mi1&STYLES=,,PixelSLD,,MaskSLD&BBOX=
x1,y1,x2,y2&WIDTH=600&HEIGHT=600&FORMAT=image/png&SRS=epsg:4326
```

The rewritten query addresses buildings within a radius of 40km, requests the military area Mi1 with the mask style *MaskSLD*, requests the building *B4* with the pixelizate style *PixelSLD* and requests other objects with the server default style.

Currently, our prototype (http://pages.upf.pf/Patrick.Capolsini/rech/protect/index.htm) is only at the proof-of-concept stage. It implements the *mask* and *pizelizate* SLD and uses some predefined examples. Our prototype considers a set of prohibited objects. First, it asks the user to set some context parameters. Second it shows how the original user WMS query is rewritten into a secure WMS query. Third, it displays the map returned by the map engine. In a near future, we will implement it as a secure proxy for publishing geo-data.

7 Related Work

Several access control models and approaches have been proposed for geo-spatial resources. Some of them such as the Geospatio-temporal Authorization Model (GSAM) focus on the visualization of raster geo-spatial data like multi-resolution satellite imagery (see [15], [16], [17] and [3] for details) while others like Geo-RBAC [2, 18] may be described as Location Based Systems. On our side, we proposed an extension to the generic Or-BAC model to derive a geospatial context aware access control system ([4] and [5]). Regarding security standards, the Open Geospatial Consortium (OGC) [14] published the Digital Rights Management Reference Model (GeoDRM RM) [19] which is a reference model for digital rights management functionality for geospatial resources and geo-XACML [20] which extends the OASIS XACML [21] language for expressing authorization policies. The interested reader can refer to [22] for a summary of the current state of the art in the field of geo-spatial databases security.

As we already mentioned, to our knowledge it is the first time that a security model includes a framework for specifying protection mechanisms to be enforced. Most of the existing works on geo-data security focus on the expressive power of the security policy and on conflict resolution between permissions and prohibitions. This is the case in [23] where the authors, in the context of an XML-based Framework, propose to use Scalable Vector Graphics (SVG) [24] to represent geo-spatial objects and layers. They then define an access control model where an authorizations rule involves a subject, an object and an action as well as a Level of Details factor and an operative region. The SVG representation of the map and R-tree based indexes are used in the policy enforcement algorithm to determine which geo-spatial objects are addressed by the request and whether they can be accessed or not. In [25], authors assume that all spatial data are stored in a spatial database accessed by a Geographic Information System (GIS). A security object may be a spatial component, a set of spatial components or indirectly a query result. The algorithm which analyzes access requests includes a step of potential conflicts detection between security rules involving geo-spatial objects which can touch, intersect or be contained in each other. The authors distinguish between two potential cases of conflict depending on whether an object is totally or partially included in another. In [26], the author makes the distinction between object-based restrictions (on a particular object), class-based restrictions (on all objects of type "Building" or type "Road" for example) or spatial access restrictions (based on the geometry of objects). Objects are encoded using the Geographic MarkUp Language (GML) [27]. Security rules are expressed using XACML and geoXACML and may contain a spatial condition. Evaluation of the security policy may result in either "Permit", "Deny", "N/A" or "indeterminate". The paper focuses on the "approximate" detection of contrary spatial permissions i.e. one spatial rule evaluates to permission while another one evaluates to prohibition. For this "approximate" detection, no actual request is required. The author states that a complex access control system has to ensure appropriate and error-free enforcement of declared permissions. He suggests using a permission repository and testing it for the a priori detection of inconsistent spatial authorization rules.

8 Conclusion

In this paper we focused on how to enforce the security policy in the framework of a Map Service supporting the creation and display of map-like views of information. We proposed a rule-based PEP which selects the protection mechanism to enforce whenever a prohibition is derived from the security policy. We suggested seven protection mechanisms, namely: {*zoom-in, pixelizate, blur, mask, paste, hide, reject_query*}. We defined the logical framework to express some protection rules. These rules specify mechanisms to enforce whenever prohibitions are derived from the security policy. If, given a prohibition, several mechanisms could be used then only one of them should be selected according to priorities which are either assigned to the protection mechanisms themselves or to the protection rules. We defined the PEP algorithm which enforces the mechanisms and builds the map to display. We presented an example to illustrate how our proposal could be used and useful in a real application. We sketched the implementation of our proposal within the framework of the SLD profile of the WMS encoding standard and finally, we implemented a prototype showing the feasibility of our approach. Finally, let us also mention the following point: geographic data are a special case of multimedia data. Therefore, we also proposed a version of our model for the more generic case of multimedia data [28]. We used our model to protect images published in a social network.

Acknowledgment. This work was conducted as part of the ANR funded project under reference ANR-SESUR-2007-FLUOR.

References

1. WikiPedia. Satellite map images with missing or unclear data (2011), `http://en.wikipedia.org/wiki/Satellite_map_images_with_missing_or_unclear_data`
2. Bertino, E., Catania, B., Damiani, M.L., Perlasca, P.: GEO-RBAC : A spatially Aware RBAC. In: ACM Symposium on Access Control Models and Technologies (SACMAT 2005), Stockholm, Sweeden, pp. 29–37 (2005)
3. Atluri, V., Chun, S.A.: A geotemporal role-based authorization system. International Journal of Information and Computer Security 1, 143–168 (2007)
4. Gabillon, A., Capolsini, P.: Dynamic Security Rules for Geo Data. In: Garcia-Alfaro, J., Navarro-Arribas, G., Cuppens-Boulahia, N., Roudier, Y. (eds.) DPM 2009. LNCS, vol. 5939, pp. 136–152. Springer, Heidelberg (2010)
5. Capolsini, P., Gabillon, A.: Security policies for the Visualization of Geo Data. In: ACM SIGSPATIAL GIS 2009 International Workshop on Security and Privacy in GIS and LBS (SPRINGL 2009), pp. 2–11. ACM, Seattle (2009)
6. Gabillon, A., Capolsini, P.: Rule-based Policy Enforcement Point for Map Services. In: ACM SIGSPATIAL GIS 2010 International Workshop on Security and Privacy in GIS and LBS (SPRINGL 2010), pp. 12–17. ACM, San Jose (2010)
7. Lupp, M.: Styled Layer Descriptor profile of the Web Map Service Implementation Specification. Open Geospatial Consortium Inc. OGC(R) 05-078r4 (2007)
8. Beaujardiere, J.d.l.: OpenGIS(R) Web Map Server Implementation Specification. Open Geospatial Consortium Inc. OGC(R) 06-042 (2006)
9. El-Kalam, A., El-Baida, R., Balbiani, P., Benferhat, S., Cuppens, F., Deswarte, Y., Miège, A., Saurel, C., Trouessin, G.: Organization Based Access Control. In: 4th IEEE International Workshop on Policies for Distributed Systems and Networks (Policy 2003). IEEE, Como (2003)
10. Yuan, E., Tong, J.: Attributed Based Access Control (ABAC) for Web Services. In: Proceedings of the IEEE International Conference on Web Services (ICWS 2005), Orlando, Florida - USA (2005)
11. Janée, G., Frew, J., Hill, L.L.: Issues in Geo-referenced Digital Libraries. D-Lib Magazine 10 (2004)
12. Rigaux, P., Scholl, M., Voisard, A.: Spatial Databases with application to GIS. Elsevier (2002)
13. Herring, J.R.: OpenGIS(R) Implementation Specification for Geographic information - Simple feature access - Part 1 : Common architecture. Open Geospatial Consortium Inc. OGC(R) 06-103r3 (2006)
14. [OGC2008] OGC. Open Geospatial Consortium Inc. - About Us (2008), `http://www.opengeospatial.org/about`
15. Chun, S.A., Atluri, V.: Protecting privacy from continuous high-resolution satellite surveillance. In: Proceedings of the 14th IFIP 11.3 Annual Working Conference on Database Security, Schoorl, The Netherlands, pp. 233–244 (2000)
16. Atluri, V., Mazzoleni, P.: A uniform indexing scheme for geo-spatial data and authorizations. In: Proceedings of the 16th IFIP WG 11.3 Conference on Data and Application Security (2002)

17. Atluri, V., Chun, S.A.: An authorization Model for Geospatial Data. IEEE Transactions on Dependable and Secure Computing 1, 238–254 (2004)
18. Damiani, M.L., Bertino, E., Catania, B., Perlasca, P.: GEO-RBAC: A spatially Aware RBAC. ACM Transactions on Information Systems and Security, 1–34 (2006)
19. Volwes, G.: Geospatial Digital Rights Management Reference Model (GeoDRM RM). Open Geospatial Consortium Inc. OGC(R) 06-004r3 (2006)
20. Matheus, A., Herrmann, J.: Geospatial eXtensible Access Control Markup Language (GeoXACML). Open Geospatial Consortium Inc. OGC(R) 07-026r2 (2008)
21. [XACML22005] OASIS. eXtensible Access Control Markup Language (XACML) Version 2.0 (2005), `http://www.oasis-open.org`
22. Chun, S.A., Atluri, V.: Geospatial Database Security. In: Gertz, M., Jajodia, S. (eds.) Handbook of Database Security Applications and Trends, pp. 247–266. Springer US (2008)
23. Purevjii, B.-O., Amagasa, T., Imai, S., Kanamori, Y.: An access control model for geographic data in an XML-based framework. In: 2nd International Workshop on Security in Information Systems (WOSIS 2004), Porto, Portugal, pp. 251–260 (2004)
24. [SVG2010] W3C. Scalable Vector Graphics (SVG) 1.1, 2nd edn. (2010), `http://www.w3.org/Graphics/SVG/`
25. Sasaoka, L.K., Medeiros, C.B.: Access Control in Geographic Databases. In: Roddick, J., Benjamins, V.R., Si-said Cherfi, S., Chiang, R., Claramunt, C., Elmasri, R.A., Grandi, F., Han, H., Hepp, M., Lytras, M.D., Mišić, V.B., Poels, G., Song, I.-Y., Trujillo, J., Vangenot, C. (eds.) ER Workshops 2006. LNCS, vol. 4231, pp. 110–119. Springer, Heidelberg (2006)
26. Matheus, A.: Declaration and enforcement of fine-grained access restrictions for a service-based geospatial data infrastructure. In: 10th ACM Symposium on Access Control Models and Technologies (SACMAT 2005), Stockholm, Sweden, pp. 21–28 (2005)
27. Portele, C.: OpenGIS(R) Geography Markup Language (GML) Encoding Standard. Open Geospatial Consortium Inc. OGC(R) 07-036 (2007)
28. al Bouna, B., Chbeir, R., Gabillon, A.: The Image Protector - A Flexible Security Rule Specification Toolkit. In: SECRYPT 2011: Proceedings of the International Conference on Security and Cryptography, Seville, Spain, July 18-21, pp. 345–350 (2011)

Integrating Trend Clusters for Spatio-temporal Interpolation of Missing Sensor Data

Anna Ciampi, Annalisa Appice, Pietro Guccione, and Donato Malerba

Dipartimento di Informatica, Università degli Studi di Bari Aldo Moro
via Orabona, 4 - 70126 Bari, Italy
{aciampi,appice,malerba}@di.uniba.it

Abstract. Information acquisition in a pervasive sensor network is often affected by faults due to power outage at nodes, wrong time synchronizations, interference, network transmission failures, sensor hardware issues or excessive energy consumption for communications. These issues impose a trade-off between the precision of the measurements and the costs of communication and processing which are directly proportional to the number of sensors and/or transmissions. We present a spatio-temporal interpolation technique which allows an accurate estimation of sensor network missing data by computing the inverse distance weighting of the *trend cluster* representation of the transmitted data. The trend-cluster interpolation has been evaluated in a real climate sensor network in order to prove the efficacy of our solution in reducing the amount of transmissions by guaranteeing accurate estimation of missing data.

1 Introduction

Nowadays wireless sensor networks have become emerging tools for the real-time monitoring of geo-spatial phenomena (e.g. climate data). A sensor is characterized by computing and storage abilities and its longevity depends on the smart use of energy. The traditional uncertainty of the application environments, the scarcity of communication ability and of the transmission bandwidth, suggest to adopt a meaningful subset of the whole network and estimate the un-sampled values from the available measures as accurately as possible. In this scenario, high densities of sensor transmissions are desirable to achieve high resolution and accurate estimates of the environmental conditions. On the other hand, high densities place heavy demands on bandwidth and energy consumption for communication [9]. These considerations impose a trade-off between the precision of the measurements and the costs of communication and processing, which are potentially proportional to both the number of sensors and frequency of transmissions. Following this idea, the paper proposes a sensor network management framework, called TRECI (TREend Cluster based Interpolate), which cuts down on both sensor communication costs and energy consumption while keeping a reasonable degree of accuracy. This goal is reached by reducing, someway, the number of sensor transmissions in the network and by allowing the interpolation of the unknown measures using a *trend cluster* representation of the stream.

A trend cluster is a spatial cluster of sensors whose transmitted values show a similar trend (represented by a time series) in a time window. In our seminal work [1], trend clusters are discovered online from data as they arrive from the network and stored in database as a compact representation (summary) of the stream. Then, given a sensor and a time point, the past measure of the sensor is naively approximated by selecting the appropriate value timestamped with the time point in the trend of the cluster grouping the sensor itself. In this paper, we extend this interpolate ability in both the space and the time direction. TRECI, by using the trend clusters, allows us to accurately interpolate data at any spatial location of the networked area, and not only at the locations where sensors are installed. Similarly, it allows us to interpolate data at any time point of the past, and not only at the time points when the network transmitted. We have defined a sampling technique to extract the key sensors of a cluster, according to the cluster shape, to spatially locate any unknown data and a polynomial to derive an estimate of a trend value at any time point. The Inverse Distance Weighting (IDW) interpolation scheme is considered to estimate the value at the space-point location by using a weighted average of the nearby scatter points. It is noteworthy that the sampling and the polynomial allow us to derive a highly compact representation of each trend cluster that, if stored in database, contributes to pinpoint the goal of stream summarization. At the same time they provide the natural ingredients for the ubiquitous interpolation.

In short the innovative contributions of this paper are those highlighted in the following.

1. A sample of centroids is now used to compactly describe the shape of a cluster; each centroid is the center of a *dense* region of the cluster and it is associated to the polynomial representation of the trend of the cluster;
2. A polynomial function is learned from the time series representation of the trend; it provides a *functional model* of the trend that is computable at any possible time point in the associated time horizon;
3. The *inverse distance weighting* interpolation allows, for each location (x, y) and time point t, the retrieval of the centroids which are close to (x, y) in a time horizon comprising t, and and to provide the output by means of a weighted weighted average of the scatter space-time estimated points.

The paper is organized as follows. In Section 2, we describe trend cluster representation of a stream. In Section 3, we provide the formulation of the interpolation task and its perspective in the context of trend cluster discovery. In Section 4, the spatio-temporal interpolate computed by TRECI is described. Results on a real dataset and discussion are reported in Section 5 and then conclusions follow.

2 Trend Cluster: Background and Discussions

A trend cluster is defined in [1] as a summarization pattern for a numeric quantity which is periodically measured across a sensor network. It is a *cluster* of spatially close sensors which transmit numeric data whose temporal variation, called *trend*, is similar across a time horizon. Formally, a trend cluster is a triple:

$$[\ W, C, T\ ], \tag{1}$$

where:

1. W is a *time horizon* along which network transmissions were collected;
2. C is a *spatial cluster* of spatially close sensors which transmitted values whose temporal variation was similar along the time horizon of the window;
3. T is the time series which represents the trend of the clustered measures as they were collected at the transmission time points comprised in W.

Trend clusters are discovered in real time by the system SUMATRA [1], and the representation of a trend can be compressed by applying some signal processing techniques, such as DFT and Haar Wavelets [2] before its storage in the database. SUMATRA operates under some basic assumptions which are largely supported by the most of the real world sensor network applications. The time domain is considered absolute, linear and discrete and it is segmented in count based windows; a window comprises w equally spaced consecutive time points. Each window is the time horizon unit of the trend cluster discovery process which is defined according to the domain expert knowledge. The spatial closeness relation between sensors is computed according to the nearby relation which is implicitly defined by the (latitude-longitude) coordinates of each sensor in the network. The trend is the series of equally spaced timestamped prototypes computed as a median of the clustered sensed values at each w time point of the window. The similarity of the temporal variation between transmissions of clustered sensors depends on a user-defined parameter called trend similarity threshold and denoted by δ. In practice, each sensor that is grouped in a cluster transmits a series of values along the window horizon which differs at worst δ from the trend prototype of the cluster. Trend differences are computed time point-by-time point by means of the Euclidean distance.

Experiments reported in [1] show that the trend clusters can be discovered by SUMATRA in real time, they accurately represent the stream and are significantly more compact than the original data stream. Additionally, SUMATRA integrates the inverse transform to accurately estimate the past stream from the trend prototype associated to each sensor in each cluster, with a given degree of accuracy. The point is that SUMATRA does not include the purpose of interpolate (i.e. estimate) data anywhere (also where no sensor is installed) and anytime (also when no transmission is done).

3 Trend Cluster Based Interpolation: The Task

In a spatio-temporal streaming environment, the interpolation functionality should allow the estimation of (missing) data at any location of the space and at any time point of the past. Formally, the interpolation task we address in this paper can be formulated as follows. Given:

1. The geographic location of the sensors of a network;
2. A trend cluster representation of the data transmitted from this network;
3. A 2D location (x, y) and a past time point t.

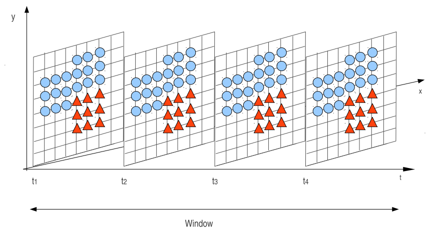
Find an accurate estimate of the unknown (virtually streamed) data at the spatial location (x, y) and at the time point t. This formulation implies that the streamed data are discarded and, then, we have to estimate the unknown data at (x, y, t) by using only the trend cluster representation. To this aim we have to search for the spatio-temporal *neighborhood* of (x, y, t) in the trend cluster representation. This neighborhood is naively defined as that having the time horizon of the trend window hosting t, and the space arrangement of the clusters enclosing (x, y). The computation of the interpolated data on such a neighborhood poses at least two issues:

1. By definition, a spatial cluster is discretely represented as a finite set of 2D locations; (x, y) may not coincide with any clustered location, hence we need a mechanism to determine which clusters enclose (x, y) and to identify the key points of this surrounding for the weighted computation.
2. By definition, a trend is discretely represented by a time series; t may not coincide with any timestamped value of this series, hence we need a mechanism to estimate trend values at each continuously defined time point t.

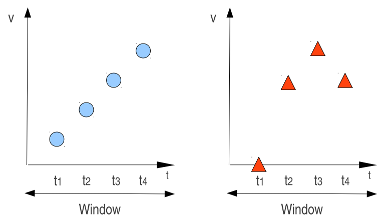
To deal with both these issues we propose to change the way clusters and trends discovered by SUMATRA are stored in the database.

For the clusters, we need a method to select the clusters which enclose (x, y) and then reasonably contribute to the interpolation. To this aim, we assume that a spatial cluster encloses (x, y) if it groups sensors whose locations are close to (x, y). Therefore, we have to compute the distance between (x, y) and the location of each sensor falling in that cluster. In case of a large network, computing all distances may be too much costly. The naive solution to reduce the computation charge is the choice of a single sensor (centroid) as representative of the cluster [5]. But, for irregularly shaped clusters or concentric clusters, one centroid can be a badly biased representation of the cluster shape. According to us, a sampling procedure tailored to identify the sensors which preserve the information on the cluster shape should be a more reasonable choice to save both the computation charge and the interpolation accuracy. On the other hand, it is a fact that the storage in database of a sample has the free-of-charge benefit of reducing the memory space requested for the cluster storage. Based on these considerations, we propose a shape-dependent way to sample sensors of each cluster and we store the samples in database (see Figure 1(a) and Figure 1(c)).

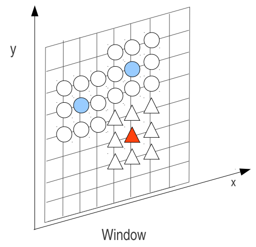
For the trend, we need a way to estimate a trend value at any generic time point in the window. To this aim, we opt for looking for a polynomial fitting of the time series representing the trend. The coefficients of this polynomial will be stored in database in place of the values of the series itself (see Figure 1(b) and Figure 1(d)). It is noteworthy, that by guaranteeing that the degree of learned polynomial is always less than w, storing the polynomial coefficients does not occupy more memory than storing the time series.



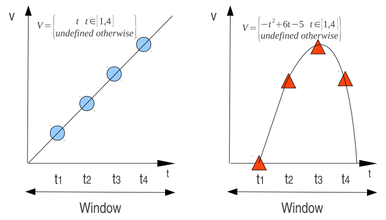
(a) Clusters (SUMATRA)



(b) Trends (SUMATRA)



(c) Clusters (sampling)



(d) Trends (polynomial)

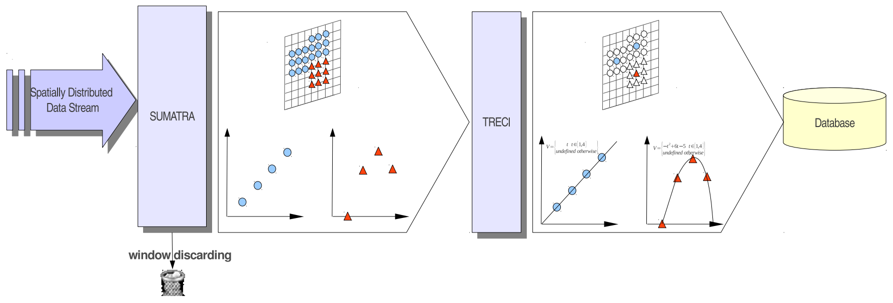
Fig. 1. Samples of sensors (Fig. 1(c)) which are extracted from spatial clusters (Fig. 1(a)). These samples maintain someway the information on the cluster shape. Representation of the polylines (Fig. 1(d)) as fit of the time series (Fig. 1(b)) representing the trends of the associated clusters.

The use of the sampling and the polynomial in the trend cluster storage provides several advantages. First, further space is saved in the task of summarization as we store only a sample of the clustered sensors and few coefficients to represent the associated trend. Second, we now are able to interpolate values ubiquitously in space and in time as the trend values estimated for any t can be averaged for the sampled sensors computed as the realistic surroundings of any (x, y). Finally, as the weights of the average are properly-defined functions of the distance between (x, y) and each known cluster representative in the surroundings, we correctly account for the correlation existing between two or more clusters [7].

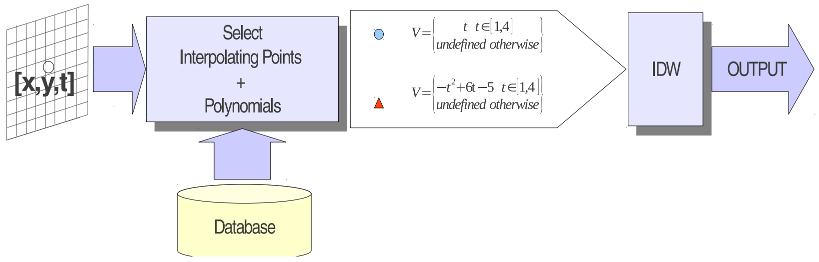
4 Trend Cluster Based Interpolation: The Algorithm

The interpolation framework operates in two steps, an online steps performed as the stream flows and an offline step which can be performed at any time. In the followings Sub-Sections the two steps are detailed.

Online Step. In an on-line step (see Figure 2(a)), TRECI integrates the SUMATRA system for the trend cluster discovery. The discovery process is performed



(a) Treci ONLINE


[x,y,t]
Select Interpolating Points + Polynomials
$V=\begin{cases} t & t\in[1,4] \\ \text{undefined otherwise} \end{cases}$
$V=\begin{cases} -t^2+6t-5 & t\in[1,4] \\ \text{undefined otherwise} \end{cases}$
IDW
OUTPUT
Database

(b) Treci OFFLINE

Fig. 2. Treci:on-line summarization step and off-line interpolation step

window by window. For each trend cluster discovered by SUMATRA along the time horizon of a window, the key sensors of the cluster are determined and the polynomial which fits the time series of the trend is estimated. The sample sensors and the polynomial coefficients are stored in the database as a representation of the trend cluster, while the streamed window data are discarded.

Offline Step. In an off-line step (see Figure 2(b)), for any unsampled space-time point (x, y, t), the trend clusters associated to the window containing t are retrieved from the database. The polynomial associated to the clusters closest to (x, y) are selected and used to provide an estimate at time t. Several estimates are combined in a single one by the Inverse Distance Weighting (IDW) interpolation. Details on spatial sampling, polynomial learning and IDW are provided in the next Sub-Sections.

4.1 Quadtree Sampling

Let C be a cluster of sensors. The goal is to find a shape based sample S of C such that S is stored in the database in place of C and it is representative of C for the

Algorithm 1. Function sampling(C) **return** S

```
Require: $C$ {set of sensors}
Ensure: $S$ {sample of sensors extracted from $C$}
 1: $S \Leftarrow \oslash$
 2: $Q \Leftarrow \mathrm{mbr}(C)$
 3: if density$(C,Q) \neq 0\%$ then
 4:    if density$(C,Q) > 75\%$ then
 5:       $S \Leftarrow S \cup \{\mathrm{centroid}(C)\}$
 6:    else
 7:       $List < C > \Leftarrow$subClusterQuadtree$(C)$
 8:       for all $C_i \in List < C >$ do
 9:          $S \Leftarrow S \cup \mathrm{sampling}(C_i)$
10:       end for
11:    end if
12: end if
```

purpose of interpolation. The random sampling is the simplest way to address this task, but it poses two issues. How can we choose the number of sensors to be sampled? How can we guarantee that the randomly selected sensors maintain the information on the cluster shape? To answer both questions we formulate a sampling algorithm which resorts to a *quadtree* decomposition of the clustered area. The quadtree decomposition is an adaptive sampling method largely used in image processing [10,8] that we tailor here to identify the key sensors of the cluster which are centroids in the densely populated subareas of the cluster itself. Thus, the number of sampled sensors and their location depend on how the cluster shape is spread across the space.

The sampling of a cluster is performed according to Algorithm 1. First, the minimum boundary rectangle Q of the cluster C is computed (Algorithm 1, line 2). The minimum bounding rectangle, also known as minimum bounding box, is the rectangle enveloping C that is unambiguously identified by its left inferior vertex (min(x), min(y)) and right superior vertex (max(x), max(y)) with:

$$\begin{array}{ll} min(x) = \min_x \ \{x|(x,y) \in C\} & min(y) = \min_y \ \{y|(x,y) \in C\} \\ max(x) = \max_x \{x|(x,y) \in C\} & max(y) = \max_y \{y|(x,y) \in C\} \end{array} \tag{2}$$

Then, the density of the cluster C inside Q (Algorithm 1, lines 3-4) is computed according to density measure defined as follows:

$$density(C,Q) = \frac{\sharp(C \cap Q)}{\sharp Q} \times 100 \tag{3}$$

where $\sharp(C \cap Q)$ denotes the number of sensors clustered in C which are spatially contained in Q, $\sharp Q$ is the number of sensors of the network falling in Q. The spatial containment relation between a 2D location (x,y) and a rectangle $[(x_i,y_i),(x_s,y_s)]$ is defined as follows:

$$(x,y) \subseteq Q \ \Leftrightarrow \ x_i \leq x \leq x_s \ \wedge \ y_i \leq y \leq y_s \tag{4}$$

If $density(C, Q)$ is equal to zero, then Q is empty and it can be discarded for the sampling. If $density(C, Q)$ is greater than 75%, than $C \cap Q$ can be considered a dense sub-area of C and its centroid node is sampled (Algorithm 1, lines 4-5). Otherwise Q is decomposed in four sub quadrants ($Q1$, $Q2$, $Q3$, $Q4$), and then C is coherently decomposed in the four subclusters falling in the those quadrants ($C1 = C \cap Q1$, $C2 = C \cap Q2$, $C3 = C \cap Q3$ and $C4 = C \cap Q4$). The sampling is then recursively applied to each subcluster C_i (Algorithm 1, lines 8-10).

The quadrant decomposition of Q is defined orthogonally to the axes according to $x = \frac{max(x)+min(x)}{2}$ and $y = \frac{max(y)+min(y)}{2}$ such that:

$$\begin{array}{llll}
Q1 : [& \left(min(x), \frac{max(y)+min(y)}{2}\right) & , \left(\frac{max(x)+min(x)}{2}, max(y)\right) &] \\
Q2 : [& \left(\frac{max(x)+min(x)}{2}, \frac{max(y)+min(y)}{2}\right) & , (max(x), max(y)) &] \\
Q3 : [& \left(\frac{max(x)+min(x)}{2}, min(y)\right) & , \left(max(x), \frac{max(y)+min(y)}{2}\right) &] \\
Q4 : [& (min(x), min(y)) & , \left(\frac{max(x)+min(x)}{2}, \frac{max(y)+min(y)}{2}\right) &]
\end{array} \tag{5}$$

The centroid of a set of sensors C is computed as follows. First, the centroid location $(\widehat{x}_C, \widehat{y}_C)$ of C is determined as follows:

$$\widehat{x}_C = \frac{1}{\sharp C} \sum_{(x,y)\in C} x \quad \widehat{y}_C = \frac{1}{\sharp C} \sum_{(x,y)\in C} y \tag{6}$$

Then, the sensor of C which is the nearest neighbor to $(\widehat{x}_C, \widehat{y}_C)$ is selected as key sensor (centroid sensor) for the sampling. The centroid of C is the point location defined as follows:

$$centroid(C) = \operatorname*{argmin}_{(x,y)\in C} EuclideanDistance((x, y), (\widehat{x}_C, \widehat{y}_C)) \tag{7}$$

This recursive subdivision algorithm has a time complexity of O(n), where n is the size of the cluster. It allows us to select a variable number of centroids from C, each one is strategically located in a dense area of C.

4.2 Polynomial

Let (T, V) be a time series of length w, such that, $(T, V) = \langle t_1, v_1\rangle, \langle t_2, v_2\rangle, \ldots,$ $\langle t_w, v_w\rangle$. The goal is to estimate the unknown coefficients of a polynomial p defined as follows:

$$p : V = \alpha + \beta_1 T + \beta_2 T^2, + \ldots + \beta_d T^D \tag{8}$$

such that $D \leq w$ and $p(T)$ fits V according to the minimization of a cost function. The degree D is automatically chosen ($0 \leq D \leq w$) by the forward selection strategy [3] tailored for the polynomial construction and combined with a test to estimate the ability of a polynomial to fit the time series.

Algorithm 2. Function polynomial(T, V, d) **return** p

```
– Main routine(T, V, d) return p
Require: $(T, V)$ {a time series}
Require: $d$ {maximum degree allowed for $p$; in TRECI $d = w$}
Ensure: $p$ {coefficients of the polynomial $V = p(T)$ fitting the time series $(T, V)$}
1: $p \Leftarrow$ forwardPolynomial(costantPolynomial($V$),$T, V, 1, d$)
– forwardPolynomial(previusP, T, V, D, d) return p
1: if $D \leq d$ then
2:     $p \Leftarrow previousP$
3: else
4:     $newP \Leftarrow$ straightLine(residual($T^D$), residual($V$))
5:     if f-test($newP$) then
6:         $p \Leftarrow previuousP$ {The forward addition of the variable $T^D$ to the polynomial
           is not statistically significant for the fitting of the time series $(T, V)$}
7:     else
8:         $p \Leftarrow$forwardPolynomial($newP, T, V, D + 1, d$)
9:     end if
10: end if
```

The polynomial is built stepwise according to Algorithm 2. We start with $D = 1$ and compute the (straight-line) polynomial of V in T (see line 1 of the main routine in Algorithm 2).

The ability of a D-degree polynomial (named *newp*) in fitting (T, V) is evaluated according to the partial F-test. The F-test [3] allows us to evaluate the statistical significance of improvement in the time series fitting due to the addition of the term T^D to the currently constructed polynomial. If this improvement is not statistically significant, the polynomial *previousP*, that is, the polynomial pasty constructed with degree $D - 1$ (the constant polynomial if $D = 1$), is kept and no higher-degree variable is added to the final polynomial (stopping criterion, as reported in line 6 in Algorithm 2). Differently, D is incremented by one and the polynomial of degree D is forward computed by means of the straight-line regression between the residual of V and residual of T^D (see line 8 in Algorithm 2 for the recursive call of the function $forwardPolynomial()$ and line 4 in Algorithm 2 for the computation of straight line between residuals).

The residual of a variable is computed as the difference between the variable and the polynomial of degree $D - 1$ predicting that variable. In particular, the residual of the dependent variable V is the difference between the variable itself and polynomial in T of degree $D - 1$ and fitting V. Similarly, the residual of the independent variable T^D is the difference between the variable itself and polynomial in T of degree $D - 1$ fitting T^D.

The procedure is iterated until $D > w$ (see line 1 in Algorithm 2) or F-test (see line 5 in Algorithm 2) are satisfied.

An example, of this stepwise construction of a polynomial performed according to Algorithm 2 is reported in Example 41.

Example 41 (Forward selection based construction of a polynomial). *Let us consider the case we intend to build the polynomial p of degree $D = 2$,*

$$p\colon V = \alpha + \beta T + \gamma T^2 \tag{9}$$

through a sequence of parametric straight-line regressions. At this aim, we start by regressing V on the 1-degree variable T and building the straight line:

$$\hat{V} = \alpha_1 + \beta_1 T \tag{10}$$

The slope α_1 and intercept β_1 are computed on the time series (T, V). This equation does not fit V exactly. By adding the 2-degree variable T^2, the fitting might improve. However, instead of starting from scratch and building a new polynomial with both T and T^2, we exploit the forward strategy in the polynomial construction. First we build the parametric linear polynomial for T^2 if T is given, that is, $\hat{T^2} = \alpha_2 + \beta_2 T$. Then we define the residuals on both the independent variable T^2 and the dependent variable V, that is:

$$\begin{array}{ll} T^{2'} & = T^2 - (\alpha_2 + \beta_2 T) \\ V' & = V - (\alpha_1 + \beta_1 T) \end{array} \tag{11}$$

Finally, we determine the straight-line regression between residuals V' and $T^{2'}$ in the time series, that is,

$$\hat{V}' = \alpha_3 + \beta_3 T^{2'}. \tag{12}$$

By substituting the straight-line regressions of Equation 11, we reformulate the latter Equation as follows:

$$V - (\alpha_1 + \beta_1 T) = \alpha_3 + \beta_3 (T^2 - (\alpha_2 + \beta_2 T)) \tag{13}$$

This equation can be equivalently written as:

$$V = (\alpha_3 + \alpha_1 - \alpha_2\beta_3) + (\beta_1 - \beta_2\beta_3)T + \beta_3 T^2.$$

It is proved that the polynomial reported in last Equation coincides with the polynomial model built with V, T and T^2 (in Equation 9) since:

$$\alpha = \alpha_3 + \alpha_1 - \alpha_2\beta_3 \tag{14}$$

$$\beta = \beta_1 - \beta_2\beta_3 \tag{15}$$

$$\gamma = \beta_3. \tag{16}$$

A final consideration concerns the time complexity of this forward selection based computation of a polynomial of degree D, that is $O(w \times \frac{D(D-1)}{2})$. This result can be interestingly combined with the consideration reported in [4] (Section 3.3.1, pg 90) according to the degree of a polynomial adequately fitting w values rarely exceeds $\frac{w}{3}$.

4.3 Inverse Distance Weighting

The *Inverse Distance Weighting (IDW)* [7] is here adapted in order to estimate off-line the unknown network measure at any space-time point (x, y, t). First, we identify the past window W of the on-line stream processing which hosts t. Once W is identified, we retrieve the key sensors of the summary (trend clusters) stored in database for the window W and the polynomial of the trends associated to each key sensor.

Let c be a centroid with $c \in centroids(W)$; (x_c, y_c) be the space position of c; $p_c \colon T \mapsto V$ be the polynomial representation of the trend associated to in c; $p_c(t)$ be the value estimated for c at time point t according to the polynomial $p_c(\cdot)$. Then, the interpolate $\hat{v}(x, y, t)$ is computed as follows:

$$\hat{v}(x,y,t) = \begin{cases} p_c(t) & \text{if } \exists c \in centroids(W) \\ & \text{with } (x,y) \equiv (x_c, y_c) \\ \dfrac{\sum\limits_{c \in centroids(W)} w_{(x,y)(x_c,y_c)} \times p_c(t)}{\sum\limits_{c \in centroids(W)} w_{(x,y)(x_c,y_c)}} & \text{otherwise} \end{cases} \tag{17}$$

The idea inspiring Equation 17 is that the interpolation at an un-sampled point location is a function of the known values around it, and depends from them in a relation inversely proportional to the distance, i.e. the nearest is a known value the strongest is its influence. According to this idea, a weight $w_{(x,y)(x_c,y_c)}$ is defined by the inverse of a power of the Euclidean distance:

$$w_{(x,y)(x_c,y_c)} = \mathbf{d}((x,y)(x_c,y_c))^{-p} \tag{18}$$

The power of this influence is driven by the *power parameter* p, which is a positive and real number. If the value of p is less than the dimensionality of the problem (in our spatial context the dimensionality is 2) the interpolation tends to be equally dominated by distant and close points. On the other hand, greater values of p give more influence to the values closer to the un-sampled position. For $p \to \infty$, the IDW method provides the same results of the 1-nearest neighbor interpolation, while for $p \to 0$ it resembles an arithmetic mean. $p = 3$ is a reasonable choice for a 2-D space.

Although Equation 17 considers the entire set of centroids identified across the networked space, we suppose reasonable that an *influence boundary* can be set so that the centroids which are outside this limit should not be taken at all in the computation. Thus, we fix a spheric area around the un-sampled location (x, y); the centroids contribute to the interpolation only if they are inside this spherical region. The center of this *interpolation sphere* is (x, y) and the radius is a boundary parameter b. Under these consideration we reformulate Equation 18 as follows:

$$w_{(x,y),(x_c,y_c)} = \begin{cases} \mathbf{d}((x,y),(x_c,y_c))^{-p} & \text{if } \mathbf{d}((x,y),(x_c,y_c)) \leq b \\ 0 & \text{otherwise} \end{cases}. \quad (19)$$

In TRECI, an automatic mechanism is implemented to choose b at each window by guaranteeing that, independently on (x, y), at least one centroid should be close to (x, y). This requirement has inspired the idea of automatically detect b_W at the window W as the maximum among the distances computed between each pair of closest centroids in the set $centroids(W)$. Formally, let $c \in centroids(W)$ be a centroid of a cluster in W, $min_c[W]$ is the minimum Euclidean distance between c and any other centroid $c' \in centroids(W)$, that is:

$$min_c[W] = \min_{c' \in centroids(W) \wedge c' \neq c} \mathbf{d}(\hat{c}_{C_k}, \hat{c}_{C_k h}). \quad (20)$$

Then b_W is computed as the maximum of the $min_c[W]$, by varying c, that is:

$$b_W = \max_{c \in centroids(W)} min_c[W]. \quad (21)$$

5 Experimental Results

The TRECI framework is written in Java and interfaces a database managed by a MySQL DBMS. We have performed experiments to evaluate both the on-line component (summarizer) and the off-line component (space-time interpolator) of TRECI on an Intel(R) Core(TM) 2 DUO CPU $E4500$ @2.20GHz with 2.0 GiB of RAM Memory, running Ubuntu Release 11.10 (oneiric) Kernel Linux $3.0.0 - 12 - generic$.

Experiments reported i nthis study are performed with the publicly available South American Air Temperature data stream [6]. The main goals of our experiments are to demonstrate that the proposed framework is able to:

Goal 1: Give an accurate and compact summarized representation of the stream by combining the trend clusters with the quadtree based sampling of clusters (goal 1.1) and polynomial representation of trends (goal 1.2).

Goal 2: Use the trend cluster representation of the network stream to interpolate the network measure at any location of the space and at any time point of the past.

5.1 Experimental Setting: Data and Measures

Details on the real sensor network data stream and the evaluation measures considered for the empirical evaluation in this study are reported below.

Data. The South American Air Temperature data stream [6] collects monthly-mean air temperature measurements (in ^{o}C) between 1960 and 1990 over a 0.5°

by 0.5° of latitude/longitude grid of South America for a total of 6477 sensors. In our experiments, the network is obtained by virtually linking each sensor to the sensors which are located in the $1° \times 1°$ around cells of the grid. The recorded temperature values range between -7.6 and 32.9^oC.

Evaluation Measures. The performances of TRECI are evaluated in terms of the memory size consumed to store the trend cluster summarization of the stream and the error of the interpolate.

The size is measured in bytes (b) under the consideration that $bytes(float) = 4b$ and $bytes(integer) = 2b$. In the storage phase we exploit the fact that the nodes of a network can be uniquely identified by an integer key which is one-to-one mapped to the spatial coordinates of the node in the network and the network transmissions are equally spaced in time. According to both these considerations, the network is stored only once for the entire stream (it is updated at any node addition). The transmission time points are not stored in database as they are implicitly obtained by the starting transmission time and the period occurring between consecutive transmission (e.g. the i-th snapshot arrives at $starting + (i-1) \times p$ time point). Therefore, let N be the network, N_s be the subset of the nodes of the network which are sampled at least once by the TRECI summarizer, St be the stream, $snap$ a snapshot of the stream, C be a cluster with trend T, $s(C)$ the sample of centroids extracted from C, $D_{p(T)}$ the degree of polynomial $p(T)$ fitting T and W be a window of the stream segmentation, $\sharp X$ be the number of nodes in a node set X, we obtain that:

$$\begin{aligned}
bytes(N) &= \underbrace{(2+4+4)}_{id\ node + x\ coordinate + y\ coordinate} \times \sharp N \\
bytes(St) &= bytes(N) + \sum_{snap}^{St} \left(\underbrace{2}_{id\ snap} + \underbrace{(2+4)}_{id\ node+value} \times \sharp snap \right) \\
bytes(SUMATRA) &= bytes(N) + \sum_{W}^{St} \left(\underbrace{2}_{id\ W} + \sum_{[C,T]}^{W} (\underbrace{4w}_{T} + \underbrace{2}_{id\ node} \times \sharp C) \right) \\
bytes(TRECI) &= bytes(N_s) + \sum_{W}^{St} \left(2 + \sum_{[s(C),p(T)]}^{W} \left(4(D_{p(T)}+1) + 2 \times \sharp s(C) \right) \right)
\end{aligned} \tag{22}$$

The error in interpolation is measured as an average computed of error committed in interpolating values which are known in several space-time locations. Usually the root mean square error, which is computed as follows, represents a valid estimate of the accuracy of the interpolation:

$$rmse(St) = \sqrt{\frac{\sum_{snap}^{St} \sum_{n}^{snap} (v(n) - interpolate(x_n, y_n, t_{snap}))^2}{\sum_{snap}^{St} \sharp snap}} \qquad (23)$$

where n is a node, $v(n)$ is the value truly measured from the node in the snapshot, (x_n, y_n) is the space location of the node n, t_{snap} is the time point of the snapshot *snap*, $interpolate(x_n, y_n, t_{snap})$ is the interpolated value.

5.2 Results

We present now results of the empirical evaluation of both the summarizer (Goal 1) and the interpolation functionality (Goal 2) available in TRECI. In particular, for the first goal we evaluate how the quadtree-based sampling (Goal 1.1) and polynomial (Goal 1.2) improves the summarization compactness of the trend cluster storage in database. For the second goal, we evaluate the interpolating accuracy by comparing the IDW interpolation with a simple interpolation technique like the Nearest Neighbor (1NN). We also evaluate how the interpolation accuracy may be affected in case we switch off some sensors in the network.

Goal 1.1: Quadtree Sampling Evaluation. Initially, we evaluate the summarizer that, for each trend cluster discovered by SUMATRA, runs the quadtree sampling: the identifiers of the sampled sensors are stored in the database as representative of the cluster, while, as in SUMATRA, the w values of the time series are stored as representative of the trend. We call Qs-S this summarizer which combines SUMATRA with the quadtree sampling.

We compare Qs-SUMATRA with SUMATRA (where for a cluster the identifier of each grouped node is stored in the database), Ss-S (where for a cluster only the central sensor is sampled for the storage in database) and Rs-S (where for a cluster a random choice of a sample of sensors is selected for the storage in database). As in the experiments we intend to compare a sample which is chosen randomly with a sample which is chosen by the quadtree, we set the number of nodes randomly sampled equally to the number of nodes automatically decided with the quadtree.

For the comparison, we evaluate the size of summarized stream and the error performed in reconstructing the stream from its summary. The stream reconstruction is a special case of the space-time interpolation with (x, y, t) the space-time location of each sensor truly transmitting at the time t.

We run SUMATRA by repeating the experimental setting described in [1], window size w=12 and intra-cluster trend similarity threshold[1] $\delta = 4^o$C. According to the definition of a trend cluster, δ is a reasonable upper bound for the error.

[1] SUMATRA groups sensors in a cluster if the values transmitted along the window differ at worst δ from those in the trend polyline of the cluster.

Table 1. Quadtree sampling evaluation: average number of clusters and (sampled) sensors per window, size (KBytes) and error (rmse) of the (summarized) stream

	Stream	SUMATRA	Qs-S	Ss-S	Rs-S
Average number of clusters/window	-	62.63	62.63	62.63	62.63
Average number of (sampled) nodes/window	6477	6477	668.23	62.63	668.23
size	1325.6	548.6	200.1	163.7	200.1
rmse	-	1.25	2.1	4.69	3.78

Table 2. Polynomial evaluation: average degree of learned polynomials, size (KBytes) and error (rmse) of the (summarized) stream

	SUMATRA	Qs-S	TRECI
Average degree of trend polynomials	-	-	5
size	548.6	200.1	168.8
rmse	1.25	1.86	1.97

Results, reported in Table 1, confirm that, as expected, the sampling significantly reduces the size of the stream. Obviously, this size reduction is more impressive whenever a single centroid sensor is sampled for each cluster, but this size reduction is at the expense of the accuracy (see Qs-S vs Ss-S). On the other hand, the quadtree decomposition for the sampling is highly beneficial. First, the sample size is automatically tuned. Second, the strategic selection of those sensors which maintain the information on the cluster shape guarantees an error that is close to the error performed when all sensors are stored in database (see SUMATRA vs Qs-S ans Rs-S). We can conclude that the use of quadtree sampling allows us to obtain the best trade-off between the size and the error of the summary.

Goal 1.2: Polynomial Evaluation. We consider now TRECI (online component) that is the summarizer which combines the trend clusters discovered by SUMATRA with the quadtree sampling of the clusters and the polynomial representation of the trends. Once again, we evaluate the size of the summarized stream and the error in the stream which is reconstructed from the summarize. We compare TRECI with SUMATRA and Qs-S (which is SUMTRA with only the quadtree sampling).

Results, reported in Table 5.2, show that the learned polynomials have an average degree which is definitely lower than w and close enough to the theoretical threshold of $w/3$ suggested in [4] . The use of a polynomial representation of the trends leads to a further reduction of the summary size (see size of TRECI vs Qs-S vs SUMATRA in Table 5.2) which is not at expenses of the accuracy (see accuracy of TRECI vs Qs-S in Table 2).

Table 3. Interpolate evaluation: error (rmse) of the stream interpolation when 50% of sensors are switched-off in the network and/or 50% of time points are jumped in the streaming line

Baseline	Sensors switching-off	Time points jumping	Sensors switching-off and time points' jumping
1.97	2.48	2.72	2.90

Table 4. Interpolate evaluation by varying the percentage of switched-off sensors in the network: IDW vs NN

switched-off sensors %	avg(number of clusters per window)	rmse(stream)	
		1NN	IDV
80%	471.93	7.35	0.77
60%	261.2	5.35	1.08
40%	111.73	9.33	1.23
20%	75.9	9.58	1.36
0%	62.63	8.87	1.97

Goal 2: Interpolate Evaluation. We evaluate the capability of TRECI (offline component) of accurately interpolating the network measure everywhere and anytime also where/when no measure was pasty collected. At this aim, we switch-off some sensors of the network and jump some transmission time points in the streaming line. We use TRECI summarizer (on-line component) to summarize the training stream transmitted from switched-on sensors/time points and we use TRECI interpolate (off-line component) to interpolate the entire stream.

We consider several experimental settings. First, we decide to switch-off 50% of the sensors. Second, we jump the 50% of the transmission points in the streaming line. Finally, we switch off 50% of sensors and jump the 50% of the time points in the streaming line (S50_T50). We analyze accuracy of interpolation compared to the baseline case (S0_T0) where no sensor is switched-off and no time point is jumped. Results, reported in Table 3, show that interpolation error remains significantly below $\delta = 4$, that is, the trend similarity threshold we used in the trend cluster discovery, although we are now able to interpolate at any location across the space and at any past time point.

To complete this study we also evaluate how the accuracy of IDW interpolation is influenced by the density of switched-on sensors in a network. For this kind of analysis, we compare the accuracy of IDW interpolation to the accuracy of a traditional interpolation performed by 1NN; the techniques are compared in case that a percentage of sensors (ranging from 0% to 20%, 40%, 60% and 80%) is switched-off in the network . Results, reported in Table 4, include the average number of trend clusters discovered per window and the rmse in interpolating data of the entire stream (considering both switched-on and switched-off sensors for the interpolation). The analysis of these results confirms that, as expected, IDW greatly improves 1NN. Additionally, they reveal that by increasing the number of switched-off sensors, the number of discovered trend clusters (and reasonably sampled centroids) increases too. This result is mainly due to the fact that the switching-off of sensors has the implicit drawback of breaking down virtual edges between spatially close sensors in the network. By considering that spatially sparse data tend to group in different clusters, we are now able to justifiy the fact that the observed number of discovered trend clusters per window tends to be higher in a network that is less densely populated. The umpteenth consequence of an higher number of discovered trend clusters is that the summary stored in database is larger although less data are streamed from the network. The opposite party of storing more information in database is that interpolation based on this richer knowledge tends to be more accurate. This analysis reveals the crucial role of the network engineering phase which should account both number and position of sensors in order to gain the best trade-off between accuracy of interpolation and compactness of summarization.

6 Conclusion

Trend clusters have been proved to be an effective way to real-time summarize the spatio-temporal data of a sensors network, and that reconstruction of the streamed data from the trend-cluster summary stored in a database is accurate enough. In this paper, we perform a step forward in our research line on the trend cluster discovery by investigating a quadtree sampling algorithm to extract the sample of sensors which are representative of the cluster shape and a polynomial learning to derive a functional representation of the trend. Thus, the storage of a trend cluster in database requires now the storage of both the sampled sensors (an not all the network sensors falling in the cluster), and few polynomial coefficients (and not the entire time series). To interpolate the network measure everywhere in space and anytime in time, we define a version of the Inverse Distance Weighting technique which allows an accurate estimation of the missing/unknown data by computing the inverse distance weighting of this trend cluster representation of observed data.

Both the summarizer and the interpolate have been empirically evaluated in a real sensor network stream and results show effectiveness (compactness of summary and accuracy of interpolation) of our proposal.

As further work, we plan to evaluate the use of Kriging as an alternative to the IDW interpolation technique.

Acknowledgments. This work is supported in fulfillment of the research objectives of both the project: “EMP3: Efficiency Monitoring of Photovoltaic Power Plants” funded by “Fondazione Cassa di Risparmio di Puglia,” and the PRIN 2009 Project “Learning Techniques in Relational Domains and Their Applications” funded by the Italian Ministry of University and Research (MIUR).

References

1. Ciampi, A., Appice, A., Malerba, D.: Summarization for Geographically Distributed Data Streams. In: Setchi, R., Jordanov, I., Howlett, R.J., Jain, L.C. (eds.) KES 2010, Part III. LNCS, vol. 6278, pp. 339–348. Springer, Heidelberg (2010)
2. Ciampi, A., Appice, A., Malerba, D., Guccione, P.: Trend cluster based compression of geographically distributed data streams. In: CIDM 2011, pp. 168–175. IEEE (2011)
3. Draper, N.R., Smith, H.: Applied regression analysis. Wiley (1982)
4. Fabbris, L.: Statistica multivariata. McGraw-Hill (1997)
5. Guccione, P., Ciampi, A., Appice, A., Malerba, D.: Trend cluster based interpolation everywhere in a sensor network. In: Proceedings of the 2012 ACM Symposium on Applied Computing, Data Stream, ACM SAC(DS) 2012 (2012)
6. S.A.A. Temperature, `http://climate.geog.udel.edu/climate/html_pages/sa_air_clim.html`
7. Tomczak, M.: Spatial interpolation and its uncertainty using automated anisotropic inverse distance weighting (IDW) - cross-validation/jackknife approach. Journal of Geographic Information and Decision Analysis 2(2), 18–30 (1998)
8. Kim, B., Tsiotras, P.: Image segmentation on cell-center sampled quadtree and octree grids. In: SPIE Electronic Imaging / Wavelet Applications in Industrial Processing VI (2009)
9. Willett, R., Martin, A., Nowak, R.: Backcasting: A new approach to energy conservation in sensor networks. In: Information Processing in Sensor Networks, IPSN 2004 (2003)
10. Yong, J., Xiao-Ling, Z., Jun, S.: Unsupervised classification of polarimetric sar image by quad-tree segment and svm. In: 1st Asian and Pacific Conference on Synthetic Aperture Radar, APSAR 2007, pp. 480–483 (2007)

Discovering Sensor Services with Social Network Analysis and Expanded SQWRL Querying

Mohamed Bakillah[*] and Steve H.L. Liang

Department of Geomatics Engineering, University of Calgary, Alberta, Canada
mohamed.bakillah.1@ulaval.ca, steve.liang@ucalgary.ca

Abstract. The Open Geospatial Consortium (OGC) Sensor Web Enablement (SWE) initiative enables access of sensor data over the Web. The growing amount of available sensors requires discovering mechanisms to find sensor services relevant for users. These mechanisms must rely on explicit sensor metadata model and address the problem of semantic heterogeneity. We present a new discovering approach that is based on a novel sensor metadata model compliant with SWE standards, where sensors' descriptions are referenced to a common semantic reference frame to support resolution of semantic heterogeneities. The discovery mechanism comprises two steps: first, a network-analysis-based partitioning algorithm modularizes the set of sensors into meaningful sensor clusters, which semantics is generated with aggregation operators. Secondly, an expanded SQWRL rule-based inference engine processes the queries over the sensor clusters and issues the relevant sensor services. This approach allows users to find the sensor services relevant to their needs.

Keywords: Semantic Sensor Web, Sensor Discovery, Sensor Web Enablement, Network Analysis, SQWRL.

1 Introduction

The recent improvements in sensor technologies have modified the production and processing of real-time geospatial data. The main objective of the OGC's Sensor Web Enablement (SWE) initiative is to make sensors discoverable and accessible on the Web through standardized interfaces and specifications. For example, the Sensor Observation Service (SOS) enables the registration and storage of sensors and their observations, which are then accessible to clients. Despite efforts enabling syntactic interoperability, efficient mechanisms for discovering relevant sensor service that fits user's needs are still missing. Every community has its own perception of the geographical space, which results in semantic heterogeneity of sensor descriptions [1][2][3]. The lack of efficient mechanisms for discovering relevant sensor service is therefore due to the lack of consistent semantic representation of sensor service and the lack of discovery mechanism that can incorporate semantics. This issue is acknowledged by Sheth et al. [2], who proposed the concept of Semantic Sensor Web, a

[*] Corresponding author.

combination of the Sensor Web technology with the Semantic Web technologies. Several researches have contributed to this field, by introducing semantic-enabled Sensor Observation Services and semantic annotation services for sensor data [4] [5] [6]. In this paper, we propose a discovery approach for semantically heterogeneous sensor services that addresses this semantic challenge. We propose a model for sensor metadata that integrates the elements needed to assess sensor service relevance with respect to users' requirements. We also consider that due to the large volume of available sensor services, it is not realistic to query them all. Therefore, with respect to existing approaches, our contribution is to:

- Develop a semantic partitioning approach that modularizes the set of sensors services into meaningful sensor clusters; the partitioning approach is based on network analysis techniques. This paper is methodologically innovative as network analysis has not yet been applied to support sensor service discovery.
- Propose an approach based on thematic, spatial and temporal aggregation operators to generate semantic descriptions of sensor clusters;
- Integrate the partitioning approach into the discovering process and exploit the Semantic Query-enhanced Web Rule Language (SQWRL) to support semantic reasoning and process queries.

The paper is structured as follows: first, we provide a background on the SWE initiative and related work on sensor service discovery. In Section 3, we present our model for sensor metadata. Section 4 presents the discovering approach, and Section 5 illustrates the approach with an application example. Conclusions and future work are given in Section 6.

2 Enabling Discovery of Sensor Services: Related Work

2.1 Sensor Web Enablement

The OGC SWE initiative has developed standards to make sensor data accessible on the Web; for example, the Sensor Observation Service (SOS) is a standard interface for accessing descriptions of sensors and their observations [7]. To support the discovery of relevant sensors, we need a rich and formalized description of the sensors and of the data they produce [6][8][4]. Sensor Model Language (SensorML) is the SWE standard to describe metadata of sensors [7]. However, one of the drawbacks of SensorML is that it is a generic standard that allows specifying the same information through different structures [6]. This makes it difficult to process and compare different SensorML descriptions automatically. To support discovering, we need a consistent set of relevant metadata elements to describe all available sensors services. To address this issue, we have developed a sensor metadata model that will be used in the proposed discovering approach.

2.2 Sensor Data and Service Discovery

Semantic Sensor Web is still at an early stage of development [4][9]. Existing approaches related to sensor discovery emphasize that comprehensive and consistent sensor metadata, including description of sensors, observations, and feature types, are needed to support discovery [10][2]. For example, Bröring et al. [4] indicate that a service supporting the semantic annotation of new sensors and observations being added to an SOS is needed. To do so, Devaraju et al. [10] proposed an ontology of processes that could support the semantic annotation of sensor ontologies. Similarly, Henson et al. [5] highlight that to support reasoning over sensor observations, sensor data must be semantically referenced to meaningful concepts that can be processed by reasoning engines. They proposed a semantic SOS, named SemSOS, which provides access to ontological knowledge on sensor observations. Information on sensor instances is stored in a knowledge base, against which SPARQL queries are run to access sensor data. A similar service is proposed by Knoechel et al. [1], where Unique Resources Identifiers (URIs) used by SOS to represent the observed phenomena are linked to a dictionary to resolve URIs heterogeneities. Jirka et al. [6] proposed a service within the OSIRIS Sensor Web Discovery Framework, namely the Sensor Instance Registry (SIR), which allows the user to search for sensors that match a given set of criteria. To match the query with the sensor descriptions, they employ subsumption reasoning which relies on the Semantic Web for Earth and Environmental Terminology (SWEET) ontology. The above-mentioned approaches provide different ways of specifying the semantics of sensor data and services, essentially through ontologies. As a result, the discovery of sensor data and services can be automated or semi-automated. However, they still do not propose a comprehensive sensor metadata model that would provide a uniform structure to describe semantics. As a result, the issue being raised by the generic aspect of SensorML still remains. Therefore, in our approach, we emphasize the need of providing a comprehensive and formalized sensor description structure to support the discovery process. In addition, we also address another issue that affects the efficiency and scalability of the discovery process. Because the volume of available sensor data sets is constantly growing [4], we need to avoid comparing each sensor description with the query each time a query is submitted, as it is done in existing approaches. To address this additional issue, we propose a partitioning approach that structures the set of available sensors into meaningful clusters of sensors and computes a semantic description for these clusters. The partitioning approach is integrated into the global discovery approach, which uses the more expressive SQWRL language.

3 Sensor Metadata Model

The sensor metadata model defines the parameters that are needed to assess the relevance of a sensor service with respect to the user's needs. By committing to a common metadata model, applications can benefit from shared semantics and therefore enhance semantic interoperability [8]. The proposed model, which is formalized on Figure 1 with UML, is compliant with the OGC's SWE standards for modeling sensor data and observations.

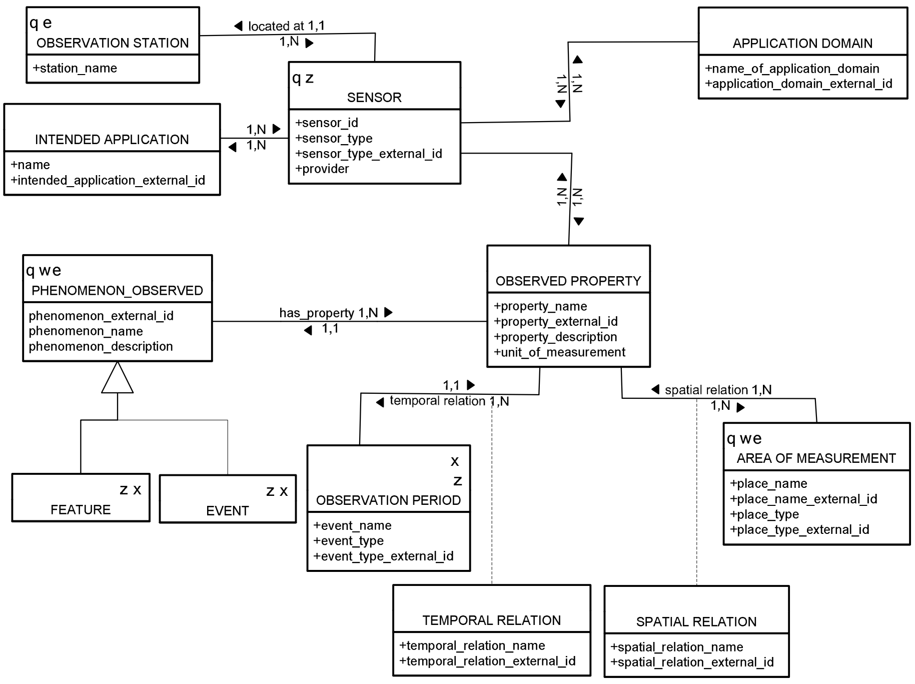


Fig. 1. Sensor Metadata Model

A sensor is described by the sensor type, the provider of the sensor, its localization, and the observation station. We associate the sensor to its *intended application* and its *application domain* (e.g., hydrology, meteorology, etc.). The intended application is the type of activity that is originally meant to be performed with the data (e.g., soil moisture monitoring, toxic gas dissemination monitoring, etc.). In the O&M sensor data model, each sensor is linked to one or several observations performed on a phenomenon. In our model, the phenomenon is abstracted with the class *phenomenon observed*. The phenomenon observed can be a physical object, represented with the class *feature*, or an *event* (e.g. a storm, an earthquake, etc.). Observed properties are qualities of the phenomenon that can be measured, such as soil moisture, wind speed, etc. The *observed property* class is linked to the *area of measurement* class, which represents the point, line or polygon where the observations were made. The area of measurement is associated to a *place*, which is the name of the location (e.g. university of Calgary) and the type of place (e.g. university). This alternative way (besides spatial coordinates) of representing the location of the measurements is offering several ways of specifying the location of interest. The observed *property* class is linked to the area of measurement with a *spatial relation*. For example, the observations could have been made *near* university of Calgary. Including spatial relations into the model allows the user to specify more intuitive queries involving places of interest.

The *observed property* class is also linked to an *observation period*. This period covers the time during which the measurements were made; it can be a time interval or an event. Just as spatial coordinates can be unintuitive compared to the place of interest (e.g. name of a city) [8], a time interval to represent the timeframe of interest can be harder to specify than the corresponding event (e.g., hurricane Katrina). The *observed property* class is linked to the *observation period* with a temporal relation. For example, a set of observations could have been performed *before* the hurricane. Because each sensor is made available by different providers, sensor descriptions are semantically heterogeneous. To resolve those heterogeneities, the terms used in sensor descriptions are referenced to a common and formal vocabulary, i.e., a semantic reference system, or reference ontology [11]. According to Kuhn and Raubal [12], the semantic reference system consists of a semantic datum, which is the basic vocabulary used to describe a given universe of discourse; a semantic reference frame (SRF), which is a concept structure defining the conceptualization underlying the use of this vocabulary; and a function that links a term used in an application (for example, a term used within the description of a sensor) to a concept in the SRF. We propose to reference the terms used within the sensor descriptions to the SWEET (Semantic Web for Earth and Environmental Terminology) ontologies. They contain categories such as *realm*, which corresponds to the *application domain* and includes the subcategories *ocean*, *atmosphere*, etc. SWEET also includes an ontology for *observations*, *phenomena*, *human activities* and *processes* (which encompasses the concepts falling under *intended application*) and temporal concepts. Similarly, places are referenced to GeoNames, which is a geographical dataset that contains over 8 million geographical names and where location names are associated to coordinates but also to a type of place (building, city, etc.). Places in GeoNames are also linked by inclusion relations.

4 Our Approach

The discovery approach is illustrated on Figure 2. While it is out of the scope of this paper to describe how sensor descriptions are generated, we assume that they are available as OWL instances of the proposed sensor metadata model. Firstly, the partitioning algorithm is deployed to create meaningful subsets of sensor, called sensor clusters. For every cluster created, we generate a cluster description that is analogous to the individual sensors' descriptions. This generation can be automated with aggregation operators that we have defined. The result is an OWL knowledge base of sensor clusters' descriptions. Queries are formulated as SQWRL query which are first processed against the cluster knowledge base, resulting in a set of relevant clusters. To retrieve the final set of relevant sensors, the query is processed against the sensors that are part of the selected clusters.

4.1 Semantic Partitioning with Social Network Analysis

The goal of the partitioning algorithm is to discover meaningful sensor clusters. To do so, we employ social network analysis methods. Network analysis is a set of methods for discovering structures in various types of networks [13], using concepts such as centrality, closeness and density. Our algorithm identifies, within the available sensors, those that can be considered as central because their characteristics encompass those of other sensors. For example, a sensor that "measures density of gas" encompasses sensors that "measure density of CO_2", "measure density of air pollutant," etc. Meaningful clusters of sensors are formed around those central sensors. To do so, the algorithm first determines the semantic affinity between pairs of sensors. The affinity values are analyzed to discover the central sensors. For each identified central sensor, we search its semantic neighbourhood to select the sensors that will be part of the cluster formed around this central sensor.

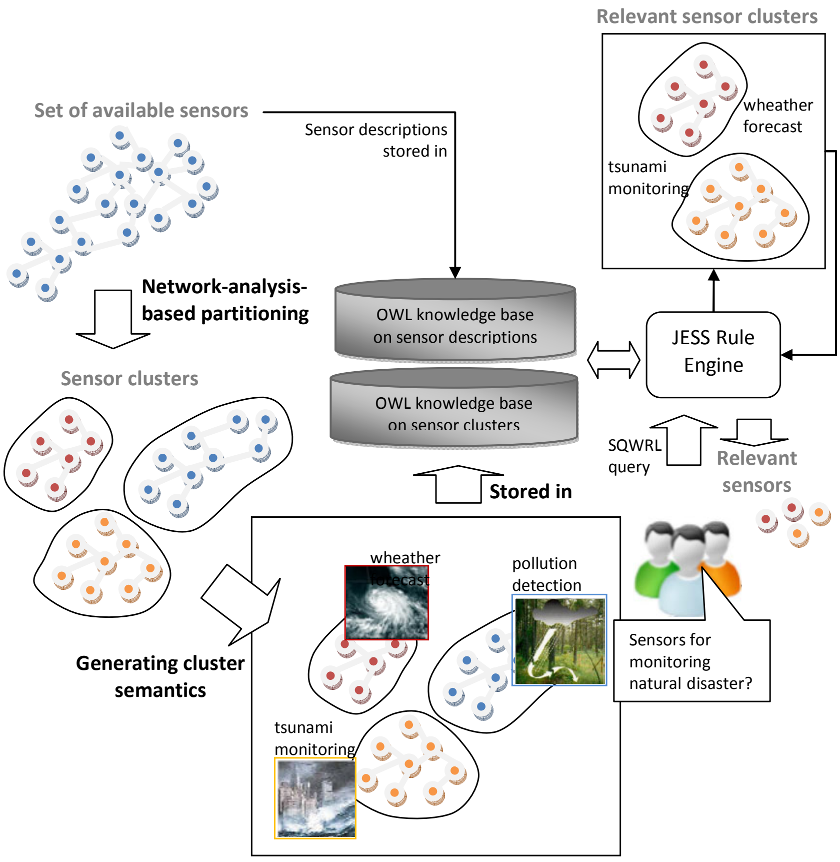

Set of available sensors
Sensor descriptions stored in
Relevant sensor clusters
wheather forecast
tsunami monitoring
Network-analysis-based partitioning
OWL knowledge base on sensor descriptions
OWL knowledge base on sensor clusters
JESS Rule Engine
Sensor clusters
SQWRL query
Relevant sensors
Stored in
wheather forecast
pollution detection
Sensors for monitoring natural disaster?
Generating cluster semantics
tsunami monitoring

Fig. 2. The discovery approach

4.1.1 Semantic Affinity

The role of the semantic affinity in the partitioning algorithm is to support the discovering of sensors which description partly or completely includes the description of other sensors. Consequently, the semantic affinity we will use must measure the degree of inclusion of a sensor description S_i into another sensor description S_j. We decompose the problem of measuring the degree of inclusion of S_i into S_j into the problem of measuring the degree of inclusion of each element ε_{ik} of S_i into the corresponding element ε_{jk} of S_j (with k = *phenomenon observed*, *observed property*, *application domain*, etc.). We employ an ontology-based approach, where the degree of inclusion of an element ε_{ik} into ε_{jk} is a function of the semantic distance between ε_{ik} and ε_{jk}. This semantic distance is measured into the SRF, denoted O. The semantic distance is more appropriate than string-based comparison measures, because it takes into account the semantics of the terms being compared. The semantic distance is based on the relative position of ε_{ik} and ε_{jk} in ontology O. Let $<_O$ be a hierarchical, is-a relationship between terms in O, where $t <_O t'$ means that the term t is more specific than t'. Let $P(\varepsilon_{ik}, \varepsilon_{jk})$ be the path that links ε_{ik} to ε_{jk} in O, according to $<_O$: $P(\varepsilon_{ik}, \varepsilon_{jk}) = \{\varepsilon_{ik}, t1, t2, \ldots \varepsilon_{jk}\}$ so that $t1$, $t2$, … is the ordered set of nodes from ε_{ki} to ε_{kj} in O. The semantic distance is defined as:

$$\delta_k(\varepsilon_{ik}, \varepsilon_{jk}) = \begin{cases} 1 & \text{if } \varepsilon_{ik} = \varepsilon_{jk} \\ |P(\varepsilon_{ik}, \varepsilon_{jk})| & \text{if } \varepsilon_{ik} \prec \varepsilon_{jk} \\ 0 & else \end{cases}$$

The semantic distance is asymmetric: if an element ε_{ik} is more general than ε_{jk}, then $\delta(\varepsilon_{ik}, \varepsilon_{jk})$ is null. We define the semantic affinity as follows: let two sensor descriptions S_i and S_j. Consider $\delta_k(\varepsilon_{ki}, \varepsilon_{kj})$, the semantic distance between the elements of type k of S_i and S_j. Let φ_k, with φ_k part of the [0, 1] interval, be the weight given to element k in the computation of the semantic affinity. The semantic affinity between S_i and S_j is given by:

$$SA(S_i, S_j) = \sum_k \frac{\varphi_k}{\delta_k(\varepsilon_{ki}, \varepsilon_{kj})} \text{ with } \sum_k \varphi_k = 1$$

The weights given to the different types of elements allow partitioning the set of available sensors according to chosen elements.

4.1.2 Partitioning Algorithm

The input of the algorithm is a set of n sensors. The output is the set of meaningful sensor clusters. First, we compute the semantic affinity among couples of sensor descriptions. The computed values are stored in an adjacency matrix A:

$$A = \begin{pmatrix} 1 & SA(S_1, S_2) & \ldots & SA(S_1, S_n) \\ SA(S_2, S_1) & 1 & \ldots & \ldots \\ \ldots & \ldots & 1 & \ldots \\ SA(S_n, S_1) & \ldots & \ldots & 1 \end{pmatrix}$$

The entries of A form a semantic network where nodes represent sensors and edges represent semantic affinity between sensors' descriptions. To determine whether a sensor is central, we use the degree of centrality index [14], which is an indicator of the density of links that emanate from a node. The higher the degree of centrality of a sensor, the more we can consider this sensor as central. The degree of centrality of a sensor S_i is given by:

$$DC(S_i) = \frac{1}{nb\_sensors} \sum_{j=1}^{n-1} SA(S_i, S_j)$$

To determine if the degree of centrality of a sensor S_i is high enough to consider it as a central sensor, we compute the deviation of $DC(S_i)$ from the mean value of DC for all sensors. The mean value of DC, denoted by μ, is given by:

$$\mu = \frac{2}{nc(nc-1)} \sum_{i=1}^{n-1} \sum_{j=i+1}^{n} SA(S_i, S_j)$$

The deviation of $DC(S_i)$ from the mean value is given by:

$$\Delta(S_i) = (DC(S_i) - \mu)$$

If the deviation $\Delta(S_i)$ of a sensor is higher than a given threshold value Δmin, S_i is designated as a central sensor. If no central sensor is detected at this stage, we can consider a lower value for Δmin. When central sensors are identified, for each of them a meaningful cluster of sensors is formed. A cluster formed around a sensor S_i is denoted with Π_i, and the set of sensors that are part of Π_i is denoted with $members(\Pi_i) = \{S_1, S_2, \ldots S_i \ldots\}$. To select the sensors that will be part of Π_i, we follow an iterative process where we examine the successive neighbourhood layers of the central sensor. We denote the z^{th} neighbourhood layer the z^{h}-order neighbourhood. The first order neighbourhood of a central sensor S_i, denoted with FON(S_i), is the set of sensors for which semantic affinity with S_i is higher than the semantic affinity threshold SA_{min}. The second order neighbourhood of a central sensor, denoted with SON(S_i), is the set of sensors for which semantic affinity with a sensor of FON(S_i) is higher than SA_{min}. The general rule is that the z-order neighbourhood of a central sensor z-ON(S_i) is the set of sensors for which semantic affinity with a sensor of (z-1)ON(S_i) is higher than SA_{min}. The role of the z-order neighbourhood is to indicate how semantically close from the central sensor another sensor is. The sensors that will be part of Π_i are those that are situated within the x-order neighbourhood of S_i; x is a parameter that can be set according to the number of available sensors. The partitioning algorithm goes as follow:

Algorithm 1. Partitioning algorithm

Partition (List <sensor description>): List <sensor cluster>

```
1   declare and initialize a list of sensor cluster sensor_cluster_list
2   declare and initialize an adjacency matrix A
3   declare and initialize a list of central sensor central_sensor_list
```

```
compute affinity among sensor descriptions and store values in $A$
for all sensor description $S_i$
    compute degree of centrality $DC(S_i)$
compute centrality mean $\mu$ for adjacency matrix $A$
for all sensor description $S_i$
    compute deviation $\Delta(S_i)$
    if $\Delta(S_i) > \Delta_{min}$
        select $S_i$ as a central sensor
        add $S_i$ to *central_sensor_list*
for all central sensor $S_i$ in *central_sensor_list*
    create cluster $\Pi_i$
    initialize *members*($\Pi_i$) = {$S_i$}
    initialize *previous_neighbourhood*($S_i$) = {$S_i$}
    initialize *nb_neighbourhood* = 0
    while *nb_neighbourhood* < $x$ || total number of sensors is reached
        create *next neighbourhood*($S_i$)
        for all sensor $S_j$ not included in *previous_neighbourhood*($S_i$)
            for all sensor $S_k$ included in *previous_neighbourhood*($S_i$)
                if $A_{jk} > SA_{min}$
                add $S_j$ to *next neighbourhood*($S_i$)
                add $S_j$ to *members*($\Pi_i$)
         *previous_neighbourhood*($S_i$) = *next neighbourhood*($S_i$)
        *nb_neighbourhood*++
    add $\Pi_i$ to *sensor_cluster_list*
    return *sensor_cluster_list*
```

Once sensor clusters are formed, we need to determine the descriptions of these clusters. To obtain a cluster's description, we merge the descriptions of sensors that are part of the cluster. To do so, we have defined aggregation operators.

4.1.3 Aggregation Operators

We provide three operators: *thematic*, *spatial*, and *temporal aggregation operators*.

Thematic Aggregation Operator. This operator produces the thematic elements of a cluster description (e.g., taking the intended application of every sensor of a cluster and returning a global intended application for the cluster). The operator identifies the thematic elements of a cluster' members in the SRF, and retrieves the nearest common subsumer of the input elements. **Them_Agg(Π_i, themEl)** takes as input a cluster Π_i and a type of thematic element, *themEl*, and returns the corresponding element for the cluster. The types of thematic elements are the application domain *AD*, the intended application *IA*, the phenomenon observed *PhO*, and the property observed *PrO*. The thematic aggregation operator is defined as:

$$Them\_Agg(\Pi_i, themEl) = \bigcup_{j=0}^{|\Pi_i|} themEl(S_j), \forall S_j \in \Pi_i ,$$

where *themEl*(S_j) is the thematic element of type *themEl* of sensor S_j. The union operator is defined as:

$$\bigcup_{j=0}^{|\Pi_i|} themEl(S_j) = NCS(themEl(S_1), themEl(S_2), \ldots, themEl(S_j, \ldots) \ldots)$$

where *NCS* is the nearest common subsumer of elements $themEl(S_1)$, $themEl(S_2)$, …, $themEl(S_j)$ … in the SRF.

Spatial Aggregation Operator. The function of the spatial aggregation operator is to compute the spatial elements of a cluster's description. The types of spatial elements *spatialEl* include the area of measurement; the location of sensors; and the geometry of the phenomena observed. The spatial element of type *spatialEl* for a cluster is the smallest area that encloses the spatial elements of type *spatialEl* of the cluster's members. **Spatial_Agg(Π$_i$, spatialEl)** takes as input a cluster Π_i and a type of spatial element, *spatialEl*, and returns the corresponding element for the cluster. The spatial aggregation operator is defined as:

$$Spatial\_Agg(\Pi_i, spatialEl) = \bigcup_{j=0}^{|\Pi_i|} spatialEl(S_j), \forall S_j \in \Pi_i ,$$

where $spatialEl(S_j)$ is the spatial element of type *spatialEl* of sensor S_j. The union operator is defined as follows:

$$\bigcup_{j=0}^{|\Pi_i|} spatialEl(S_j) = SEA(spatialEl(S_1), spatialEl(S_2), \ldots, spatialEl(S_j, \ldots) \ldots)$$

where *SEA* is the Smallest Enclosing Area that includes elements $spatialEl(S_1)$, $spatialEl(S_2)$, …, $spatialEl(S_j)$. Currently, the spatial aggregation operator takes as input only places (no coordinates), and return the smallest place that include the input places, as provided by the GeoNames database. This smallest place is the *SEA*.

Temporal Aggregation Operator. The function of the temporal aggregation operator is to compute the temporal elements of a cluster's description. The sole type of temporal element is the observation period. The observation period for a cluster is the shortest time period that includes the observation periods of the cluster's members. **Temporal_Agg(Π$_i$, obsPeriod)** takes as input a cluster Π_i and returns the observation period for the cluster. The temporal aggregation operator is defined as:

$$Temporal\_Agg(\Pi_i, obsPeriod) = \bigcup_{j=0}^{|\Pi_i|} obsPeriod(S_j), \forall S_j \in \Pi_i ,$$

The union operator is defined as:

$$\bigcup_{j=0}^{|\Pi_i|} obsPeriod(S_j) = STP(obsPeriod(S_1), obsPeriod(S_2), \ldots, obsPeriod(S_j, \ldots) \ldots)$$

*STP*is the Shortest Time Period that includes all individual observation periods. If the observation periods of sensors are described by events, an event base is needed to retrieve the event that corresponds to this shortest time period. The result of partitioning and merging sensors' descriptions is a set of sensor cluster and the semantic description of these clusters.

4.2 Rule-Based SQWRL Discovery of Relevant Sensor Services

Sensor and clusters descriptions are formalized as OWL individuals, so the problem of discovery becomes a matter of querying OWL base. In our approach, we propose to use rule-based query language, which is more expressive and adapted to OWL semantics than SPARQL, as demonstrated in O'Connor and Das [15]. Rules are typically employed to derive implicit knowledge from existing facts [16][17]. In our approach, we used SQWRL to express queries. The Jess Rule engine [18] is leveraged to process the queries. Figure 3 summarizes the query processing.

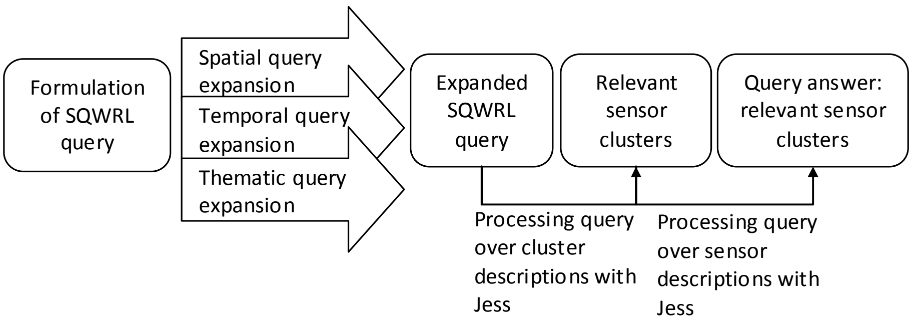

Formulation of SQWRL query
Spatial query expansion
Temporal query expansion
Thematic query expansion
Expanded SQWRL query
Relevant sensor clusters
Query answer: relevant sensor clusters
Processing query over cluster descriptions with Jess
Processing query over sensor descriptions with Jess

Fig. 3. Two-level query processing

Firstly, the SQWRL query is formulated by the user. Then, we proceed to a spatial, temporal and thematic query expansion. The principle of query expansion is to derive alternative ways of formulating the query, in order to encompass the diversity of ways used to formalize sensors' descriptions. The query expansion phase enables the resolution of semantic, spatial and temporal heterogeneities among sensors' descriptions. The expanded query will be processed with Jess rule engine at two levels: first, it is processed over the sensor clusters' descriptions, in order to discover the relevant sensor clusters. Then, it is processed over the sensors of relevant clusters, in order to retrieve the relevant sensors.

4.2.1 Query Formulation with SQWRL

SQWRL is a query language for OWL that is built on SWRL. SWRL expresses Horn-like rules in terms of OWL classes. A SWRL rule is composed of several atoms, which are statements of the form $C(x)$ (variable x is instance of the OWL class C), $P(x, y)$ (x is linked to y via property P), or built-ins, such as SameAs(x, z). A SWRL rule expresses a logical implication between an antecedent and a consequent. SQWRL takes a SWRL rule antecedent and considers it as a pattern specification for the query; the consequent is replaced with a retrieval specification [15]. The SQWRL: select operator takes as input the variables used in the query's pattern specification, and issues a table which columns correspond to the variables. For example, consider the following: *what are the sensors that monitor wind speed near university of Calgary*? The corresponding SQWRL query writes as:

```
Sensor(?x)∧PhenomenonObserved(?y)∧ObservedProperty(?z)∧
hasPhenomenonObserved(?x, ?y)∧hasObservedProperty(?y, ?z)∧
AreaOfMeasurement(?a)∧near(?z, ?a)∧SameAs(?y, wind)∧
SameAs(?z, wind speed)∧SameAs(?a, university of Calgary) → sqwrl:select(?x)
```

The sensors which description matches the conditions stated in the antecedent are returned.

4.2.2 Query Expansion and Processing with Jess Rule Engine

Query expansion takes the initial query and returns a set of equivalent queries. This enables to resolve semantic heterogeneities such as thematic granularity heterogeneities. For example, consider a query with required value *gas property* as *observedProperty*. The sensors which description contains *physical gas property* as *observedProperty* should be part of the query results. Similarly, processing a query which requires Calgary as measurementArea should produce results that include sensors which measurementArea corresponds to administrative subdivisions within Calgary.

The query expansion algorithm (see below) searches for required values *v* in the antecedent of *q*. For example, in the above example, *wind* is the required value for *PhenomenonObserved*. Then, the algorithm verifies if *v* is required for an expandable class, i.e., a class which instances are referenced in the SRF. *Intended application*, *application domain*, *sensor type*, *phenomenon observed*, and *observed property* are the expandable classes representing thematic elements. The algorithm retrieves the concepts of the SRF which are subsumed by the concept to which the required value is referenced.

Algorithm 2. Query expansion algorithm for thematic elements and places

Expand (query *q*): List <query>

```
29 declare and initialize a list of queries equivalent_Query
30 add q to equivalent_Query
31 for all required value v in antecedent of q
32     if v is required for an expandable class C
33         get concept c1 corresponding to v in SRF
34         for all instance I of current expandable class C
35             get concept c2 corresponding to I in SRF
36             get relation r between c1 and c2 in SRF
37             if r == subsumes || equal
38                 create a copy q' of q
39                 replace v with c2 in q'
40                 add q' to equivalent_Query
41 return equivalent_Query
```

Figure 4 gives an example of spatial query expansion that uses Open Cyc Spatial Relation Ontology (Fig. 5) as SRF to expand a spatial relation *near*.

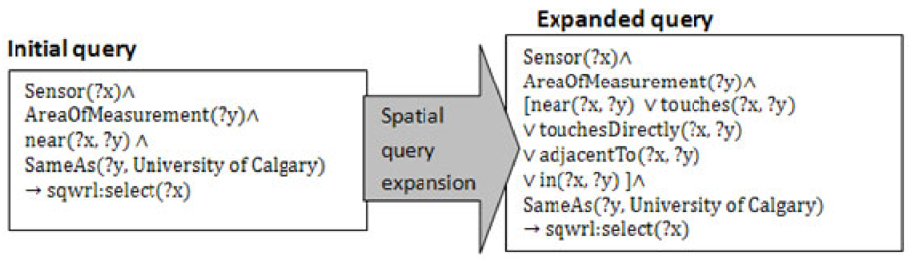


Fig. 4. Example of spatial query expansion

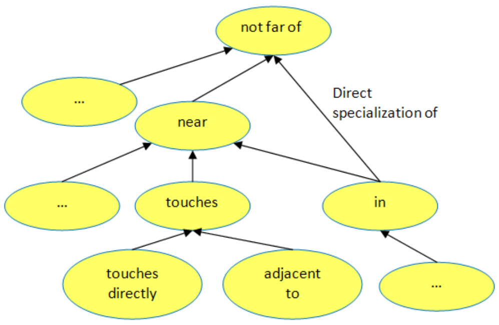


Fig. 5. Fragment of Open CyC Spatial Relations Ontology

The spatial query expansion deals with the spatial relations that compose queries, such as "sensors *near* university of Calgary." The user would expect retrieving not only the sensors that are *near* the university, but also those that are *close* or *connected* to the university, for example. The query expansion algorithm uses the SWEET and Open Cyc ontologies; therefore, it deals with the synonyms that are included in these ontologies. However, to improve the ability to resolve naming heterogeneities, other lexical databases could be employed during query expansion. However, this would also raise the issue of the possible blow-up of the rewritten query. To reduce this impact, we consider to restrain the number of hierarchical levels in the SRF that are considered during query expansion (e.g., expanding a term with only the concepts situated at no more than two levels below the concept matched to the term in the SRF).

When the query is expanded, it is first processed against the descriptions of clusters. We use the SQWRL editor available in Protégé, while the Jess rule engine implemented in the JessTab plugin of Protégé ontology editor supports the execution of SQWRL queries. Jess automatically converts an OWL knowledge base into Jess assertions, and retrieves the instances that satisfy the extended query. Running the Jess engine over clusters' descriptions results into a set of relevant clusters. Then, the query is processed against the description of sensors that are part of the retrieved clusters; the result is the set of relevant sensors.

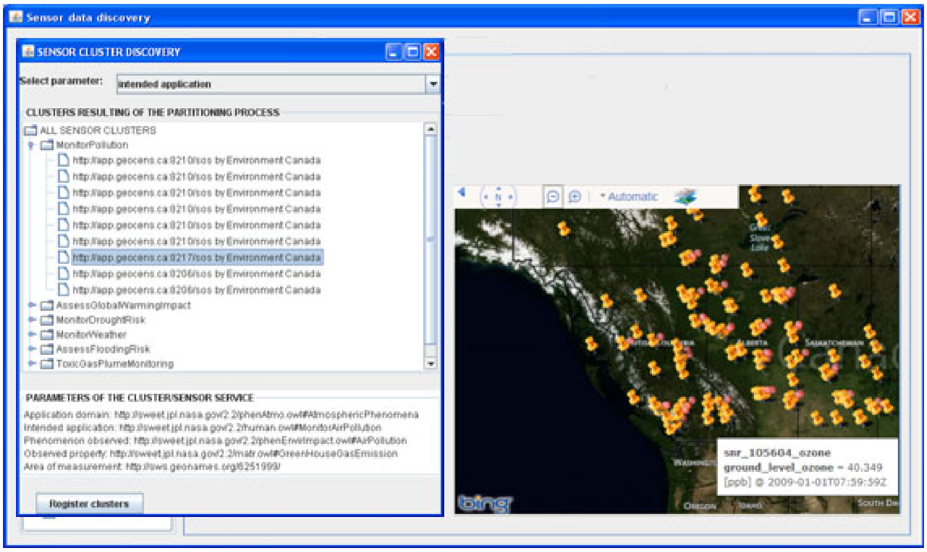
5 Application Example and Experimentation

This section shows the advantages of the proposed approach. We used the SensorML descriptions available on the Geospatial Cyberinfrastructure for Environmental Sensing platform (GeoCens) as a basis for building an OWL repository of sensor descriptions. The sensor data cover application domains such as meteorology, hydrology, atmosphere and soil for regions across Canada. Because the SensorML descriptions did not comprise all the parameters included in our model (such as intended application), elements were added to these descriptions. We ran the partitioning algorithm with different values of weights φ_k for the semantic affinity, the deviation threshold Δmin, and $SA_{\min}$. The more meaningful clusters were obtained with maximal weight being given either to the intended application and phenomena observed parameters, and with Δmin = 0,30 and $SA_{\min}$ = 0,25. Table 1 lists a sample of the resulting clusters.

Table 1. Sample of discovered sensor clusters based on phenomena observed

Cluster Phenomena Observed in SRF	Number of sensor services in cluster
http://sweet.jpl.nasa.gov/2.2/phenAtmo.owl#MeteorologicalPhenomena	35
http://sweet.jpl.nasa.gov/2.2/phen.owl#Precipitation	4
http://sweet.jpl.nasa.gov/2.2/phenAtmoWind.owl#Wind	9
http://sweet.jpl.nasa.gov/2.2/phenAtmoPrecipitation.owl#Snowfall	9
http://sweet.jpl.nasa.gov/2.2/matrWater.owl#Ice	2
http://sweet.jpl.nasa.gov/2.2/phenOcean.owl#Ocean	3
http://sweet.jpl.nasa.gov/2.2/phen.owl#RadiationalHeating	9
http://sweet.jpl.nasa.gov/2.2/matr.owl#SuspendedSubstance	6
http://sweet.jpl.nasa.gov/2.2/matrWater.owl#GroundWater	2
http://sweet.jpl.nasa.gov/2.2/realmHydro.owl#HydrosphereFeature	9
http://sweet.jpl.nasa.gov/2.2/realmCryo.owl#FrozenGround	9
http://sweet.jpl.nasa.gov/2.2/realmGeolContinental.owl#ContinentalLithosphere	8
http://sweet.jpl.nasa.gov/2.2/phenEnvirImpact.owl#AirPollution	8

The clusters are described with the concepts of the SWEET ontologies. The user can use different parameters to discover clusters, which are appropriate to the focus of its query. For example, a user who needs data on a phenomenon (e.g., soil moisture) can use phenomenon-observed-based cluster discovery. A user who searches for sensor services to fulfill a task (e.g., drought risk assessment) can use intended-application-based sensor discovery. Existing discovery approaches dot not propose these different perspectives. Figure 6 shows the proposed interface for visualizing the discovered clusters and the sensor services that compose them on the GeoCens platform.



Select parameter: intended application

CLUSTERS RESULTING OF THE PARTITIONING PROCESS

- ALL SENSOR CLUSTERS
  - MonitorPollution
    - http://app.geocens.ca:8210/sos by Environment Canada
    - http://app.geocens.ca:8210/sos by Environment Canada
    - http://app.geocens.ca:8210/sos by Environment Canada
    - http://app.geocens.ca:8210/sos by Environment Canada
    - http://app.geocens.ca:8210/sos by Environment Canada
    - http://app.geocens.ca:8210/sos by Environment Canada
    - http://app.geocens.ca:8217/sos by Environment Canada
    - http://app.geocens.ca:8206/sos by Environment Canada
    - http://app.geocens.ca:8206/sos by Environment Canada
  - AssessGlobalWarmingImpact
  - MonitorDroughtRisk
  - MonitorWeather
  - AssessFloodingRisk
  - ToxicGasPlumeMonitoring

PARAMETERS OF THE CLUSTER/SENSOR SERVICE

Application domain: http://sweet.jpl.nasa.gov/2.2/phenAtmo.owl#AtmosphericPhenomena
Intended application: http://sweet.jpl.nasa.gov/2.2/human.owl#MonitorAirPollution
Phenomenon observed: http://sweet.jpl.nasa.gov/2.2/phenEnvirImpact.owl#AirPollution
Observed property: http://sweet.jpl.nasa.gov/2.2/matr.owl#GreenHouseGasEmission
Area of measurement: http://sws.geonames.org/6251999/

snr_105604_ozone
ground_level_ozone = 40.349 [ppb] @ 2009-01-01T07:59:59Z

Fig. 6. Interface for visualization of sensor clusters

This interface allows seeing the location and description of the sensors in a selected cluster. Consider that the user searches for ground water elevation in the context of drought response in the area of Edmonton. This query is formalized as:

Sensor(?x)∧PhenomenonObserved(?y)∧ObservedProperty(?z)∧hasPhenomenonObserved (?x, ?y)∧hasObservedProperty(?y, ?z)∧IntendedApplication(?i)∧hasIntendedApplication (?x, ?i)∧AreaOfMeasurement(?a)∧near(?z, ?a)∧SameAs(?y, GroundWater)∧SameAs(?z, GroundWaterElevation)∧SameAs(?a, Edmonton) ∧SameAs(?i, DroughtResponse)
→ sqwrl:select(?x)

The Jess-based SQWRL query processing engine is used to find the sensor clusters that satisfy the query. Figure 7 gives an example of discovered cluster.

SensorCluster:	SensorCluster_001
IntendedApplication:	MonitorDroughtRisk
ApplicationDomain:	AtmosphericPhenomena, Soil, Meteorology
PhenomenaObserved:	ContinentalLithosphere, GroundWater, Meteorological-Phenomena, Precipitation, RadiationalHeating
ObservedProperty:	AirProperty, PrecipitationProperty, RadiationalProperty, SoilProperty, WaterProperty, WindProperty
AreaOfMeasurement:	Canada
ObservationPeriod:	1970-November 2011

Fig. 7. Example of retrieved cluster

Then, the query is processed against the sensor descriptions that compose the discovered cluster(s) (Fig. 8). The resolution of semantic heterogeneities between the query and the sensor descriptions is resolved in part because of the use of the SRF and the query expansion.

SensorService:	http://app.geocens.ca:8189/sos
IntendedApplication:	MeasureGroundWaterLevel
ApplicationDomain:	Hydrology
PhenomenaObserved:	GroundWater
ObservedProperty:	GroundWaterElevation
AreaOfMeasurement:	Alberta, Saskatchewan, Manitoba
ObservationPeriod:	October 2011

Fig. 8. Example of retrieved sensor service

The advantage of this approach is that the query does not have to be processed against every sensor description. To demonstrate this advantage, we have compared the performance of the discovering system when no clusters are formed with its performance when the clusters are formed. The selected evaluation approach is based on an authoritative result: a set of human-determined, expected results is built; the results produced by the algorithm should be as close as possible to this authoritative result. We have built an authoritative result for five queries, i.e., the set of sensor services that should be retrieved for each query. Then, we have measured the recall in function of the percentage of nodes being queried ("node" designates both individual sensor services and clusters). The recall is the ratio of the number of relevant nodes being retrieved with respect to the total number of relevant nodes. Figure 9 shows the results of the experimentation with 122 nodes.

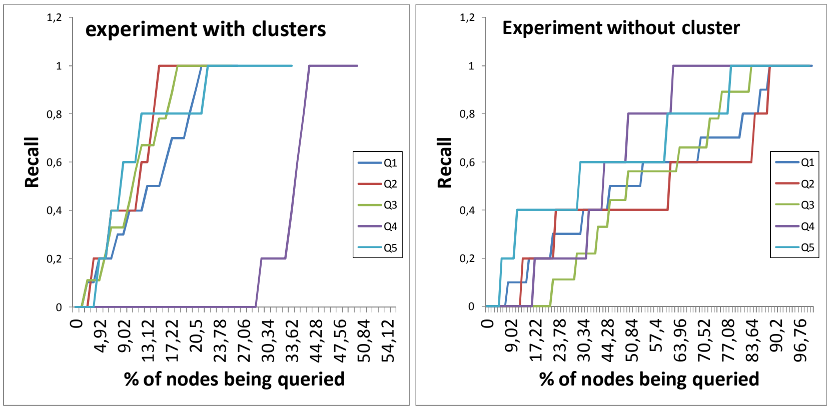


Fig. 9. Recall of the discovery system with and without clusters

When no clusters are formed, a significant number of nodes must be queried before the recall reaches 1(e.g., for query 1 (Q1), the recall reaches 1 when the percentage of queried nodes is about 77%). In comparison, when clusters are formed, the recall reaches 1 for Q1 when about 22% of the nodes are being queried. Therefore, the partitioning significantly reduces the time to retrieve relevant nodes. In the approach with clusters, not all nodes need to be queried (e.g., in Q1, the query processing was complete when percentage of queried nodes was 30%). However, Figure 9 shows that for Q4, a significant number of nodes were queried before the recall started increasing. This is because Q4 was matched with a relatively large cluster containing no relevant node. To improve the performance of the partitioning algorithm in this regard, further research is needed to increase its ability to find nested clusters.

6 Conclusion and Perspectives

The issue of discovering relevant sensors services is crucial because of the growing volume of sensor data. The issue is still not resolved due to the semantic heterogeneities between sensor descriptions and poor sensor descriptions. We have proposed an approach that supports the discovery of relevant sensor services based on three main contributions: a sensor metadata model, a partitioning algorithm and a SQWRL query processing approach. The proposed metadata model includes several parameters that are usually not included in SensorML descriptions, but that support different search perspectives, such as searching based on intended application, phenomenon observed, application domain, etc. The partitioning algorithm uses network analysis to discover meaningful sensor clusters and experiments demonstrated that it can improve the performance of discovery mechanisms when a large volume of sensor services is available. We also considered that in future work, avenues to improve the ability of the partitioning algorithm to find nested clusters could be further investigated. Finally, we have experimented successfully the use of the SQWRL language to support query processing over sensor clusters and sensor descriptions. However, to resolve more semantic heterogeneities, we had to integrate the use of a common semantic reference frame and develop a query expansion algorithm. We noted that such an approach depends on the completeness of the semantic reference frame. Also, in future work, we plan to investigate techniques to deal with the possible blow-up of expanded queries. In addition, we are currently investigating how sensor descriptions can be automatically generated and referenced to the semantic reference frame with automatic semantic extraction tools.

Acknowledgment. This research was made possible by an operating grant from Microsoft Research and Alberta Innovate Technology Future.

References

1. Knoechel, B., Huang, C.-Y., Liang, S.: Design and Implementation of a System for the Improved Searching and Accessing of Real-World SOS Services. In: Proceedings of Sensor Web Enablement, SWE 2011 (2011)

2. Sheth, A., Henson, C., Sahoo, S.: Semantic Sensor Web. IEEE Internet Computing, 78–83 (2008)
3. Bröring, A., Echterhoff, J., Jirka, S., Simonis, I., Everding, T., Stasch, C., Liang, S., Lemmens, R.: New Generation Sensor Web Enablement. Sensors 11, 2652–2699 (2011)
4. Bröring, A., Janowicz, K., Stasch, C., Kuhn, W.: Semantic Challenges for Sensor Plug and Play. In: Carswell, J., Fotheringham, A., McArdle, G. (eds.) W2GIS 2009. LNCS, vol. 5886, pp. 72–86. Springer, Heidelberg (2009)
5. Henson, C.A., Pschorr, J.K., Sheth, A.P., Thirunayaran, K.: SemSOS: Semantic Sensor Observation Service. In: Proceedings of the 2009 International Synposium on Collaborative Technologies and Systems (CTS 2009), Baltimore, MD (2009)
6. Jirka, S., Bröring, A., Stasch, C.: Discovery Mechanisms for the Sensor Web. Sensors 9(4), 2661–2681 (2009)
7. Botts, M., et al.: OGC Sensor Web Enablement: Overview and High Level Architecture (OGC 07-165), Open Geospatial Consortium with paper (2007)
8. Pschorr, J., Henson, C., Patni, H., Sheth A.P.: Sensor Discovery on Linked Data. Knoesis Center Technical Report (2010)
9. Compton, M., Henson, C., Neuhaus, H., Lefort, L., Sheth, A.: A Survey of the Se-mantic Specification of Sensors. In: Taylor, K., Ayyagari, A., de Roure, D. (eds.) Proceedings of the 2nd International Workshop on Semantic Sensor Network (SSN 2009), vol. 552, pp. 17–32 (2009)
10. Devaraju, A., Neuhaus, H., Janowicz, K., Compton, M.: Combining Process and Sensor Ontologies to Support Geo-Sensor Data Retrieval. In: 6th International Conference on Geographic Information Science (GIScience 2010), Zurich, Switzerland (2010)
11. Bittner, T., Donnelly, M., Winter, S.: Ontology and Semantic Interoperability. In: Zlatanova, S., Prosperi, D. (eds.) Large Scale 3D Data Integration: Challenges and Opportunities. CRC Taylor and Francis (2008)
12. Kuhn, W., Raubal, M.: Implementing Semantic Reference Systems. In: Gould, M., Laurini, R., Coulondre, S. (eds.) AGILE 2003 - 6th AGILE Conference on Geographic Information Science, Collection des Sciences Appliquées de l' INSA de Lyon, pp. 63–72. Presses Polytechniques et Universitaires Romandes, Lyon (2003)
13. Freeman, L.C.: The Development of Social Network Analysis: a Study in the Sociology of Science. BookSurge Publishing (2004)
14. Hoser, B., Hotho, A., Jäschke, R., Schmitz, C., Stumme, G.: Semantic Network Analysis of Ontologies. In: Sure, Y., Domingue, J. (eds.) ESWC 2006. LNCS, vol. 4011, pp. 514–529. Springer, Heidelberg (2006)
15. O'Connor, M., Das, A.: SQWRL: A Query Language for OWL. In: Hoekstra, R., Patel-Schneider, P.F. (eds.) Proceedings of OWL: Experiences and Directions, Chantilly, Virginia, USA (2009)
16. Lutz, M., Kolas, D.: Rule-based Discovery in Spatial Data Infrastructure. Transactions in GIS 11(3), 317–336 (2007)
17. Bakillah, M., Mostafavi, M.A.: Semantic Augmentation of Geospatial Concepts: the Multi-View Augmented Concept to Improve Semantic Interoperability Between Multiple Geospatial Databases. In: Joint International Conference on Theory, Data Handling and Modelling in GeoSpatial Information Science, Hong Kong, May 26-28 (2010)
18. Eriksson, H.: Using JessTab to Integrate Protégé and Jess. IEEE Intelligent Systems 18, 43–50 (2004)

Leveraging Power of Transmission for Range-Free Localization of Tiny Sensors

Marcello Cinque[1], Christian Esposito[1,2], and Flavio Frattini[1]

[1] Dipartimento di Informatica e Sistemistica (DIS)
Università di Napoli Federico II, Napoli, 80125, Italy
[2] Institute of High Performance Computing and Networking (ICAR)
National Research Council, Napoli 80131, Italy
{macinque,christian.esposito,flavio.frattini}@unina.it

Abstract. Modern environmental monitoring systems are being envisioned as constellations of tiny smart sensors, densely deployed on the territory to be monitored. Data provided by such sensors is meaningless if it is not equipped with information about the context to which it refers. Such a context is mainly related to the location of sensors, which allows to manage collected data by means of Geographic Information Systems. Due to scarce resources of tiny sensors characteristics, localization services are usually range-free and based on simple measurements provided by the majority of radio chips, such as the Received Signal Strength Indicator (RSSI). However, existing solutions suffer of inaccuracy issues, due to the unreliability of the adopted indicators. This article proposes a range-free localization algorithm based on data fusion, which combines the RSSI with the Power of Transmission (PoT), by estimating the RSSI at different PoT levels. We show, through experimentation on real tiny devices, how this simple solution allows reducing the localization error to about 16% with respect to state-of-art algorithms.

Keywords: Wireless sensor network, Localization, Received Signal Strength Indicator (RSSI), Power of Transmission (PoT).

1 Introduction

Collecting geo-spatial information is one of the main activities of a *Geographic Information System* (GIS), which is a system designed to acquire, store, process and visualize all types of geographically referenced data [1]. Depending on the application scenario, such systems may require the real-time collection of data directly from the territory being monitored. Nowadays, this collection task is being more and more realized by means of a massive number of interconnected low-cost, low-power and multi-functional sensors. Such sensors are (*i*) characterized by a small size, (*ii*) capable to communicate in short distances, and (*iii*) part of what is known in the literature as Wireless Sensor Networks (WSNs) [2]. WSNs are facing an increasing interest, since they allow reducing installation and management cost of environment monitoring systems. Our past experience on the use of WSNs on real environmental monitoring systems [3] (such as, the control of the Ponte Liscione's Dam, and the monitoring of a ridge subject to land-slides near the station of a commuter railway, called Circumvesuviana) has demonstrated that localizing sensors within a given WSN is a crucial requirement in real scenarios. Estimating the position of sensors simplifies the geographical representation of the data, and hence the integration in GISes, and it is particularly required when monitoring given location-dependent phenomena, such as slow movements of a ridge susceptible to land-slides.

To better comprehend why it is necessary to have a good understanding of the location of sensors in a WSN, let us consider a real world example: a WSN is deployed for rural and forest fire detection [4]. Specifically, sensors may be dropped from an aircraft since the high geographical extension of the area of interest makes unfeasible to manually place them, apart from some anchor nodes ad-hoc positioned. Parachuted sensors can measure any significant change of the temperature and humidity, which determines the presence of fire. When fire is detected, an alarm can be sent to a central server, so to alert the fire department that reach the interested area. It is crucial that the alarm contains a good estimation of the location of the sensor; otherwise, fire fighters only know that fire has occurred in the monitored area, but they do not know where to operate. This example also demonstrates the problem caused by inaccuracy in the location estimation. Specifically, if the sensor location is incorrect, the fire fighters cannot reach on time the correct location of the fire (reached only after several tentatives), which has time to propagate.

Localization has a long history of approaches proposed by both academia and industry. Some of them have become mature technologies commonly used in our daily life, such as GPS. However, literature on localization cannot be applied to the context of WSNs, since traditional localization algorithms have not been designed specifically for cheap and battery dependent devices. A localization algorithm results suitable to be executed within a WSN if it focuses on two key principles: minimize the battery consumption, and avoid the use of extra hardware. For these reasons, several new algorithms, more tailored on WSNs, have been proposed. In most of the cases, they are based on the Received Signal Strength Indicator (RSSI), since this indicator is usually provided for free in almost all the radio chips adopted today. In particular, one of the simplest and most popular RSSI-based localization algorithms for WSNs is *ROCRSSI* (Ring Overlapping based on Comparison of RSSI) [5], which is based on the comparison of RSSI values estimated between the node to locate (unknown node) and a set of "anchor nodes (also defined as beacon nodes), and those between the beacons themselves. However, it is known that the location estimation based on RSSI may suffer of inaccuracy issues, due to multi-path fading and shadowing phenomena [6]. In particular, our experiments on ROCRSSI over real sensor devices highlight an average localization error of about 44% in indoor environments (such as the internals of a dam), with three beacons, which leads to a quite useless location estimate [7].

To alleviate this issue, a winning solution is to combine several measures or jointly adopting different localization techniques. However, the use of different techniques may require the adoption of extra hardware, which is hardly discouraged. Moved by these observations, we propose a novel localization algorithm based on data fusion, which combines the RSSI estimate with the Power of Transmission (PoT) of beacon nodes. The main idea is to evaluate the RSSI by using different PoT levels. This is based on the intuition that beacon nodes sensed with high RSSI and low PoT are more likely to be close to the unknown node. Experiments conducted on real world tiny mobile devices show that the proposed solution is able to half the localization error of ROCRSSI to about 16%, which corresponds to 125 cm in a 41 square meters environment.

2 Localization Techniques

Methods for localization can be broadly divided in two distinct groups: the ones that make use of dedicated infrastructures, and the ones that use non-dedicated infrastructures. Specifically, methods belonging to the first class use particular technologies, such as infrared [8] or ultrasound signals [9], that are specifically deployed for positioning concerns. On the other hand, methods that are not based on dedicated infrastructures adopt standard wireless networking technologies based on Radio Frequency (RF) signals, *i.e.*, Bluetooth (BT), WiFi, or RFID, which have been designed not only for positioning concerns. Systems with dedicated infrastructures grant high positioning quality at the expenses of a too strong financial investment. Therefore, it is more convenient the use of technologies that perform other duties jointly to positioning ones. Traditional localization methods cannot be applied for treating localization issues in WSNs, since they do not properly address the peculiar requirements that WSNs impose on the adopted localization service, as we have discussed in [7]. Moreover, there are two main classes of approaches for location estimation: *centralized*, *i.e.*, only one node in the network is in charge of estimating user device position, and *distributed*, *i.e.*, each device takes care of computing its own position. In WSNs, it is unsuitable to apply centralized approaches due to the high number of deployed sensors; therefore, all approaches have the characteristic of being distributed.

Localization algorithms in WSNs can be classified as *range-based* and *range-free*. The former ones are based on the estimation of the range between sensors and beacons, in terms of distances and angles, and use measurements such as Time of Arrival (ToA) [10], Time Difference of Arrival (TDoA) [11], RSSI [12], or Angle of Arrival (AoA) [13]. Specifically, TOA is measured by using the Global Positioning System (GPS) [14]: the reference sensor sends to the one to be located a message with its exact sending time and its position obtained from GPS. The receiver can compute a pseudo-range centered at the reference sensor, and the location can be obtained by interpolating the pseudo ranges from several reference sensors. AHLos [15] adopts TDoA measures based on ultrasound technology, in combination with a sensing approach based on RSSI. APS [13] is a system based on AoA with sensors equipped with an antenna array. Although such systems are characterized by good localization accuracy, they require additional hardware to precisely measure distances and angles. This violates the requirements of localization in WSN [7]. Range-based algorithms are part of the previously mentioned first class, since they make use of dedicated technologies.

Range-free algorithms [16] are able to estimate locations without any information on distances and angles, but with measures of the connectivity from sensors to beacons, *e.g.*, by using the strength of signals received from known sources. Range-free algorithms can be further classified into anchor-based and anchor-free algorithms [17]. The former assume that there are nodes with knowledge about their position; anchor-free, instead, do not require special nodes for localization. Some concrete examples of anchor-based approaches are the centroid scheme in [18], where the location is computed as the centroid among a set of anchors, SeRLoc [19], where anchors are equipped with directional sectored antennas and sensor location is estimated by intersecting the radio range of the anchors, and MDS-MAP [20], which is based on the data analysis technique named Multi-Dimensional Scaling (MDS). Some concrete examples of anchor-free approaches are Spotlight [21], where location is inferred using the time when an event (*e.g.*, light in the area) is perceived by a sensor node and the spatio-temporal properties of the generated events, and Walking GPS [22], where a special node is equipped with a GPS component which broadcast its position, and all other sensors determine their position from the broadcasted location. Range-free algorithms mainly adopt RF technologies, therefore, they belong to the previously mentioned second class. For this reason, range-free algorithms do not generally need any additional hardware, so they represent a cost-effective alternative, especially in WSNs. However, they are characterized by lower accuracy and higher communication overhead, if compared to range-based solutions. So, further research is needed in order to make them be effectively used in real world scenarios.

In our previous work presented in [7], we have summarized the main characteristics of the most adopted localization algorithms ad-hoc for WSNs and concluded that ROCRSSI provides the best trade-off in terms of complexity, extra hardware, and accuracy. Therefore, we decided to use it as a starting point to introduce a novel localization algorithm able to improve the accuracy without increasing costs, complexity, and power consumption.

The current challenge in the research on localization is how improving the accuracy of RF positioning methods. The preferred approach is to use data fusion, *i.e.*, combining outputs of several position estimators so to obtain a more accurate location estimation. In our previous work [7], we have characterized a positioning system as a layered architecture composed of (*i*) technology, *i.e.*, dedicated or not technology used for obtaining measures upon which location is estimated, (*ii*) method, *i.e.*, the features of the workspace measured through adopted technology, and (*iii*) technique, *i.e.*, algorithms that transform raw measures into canonical position information. In localization systems, we can apply data fusion at different layers of the localization stack:

– a localization system can combine heterogeneous technologies, by using always the same positioning method [23];
– location can be obtained by fusing outputs from several estimators, each adopting a particular positioning method or technique on measures obtained by a certain technology [24].

In WSNs, the first solution is not feasible since the sensors typically used to realize WSNs supports only one RF technology. For instance, in [25] only ZigBee is provided. On the other hand, using more than one positioning method may increase the complexity of the overall location estimation process. Moreover, it can also introduce the issue that outputs of the different estimators are not comparable, *e.g.*, the output of a location estimator that uses proximity is different to the one that uses distances. Therefore, prior to apply fusion, a conversion action is required, which implies a further increase in complexity that may negatively affect the performance of the system in terms of latency, and resource consumption. Based on these considerations, our approach is to use only one positioning technology, *i.e.*, ZigBee provided by our sensors, and one positioning method, *i.e.*, ROCRSSI. Data fusion is applied on the outputs of our ROCRSSI estimators that receive different kinds of measures from ZigBee: our driving idea is that we can vary the *Power of Transmission* (PoT) when generating RF signals, and location computed at different PoT levels can be combined to achieve more accurate estimations. In the current literature, adjustments in the applied PoT level are only used for optimizing the energy consumption of the radio chip [26].

3 From APIT to ROCRSSI++

Since RSSI is typically provided by all RF technologies, such as Bluetooth, WiFi, and ZigBee, and it does not require special hardware components, algorithms that use it are widely used in the literature. The driving idea of all RSSI-based localization methods is that RSSI has a decreasing trend with respect to the distance, *i.e.* higher is the distance between two sensors, lower is the RSSI value measured on exchanged signals. So that, estimating a user location is possible by inferring relative distances between sensors based on experienced RSSI values. For instance, if sensor A receives signals from sensor B with an $RSSI_{AB}$ greater than the one relative to signals received from C, namely $RSSI_{AC}$, it is reasonable to suppose that B is closer to A than C.

3.1 APIT and ROCRSSI

In 2003, He et al. proposed APIT [27] (Approximate Point-In-Triangulation Test) as one of the first range-free localization algorithm based on such a principle. In this algorithm, nodes are distinguished in beacons and unknown nodes. The former ones have a known position, while unknown nodes are the ones for which the position is to be estimated by using the location of beacons, RSSI values among beacons and RSSI measures between the unknown sensor and the beacons.

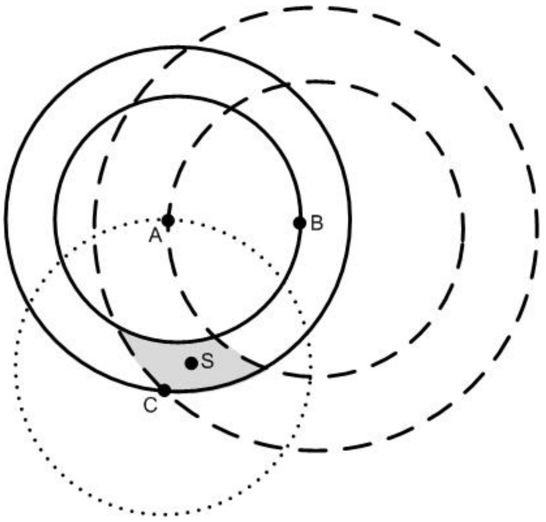


Fig. 1. Localization using ROCRSSI

An unknown node selects three beacon sensors among the nearest ones (*i.e.*, we mean nodes that are close enough to establish a communication channel) and uses RSSI to estimate if it is inside the triangle whose vertices are the three beacons. The estimation is repeated with different reachable beacons combinations until all combinations are exhausted or the required accuracy is achieved. Last, the estimated position of the unknown node is the center of gravity of the intersection of all the identified triangles.

To get more accurate estimations and reduce communication overhead, in 2004 Liu et al. proposed Ring Overlapping based on Comparison of RSSI (ROCRSSI) [5]. It is a collaborative ad-hoc localization algorithm based on signal strength. To explain how the algorithm works, let us consider the concrete example shown in Figure 1.

A, B and C are three beacon nodes, while S is an unknown sensor. By exchanging messages, all the sensors have estimations of the RSSI values between themselves and the nearest sensors. Furthermore, let us consider that the following inequalities are verified:

$$RSSI_{AC} < RSSI_{SA} < RSSI_{AB} \tag{1}$$

$$RSSI_{BC} < RSSI_{SB} < RSSI_{AB} \tag{2}$$

$$RSSI_{AC} < RSSI_{SC} \tag{3}$$

Equation 1 implies that the unknown node estimates to be in the ring (drawn with unbroken line in Figure 1) with center A, inner radius equal to the distance between sensors A and B and outer radius equal to the distance between sensors A and C. Equation 2, instead, determines another ring (drawn with a dashed line in figure) centered in sensor B, with inner radius equal to the distance between sensors B and A and outer radius equal to the distance between sensors B and C. Equation 3 indicates that S is placed within a circle (drawn with dotted line in figure) centered in sensor C and with radius equal to the distance between sensors C and A. The unknown node S is located in the center of gravity of the intersection area of the rings and the circle.

3.2 ROCRSSI+

In 2006 Crepaldi et al. proposed a new version of ROCRSSI, namely ROCRSSI+ [28], in order to solve a drawback of such an algorithm. In fact, ROCRSSI assumes that sensors are located inside an area delimited by beacons, as in the case of Figure 1. However, an unknown sensor may also be located outside the area delimited by beacons; subsequently, the RSSI values between the unknown sensor and each beacon are lower than the RSSI values between two given beacons, achieving inequities like the following one:

$$RSSI_{SC} < RSSI_{AC} \tag{4}$$

ROCRSSI+ also considers RSSI values assigned to sensors whose range does not contain the unknown node. This allows achieving a greater accuracy then basic ROCRSSI.

3.3 ROCRSSI++

In our previous work of 2011, we showed that also ROCRSSI+ exhibits some drawbacks inherited by ROCRSSI, which caused by *inconsistency*, *variability of RSSI*, *channels asymmetry*, and *memory and communication inefficiency* [7]. The problems and their solutions are briefly shown in the following.

Inconsistency. An algorithm is *consistent* if it returns the same result regardless of the order in which inputs are processed. ROCRSSI and ROCRSSI+ are inconsistent since the localization depends on the order in which RSSI values are considered. This happens when two channels with equal RSSI values correspond to different distances. Indeed, in such cases, the radius of a circle in which an unknown sensor is located may depend on the order in which RSSI values are compared. In order to overcome this inefficiency, in such cases the RSSI value estimated on the longest distance is considered. In fact, RSSI values depend on the attenuation faced by signals on their path from emitter to receiver, which is function of the distance, but also of possible obstacles. So, if same RSSI values are related to beacons at distinct distances, there must be more obstacles between beacons at lower distance, than the ones at higher distance. The RSSI with highest distance is surely to be more dependent on distance and less on possible met obstacles.

Variability of RSSI. RSSI values are not stable but can vary over the time. In fact, the RSSI between two nodes may be different in different times. Reflection, refraction and interferences are the main causes of such fluctuations. Hence, the RSSI value of a couple of nodes is not computed with a single message, but considering the average of the RSSI measured within a succession of messages.

Channels Asymmetry. To simplify the evaluation of RSSI, in [5] communication channels are considered symmetric. However, we performed several experiments that showed that it is not true. Given two nodes, namely A and B, the strength of the signal emitted by A and received by B may be different from the strength of the signal emitted by B and received by A. To overcome this issue, it is necessary to solve a simple problem of consensus for each couple of nodes. Specifically, if the RSSI values evaluated by the two nodes of a communication channel are not equal, they suit to use the biggest one. This is due to observations similar to the ones about inconsistency.

Store RSSI Values. Inputs of the ROCRSSI algorithm are the RSSI values relative to pairs of beacon nodes, their distances, and RSSI values between the unknown node and each beacon. Therefore, such values are to be communicated and stored. Let us consider the case of a network with N beacon nodes. Each beacon needs a *N-by-N* matrix, where the element $a(i,j)$ is the RSSI value estimated by i about signals received by j. However, thanks to the achieved symmetry in RSSI values, element $a(i,j)$ is equal to $a(j,i)$. Furthermore, a node does not need to know the RSSI to itself. The same is for distance values of each couple of beacon nodes. As a result, the matrices used to store these data are strictly triangular, hence we propose to use packed storage matrices. That implies a memory saving of $50 \cdot (1 + \frac{1}{N}) \cdot 100$ % and to reduce the number of messages exchanged among nodes.

Algorithm 1 is the pseudo-code of ROCRSSI++. It requires as input parameters the RSSI values of each couple of beacons, the position of each beacon, also used to compute distances between beacons, and the RSSI between the unknown node and near beacons. For each beacon close to the unknown sensor, the algorithm performs the steps shown in 3.1 to build circles and rings; it also considers the solutions to the problems discussed above. The output is the estimated position of the unknown node.

4 Power-Based ROCRSSI

Despite the numerous improvements proposed in the literature, ROCRSSI++ still exhibits a low accuracy if compared to other range-based localization algorithms. That compromises its successful application in real case studies. As mentioned, localization quality can be improved by fusing outputs from several estimators, each using the same technology and method, but varying the measures used for the estimation.

Algorithm 1. Pseudo-code of the ROCRSSI++ algorithm

```
Input: $BeaRSSI$, $BeaPositions$, $BeaUknRSSI$
Output: $CoG$ : $Estimated\ position\ of\ the\ unknown\ node$
     $S$: unknown node
     $S_n$: set of beacons near $S$
     $A$: beacon in $S$
     $R \leftarrow \emptyset$
 5:  while $S_n$ has elements do
       $A \leftarrow S_n.nextElement()$
       $S_A \leftarrow$ set of beacons near $A$
       split $S_A$:
         $S_{A1} : \forall\ I \in S_{A1}, RSSI_{AI} \geq RSSI_{AS}$
10:      $S_{A2} : \forall\ J \in S_{A2}, RSSI_{AJ} < RSSI_{AS}$
       if $S_A == \emptyset$ then
         goto 6
       else
         $d_1 \leftarrow 0$
15:      $d_2 \leftarrow 0$
         if $S_{A1}! = \emptyset$ then
           $I \leftarrow$ element with maximum distance among ones with the smallest RSSI
           in $S_{A1}$
           $d_1 \leftarrow$ distance($I$, $A$)
         end if
20:      if $S_{A2}! = \emptyset$ then
           $J \leftarrow$ element with maximum distance among ones with the greatest RSSI
           in $S_{A2}$
           $d_2 \leftarrow$ distance($J$, $A$)
         end if
         if $d_1! = 0$ && $d_2! = 0$ && $d_1 == d_2$ then
25:        $d_1 \leftarrow 0$
         end if
         if $d_1! = 0$ && $d_2! = 0$ && $d_1 > d_2$ then
           swap($d_1$, $d_2$)
         end if
30:      if $d_1! = 0$ && $d_2! = 0$ then
           $r \leftarrow$ ring with center $A$, inner radius $d_1$, outer radius $d_2$
         end if
         if $d_1 == 0$ then
           $r \leftarrow$ circle with center $A$ and radius $d_2$
35:      end if
         if $d_2 == 0$ then
           $r \leftarrow$ exterior of circumference with center $A$ and radius $d_1$
         end if
         $R \leftarrow R + \{r\}$ {//insert $r$ in $R$}
40:    end if
     end while
     $Int \leftarrow$ intersection of areas in $R$
     $CoG \leftarrow$ center of gravity of $Int$
     return $CoG$
```

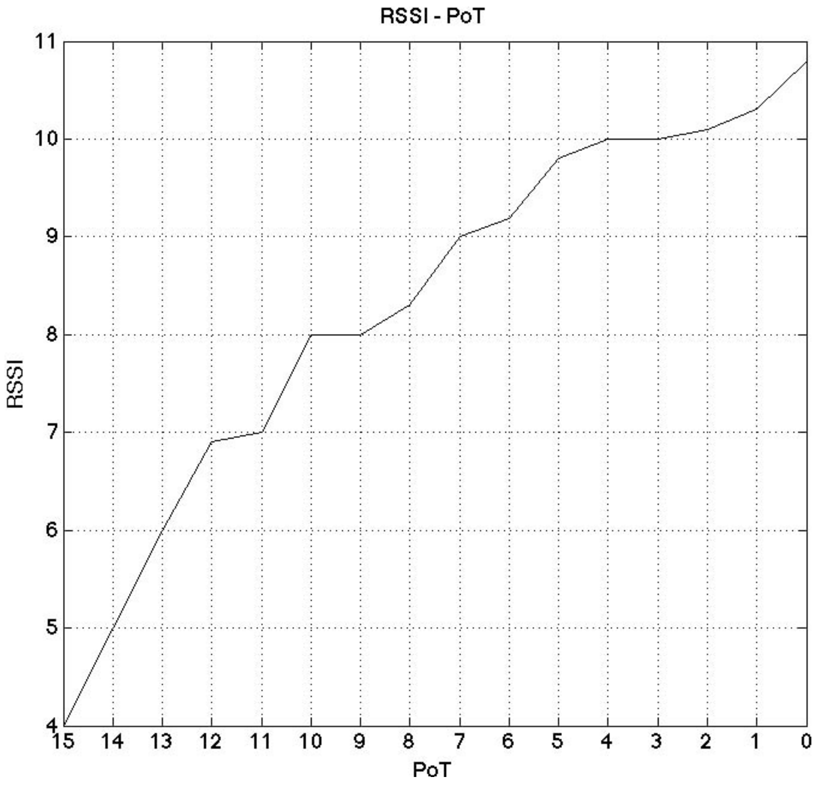


Fig. 2. Trend of the RSSI between two sensors about 5 meters apart when incrementing the power of the transmitted signal

Power-Based ROCRSSI (PB-ROCRSSI) is a range-free localization algorithm derived from ROCRSSI++. It is based on data fusion of parameters already available on radio chips. In particular, PB-ROCRSSI combines multiple RSSI values obtained by varying the Power of Transmission (PoT), so without requiring extra hardware and by keeping the same complexity level of its predecessors.

Figure 2 shows the trend of the RSSI of a signal with respect to the power at which it is emitted. When the PoT is equal to 0 (*i.e.*, the highest power of transmission), RSSI assumes its maximum value. Vice-versa, when PoT is equal to 15 (*i.e.*, the lowest power of transmission) RSSI assumes its minimum value.

PB-ROCRSSI algorithm exploits the increasing trend of RSSI with respect to the PoT. In fact, localization performed with a reduced power of transmission only uses closest beacon nodes that have shown to be more indicative of the actual location of the unknown node. To explain such a behavior, consider Figure 3. There are six beacons 1, 2, 3, 4, 5, 6 placed in cells (2;2), (9;2), (2;7), (9;7), (2;12), (9;12), respectively; sensor 7 is an unknown node placed in cell (x_7;y_7)=(5;12).

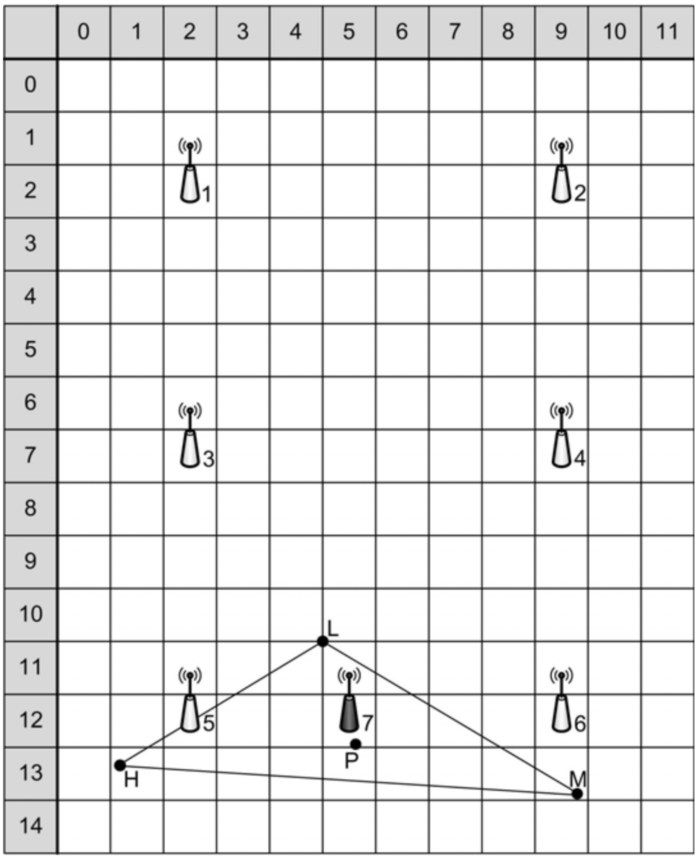


Fig. 3. A rectangular environment with six beacon nodes (1-6) and an unknown node (7). To simplify the localization, the environment is organized like the first quadrant of a Cartesian coordinate system; each cell is a square with the side of about 41.5 cm.

The percentage error is evaluated by comparing the absolute error (distance from estimated position to real position) to the maximum distance detectable in the environment (the diagonal when the environment has a rectangular shape). In the case shown in Figure 3 the diagonal is 17.80 (it is evaluated as the Eculidean distance from (0;0) to (11;14)), or 796.17 cm. When using ROCRSSI++ with signals transmitted at a low PoT level, *i.e.* 11, the unknown sensor 7 is localized at point L, with coordinates $(x_L;y_L)$=(4.50; 10.50); in such a case, the Euclidean distance is 1.58 (it is evaluated as $\sqrt{(x_7 - x_L)^2 + (y_7 - y_L)^2}$= $\sqrt{(5.00 - 4.50)^2 + (12.00 - 10.50)^2}$); the percentage error is $\frac{1.58}{17.80} \cdot 100$=8.88%, which corresponds to 70.70 cm in absolute (8.88% of the diagonal = 796.17 cm ·0.0888). When running the same algorithm at PoT level 0, the unknown node is localized in the point H (0.67; 12.83), obtaining an error equals to 24.78% (197.07 cm).

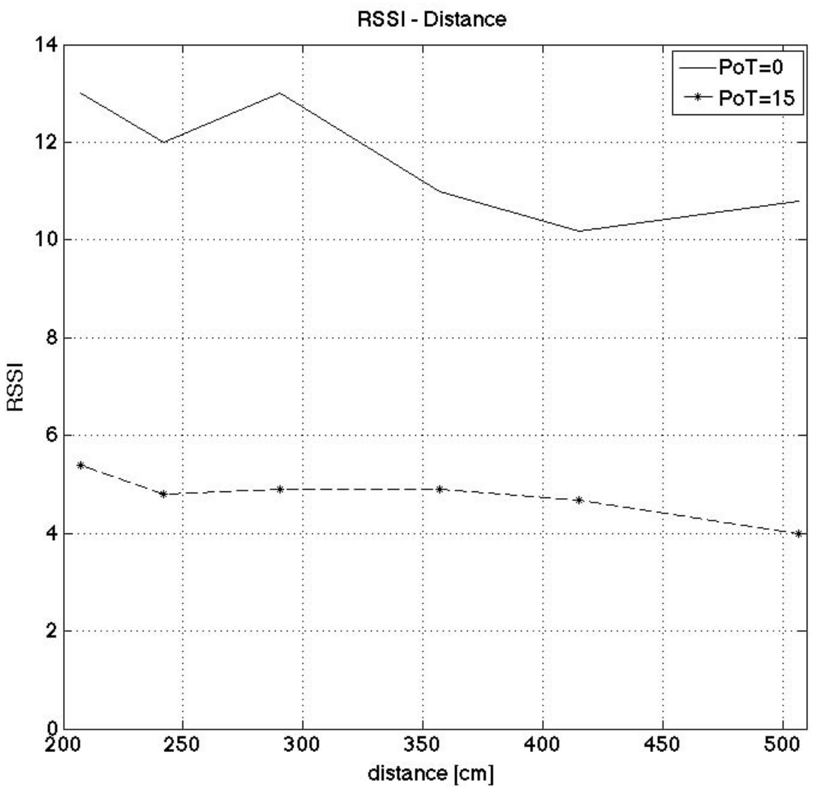


Fig. 4. Trend of the RSSI between two sensors when varying the distance at different PoT levels

Best performances achieved at lower PoT levels can be justified by the trend of RSSI values with respect to the distance when decreasing the power of the transmitted signal. As shown in Figure 4, RSSI estimates between distant sensors exhibit more fluctuations with respect to the ones between close sensors.

However, the lower the PoT, the lower the number of reachable nodes. Hence, when using low levels of PoT it is possible that less than 3 beacons are reachable by the unknown node. In this situation its position cannot be estimated.

Based on these considerations, we defined a simple localization algorithm based on data fusion, which consists in running ROCRSSI++ at three different PoT levels, low, medium and high, and in using the three estimated positions to compute the position of the unknown node.

Algorithm 2 shows the pseudo-code of ROCRSSI++. It requires the RSSI values between beacons and beacons and the unknown node as for ROCRSSI++, but at the three PoT levels. The position of the beacons is, obviously, the same at any PoT level.

Algorithm 2. Pseudo-code of PB-ROCRSSI

```
Input: $BeaRSSI_{PoT_{low}}$, $BeaRSSI_{PoT_{med}}$, $BeaRSSI_{PoT_{high}}$ $BeaPositions$,
    $BeaUknRSSI_{PoT_{low}}$, $BeaconsUknRSSI_{PoT_{med}}$, $BeaconsUknRSSI_{PoT_{high}}$
Output: $CoG$ : $Estimated\ position\ of\ the\ unknown\ node$
 1: $L \leftarrow$ ROCRSSI++($BeaRSSI_{PoT_{low}}$, $BeaPositions$, $BeaUknRSSI_{PoT_{low}}$)
 2: $M \leftarrow$ ROCRSSI++($BeaRSSI_{PoT_{med}}$, $BeaPositions$, $BeaUknRSSI_{PoT_{med}}$)
 3: $H \leftarrow$ ROCRSSI++($BeaRSSI_{PoT_{high}}$, $BeaPositions$, $BeaUknRSSI_{PoT_{high}}$)
 4: $CoG \leftarrow$ center of gravity of the triangle $LMH$
 5: return $CoG$
```

The main problem to be solved in the setup phase of the network is the choice of the three PoT levels. PoT_{high} is the maximum one which guarantees to reach the maximum number of beacons. PoT_{medium} may be the central value of the scale of values. The choice of PoT_{low} is critical and strictly dependent on the environment. As shown, while low levels may have better performances, they also increase the number of unreachable beacons. Hence, a tuning of this parameter is needed, based on size of the considered environment.

In Figure 3 the points L, M, and H are the three localization estimations of the unknown node 7 with PoT levels equals to 0, 7, and 11, respectively. The center of gravity of the triangle LMH corresponds to point P, which is taken as the final estimation, corresponding to a localization error of 1.73% (12.75 cm in absolute).

To avoid increasing the cost of the algorithm, in terms of number of exchanged messages, we decided to make only one third of the RSSI measures required by a typical run of ROCRSSI+ per each given PoT level.

5 Observations and Comparison of Localization Algorithms

To test the described algorithms we performed several experiments with real devices in an indoor environment. Sensors used for experiments are Iris Motes by Crossbow Technology equipped with a ZigBee RF Transceiver and TinyOS 2.0 operating system [25]. The environment is a computer laboratory with the shape of a rectangle, whose dimensions are 497 cm as length and 622 cm as width, which corresponds to an area of about 31 square meters. In this environment there are no obstacles among sensors. Beacon nodes, numbered from 1 to 6, are placed like in Figure 3. We performed 15 experiments, each one with a different position of the unknown node and repeated 10 times. We implemented ROCRSSI, ROCRSSI+, and ROCRSSI++ apart from PB-ROCRSSI so to compare the performances of all the algorithms.

Experimental results about the localization error of the algorithms are in Figure 5. The figure shows the trend of the average error of the algorithms when varying the number of beacons in the network from 3 to 6. Localizations with ROCRSSI, ROCRSSI+, and ROCRSSI++, are performed with the maximum power of transmission. The percentage error is evaluated as described in Section 4.

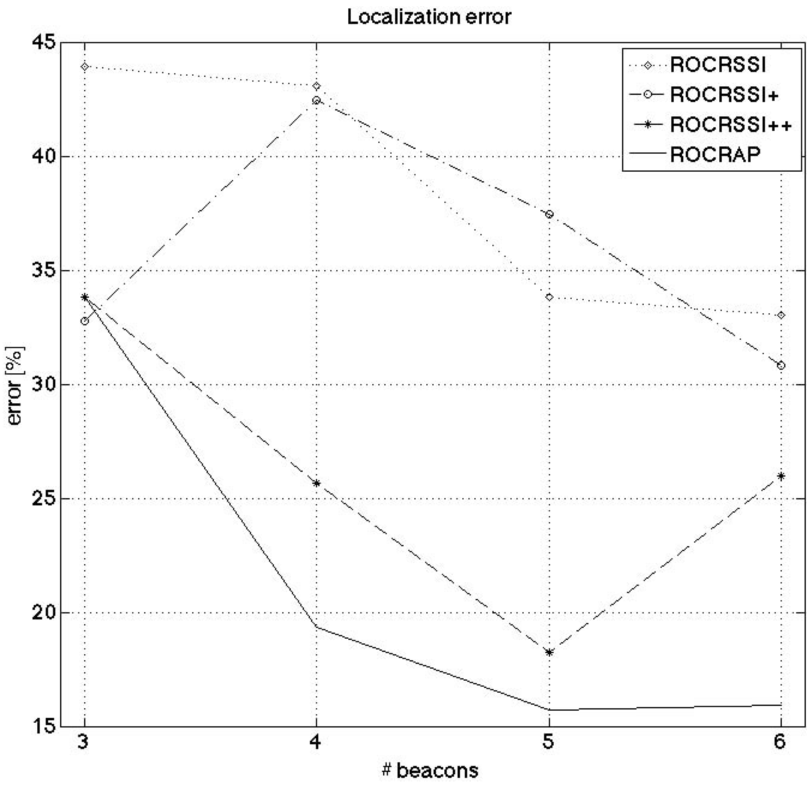


Fig. 5. The mean error of the illustrated algorithms when incrementing the number of beacon nodes in the network

Figure 5 shows that PB-ROCRSSI allows to achieve the best localization performances among the considered algorithms. The average error of PB-ROCRSSI is the lowest whatever is the number of beacon nodes in the network, except for 3 beacons, where the performance is comparable to the one of other solutions; in particular, when using at least four beacons, the localization error of PB-ROCRSSI is about half of localization error of ROCRSSI and ROCRSSI+.

Furthermore, we notice that usually the error decreases as the number of beacons increases. Then we can deduce that the localization accuracy of anchor-based algorithms depends on the number of beacons. In fact, when having more beacons, each unknown sensor could obtain more information useful for localization. In the case of ROCRSSI algorithms more information allows to build more rings and circles so that the intersection area would be smaller. Hence, the number of beacon nodes is a crucial parameter to be set in the setup phase. Moreover, the number of beacons cannot be chosen with the only scope to maximize the localization accuracy. Indeed, when the number of beacons is increased, also the number of messages to be exchanged among sensors grows. That implies the increase of battery consumption and time to localize, then we have to consider a trade-off among localization error, battery consumption and time to localize.

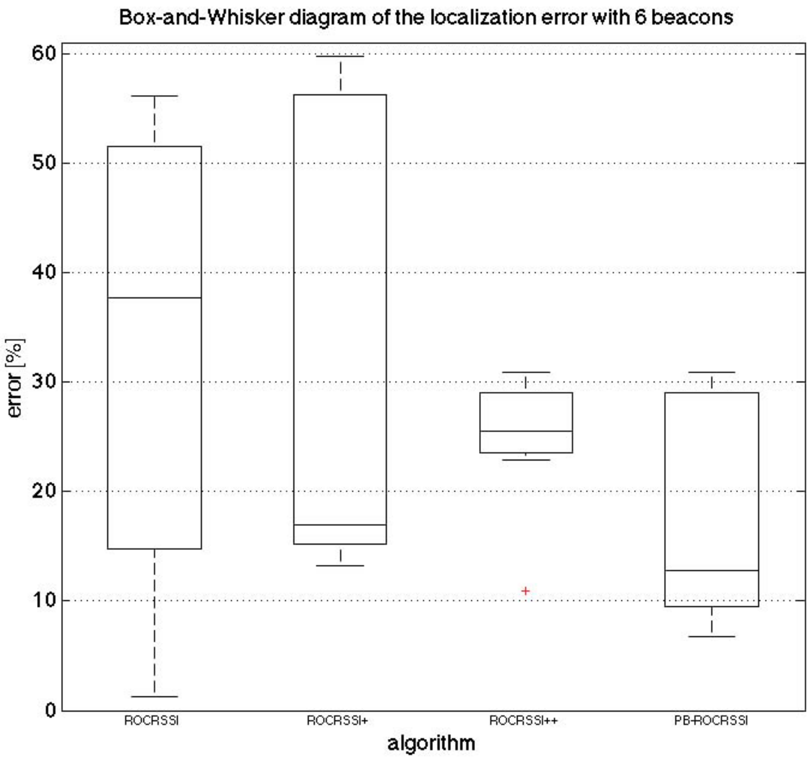


Fig. 6. Box-and-Whisker diagram (or Box-plot) of the localization error of the four algorithms when using 6 beacons

Figure 6 shows the Box-and-Whisker diagram (also known as Box-plot) of the error of the algorithms when performing localization in a 31 square meters environment with 6 beacon sensors. The ends of the whiskers are the maximum and minimum localization error of each algorithm; the top of the boxes is the upper quartile (75th percentile), while the bottom is the lower quartile (25th percentile); the line in the boxes is the median (50th percentile). The mean value is the center of the box. The symbol "+" represents an outlier of the data relative to the error of the ROCRSSI++ algorithm.

The plot shows that, apart from the mean, also the variation is greater for ROCRSSI and ROCRSSI+ algorithms. The latter algorithm also has a very low median that is closer to the lower quartile and very different by the mean value. This means that most of values are greater than the median. Such a phenomena is referenced in the literature as *negative skewness value*. On the other hand, ROCRSSI++ exhibits a lower variance than PB-ROCRSSI, but the maximum and the upper quartile of the two distributions are almost the same. The median of the errors relative to PB-ROCRSSI is the lowest one. However, also in this case, such a median is different by the mean and is closer to the 25th percentile; then, the bulk of the values is greater than the mean.

6 Conclusions and Future Work

This paper has proposed the empirical comparison of a family of range-free, anchor-based localization algorithm for Wireless Sensor Networks, an enabling technology for the timely collection of data in Geographic Information Systems. Furthermore, it has highlighted positive and negative aspects of such algorithms and shown how to achieve better localization performances. This target has been reached using a data-fusion approach and observing the strict requirements of WSNs about costs, power consumption, and time required for localization. Specifically, we have improved the localization estimates of the ROCRSSI algorithm exploiting the trend of the signal strength when varying the power of transmission. The proposed algorithm is able to locate a sensor with a precision of about 125 cm in a 31 square meters environment when using 6 beacons, outperforming its predecessors. Furthermore, previous experiments with ROCRSSI++ [7] showed that outdoor localization is usually better than the indoor one. Also the dimension of the environment is not a restriction for the algorithm if the number of beacon sensors is suitably incremented. However, experimental results also show that the PoT level for a good localization is strictly reliant on the distance of the nodes and on the dimension of the environment. Hence, in the next future we will investigate how to find the most suitable PoT levels for a given environment. Further experiments in an outdoor large environment will be also performed, in order to evaluate location estimations in a situation more similar to a real one.

References

1. Longley, P.A., Goodchild, M.F., Maguir, D.J.: Geographic Information Systems & Science, 3rd edn. John Wiley & Sons (August 2010)
2. Akyildiz, I.F., Melodia, T., Chowdhury, K.R.: A Survey on Wireless Multimedia Sensor Networks. Computer Networks 51, 921–960 (2007)
3. Di Martino, C., D'Avino, G., Testa, A.: iCAAS: An Interoperable and Configurable Architecture for Accessing Sensor Networks. International Journal of Adaptive, Resilient and Autonomic Systems (IJARAS) 1, 30–45 (2011)
4. Lloret, J., Garcia, M., Bri, D., Sendra, S.A.: Wireless Sensor Network Deployment for Rural and Forest Fire Detection and Verification. Sensors 9, 8722–8747 (2009)
5. Liu, C., Wu, K., He, T.: Sensor Localization with Ring Overlapping based on Comparison of Received Signal Strength Indicator. In: Proceedings of the IEEE International Conference on Mobile Ad-hoc and Sensor Systems, pp. 516–518 (2004)
6. Gu, Y., Lo, A., Niemegeers, I.: A Survey of Indoor Positioning Systems for Wireless Personal Networks. IEEE Communications Surveys & Tutorials 11(1), 13–32 (2009)
7. Frattini, F., Esposito, C., Russo, S.: ROCRSSI++: an Efficient Localization Algorithm for Wireless Sensor Networks. International Journal of Adaptive, Resilient and Autonomic Systems (IJARAS) 2(2), 51–70 (2011)
8. Aitenbichler, E., Mhlhuser, M.: An IR Local Positioning System for Smart Items and Devices. In: Proceedings of the 23rd IEEE International Conference on Distributed Computing Systems Workshops (IWSAWC 2003), pp. 334–339 (May 2003)

9. Want, R., Hopper, A., Falcao, V., Gibbons, J.: The Active Badge Location System. ACM Transactions on Information Systems 10(1), 91–102 (1992)
10. Guvenc, I., Sahinoglu, Z.: Threshold-Based TOA Estimation for Impulse Radio UWB Systems. In: Proceedings of the IEEE International Conference on Ultra-Wideband, pp. 420–425 (September 2005)
11. Wei, X., Wang, L., Wan, J.: A New Localization Technique Based on Network TDOA Information. In: Proceedings of the IEEE International Conference on ITS Telecommunications, pp. 127–130 (June 2006)
12. Hatami, A., Pahlavan, K., Heidari, M., Akgul, F.: On RSS and TOA Based Indoor Geolocation A Comparative Performance Evaluation. In: Proceedings of the IEEE Wireless Communications and Networking Conference, pp. 2267–2272 (April 2006)
13. Niculescu, D., Nath, B.: Ad Hoc Positioning System (APS) Using AOA. In: Proceedings of IEEE INFOCOM, pp. 1734–1743 (April 2003)
14. Wellenhoff, B.H., Lichtenegger, H., Collins, J.: Global Positions System: Theory and Practice. Springer, Heidelberg (1997)
15. Savvides, A., Han, C.C., Srivastava, M.B.: Dynamic Fine-Grained Localization in Ad-Hoc Networks of Sensors. In: Proceedings of MOBICOM (July 2001)
16. Stoleru, R., He, T., Stankovic, J.A.: Secure Localization and Time Synchronization for Wireless Sensor and Ad Hoc Networks. In: Poovendran, R., Wang, C., Roy, S. (eds.) Range-free Localization. Advances in Information Security Series, vol. 30. Springer, Heidelberg (2007)
17. Stolern, R., He, T., Stankovic, J.A.: Range-Free Localization. In: Sensor Network Handbook. Springer, Heidelberg (2006)
18. Bulusu, N., Heidemann, J., Estrin, D.: GPS-less low cost outdoor localization for very small devices. IEEE Personal Communications Magazine 7(5), 28–34 (2000)
19. Lazos, L., Poovendran, R.: SeRLoc: Secure range-independent localization for wireless sensor networks. In: Proceedings of the ACM Workshop on Wireless Security, WiSe (2004)
20. Ji, X., Zha, H.: Sensor Positioning in Wireless Ad-hoc Sensor Networks Using Multidimensional Scaling. In: Proceedings of the Twenty-third Annual Joint Conference of the IEEE Computer and Communications Societies (INFOCOM), pp. 2652–2661 (March 2004)
21. Stoleru, R., He, T., Stankovic, J.A., Luebke, D.: A high-accuracy low-cost localization system for wireless sensor networks. In: Proceedings of the ACM Conference on Embedded Networked Sensor Systems, SenSys (2005)
22. Stoleru, R., He, T., Stankovic, J.A.: WalkingGPS: A practical localization system for manually deployed wireless sensor networks. In: Proceedings of the IEEE Workshop on Embedded Networked Sensors, EmNetS (2004)
23. Bowen, C., Martin, T.: Combining Position Estimates to Enhance User Localization. In: Proceeding of the 9th International Symposium on Wireless Personal Multimedia Communications (WPMC 2006), pp. 648–652 (September 2006)
24. Kleine-Ostmann, T., Bell, A.E.: A Data Fusion Architecture for Enhanced Position Estimation in Wireless Networks. IEEE Communications Letters 5(8), 343–345 (2001)
25. Crossbow Technology, Iris OEM Datasheet (2009), `www.xbow.com/Products/Product_pdf_files/Wireless_pdf/IRIS_OEM_Datasheet.pdf` (retrieved on 4th January 2010)
26. Correia, L.H.A., Macedo, D.F., dos Santos, A.L., Loureiro, A.A.F., Nogueira, J.M.S.: Transmission Power Control Techniques for Wireless Sensor Networks. Computer Networks 51(17), 4765–4779 (2007)

27. He, T., Huang, C., Blum, B.M., Stankovic, J.A., Abdelzaher, T.: Range-free Localization Schemes for Large Scale Sensor Networks. In: Proceedings of the 9th Annual International Conference on Mobile Computing and Networking (MobiCom), pp. 81–95 (2003)
28. Crepaldi, R., Casari, P., Zanella, A., Zorzi, M.: Testbed Implementation and Refinement of a Range-based Localization Algorithm for Wireless Sensor Networks. In: Proceedings of the 3rd International Conference on Mobile Technology, Applications & Systems, Mobility (2006)

Author Index