

A Mobile Application Framework for the Geospatial Web

Rainer Simon, Peter Fröhlich

Telecommunications Research Center Vienna (ftw.)

Donau-City-Str. 1

A-1220 Vienna, Austria

+43 1 5052 830 10

{simon, froehlich}@ftw.at

ABSTRACT

In this paper we present an application framework that leverages geospatial content on the World Wide Web by enabling innovative modes of interaction and novel types of user interfaces on advanced mobile phones and PDAs. We discuss the current development steps involved in building mobile geospatial Web applications and derive three technological pre-requisites for our framework: spatial query operations based on visibility and field of view, a 2.5D environment model, and a presentation-independent data exchange format for geospatial query results. We propose the *Local Visibility Model* as a suitable XML-based candidate and present a prototype implementation.

Categories and Subject Descriptors

H.5.2 [Information Interfaces and Presentation (e.g., HCI)]: User Interfaces – *input devices and strategies, interaction styles, prototyping, standardization*

General Terms

Algorithms, Experimentation, Human Factors, Standardization.

Keywords

Geospatial Web, mobile Web, location based applications, geographical information services.

1. INTRODUCTION

Geospatial applications on the World Wide Web are quickly growing in popularity. Web based mapping applications like Google Maps, Yahoo Maps or Microsoft Windows Live Local, as well as 3D geo-browsing applications like Google Earth or NASA World Wind have attracted considerable interest among Web users and developers alike. Communities that engage in geographically related activities have emerged: By *geo-tagging* Web pages, RSS feeds or photos on photo sharing sites, users are essentially assigning Web content a location in the real world. By using open APIs and online mapping toolkits to build *map mash-ups*, developers are demonstrating how this geospatial data can be brought to new use for a variety of purposes – from education, to planning of day-to-day activities, to entertainment.

Meanwhile, new types of powerful mobile phones and handheld computing devices are appearing on the market: Combining powerful processors, high resolution displays, wireless LAN- or 3G mobile network connectivity with navigation-related features

like GPS receivers or digital compasses, these devices promise to drive the adoption of mobile geospatial services and applications in the near future.

In this paper, we discuss the development of mobile geospatial applications in the light of these recent trends. We remark the discrepancy between mobile applications today, and the unique possibilities offered by new location- and orientation-aware devices. We present an application framework that allows developers to create innovative geospatial user interfaces on high-end devices with advanced navigation features, but at the same time retain interoperability with more conventional devices. The paper is organized as follows:

- Section 2 discusses related work, focusing in particular on new types of user interfaces enabled by the combination of location- and orientation-awareness.
- Section 3 discusses the development of mobile geospatial Web applications using current tools and technologies. We identify three functional steps of mobile location-based interaction and illustrate them by an example. Based on the example, we derive requirements for a generic application framework.
- Section 4 introduces our application framework, which meets the requirements derived in Section 3.
- Section 5 describes the *Local Visibility Model*, a device independent XML format for local geospatial data, which is the key innovation of our framework.
- In Section 6, we apply our framework to the example from Section 3: Due to the *Local Visibility Model*, the example application can retain its user interface on conventional location-aware devices, while at the same time offer a more compelling user experience on devices with compass and tilt sensors.

2. RELATED WORK

Research on location-aware interaction has been actively pursued for several years. A number of research projects have experimented with the scenario of attaching digital information to real-world locations like a virtual post-it note or digital graffiti [4], [7], [16], [24]. By using location aware devices, users can discover and access this information, participate in collaborative mapping activities or engage in location-based social interactions [17], [23]. Other efforts have focused on specific sub-domains within the field of mobile geospatial interaction: Extensive research has, for example, been conducted on the inherent privacy and anonymity problem associated with the accumulation of user location data [3], [5], [29]. Also, the reliability and accuracy problems of GPS in urban areas, where GPS signal shadowing

occurs frequently, have been examined: Alternative positioning methods better suited for urban or indoor environments have been proposed, for example technologies that rely on known locations of wireless LAN and/or GSM cell tower radio beacons [14], [15]; other approaches seek to mitigate the impact of GPS positioning inaccuracy on the user experience by informing the user about it rather than concealing it [35] or, in fact, by exploiting it as a design resource, for example in the context of multiplayer gaming [2].

We consider all of these activities important pre-requisites for our work. The primary topic addressed in our paper, however, is the *interaction* between user and geospatial information. Most mobile geospatial applications today replicate the GUI metaphors of the desktop: Mobile versions of established services like Yahoo! local search [38] or driving directions [37], as well as mobile versions of Google Maps [11], [19] are merely down-sized versions of their large-screen counterparts that shrink the same user interface onto the tiny mobile device. As empirical research has shown, however, it is inappropriate to apply desktop idioms to mobile user interfaces [13]: mobile users are typically occupied with real-world tasks; interactions are rapid and driven by the external environment [22]. Their intentions are likely to be more immediate and goal-oriented than those of desktop users [33]. User interfaces that demand too much visual attention are therefore problematic, and fact may even be a potential hazard in some situations [35].

As a result, considerable efforts have been made to develop alternative user interface concepts for mobile geospatial applications: Already in the late 1990s, Egenhofer [6] predicted *Spatial Information Appliances* – portable tools for professional users as well as a public audience, relying on fundamentally different interaction metaphors than desktop applications: *Smart Compasses* that point users into the direction of points of interest, *Smart Horizons* that allow users to look beyond their real-world field of view or *Geo-Wands* – intelligent geographic pointers that allow users to identify geographic objects by pointing towards them.

Further related work in this direction includes Wasinger et al [34], who equipped a mobile device with a digital compass to realize Geo-Wand-like pointing functionality, as well as Mitchell et al [18] and Strachan et al [28] who applied similar concepts in the context of a mobile multiplayer game and a handheld audio navigation tool, respectively. Recent commercial efforts that aim to bring orientation-aware geospatial applications to the market include [10] and [12], as well as the activities of several well-known mobile handset manufacturers who are beginning to introduce a growing number of mobile phone models with suitable hardware features: for example phones with integrated GPS (e.g. [1], [21] and [26]), with GPS and compass (e.g. [10]), as well as mobile phones with accelerometer-based tilt sensors [20], [25].

3. DEVELOPING MOBILE GEOSPATIAL WEB APPLICATIONS

Our work is motivated by the vision that mobile phones will soon serve as generic hard- and software-platforms for a variety of *Spatial Information Appliances*. The necessary advanced navigation features are being integrated into state-of-the-art mobile devices now and can be expected to be even more widespread in the near future. We argue that, as a next step, an open,

standardized application framework is needed that allows developers to experiment with the innovative features offered by these devices: In the same way that open Web APIs and desktop mapping toolkits have fostered the formation of an avid community of map mash-up developers on the desktop, such a framework can potentially leverage innovative geospatial Web applications on the mobile platform.

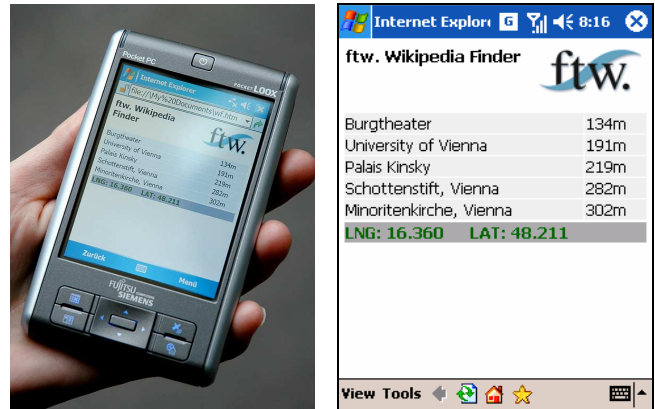


Figure 1. Wikipedia Finder example application.

In this section, we therefore derive a set of requirements for such a framework. Using a simple location-based application as an example, we discuss some of the issues that developers need to address when building mobile geospatial Web applications with current tools and APIs. We thereby want to define the term “mobile Web application” in a slightly broader sense, insofar as we include in this definition all applications that use the Web as communication medium, even if they do not necessarily run in a browser. In that sense, we explicitly include applications that use dedicated client-side software (i.e. software that must be manually installed to the device, unlike e.g. JavaScript code that is embedded in a Web page).

Figure 1 shows a photo and an emulator screenshot of the application we use as our example: The application was implemented on a GPS-enabled PDA, runs in the device’s browser and lists geo-coded Wikipedia articles related to landmarks in the user’s vicinity.

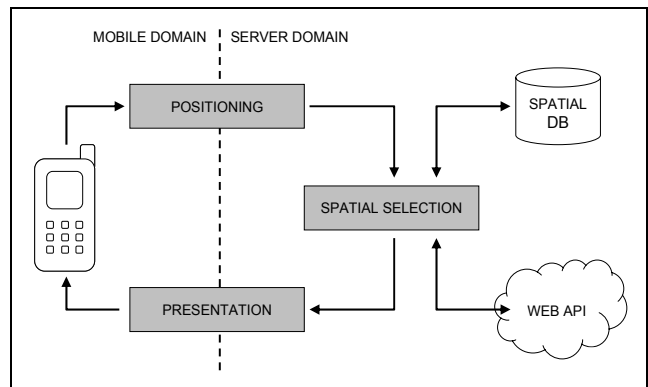


Figure 2. Geospatial application: client/server interaction flow.

For the sake of discussion, we subdivide the sequence of a location-based interaction between mobile client and server into three functional steps, as shown in Figure 2:

Positioning, i.e. the acquisition of information about the mobile client device's location. Depending on the positioning method, this step may require actions on the client-side (e.g. in case of GPS positioning), or on the server-side (e.g. in the case of network based positioning methods [15]).

Spatial selection of content based on the device's location (and possibly further search parameters). Depending on the application, this step may involve querying a local spatial database, an online source over a Web-based geospatial search API, or any other proprietary mechanism to select a subset of geo-referenced content from a larger content base.

Preparation of the content for **presentation** on the client device. Depending on the application architecture, this functional step may be performed in either the mobile domain (e.g. if application-specific XML data is transferred over the wireless link and parsing/formatting is done on the client) or in the server domain (e.g. if the server transforms the spatial query result into an XHTML page, which is rendered on the client without modification).

3.1 Positioning

The possibility to determine the user's location is obviously the primary pre-requisite for building location-based applications. Low-cost standalone Bluetooth GPS receivers and recent PDAs and smartphones with integrated GPS make it possible for developers to include location in their applications, without resorting to costly mobile network operator-provided location services [15]. To date, however, it takes considerable effort to integrate GPS with Web-based applications: Dedicated client-side software is required to obtain location coordinates from GPS (or alternative client-side positioning technologies) and communicate them to the server.

We expect that in the future, devices with integrated GPS will most likely feature a JavaScript API for accessing GPS from within the browser. This way, location-based applications are no longer dependent on proprietary code that must be installed to the device separately. Meanwhile, however, we worked around this limitation by using a custom ActiveX control. The control can be embedded in a Web page, where it exposes a proprietary JavaScript interface to the GPS. The rest of the application is browser based; GPS coordinates are transported to the server in the query string of the HTTP requests.

Discussion. We note that it is not yet possible to develop mobile geospatial applications that can perform the functional step of *positioning* without resorting to proprietary client-side code. This potentially hinders the quick mass-market adoption of mobile geospatial Web applications. We therefore underline the need for a standardized API to access location from within the browser. Since client-side operating system- and device-hardware-specific aspects are, however, beyond the scope of our work, we do not derive any particular requirements for our framework for the functional step of *positioning*.

3.2 Spatial Selection

Unlike a traditional Web search query, which is defined by keywords and Boolean expressions, a geospatial search is defined by (or at least includes) a geographical area expression. In the case of our example implementation, the Wikipedia search feature of the *geonames.org* online geographical database was used to select articles related to the vicinity of the mobile user. The database features a convenient API that allows specifying a radius

around the user's location (or, alternatively, a geographical bounding box). References to articles inside the specified area – together with article-specific meta-data and distance information – are returned in an XML document¹.

Discussion. Spatial queries based on an *extrinsic* reference frame, such as a bounding box or a radius around a center location, are currently the standard query mechanism supported by spatial databases and Web-based geospatial search APIs. This query mechanism reflects the interaction metaphor used on the desktop well: e.g. on a map interface, a user might compose the search query visually, by dragging a rectangle on a graphical map.

We argue, however, that it is not ideal to apply the same metaphor to mobile geospatial search: Unlike a desktop user with a top-down view on a map, the mobile user is physically immersed in the geographical region associated with the search space. Gardiner and Carswell [9] have found that a *deictic* reference frame, which is relative to each individual in the search space, can be considered more relevant for mobile-based search. As a user study conducted by the authors has furthermore confirmed, the concepts of *field of view* and *visibility* are indeed relevant criteria for information discovery in a mobile context [8].

Requirements. We conclude that while geospatial search is becoming available on the Web, current APIs are based on map- and desktop-centric query metaphors such as bounding boxes or circular search areas around a center location. In accordance with [9], we require that a mobile geospatial application framework must offer geospatial query mechanisms defined by *visibility* and *field of view*. An immediate consequence that arises out of this requirement is the need for three-dimensional map data, as we will discuss in more detail in Section 4.

3.3 Presentation

Following the spatial selection process, the selection result is formatted for presentation on the mobile device. In our example application, this is performed on the server, by populating a generic XHTML template with the search results retrieved from the *geonames* database.

Discussion. In our example, the only device that accesses our service is the test device. In realistic scenarios, however, the developer's control over the range of client devices is far more limited: The principal dilemma of mobile Web authoring, which is to ensure consistent presentation and optimized user experience across a broad range of client devices that differ substantially in form factor, screen size and supported markup- and scripting standards, has been generally acknowledged [27], [31], [32]. The advent of mobile devices equipped with advanced navigation sensors introduces an additional degree of freedom to the user interface design process: Devices may potentially feature an arbitrary combination of sensors – from location-only (e.g. GPS) to full 3D orientation (e.g. by combining GPS with compass and tilt sensors). This further complicates the development process, since the developer may not only need to adapt the visual appearance, but in fact the entire interaction model of the user interface: For example, a user interface based on direction – such as the *Smart Compass* – is impossible to adapt to a device without integrated compass. The developer must therefore provide an alternative user interface that may differ substantially from the compass interface, such as a text- or map-interface.

¹ <http://www.geonames.org/export/wikipedia-webservice.html>

Requirements. In line with the first two principles of the W3C’s Mobile Web Best Practices (MWBP) guidelines [33], we derive the following two key requirements for our mobile geospatial Web application framework:

- The framework must enforce *thematic consistency* (best practice no. 1 of the MWBP guidelines) by providing a single, unified data output format that is suitable for all mobile devices, regardless of their navigation sensor feature set, and regardless of the interaction model the developer chooses to implement.
- The framework must encourage developers to best exploit the *capabilities* of devices to provide an enhanced user experience (best practice no. 2 of the MWBP guidelines): On the one hand, it must be possible to derive a usable low-fidelity presentation on client devices with limited computational power, graphical output capabilities and navigation features, e.g. by transforming the output format into meaningful text. On the other hand, the framework should promote more complex real-time user interfaces on full-featured devices, such as interfaces based on the concepts of *Spatial Information Appliances*.

4. A FRAMEWORK FOR MOBILE GEOSPATIAL APPLICATIONS

In this section we describe our framework implementation. The framework features a geospatial query engine that selects content based on its *visibility* from the user’s location, thus meeting the first requirement stated in Section 3. The query results are returned in a novel, device- and presentation-agnostic XML data exchange format which fulfills the remaining requirements identified in Section 3: to enforce *thematic consistency* and encourage developers to exploit the *capabilities* of novel devices. The framework is implemented as a Java library and is designed so that it can either be included seamlessly into a Java application, or operate as an external service over a Web-based API.

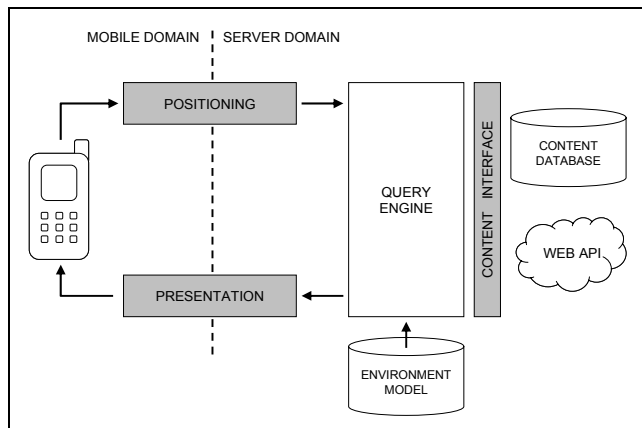


Figure 3. Framework interaction flow.

Figure 3 shows the sequence of a location-based interaction between mobile client and server using our framework: The sequence corresponds to the original process depicted in Figure 2. However, it now includes several new components: First, the spatial selection is handled by the framework’s *visibility query engine* that operates on a 2.5-dimensional environment model stored in a PostgreSQL/PostGIS spatial database. Furthermore, a *content interface* module connects an arbitrary database-, file- or remote source of geospatial content to the framework. The

content interface must be provided by the application developer, based on our framework’s API.

4.1 2.5D Environment Model

A traditional spatial search query only operates on the *content*: i.e. the geographical search area expression is applied to the dataset, and those data points that lie within the specified area limits are returned as the query result. In the case of a visibility based query, on the other hand, it is necessary to consider an additional layer of information: The user’s field of view must be constructed dynamically, based on the geometry of the surrounding environment. Since we generally expect the content dataset to be three-dimensional (i.e. each data point is defined by longitude, latitude and altitude), a three dimensional environment model is needed to compute the query result.

Our framework implementation relies on a 2.5D block model of the environment, i.e. each building in the model is represented by a two-dimensional footprint polygon, which is extruded by a height value. Two datasets were used in our implementation: First, a small sample dataset was produced manually for testing purposes, based on a traditional two dimensional map and approximated building heights. The dataset covers an area of roughly 1x1 km around our office premises (an architecturally rather unusual business district in the North of Vienna, Austria) and is shown in the left image in Figure 4.



Figure 4. Environment model test datasets.

A second, professionally surveyed block model was provided by a project partner. The dataset has identical properties – i.e. each building is represented by a single polygon and height value – and depicts the inner city district of Vienna, as shown in Figure 4, right image.

4.2 Query Engine

The spatial selection process of our framework’s query engine consists of two steps: In the first step, the query engine retrieves the 2.5D environment model of the region around the queried location from PostGIS database. In the same step it also retrieves the content that is located within this region. (As mentioned, it is therefore necessary that the developer provides an interface implementation that translates the framework’s bounding box request to the syntax of the application-specific content source.)

In the second step, the query engine performs the actual visibility detection: Per default, only those data points are included in the result data set that have a clear line of sight between query location and content location. In case a data point is located inside a building, this building is not considered in the visibility detection, i.e. points inside buildings are considered visible, if the

building itself is visible from the user's point of view, as shown in Figure 5.

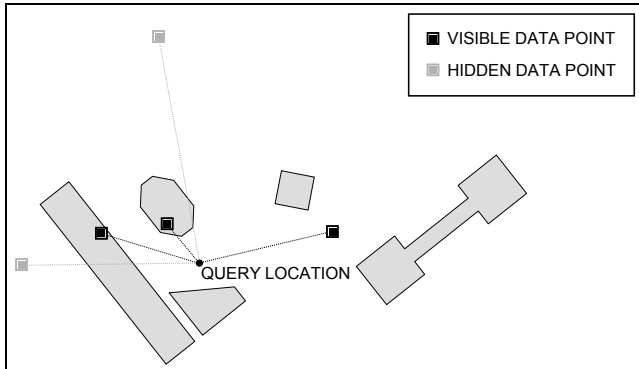


Figure 5. Spatial selection: line of sight.

The query engine can also be configured to include hidden data points in the result. In this case, however, the result syntax clearly distinguishes visible from hidden data points (see Section 5.1).

4.3 XML Query Result

In accordance with the requirements defined in Section 3, we developed a novel XML data exchange format for the query results returned by our query engine. The format is designed with the idea of the *Spatial Information Appliance* in mind: It allows developers to embrace the innovative features offered by novel mobile devices as they become available on the market; however without limiting them when developing for more conventional devices that do not feature advanced navigation features. The XML format can serve as a single, unified model for a multitude of geospatial user interfaces, thus ensuring thematic consistency across presentation types and interaction modes. In the following section, we present our presentation-agnostic XML query result format: the *Local Visibility Model* – or *LVis* in short (pronounced “Elvis”).

5. THE LOCAL VISIBILITY MODEL

The Local Visibility Model is a simplified, ego-centric abstraction of the search result space. The LVis is novel insofar as it retains the geometrical structure of the search result in three dimensions. Since it relies on standard units and measurements (i.e. meters and decimal degrees), meaningful textual output can be produced by simply styling it accordingly, e.g. by using XSLT. Due to the fact that the LVis uses a polar coordinate notation, direction-based user interfaces that would normally require transformations from Cartesian to polar coordinate space (such as e.g. the *Smart Compass* or the *Geo-Wand*) can be realized with considerably reduced development effort.

In order to keep the barrier of entry low for developers previously not involved in geospatial application development, the LVis uses simple spatial modeling metaphors to describe the geometrical arrangement of the search result space: *Points of Interest* (POI) and *Billboards*.

5.1 Points of Interest

The result of a traditional geospatial search query is normally a set of references to geo-coded content. The references are typically point-shaped, i.e. they are represented by a single geographical coordinate. (Therefore, the data points in the search result set can be visualized with graphical markers on a map, along with a list of associated hyperlinks. This common way of

visualizing geospatial search results on the Web is also referred to as the *map-and-hyperlink* architecture [30].)

Point-shaped geo-coded content of the described type is represented in the LVis as a so-called *Point of Interest* (POI). Since the LVis uses a polar coordinate notation, each POI is defined by its distance, heading and elevation relative to the user. This way, a textual description of the POI (e.g. “There is a Snack bar located 200 meters to the North”) can be produced without the need for complex computations.

```
<poi id="poi026" name="Snack Bar"
descr="Snack bar near subway station"
href="http://www.asnackbar.com/"
lng="16.4142" lat="48.23221" alt="178"
d="117" hdg="95" elev="5" />
```

Figure 6. LVis poi (“Point of Interest”) XML element.

Figure 6 shows an XML code sample for the `poi` element: The `id` attribute is a unique identifier for the POI. The `name` attribute provides a short, descriptive name that the GUI designer can use to create e.g. a graphical label. The `descr` attribute provides a longer textual description of the POI which can e.g. be used as help text or tool tip. The `href` attribute contains a URL associated with the POI.

The `lng`, `lat` and `alt` attributes denote the longitude, latitude and altitude of the POI, i.e. its absolute geographical location. The attributes `d`, `hdg` and `elev` describe the relative location of the POI with regard to the user: `d` denotes the distance from the user, expressed in meters. `hdg` is the compass heading of the POI, i.e. the direction the user must turn to face the POI. The heading is always expressed as an integer value in decimal degrees and lies in the range between 0 and 359. The `elev` attribute represents the elevation (or tilt) angle of the POI, i.e. the altitude at which the user must point to aim directly at the POI. In the example from Figure 6, the user would therefore need face eastward, at a heading of 95 degrees, and point almost horizontally (at a 5 degrees upwards angle) to aim directly at the POI.

As mentioned in Section 4.2, only POI that are directly visible from the user's point of view are included in the LVis per default. If the query engine is configured to also include hidden POI in the LVis, these POI will be expressed by a different XML element – the `hpoi` (‘h’ for hidden) element – which has the same attributes as the `poi` element.

5.2 Billboards

Unlike a traditional *map-and-hyperlink* Web search result, which consists only of points of interest, the LVis also includes the actual geometry of the environment in the query result. This means that the query does not only return the *content* in the search space, but in fact it also returns a simplified *three-dimensional geometrical representation* of the 3D search space itself along with the content. The motivation for this approach was twofold:

- First, we argue that local knowledge of the environment geometry on the mobile device is an interesting design feature for some advanced sensor driven interfaces. (For example, in Figure 15 in Section 6.2, we present a type of user interface that presents the spatial query result in a schematized panorama view of the environment.)
- Second, we argue that locally stored geometry is a necessary pre-requisite for real-time user interfaces: If the

client has information about the three-dimensional structure of the vicinity, it can react to changes from integrated sensors without the need to re-query the server. For example, the client might be able to re-compute distances, headings or the visibility of points of interest locally, as the user moves through the environment.

Traditional vector geometry or geographic data formats like the Geography Markup Language (GML) describe geometry using polygons or geometric primitives in Cartesian 2D or 3D space. They are therefore not human-readable as such and require complex client-side processing to produce visual output. Since this conflicts with our design goal of keeping the LVIs presentation-agnostic and, in particular, suitable for textual output, we decided to rely on a simpler, abstracted concept to describe geometry: The LVIs models the environment using a *billboard* metaphor. A billboard approximates a geographic feature, e.g. a building, by a flat, rectangular wall that faces directly towards the user.

Essentially, the LVIs can be thought of as a 360-degree panoramic “cardboard cutout” version of the search space, much like a movie set where the environment is not made up of solid buildings, but instead of building facades. Since billboards are – like POI – described in polar coordinates, textual descriptions of nearby buildings (e.g. “There is an Office Building 150 meters to the South”) can be directly derived from the XML code without the need for arithmetic computations. Figure 7 shows the principle of how our query engine produces a billboard: The illustration shows a top-down view of a building footprint in the environment model (compare Figure 4). First, the visible cross section of the building, seen from the query location, is determined. The billboard is then computed by intersecting the cross section’s center line with the building shape and computing a normal to the center line.

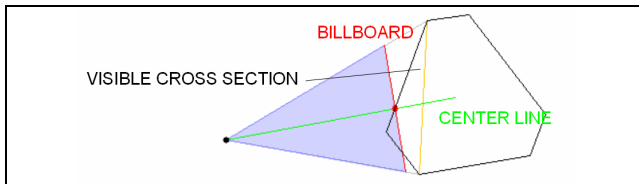


Figure 7. LVIs billboard generation principle.

```
<billboard id="bldg019" name="Tech Gate"
  descr="Tech Gate Office Building"
  href="http://www.techgate.at/"
  lng="16.4094" lat="48.232" alt="183"
  d="42" hdg="125" elev="28"
  hwidth="41" vwidth="18" />
```

Figure 8. LVIs billboard XML element.

Figure 8 shows a code sample for the billboard XML element. The billboard element has all the attributes of the poi element (*id*, *name*, *descr*, *href*, *lng*, *lat*, *alt*, *d*, *hdg* and *elev*). Since the billboard also has a spatial extension, it features two additional attributes, namely the *hwidth* and *vwidth* attributes: While *d*, *hdg*, and *elev* denote the distance, heading and elevation of the billboard’s center, the *hwidth* and *vwidth* attributes provide the horizontal and vertical width of the billboard, respectively. Like *hdg* and *elev*, both attributes are expressed in decimal degrees, with a range of values between 0 and 180 degrees.

In order to better clarify the relation between the attribute values and the corresponding geometrical arrangement, Figure 9 illustrates a representative example.

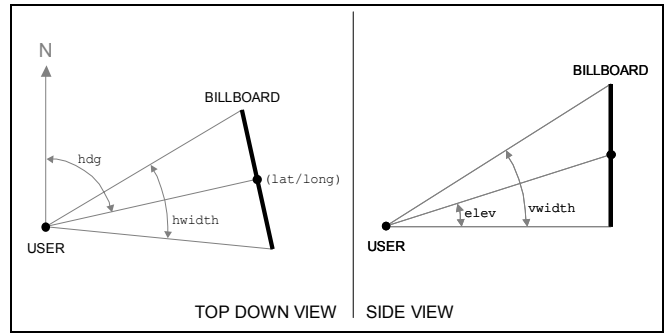


Figure 9. LVIs attributes and geometrical arrangement.

5.3 Occlusion Detection

The query engine uses a basic scan-line algorithm to determine occlusions among individual billboards (compare [9] for a similar approach). In order to speed up the billboard occlusion detection process, the algorithm does not compute occlusions among the actual buildings. Instead, it replaces each building in the scene by its visible cross section in a first step (compare Figure 7). Each cross section is then checked for full or partial occlusions against every other cross section in the scene. Since the complexity of this algorithm rises squared with the number of buildings in the scene, a more efficient algorithm based on guided visibility sampling [36] is currently being integrated into the framework implementation.

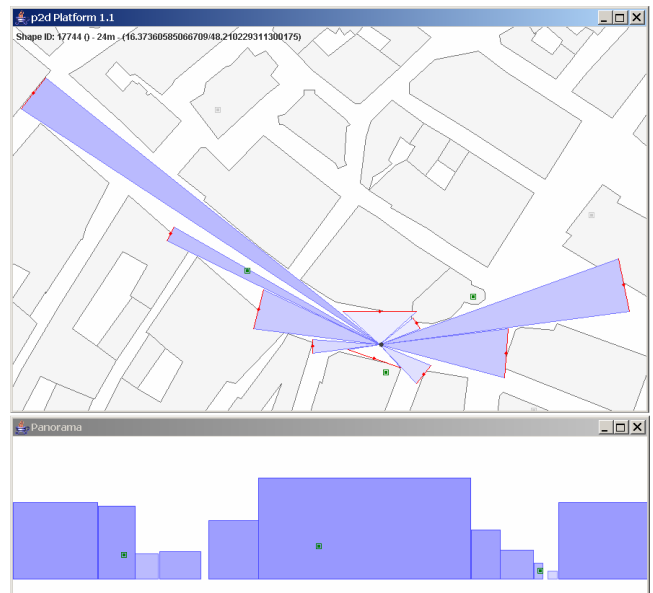


Figure 10. LVIs computation result.

The result of a complete LVIs computation performed by our query engine is shown in Figure 10: The top part of the image shows the computed LVIs in a top-down view; the bottom part of the image shows a 360 degrees panoramic visualization of the result. Visible POI are indicated as small dark rectangular dots, hidden POI are indicated in lighter color. To make the billboards easier to recognize in the top-down view, a “viewing beam” extends from the query location to each billboard, with beams of

lighter color indicating higher billboards. In the panoramic view, rectangles of lighter color represent more distant billboards.

5.4 Limitations

Although the LVIs has so far proven to be a useful data model for realizing different types of geospatial user interfaces (compare Section 6.2), there are two distinct limitations:

Arched Buildings. Our framework currently relies on a 2.5D block model, i.e. every building is modelled as a solid, extruded polygon. As a particular consequence, neither the framework nor the LVIs can describe arched buildings: For example, since the facade of the building shown in Figure 11 is represented in the model by a solid wall, the query algorithm will consider all POI behind that building as *hidden*, even though the user might actually be able to see them through the passageway in reality.



Figure 11. Arched Building.

Buildings that cover more than 180 degrees of the user's view. Per definition, a billboard cannot model buildings that surround the user, i.e. that cover more than 180 degrees of the user's field of view, as shown in Figure 12. We addressed this inherent limitation of the billboard principle with the following strategy: Every building that covers more than 180 degrees of the users view is always modelled using three equal-width billboards instead of one. This way, all buildings that cover between 180 and 360 degrees or the user's field of view can be expressed.

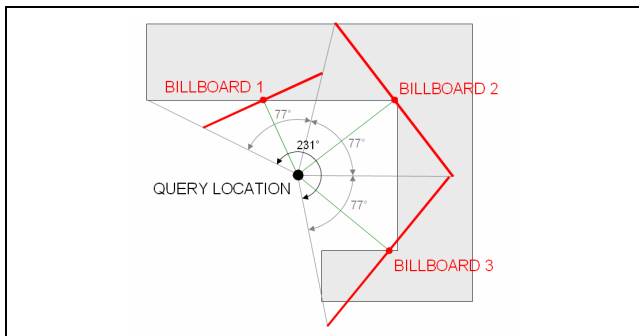


Figure 12. Buildings covering $>180^\circ$ field of view.

6. GEOSPATIAL USER INTERFACES

Based on our application framework, we implemented a modified version of the *Wikipedia Finder* from Section 3, which fully integrates with the framework: HTTP requests from the client are handled by a Servlet, which queries the framework with the GPS coordinates retrieved from the HTTP request. The framework computes the LVIs from the 2.5D environment model and the content retrieved from the *geonames* database, and returns it to the Servlet, which forwards it to the client in the body of the

HTTP response. In this section, we conclude our paper by demonstrating some of the user interfaces that are possible on devices with different sensor capabilities, due to our framework and the LVIs.

6.1 GPS-Enabled PDA

On the GPS-enabled PDA, the client-side JavaScript code was modified. The new script periodically retrieves a new LVIs from the server using a background *XMLHttpRequest* and updates the user interface accordingly. The appearance of the user interface remains identical, with the only exception being that visible Wikipedia articles are now distinguished from hidden ones by a different background color in the list, as shown in Figure 13.

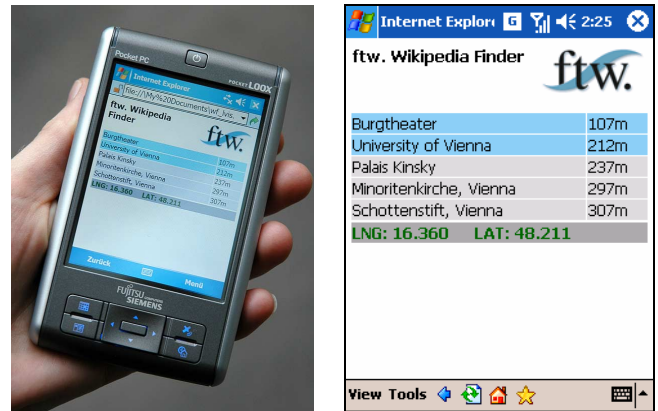


Figure 13. Modified *Wikipedia Finder* on GPS-enabled PDA.

6.2 Sensor-Equipped Mobile Phone

Due to the novelty of mobile devices that combine integrated GPS and compass, no suitable commercial test device was available to us at the time of writing. In order to demonstrate how our framework can support alternative types of geospatial user interfaces on such devices nonetheless, we developed an improvised prototype device, as shown in Figure 14.



Figure 14. Prototype device with compass/tilt sensor.

The device consists of a custom-built plastic shell that snaps onto the back of a standard mass-market mobile phone. Mounted inside the shell is an integrated digital compass/tilt sensor module purchased from a commercial vendor. The module has a standalone power supply and communicates with the mobile phone over a Bluetooth connection. We developed three alternative proof-of-concept user interfaces for the *Wikipedia Finder* application, which we tested on the prototype: A basic *Smart Compass* and a *Geo-Wand*, as described in Section 2, and a "panorama GUI": The *Smart Compass* GUI consists of a circular

compass rose with small arrows that indicate nearby Wikipedia articles. The GUI is rotated dynamically in response to the user's physical orientation, based on the input from the digital compass. The Geo-Wand is used like a scanning device: The user interface indicates visible Wikipedia article locations while the user pans the device slowly across the environment. The panorama GUI, finally, shows locations of nearby Wikipedia articles in a horizontally scrolling 360-degree-billboard view. Screenshots of the Smart Compass and the panorama GUIs are shown in Figure 15.

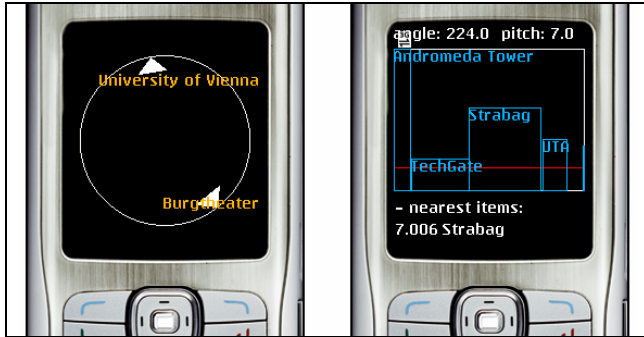


Figure 15. Smart Compass and panorama GUI.

7. ACKNOWLEDGEMENTS

The work presented in this paper is part of the Telecommunications Research Center Vienna's research project *Point to Discover* (<http://p2d.ftw.at/>). *Point to Discover* is funded by mobilkom austria, Siemens Austria and the **Kplus** competence centre programme.

8. REFERENCES

- [1] Benefon company Website. <http://www.benefon.com/>
- [2] Benford, S., Anastasi, R., Flintham, M., Drozd, A., Crabtree, A., Greenhalgh, C., Tandavanitj, N., Adams, M., Row-Farr, J. Coping with Uncertainty in a Location-Based Game. *IEEE Pervasive Computing*, vol. 2 (3), pp. 34-41, 2003.
- [3] Beresford, A. R., Stajano F. Location Privacy in Pervasive Computing. *IEEE Pervasive Computing*, 2 (1). 46-55. 2003.
- [4] Brown, P.J., Bovey, J.D., Chen, X. Context-Aware Applications: From the Laboratory to the Marketplace. *IEEE Personal Communications*, 1995. 4(5), pp. 58-64.
- [5] Consolvo, S., Smith, I.E., Matthews, T., LaMarca, A., Tabert, J., Powledge, P. Location Disclosure to Social Relations: Why, When, & What People Want to Share. *Proceedings of CHI 2005*, Portland, Oregon, USA.
- [6] Egenhofer, M. J.: Spatial Information Appliances: A Next Generation of Geographic Information Systems. 1st Brazilian Workshop on GeoInformatics, Campinas, Brazil, 1999.
- [7] Espinoza, F., Persson, P., Sandin, A., Nyström, H., Cacciatore, E., Bylund, M.: GeoNotes: Social and Navigational Aspects of Location-Based Information Systems. *Proceedings of Ubicomp 2001*, Atlanta, Georgia, September 30 - October 2, 2001.
- [8] Fröhlich, P., Simon, R., Baillie, L., Anegg H. Comparing Conceptual Designs for Mobile Access to Geo-Spatial Information. *Proceedings of the 8th International Conference on Human-Computer Interaction with Mobile Devices and Services (MobileHCI 06)*, Espoo, Finland, 2006.
- [9] Gardiner, K., Carswell, J. D. Viewer-Based Directional Querying for Mobile Applications, Third International Workshop on Web and Wireless Geographical Information Systems (W2GIS2003), IEEE CS Press, Rome, Italy, 2003.
- [10] GeoVector press release: GeoVector and Cybermap Japan (Mapion) Deliver the World's First Pointing Based Solutions for Mobile Phones. Tokyo, Japan, January 26, 2006. <http://www.geovector.com/press/mls.html>
- [11] Google Maps Mobile. <http://www.google.com/gmm/>
- [12] Intelligent Spatial Technologies company Website. <http://www.i-spatialtech.com/>
- [13] Kristoffersen, S., Ljungberg, F. "Making Place" to Make IT Work: Empirical Explorations of HCI for Mobile CSCW. *Proceedings International ACM SIGGROUP Conference on Supporting Group Work (GROUP'99)*, Phoenix, USA, 1999.
- [14] Kunczier, H., Anegg, H. Enhanced Cell ID based Terminal Location for Urban Area Location Based Applications. *Proceedings of the IEEE Consumer Communications and Networking Conference*, Las Vegas, Nevada, USA, 2004.
- [15] LaMarca, A., Chawathe, Y., Consolvo, S., Hightower, J., Smith, I., Scott, J., Sohn, T., Howard, J., Hughes, J., Potter, J., Tabert, J., Powledge, P., Borriello, G., Schilit, B. Place Lab: Device Positioning Using Radio Beacons in the Wild. *Proceedings of Pervasive 2005*, Munich, Germany, 2005.
- [16] Lane, G. Urban tapestries: Wireless networking, public authoring and social knowledge. *Personal Ubiquitous Computing*. July 2003. Vol. 7, no. 3-4, pp 169-175.
- [17] Melinger, D., Bonna, K., Sharon, M., SantRam, M. Socialight: A Mobile Social Networking System. Poster *Proceedings of the 6th International Conference on Ubiquitous Computing*, Nottingham, England, 2004.
- [18] Mitchell, K., McCaffery, D., Metaxas, G., Finney, J., Schmid, S., Scott, A. Six in the City: Introducing Real Tournament – A Mobile Context-Aware Multiplayer Game. *Proceedings of the 2nd Workshop on Network and System Support for Games*, ACM Press, 2003.
- [19] Mobile GMaps. <http://www.mgmaps.com/>
- [20] Nokia 5500 Sport product Website. <http://www.nokia.co.uk/nokia/0,,92079,00.html>
- [21] Nokia press release: It's What Computers Have Become – the New Nokia N95. September 26, 2006. http://www.nokia.co.uk/nokia/0,,28358,00.html?press_release_id=http://press.nokia.com/PR/200609/1077775.xml
- [22] Pascoe, J., Ryan, N., Morse, D. Using While Moving: HCI Issues in Fieldwork Environments. *ACM Transactions on Computer-Human Interaction (TOCHI)*, vol.7, issue 3, pp 417 – 437. 2000.
- [23] Rantanen, M., Oulasvirta, A., Blom, J., Tiitta, S., Mäntylä, M. InfoRadar: Group and Public Messaging in the Mobile Context. *Proceedings of NordCHI'04*, Finland, 2004.
- [24] Reid, J., Hull, R., Cater, K., Clayton, B. Riot! 1831: The design of a location based audio drama. *Proceedings of UK-UbiNet 2004*, October 2004.

- [25] SAMSUNG's Digital World press release: SAMSUNG Introduces World's First "3-dimensional Movement Recognition" Phone. January 12, 2005. http://www.samsung.com/PressCenter/PressRelease/PressRelease.asp?seq=20050112_0000094230
- [26] Siemens SXG75 product Website. <http://www.sxg75.com/>
- [27] Simon, R., Wegscheider, F., Tolar, K. Tool-Supported Single Authoring for Device Independence and Multimodality. Proceedings of the 7th International Conference on Human Computer Interaction with Mobile Devices & Services (MobileHCI'05). Salzburg, Austria, 2005. ACM Press.
- [28] Strachan, S., Eslambolchilar, P., Murray-Smith, R. gpsTunes – Controlling Navigation via Audio Feedback. Proc. of MobileHCI '05, September 19 – 22, 2005, Salzburg, Austria.
- [29] Tang, K.P., Keyani, P., Fogarty, J., Hong, J.I. Putting People in their Place: An Anonymous and Privacy-Sensitive Approach to Collecting Sensed Data in Location-Based Applications. Proceedings of CHI 2006, Montreal, Canada.
- [30] Tezuka, T., Kurashima, T., Tanaka, K. Toward Tighter Integration of Web Search with a Geographic Information System. Proceedings of the 15th International World Wide Web Conferenc. Edinburgh, Scotland, 2006.
- [31] W3C. Authoring Techniques for Device Independence. W3C Working Group Note 18 February 2004. <http://www.w3.org/TR/2004/NOTE-di-atdi-20040218/>
- [32] W3C. Device Independence Principles. W3C Working Group Note. September, 1 2003. <http://www.w3.org/TR/2003/NOTE-di-princ-20030901/>
- [33] W3C. Mobile Web Best Practices 1.0. Basic Guidelines. W3C Proposed Recommendation. November, 2, 2006. <http://www.w3.org/TR/mobile-bp/>
- [34] Wasinger, R., Stahl, C., Krüger, A. M3I in a Pedestrian Navigation & Exploration System. Proceedings of MobileHCI 2003, Udine, Italy, September 8-11, 2003.
- [35] Williamson, J., Strachan, S., Murray-Smith, R. It's a Long Way to Monte Carlo: Probabilistic Display in GPS Navigation. Proceedings of MobileHCI 2006, Helsinki, Finland.
- [36] Wonka, P., Wimmer, M., Zhou, K., Maierhofer, S., Hesina, G., Reshetov, A. Guided Visibility Sampling. ACM Transactions on Graphics. vol 25, no 3. pp 494 - 502. 2006.
- [37] Yahoo! Mobile. Tutorial: driving directions on mobile http://mobile.yahoo.com/resource_center/tutorials/tutcontent?name=dd
- [38] Yahoo! search on mobile. <http://mobile.yahoo.com/search/>