iVu.KOM: A Framework for Viewer-centric Mobile Location-based Services

Farid Zaid, Parag S. Mogre, Andreas Reinhardt, Diego Costantini and Ralf Steinmetz
Multimedia Communications Lab, Technische Universität Darmstadt
Rundeturmstr. 10, 64283 Darmstadt, Germany
{fzaid, pmogre, areinhardt, dcosta, rst}@kom.tu-darmstadt.de

***Abstract*—Users, especially when in unfamiliar areas, would appreciate services that provide them with information about their surroundings and objects in the neighborhood. A natural way to find such information is by *looking* at the objects. Therefore, it would be very useful if Location-based Services could provide a kind of viewer-centric interaction between the users and their surroundings. In this paper, we outline iVu.KOM as a framework for running efficient and prompt mobile viewer-centric queries. We describe our concepts of efficiently modeling the geometry that is surrounding the user, and the modeling of the user's field of view using orientation sensors existing in emerging smartphones. Based on real-world experiments, we provide also an evaluation of the framework capabilities to recognize and react to sensors' uncertainties.**

I. Introduction

"Ali is a researcher participating at a scientific event in an unfamiliar city. While he is hanging around, Ali starts feeling hunger, and he wishes to get to the nearest restaurant. To deal with such a situation, he uses his smart phone as it provides him with access to a *viewer-centric* guide. Ali points his device to scan the street to find a restaurant in his viewing range. Ali is lucky, as the service identifies a matching restaurant and tells him of the direction and distance to reach it. Before going into the restaurant, Ali wishes however to know the menu of the day. For this purpose, he points his device again towards the restaurant, and gets the menu displayed on his device. Ali is satisfied, as he found his destination in a very short time and without too much interaction with his device."

The aforementioned scenario describes an example of *viewer-centric mobile services*; a new class of mobile Location-based Services (LBS) in which the information or services are customized not only to the user's current *location*, but also to the user's *view*. This is possible nowadays by the integration of orientation sensors, like digital compasses and accelerometers, besides GPS modules in mobile devices. The combination of these sensors allows basically to model the user's view elements (e.g. line of sight and field of view) which can be used to perform *viewer-centric* queries.

In general, viewer-centric services offer a two-fold advantage over classical LBS:

- in classical LBS, the search can be restricted to some search radius. Consequently, some of the search results, although lying within the specified search radius, will lie outside the *viewing range* of the user because they may be blocked by existing obstacles (like buildings). In contrast to this, a viewer-centric service suggests that a returned result does really lie within the user's visible range. This is achieved by considering the 3D geometry of the objects surrounding the user.
- typically, existing LBS (e.g. Google Mobile) offers an interface to enter the search query in an input box. The user has then to pick out a relevant search result, probably to iterate the search, and eventually to navigate to the desired destination. In contrast to this, a viewer-centric interaction suggests a *pointing* metaphor to query the service which is more relevant in a mobile setting. The user simply points towards the area she wishes to search in, or towards the objects she wishes more information about. The returned results have then implicit spatial association with the area or objects being pointed at.

However, a still missing link to reach the actual potential of viewer-centric services is the modeling of viewer-centric queries where a level of query accuracy and query efficiency is provided. While query accuracy is mainly affected by the uncertainties of the sensors (e.g. GPS, compass and accelerometer) that are used to model the viewer-centric queries, the query efficiency is mainly affected by the communication needed with the service and executing the query against a geometry model that represents the physical world being searched. Therefore, this paper presents *iVu.KOM*[1]; a framework for supporting viewer-centric mobile services with main focus on exactly tackling the aforementioned issues, i.e. query accuracy and query efficiency.

II. Modeling Viewer-centric Queries

The field of view (FoV) describes "the angular extent of the observable world that is seen from a given viewing point". The human FoV spans approximately 200° horizontally (considering both eyes) and 135° vertically. Consequently, a viewer-centric query based on the FoV can retrieve all POI's lying within the FoV. As shown in Figure 1, the FoV itself is modeled as a pyramid-shaped frustum with a rectangular base such that:

- the head of the pyramid lies at the user's current location, elevated by the height of the user,
- the axis of the pyramid aligns with the line of sight,

[1]Pronounced *eye view* stressing viewer-centricity

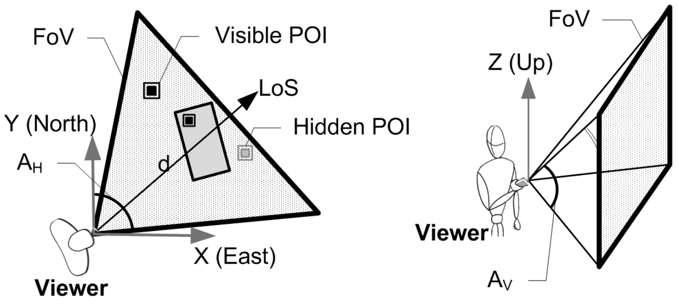


Figure 1: Modeling the user's FoV: top view (left) and 3D view (right)

- the height of the pyramid equals the viewing range of the user d,
- the angle between the vertical triangular faces equals the user's horizontal viewing angle A_H, and
- the angle between the horizontal triangular faces equals the user's vertical viewing angle A_V.

The line of sight (LoS) itself is modeled as a vector originating at the user's current location (as read by the GPS receiver). The vector is aligned in 3D space by the heading from north (as read by the compass) and the tilt (as read by the accelerometer).

As shown in Figure 1, in real world, besides the user's viewing direction and viewing range, the geometry of objects surrounding the user plays a major role in deciding the visibility of POI's.

III. The iVu.KOM Framework

Figure 2 depicts the internal architecture of the iVu.KOM framework which has the following main components[2]:

Scene Computing receives from the client a $QUERY\,(\rho, d)$ and selects from the 2.5D/3D geometry database the geometry within a radius d around the user's position ρ. After that, it simplifies the selected geometry to bounding boxes as will be described in Section IV. Hereafter, we refer to the resulting simplification as the *scene*. This component is essential in supporting the aforementioned query promptness and efficiency requirements.

Content Lookup retrieves from the POI database the geo-tagged content that is lying within the computed scene and matching the search criteria (if one is specified by the user). Alternatively, POI can also be obtained from a geo-enabled search service like Flickr [1].

iVu Model Generation accepts the scene and the associated POI and generates a serialization in the form of an iVu model, which can be expressed in any of the well-established geometry markup languages, like GML [2].

iVu Model Consumption runs on the client and accepts the iVu model and the current position ρ of the user to compute a *skyline view*. A skyline represents a 360° panoramic view of the surrounding geometry as it appears from the current position (or viewing point) of the user. In the next section we explain how the skyline is computed.

Local Spatial Query runs on the client and accepts the skyline and the readings of the orientation sensors, namely the heading ϕ from the compass, and the tilt θ from the accelerometer, to run viewer-centric queries directly on the client.

Presentation transforms a skyline into an application-specific presentation format. This allows the same information to be presented according to the needs of the application or the user interface. Possible formats are, but not limited to, XHTML for display as classical web page, KML for map display, VRML for virtual reality browsers and ARML for augmented reality viewing.

The positioning support component is not part of the framework. iVu.KOM can interface with any location provider including device-centric providers (e.g GPS) and network-centric providers (e.g. based on GSM cell-ID).

[2]Arrow heads in the figure show the data flow between adjacent components, with each component outputting the kind of information within the parentheses.

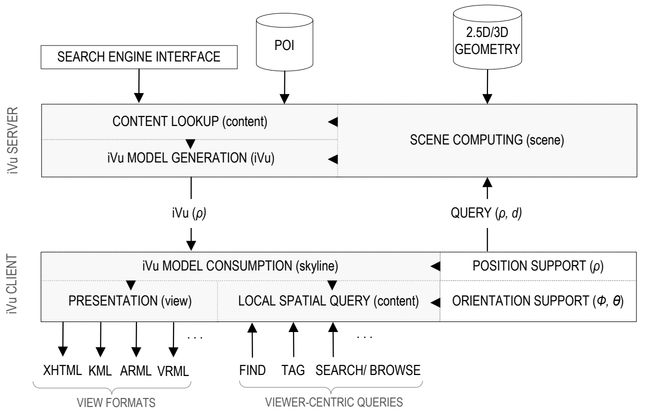


Figure 2: iVu.KOM framework architecture

IV. The iVu Model

Unlike queries in conventional geospatial search systems (e.g., [3], [4]) which return only a set of POI as a result, the returned iVu model contains two kinds of information: the scene and the POI information. Consequently, with the iVu model it is possible to recompute the visibility of the POI as the user changes her position, with minimal need to contact the server. This translates directly into having prompt queries, less network traffic, and less battery consumption on the mobile device.

A. Scene Computing: Oriented Bounding Box

Efficient client-side query execution requires that the scene is kept simple, yet with enough information to run reasonably accurate queries. For this purpose, the scene is confined to an area specified by the radius d around the user position ρ, as illustrated by Figure 3. The value of d can be adjusted for example based on the client's computing and storage capabilities.

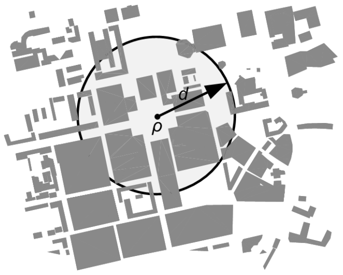


Figure 3: Range query

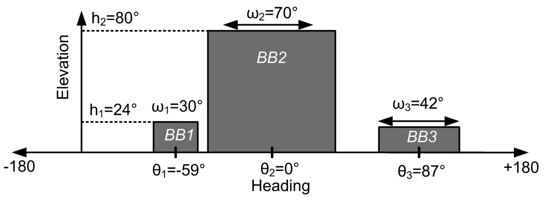
The second step in scene computing is to apply some abstraction or Level of Detail (LOD) to the geometry of each object in the scene. The abstraction, however, should not alter how big an object *effectively* appears in the FoV of the user. This effective size depends mainly on the actual size of the object, its alignment with respect to the user, and its distance from the user. Applying a Convex Hull-based LOD will replace an object by its minimum bounding shape; however, this typically will not reduce the amount of points needed to describe the object. Therefore, we adopt a LOD based on *Oriented Bounding Box (OBB)* [5], [6] which uses a fixed number of information (e.g., 8 vertices if a vertex representation is used) to capture the approximate size of an object, besides maintaining its actual alignment in space.

B. The Skyline View

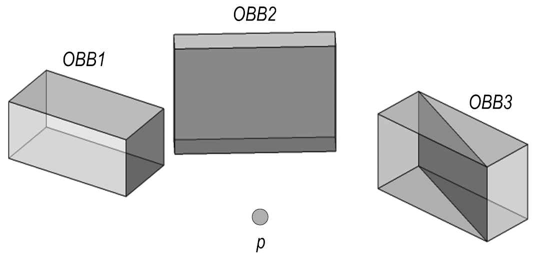
1) Computing the Skyline: As shown in Figure 4a, a skyline replaces each OBB in the iVu model with an equivalent 2D representation, which we refer to as a *billboard*. A billboard of an OBB is computed by firstly finding the side of that OBB that is visible from current position ρ. Thanks to the well-defined shape of an OBB, this task reduces to a case-selection among three options: the width side is visible (as for object 'OBB2' in the Figure 4b), the depth side (as for 'OBB1'), or the diagonal cross-section (as for 'OBB3'). We then specify the billboard using the angular width ω and the angular height h of the OBB's visible side. The billboard is placed on the skyline heading axis at a value equivalent to the heading angle ϕ of the line from ρ to the anchor point a of the OBB. A heading can lie within a range of $[-180°, +180°]$. On the vertical axis, a billboard is placed at an an equivalent angular elevation (in this case is zero as all OBBs are setting on the ground level). The distance d to the billboard is not shown in the skyline and is used to perform z-sorting of the billboards to decide the visible billboards (or the visible sections thereof).

2) Running Queries using the Skyline: With the skyline metaphor it is easy to execute viewer-centric queries. For example, a LoS-based query is executed by firstly projecting the LoS vector on the skyline. The projection appears as a single point at a heading equal to the compass heading ϕ and at an elevation equal to the pitch angle of the accelerometer θ. Then we just find the billboard that is containing that point or nearest to that point. If uncertainty models of the underlying sensors are integrated in the query, then it will be possible to run a probabilistic query which returns the target objects and the hit probability of each object.

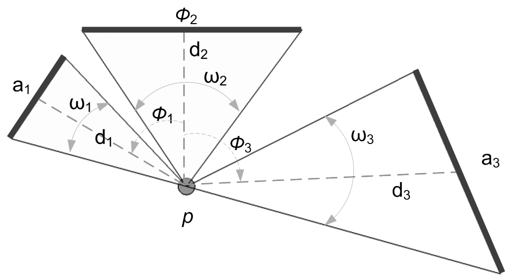
In analogous manner, a FoV-based query can be executed by projecting the user's 3D view frustum or FoV on the skyline view. The FoV will appear in the skyline as a rectangle whose width and height can be set to the horizontal and vertical viewing angles of a human, of about 180° and 120°, respectively. This rectangle can be used for example to filter out POIs that are outside its borders, and together with z-sorting of the billboards, we achieve a viewer-centric browsing/searching effect.



(a)



(b)



(c)

Figure 4: (a) A skyline at point ρ (b) Visible sides from point ρ (c) Angular parameters of a billboard

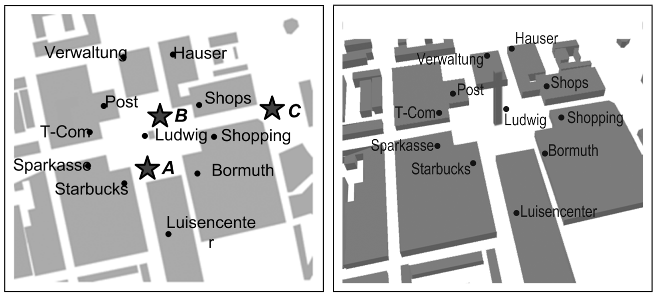
V. System Evaluation

We implemented a prototype of the main components of iVu.KOM. In this paper we demonstrate mainly the functionality of the iVu client (realized on top of an Android-based smartphone [7]) through field tests in the city of Darmstadt in July 2010.

In these tests, the focus was given to the achieved query accuracy under real conditions and by applying the iVu model and the skyline-based query execution. As far as viewer-centric queries are concerned, query accuracy measures the correctness of the query results with respect to their spatial reference.

This means that a LoS-based query should identify the actual object being pointed to, and a FoV-based query should return the visible POIs, i.e., those lying within the actual FoV and not being blocked by any object. This accuracy, in general, is affected by the quality of the sensors readings and the LOD included in the iVu model. However, as the sensors readings themselves change dynamically depending on the surrounding environment, it is essential to test the system in representative environments to gain reliable insights of its behavior under different real world conditions. These environments should represent the different kinds of locations in which iVu.KOM can be typically used, specially by *pedestrian* users. For this purpose, we differentiate between the following classes of test environments with respect to expected sensor disturbance:

Low-disturbed Area. This area is characterized by low GPS and compass disturbances. It has medium sized buildings (2 to 3 floors) that are spaced relatively far away from the user. Therefore, buildings obstruct a few or no portion of the sky, and their structures have low influence on the compass. The area has also few sources of magnetic interference like tramway power-lines. Typical examples include rural areas and open spaces in urban areas.



(a) (b)

Figure 5: Geometry in GPS-compass-disturbed test area from (a) top view and (b) 3D view

GPS-disturbed Area. This area is characterized by some GPS disturbance and low or no magnetic field disturbance. A typical example is a park environment with trees blocking considerable portion of the sky. It may have a few buildings which are placed relatively far away from the user. These cases include also forests and public gardens with dense and high foliage.

GPS- and Compass-disturbed Area. This area is characterized by some GPS and magnetic disturbance. It has medium to large size buildings that are closely spaced from each other and narrow walkways. Therefore, considerable portion of the sky is blocked and building structures are expected to affect the compass. Typical examples include town centers and streets equipped with tramway power-lines.

It needs to be emphasized here that there is no way to specify sharp quantitative values to describe a specific test environment. Besides, even in the same test environment more than one characteristic can be encountered depending on the relative location of the user with respect to the objects. The same test environment may even exhibit different properties with respect to expected sensor disturbance depending on the time of the year or the weather conditions. For example, trees without leaves in winter may become less obstructive to GPS signal, or on the contrary to that, wet trees (which maintain their leaves) can cause more attenuation to GPS signal. For the sake of demonstration, we show here the tests done in a GPS- and compass-disturbed area, as this one can explain the effects of both compass and GPS disturbance on query accuracy.

A. Testing Methodology: Pointing Tests

The goal of the pointing tests is to provide a qualitative evaluation of the accuracy achieved for the queries in the aforementioned test environment. During the test, a number of pointing samples were collected by pointing at different target objects from different positions. Each pointing sample contains two modes: without user feedback (hereafter designated as *no-feedback*) and with user feedback (hereafter designated as *with-feedback*). User feedback means that the user can compensate the compass heading by manually modifying the computed skyline view. The procedure to create a pointing sample in the two modes is as follows:

1) Firstly, the actual (ground truth) GPS position ρ_{act} is pinpointed on a map view provided by the iVu client at a sufficiently large zoom level. Distinguishing marks like ground markers and building edges help setting the actual position on the map with acceptable accuracy.
2) No-feedback mode. The mobile device is pointed towards the target object. The camera preview finder displays then the target overlaid by the currently computed skyline view (i.e., in augmented reality). On pressing the camera button, a pointing sample is stored as a KML file containing the following data:
   - Actual GPS position ρ_{act},
   - Estimated GPS data (i.e., measured by the GPS receiver) including the estimated position ρ_{est} and the estimated GPS accuracy $\Delta_{\rho,est}$ reported as the Circular Error Probable (CEP) (defined as the radius of the circle within which 50% of the position fixes would fall [8]),
   - The actual GPS accuracy $\Delta_{\rho,act}$ computed as $|\rho_{act} - \rho_{est}|$,
   - Estimated compass data including the estimated compass heading ϕ_{est} in degrees and the estimated magnetic field strength ψ_{est} (in μ Tesla), which is computed as the square root of the sum of the squares of the x, y, and z components of the magnetic field. The magnetic sensor reports as well the proposed value for the magnetic field, which is essentially derived from the World Magnetic Model (a model that captures the variations in the magnetic field intensity over space and time [9]). Therefore, this value is considered as a ground truth reference[3],

[3]Reported magnetic intensity can also be verified using an online calculator which takes the GPS position as a parameter. Found at http://www.ngdc.noaa.gov/geomagmodels/IGRFWMM.jsp?defaultModel=WMM

- A serialization of the complete skyline view, computed from the iVu model and based on ρ_{est}, and
- The current camera snapshot including the actual image and the overlaid skyline. Here, only part of the skyline view is displayed depending the camera view port (viewing angle) and the current compass heading ϕ_{est}.

3) With-feedback mode. Here, the mobile device is pointed (as in step 2) towards the object, and, by pressing a dedicated 'Give Feedback' button, a snapshot of the target object is taken and overlaid with the current skyline view. The skyline view in this case is movable, i.e., it can be shifted arbitrarily to the left and the right. To give a user feedback, the skyline view is therefore shifted until it matches with the underlying image. The amount of the shift represents the compass offset Δ_ϕ which is summed to ϕ_{est} to yield the actual compass heading ϕ_{act}. After the feedback is given, the device is pointed again to the target and the file is created.

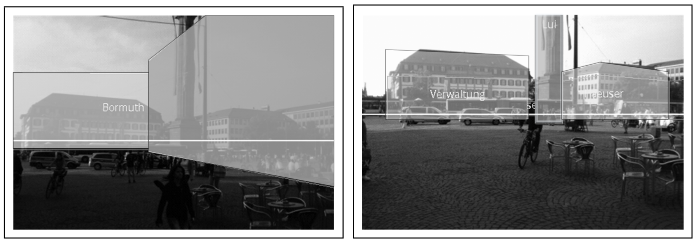
The test data itself was surveyed from different sources, including the OpenStreetMap [10] which provides 2D fingerprints of the buildings in the test areas, and field measurements of the building heights by using a long-range distance laser meter. The amount and quality of test data collected this way is enough for the purposes of the pointing tests.

B. The Tests

The tests were run in the Luisenplatz square in the city of Darmstadt with the geometry shown in Figure 5, in top view (left) and 3D view (right). A total of three representative test locations were selected in this area (marked *A*, *B* and *C* in Figure 5). Location *A* represents a point where a user is nearby a building on one side, location *B* represents a point where a user is standing relatively far away from the buildings, and location *C* represents a point where a user is standing between two buildings that are spaced close from each other. For each of the three locations, 20 pointing samples (each in both no-feedback and with-feedback modes) were collected using the aforementioned procedure. Different target objects were selected during the tests, for example the Ludwig statue in the middle of the square, the Starbucks cafe and the Luisencenter shopping mall (all targets are labelled in Figure 5).

Test Location *A*: The pointing samples collected at this location yielded an average estimated GPS error $\Delta_{\rho,est}$ of 6 meters and an average actual GPS error $\Delta_{\rho,act}$ of 11.8 meters. Figures 6a and 6b respectively depict the actual and estimated skylines at point *A* for one of the pointing samples. Comparing these figures, it is straightforward to see the variations in the sizes and visibility of the billboards depending on the current position. For example, while the the object 'T-COM' is hardly visible from the actual position, the estimated skyline shows that this object will occupy considerable portion of the FoV. Similarly, the 'Ludwig' looks a bit larger according to the estimated position, which indicates that the location was falsely considered closer to the target.

The effect of the distortion in the compass heading is best explained using the skyline-overlaid images. Pointing samples at location *A* yielded an average magnetic field strength of 63.21 μTesla, where it is expected to be about 48.50 μTesla according to the World Magnetic Model. The increase in the magnetic field intensity can be attributed to the magnetic fields induced by the metallic structure of the nearby building (the Luisencenter in Figure 5a). As the tests showed, moving far from location *A*, e.g., towards location *B*, the magnetic field exhibits a value that is closer to the expected value. Figure 7a shows the skyline view when pointing at the target Ludwig statue in the no-feedback mode. The figure shows large discrepancy between the actual target (at $\phi_{act} \approx 14.8°$) and the overlaid skyline (at $\phi_{act} \approx 70.6°$). Repeating the test in the with-feedback mode, Figure 7b depicts very good matching between the target and the skyline after introducing a compass compensation of about $+56°$.



(a) (b)

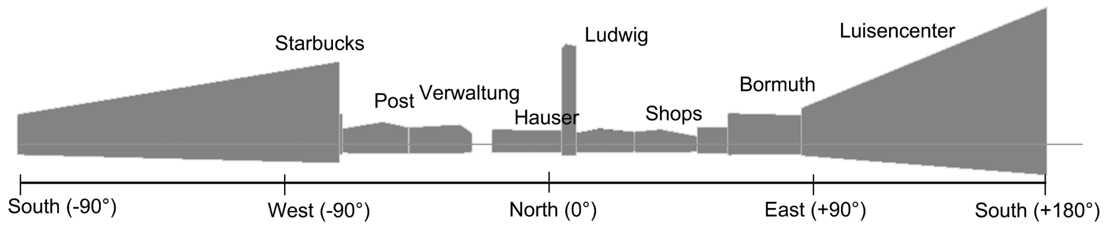
Figure 7: Skyline-overlaid image at location *A* in (a) no-feedback mode and (b) with-feedback mode

Test Location *B*: The pointing samples collected at this location yielded an average $\Delta_{\rho,est}$ of 6 meters and an average $\Delta_{\rho,act}$ of 5.8 meters. This can be attributed to having a chance to detect more GPS satellites. The discrepancy in billboards sizes due to GPS error is therefore less than that for the test location *A*.

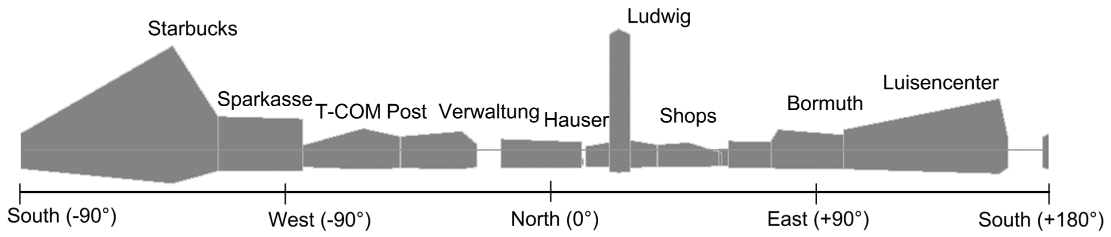
Pointing samples at location *B* yielded an average magnetic field strength of about 53.26 μTesla. Notably, this value is closer to the value reported by the World Magnetic Model (about 48.50 μ Tesla), which may be attributed to moving further away from the building structures when compared to the test location *A*. In the no-feedback mode, a discrepancy of about 14° was detected between the actual target and the overlaid skyline.

Test Location *C*: At this location we detected an average $\Delta_{\rho,est}$ of 6 meters, an average $\Delta_{\rho,act}$ of 13.6 meters and an average ψ_{est} of about 58.14 μ Tesla.

The aforementioned tests proved that iVu.KOM can provide a very good feeling of the query accuracy and using the skyline-based feedback interface it is even possible to enhance this accuracy to identify objects of different sizes and at different distances. However, it is noteworthy that complete elimination of the sensor uncertainties is not possible due to the fast and unpredictable dynamics of the environment.



(a)



(b)

Figure 6: (a) Skyline at location A based on ρ_{act} and (b) Skyline at location A based on ρ_{est}

VI. Comparison to Related Work

In [4], directional queries are built from the location and orientation information and are sent to a server to be run against a 3D spatial database. The result will be a set of visible POI. In contrast to iVu.KOM, this approach demands that a query needs to be re-built and re-executed on the server every time the location changes, which means extra communication and processing overhead. The work in [11] is most related to our approach where the query result contains both POI and geometry information (as billboards). However, the billboards are computed on the server and needs to be updated when the user moves. Besides, the billboards are oversimplified and are sensitive to perspective distortion. iVu.KOM solves this drawback by using the OBB modelling. Commercial services like [3] provide augmented reality platforms where POI are overlaid on the camera preview finder. No geometry information is considered here. Therefore, the user has to manually adjust the radius of the search area, which has less usability compared to the automatic viewing range detection offered by iVu.KOM.

Common to most of existing approaches is that they overlook the accuracy of the sensor readings and do not offer means to handle sensor uncertainty. Besides, these approaches feature only a kind of a browsing service, where the users can only retrieve POIs. In contrast to this, iVu.KOM is a framework that supports both retrieval and generation of user-generated content.

VII. Conclusions and Outlook

We outlined iVu.KOM as a framework for running accurate and prompt viewer-centric queries. We demonstrated the functionality of the system and the usefulness of the skyline metaphor in recognizing the accuracy of the queries. Our next step is to minimze the need for user's feedback by applying an image processing-based technique to automatically adjust the computed skyline to match the skyline detected from the camera image itself.

Acknowledgments

Parts of this research have been supported by the German Research Foundation (DFG) within the Research Training Group 1362 "Cooperative, adaptive and responsive monitoring in mixed mode environments". Special thanks go as well to our student Sebastian Stark who realized the iVu.KOM client.

References

[1] "Flickr: Explore Everyone's Photos on a Map," April 2010. [Online]. Available: http://www.flickr.com/map/
[2] O. G. C. Inc., "OpenGIS Geography Markup Language (GML) Encoding Standard," OpenGIS Standard, 8 2007. [Online]. Available: http://www.opengeospatial.org/standards/gml
[3] "Layar: Browse the World." [Online]. Available: http://layar.com/
[4] K. Gardiner, J. Yin, and J. Carswell, "EgoViz - A Mobile Based Spatial Interaction System," in *W2GIS*, 2009, pp. 135–152.
[5] G. Barequet and S. Har-Peled, "Efficiently Approximating the Minimum-volume Bounding Box of a Point Set in Three Dimensions," in *SODA '99: Proceedings of the tenth annual ACM-SIAM symposium on Discrete algorithms*. Philadelphia, PA, USA: Society for Industrial and Applied Mathematics, 1999, pp. 82–91.
[6] J. O'Rourke, "Finding Minimal Enclosing Boxes," *International Journal of Parallel Programming*, vol. 14, no. 3, pp. 183–199, June 1985.
[7] "Android Developers," 2010. [Online]. Available: http://developer.android.com/index.html
[8] N. G. N. Systems, "GPS Position Accuracy Measures," December 2003.
[9] S. McLean, S. Macmillan, S. Maus, V. Lesur, A. Thomson, and D. Dater, "The US/UK World Magnetic Model for 2005-2010," NOAA National Geophysical Data Center and British Geological Survey Geomagnetism Group, Tech. Rep., December 2004. [Online]. Available: http://www.geomag.bgs.ac.uk/documents/wmm_2005.pdf
[10] "OpenStreetMap: The Free Wiki- World Map," June 2010. [Online]. Available: http://www.openstreetmap.de/
[11] R. Simon, "Mobilizing the Geospatial Web - A Framework and Conceptual Model for Spatially-aware Mobile Web Applications," Ph.D. dissertation, TU Wien, 2008.