Towards Robust and Efficient Visibility-aware Sensor-based Mobile Spatial Interaction

Farid Zaid

Multimedia Communications Lab (KOM)
Technische Universität Darmstadt
Darmstadt, Germany
Email: farid.zaid@kom.tu-darmstadt.de

The emerging platforms for viewer-centric mobile spatial interaction are based on best effort performance of the underlying positioning and orientation sensors. Besides, these platforms adopt rather simplified visibility models of the scene geometry, which makes these platforms inadequate for interaction scenarios that demand accurate and visibility-aware interaction. In this paper, we present *iView*; a client-server reference architecture for mobile spatial interaction that supports both robustness against sensor errors and efficient visibility computing that respects the conventional power and computational limitations of mobile devices.

***Keywords*-**

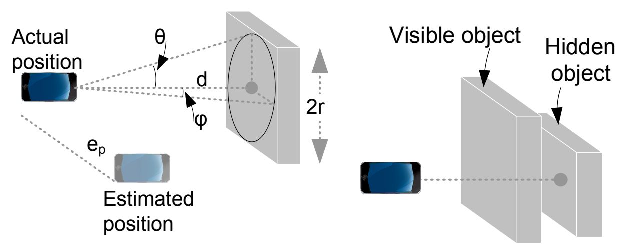
I. Introduction

In addition to GPS, mobile devices are being increasingly equipped with tilt sensors and compasses which opens the door for new forms of spatial interaction paradigms that exploit both the location and orientation of the mobile device. Layar [**?**], for example, enables the users to retrieve information about geographic points of interest (POI) appearing along the line of sight and to project them directly on top of the camera preview finder. While looking appealing for some scenarios like a rough scan of surrounding sightseeing locations in a city, the performance of such viewer-centric spatial interaction platforms may be insufficient for other scenarios. Here, we mainly envision a scenario where a user is interested in an explicit physical object in her field of view (FOV) and would like to interact with that specific object by pointing the mobile device to it. This scenario in turn places a couple of challenges that are hardly tackled by existing platforms:

Sensor Imperfection Problem: In the given scenario, three sensors affect the *pointing quality* to an object: the orientation sensor (compass), tilt sensor and positioning sensor. These sensors suffer in general from static errors like calibration errors and modelling errors, and from dynamic errors owing to interferences from the environment. Assuming we are pointing to a target with radius r and d far away, the overall pointing offset e is given as [**?**]:

$$e = \Delta\rho + d \cdot (\tan(\Delta\theta) + \tan(\Delta\phi)) \quad (1)$$

where $\Delta\rho$ is the positioning error, θ is the tilt (pitch) error and ϕ is the compass (bearing) error as shown in Figure 1(a). This means e should be less than or equal to r so that the pointer can still hit the target.



(a) (b)

Figure 1. (a) Sensor errors and (b) Visibility problem

The Visibility Problem: The described scenario requires a user-centric 3D description of the surrounding objects rather than a 2D map-centric description. Failing to integrate this information may result in interacting with objects that can be hidden from the user's viewing point, as shown in Figure 1(b). As large-scale 3D maps are not available as in the case of 2D maps, existing services [**?**] often assume simplified visibility notion by allowing the user to manually restrict the viewing range. However, this is not sufficient for our scenario as the user should know and adapt the distance to each object she is pointing to. Some research went on this issue. For example, the Local Visibility Model [**?**] proposes an ego-centric geometric model of the local environment around a geographic position. However, the geometry abstractions assumed here introduce offsets in the position of the objects, which can further impact the pointing quality.

II. Approach

Figure 2 sketches *iView*; our reference architecture to tackle the aforementioned problems. Basically, iView supports the mobile spatial interaction process by splitting the task between the *iView Server* and the *iView Client*.

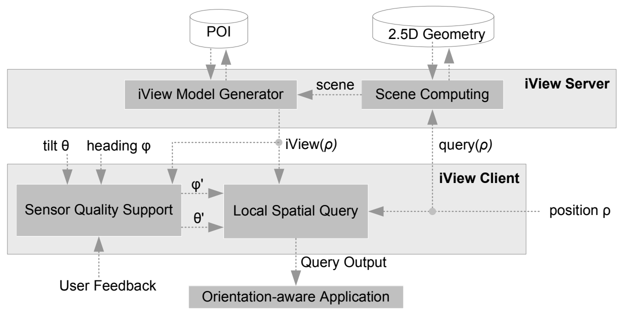


Figure 2. iView reference architecture

A. iView Server

As a response to $query(\rho)$; a query associated with position ρ, the iView server returns an $iView(\rho)$ model, which holds both geometry description of the *scene* around position ρ together with POIs registered in that scene. The *Scene Computing* component gets the scene description from a 2.5D geometry dataset and simplifies the geometry of each object to its bounding box, which confines the maximum dimensions of the object and describes its location and orientation (see Figure 3(a)). On one side, this abstraction step generates more compact scene description, that is more efficient for transmission and processing by the mobile device, and on the other side it still conveys all the information needed at the client. This is in contrast to [**?**] where the adopted abstraction might cause losing geometry details that are needed through the interaction task.

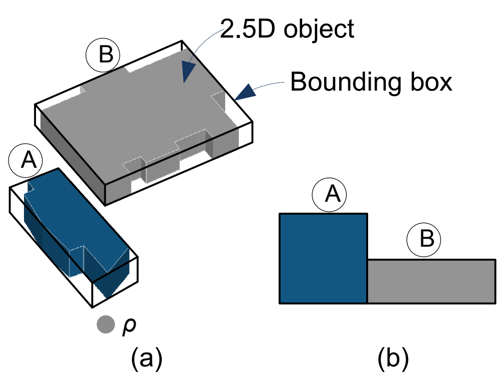


Figure 3. (a) 2.5D objects and their bounding boxes (b) Bounding boxes as they appear in the FOV from point ρ

B. iView Client

Running on the mobile device, the iView client is responsible for executing local spatial queries against the geometry included in the obtained iView model, and using the sensory data (position ρ, tilt θ and compass bearing ϕ). A spatial query aims at identifying the most probable object along the line of sight. This is achieved by both computing the effective area of the bounding boxes as they appear in the FOV (see Figure 3(b)), and by considering the error bounds $\Delta\rho$, $\Delta\theta$ and $\Delta\phi$ (see Eq.1).

The Sensor Quality Support (SQS) features mechanisms to react to estimated sensor errors. For example, by using geometry constraints derived from the already available iView model, SQS can limit the bounds of positioning errors. SQS considers as well user's feedback to achieved interaction quality to compensate for sensor errors (e.g. compass deflection). Similarly, through detection of a user's gait, SQS can filter out pointing errors due to motion [**?**].

This scheme allows real-time experience and reduced communication overhead as the visibility information of the surrounding scene is already locally available. This is superior to other approaches (e.g. [**?**]) which uses the instantaneous position and orientation data to build range queries and send them to a server where they are run against a spatial database. Clearly, such approaches produce a high number of queries as the viewing point moves, and may suffer from unpredictable latencies that can affect the perceived interactivity level.

III. Summary and Outlook

In this paper, iView was presented as a reference architecture to support robust and efficient visibility-aware sensor-based mobile spatial interaction. A first contribution is made by splitting the execution of spatial queries between the server and the clients, thereby mitigating scalability issues at the server and facilitating real-time interaction at the client. A second contribution lies in enhancing the pointing quality on the client side by adapting to errors induced from different sources. Next steps will be to finalize the iView prototype and to run a series of field trials to evaluate its performance and the achieved interaction quality.

Acknowledgement

This thesis is being advised by Prof. Ralf Steinmetz and partly supported by the DFG Research Training Group 1362 (GKmM). Special thanks go as well to M. Tech. Parag Mogre for his insightful discussions.

References

[1] "Layar: Browse the world." [Online]. Available: http://layar.com/

[2] R. Simon, "Mobilizing the Geospatial Web - A Framework and Conceptual Model for Spatially-aware Mobile Web Applications," Ph.D. dissertation, TU Wien, 2008.

[3] R. Azuma, B. Hoff, H. N. III, and R. Sarfaty, "A Motion-Stabilized Outdoor Augmented Reality System," in *Proceedings of IEEE VR '99*, 1999, pp. 252–259.

[4] K. Gardiner, J. Yin, and J. Carswell, "EgoViz - A Mobile Based Spatial Interaction System," in *W2GIS*, 2009, pp. 135–152.