# Position dependent rendering of 3D content on mobile phones using gravity and imaging sensors

**Stefan Marti, Seung Wook Kim, Han Joo Chae**

HCI Team, Computer Science Lab, Samsung SISA
75 West Plumeria Drive, San Jose, CA, U.S.A.

## Abstract

Mobile phones have become powerful computing devices, even being able to render 3D content for games and virtual worlds. However, these phones do not provide a convenient or natural user interface to such 3D content. In this paper, we describe a method to interact with 3D content on mobile phones based on position dependent rendering and ego-motion detection techniques using imaging and gravity sensors, both of which are already built into many current mobile phones. Combining multiple sensors and sensor types is an efficient way to detect position, orientation, and particularly motion of a mobile phone in space. It also makes it possible to distinguish between rotational and transitional ego-motions of constant speed, a task that gravity and orientation sensors alone cannot solve. Our results show that the system provides a convenient and intuitive interaction modality for mobile 3D applications.

## Keyword

Mobile UI, 3D interaction, spatial interaction, egomotion, optical flow, camera phone, sensor disambiguation

## 1. INTRODUCTION

Mobile phones are not only used to make phone calls anymore, but have become sophisticated portable computing devices. They are already powerful enough to render rich interactive 3D content, such as 3D games, virtual worlds, 3D maps, etc. However, there are at least two issues that limit the user experience with 3D content on mobile devices: intuitive spatial interaction and low immersive experience.

First, current interaction methods with 3D environments on mobile phones are neither intuitive nor simple. Unlike in the desktop setting, where the user may have external controllers available such as mice, joysticks, or game controllers, mobile phone users still mostly use buttons and keys, both physical and virtual (including multi touch), to interact with 3D content. Particularly missing are natural ways to change perspective (e.g., look left, up, etc.) and navigate 3D environments (walk forward, turn left, etc.)

Second, today's mobile phones do not provide an immersive user experience with 3D content. This is largely due to the fact that their displays allow only for a limited field of view (FOV): a current handheld display can cover only a small fraction of a user's FOV, because display size is limited by the overall size of the device (resulting in "claustrophobic screen sizes" [1]), and limiting the immersive experience. This limited FOV also leads to poor peripheral awareness in 3D virtual surroundings.

## 2. INTERACTION METHOD

This paper describes technologies to improve the user's interaction with 3D content on current mobile phones. E.g., instead of using button presses to change the perspective, a user utilizes the space around her, treating the mobile device's display as a "window" into, e.g., a 3D virtual world—just like looking through a small window. This window is not static, though: the user can move it around, creating a much larger "virtual view."

The interaction method is related to virtual peephole displays [7], but applied to 3D space; to handheld VR displays [1], but without external sensors; and to cellphone augmented reality systems [5], but not requiring registration of the 3D virtual content with the surrounding real world.

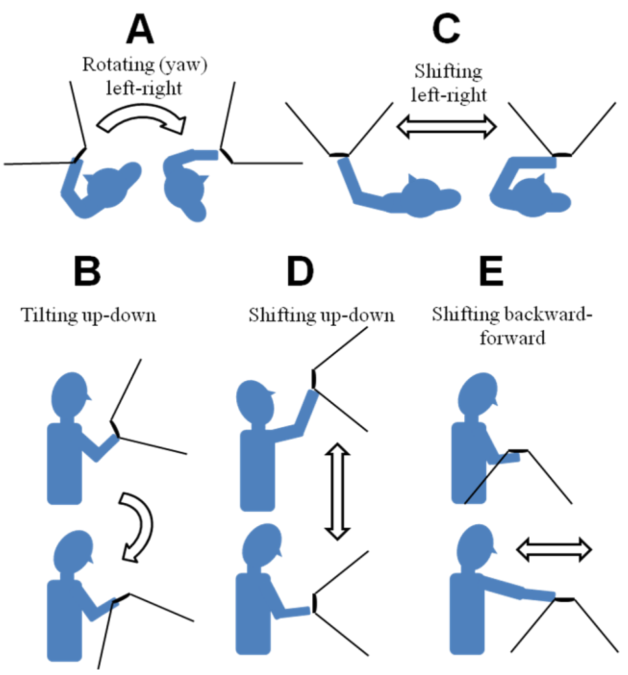

Fig 1. Examples of motions used in virtual window interaction mode: most common are rotational motions (A and B); less common are shifting motions where direction of the motion is in parallel to the display surface, e.g., sideways (C), up-down (D), or backward-forward (E).

Fig. 1 shows different possible motion types which can be used for a virtual window interaction mode. It includes some rotation options (Fig 1. A and B), but also linear motions (Fig. 1C-E) which are less used in mobile phones today[1].

Perceptually, all interaction methods related to virtual dynamic windows are based on the *kinetic depth effect* [2]. As [4] described already more than four decades ago in his seminal paper on surrounding the user with displayed three-dimensional information, “The fundamental idea behind a three-dimensional display is to present the user with a perspective image which changes as he moves. (…) Although stereo presentation is important to the three-dimensional illusion, it is less important than the change that takes place in the image when the observer moves his head. The image presented by the (…) display must change in exactly the way that the image of a real object would change for similar motions of the user’s head. Psychologists have long known that moving perspective images appear strikingly three-dimensional even without stereo presentation.”

In order to implement this interaction method of “virtual window,” it is necessary to determine the position of the mobile device in space (location and orientation) as accurately as possible. Detecting location, orientation, and motion speed of a device itself is generally referred to as “ego-motion detection” (the device determines its own motion speed without external sensors). In other words, in order to navigate a 3D scene by simply moving the device around, ego-motion detection is necessary to detect the motion of the device itself, so that the device knows exactly how and where it has been moved in space. In addition to ego-motion detection, the system has to perform 3D graphics rendering to allow the user visual feedback of her navigation and perspective changing efforts.

We set out to implement the above mentioned interaction methods on a currently available cellphone platform (Samsung Omnia i900). Our design rationale was not only that the cellphone has to run all software components, such as vision processing and 3D content rendering, locally (no remote rendering), but we also were focusing on making the solution software only.

In the following, we will describe our approach to detect motions in as many degrees of freedom (DOFs) as possible, without requiring additional hardware or sensors than are already common on current mobile devices, either from a combination of visual and inertial sensors, or with multiple visual sensors. Then we will describe the results of our preliminary user tests, and the resulting changes to the system from the design iteration.

[1] Linear motions are less commonly measured in today’s mobile phones, due to the fact that most mobile platforms have mostly orientation sensors (measuring gravity and magnetic north), which do not perform well to measure linear shifting motions, particularly if the motions involve low or almost no acceleration (constant but low speed motions).

## 3. EGO-MOTION DETECTION

The largest hurdle to implement position dependent rendering on current mobile devices is to detect the position and orientation of the device with high enough accuracy and as little lag as possible. Detecting the exact position and orientation of the device enables the user to interact with 3D content and provide visual feedback in real-time.
In general, there are two sensor types which could provide some or all of the necessary data[2]. One method to detect ego-motion on a mobile device is to use inertial sensors, such as accelerometers, gravity and orientation sensors. Those sensors are already used for gaming and navigation applications on current cellphones. Alternatively, built-in imaging sensors (i.e., the camera on a mobile phone) could be used to detect ego-motion using vision processing techniques, such as optical flow analysis of the image provided by the camera. The latter method is far less common than the former.

### 3.1. Optical Flow

Generally speaking, optical flow detects the pattern of apparent motion of objects, surfaces, and edges in a visual scene caused by the relative motion between an observer and the scene. Optical flow analysis detects the change of motion in a sequence of images and returns a vector representing the motion. This technique is most often used to track the motion of one or several objects, or is performed on the entire image creating an optical flow field (Fig. 2, top). However, since the goal in this application is to detect the motion of the device itself rather than tracking an object, it is enough that the algorithm returns a single vector by taking the average of all the vectors created as a result of the optical flow (Fig. 2, bottom).

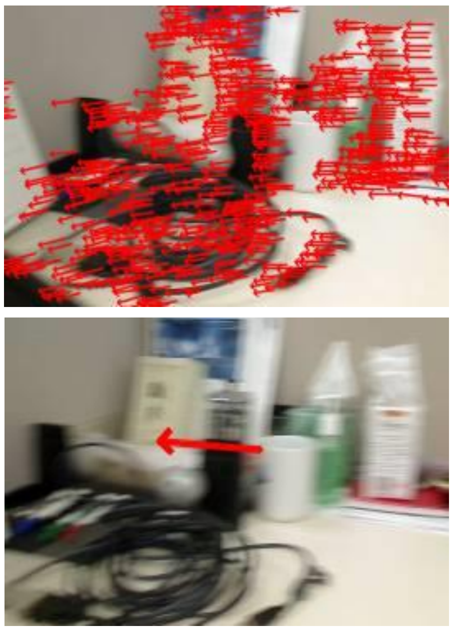
Fig 2. Example of optical flow field (top) and resulting overall optical flow 2D vector (bottom).

[2] Note that GPS (and other trilateration and time-of-flight methods) is not a viable solution, because its resolution is not high enough for our context (currently GPS allows for 2-3m resolution, but we need at least centimeter range resolution), it doesn’t work reliably indoors (where users often are), and it makes the mobile device dependent on external infrastructure, which may or may not be available at all times and places.

This method of egomotion estimation works because the user moves the imager itself, creating a flow of change on the entire image. Hence, the average vector is most likely caused by the motion of the device itself. By taking the average of all vectors, moving objects in the background are disregarded. Thus, the user is not required to be in a completely static environment, but can use the device almost anywhere, and moving objects in the field of view will not affect the general performance of the system.

This method has been previously used, e.g., to create a spatial input device with which the user can draw characters in space [6].

### 3.2. Imaging and Gravity Sensors for Differentiating Rotation from Translation

Measuring ego-motion using optical flow is a robust method to detect slow movement with constant speed, since it is a vision based technique (instead of inertia based). However, it does not allow differentiating between rotation and translation, because the two motion types create quite similar optical flow fields (Fig. 3), resulting in almost identical overall optical flow 2D vectors.

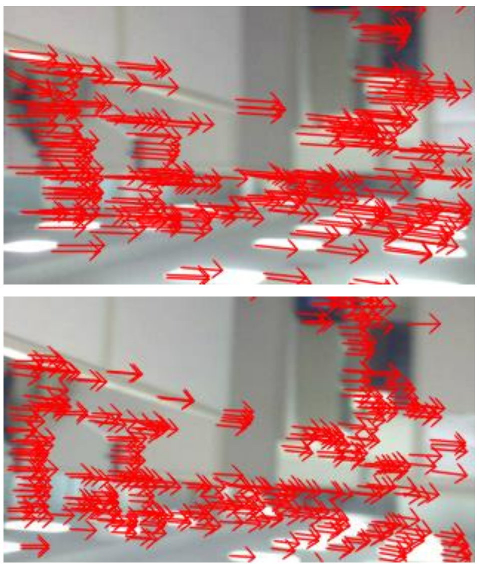
Fig. 3. Optical flow fields from rotation (top) and translation (bottom) are similar, hence difficult to disambiguate

Differentiating those two motions is important for user interfaces, as it can provide additional degrees of freedom for the user. Hence, an additional sensor is needed to disambiguate. Gravity sensors, or orientation sensors, which are already embedded in many mobile devices, can detect rotational movements, complementing the optical flow measurements. Since a gravity sensor can accurately detect both rotational movements as well as absolute orientation angles in two dimensions, the combination of optical flow and gravity sensor maximizes the possible degrees of freedom of motion. Therefore, optical flow analysis is performed to detect translational motions (in two linear dimensions in parallel to the camera plane, e.g., shift left-right and shift up-down), and the gravity sensor is used to detect rotational motions (along two rotational dimensions, e.g., roll and pitch) (Fig. 4).

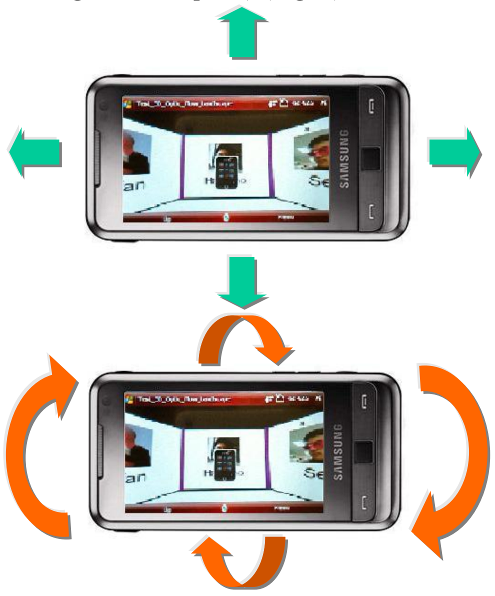
Fig. 4. Visual sensor detects translation motions (top), gravity sensor detects rotation motions (bottom).

Note that the erroneous data created by the optical flow sensor while the device is rotating is filtered out using the rotational data from the gravity sensor, which does not react to shifting motions. The two “fuzzy” sensors types therefore can disambiguate each other conveniently.

### 3.3. Multiple Imaging Sensors for Differentiating Rotation from Translation

Alternatively to using a gravity sensor to complement an imaging sensor, multiple imaging sensors can be used to add additional degrees of freedom (Fig. 5). This solution is still compatible with our initial requirement of increasing the DOF for the user without adding additional sensor hardware, since many of today’s cellphones (including MIDs and UMPCs) are equipped with two cameras: one facing away from the user (for taking pictures), and a second one facing towards the user (for videoconferencing and similar uses).

Since such two cameras are rigidly mounted facing in opposite directions on a single axis, they each provide different overall 2D optical flow vectors when moved in sync. Simple vector addition and subtraction allows for distinction between rotating and shifting motions [3]. For example, the optical flow vector of the first camera c1 has a horizontal and vertical component of x1 and y1 respectively, and the optical flow vector of the second camera c2 has a horizontal and vertical component of x2 and y2 respectively. The shifting (or linear) speed of the device can be derived from taking the difference between x1 and x2 (x1-x2), where as the rotational speed can be derived from taking the sum of x1 and x2 (x1+x2). Similar calculations are used to calculate shifting and rotation motions along the y axis. Furthermore, using more than two imagers (e.g., three orthogonally arranged sensors) will allow the system to measure full 6DOF from simple 2D overall optical flow vectors provided by each imager.

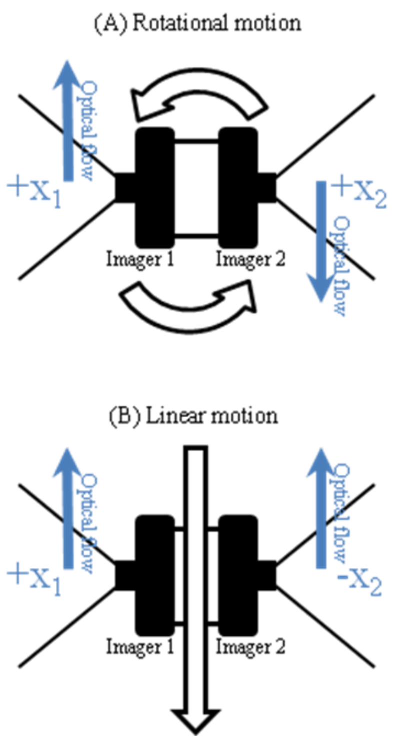

Fig. 5. Illustration of disambiguation principle. The figure shows the top view of a dual imager assembly (imagers pointing sideways). The overall rotational motion of the assembly (A, top) is measured by adding the overall optical flow components x1 (from imager 1) and x2 (from imager 2); the overall linear motion (B, bottom) is measured by subtracting components x2 from x1. Although only the x components are shown, the same goes for the y components of the 2D overall optical flow.

We have implemented multiple imager egomotion detection on several platforms (Netbook with dual webcams, UMPC, Fig. 6), but the implementation on our cellphone platforms is not complete because low level access to both imagers simultaneously is not supported yet with current SDKs, or it is supported, but requires lengthy switching times (possibly due to a single controller for both image sensors). We expect these limitations to disappear.

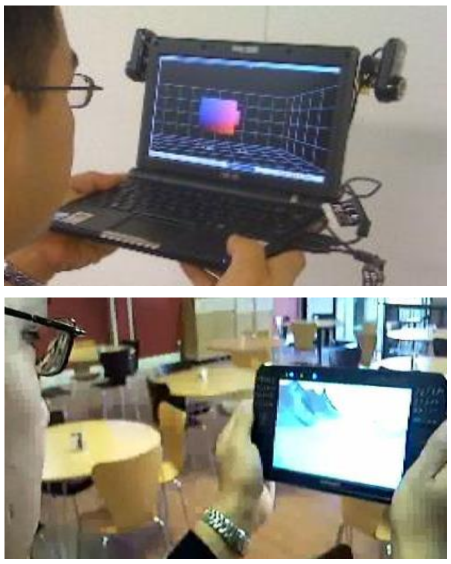

Fig. 6. Optical flow from two imagers for disambiguating rotation from rotation, on a Netbook with sideways pointing webcams (top), and UMPC with two built in cameras (bottom)

## 4. 3D GRAPHICS RENDERING

In addition to detecting ego-motion of the device, our system renders 3D content in real-time according to the devices location and orientation. In the current proof-of-concept, OpenGL ES is used to display a simple 3D environment on the device display, and the virtual perspective (virtual camera position and orientation) is adjusted depending on the motion, orientation, and spatial position of the mobile device. All motion detection and graphics rendering is done in real-time on the device, so that the user receives instant visual feedback from the display (Fig. 7).

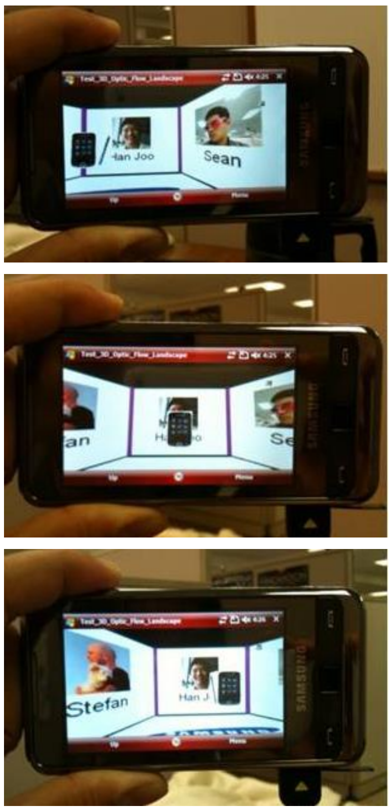

Fig. 7. Example of a sideways shifting motion, at a slow yet constant speed. This is detectable reliably with optical flow from an imager in parallel to the motion plane, but not reliably with inertial sensors alone.

In summary, the initial set of motions and gestures include the following degrees of freedom:

- Shifting the device left, right, up, or down (all motions in parallel to the screen surface, but regardless of the overall orientation of the device) results in equivalent shifting of the perspective in the virtual world
- Rotating the device along the pitch and roll axis results in equivalent rotation of the perspective in the virtual world. (Note that yaw rotation is not supported: an additional sensor, such as a digital compass or second imager, as described above, could provide this DOF.)
- "Walk" mode: Flicking the device (abrupt motion in any direction) initiates navigation (walking forward); another flicking gesture stops the walking motion.

In addition, some haptic feedback cues were implemented in the first prototype. When “Walk” mode is active, the device vibrates (simulating engine vibration). This is intended to give the user additional haptic feedback about the navigation state.

## 5. PRELIMINARY USER TESTING

In our initial study, the application was given to eight users for an informal feedback session. Based on this informal pilot study, we iterated the design of the application as follows:

It turns out that including rolling (rotating the phone along a horizontal axis that is perpendicular to the screen) as a degree of freedom induces the feeling of “being on a boat.” The delay of the gravity sensor along this axis made the 3D rendering engine overshoot, creating a rocking motion that is similar to being on a moving object on water. Users were distracted by this effect, so we disabled roll. It also allowed us to re-use the roll DOF for steering wheel functionality: when the user is “walking,” holding the device vertically (virtual camera moving forward in 3D space), turning the device left and right would result in a direction change, like a steering wheel in a car would. Therefore, this interaction feature was intuitive and easy to understand for the users.

Furthermore, rotating the device to a horizontal position (so that the screen is parallel to the ground), did not have the effect the users expected. Since there was no content on the floor of the OpenGL 3D environment, it appeared as simply black. So we decided to let the virtual camera “rise” smoothly as soon as the device was flipped to a nearly horizontal position. This allowed the user to see her position from above, in effect switching from first person view to a map view (“God’s view,” Fig. 8). Although this tweak broke the original metaphor of directly mapping the device orientation in real space to the virtual perspective in virtual space, it felt natural and intuitive to the users.

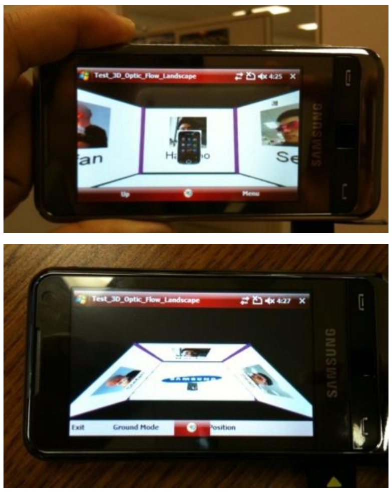

Fig. 8. Gradual transition from first-person view (device held vertically, top) to map view (device held horizontal, bottom), with the virtual camera perspective smoothly rising to an overhead position.

## 6. CONCLUSION

In this paper, we present our current efforts to build a system that combines optical flow and gravity sensors on a mobile device to enable users to interact naturally with spatial content. Compared to existing solutions which use mostly inertial sensors to measure the ego-motion of a mobile device, our system synergistically combines two complementary sensor types which disambiguate each other well. Our working prototype gives users more degrees of freedom when moving the handheld device around like a virtual window, compared to other solutions based on inertial or orientation sensors alone. Therefore, our approach enables the user to interact with 3D content in a more natural way. The work described in this paper has high impact on all 3D application on current mobile phones (as well as media players, MIDs, UMPCs, etc.), taking advantage of position-dependent rendering, without the need for additional sensor and controller hardware. Once mobile devices with more graphics processing power will become available, our approach can easily be extended to render rich interactive 3D content such as virtual worlds, 3D maps, and complex games.

## 7. ACKNOWLEDGMENTS

We would like to thank all the participants of our study for their input, and the members of SAIT FX Part for their support of this project.

## 8. REFERENCES

1. Fitzmaurice, G., Zhai, S., Chignell, M. Virtual reality for palmtop computers. In *ACM Transactions on Information Systems* (1993), 11(3), 197-218
2. Green, B.F. J. Figure coherence in the kinetic depth effect. In *Journal of Experimental Psychology*, *62,* 3 (1961), 272-282
3. Moghaddam, B., Brand, M. E., Waters, R. C., Freeman, W. T., Beardsley, P. A. Optical sensing and control of movements using multiple passive sensors. European patent application EP1089215A1, filed July 5, 2000
4. Sutherland, I. A Head-Mounted Three Dimensional Display. In Proc of *Fall Joint Computer Conference* (1968), 757-764
5. Wagner, D., Pintaric, T. & Schmalstieg, D. The invisible train: a collaborative handheld augmented reality demonstrator. In *ACM SIGGRAPH 2004 Emerging Technologies* (2004), 12
6. Wang, J., Zhai, S., Canny, J. Camera Phone Based Motion Sensing: Interaction Techniques, Applications and Performance Study. In *ACM UIST 2006*, 101-110
7. Yee, K. Peephole displays: pen interaction on spatially aware handheld computers. In *Proc of SIGCHI Conference on Human Factors in Computing Systems* (2003), 1-8