

# Augmented Reality on Smartphones

Anders Henrysson and Mark Ollila  
Norrköping Visualization and Interaction Studio  
Linköping University, Norrköping, Sweden  
{andhe, marol}@itn.liu.se

## Abstract

*Smartphones with built-in cameras are becoming ubiquitous. These devices couple a camera with communication and processing units and are increasingly capable of 3D rendering. We have conducted camera sensitivity tests on current smartphones to evaluate in what extent they can be used for Augmented Reality (AR) using optical tracking.*

## 1. Introduction

Augmented Reality (AR) is a tool to enhance the user's perception of the environment by mixing a view of it with visual computer generated information. The information can be fetched from a virtual model of the real environment or by direct object recognition. The combination of real and virtual information can be used in applications such as guiding, manuals etc. [3].

### 1.1 The Smartphone

Current smartphones, e.g. the SonyEricsson P800 and Nokia 3650, are handheld terminals that combine a camera with communication capabilities by both GSM/GPRS and Bluetooth, a processor and a color display. They are powerful enough to decode and display video and they are also capable of limited, interactive 3D rendering. Since the MMS-service allows the user to send images and video clips, most smartphones have a built-in camera that can capture still images at VGA resolution (640 x 480) and in some cases video sequences at QQVGA resolution (160 x 120). Because of the optical capability it lay at hand to use optical/video tracking for AR applications.

### 1.2 ARToolkit

One toolkit that uses video tracking is the ARToolkit. The ARToolkit is based on special markers that is known in advance. It is a outside-in system where the object to

be tracked wears the marker. By identifying markers in a video stream the position and orientation of the camera can be calculated. More exactly a reference coordinate system is centered at a marker image and the system calculate the camera transformation matrix which is then used for rendering 3D objects. ARToolkit performs the following steps:

1. Turn captured video images into binary images.
2. Search the image for black square regions.
3. For each square capture and match its pattern against templates.
4. Use the known square size and pattern orientation to calculate a camera transformation matrix.
5. Draw the virtual objects.

## 2 Related Work

There have been numerous implementations of AR and Mobile Augmented Reality (MAR) which is of particular interest for us. A lot of work has been done at Columbia University's Computer Graphics and User Interfaces Laboratory where the "Touring Machine" [2] has been developed. Characteristic for such systems are that they use relatively expensive custom built hardware and only exists in very few numbers. The AR-PDA project at Siemens [1] uses a Compaq PDA with a WLAN module and a camera module. In their system the PDA captures and transmit video to a dedicated AR-server on which all processing such as image analysis, 3D rendering and compositing is made.

## 3 AR on Smartphones

It can be concluded that smartphones meet all of the AR requirements to some extent. There is no need for extra hardware such as tracking devices or communication gear. The prospect of smartphones to be the most ubiquitous AR-capable device has motivated us to look closer at what needs to be done and what solutions are available.

### 3.1 Tracking Capabilities

The proposed tracking technology for AR on smartphones is the optical/video approach. We performed a camera sensitivity test to evaluate the possible tracking performance. We used a SonyEricsson P800 and a Nokia 3650 to capture the images. Since the P800 isn't capable of video capture but only a direct-to-screen display, we took a screen dump of the camera seeker window to get an image of the same geometric resolution (QQVGA) as we would get if we could access the camera buffer. We did the same with the 3650 since we did not have any software that could decode the 3GP video format. While the cameras have a 24-bit photometric resolution, the displays are 12-bit (4096 colors). We don't consider this to be an issue since ARToolkit converts the image into a binary one. The test images were loaded into a simple test program using SDL. We used three patterns provided by ARToolkit; *kanji*, *hiro* and *sample1*, each printed on a separate A4 paper. The images were captured from the front at 1 and 2 meters distance and from the side (45°) at 1 meter. For the P800 the only pattern that was detected was the *kanji* pattern in the image taken from the front, and this while the program was looking for the *sample1* pattern. The 3650 turn out to have a more sensitive camera and the program was able to detect the *hiro* and the *kanji* patterns. It also managed to detect the *sample1* pattern from the side (table 1). These results give us a hint about the sensitivity, which markers to use and how much they much be scaled.

Pattern	1 meter	2 meters	45° (1 meter)
<i>hiro</i>	N	-	-
<i>sample1</i>	-	-	N
<i>kanji</i>	N+SE*	-	-

Table 1: Results for P800 (SE) and 3650 (N).

\* = Program was looking for *sample1*.

### 3.2 Server Assistance

One important question is how to distribute the computation between the client and a possible server. The two systems mentioned represent the two ends of the spectrum ranging from autonomous clients (Touring Machine) to client-server solutions where the server performs all the processing (AR-PDA). Due to bandwidth and latency issues (together with billing models) of current and possibly future phone networks technologies, we conclude that the AR-PDA approach is not applicable for AR on smartphones lacking WLAN communication gear. If it turns out to be too computationally expensive to implement all steps of ARToolkit on the client, a server can be used to process some of the steps. This requires splitting the ARToolkit library into several components as has been done to build

wide-area applications [4]. However, we believe that the required bandwidth between the client and the server can be significantly reduced by implementing step 1,2 and 5 on the client. Assuming that the client is capable of rendering the virtual objects only the 3 x 4 camera transformation matrix has to be transmitted as opposed to a (Q)QVGA video frame. If the latency turns out to be large, prediction based on previous matrices might be used. Further assuming that the client is able to detect the squares these can be transmitted using a polygonal representation (for this assumption to hold, the pattern should be a simple one such as the *sample1* provided by the ARToolkit) reducing the data from a video frame to a few bytes. Given this small amount of data and the possibility of predicting movements, it might be possible to use current GSM/GPRS/EDGE/HSCSD/UMTS networks for client-server communication given a round-trip latency of e.g. 1000-3000 ms associated with GPRS networks [5].

## 4 Conclusion and Future Work

Even though the camera is a very limited tracking device, we regard the smartphone to be an interesting platform for AR thanks to its ubiquity and constant progress. We will be performing further studies by porting ARToolkit to the Symbian platform and experimenting with different implementations using various communication technologies.

## References

- [1] Geiger C, Kleinjohann B, Reimann C, Stichling D. *Mobile AR4ALL*, ISAR 2001, The Second IEEE and ACM International Symposium on Augmented Reality, New York, (2001)
- [2] Feiner S, MacIntyre B. and Webster A. *A Touring Machine: Prototyping 3D Mobile Augmented Reality Systems for Exploring Urban Environments*. Proc. First International Symposium on Wearable Computers, Cambridge, MA, (1999).
- [3] Feiner S. *Tutorial: Research Directions in Augmented Reality*. MUM 2002, Oulu, Finland, (2002)
- [4] Wagner M. *Building Wide-Area Applications with the AR Toolkit*. First IEEE International Augmented Reality Toolkit Workshop, Darmstadt, Germany (2002)
- [5] [www.source2.com/O2\\_Developers/O2\\_technologies/GPRS/Technical\\_overview/gprs\\_latency\\_factors\\_diagram.htm](http://www.source2.com/O2_Developers/O2_technologies/GPRS/Technical_overview/gprs_latency_factors_diagram.htm)