Augmented Reality on Mobile Phones - Experiments and Applications

Anders Henrysson*
NVIS
Linkoping University, Sweden

Mark Billinghurst†
HITLab NZ
University of Canterbury, New Zealand

Mark Ollila‡
NVIS
Linkoping University, Sweden

Figure 1: Three examples of Augmented Reality applications running on a mobile phone. The left image shows visualization of a static object. The center image shows a face-to-face collaborative game. The right image shows an object manipulation and scene assembly application.

Abstract

Mobile phones have reached a level where it is possible to run self-contained Augmented Reality applications using the built-in camera for optical tracking. In this paper we present some of our work in this area. We have created a custom port of the ARToolKit library to the Symbian mobile phone operating system and then developed sample applications which have been evaluated. These include a face-to-face collaborative AR game where we conducted a user study to evaluate multi-modal feedback. We also examined user interface issues where an AR enabled mobile phone acts as an interaction device. Additionally, we discuss how traditional 3D manipulation techniques apply to this new platform. We also describe a mobile phone based Augmented Reality application for 3D scene assembly, which adds a 6 DOF isomorphic interaction technique for manipulating 3D content.

1 Introduction

With the integration of cameras and full color displays, mobile phones have developed into an ideal platform for Augmented Reality (AR). Now that it is technically possible, it is important to conduct research on the types of AR applications that are ideally suited to mobile phones and user interface guidelines for developing these applications. This is significant because the widespread adoption of mobile phones means that this platform could be one of the dominant platforms for AR applications in the near future.

*e-mail: andhe@itn.liu.se
†e-mail:mark.billinghurst@canterbury.ac.nz
‡e-mail:marol@itn.liu.se

AR is a technology that allows a user to see virtual imagery overlaid and registered with the real world. Traditionally the AR content was viewed through a head mounted display (HMD). Wearing a HMD leaves the users hands free to interact with the virtual content, either directly or using an input device such as a mouse or digital glove.

For handheld and mobile phone based AR the user looks through the screen of the device to view the AR scene and needs at least one hand to hold the device. The user interface for these applications is very different than those for HMD based AR applications. Thus there is a need to conduct research on interaction techniques for handheld AR displays, and to produce formal user studies to evaluate these techniques.

New opportunities in mobile phone interaction have emerged with the integration of cameras into the phones. By analyzing the video stream captured by the camera, using simple image processing on the phone, it is possible to estimate the movement of the device. Such estimation is the essential component in a Augmented Reality setup.

In this paper we present the implementation of several possible manipulation techniques and the results of a user study conducted to identify which of these techniques is the most usable. These techniques can be used to provide a 6 DOF interface. We show how different strategies can be combined for manipulation of a general 3D scene using a standard mobile phone. We describe the first example of using phone motion to manipulate graphical objects in 6 DOF to create virtual scenes.

Another interesting area for mobile phone based AR is for supporting collaborative AR applications. Mobile phones are already designed to support local and remote communication and so provide a natural platform for collaborative AR. For example a Bluetooth enabled mobile phone can be used for face-to-face gaming or messaging, while the cellular network supports voice and video calls.

In the next section we review related work in the area of mobile AR, collaborative AR and virtual object manipulation on a handheld platform. Next we talk about user interface aspects of mobile phone AR and the software platform we have developed to support phone based AR applications. We then describe our work in different areas of mobile phone AR, especially virtual object manip-

ulation, scene assembly and collaborative AR. Finally we provide some directions for future research.

2 Related Work

Our work draws on a rich legacy of previous work in handheld augmented reality, collaborative augmented reality, AR interaction techniques and mobile phone gaming.

The first mobile AR set-ups such as Feiners Touring Machine [1997] featured a backpack computer and an HMD. From these days it was obvious that what was carried in a backpack would one day be held in the palm of the hand. Unlike the backpack systems, handheld collaborative AR interfaces are unencumbering and ideal for lightweight social interactions.

Rekimotos Transvision system explored how a tethered handheld display could provide shared object viewing in an AR setting [1996]. Transvision consists of a small LCD display and a camera. These are connected by a cable to a computer that performs the augmentation. Two users sit across the table and see shared AR content shown on the displays. They can select objects by ray casting and once selected objects are fixed related to the LCD and can be moved. The AR-PAD interface [Mogilev et al. 2002] is similar, but it adds a handheld controller to the LCD panel. AR-PAD decouples translation and rotation. A selected object is fixed in space relative to the LCD panel and can be moved by moving the panel. Rotation is performed using a trackball input device. These custom configurations show that if the AR display is handheld the orientation and position of the display can be used as an important interaction tool.

The first commercially available handheld platform to be used for AR applications was the PDA. First there was work such as the AR-PDA project [2001] in which the PDA was used as a thin client for showing AR content generated on a remote server. This was necessary as the early PDAs did not have enough capability for stand-alone AR applications. Then in 2003 Wagner [2003] ported ARToolKit [ART] to the PocketPC and developed the first self contained PDA AR application. Handheld AR applications such as the Invisible Train [Wagner et al. 2005] also show an interesting combination of interacting with the AR content by interacting in the world and with the device itself.

Mobile phone based AR has followed a similar development path. Early phones did not have enough processing power so researchers also explored thin client approaches. For example, the AR-Phone project [Cutting et al. 2003] used Bluetooth to send phone camera images to a remote sever for processing and graphics overlay, taking several seconds per image. However, Henrysson recently ported ARToolKit over to the Symbian phone platform [2004], while Moehring developed an alternative custom computer vision and tracking library [2004]. This work enables simple AR applications to be developed which run at 7-14 frames per second, but requires a 3D marker.

By visually tracking real objects, the camera phone can be used for 6 DOF input. Hachet [2005] has developed a 3 DOF bimanual camera based interface for interaction both on the device itself and for using a PDA as a 3D mouse. The approach is similar to ours in that it establishes the position and orientation of the device by analyzing the video stream captured by the camera. Rohs Visual Codes [2004] is an example of mobile phone barcode reading. By recognizing and tracking a pattern, the phone movements can be estimated and used as input. The pattern can also be associated with phone functions and act as a menu item. Hansens Mixed Interaction Spaces [2005] uses a similar approach by tracking a circle. Non of these works have proven to be sufficient for 3D AR applications.

Finally, work in mobile phone gaming has been used to inform our AR application design. There are several examples of 3D graphics applications on mobile phones. The vast majority are games that provide joystick type control of vehicles and objects in 3D environments. Larsen [2002] describe one of the first 3D applications for the mobile phone with more complex object manipulation. This is a client server setup where the rendering of the bricks is made on the server in addition to collision detection. There is no mentioning of interactive change of the view. Transformation is restricted to 2D translation. Although there are thousands of games available for mobile phones, there is only a handful that use camera input. Two of the best known are Mosquito Hunt and Marble Revolution. Neither of these games are collaborative or true AR applications, but they do show that camera and phone motion can be used to create compelling game experiences.

The application most related to our work in collaborative AR is Hakkarainens Symball game [2005]. This is a two person collaborative table tennis game which uses camera phones that are Bluetooth equipped. The user control a virtual paddle by moving the phone relative to a colour that is tracked. Once again this is not a true AR experience, but it is the first example of a compelling collaborative game on phone that user camera input.

3 Interaction

There have been several interface metaphors developed for desktop based 3D virtual object manipulation. However these may not be appropriate for handheld phone based systems because of important differences between using a mobile phone 3D interface and a traditional desktop interface, including:

- Limited input options (no mouse/keyboard)
- Limited screen resolution
- Little graphics support
- Reduced processing power

There are also several key differences between using a mobile phone AR interface compared to a traditional head mounted display (HMD) based AR system, including:

- The display is handheld rather than headworn
- The phone affords a greater peripheral view
- The display and input device are connected

There are also some key differences between a mobile phone and a PDA. Mobile phones are operated using a one-handed button interface in contrast to the two-hand stylus interaction of the PDA. Due to the easy one-handed maneuvering it is possible to use the mobile phone as a tangible input object itself. In order to interact we can move the device relative to the world instead of moving the stylus relative a fairly static screen. Having one hand free allows the utilization of bimanual interaction techniques.

We assume that the phone is like a handheld AR lens giving a small view into the AR scene. We also assume that the user will be more likely move the phone-display than change their viewpoint relative to the phone. Thus the small form factor of the mobile phone lets us go beyond the looking-glass metaphor to an object-based approach. This metaphor can be applied to other AR applications that do not use a HMD, such as applications developed for projection screens,

tablet-PC and PDAs. Our input techniques are largely going to be based around motion of the phone itself, rather than keypad input into the phone.

4 Platform

In order to develop AR applications for Symbian based mobile phones there were several key steps we needed to perform:

- Port the ARToolKit tracking library to the Symbian operating system
- Develop a peer to peer communications layer
- Build a game application using 3D graphics
- Provide support for audio and haptic feedback

Henrysson was the first to implement ARToolKit for Symbian [2004]. To do this he wrote a C++ wrapper class in order to get rid of global variables, which are prohibited by Symbian. However, both the mobile phones we are targeting and the PDA used by Wagner lack a floating point unit, making floating-point arithmetic orders of magnitude slower than integer arithmetic. To overcome this, we wrote our own fixed-point library featuring variable precision. We did extensive performance tests to select the algorithms that ran fastest on the mobile phone. The average speed-up compared to corresponding floating-point functions was about 20 times. We started out by porting the functions rewritten by Wagner and continued backwards to cover most of functions needed for camera pose estimation. The resulting port runs several times faster than the original port. Some accuracy was lost when converting to fixed point but was perceived as acceptable.

Our graphics application was developed using OpenGL ES. In comparison to desktop OpenGL, memory and processor demanding functions such as 3D texturing and double precision floating point values have been removed along with GLU. A 16:16 fixed-point data type has been added to increase performance while retain some of the floating-point precision. The most noticeable difference is the removal of the immediate mode in favor of vertex arrays. Since Symbian does not permit any global variables the vertex and normal arrays must be declared constant, which limits the dynamic properties of objects.

The phone we were developing for, the Nokia 6630, ships with a software implementation of OpenGL ES. While this takes care of the low level rendering there is still need for a higher-level game engine with ability to import models created with 3D animation software and organize the content into a scene graph. Though M3G (JSR 184) provides model loading features it does not allow us to invoke the ARToolKit tracking library written in C++ since there is no equivalent to Java Native Interfaces (JNI) for J2ME. There are a few commercial game engines written in C++ but they are not suited for AR research applications that use calibration data and a tracking library to set the camera parameters.

To be able to import textured models from a 3D animation package we used a 3D converter application to exported the model to C++ code with OpenGL floating-point vertex arrays and then wrote a simple program that converted this into OpenGL ES compatible fixed point vertex arrays.

For our experiments in collaborative mobile phone AR we needed a way to transfer data between phones. We wrote a simple Bluetooth peer-to-peer communications layer. Our collaborative set-up consists of two mobile phones where one is a server that announces the game as a service and provides a channel for the client to connect to. The client makes an active search for the device and the service. There is thus no need for IP configuration.

Finally, we added support for audio and tactile feedback to our platform by using vibration and the media server from the Symbian API.

5 Featured Work 1: Object Manipulation and Scene Assembly

We need to develop input techniques that can be used one handed and only rely on a joypad and keypad input. Since the phone is handheld we can use the motion of the phone itself to interact with the virtual object. For example, as in AR-PAD, we can fix the virtual object relative to the phone and then position objects by moving the phone relative to the real world. Two handed interaction techniques can also be explored; one hand holding the phone and the second a the marker paper on which AR graphics are overlaid. This approach assumes that phone is like a handheld lens giving a small view into the AR scene. The small form factor of the phone lets us explore more object-based interaction techniques based around motion of the phone itself. Given these requirements there are several possible manipulation methods that could be tried. The following table shows the techniques we have implemented.

Positioning	Rotation
A Tangible 1: The object is fixed relative to the phone and moves when the user moves the phone. When released the object position is set to the final translated position while its orientation is reset to its original orientation.	A ArcBall: When the phone moves the relative motion of the phone is used as input into the arcball technique to rotate the currently selected object.
B Keypad/Joypad: The selected object is continuously translated in the X, Y or Z directions depending on the buttons currently held down.	B Keypad/Joypad: The object rotates about its own axis according to joypad and keypad input. Left and right joypad input causes rotation left and right about the vertical axis etc.
C Tangible 2: The same as Tangible 1, but the user can use bimanual input, moving both the phone and the object that the phone is tracked relative to.	C Tangible 1: The object is fixed relative to the phone and moves when the user moves the phone. When released the object orientation is set to the final phone orientation and position reset to its original position.
	D Tangible 2: The same as tangible 1, but the user can use bimanual input, moving both the phone and the object that the phone is being tracked relative to.

A user study with these techniques showed that the tangible translation was faster than the button interface, but most people felt that the keypad provided higher accuracy. For rotation the arcball and keypad interfaces were the fastest ones but there was no difference between the techniques when it came to perceived accuracy. For implementation details and the complete user study see the original paper [Henrysson et al. 2005a].

Based on these results we developed scene assembly application for the purpose of exploring how translation and rotation can be combined using both tangible and keypad interfaces. The application consists of a minimal scene with two boxes and a ground plane (see Figure 2). The boxes can be moved freely above the ground plane. In the center of the image plane are virtual cross hairs that are used for selection. Selection is made by pressing the joypad button when the box is in the cross hairs. The selection is based on a unique alpha value for each object and the selection is accomplished by sampling the alpha value of the central pixel, indicated by a crosshair. To indicate which object is selected, a yellow wireframe box is drawn around the object. When the joypad key is pressed the object is locked to the phone and highlighted in white. The virtual model is fixed in space relative to the phone and so can be rotated and translated at the same time. When the button is released the new transformation in the global (marker) space is calculated. The ambition for the keypad interface is for it to allow modification of all six degrees of freedom.

Figure 2: Ground plane and two boxes.

To switch between rotation and translation mode using the keypad interface, we have implemented a semi-transparent menu activated by pressing the standard menu button to the left of the joypad. By making the menu semi-transparent we allow the user to see the object to be transformed in the background. This will reduce the risk of forgetting which transformation to apply when browsing the menu. Since the selection is based on the alpha value of the central pixel, no selection can be made in menu mode and no object may have the same alpha value as the menu.

The menu layout consists of a 3 by 3 grid of icons that are mapped to the keypad buttons 1 to 9. (See Figure 3). The chosen transformation will by applied to the object highlighted by a yellow wireframe.

To translate the object in the x-z plane we use the four directions of the joypad and complement it with the 2 and 5 keys for translation along the y-axis. For rotation using the keypad we use the joypad to rotate around the x and z-axis, while the 2 and 5 buttons rotate the object around the y-axis.

5.1 Case study: Virtual Lego

So far we have only considered a minimal but general application allowing virtual block manipulation on a mobile phone. It can be used as a base for any 3D application where altering of the spatial relationship between objects are of interest. To demonstrate this we

Figure 3: The semi-transparent menu for selecting transformation mode

have implemented a simple virtual LEGO application (see Figure 4).

In this application the user can build structures by attaching virtual LEGO bricks to each other in any configuration that would be possible with the physical counterpart. The virtual bricks form substructures when attached to each other. These sub-structures can be treated as a group by selecting the bottom brick. The transformation made to this brick is propagated to the other brick in the sub-structure. This grouping into sub-structures is limited by the fact that a top brick cannot be attached to more than one bottom brick in the current implementation. However, one bottom brick can be the base for two or more top bricks. There is no restriction on how the number of bricks attached to each other.

When selected, the brick is detached from the brick below and can be moved freely. If other bricks are attached directly or indirectly to the selected brick, they will remain fixed in the local coordinate system of the selected brick.

Figure 4: Virtual LEGO bricks

Once released the application checks if the released piece is positioned within the margin of error to be attached to another piece. A grid restricts the transformations, making it easy to attach one piece on top of another as expected from the physical equivalent. We have not implemented any proper collision detection at this stage and the attachment is not checked continuously.

The phone vibrates when bricks are joined or pulled apart to give tactile feedback on detachment and attachment events.

The keypad interface works as before, but the transformation increments and decrements are adapted to the grid.

For further implementation details see the original paper [Henrysson et al. 2005c].

6 Featured Work 2: Collaborative AR

We have developed a simple tennis game to explore face-to-face collaborative AR on mobile phones. Tennis was chosen because it could be played in either a competitive or cooperative fashion, awareness of the other player is helpful, it requires only simple graphics and it is a game that most people are familiar with.

Our tennis application uses a set of three ARToolKit markers arranged in a line. These are printed on a piece of paper that is placed between the players. When the player points the camera phone at the markers they see a virtual tennis court model superimposed over the real world (see Figure 5). As long as one or more of these markers are in the field of view then the virtual tennis court will appear. This marker set is used to establish a global coordinate frame and both of the phones are tracked in this coordinate frame.

To serve the ball the player points their phone at the court and hits the 2 key on the keypad. Once the ball is in play there is no need to use the keypad any more. A simple physics engine is used to bounce the ball off the court and respond to when the player hits the ball with their camera phones

The simulation takes place in marker space. To check for possible collision with the racket, the position of the ball is transformed into camera space. This transformation is given by the ARToolKit tracking. The racket is defined as a circle centered on the z-axis in the xy-plane of the camera space. If there is an intersection between the racket plane and the ball, the direction of the z-axis is transformed into marker space and used to initialize the simulation.

By sending the direction and position vectors of the ball, the simulations will be synchronized each round. Both devices check for collision with the net and if the ball is bounced outside the court. If an incoming ball is missed the user gets to serve since the other devices Bluetooth is in listening mode. The simulation will always be restarted when data is sent and received. Each time the ball is hit there is a small sound played and the phone of the person that hits the ball vibrates, providing multi-sensory cues to help the players. We have not implemented score keeping yet, relying on players to keep score themselves. However this could be added in the future.

In order to evaluate the usability of mobile phones for collaborative AR we conducted a small pilot user study. We were particularly interested in two questions:

1 Does having an AR interface enhance the face to face gaming experience?

2 Is multi-sensory feedback useful for the game playing experience?

To explore these questions we conducted two experiments, both using the AR tennis game we have developed.

The user study showed that the AR was useful even though it was not necessary for the game to be playable. The users appreciated the multi-sensor feedback. However, sound turned out to be much more important than haptic feedback. For the complete user study see the original paper [Henrysson et al. 2005b].

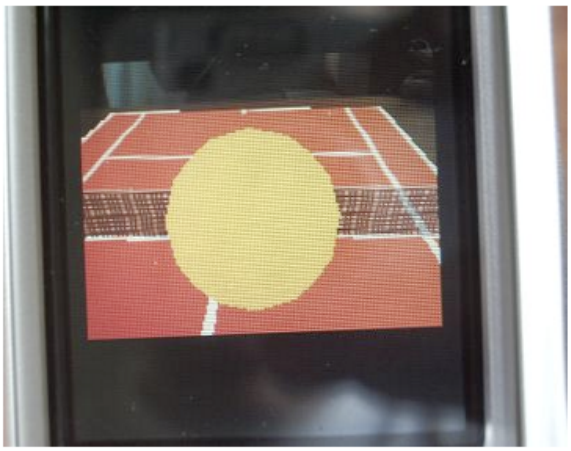

Figure 5: View of the tennis court.

7 Discussion

Even though AR is not essential for any of the presented applications we believe that using the video as background helps the users navigate in 3D. It also gives valuable feedback about the tracking. If the markers are lost the user can use the video feed to quickly maneuver the phone so that a marker becomes visible.

Tracking is the main limitation as the square must be visible at all times. We use multiple markers to extend the tracking range. This adds complexity to the calculations but we have managed to solve the associated problems. We have also experimented with motion flow tracking to allow one corner of the square marker to be outside the image, but this needs more work in order to be an enhancement. Possibly other sensors such as accelerometers or digital compasses could assist the tracking and together with a GPS module make outdoor mobile phone AR possible.

Our initial user experiences indicate that our manipulation set-up allows 6 DOF manipulation for scene assembly applications. By using an easily accessible menu we can map keys to axis instead of functions. Thus we can extend the interface to other operations such as scaling, cloning and various object specific features.

We believe our sample application can serve as a base for tabletop 3D applications where the spatial relationship between the objects is important. We assume most such applications will be games similar to the described virtual LEGO example, but some Virtual Reality applications that require 6 DOF could possibly be developed.

Our work in collaborative AR will be extended with such manipulation techniques rather than being limited to simple object intersections.

In developing a collaborative AR game for mobile phones we have learned a little about design guidelines that can be applied to future collaborative games:

- Face-to-face mobile games could benefit from adding AR interface technology.
- The use of multi-sensory feedback, especially audio is important for increasing game enjoyment.
- If visual tracking is used then the ideal games have a focus on a single shared game space, such as with our tennis game. This enables the players to easily see each other at the same time as the virtual content.

- Due to the slow tracking performance of the current generation of phones games should not rely on quick reflexes or fast competition.
- The screens on mobile phones are very small so collaborative AR games need only use a limited amount of graphics and should mainly focus on enhancing the face to face interaction.
- The use of an appropriate tangible object metaphor is also important for the usability of mobile phone AR applications. Physical manipulation of a phone is very natural so provides and intuitive interaction approach for collaborative AR games.

8 Conclusion and Future Work

In this paper we have presented some of our works in mobile phone AR that are the first of their kind. We developed a basic interaction application for 6DOF object manipulation and scene assembly on mobile phones using AR technology. We have also presented the first collaborative AR game for mobile phones and presented the results of our user studies, which might serve as design recommendations for others who want to develop 3D applications on mobile phones or PDAs. The user studies show that our platform is enough for creating an enjoyable multi-player game using only simple 3D graphics.

We will continue to explore the field of mobile phone AR and in the future we would like to employ the 6DOF manipulations in a collaborative set-up and conduct in-depth user studies. More applications will be developed to explore other aspects of mobile phone AR such as content creation and interfacing intelligent environments.

References

ARToolKit. www.hitl.washington.edu/artoolkit/.

CUTTING, D., ASSAD, M., CARMICHAEL, D. J., AND HUDSON, A. 2003. AR phone: Accessible Augmented Reality in the Intelligent Environment. In *OZCHI2003*.

FEINER, T. H. S., MACINTYRE, B., AND WEBSTER, T. 1997. A touring machine: Protoyping 3D mobile augmented reality systems for exploring the urban environment. In *ISWC (First IEEE International Symposium on Wearable Computers)*.

GEIGER, C., KLEINNJOHAN, B., REIMAN, C., AND STICHLING, D. 2001. Mobile AR4ALL. In *2nd IEEE and ACM International Symposium on Augmented Reality (ISAR 2001)*.

HACHET, M., POUDEROUX, J., AND GUITTON, P. 2005. A camera-based interface for interaction with mobile handheld computers. In *SI3D '05: Proceedings of the 2005 symposium on Interactive 3D graphics and games*, A. Press, Ed., 65–72.

HAKKARAINEN, M., AND WOODWARD, C. 2005. SymBall - Camera driven table tennis for mobile phones. In *ACM SIGCHI International Conference on Advances in Computer Entertainment Technology (ACE 2005)*.

HANSEN, T., ERIKSSON, E., AND LYKKE-OLESEN, A. 2005. Mixed Interaction Spaces Designing for Camera Based Interaction with Mobile Devices. In *Proceedings of CHI 2005*.

HENRYSSON, A., AND OLLILA, M. 2004. UMAR - Ubiquitous Mobile Augmented Reality. In *Proceedings of MUM 2004*, 41–46.

HENRYSSON, A., BILLINGHURST, M., AND OLLILA, M. 2005. Virtual Object Manipulation using a Mobile Phone. In *Proceedings of ICAT 2005 (to appear)*.

HENRYSSON, A., BILLINGHURST, M., AND OLLILA, M. 2005. Face to Face Collaborative AR on Mobile Phones. In *Proceedings of ISMAR 2005*, 80–89.

HENRYSSON, A., OLLILA, M., AND BILLINGHURST, M. 2005. Mobile Phone Based AR Scene Assembly . In *Proceedings of MUM 2005 (to Appear)*.

LARSEN, B., BÆRENTZEN, J., AND CHRISTENSEN, N. 2002. Using cellular phones to interact with virtual environments. In *ACM Siggraph 2002, Conference Abstracts and Applications*.

MOEHRING, M., LESSIG, C., AND BIMBER, O. 2004. Video See-Through AR on Consumer Cell Phones. In *International Symposium on Augmented and Mixed Reality (ISMAR'04)*, 252–253.

MOGILEV, D., KIYOKAWA, K., BILLINGHURST, M., AND PAIR, J. 2002. AR Pad: an interface for face-to-face AR collaboration. In *CHI '02: CHI '02 extended abstracts on Human factors in computing systems*, ACM Press, New York, NY, USA, 654–655.

REKIMOTO, J. 1996. Transvision: A Hand-held Augmented Reality System for Collaborative Deisgn. In *Virtual Systems and Multi-Media (VSMM)'96*.

ROHS, M. 2004. Real-World Interaction with Camera Phones. In *2nd International Symposium on Ubiquitous Computing Sysmtes (UCS2004)*.

WAGNER, D., AND SCHMALSTIEG, D. 2003. First steps towards handheld augmented reality. In *Seventh IEEE International Symposium on Wearable Computers*, IEEE, 127–135.

WAGNER, D., PINTARIC, T., LEDERMANN, F., AND SCHMALSTIEG, D. 2005. Towards Massively Multi-User Augmented Reality on Handheld Devices. In *Third International Conference on Pervasive Computing (Pervasive 2005)*.