Mobile Phone Based AR Scene Assembly

Anders Henrysson[1], Mark Ollila[1] and Mark Billinghurst[2]
[1]NVIS, Linköping University, Sweden *{andhe, marol}@itn.liu.se*
[2]HIT Lab NZ, University of Canterbury, New Zealand *mark.billinghurst@hitlabnz.org*

Abstract

In this paper we describe a mobile phone based Augmented Reality application for 3D scene assembly. Augmented Reality on mobile phones extends the interaction capabilities on such handheld devices. It adds a 6 DOF isomorphic interaction technique for manipulating 3D content. We give details of an application that we believe to be the first where 3D content can be manipulated using both the movement of a camera tracked mobile phone and a traditional button interface as input for transformations. By centering the scene in a tangible marker space in front of the phone we provide a mean for bimanual interaction. We describe the implementation, the interaction techniques we have developed and initial user response to trying the application.

Key words: Mobile Phone. Augmented Reality, CAD

1. Introduction

Unlike desktop computer applications, input options on mobile phones are very restricted. Current mobile phone interfaces are dominated by keypad input. Although limited, it is convenient to use the dial buttons for accessing menus and controlling applications. For improved navigation control, the keypad has been extended by five-way joypads or joysticks. In addition, some high-end smartphones feature stylus input similar to PDAs. Another interface technique is speech recognition e.g. for selecting which contact to call.

None of these techniques are suited for interaction with 3D objects or graphical scenes. In this case both the user viewpoint and individual object position and orientation need to be set interactively. In recent years the range of interface techniques with desktop computer interfaces has been extended with motion sensors and trackers. These allow the user to make inputs by moving the device itself. For example, in the BAT interface [31] virtual objects can be manipulated in 3D space by moving wooden handles that have a magnetic tracker embedded inside them. Instead of pressing a mouse button the user moves the BAT handle naturally and this is mapped into virtual object translation and rotation. Using a range of sensors, desktop interfaces can be developed that provide intuitive six degree of freedom (DOF) input for 3D applications. However there has been no such work presented on the handheld or mobile phone platform.

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. This can be used in a number of ways such as providing a 6 DOF interface, or recognizing objects to make the phone context aware. In this paper we provide the first example of using phone motion to manipulate graphical objects in 6 DOF to create virtual scenes.

Our work also uses Augmented Reality display technology. Augmented Reality (AR) [33] is a technique for visualizing virtual information where the physical reality is part of the scene. The virtual information is registered in three dimensions with the real world and is rendered with a different projection depending on the position and orientation of the viewing device. The main advantage of AR is the elimination of context switching between the real and virtual domains. Another advantage is that real world reference points can aid the navigation of virtual content. AR applications typically use desktop computers. It is only recently that such applications have begun to appear on mobile phones, and ours is the fix example of an AR scene assembly application on a phone.

Traditionally the way to display AR content was to view it through a head mounted display (HMD). Wearing a HMD leaves the users hands free to interact with the virtual content, either directly or through an input device such as a mouse or digital glove. One disadvantage with the HMD is that a 2D GUI is likely to block the user's view of the real world. An alternative approach is handheld AR where the display is non-obscuring and can be compared to a looking glass. For handheld 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.

In recent years AR applications have migrated to a variety of handheld platforms, including Tablet PCs [27], PDAs [28] and mobile phones [11]. Mobile phones are an ideal platform for AR thanks to the integrated camera that allows high quality optical tracking. In addition to integrated cameras the current generation of phones have full color displays, fast processors and even dedicated 3D graphics chips. Henrysson [11] and Moehring [16] have shown how mobile phones can be used for simple single user AR applications. In their work they create custom computer vision libraries that allows developers to build video see through AR applications that run on a mobile phone. Henrysson [9] has also researched face-to-face collaborative AR on mobile phones, finding that users prefer to collaborate together in an AR application than a purely graphical application on the phone.

In this paper we present an application that partly builds upon the conclusions of our previous user study [10] conducted in order to compare different interaction techniques for virtual object manipulation. We show how different strategies can be combined for manipulation of a general 3D scene using a standard mobile phone.

In the next section we review related work on handheld AR, camera based mobile HCI and 3D interaction on mobile phones. Next we discuss the relevant user interface aspects of the mobile phone. We then describe our platform and the scene assembly application in detail, and initial user feedback to the application. Finally we provide some directions for future research.

2. Related Work

There are several examples of camera-based interaction with mobile devices. Two of the best known are "Mosquito Hunt" [18] and "Marble Revolution" [15]. In "Mosquito Hunt", virtual mosquitoes are superimposed over a live video image from the camera and simple motion flow techniques are used to allow the user to shoot the Mosquitos by moving the phone. Similarly, in the "Marble Revolution" game the player can steer a marble through a maze by moving the phone and using motion flow techniques. Neither have 3D registration of the graphics overlaid on the real world.

The virtual soccer game of KickReal [12] allows people to see a virtual ball superimposed over video of the real world and kick it with their feet, but there is no 3D object manipulation. The "Symball" application [7] allows users to hit balls at each other, although with limited 3D tracking. On their phone screen players can see a table tennis table and a virtual paddle. They select a real color that they would like their phone to track and as they move the phone relative to this color the paddle moves in the x-y direction on the screen. Thus, while intuitive, the "Symball" interaction uses 2D rather than 3D object tracking.

By visually tracking real objects, the camera phone can be used for 6 DOF input. Rohs' Visual Codes [23] is an example of mobile phone barcode reading combined with 3 DOF tracking. 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 [24]. Other similar phone applications are the Spotcode [25] and QR-Code pattern tracking systems [21]. Spotcode is a two-dimensional ring like bar code that can be tracked in real time with a phone camera. The Spotcode software performs image processing techniques to extract the identity of the pattern and its angular orientation relative to the phone. Similarly QR-Code is a two-dimensional bar code developed in Japan that can also be recognized by mobile phones, although it does not provide full 6 DOF tracking.

Mobile AR started with backpack configurations such as Feiner's Touring Machine [4]. A similar setup, ARQuake [26], showed how these same systems could be used for outdoor gaming. Both these systems make use of a HMD. At the same time Rekimoto started to experiment with handheld AR. Transvision [22] 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 ARPAD interface [17] is similar, but it adds a handheld controller to the LCD panel. ARPAD 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.

The AR-PDA project [5] was the first to use a camera equipped PDA for AR. The video stream was sent to a server for image analysis and rendering of graphics. Wagner developed the first self-contained PDA AR application [30] by porting ARToolKit [2] to the PocketPC. The Invisible Train [29] uses a stylus for interaction with 3D data on a PDA AR application. The user taps rendered objects on the screen with the stylus to change the position of tracks on the train set. Klein used a similar setup for a game on a tablet PC [13]. The movement of the character is controlled with a stylus. The user can also toss plastic tokens in the game area to create effects.

AR on mobile phones took a similar path beginning with client server solutions. The first to explore self-contained AR on mobile phones were Moehring and Henrysson. Moehring [16] used 3D markers on which a coordinate system was printed. Henrysson ported

ARToolKit to Symbian and created an application that augmented a map with the current tram positions derived from a timetable [11]. A first step to towards interaction with 3D data using an AR enabled mobile phone was [9] were two players sitting face-to-face played tennis using the mobile phones as rackets. The interaction is limited to the collision between the device and a virtual ball being simulated in the marker space between the players.

Henrysson conducted a user study [10] comparing different interaction techniques for translation and rotation of 3D objects using a mobile phone with AR. Similar to the AR-PAD the translation and rotation were decoupled and studied separately. The user selects the object using a button on the keypad. Once selected the object is locked relative to the device. Translation and rotation can be performed by translating and rotating the device respectively. Translation and rotation can also be performed using the keypad. Each axis is mapped to two buttons for decrementing and incrementing the transformation. As an alternative for rotation an Arcball was implemented and controlled by moving the device in a 2D plane. The application presented in this paper builds upon this work and look at how to combine different transformation techniques for general object manipulation and scene assembly.

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 [14] describe one of the first 3D applications for the mobile phone with more complex object manipulation. This is a brick-modeling program where the user selects and moves virtual bricks using the arrow keys on the phone. This is a client server setup where the rendering of the bricks is made on the server in addition to collision detection. The rendered sub-images are sent to the mobile phone client on which only wireframe rendering is possible. There is no mentioning of interactive change of the view. Transformation is restricted to 2D translation.

Hachet [6] 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. For the PDA there is also "3D Blockout" [1], which is a falling block game similar to Tetris. Since the block is falling there is only need for 5 DOF. The interface consists of a menu bar to the right of the screen. The user can move and rotate the block while it is falling to the floor. Watsen et al. [32] has implemented a 6 DOF interface on a PDA for interaction with CAVE-like Virtual Environments. The user controls each degree of freedom with a one-dimensional slider.

Although our application is based on ARToolKit [2], the real time performance of some of these systems led us to believe that we should also be able to get good performance from our code. The mobile phones have developed so much in recent years that there is no need for client server setups for either basic 3D rendering or the image analysis required for 6 DOF tracking. We believe it is now time to explore the possibilities of scene assembly and 3D object manipulation on mobile phones using the camera tracking as input.

The use of AR is not essential for a 3D application, but it is convenient when using camera tracking of markers. If the marker is lost the graphics disappear and the user can adjust the orientation of the device given the video feedback. As stated before it also gives more reference points when navigating the scene.

3. Interaction Methods

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

This suggests that we look at the PDA for appropriate interface metaphors. However there are 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. The pattern we use for tracking is printed on a piece of paper that can be translated by the users' non-dominant hand.

Hansen introduce the term mixed interaction space [8] and argue that the possibility of using mixed interaction spaces is what distinguishes camera-based interaction from other types of sensor-based interaction on mobile devices. The mixed interaction space has the shape of an inverted pyramid i.e. the space spanned from a fixed

point obtained by image processing to the end of the camera view. Movements in the mixed interaction space are used as input in menus and image browsing applications.

In [10] we developed input techniques that can be used one handed and only rely on a joypad and keypad input. Since the phone is handheld we use the motion of the phone itself to interact with the virtual object by fixing it relative to the phone and then position the object by moving the phone relative to the real world. Table 1 shows the techniques we implemented for translation and rotation.

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 [3]: 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.

Table 1: Handheld input techniques

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. These results must be reflected in the application.

When using keypad/joypad input the objects continuously rotate or translate a fixed amount for each fraction of a second while the buttons are pressed. In contrast when the virtual object is fixed relative to the phone, the user can move the object as fast as they can move the phone. Based on this we should expect that the user should be able to translate or rotate the objects faster using tangible input techniques than with keypad input. The fact that the keypad interface was faster for rotation can be explained by its higher accuracy and of the clutching effect associated with tangible rotation.

It is also dependent on where we put the error threshold i.e. when the transformation is considered successful. Based on the user study we have reason to believe that for the average user the error decreases according to figure 1 below. The reason is that for the keypad interface to be useful, it must not increment or decrement the transformation more than one unit for each update. Otherwise it would be useless for fine-tuning, which is its main purpose. The unit size depends on the application. In the tangible mode the velocity of the movement is only limited by the users ability to move his or her hand. This might correspond to several units per update. The current update rate is about 8 Hz. Depending on the error threshold the user would want to minimize the derivative of the error curve at all times. If the threshold is below the intersection point it means switching from tangible to keypad mode. The actual curves depend on the users' fine motor skills. If the user has excellent fine motor skills, the error curves would intersect at a very low error value with no need for keypad input.

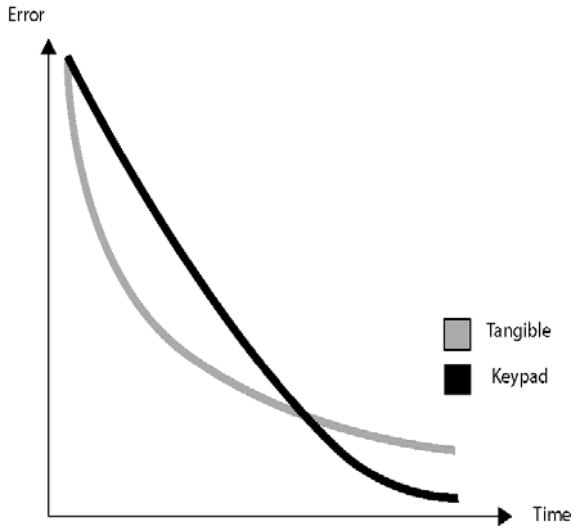

Figure 1: Error curves for tangible and keypad interaction

In the next section we outline in more detail the mobile phone software and hardware platform we are using. Then we describe the scene assembly application.

3. Platform

We build upon the same platform as [11] and [10]. The platform originates from the first custom port of the ARToolKit computer vision tracking library [2] to the Symbian operating system presented in [11]. In order to make it run faster it has been partly converted from floating points to fixed points due to the lack of FPUs on the current generation of mobile phones. To do this a custom made fixed point library was created, partly in assembler language. The average speed-up of the functions implemented was about 20 times compared to floating point versions.

ARToolKit works by recognizing detecting a black square in a binary image. The corners of the square are then used for calculating the position and orientation of the camera in a coordinate system centered on the square. To get the identity of the marker and its initial rotation a known pattern is inscribed inside the square. The square is printed onto an ordinary sheet of paper. Because this can be moved the interface is potentially bimanual.

The 3D graphics is rendered using OpenGL ES [19], which is an embedded subset of OpenGL 1.3, and is suitable for low-power, embedded devices thanks to the removal of redundant APIs and functions. The device we are using, a Nokia 6630, ships with a software implementation of OpenGL ES. The Nokia 6630 has a 220Mhz processor and an integrated 1.3 megapixel camera. The screen size is 178 x 208 pixels and the video capture resolution is 160x120 pixels. The applications we have developed so far run about 8 frames per second.We have also developed a routine for converting 3D meshes into OpenGL ES compatible vertex arrays, but for this application we settled with simple boxes in order to generalize as much as possible.

Combining ARToolKit on the mobile phone with OpenGL ES allows us to create mobile phone applications in which show 3D graphics superimposed over the real work on the phone display. See Figure 2 for a simple example of this.

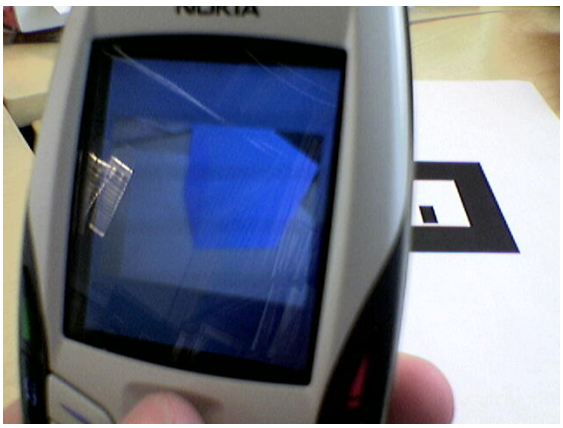

Figure 2: Mobile Phone AR application

3. Application

In our initial study [10] we wanted to consider positioning and rotation separately. The purpose of the application presented in this paper is to show see how these transformations can be combined given the limited interface of the mobile phone.

The application consists of a minimal scene with two boxes and a ground plane. 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 (see Figure 3).

Figure 3: The scene with boxes

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 (see Figure 4). When the button is released the new transformation in the global (marker) space is calculated.

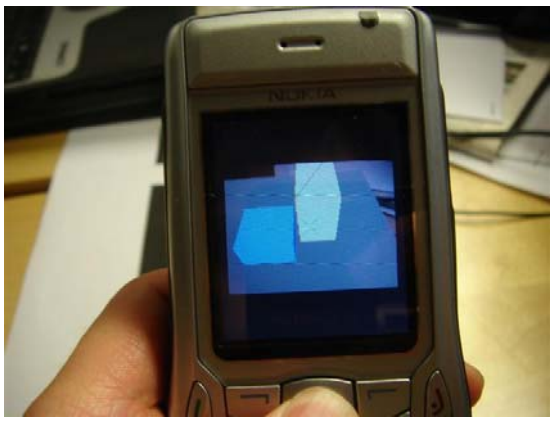

Figure 4: Box locked to phone

The ambition for the keypad interface is for it to allow modification of all six degrees of freedom. We could have achieved this by using twelve available buttons and map each degree of freedom onto two of them, one for decrementing and the other for incrementing the position or orientation, but it would be unintuitive to use. Instead we have chosen to use the same buttons for both translation and rotation. To switch between these two modes 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 5. In our case the number 5 button is mapped to the rotation mode and the number 4 button is mapped to the translation mode. By hitting 4 or 5 the user enters a rotation or translation transformation mode. Once a transformation mode is entered the menu disappears. The user can also toggle the menu by pressing the menu button repeatedly. The transformation will by applied to the object highlighted by a yellow wireframe.

Figure 5: Semi-transparent menu

In both modes we are handling transformation in three dimensions corresponding to the x, y and z-axes of the local object coordinate system. Since the joypad is 5-way and pressing it always means selection, it can only handle two of the dimensions. This is not surprising given that the majority of mobile applications are 2D applications. We map two of the dimensions to the joypad and the third to the 2 and 5 keys.

To translate the object in the x-y plane we use the four directions of the joypad and complement it with the 2 and 5 keys for translation along the y-axis. The translation speed is 4 units/frame yielding a speed of about 30 units per second. 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. The speed of rotation is 4 degrees per update i.e. around 30 degrees per second.

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 have implemented a simple virtual LEGO® application (see Figure 6).

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 sub-structures 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.

Figure 6: Virtual LEGO®

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.

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 so the attachment is not checked continuously.
The phone vibrates when bricks are joined or pulled apart to give haptic feedback on detachment and attachment events. Pressing the C button, located to the right of the keypad, resets the scene. This button was chosen due to its offset from the buttons used for manipulation.

The keypad interface works as before, but the transformation increments and decrements are adapted to the grid. The selected brick is rotated 90 degrees for each update and the translation is made one grid step per update. After each update there is a check for attachment. The attachment routine cannot properly handle cases where the z-axes of two bricks are not parallel.

4. Discussion

Our initial user experiences indicate that our 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 have not conducted any formal usability studies yet, but a handful of people have tried the interface informally. The majority felt that they were able to manipulate the objects as intended. However many seemed to release the joypad button immediately on selection thinking that the object would still be attached to the phone. We believe that toggling selection in tangible mode by pressing the joypad button would introduce a source of error at the moment of release due to muscle contractions in the users' hand. This can be noticed slightly in the keypad mode when the object to be manipulated is selected. If the joypad button is not released immediately the selected object is likely to move due to fine motions of the users' hand. This is a common interface problem where selection and translation are performed in similar ways.

The main limitation is the tracking 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.

We believe our work 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 ported.

In fact there are currently few interaction devices that offers true 6 DOF input and most of these are used in advanced Virtual Reality environment requiring tracking. Our solution could provide an inexpensive 3D mouse solution for PCs by sending the position and orientation information via Bluetooth to the computer. Since the tracking is performed on the phone there is no computational overhead for the PC compared to a 2D mouse.

5. Future Work

The present platform allows a single user to manipulate position and orientation of 3D objects. In future versions we will extend this to allow collaboration between multiple users sending scene graph updates via Bluetooth. We also want to go beyond treating the objects as rigid and let the user edit the geometry. To do this we need to come up with a way to select individual vertices, edges and polygons. A rigorous user study with different tasks needs to be done..

6. Conclusion

We have presented an application that represents a new way of 3D interaction using mobile phones. We have built a general scene assembly application using an optimized port of ARToolKit. The application allows both isomorphic (keypad) and isometric (tangible) 6 DOF manipulations while allowing interactive viewing. The current restriction is the tracking range of the marker based optical tracking.

This is the first AR scene assembly program developed for the mobile phone. The lessons that we have learned developing this application and the feedback from users will allow us to develop further AR modeling and graphics applications for the mobile phone environment.

Acknowledgements

The first author is funded by the Research Center for High technology Construction (Brains & Bricks) as well as receiving supervisory support from the Swedish National Graduate School in Computer Science.

References

[1] 3D Blockout website: www.pda3dware.com/3dblockout.htm

[2] ARToolKit website: www.hitl.washington.edu/artoolkit/.

[3] Chen, M., Mountford, S., Sellen, A. (1988) A Study in Interactive 3-D Rotation Using 2-D Control Devices. Computer Graphics 22(4): 121-129.

[4] Feiner T. H. S., MacIntyre B. and Webster T. *A touring machine: Prototyping 3d mobile augmented reality systems for exploring the urban environment.* In Proc. ISWC '97 (First IEEE Int. Symp. On Wearable Computers), Cambridge, MA, 1997.

[5] 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).

[6] Hachet M,, Pouderoux J. and Guitton P.: A Camera-Based Interface for Interaction with Mobile Handheld Computers. To be published in proceedings of I3D'05 SIGGRAPH symposium on interactive 3D graphics and games, April 2005.

[7] Hakkarainen, M., Woodward., C., "SymBall - Camera driven table tennis for mobile phones", submitted to *ACM SIGCHI International Conference on Advances in Computer Entertainment Technology (ACE 2005)*, Valencia, Spain, 15-17 June, 2005 .

[8] Hansen, T.R. Eriksson, E., Lykke-Olesen, A., Mixed Interaction Spaces – Designing for Camera Based Interaction with Mobile Devices, short paper accepted at CHI 2005.

[9] Henrysson, A., Billinghurst, M., Ollila, M. Face to Face Collaborative AR on Mobile Phones. In Proceedings of ISMAR 2005, 2005 (To Appear) Vienna, Austria.

[10] Henrysson, A., Billinghurst, M., Ollila, M. Virtual Object Manipulation using a Mobile Phone. In Proceedings of ICAT 2005, 2005 (To Appear) Christchurch, New Zealand.

[11] Henrysson A. and Ollila M. *UMAR - Ubiquitous Mobile Augmented Reality* In Proc. Third International Conference on Mobile and Ubiquitous Multimedia

[12] KickReal website: http://www.kickreal.de/

[13] Klein, G. and Drummond T,: Sensor fusion and occlusion refinement for tablet-based AR, In Proc. of the IEEE and ACM ISMAR, 2004, pp. 38–47.

[14] Larsen, B., Bærentzen, J., Christensen, N. Using cellular phones to interact with virtual environments ACM Siggraph 2002, Conference Abstracts and Applications, pp. 274.

[15] Marble Revolution. http://www.bit-side.com/entertainment/MOBILE%20GAMES/Marble

[16] Moehring, M., Lessig, C. and Bimber, O. *Video See-Through AR on Consumer Cell Phones.* In Proc. of International Symposium on Augmented and Mixed Reality (ISMAR'04), pp. 252-253, 2004.

[17] Mogilev, D., Kiyokawa, K., Billinghurst, M., Pair, J. *.AR Pad: An Interface for Face-to-face AR Collaboration*, Proc. of the ACM Conference on Human Factors in Computing Systems 2002 (CHI '02), Minneapolis, pp.654-655, 2002.

[18] Mosquito Hunt website: http://w4.siemens.de/en2/html/press/newsdesk_archive/2003/foe03111.html

[19] OpenGL ES website: http://www.khronos.org/opengles/

[20] Piekarski, W. and Thomas, B. H. Tinmith-Hand: Unified User Interface Technology for Mobile Outdoor Augmented Reality and Indoor Virtual Reality. In IEEE Virtual Reality Conference, Orlando, Fl, Mar 2002.

[21] QR Code website: www.qrcode.com/

[22] Rekimoto J., TransVision: *A Hand-held Augmented Reality System for Collaborative Design.* Virtual Systems and Multi-Media (VSMM)'96, 1996.

[23] Rohs, M: Real-World Interaction with Camera-Phones 2nd International Symposium on Ubiquitous Computing Systems (UCS 2004), Tokyo, Japan, November 2004.

[24] Rohs, M., Gfeller, B.: Using Camera-Equipped Mobile Phones for Interacting with Real-World Objects. Advances in Pervasive Computing, Austrian Computer Society (OCG), ISBN 3-85403-176-9, pp. 265-271, Vienna, Austria, April 2004

[25] SpotCode website: http://www.highenergymagic.com

[26] Thomas, B., Close, B., Donoghue, J., Squires, J., De Bondi, P., Morris, M., and Piekarski, W. *ARQuake: An Outdoor/Indoor Augmented Reality First Person Application.* Proc. 4th Int'l Symposium on Wearable Computers, pp 139-146, Atlanta, Ga, USA, Oct 2000.

[27] Träskbäck M., Haller, M. Mixed reality training application for an oil refinery: user requirements., In ACM SIGGRAPH International Conference on Virtual Reality Continuum and its Applications in Industry, VRCAI 2004, pp. 324- 327, Singapore.

[28] Wagner, D., Schmalstieg, D.: *First steps towards handheld augmented reality.* Proc. of the 7th International Symposium on Wearable Computers (ISWC2003), White Plains, NY, USA, IEEE Computer Society (2003) 127–137.

[29] Wagner D., Pintaric T., Ledermann F., Schmalstieg D. Towards Massively Multi-User Augmented Reality on Handheld Devices. Proc. of the Third International Conference on Pervasive Computing (Pervasive 2005), Munich, Germany (to appear).

[30] Wagner, D., Barakonyi, I.: Augmented reality kanji learning. Proc. of the ISMAR 2003, Tokyo, Japan, IEEE Computer Society (2003) 335–336.

[31] Ware, C., and Jessome, D.R. "Using the Bat: A six-dimensional mouse for object placement," *IEEE Computer Graphics and Applications*, November 1988, pp. 65-70.

[32] Watsen K., Darken R. and Capps M.: A Handheld Computer as an Interaction Device to a Virtual Environment, 3rd International Immersive Projection Technology Workshop (IPTW'99), 1999.

[33] Wellner, P., Mackay, W. & Gold, R. Eds. Special issue on computer augmented environments: back to the real world. Communications of the ACM, Volume 36, Issue 7 (July 1993).