An Authoring Tool for Mobile Phone AR Environments

Yuan Wang[1]
HIT Lab NZ
University of Canterbury
Private Bag 4800,
Christchurch, NZ
+64 3 364 2349

Tobias Langlotz[2]
Graz University of Technology
Inffeldgasse 16
8010, Graz, Austria
+43 316 873 5011

Mark Billinghurst[3]
HIT Lab NZ
University of Canterbury
Private Bag 4800,
Christchurch, NZ
+64 3 364 2349

Tim Bell[4]
Dept. of CSSE
University of Canterbury
Private Bag 4800,
Christchurch, NZ
+64 3 364 2987

[2]langlotz@icg.tugraz.at, {[1]yuan.wang, [3]mark.billinghurst, [4]tim.bell} @canterbury.ac.nz

ABSTRACT

This paper describes an authoring tool for mobile phone Augmented Reality (AR) applications. This work is based on earlier work at the HIT Lab NZ on a tool for authoring PC based AR applications called ComposAR. In this paper, we describe modifications to ComposAR that allows end-users to prototype mobile AR applications on a PC, and mobile player software that will allow the prototype applications to be delivered on a mobile phone. In this way, end-users with little programming experience can develop simple mobile AR applications.

Categories and Subject Descriptors

H.5.2 [**Information Interfaces and Representation (HCI)**]: User Interfaces – *Graphical user interfaces, Prototyping*; J.9.c [**Computer Applications**]: Mobile Applications – *Multimedia applications and multimedia signal processing.*

General Terms

Interface Design, Human Factors.

Keywords

Augmented Reality, Authoring, Mobile Phones.

1. INTRODUCTION

Augmented Reality (AR) [1] is a technology that allows virtual images to be seamlessly mixed with the real world. In the past, AR applications needed high end PCs and specialized equipment such as fast cameras and head mounted displays, but recently the first AR applications have appeared on PDAs [10] and mobile phones [8]. The widespread use of camera-equipped mobile phones makes them a very attractive platform for deploying AR applications. However, creating mobile phone AR applications requires a lot of low level coding in C or C++ for operating systems such as Symbian or Windows Mobile. There are few high level authoring tools that allow developers to rapidly build mobile AR applications, and no tools for non-programmers.

In this paper, we describe an authoring tool for building mobile AR applications for non-programmers. First, we review related work in the area and then discuss our implemented tool.

2. RELATED WORK

Although there is no existing work on mobile AR authoring tools for non-programmers, there are several previous AR authoring tools for PC applications that can be built on. In this section, we first discuss AR authoring tools in general and then software libraries for mobile AR applications.

2.1 AR Authoring Tools

There are several existing authoring tools for building desktop AR applications. These can be broadly organized into two types: 1) AR authoring tools for programmers, 2) AR authoring tools for non programmers.

Authoring tools for programmers are typically code libraries that require programming knowledge, while tools for non-programmers include drag and drop interfaces for building applications without writing any lines of code. These categories can be further organized into low level tools which require coding/scripting skills, and higher level application builder tools which use higher level libraries or visual authoring techniques. Examples are shown in Table 1 and described later in this section.

Table 1. Types of authoring tools

	Programmers	**Non-programmers**
Low level	ARToolkit arTag[4]	DART[9] ComposAR [3]
High level	Studierstube[11] osgART [5]	AMIRE[6] MARS[7]

2.1.1 AR authoring tools for programmers

A number of programming libraries enable developers to author AR applications. For example, ARToolKit[1] is an open-source library which provides computer vision based tracking of black square markers. However, to develop an AR application with ARToolKit requires further code for 3D model loading, interaction techniques, and other utility functions.

An example of a high level programming library is Studierstube [11] which provides a complete system for developing AR applications. It includes all the functions needed for building an AR application such as scene graph rendering, networking, window management and support for input devices, etc.

A common feature of these libraries is that they typically require C or C++ programming ability, other development tools to produce the AR content, and it takes a relatively long time using them to produce an AR application.

[1] http://www.sourceforge.net/projects/artoolkit

2.1.2 AR authoring tools for non-programmers

There is another set of authoring tools that have been developed for non-programmers such as artists or designers. At the most basic level, tools such as BuildAR[2] allow users to associate virtual models with visually tracked AR markers. BuildAR only allows the user to scale, translate and position objects on the markers and see a live AR view. There is no support for object interaction or more complicated behaviours.

One of the first AR authoring tools to support interactivity is DART [9], the Designer's AR Toolkit, which is a plug-in for the popular Macromedia Director software. The main aim of DART is to support application designers. DART is built to allow non-programmers to create AR experiences using the low-level AR services provided by the Director Xtras, and to integrate with existing Director behaviours and concepts. DART supports both visual programming and a scripting interface.

AMIRE [6] is an authoring tool for the efficient creation and modification of augmented reality applications. The AMIRE framework provides an interface to load and to replace a library at runtime and uses visual programming techniques to interactively develop AR applications. AMIRE is designed to allow content experts to easily build applications without detailed knowledge about the underlying base technologies.

Some of these PC based authoring tools were also designed for building mobile AR applications, although not for mobile phones. For example, the MARS (Mobile Augmented Reality Systems) authoring tool [7] uses a 3D graphical user interface to allow users to create mobile outdoor AR applications using backpack mobile AR systems. It is designed for non-programmers, and allows them to preview their results on a desktop workstation, as well as with an augmented or virtual reality system.

A common feature of these tools for non-programmers is that they use visual programming techniques or simple scripting to support quick prototyping, they are interpretive rather than compiled allowing for fast redesign of ideas, and they are integrated into other design tools. However none of these tools can be used for authoring mobile AR applications.

2.2 Authoring Tools for Mobile Phones

Although there are several tools for building desktop AR applications, there is less support for mobile AR. At the low level, the ARToolKit tracking library has been ported over to the Symbian operating system [8] but this requires the use of other code such as the OpenGL ES graphics library in order to complete a mobile AR application. The Studierstube Tracker library [11] is another low level AR tracking library that is available for multiple platforms such as Symbian, iPhone and Windows Mobile.

One of the only higher level programming libraries for mobile AR applications is the Studierstube ES [12] (StbES) library. This is a low-level C++ based application framework for developing AR applications for mobile devices. It is cross-platform, running on Windows, Windows Mobile, or the Symbian operating systems. Studierstube ES provides support for 2D and 3D graphics, video capture, tracking, multimedia output, persistent storage, and application authoring. It requires a high level of programming skill to use and so is not suitable for non-programmers.

[2] BuildAR, http://www.hitlabnz.org/wiki/BuildAR

There are also Java based J2ME game engines, for example Ace3D[3]. The Ace3D Mobile Engine is designed to meet the needs of common 3D J2ME applications, including interactive 3D games. Although the structure of Ace3D is versatile, the limited hardware capabilities of modern mobile handsets reduce its functionality.

M3GE[4] (Mobile 3D Game Engine) is a Java game engine based on the Mobile 3D Graphics API for JME spec (M3G - JSR 184). It has a development library that allows graphical rendering to be handled by the application; image loading, input, output, and general functions like AI, collision detection and other rendering facilities are also managed. M3GE aims to perform all global functions in the application in a single core block, separating graphical routines and application logic.

For non-programmers, Python[5] is available for rapid development of mobile applications. The Symbian version of Python allows a user to develop python scripts on their desktop and then run them on their phone using a native interpreter. It has support for 2D and 3D graphics, camera input, file handling and networking, and many other functions for rapidly prototyping mobile applications. However it does not support a visual development tool and so requires the developer to learn scripting.

Other high level visual design tools are available to author mobile graphics applications. Among them, the most popular is Flash Lite[6], a version of the Adobe Flash Player that has been specifically designed for use on mobile phones. With FlashLite a developer can use a combination of visual authoring and ActionScript scripting to easily build interactive phone applications such as games, and e-learning applications. However there is no support for 3D graphics or camera input.

From Table 2, it can be seen that authoring tools currently available for mobile AR applications (such as Studierstube ES) requiring C++ programming experience. There are some high level tools for making mobile graphics applications for non-programmers (such as FlashLite), but none of them has been adapted for mobile AR yet. So there is a need to develop a high level mobile AR authoring tool for non-programmers.

Table 2. Authoring tools for mobile phones

	Programmers	**Non-programmers**
Low level	Studierstube Tracker [12] M3GE ARToolkit for Symbian [8]	Python
High level	Studierstube ES [11]	FlashLite

3. IMPLEMENTATION

In this section we describe the prototype of a tool we are working on for mobile AR authoring. Our goal is to develop a high-level tool that will allow non-programmers to build simple mobile AR applications. The basic approach we are following is to allow the user to author an AR scene on the desktop PC and then view the scene on a mobile phone in a mobile AR viewer. The next section describes this in more detail.

[3] Ace 3D Mobile Engine, http://www.program-ace.com/games/ace-tools/ace-3d-mobile-engine/

[4] Java Technology Collaboration, https://m3ge.dev.java.net/

[5] Python, http://www.python.org/

[6] Adobe Systems Incorporated, http://www.adobe.com/products/flashlite/

3.1 Overview of the System

There are two components to our system, a desktop PC authoring tool, and an AR viewer for either the PC or mobile phone.

The desktop authoring system is based on a modified version of the ComposAR [3] application developed at the HIT Lab NZ. ComposAR is a PC application that allows users to easily create AR scenes. It is based on osgART [5], and it is also a test bed for the use of scripting environments for OpenSceneGraph [2], ARToolKit and wxWidgets[7]. In our work we have developed a special version of ComposAR that provides an interface similar to a mobile phone. This makes it easier for the application developer to simulate the AR experience of mobile phones like a small screen and reduced keyboard size on a desktop PC.

In addition to creating a PC based authoring tool, it is necessary to create a viewing application for phones and desktop PCs, so that the developed application that can be viewed on a mobile phone outside the authoring environment. For this we will use the Studierstube ES library, a multi-platform framework for AR that can run on both PCs and mobile phones.

After finishing the authoring process the customized ComposAR creates an XML file as output, which specifies the AR scene content in the application. An AR viewing application based on the Studierstube ES library reads the XML file and renders the AR scene on a mobile platform. Thus, the end-users are able to author the application on a PC and run it on a mobile phone. Figure 1 shows how the components of the system are related.

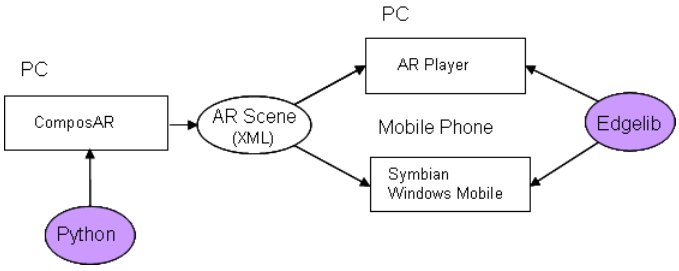


Figure 1. The structure of the mobile AR authoring system.

3.2 ComposAR Customization

In order for ComposAR to be used for developing mobile AR applications, it needs to be modified to emulate the small screen size and limited input options of mobile phones. The Python language is used to develop the overall ComposAR system, so the ComposAR interface can be changed using Python code.

A simplified graphical user interface (GUI) for ComposAR has been developed to match the form factor of the target mobile phone (See Figure 2). It is composed of a live video view of the scene with the same resolution as the typical mobile phone camera (320x240 pixel or 640 x 480 pixels), and a virtual keypad that emulates a mobile phone keypad. With this GUI, the end-users can associate 3D virtual models with real AR tracking markers. In addition, it allows users to add simple interaction to a single marker or a set of markers.

Figure 2 shows the four panes in the new ComposAR interface: (a) Scene pane (b) Augmented Reality Scene pane (c) Keypad pane and (d) Script pane.

[7] wxWidgets: http://www.wxwidgets.org/

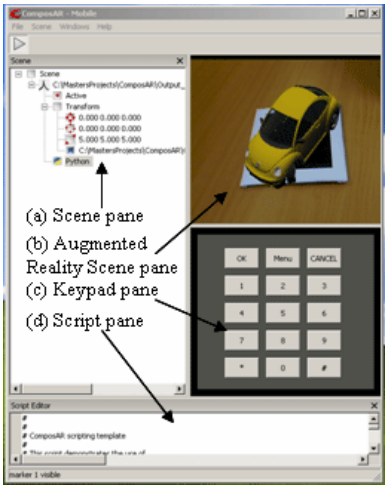


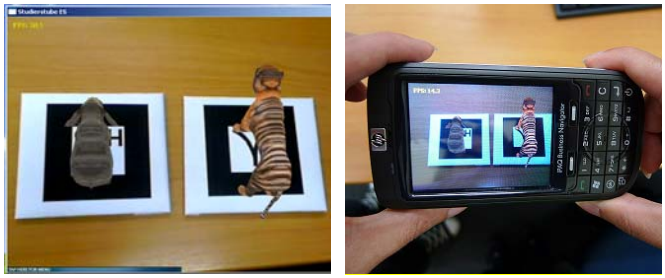
Figure 2. The ComposAR-Mobile Interface.

The Scene pane enables the designer to select markers and 3D models stored on the local system and to create links between markers and 3D models. Once the marker and corresponding 3D models are linked it is possible to change the position, rotation and scale of the assigned 3D models in the AR scene. An interaction script in Python can be written in the Script pane for specifying the virtual object interactions in an AR scene. Keypad based interaction within the AR scene can be simulated using the virtual keypad in the Keypad pane. An example code for integrating the keypad pane in the python code can be seen in Figure 3. One of the advantages of using ComposAR is that scripts are interpreted so that immediate feedback on the fly can be seen from the Augmented Reality Scene pane if any contents of it are updated.

```
def OnOne( self,event ):
    path = os.path.join(Utils.get_modelpath(),"cessna.osg")
    obj_info = self.canvas.scene.findMarker(0)
    print obj_info
    path2 =obj_info.replaceModel(path)


def OnTwo( self,event ):
    path = os.path.join(Utils.get_modelpath(),"spaceship.osg")
    obj_info = self.canvas.scene.findMarker(0)
    print obj_info
    path2 =obj_info.replaceModel(path)

def OnThree( self,event ):
    path = os.path.join(Utils.get_modelpath(),"camel.lwo")
    obj_info = self.canvas.scene.findMarker(0)
    print obj_info
    path2 =obj_info.replaceModel(path)

def OnFour( self,event ):
    path = os.path.join(Utils.get_modelpath(),"dumptruck.osg")
    obj_info = self.canvas.scene.findMarker(0)
    print obj_info
    pa 2 =ob info replac Model ath)
```

Figure 3. The python code for keypad functionality.

We have also developed an AR viewing application for the PC to test the XML output generated and to view the AR scene. The PC AR viewer is based on an integration of ARToolKit with the Studierstube ES rendering library.



(a) **(b)**

Figure 4. The PC and mobile phone AR viewing application.

Figure 4 shows a screen snapshot for the AR viewing. The desktop PC AR viewing is shown in (a) while mobile phone AR viewing is shown in (b).

3.3 Mobile AR Viewing Tool

To replay the created AR scene from the PC based authoring tool, we developed a mobile AR viewing tool that runs on a mobile phone. This was implemented using the Studierstube ES framework which is a set of cross platform libraries for developing AR applications. Studierstube ES was designed with the focus on enabling AR for small form factor devices such as smart phones and other handheld devices. All processing is done natively on the handheld device, so that applications can run independently of any infrastructure.

Studierstube ES uses Studierstube Tracker to track the camera in 3D. Studierstube Tracker support about 6 different marker designs. Due to the fact that ComposAR is not based on Studierstube ES we are limiting our current implementation to markers, which are supported in both applications. The speed of the prototyped applications on current smartphones is depending on the shown 3D content and the number of visible markers and is typical in the order of 5-30 fps on current smartphones.

The AR scene description created by our customized version of ComposAR is the input for our implemented Studierstube ES based viewer application. It configures the viewer and defines the used tracking system as well as the used marker and 3D content. While the majority of the configuration settings are shared between the desktop and the mobile system there are still specific parameters for each platform (e.g. window size and camera calibration files). Thus it is necessary to produce a configuration for both; the mobile and the desktop setup. The viewer application chooses at runtime the appropriate configuration. Figure 4 shows a rendering of the same composed AR scene on a smartphone and on a desktop PC.

4. CONCLUSION AND FUTURE WORK

This paper describes early work on developing a high level AR application for mobile environment. With this approach, quick prototyping of the GUI and the necessary components can be quickly assembled to create an end-user application.

In the future we will use the Studierstube ES for Symbian interpreter to read in python code for the AR application. An evaluation study will also be conducted, focusing on ease of use, the features that should be added, the types of applications build and the kind of functionality should be added.

5. REFERENCES

[1] Azuma,R., Baillot,Y., Behringer,R., Feiner,S.,Julier, S., and MacIntyre, B. 2001. Recent advances in augmented reality. Computer Graphics and Applications, IEEE 21, 34-47.

[2] Burns, D. and Osfield, R. 2004. Open Scene Graph A: Introduction, B: Examples and Applications. In Proceedings of the IEEE Virtual Reality 2004 (March 27 - 31, 2004). VR. IEEE Computer Society, Washington, DC, 265. DOI= http://dx.doi.org/10.1109/VR.2004.57

[3] Dongpyo, H., Looser, J., Seichter, H., Billinghurst, M., and Woontack, W. 2008. A Sensor-Based Interaction for Ubiquitous Virtual Reality Systems. In Ubiquitous Virtual Reality, 2008. ISUVR 2008. International Symposium on, 75-78.

[4] Fiala, M. 2005. ARTag, a fiducial marker system using digital techniques. In Computer Vision and Pattern Recognition, 2005. CVPR 2005. IEEE Computer Society Conference on. 590-596 vol. 592.

[5] Grasset, R., Looser, J., and Billinghurst, M. 2005. OSGARToolKit: tangible + transitional 3D collaborative mixed reality framework. In Proceedings of the 2005 international Conference on Augmented Tele-Existence (Christchurch, New Zealand, December 05 - 08, 2005). ICAT '05, vol. 157. ACM, New York, NY, 257-258.

[6] Grimm, P., Haller, M., Paelke,V., Reinhold, S., Reimann, C., and Zauner, R. 2002. AMIRE - authoring mixed reality. In Augmented Reality Toolkit. The First IEEE International Workshop, 2 pp.

[7] Guven, S. and Feiner, S. 2003. Authoring 3D hypermedia for wearable augmented and virtual reality. In Wearable Computers, 2003. Proceedings. Seventh IEEE International Symposium on, 118-126.

[8] Henrysson, A., Billinghurst, M. and Ollila, M. 2005. Face to face collaborative AR on mobile phones. In Mixed and Augmented Reality, 2005. Proceedings. Fourth IEEE and ACM International Symposium on, 80-89.

[9] MacIntyre, B., Gandy, M., Dow, S., and Bolter, J. D. 2005. DART: a toolkit for rapid design exploration of augmented reality experiences. In ACM SIGGRAPH 2005 Papers (Los Angeles, California, July 31 - August 04, 2005). J. Marks, Ed. SIGGRAPH '05. ACM, New York, NY, 932-932.

[10] Pasman, W., and Woodward, C. 2003. Implementation of an Augmented Reality System on a PDA. In Mixed and Augmented Reality, 2003. Proceedings. The Second IEEE and ACM International Symposium on, 276-277.

[11] Schmalstieg, D., Fuhrmann, A., Hesina, G., Szalavári, Z., Encarna, L. M., Gervautz, M., and Purgathofer, W. 2002. The studierstube augmented reality project. Presence: Teleoperators & Virtual Environments 11. 33-54.

[12] Schmalstieg, D. and Wagner, D. 2008. Mobile Phones as a Platform for Augmented Reality. In Proceedings of the IEEE VR 2008 Workshop on Software Engineering and Architectures for Realtime Interactive Systems, IEEE, Reno, NV, USA: Shaker Publishing, 43-44.