Augmented Reality Authoring: Generic Context from Programmer to Designer

Alastair Hampshire
School of Computer Science and Information Technology
University of Nottingham
Jubilee Campus
Wollaton Road
Nottingham NG8 1BB
United Kingdom
axh@cs.nott.ac.uk

Hartmut Seichter, Raphaël Grasset, Mark Billinghurst
HIT Lab NZ
University of Canterbury
Private Bag 4800
Christchurch
New Zealand
{Hartmut.Seichter|Raphael.Grasset.Mark|Billinghurst}@hitlabnz.org

ABSTRACT
Developing an Augmented Reality (AR) application is usually a long and non-intuitive task. Few methodologies address this problem and tools implementing these are limited or non-existent. To date there is no efficient and easy development tool tailored to the needs of Mixed Reality (MR). We are presenting an initial taxonomy of MR applications, addressing the different levels of abstraction for defining the relation between real and virtual world. We then demonstrate some development approaches and describe tools and libraries that we implemented in order to illustrate aspects of our authoring taxonomy. Finally, we provide a definition addressing the requirements for new generation of AR rapid application development (RAD) tools based on actual implementations.

Author Keywords
Augmented Reality, Authoring, RAD, Software Framework

ACM Classification Keywords
H.5.1[Information Interfaces and Presentation]:Multimedia Information Systems—Artificial, augmented, and virtual realities; D.2.2 [Software Engineering]: Design Tools and Techniques—Software libraries.

INTRODUCTION
Mixed Reality (MR) and Augmented Reality (AR) define a new type of environment where real and virtual converge. The last decade has seen a large increase of the interest of this technology; some tools have been proposed to reply to the development of AR applications, like the well renowned *ARToolKit* (Kato and Billinghurst, 1999), which is largely used in the research community.

OZCHI 2006 Proceedings ISBN: x-xxxxx-xxx-x

But this first response to the needs for AR tools is quite partial and unsteady. Firstly, because the requirements for an AR application remains undefined from both a usability viewpoint and a software architecture one. Secondly, few methodologies or specifications are actually recognized or have been largely used for AR applications (Dubois et al., 2001). Lastly, the sparse propositions and demonstrations of these frameworks or tools can not be categorised in a general and a coherent context.

We propose a first taxonomy of development tools for AR applications. Based on this general context, we can therefore structurally explore different items of this framework. We will describe in this paper different tools we have implemented, their advantages and limitations we identified during industrial demonstrations, academic projects or teaching course.

TAXONOMY AND RELATED WORKS
Designing content for MR is driven by the need to define and fuse the relationship between entities in physical world and virtual world (MacIntyre, 2002). These relations can be described for a large part through the model of affordances (Gibson 1979, Norman 1988) due to the fact that they create an additional or an overlayed property to a physical object (Seichter, 2005). However, in MR the relation between physical and virtual world can change dynamically and it depends on the level of integration between reality and virtuality (Milgram and Kishin, 1994) how a user perceives an augmented environment.

Authoring tools can provide different levels of assembly functions and control over the relationships between the real and virtual objects. Similar to conventional digital content creation a trade-off had to be made between low level, programming driven systems and high level content based systems, affecting various aspects of the actual product (see Figure 1). This conceptional view provides different levels of control by predefining underlying concepts.

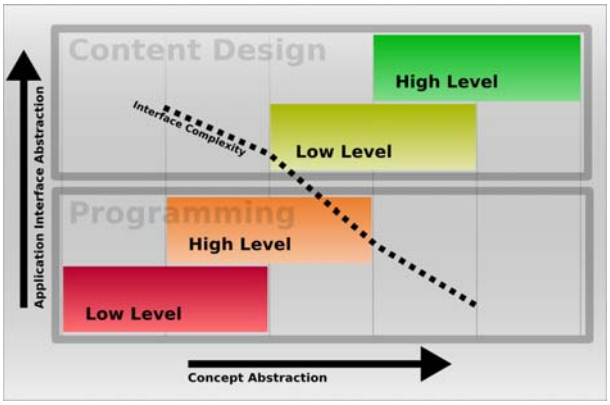

Figure 1: Schematic view on digital media authoring

Low level programming frameworks in AR implement APIs for core tasks like computer vision, visual and spatial registration of objects, generic 3D rendering and the like (Uchiyama 2002, Dias, 2003). Their thin level of abstraction yields a high degree of performance and flexibility but at the cost to require the user defining manually the interaction techniques, visualisation and simulation aspects. Logically, *high level programming frameworks* address this issue by generalising common aspects and provide them for an authoring task through a generalised meta structure. In MR, and therefore inherently also in AR, high level programming libraries address conceptual aspects like 3D graphic, sound and various input/output through a generalised API (Bauer et al., 2001; Schmalstieg et al., 2002; Grasset and Gascuel, 2002).

Low level content design frameworks provide another level of abstraction with removing the direct reference to a programming language and replace it with a datadriven model. This model addresses directly the content and its relationship within a MR environment. Hence, the description is still programmatic but relates to content. An example for this approach is APRIL (Ledermann and Schmalstieg, 2002).

Based on this approach the description of the content itself becomes an aspect of authoring. Graphical user interfaces like AMIRE or DART (Dörner et al., 2002; MacIntyre et al., 2003) help to provide this additional abstraction by hiding the actual programmatic description of the content.

All of these approaches build upon each other. Depending on the implementation, abstraction is added and low level functionality is removed or hidden. Thus, it is less likely that a content driven authoring tool will provide an author with control over computer vision aspects. However, the higher the abstraction the more conceptual models are included within the authoring environment providing the user with templates for common tasks within an AR environment.

LIBRAIRIES AND TOOLS

Based on this framework, we tried to explore the different layers based on explorative study and case studies on different type of projects. We therefore developed and enhanced different frameworks and present them here in this section.

Low Level Programming Framework: ARToolKit

The HIT Lab NZ has developed and enhanced ARToolKit over the past few years. It has given researchers easy access to AR and is been used in our laboratory for various projects (e.g. research or teaching courses related to AR) and also in industry, education and medicine. The ARToolKit implements a C interface based on a modular toolkit structure, which is derived from tasks in AR. The three modules are video capture, tracking and rendering. In its latest version it includes support for physical input, which enables users to enrich their AR applications using tangible user interface (TUI) with sliders, dials, etc.

An analysis of the ARToolKit community through our forum, mailing list and feedback collected during our in-house development projects, helped us to cluster some initial remarks on this platform. The simplicity of the programming interface and the all-in-one (*hidden for review*) addresses a broad spectrum of needs for the development of simple demonstrations and lowers the learning curve on AR application mechanisms.

However, the use of C programming language for the API and the lack of robust extensions limit the use of ARToolKit in large projects and in industrial application which require more flexibility.

High Level Programming Framework: OSGART

In response to the limitations we observed by working on the previous toolkit for complex projects, we developed a simple and efficient scene-graph based framework, named *OSGART*. Implemented in C++, we focused here to propose a simple and intuitive framework following the same principles of *ARToolKit* by providing an easy way for prototyping AR applications. By choosing a more generalising approach and fostering the flexibility of the underlying libraries our software architecture gained a magnitude of usability.

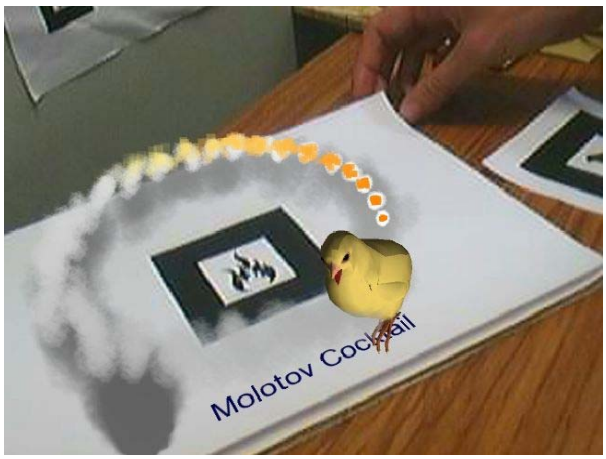

Figure 2: MagicBook art installation presented at Experimenta Vanishing Project (Australia).

In addition to the abstraction of various AR concepts the *OSGART* framework provides a high level programming approach possible through the usage of a scripting environment like Python, Ruby or Lua. Large scripting environments like Python provide a rich set of infrastructural tools. Thin low level scripting languages like Lua are integrated in order to change runtime behaviour for computing intensive and often updated branches in the framework. This way we make optimal use of the different capabilities of runtime interpreted programming without compromising functionality or speed.

Recently, our framework has been deployed in various projects from industrial demonstrations to art installations (CONVERGE'05), museum exhibits (*Experimenta Vanishing Project*, see Figure 2) or research projects. Our first observations show a large speed up on the integration of a large diversity of multimedia content, helping the developer to focus more on the application than low level problems (tracking, animation, video problems, etc.). An actual academic project has also shown the fast development for prototyping and easy learning curve of the scripting interface even for users without experience on AR development.

Low Level Designing Framework: ECT

The Equator Component Toolkit (ECT) (Greenhalgh et al., 2004) is a component oriented software toolkit with a visual programming interface designed to ease the process of constructing ubicomp experiences. ECT provides a set of software components which accomplish common tasks such as interaction with IO hardware and 3D output; component interfaces consist of a set of properties (rather than a set of operations). This underlines the content orientation of this approach.

Users select and create the desired components from a list of available components. Applications are then constructed by linking properties, e.g. property A on component X to property B on component Y. Subsequent changes to property A will be propagated to property B. The ECT Graph Editor (see figure 3) allows the user to configure created components by setting property values and visually connecting components as described above.

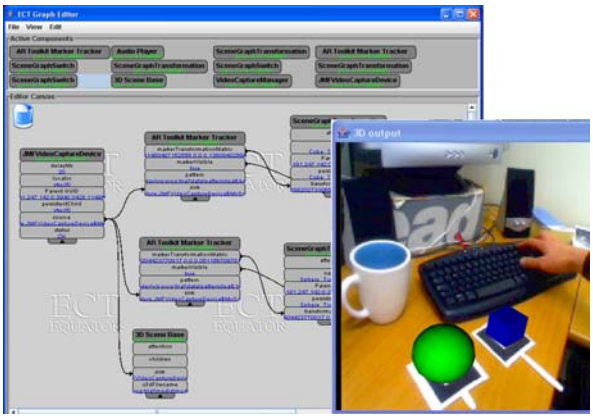

Figure 3: The ECT Graph Editor allows visual construction of MR applications.

ECT has already been used by a number of designers with little or no computer programming experience to allow the construction of computer applications. A human-centred research group at Sussex University used ECT to create an engaging 'digital skipping' experience for children which was displayed at a science-arts festival called the 'Big Blip'. The 'experience builders' had little or no experience of sequential programming techniques, yet where able to construct the required experience using components created by a software engineer.

A number of components have been created at our laboratory to allow ECT to support AR authoring. Whilst this is not a definitive list of components required for AR authoring (development work continues to add new components and functionality), the following components that provide a base level of AR authoring functionality are currently available:

- A 3D output component, capable of loading, displaying and manipulating a 3D scene. This allows a graphics designer to build a 3D model using a complex 3D modelling package, whilst permitting an interaction designer to modify content and implement behaviours.
- A video capture component which provides video input into ECT applications.
- An AR marker tracking component based on *ARToolKit* which calculates the transformation required to make an object appear on an AR marker. The transformation value produced by an AR marker tracking component can be linked to parts of a 3D scene to associate a 3D model with a particular marker.
- A component which calculates the relative position and orientation of two AR markers. This could be used to, for example, trigger the animation of a virtual object when two markers are close.
- An audio player component providing sound playback.
- A component which interfaces with a hardware IO device allowing sensors and tangible input devices to manipulate virtual objects in an AR application.

Developing extensions for ECT is straight forward and the system in its current implementation can foster the large infrastructure of the Java platform. In a next step the editor will be evaluated in a user centric test.

DISCUSSION

Our taxonomy from low level programming toolkit to low level content creation has been implemented in above libraries and tools addressing different target audiences with a generalised set of use cases. One common issue yet to be addressed is that of user control over specific components. The need for fine grain control is needed when the user wants to deploy its application or has reached over time a higher level of expertise and tries to tune components of the framework that are meant to be in a level that is underneath the current.

Currently we are also reviewing possibilities to implement an authoring tool that can also be used by

laymen. At the current point it seems that such a tool again need to be tailored to a user group. A product designer has other needs to compose AR content then a teacher who creates a MagicBook (Billinghurst et al., 2001). Thus, in the highest level of abstraction for that application the abstraction of concepts becomes domain specific. Tools like AMIRE have attempted to create such a utility but have failed to become a common standard due to the fact that the domain unspecific implementation introduces a high complexity in the user interface as it refuses to make any assumptions.

CONCLUSIONS

We introduced in this paper a taxonomy representing the different approaches on digital media authoring for AR applications. We successively explored the first layers, providing some feedback about adapted tools for dedicated case and applications.

We are currently developing a high level design tool (IDE) that will facilitate high level content design. In a first step we explore this approach in the context of educational programs. We hope to obtain a fully functional and explorative framework that will guide us after a thorough analysis to refine our taxonomy and help us to create guidelines and requirements for AR development platforms.

REFERENCES

Bauer, M., Bruegge, B., Klinker, G., MacWilliams, A., Reicher, T., Riss, S., Sandor, C. and Wagner, M. Design of a component-based Augmented Reality Framework. In Proc. ISAR 2001 (2001).

Billinghurst, M., Kato, H. and Poupyrev, I. The MagicBook: a transitional AR interface. Computers & Graphics 25(5) (2001), 745-753.

Dias, J.M.S., Santos, P., Bastos, R., Monteiro and L., Silvestre, R. Developing and Authoring Mixed Reality with MX Toolkit. In Proc. International Augmented Reality Toolkit Workshop, (2003).

Dörner, R., Geiger, C., Haller, M. and Paelke, V. Authoring Mixed Reality. A Component and Framework-based approach. In Proc IWEC 02, (2002).

Dubois, E., Gray, P.D. and Nigay L. ASUR++: a Design Notation for Mobile Mixed Systems. In Proc. MobileHCI'02, (2001), 123-139.

Gibson, J.J. The Ecological Approach to Visual Perception. Houghton Mifflin, Boston (1979).

Grasset, R. and Gascuel, J.P., Mare : Multiuser augmented reality environment on table setup. In Proc. ACM SIGGRAPH Conference Abstracts and Applications (2002).

Greenhalgh, C., Izadi, S., Mathrick, J., Humble, J. and Taylor, I. *A Toolkit to Support Rapid Construction of Ubicomp Environments*. In Proc UbiSys 2004 (2004)

Kato, H. and Billinghurst, M. Marker Tracking and HMD Calibration for a video-based Augmented Reality Conferencing System. In Proc. IWAR 1999 (1999).

Ledermann, F. and Schmalstieg, D., "APRIL: A High-Level. Framework for Creating Augmented Reality Presentations, Proc. IEEE Virtual Reality 2005, 2005

MacIntyre, B., Gandy, M., Bolter, J., Dow, S., and Hannnigan, B. DART : The Designer's Augmented Reality Toolkit. In Proc ISMAR'03, (2003).

MacIntyre, B. Authoring 3D Mixed Reality Experiences: Managing the Relationship Between the Physical and Virtual Worlds. In Proc. ACM SIGGRAPH and Eurographics Campfire: Production Process of 3D Computer Graphics Applications - Structures, Roles and Tools. (2002).

Milgram, P. and Kishin, F. A taxonomy of mixed reality visual displays. IEICE Transactions on Information and Systems, 77(9) (1994) 1321–1329.

Norman, D. The Psychology of Everyday Things. Basic Books, New York (1988).

Schmalstieg, D., Fuhrmann, A., Hesina, G., Szalavari, Z., Encarnacão, L.M., Gervautz, M. and Purgathofer, W.. The Studierstube Augmented Reality Project. Presence - Teleoperators and Virtual Environments, 11(1) (2002) 33–54.

Seichter, H. Assessing Virtual Tangibility In Proc. CAAD futures'05, Vienna, Austria, (2005) 151-159.

Uchiyama, S., Takemoto, K., Satoh, K.,Yamamoto, H. and Tamura, H. MR platform: A basic body on which mixed reality applications are built. In Proc. ISMAR'02 (2002).