Towards Massively Multi-User Augmented Reality on Handheld Devices

Daniel Wagner[1], Thomas Pintaric[1],
Florian Ledermann[2] and Dieter Schmalstieg[1]

[1] Graz University of Technology
{wagner,pintaric,schmalstieg}@icg.tu-graz.ac.at
[2] Vienna University of Technology
ledermann@ims.tuwien.ac.at

Abstract. Augmented Reality (AR) can naturally complement mobile computing on wearable devices by providing an intuitive interface to a three-dimensional information space embedded within physical reality. Unfortunately, current wearable AR systems are relatively complex, expensive, fragile and heavy, rendering them unfit for large-scale deployment involving untrained users outside constrained laboratory environments. Consequently, the scale of collaborative multi-user experiments have not yet exceeded a handful of participants. In this paper, we present a system architecture for interactive, infrastructure-independent multi-user AR applications running on off-the-shelf handheld devices. We implemented a four-user interactive game installation as an evaluation setup to encourage playful engagement of participants in a cooperative task. Over the course of five weeks, more than five thousand visitors from a wide range of professional and socio-demographic backgrounds interacted with our system at four different locations.

1 Introduction and Related Work

Augmented Reality (AR) can naturally complement mobile computing on wearable devices by providing an intuitive interface to a three-dimensional information space embedded within physical reality [1]. Correspondingly, the human-computer interfaces and interaction metaphors originating from AR research have proven advantageous in a variety of real-world mobile application scenarios, such as industrial assembly and maintenance, location-based information systems, navigation aides, computer-supported cooperative work, and entertainment.

However, existing AR systems aiming at unconstrained mobility, like the Touring Machine [2], MARS [3], or Tinmith [4] have typically emerged as wearable variants of existing desktop setups. Their creators would commonly package a single (notebook) computer with a variety of sensors and peripheral devices in such a way that users can wear it on their backs, with both arms through shoulder straps. Graphical augmentations were usually shown through

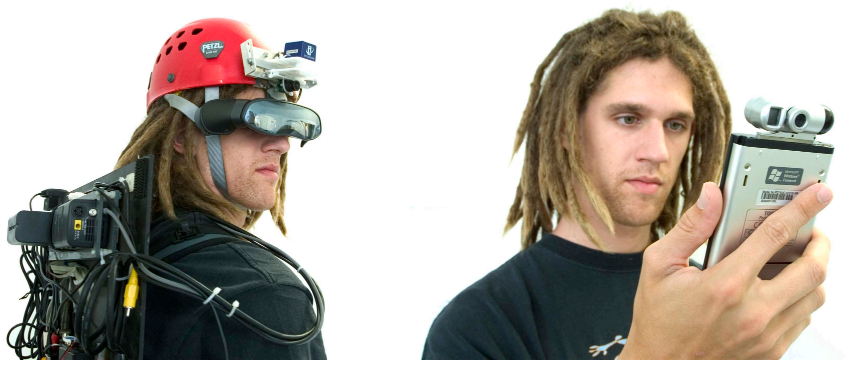

Fig. 1. A typical "backpack" setup (left) for Mobile AR versus a lightweight handheld device (right).

an optical see-through head-mounted display (HMD). A typical example of this type of setup is depicted in figure 1.

Although these "backpack" systems have been successful proof-of-concept prototypes within their respective range of applications, they are maintenance-intensive and lack the robustness demanded by permanent deployment outside a constrained laboratory environment. Their prohibitive cost not only prevents the development of a commercial market (and with it widespread availability) for the foreseeable future, but also thwarts researchers in their plans to conduct multi-user experiments exceeding a handful of participants. Finally, there are situations and social contexts in which the unwieldy and rather conspicuous AR "backpacks" seem impractical, or even inadequate.

Part of the usability and scalability issues of these mobile yet monolithic wearable AR systems have been addressed by other lines of research investigating the use of lightweight wearable devices as thin clients supported by a ubiquitous computing infrastructure. Instead of delivering graphical overlays by means of a head-mounted display, those systems convey information using the magic-lens [5,6] property of camera-equipped portable displays, such as keyboard-less personal digital assistants (PDAs) built for pen-computing. Rekimoto used a tethered analog display and a CCD camera in the NaviCam [7] project to track color-coded fiducial markers on a connected workstation. The Batportal [8] by Newman et al. was driven by a remote VNC display server using ultrasonic outside-in tracking, while the AR-PDA project [9] relied on a digital image streaming mechanism in order to outsource machine vision and graphics rendering tasks to an application server. In summary, the inherent infrastructure-dependence of thin-client approaches has confined these projects to restricted working volumes, thereby preventing them from evolving into self-sufficient mobile AR systems.

We believe there is a need for an unconstrained, infrastructure-independent AR system running on lightweight wearable devices to "bridge the gap" in situations where traditional “backpack” systems are too costly and unnecessarily cumbersome, but thin-client implementations exhibit inadequate deployability, scalability or interactive behavior. Particular examples include sporadic use over lengthy time spans, in between which devices must be stowed away, mixed indoor/outdoor use in wide-area environments, and massively multi-user application scenarios. This has motivated us to develop our own software framework targeting lightweight handheld devices.

2 System Architecture

We have identified three distinct classes of commercially available wearable computers as potential candidates for a stand-alone handheld AR platform: cellular phones, PDAs and Tablet PCs. All of these device designs make specific tradeoffs between size, weight, computing power and cost. While cellular phones are extremely portable and widespread, their current lack of adequate processing power and local network connectivity renders them a suboptimal platform for rich and meaningful AR applications. Furthermore, their small display size and limited data input capabilities are less than ideal for three-dimensional user interfaces. Tablet PCs do not share the aforementioned drawbacks but are considerably more expensive and too heavyweight for single-handed, or even prolonged two-handed use. Taking into account all of these constraints, we chose the PDA as target platform for our Handheld AR framework. PDAs are a good compromise between processing power, size and weight; they are socially acceptable and their touch screen input mechanism is familiar, which we considered crucial for our planned deployment of AR applications to untrained users.

We have devised a component-based software architecture to accelerate the task of developing and deploying collaborative applications on handheld devices. At its center lies a lightweight variant of the *Studierstube* [10] framework that has been stripped of all the functionality deemed unnecessary for running applications on mobile devices, where computing resources are scarce. Concurrent multi-application execution and application-migration capabilities were among the features we omitted. The blueprint of our framework's design was published in an earlier paper [11], but has since undergone major revisions.

Studierstube extends OpenInventor[3], a scene-graph rendering library, and uses OpenTracker [12], a modular dataflow middleware for pose tracking. Both, *Studierstube* and OpenTracker make use of ACE (the Adaptive Communication Environment)[4] for network communication abstraction. *Studierstube* allows application development in C++ or, more rapidly, via OpenInventor scripts. Alternatively, application developers can choose to use APRIL [13], a high-level descriptive language for authoring story-driven AR presentations.

There are several software libraries that provide hardware abstraction mechanisms on PDAs, however, we considered them insufficient for our purposes as

[3] SGI OpenInventor, http://oss.sgi.com/projects/inventor

[4] The ADAPTIVE Communication Environment,
http://www.cs.wustl.edu/ schmidt/ACE.html

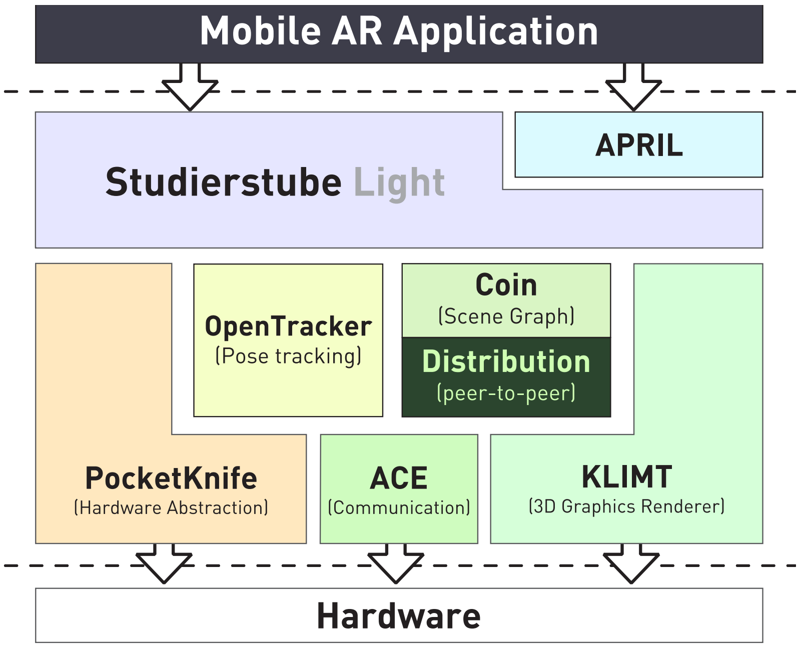

Fig. 2. Software architecture overview

they were primarily intended for 2D game development. Hence, we developed our own tool, called PocketKnife, which resembles a collection of classes designed to facilitate the development of graphical applications on mobile devices.

Since no OpenGL-compatible 3D graphics libraries for mobile devices were available at the time, we developed our own software renderer, called Klimt[5]. Its API is very similar to that of OpenGL and OpenGL|ES. Applications built with Klimt are able to run Coin[6], an OpenInventor-compatible scene-graph rendering library, which we ported to Windows CE. Klimt is able to render up to 70,000 triangles and up to 11 million textured pixels per second on a standard PocketPC PDA with a 400Mhz Intel XScale processor.

Network connectivity is central to any distributed system. Collaborative AR applications share scene- as well as application-specific data. Hence, a reliable, proven communication framework is essential in order that developers can concentrate on higher-level design aspects. ACE includes a wrapper library that provides platform-independent access to system resources, such as network sockets, threads and shared memory. ACE simplifies the development of object-oriented network communication code, shields developers from directly programming the

[5] Klimt an Open Source 3D Graphics Library for Mobile Devices, http://studierstube.org/klimt
[6] Coin3D scene graph renderer, http://www.coin3d.org

socket API and instead allows concentrating on application-level architecture design.

Tracking is an indispensable requirement for every Virtual Reality (VR) and Augmented Reality (AR) system. While the quality of tracking, in particular the need for high performance and fidelity have led to a large body of past and current research, little attention is typically paid to software engineering aspects of tracking software. OpenTracker was developed as a generic framework for accessing and manipulating tracking data. It implements the well-known *pipes & filters* dataflow pattern and provides an open software architecture based on a highly modular design and an XML configuration syntax. OpenTracker incorporates drivers to most commercial tracking devices. Using OpenTracker, AR developers can configure their tracking setup (including data fusion from multiple sources) with a few lines of XML code. Switching between different tracking setups does not require changes to the application code, instead, editing of a single XML file is sufficient. Since OpenTracker provides a network data-transport mechanism, mobile devices can easily access outside-in tracking hardware via Wireless LAN. Independence from pose tracking infrastructure was achieved by porting the ARToolKit[7] library to WindowsCE, which can be accessed in a generic way through a specialized module within OpenTracker. ARToolKit can process up to 150 images per second natively on a PDA equipped with a 624MHz XScale processor (depending on the number and size of the fiducial markers visible in the video image). Alternatively, the processing task can be outsourced to a server.

Programming tools for PDA-development are error-prone, and debugging remains a slow and cumbersome task — even on device emulators. Consequently, developers attempt to write and test most of their code natively on a workstation. All previously described components are available for PDAs running WindowsCE, as well as regular PC-based workstations running Windows 2000/XP, which greatly facilitates the software development process. The hardware abstraction module allows software engineers to develop applications on the PC, while only the final testing is done on the PDA itself. Finally, all of the software described above is available under an open source license. Readers interested in evaluating our framework are invited to obtain further information and a copy of the software from our website[8].

3 Application Requirements

In order to assess the practical deployability and usability of our framework, we considered it imperative to conduct a field test in which as large a number of users as possible would be asked to try their hand at a PDA-based AR application. Ideally, some participants would not have had prior experience with AR interfaces.

[7] ARToolKit computer vision library, http://www.hitl.washington.edu/artoolkit
[8] The Handheld Augmented Reality Project, http://studierstube.org/handheld_ar

The application that we eventually presented to end users in the evaluation was chosen according to several criteria: first and foremost, the application should expose our framework's major features and key properties to the end user while simultaneously allowing us to draw early conclusions about the practical value of our developments. While the application would have to be designed for a high anticipated volume of use, it should not offer a reduced experience to users who would only try it briefly. Finally, the application task should be kept simple and straight forward in order to avoid discouraging participation from non-technical audiences.

While it would have been possible to draw inspiration from a number of marker-based augmented reality applications published by other researchers, none of those met all of our requirements or made use of the unique possibilities gained by bringing marker-based AR to handheld devices. Naturally, we were looking for an interactive application that would allow its users to participate in a collaborative or concurrent task. The application should be distributed, synchronizing state between multiple clients through wireless networking. Many marker-based AR applications that have so far been presented make heavy use of fiducials as tangible interface components, allowing their users to flip through the pages of marker-enhanced "magic books" [14], to use markers as cards in an augmented memory game [15], or to use markers for positioning various objects such as design elements or video surfaces in the user's workspace [16, 17].

In contrast to these applications, which focus on the use of fiducial markers as moveable, dynamic parts of the application, we sought to employ the handheld's tracked display itself as the tangible, dynamic interaction element. Therefore, we decided to focus on pen-based touch-screen input as the main interaction technique. A static arrangement of multiple fiducial markers in the environment would enable the PDAs to perform self-tracking from a variety of different angles and distances. Another important requirement was that the application should be sufficiently spatially distributed to give an impression of the properties of our tracked display surface with respect to panning and zooming interactions — users should be required to move in closely with environment to discover important details, and to move the perspective away from the setting in order to gain an overview of the scene. This differs from other applications such as the magic book, which are designed to be fully visible within the field of view of the user, and therefore require no navigational actions from the user. Usually, marker-based tracking techniques are sufficiently accurate for computing a marker's position and orientation relative to a camera, but not to perform inside-out tracking of the camera in relation to the environment. Making use of the multi-marker tracking capabilities of ARToolKit, we could overcome this limitation, using all visible markers in the camera image to average out tracking errors and provide stable tracking of the device with respect to the environment.

Since our application was intended for use by non-expert audiences, and because we also wanted to allow primary school children to participate, we invented a simple game called *"The Invisible Train"*. We specifically chose the game genre because we expected its playful nature of engaging in cooperative tasks would

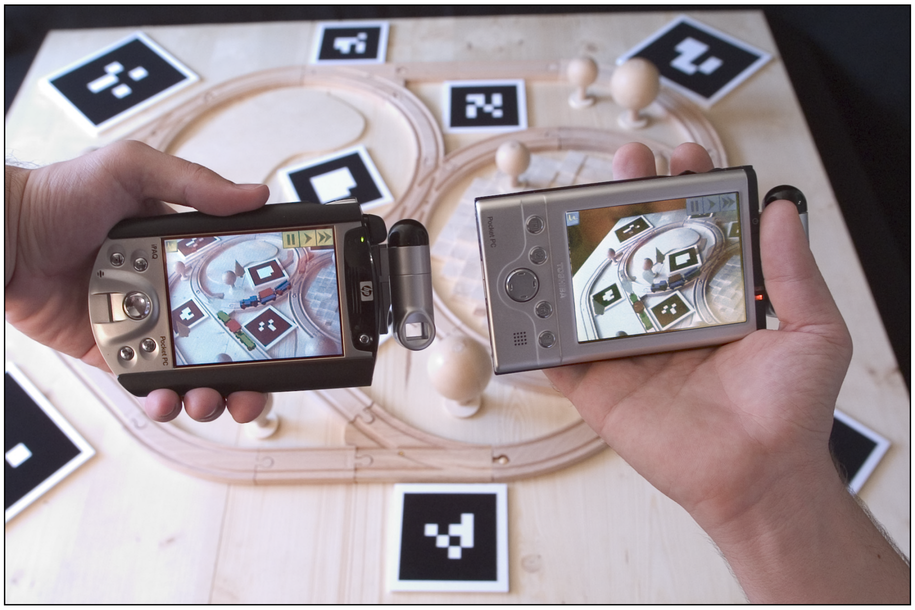

Fig. 3. Two PDAs running the *Invisible Train* game

encourage users to participate in our evaluation. Furthermore, we assumed a game would make the concept of AR interfaces more intuitively accessible to users without prior exposure to this type of technology. The *Invisible Train* lets players control virtual trains on a real wooden miniature railroad track. We deliberately left the decision whether the game should be collaborative or competitive open. As a result, the game can be played either collaboratively (trying to avoid a collision between trains for as long as possible) or competitively (attempting to crash into the other player's train). Since people were anticipated to use the application for just about a minute each, we omitted a scoring mechanism and left the decision whether to cooperate or compete to the players.

4 The Invisible Train

The *Invisible Train* is a simple multi-player game, in which players steer virtual trains over a real wooden miniature railroad track. These virtual trains are only visible to players through their PDA's video see-through display, since they do not exist in the physical world. Figure 4 shows the game's user interface elements, as seen from a player's perspective.

Users are offered two types of actions: operating track switches and adjusting the speed of their virtual trains, both of which are triggered in response to a tap on the PDA's touch screen. There are two different kinds of track junctions: three-way (Y-shaped) and four-way (X-shaped) interconnections. Both are visualized through semi-transparent graphical icons floating above the physical

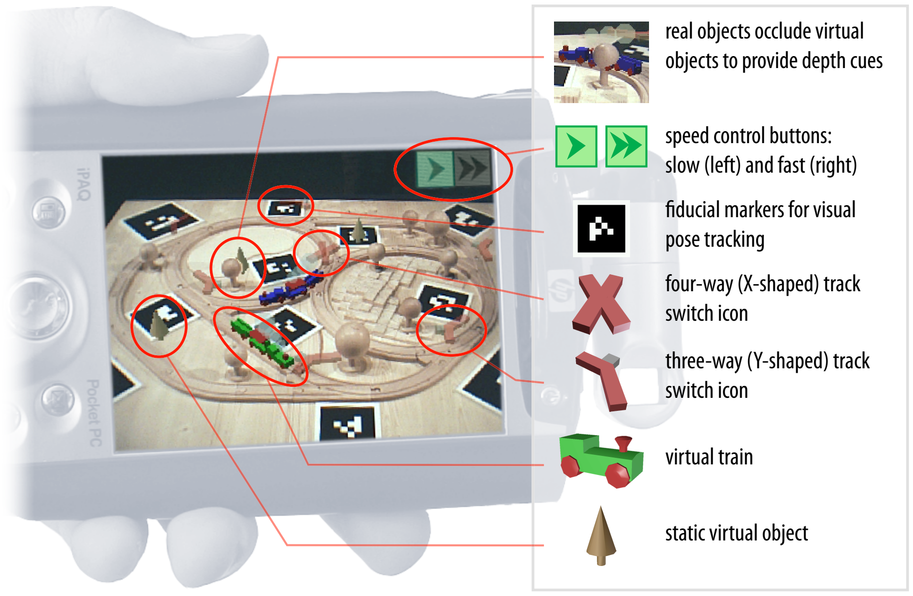
real objects occlude virtual objects to provide depth cues
speed control buttons: slow (left) and fast (right)
fiducial markers for visual pose tracking
four-way (X-shaped) track switch icon
three-way (Y-shaped) track switch icon
virtual train
static virtual object
iPAQ
Pocket PC

Fig. 4. User interface elements and graphical features

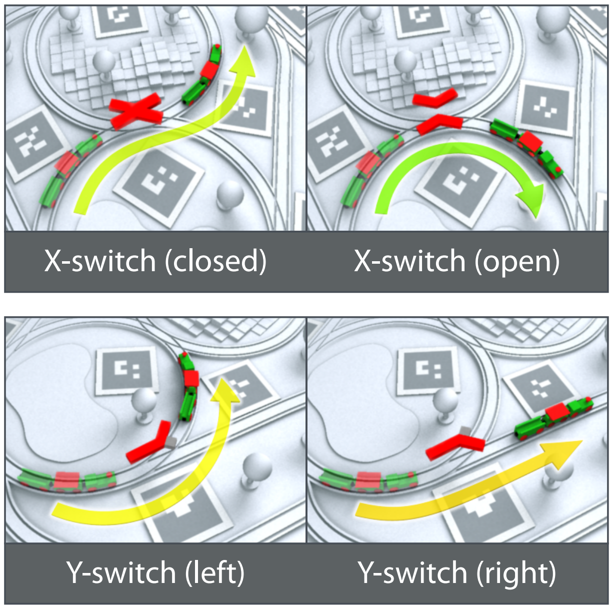

Fig. 5. Track switch icons and their effect on train routes

junction element. These track switch icons serve as clickable buttons and indicate their current state and effect on train routes by their visual appearance (see Figure 5).

Whenever users activate a track switch, its icon turns fully opaque for one second, during which other track switch buttons become unclickable. This mechanism was primarily intended to provide users with visual feedback, but will also prevent "racing conditions" where multiple users rapidly try to operate the same track switch. Users need not exercise great precision when aiming at their touch-screens: a ray-casting algorithm automatically selects the appropriate track switch depending on the closest virtual track being pointed at.

Virtual trains can ride at two different speeds, which can be controlled via two dedicated buttons in the upper right screen corner. The active button is shown in color while the inactive button is grayed out.

During the game, application state is constantly synchronized between all participants via wireless networking. Whenever a collision occurs, the game ends.

5 Evaluation Scenario

We consecutively deployed the *Invisible Train* at four different locations: the SIGGRAPH 2004 computer graphics convention in Los Angeles (USA), an orientation day for incoming freshmen at Vienna University of Technology, a career information event for secondary school students, and inside the Ars Electronica Center's (AEC) "Museum of the Future" in Linz (Austria).

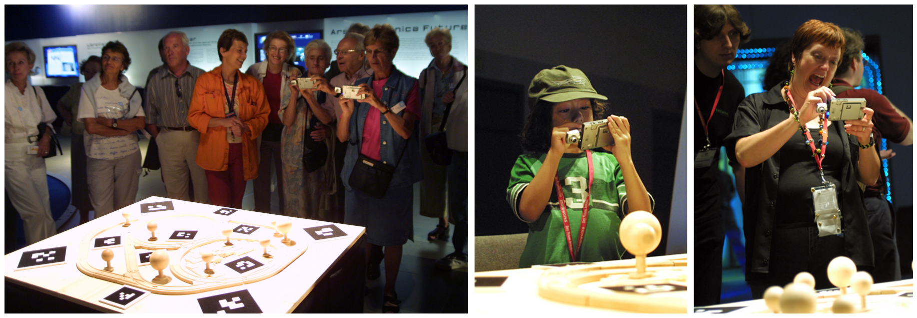

Fig. 6. Visitors playing the *Invisible Train*

Over the course of these exhibitions, we gradually moved from expert audiences, who were familiar with AR technology, to a general public (see figure 6) with little or no previous exposure to AR. An estimated five to six thousand visitors have engaged in playing the *Invisible Train* game during the four evaluation cycles, the last of which lasted for over four weeks and was partially unsupervised (with occasional maintenance work done by AEC museum staff). To our knowledge, these quantities lie at least an order of magnitude above comparable informal field tests of mobile AR system, denoting the first time a mobile AR application has successfully withstood a field-test of sizeable proportions.

6 Usability Assessment - Lessons Learned

Although we did not perform a formative user study, we solicited user feedback through informal, unstructured (i.e. no specific or predetermined sets of questions were asked) interviews and conducted a summative evaluation of user performance and behavior, which led to small iterative refinements of the game's user interface. More importantly, however, we successfully completed a rigorous stress-test of our system architecture's overall robustness.

Several of our empirical observations, some of which were directly comparable to our past experience involving HMD-equipped "backpack"-style setups, confirmed our generic assumption that handheld devices would generally be more accessible to a general public, and exhibit flatter learning curves, than traditional mobile AR systems: We found that visitors had little to no reservations towards using our system. Several participants figured out how to play the *Invisible Train* on their own by simple trial and error, others would learn the gameplay by looking over another player's shoulder while awaiting their turns — some children would intuitively grasp the concept and outperformed even seasoned computer science professionals. Consequently, our supervision overhead was considerably lower than administrators of traditional mobile AR application would normally experience. On many occasions, we could observe unsupervised user experience in which visitors would pass around the PDAs while explaining the game to each other. Most participants would play at least a single game (averaging roughly 60 seconds) before handing their PDA to the next visitor.

In stark contrast to our past experiences with "backpack" setups, we experienced almost no hardware-related failures, with the exception of a small number of application crashes, whenever users removed the add-on camera from its SDIO slot. These incidents have only further confirmed our observation that wearable devices intended for public deployment must resemble robust monolithic units without any loosely attached parts.

According to user feedback, our application was considered sufficiently responsive for the intended type of spatial interaction: only a negligibly small fraction of players felt their PDA's display update rate and delay would impair their ability to play the game. We measured our system's average performance at 7 frames per second (on devices equipped with Intel's XScale PXA263 processor clocked at 400MHz, and an add-on SDIO camera from Spectec Computer Ltd), while wireless network latency was measured at about 40-50ms. Camera blur caused loss of registration during rapid movements, but was not considered a major problem. Thanks to code optimizations and a new generation of PDAs that feature hardware-accelerated 3d-graphics, we were most recently able to raise our system's overall performance to approximately 12 frames per second.

Another observation we made during the four week evaluation period at the AEC is that some users will eventually try everything to forcibly break the software. Thus, we came to believe it was important for applications to automatically restart in case of a failure, e.g. after a user deliberately presses the hardware reset switch.

The major cause of interruptions arose from our devices' short battery life, which lasted approximately two hours (per 1800 mAh rechargeable Li-ion module), thus requiring regular battery replacements.

7 Conclusion and Future Work

Overall, our large-scale system evaluation yielded very satisfying results. Although we have yet to conduct formal usability testing, empirical evidence suggests that our handheld AR interface was sufficiently intuitive to grasp, enabling untrained users to complete simple collaborative spatial tasks with minimal or no prior instruction. Our system proved exceptionally stable and robust under heavy use conditions. It continued to function without interruption during the final two-week period of unsupervised use by museum visitors.

As a next step, we plan to move from a four-user setup to an application scenario that simultaneously involves dozens of participants in a wide-area setting, which will bring us one step closer to our goal of *"AR anytime, anywhere"*.

Acknowledgements

This research was funded in part by the Austrian Science Fund *FWF* under contract No. Y193.

References

1. Starner, T., Mann, S., Rhodes, B., Levine, J., Healey, J., Kirsch, D., Picard, R.W., Pentland, A.: Augmented reality through wearable computing. Presence: Teleoperators and Virtual Environments **6** (1997) 386 – 398
2. Feiner, S., MacIntyre, B., Höllerer, T., Webster, A.: A touring machine: Prototyping 3d mobile augmented reality systems for exploring the urban environment. In: Proceedings of the First International Symposium on Wearable Computers (ISWC), Cambridge, Massachusetts, USA (1997) 74–81
3. Höllerer, T., Feiner, S., Terauchi, T., Rashid, G., Hallaway, D.: Exploring mars: developing indoor and outdoor user interfaces to a mobile augmented reality system. Computers & Graphics **23** (1999) 779–785
4. Piekarski, W., Thomas, B.H.: Tinmith-evo5 a software architecture for supporting research into outdoor augmented reality environments. Technical report, Wearable Computer Laboratory, University of South Australia (2001)
5. Bier, E.A., Stone, M.C., Pier, K., Buxton, W., DeRose, T.D.: Toolglass and magic lenses: The see-through interface. In: SIGGRAPH '93: Proceedings of the 20st Annual Conference on Computer Graphics and Interactive Techniques, New York, NY, USA (1993) 73–80

6. Viega, J., Conway, M.J., Williams, G.H., Pausch, R.F.: 3d magic lenses. In: UIST '96: Proceedings of the 9th annual ACM symposium on User interface software and technology, Seattle, Washington, USA, ACM Press (1996) 51–58
7. Rekimoto, J., Nagao, K.: The world through the computer: Computer augmented interaction with real world environments. In: UIST '95: Proceedings of the 8th annual ACM symposium on User interface and software technology, Pittsburgh, Pennsylvania, USA (1995) 29–36
8. Newman, J., Ingram, D., Hopper, A.: Augmented reality in a wide area sentient environment. In: Proceedings of the 4th IEEE and ACM International Symposium on Augmented Reality (ISAR'01), New York, NY, USA (2001) 77–86
9. Gausemeier, J., Fründ, J., Matysczok, C., Bruederlin, B., Beier, D.: Development of a real time image based object recognition method for mobile ar-devices. In: AFRIGRAPH '03: Proceedings of the 2nd International Conference on Computer Graphics, Virtual Reality, Visualisation and Interaction in Africa, Cape Town, South Africa (2003) 133–139
10. Schmalstieg, D., Fuhrmann, A., Hesina, G., Szalavári, Z., Encarnação, L.M., Gervautz, M., Purgathofer, W.: The studierstube augmented reality project. Presence: Teleoperators and Virtual Environments **11** (2002) 33–54
11. Wagner, D., Schmalstieg, D.: First steps towards handheld augmented reality. In: Proceedings of the 7th International Symposium on Wearable Computers (ISWC 2003), White Plains, NY, USA, IEEE Computer Society (2003) 127–137
12. Reitmayr, G., Schmalstieg, D.: An open software architecture for virtual reality interaction. In: VRST '01: Proceedings of the ACM symposium on Virtual reality software and technology, Banff, Alberta, Canada, ACM Press (2001) 47–54
13. Ledermann, F., Schmalstieg, D.: APRIL: A high-level framework for creating augmented reality presentations. In: Proceedings of the 2005 IEEE Virtual Reality Conference (to appear), Bonn, Germany, IEEE Computer Society (2005)
14. Billinghurst, M., Kato, H., Poupyrev, I.: The magicbook: a transitional ar interface. Computers & Graphics **25** (2001) 745–753
15. Wagner, D., Barakonyi, I.: Augmented reality kanji learning. In: Proceedings of the 2003 IEEE and ACM International Symposium on Mixed and Augmented Reality (ISMAR 2003), Tokyo, Japan, IEEE Computer Society (2003) 335–336
16. Billinghurst, M., Cheok, A.D., Prince, S., Kato, H.: Real world teleconferencing. IEEE Computer Graphics and Applications **22** (2002) 11–13
17. Barakonyi, I., Fahmy, T., Schmalstieg, D., Kosina, K.: Collaborative work with volumetric data using augmented reality videoconferencing. In: Proceedings of the 2003 IEEE and ACM International Symposium on Mixed and Augmented Reality (ISMAR 2003), Tokyo, Japan, IEEE Computer Society (2003) 333–334