Meeting Technology Challenges of Pervasive Augmented Reality Games

Wolfgang Broll, Jan Ohlenburg, Irma Lindt, Iris Herbst, Anne-Kathrin Braun
Collaborative Virtual and Augmented Environments Department
Fraunhofer FIT
Sankt Augustin, Germany
{wolfgang.broll, jan.ohlenburg, irma.lindt, iris.herbst, anne-kathrin.braun}@fit.fraunhofer.de

ABSTRACT

Pervasive games provide a new type of game combining new technologies with the real environment of the players. While this already poses new challenges to the game developer, requirements are even higher for pervasive Augmented Reality games, where the real environment is additionally enhanced by virtual game items.

In this paper we will review the technological challenges to be met in order to realize pervasive AR games, show how they go beyond those of other pervasive games, and present how our AR framework copes with them. We will further show how these approaches are applied to three pervasive AR games and draw conclusions regarding the future requirements regarding the support of this type of games.

Categories and Subject Descriptors

K.8.0 [**Personal Computing**]: General – Games. H.5.1 [**Information Interfaces and Presentation**]: Multimedia Information Systems – *Artificial, augmented and virtual reality*. C.2.4 [**Computer-Communication Networks**]: Distributed Systems – *Distributed applications*. H.2.8 [**Database Management**]: Database Applications – *Spatial databases and GIS*. B.4.2 [**Input/Output and Data Communications**]: Input/Output Devices. C.2.1 [**Computer-Communication Networks**]: Network Architecture and Design – *Network communications, Wireless communication*. D.2.12 [**Software Engineering**]: Interoperability – *Data mapping, Distributed objects*. H.5.2 [**Information Interfaces and Presentation**]: User Interfaces – *Graphical user interfaces (GUI), Input devices and strategies.* H.5.3 [**Information Interfaces and Presentation**]: Group and Organization Interfaces: Collaborative computing, Computer-supported cooperative work, Synchronous interaction

Keywords

Ubiquitous computing, pervasive gaming, mixed reality, augmented reality.

Netgames'06, Oct.30-31, 2006, Singapore.

1. INTRODUCTION

Much more than traditional games, computer games usually provide more challenges, create larger curiosity, and require more players to enter a more fantastical world, all of which combine to form the major factors for creating fun games [18]. On the other hand traditional games provide easy interaction with other users and the environment, and typically do not constrain the users in what they can do.

Today we can observe a trend towards the next generation of computer games going beyond the desktop PC: Pervasive games. Pervasive games use information and communication technology to overcome the boundaries of conventional games, creating new enhanced gaming environments, making the real environment an intrinsic component of the overall game. Thus, bridging the gap between the virtual world of today's computer games and the real world - home of traditional board games and outdoor games, pervasive games integrate the social quality of traditional non-computer games into the game play. Therefore a key aspect is the seamless integration of such games into our everyday life. This development can especially be observed in location-aware games [17].

The topic of pervasive gaming has only recently become important to the European Commission, which is currently funding a large integrated project on pervasive gaming (www.pervasive-gaming.org). First approaches to commercialize pervasive games can be observed at www.itsalive.com, but are still highly constrained. It is also an open issue, as to how design guidelines may help to ensure robust pervasive games or how to design games that embrace existing technical shortcomings [3].

Pervasive Augmented Reality (AR) games are a special type of location-aware pervasive games, which use Augmented Reality to enhance the real world of the players by virtual objects. Most of these games are either outdoor games or combined indoor/outdoor games. According to Stapleton et al. [25] they provide the ultimate entertainment experience by combining virtual content, storytelling, physical imagination, and the imagination of the users. *Epidemic Menace* is the first prototype of a crossmedia game involving different types of media (including AR) that offers a comprehensive gaming experience to the players [14][15][21] (see also sec. 5.2). While some approaches have been made to create a pervasive game infrastructure for table-top games (e.g. [16]), neither an appropriate framework nor its fundamental components exist for pervasive AR games. Furthermore appropriate mechanisms to support rich, adaptable and flexible interaction techniques in crossmedia environments are missing and require further research [4].

While existing work mainly focuses on the presentation of individual solutions, design criteria, and the evaluation of prototypes, this paper tries to provide an overview of the technical requirements for pervasive AR games. Thus, in this paper we will briefly review technologies used by existing pervasive and AR games in section 2, before we introduce the major challenges and requirements identified in section 3 - including typical short comings and pitfalls. In section 4 we will show how we address these challenges by our environment based on our general purpose AR framework. In section 5 we will present three sample projects, before discussing the lessons learned in section 6.

2. RELATED WORK

Pure outdoor, pure indoor or combined indoor/outdoor games are used to explore technologies of ubiquitous computing in combination with mobile and wearable computing. A number of these games work with similar design features and technology. In the first part of this section, an overview of existing pervasive games is given, where the player has to find items using a GPS-receiver and the Internet. In the second part, examples of AR Games are presented.

2.1 Pervasive Games

Basic requirements of all kinds of location-aware pervasive games are suitable technologies for localizing the user and communication between players and the control staff. Many games use existing WiFi and/or GSM infrastructure for communication and GPS for location determination. Examples are *Uncle Roy All Around You* [1], *Capture the Flag* [26] and *Can you see me now?* [11].

In *Can you see me now?*, outdoor players chase online players through the streets. The outdoor player is equipped with a handheld PDA with WIFI, a GPS receiver and a walkie-talkie for communication with the other players. The talk is streamed to the online-players. The disconnectivity of WIFI and GPS and the fact that wireless networking is not always available seems to cause problems.

Uncle Roy All Around You is a successor to *Can you see me now?* developped by the same team. The players in the street now use a 3G mobile phone instead of a PDA. In this game, the real players have to find a mysterious character called Uncle Roy, somewhere in the streets. They are supported by online players.

A smart phone as main interface is also used in the game called *Capture the Flag*. The aim of the game is that a team has to catch the opponent's flag. A team consists at least of one knight and one guide. The knight is the real world player equipped with a Symbian-based smartphone and a GPS receiver and the guide, playing in the virtual word, is using a desktop PC with a wireless connection to the outdoor player. The flag is represented by a real small wooden box, with a Linux-based Bluetooth device inside. The Bluetooth device is build by a Bluetooth dongle, a single-board computer, a touch sensor and a power supply. The knights capture the flag by connecting to it through their smartphone, activating its touch sensor and physically picking it up. The core of the system is the Active Game Server, which stores all the game information and database. It communicates with all clients, which are at least, two smartphones and two PCs.

A game, making use of areas without wireless network in order to enhance experience is the *Bill Game* [6]. In the *Bill Game* the players have to collect virtual coins from outside the wireless network and bring them to an area of sufficiently high network signal strength and then upload them to a server. The game runs on a PDA connected via WiFi and uses GPS for position detection. Due to the frequently changing network connections and temporal disconnectivity, a UDP based messaging system is used.

An indoor game where the position is not determined by an absolute positioning system such as GPS, but uses a relative positioning system like a radio frequency (RF) based sensor is the game *Pirates!* [4][10]. In *Pirates!* each player is a captain of a ship, who have to solve a different mission. The handheld PDAs of the players are connected to a WiFi network. Each handheld device has a radio frequency based proximity sensor (RFID) to determine the location and the proximity to other players and local resources. The game events depend on the position of the players.

2.2 Augmented Reality Games

One of the most popular games using techniques of Augmented Realitiy is *EyeToy*® (www.eyetoy.com) for Sony Playstation 2. In this game the player can interact with his motion or sound captured by a USB-camera or a microphone.

An example of an indoor game is The *Invisible Train* [24], where the player uses a PDA to steer a virtual train over a little railroad track. The player may change the speed of the train and modify the track switches. The player interacts with the touch screen of the PDA. Moreover, it can be played as multi-user game, where the users are connected via WiFi. The game state is synchronized such that all players see the same virtual trains and virtual track switches. The position of the PDAs are determined via small cameras attached to the PDAs and marked-based computer vision software running on the PDAs. The corresponding makers are located all over the game area and have to be visible to the PDAs camera in order to allow for an appropriate augmentation.

Examples for AR outdoor games include *ARQuake* [22], Human *Pacman* [8] and *NetAttack* [13] (see sec. 5.1). These games are not PDA-based. The players are rather equipped with a wearable computer using various kinds of sensors. The wearable computer typically consists of a notebook in a backpack and a head mounted display (HMD).

In *ARQuake*, which is an AR version of the famous ego-shooter Quake, the player runs around shooting monsters and collecting items. In the augmented version, the player is situated in the real world, equipped with a differential GPS receiver and marker-based computer vision tracking for positioning and a digital compass for orientation. ARQuake is built upon the freely available source code of Quake. The real environment in which the game is played is modeled and imported as Quake environment.

Human Pacman is the AR variant of the famous Arcade game Pacman. In contrast with *ARQuake* it supports multiple players. The story of AR *Pacman* is the same as that of the original game, thus, Pacman (represented by one player) has to collect all the cookies in the world, while a ghost (represented by another player) has to catch *Pacman*. In this game, so called *Treasure*

Boxes (a special kind of cookie) are represented by real world objects. The game system consists of four entities: a central server, the wearable client system, Bluetooth devices (e.g. in the treasure box) and a helper laptop. The client and server communicate via WiFi, and the client with the physical devices via Bluetooth. The users head motion is tracked using a 3-DOF orientation sensor (Intersense InertiaCube2) built into their helmet. The position is determined via GPS. A special touch-sensor detects whether the player is touched by the enemy player. The helper player of ghost and *Pacman* plays remotely via WiFi. Thus online players can be included. The helper sees the world in VR mode, while the players representing the ghost and *Pacman* see it in AR mode.

3. TECHNOLOGICAL CHALLENGES AND REQUIREMENTS

In this section we have summarized the major technological challenges and the requirements for pervasive AR games. Where applicable these include a comparison with the requirements of standard (non AR) pervasive or other (non pervasive) 3D games.

3.1 Localization

The major challenge for almost all types of pervasive games is the position of the users. As game content and action typically depend on the current location of the individual player, pervasive games require and make use of tracking technologies. While for most types of pervasive games a rather rough localization (in the order of 10 meters) is sufficient [4][6], requirements are much higher for pervasive AR games. In order to facilitate a proper augmentation of the real environment, tracking accuracy should be in the meter or even centimeter range (depending on the distance to the closest augmented object). In addition to position tracking, AR games additionally require accurate orientation tracking as the viewing direction is even more important for the augmented view. Thus, additional tracking technologies have to be set-up.

Figure 1. Urban canyons – a major reason for GPS signal dropouts in urban environments

GPS is the tool-of-choice for most wide-area outdoor environment. The use of satellite-based localization services is also likely to increase when the European Galileo system will finally become available. However the quality of the data received via GPS is often poor due to the small number of available satellites. This problem also arises when the signal is blocked due to the environment such as in forests, city centers (also known as the urban canyoning effect, see figure 1) or mountainous areas.

There are a number of enhancements available to GPS such as D-GPS or WAAS/EGNOS, however all of these require additional setups or are limited to certain areas. While all these technologies improve the overall accuracy of signals they cannot overcome the problem of objects blocking the signal.

Computer vision represents another common tracking solution, which may be used to enhance GPS data or to provide user tracking in areas, where no GPS signal is available (e.g. in indoor environments or for the same reasons as mentioned before). We can distinguish between marker-based and markerless tracking. Marker-based tracking techniques (such as ARToolkit or ARTag) require appropriate markers within the field-of-view of the observer's camera. In this solution the user looks for markers which are attached to buildings, monuments or other features. However obtaining permission to attach markers to buildings is often problematic. Markerless tracking techniques recognize certain object features such as color, appearance, edges, or corners. However in contrast with marker-based approaches they are often slow and the geometry-based approaches do not work well when the user is undertaking fast movements. An alternative and very promising approach is the use of SIFT (scale-invariant feature transforms) for position tracking [24].

Alternatives to GPS include estimating the current location based on GSM cell information or using the signal strength of well known WiFi hotspots. The resolution of GSM cell information is very coarse – especially in rural environments. While this maybe acceptable for some pervasive games it definitely is not sufficient for pervasive AR games. A-GPS provides an enhancement of GPS localization by using GSM cell information. In inner cities where the GSM cells are typically very small, but GPS reception is rather bad, this combination may be used to improve localization. This service however, is currently supported by a rather small number of providers in certain countries and requires a particular cell phone. Localization based on WiFi hotspots may even use stations not accessible to compute localization information, but the signals are vulnerable to interferences. A hybrid approach making use of several radio beacons was developed by Place Lab (www.placelab.com). In this system the signals from WiFi, GSM, and stationary BlueTooth stations are cached in a local map which is then used to calculate the position.

Orientation tracking as required for AR applications typically makes use of 3-DOF orientation trackers, based on inertial sensors, gyoscopes, or magnetometers (or a combination of them). Examples of such trackers are the InterSense InertiaCubes or the Xsens MTx tracker.

Thus, typical solutions for pervasive AR games will be based on a hybrid approach, typically combining GPS information with 3-DOF orientation and enhanced by additional mechanisms such as computer vision if additional accuracy is required by the application.

3.2 Communication

As pervasive game applications represent distributed environments, communication between the individual participants and central components or services is essential. For example, any actions of individual players which alter the game state must be transmitted to the other players. In urban environments WiFi hotspots are one method of transferring such data, with cell phone based services such as GPRS or 3G (UMTS) providing alternative options. However, it should be noted that technologies such as 3G HSDPA allow the transfer of such data without a drop in bandwidth. Another alternative includes the soon to be available WiMax.

While technologies such as GPRS or public hotspots work on a global basis, i.e. transmitting information between the user and the Internet, much of the communication required in pervasive games is between players, or between players and game-related objects. Technologies which support local communication include WiFi, BlueTooth and RFID. However technologies which support local communication may not work when players are outside a certain range with respect to other players or objects.

Pervasive games also suffer from problems related to the communication signals failing or not being available in certain areas. Therefore it is very important that all game components can handle such disconnections and do not rely on continuous connections with all the participants. One way to cope with temporal disconnections is not to hide them from the user, but to exploit them in the game design. Using this approach means that players will not see those periods as a system malfunction, but rather as a special game situation, which may even be advantageous for a certain period. This approach has successfully been used in [1][6].

The wide range of end-user devices introduces another well-known problem: interoperability. This problem is even more challenging in pervasive games. While current solutions (e.g. CORBA) handle endianess and some platform and programming language specific issues, some devices even lack standard data types such as floating points (J2ME 1.0)[1]. Although conversions and emulations exist for floating points, these devices are often under powered and such support would put a heavy burden on them. Therefore, it is necessary to have a universal event and data format, which allows the sender and the receiver to quickly send and retrieve the desired information. This format should be flexible with respect to mandatory and optional fields, e.g. an event containing GPS tracking information should at the very least contain longitude, latitude, altitude and a time stamp. However it can also contain all the other information contained in the NMEA protocol. Therefore a smartphone should be able to send GPS information containing only the mandatory information, and be able to ignore any optional information it receives. A solution for the lack of floating points is to send them in a fixed point notation, i.e. 32 bit integer for the mantissa and 32 bit for the exponent, allowing a J2ME 1.0 client to ignore the exponent.

The same is true for the underlying network protocol, restricting it to a specific one, e.g. CORBA, TCP/IP or UDP, makes it difficult for some clients to comply. The Mobile Information Device Profile (MIDP) 1.0, which is part of J2ME 1.0 only supports HTTP networking. Hence any event distribution service should allow the clients to send and receive data using the network protocol which they support. Such a service would operate on a central server and apply filters and combiners as required. It could also provide additional functionality such as event loggging and the ability to extrapolate information.

Like other multiplayer games, pervasive games are also affected by participants joining and leaving during play. The only difference is that players in pervasive games may leave the game without sending any notification - this could be due to lack of a network connection. Such spontaneous disconnections are much more common in pervasive games than in traditional games and a game server must handle this issue. In contrast no additional efforts have to be taken in respect to other multiplayer games, except for situations where ad hoc connections between players are setup to exchange game items.

In pervasive games it is essential that the communications services provide consistent and reliable information. Such a service must have two levels of consistency control. The first has ensure that all clients receive game status updates on a regular basis in order to synchronize their local replicated game state - this is the case even if they temporary do not receive updates due to missing network availability. Synchronization can be achieved by either sending complete game states or allowing clients to query all changes since a specific time period. Limited bandwidth tempts game developers to send the minimal amount of data in order to update the game state on remote clients, e.g. by sending only changed information or single events such as *remove item*. Such mechanisms have to be designed very carefully, since clients may miss those updates. Thus additional synchronization mechanisms have to accompany relative updates.

As participants may continuously join and leave the game a persistence service is also essential for all pervasive games. The complexity of such a service depends on the game and whether it can easily be handled within the game engine or if something more advanced is required. In cases where players are able to continue playing while they are disconnected, changes to the game state have to be transmitted as soon as the player is connected again. This of course raises consistency issues, e.g. if actions can be taken while the player is offline (collecting an item), the game engine cannot interfere with this action, because another player has taken a concurrent action (collecting the item first). In this case the game state is radically altered as the offline players now have an advantage over the other participants. Furthermore, if more of the game logic is handled by the clients this raises the prospect of players being able to cheat more easily. However discussions related to cheating are beyond the scope of this paper.

Including AR into a pervasive game does not necessarily introduce new issues for the communication, but this depends of course on the data that needs to be distributed. In case the whole AR environment already recites on the client, neither the communication load nor the type of data which is transferred differs from other clients, e.g. position updates of objects, remove, and insert events. Since handheld devices might not be able to store the whole environment in their memory, other strategies have to be applied, e.g. the aura-nimbus approach. A

[1] Although J2ME 1.1 introduced floating points, phones that support only J2ME 1.0 will be around over a couple of years.

good overview of scalability management in distributed VR environments is given in [23].

3.3 Crossmedia Augmentation

While certain AR games use a particular system setup for their players, crossmedia AR games provide a variety of different augmentation solutions for their participants. Regarding the AR equipment of the individual players we can distinguish between head-mounted AR systems, handheld AR systems and augmented video streams.

Head-mounted AR systems use a head-mounted or head-worn display (HMD/HWD) for providing an augmented view of the environment to the user. In general we can distinguish between monocular and binocular displays. While binocular displays allow superimposing the real world by individual virtual objects for each eye separately, creating a stereoscopic perception of the AR content, covering both eyes implies some safety issues, which may not be acceptable in public areas. We can further distinguish between optical and video see-through technologies. While the first uses semitransparent displays for superimposing the real environment by virtual objects, the second alternative superimposes the video image captured by a head-mounted camera. In the latter case the user does not directly see the real environment. This may also imply safety issues in case of display malfunctions. For these reasons outdoor AR in public areas should use optical see-through displays. The virtual image in the display will be supplied either by laptop computer mounted at a backpack (see Figure 2), by a wearable computer or Micro-PC (a full laptop computer about twice the size of a PDA), or by a PDA providing an external display adapter.

Figure 2. Head-mounted AR system

Handheld AR systems use a device providing a display which shows the video image from a camera mounted or integrated to the opposite side of the device and augmented by the virtual objects. This technology can be applied to tablet PCs, UMPCs (ultra mobile PCs), Micro-PCs, PDAs, smart phones, and even game consoles such as the PSP. The disadvantage of handheld AR devices is that the viewpoint of the augmented image always differs from the viewpoint of the observer. Thus the image is never correct and may result in a complete false impression of the augmentation if the observer looks from a viewpoint not perpendicular or too far away from the display. On the other hand it is much easier to use for occasional game participants and provides the advantage that the devices can be easily shared between several players.

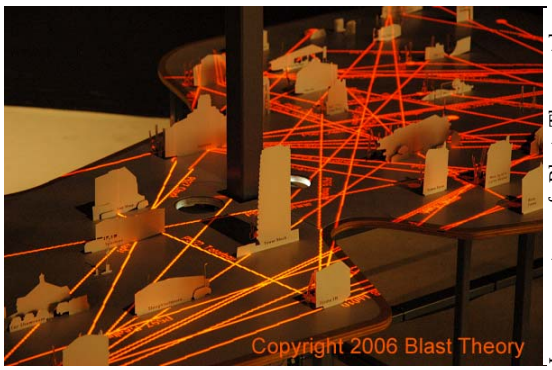

Image courtesy of Blast Theory taken from Day Of The Figurines test at the Sonar Festival, Barcelona 2006

Figure 3. Augmented game board using projection-based AR

Finally augmented video streams allow providing information about virtual objects in real environments over a distance and to observers. Thus, the video streams of particular or public web cams can be augmented by virtual objects within the field-of-view of the camera and transferred to individual players, team headquarters, public viewing areas, or even the Web.

In addition to these AR techniques, projection-based AR is often used for projecting the game content such as items or player representations onto a real map or a 3D environment model (see Figure 3). This may be used e.g. for orchestration purposes (see sec. 3.6) or as additional aid for the players.

3.4 Devices

Pervasive games and especially crossmedia games typically include a large variety of individual devices. We can distinguish between

- Personal I/O devices
- Public I/O devices, and
- Environmental input devices

Personal input/output devices are those which are used by a single player during the game or for a certain period of the game, and which are mounted to the player or otherwise carried by the player. Typical output devices include head-mounted or head-worn displays, PDA's, smartphones, or game consoles, etc. for video output, and head-sets, etc. for audio output including spatialized sound and speech output. Less common devices include tactile devices such as vibrating elements integrated into the players' clothes. Input devices include all kinds of tracking devices (see sec. 3.1), speech input, and the players' action devices. The latter may be rather traditional (often wireless) mice or keyboards, or PDA's, or special customized game input

devices. As not all players may have identical sets of devices (especially in crossmedia games) the individual devices should be easy replaced without needing to adapt of the game application. As a result appropriate device abstraction mechanisms are required.

Public I/O devices are game devices accessible by groups of players, all players of the game, or even spectators of the game. In most games they are restricted to output devices such as large-scale displays or projections, or web live streams. Video output is often combined with (spatial) audio and/or speech output. Consequently input is often related to those output devices and includes the use of touch screens as well as traditional mouse and keyboard input especially in team headquarters, etc. As these devices are of a rather permanent character, device abstraction usually is not an issue here.

Beside this we often find environmental input devices. There are devices which measure temperature, wind direction and velocity. This information is used by the game engine influencing the game progress, but players typically do not interact directly with these devices. The same is true for web cams that transmit game locations either to the players, the team headquarters, or to public viewing spaces. In AR games those video streams will typically be augmented by additional information including virtual game items.

3.5 Authoring and Game Engine

Regarding the realization of pervasive AR games we can distinguish several phases, during the development and preparation of a game or game event. After the overall game design has been finished, usually the hardware to be used is assembled (see sections 3.3 and 3.4), then the game items are modeled and the game actions are realized as part of the game engine (some actions may run independently on the player's client), and finally for playing the game in a particular location using a particular set of devices, it is set-up and configured – i.e. orchestrated (see sec. 3.6).

Game engines used for pervasive AR games do not differ totally from existing game engines for established computer games [12]. Thus we can find the typical action triggers including proximity, collision, visibility, time-dependent actions, etc. The reason is that while the overall game is a mixed reality application combining the real and the virtual, the game engine actually does not need to be aware of this fact. For the game engine everything might just be virtual. However, in contrast to other computer games, the input regarding players' locations refers to real movements. Thus all game items including the real ones require a virtual representation in the game engine, which has to represent the state of the real item. The same is true for environmental input devices. The game engine of course must be programmed or configured to react to changes appropriately, but it does not have to care whether the input is from a real sensor or whether this is just a simulator. Actually, in order to allow easy and independent testing, it should not distinguish between real and virtual items and I/O.

3.6 Orchestration and Surveillance

Game orchestration is a major issue in pervasive games and even more in pervasive AR games. While the overall game design and the game actions should be rather independent of a particular location, the individual game session always has to have a strong relation to the current real environment – otherwise the embedding into the real environment may appear arbitrary to the players, dramatically decreasing the quality of the gaming experience.

The necessary steps for pre-game orchestration include

- the appropriate registration of the game area into real world coordinates,
- the virtual representations of real world objects (items, buildings, etc.)
- the initialization and positioning of the game items,
- the initialization of the players including the setup of the individual equipment and the assembling of teams
- the initialization of the game state

Pervasive games usually require one or several moderators to survey the game progress and to influence the game in case of undesirable developments. to the moderators have to be able to:

- add, remove, or re-locate game items, or modify their state
- add or remove players or player equipment
- modify individual aspects of the game state

Supporting game orchestration and surveillance requires an appropriate support tool. Such orchestration tools will typically allow importing maps or images of the local gaming environment and mapping them to real world coordinates, thus providing a map of the overall game area. They should also allow the import and positioning of representations (2D or 3D) of real world objects within the game map. They provide access to the individual teams and players, allowing for modification of their state and settings, or adding and removing players. Usually each player will be represented by a photo, his (nick) name, contact information (depending on the communication facilities available within the game), and an icon also used for representing the player on the game map. Further game items will be accessible and may be (re-) located on the map. If computer vision based tracking is applied, appropriate markers or feature sets can be associated with particular locations or items. The tool further has to provide information about connectivity, i.e. input from players and environmental devices. The maps can also be used to visualize WiFi coverage and quality, as well as GPS availability. More advanced tools might allow not to restrict surveillance to a 2D map, but providing a full 3D interface where a 3D game environment is visualized using stereoscopic VR or AR technologies (e.g. in 3D projection environment or a roundtable scenario). This may also allow for setting the viewpoint of the surveying person to that of a real player or to follow a real player in a virtual representation of his real environment.

The technologies used for orchestration and surveillance may also be used for other game purposes. For instance a map augmentation may also be provided to individual players for navigation purposes or be displayed in team headquarters. Participants could also have access to individual player information about their team members showing their particular

communication and player state information. These tools can also be used for post-game visualization, e.g. for evaluation purposes.

4. FRAMEWORK AND TOOLS SUPPORT

For the realization of our pervasive AR games we have used the MORGAN AR/VR framework [20]. MORGAN is a platform-independent, component based framework that uses CORBA middleware for most of its communication, due to the simple access to remote components. It makes heavily use of popular design patterns, such as the *publish-subscribe* pattern and the *factory* pattern. Several components can directly be used by most pervasive AR games.

The *publish-subscribe* pattern is provided by components distributing data. Processes which are interested in this data are able to subscribe at the component and the new data will be sent to the subscriber. This reduces the network load, since components do not have to query the current state, if it has not changed. A special functionality allows each subscriber to receive updates at a maximum frequency, therefore less powerful devices, like a PDA or a smartphone will not be overwhelmed by the communication load. On the other hand powerful laptops do not suffer from higher latencies. Due to disconnectivity issues of pervasive AR games, CORBA and especially the *publish-subscribe* pattern is not appropriate for all communication purposes, since it relies on a continuous network connection. Other communication protocols such as HTTP, UDP, TCP/IP and Multicast are used where applicable.

Mobile Players have to report their current location regularly in order to take actions during the game. Since disconnectivity cannot be avoided, a *publish-subscribe* pattern is not appropriate for all situations - especially, if the clients do not have a fixed IP address. Although a *publish-subscribe* mechanism could be developed that handles temporarily disconnections and changing IP addresses, this would be to complex. Additionally CORBA is much too heavy-weight for most handheld devices, depending on the other functionality that has to run on that device. Therefore we have defined a universal positioning format, which is flexible enough to hold arbitrary location information, e.g. GPS, 6-DOF pose, GSM cell id (compare sec. 3.2). The requirements and the complexity on handheld devices accessing or providing the information is kept very small, since the sender may choose which information to send – e.g. just longitude and latitude and not speed, altitude, number of satellites and so on – also the receiver can easily ignore packages that it does not support (e.g. GSM cell id). This information can be sent over any communication protocol available, e.g. UDP.

As mentioned above Pervasive AR games use different tracking technology based on availability and accuracy. In order to keep the complexity for application developers small, we classify all input and output devices in a device hierarchy, allowing abstracting from a specific device and focusing on the provided functionality and data. The whole class of 6DOF tracking devices shares the same common interface enabling application to receive data from different trackers without recompiling the application. Adapters are special components which convert the input of one device class into another, e.g. converting GPS information into a Cartesian coordinate system, therefore providing GPS location through the 6DOF tracker interface.

Another important component is the distributed render scene graph (RSG) of the MORGAN framework. The RSG keeps the objects of a scene graph in a tree like structure and renders the contents into a frame buffer like OpenGL or Direct3D. For efficiency reasons, it is limited to only store objects which are necessary for the render process, i.e. graphical primitives, triangle mesh, lights, groups etc. Other scene graph information is usually file format specific and is kept in so called external scene graphs (XSG) [19]. Each supported file format, e.g. VRML 97, implements its own XSG which stores such meta data and defines the mapping onto the RSG. Since the RSG uses an abstraction layer for the existing frame buffers, namely OpenGL, Direct3D, OpenGL ES and Direct3DMobile, it can be used on a wide variety of devices including PDA's. The RSG has special functionality to support AR solutions, such as phantom objects, and supports different AR rendering modes for AR see-through and video-augmented AR. The later uses the input of a video camera or a web-cam and superimposes the contents of the scene graph onto it.

The contents of the frame buffer may be published in real-time as a video stream over the internet by another component, allowing a broad audience to watch these augmented video streams from all over the world. This functionality was used in the crossmedia game Epidemic Menace (see 5.2), where spectator could observe the augmented game area over stationary web cams.

5. SAMPLE PROJECTS

In this section we will present three sample projects in the area of pervasive AR games we have been working on so far.

5.1 NetAttack

One of the first games we developed was *NetAttack*® – a combined indoor and outdoor "scavenger hunt"-like game [13]. *NetAttack* is based around the story of an evil corporation trying to dominate the world by a huge database containing data of every human being. In the game, players are tasked to destroy the central database of the corporation. In order to achieve this goal, players collect artifacts allowing them to compose and decode a secret password, which eventually enables them to access and destroy the database.

Figure 4. A NetAttack outdoor player hiding behind a tree before snatching the "brain tank"

NetAttack is played by two competing teams, each team consisting of an indoor and an outdoor player. The indoor player (see Figure 5) sits in front of his or her desktop computer and supports the outdoor player via an audio link with valuable information: where to find hidden game artifacts, how to defeat the competing team and what to do next to win the game. The outdoor player (see Figure 4), equipped with a stereoscopic head-mounted display, laptop, position and orientation sensors walks around on a defined game area trying to collect game artifacts and to hide from the other competing outdoor player.

We implemented the localization of the outdoor players using a GPS receiver and an inertial tracker. While GPS-based position accuracy was sufficient for collecting game artifacts, it was not sufficient for measuring the distance between the competing outdoor players (an outdoor player could disturb the competing outdoor player if he or she was close enough). Therefore, we refined the position tracking using computer vision (ARToolkit) and markers carried by the outdoor players.

NetAttack was developed as a distributed MORGAN-based application. A central component implements the game logic and guarantees consistency. The communication between the different application components is based on WiFi. We tried to achieve a continuous WiFi coverage on the game area, such that we did not need to care about disconnections. This worked only to a certain extent. From time to time disconnections occurred, outdoor players could not continue to play the game and the complete game play was typically disrupted for several minutes.

Figure 5. A NetAttack indoor player

The implementation of *NetAttack* did not include an orchestration interface, but instead the possibility to configure many parts of the game before a game session. The configuration includes the game area, number and position of the game artifacts and the total gaming time. We found this to be sufficient since *NetAttack* is a rather short-time game played for one to two hours and no modifications are required during game play.

5.2 Epidemic Menace

Epidemic Menace is a crossmedia game developed within the IPerG project (http://www.epidemic-menace.de). It focuses on a game play across a large number of different devices and different media channels, as well as trans-reality player interaction. The devices employed in this game include smartphones, PDA's, mobile AR systems and stationary PC's, the players are able to communicate through the mobile phone and a chat application. Different media channels like a live augmented video stream of the game area and pre-produced video messages were also part of the game allowing a truly crossmedia and trans-reality game experience.

The story line of the game is about a deadly virus, which has been released on the campus of castle Birlinghoven by a villain scientist. Two competing teams of four players each are sent to the campus as squad teams to challenge the virus and find out who is behind this threat. After they have arrived both teams are lead into their headquarters and they are given an introduction movie showing an interrogation of one of the scientists. The headquarters are equipped with several computers to communicate with team members and monitor different aspects of the game. One large touch screen is used as an overview of the whole game area (see Fig. 6), the map shows the real-time location of each of the outdoor players and all viruses. The screen also displays the scores of both teams, the health status of each player and the current weather conditions (temperature, wind direction and humidity).

Two players of Epidemic Menace hunting a virtual virus The first viruses appearing on the map trigger an alarm in both headquarters, shortly afterwards an actor storms into the rooms and briefs the players and leads two of them to the outdoor game area. Here they are equipped with their first hunting devices, a smartphone showing a small portion of the map and the surrounding viruses. This device can be used to capture nearby viruses, the map is update according to the current location of the player. Since the phone only shows a small portion of the map, the outdoor players have to communicate with the indoor players who have to lead them to other viruses.

Image courtesy of Sony Netservices taken from the Epidemic Menace gaming event, Sankt Augustin, 2006.

Figure 6. Two players of Epidemic Menace hunting a virtual virus

At some point during the game all players are sent back to their headquarters since new video messages have arrived, giving new clues what happened. Depending on the actual score of the teams, the outdoor players are then equipped with a mobile AR system, which superimposes the viruses in the real world (see Fig. 5). This device can be used to destroy a nearby virus and is much more efficient than the smartphone. After a certain score has been reached one of the screens in the headquarters, provides a list of

decisions about what action to take next. They have to decide whom to arrest or to destroy the laboratory. The decisions have to be taken very carefully, because a team may loose instantly taking a wrong decision. After the correct decision has been taken by any team, both teams have to hurry up and clean the campus of the remaining viruses.

The game has been built using the MORGAN AR/VR framework and the described components. A lot of elements described in the previous sections play an important part of the game.

In order to enable the communication of the outdoor players with the game engine, a number of WiFi routers were setup on the campus to ensure a sufficient coverage of the game area. Contrary to the PDA's and the mobile AR systems, which utilize this infrastructure, the smartphones communicate using GPRS and 3G.

The location of the outdoor players is tracked by a wireless Bluetooth GPS tracker; the current GPS position of each player is transmitted to the game engine using the positioning format described in sec. 4 over a UDP connection. Players using the smartphone are also equipped with a PDA and a GPS tracker in order to report the current location. The phone receives its current location afterwards from the game engine which knows which player is equipped with this phone. The mobile AR system is also located using these GPS trackers, but also with an inertial tracking system for the head orientation. Actions taken by the players by either device are sent to the game engine and the result of the action is transmitted back. Since the result of the actions is location dependent, the players get an audio feedback of the positioning tool, whether the current location could be sent. In case the player moves into an area with no WiFi coverage, a specific sound is played notifying her to leave the area.

Figure 7. Epidemic Menace team headquarter

The stationary game board in the headquarters displays all game related information, and it is also used for game orchestration (see. sec. 3.6) by the game administrators. Of course the game administrators have in addition to the monitoring functionality the ability to influence the game dynamics by adding and removing viruses, as well as player management, i.e. adding/modifying/ removing players and assigning devices to them.

The weather information is provided by a MORGAN component that connects to a weather station on the campus, updating the weather conditions in real-time. This was very important for the game, since the viruses move according to the wind direction and strength.

The mobile AR system uses the MORGAN render engine and the MORGAN AR/VR Viewer to display the viruses in the head-mounted display and to update the viewpoint to match the location and orientation of the AR player. The device abstractions described in sec. 4 is used to abstract from the tracking system as well as the input device, which is used to destroy the viruses. Allowing to switch the localization and the input devices with no extra work. The Morgan AR/VR Viewer is also used for the live web streams for the spectators. The video feed from web cams on the campus are augmented by the viruses in the game area and streamed live over the web. Therefore spectators are able to watch the ongoing game through the epidemic menace website. The website also informs about the current score of both teams and the time each team has left.

In order to evaluate the game experience two sessions were performed on the campus of castle Birlinghoven. The first prototype was tested in August 2005 with 8 players, mainly undergraduate students, playing for two days. The final prototype one year later in July 2006 was played with 32 players aging from 18 to 60, four sessions with 8 players each playing for 3 hours. The evaluations indicate that there is significant potential for this type gaming genre. While the evaluation of the second event has not been finished yet, the report on the evaluation of the first event has already been published by Lindt et. al. [14].

5.3 TimeWarp

TimeWarp is a showcase application of the IPCity project (ipcity.eu) still under development. It is an edutainment adventure game played in a city environment to learn about the history in a playful way. This game contains a lot of requirements in common with other pervasive AR games. The first test runs are planned for the city of Cologne early next year.

TimeWarp is based on a saga of brownies, so-called Heinzelmen, who did all sorts of work in the city of Cologne. They baked bread, washed, and did any sort of domestic labor at night. This went on till a tailor's wife overcome with curiosity, decided to try and take a look at them. She threw peas up and down the stairs, so that they might fall and hurt themselves, and so that she might get a look at them the next morning. But she failed and since then no one has seen a Heinzelman. Of course the Cologne craftsmen are curious as to where the Heinzelmen have gone and have invented some new technologies to track them down. They found that the Heinzelmen have fallen into worm holes and now offer their services across time periods. However although the new technology allows for time traveling, it remains quite dangerous. Thus the Cologne craftsmen are looking for adventures that will bring the Heinzelmen back without risk to themselves or others and are offering a generous award. The game play allows for multiple players which might compete or collaborate.

TimeWarp is an outdoor AR game staged in the city center of Cologne. Players are equipped with a mobile AR system and a PDA. At certain places – 'challenge locations' – the real environment is enriched with virtual characters or virtual reconstructions of historical buildings using the AR system (see Figure 8). The PDA serves as an 'information terminal' showing the current position of a player and location of 'time portals' on a map. A public web page shows augmented video streams of the challenge locations and allows spectators to observe the game

event. The implementation is based on top of our MORGAN AR/VR framework and the provided tools.

Image courtesy of Sony Netservices

Figure 8. Virtual Heinzelman in the city of Cologne

TimeWarp requires localization using different qualities or features (see sec. 3.1); for example when there are no augmentations a rough tracking system is sufficient to track player movements, whereas higher accuracy is required if virtual buildings are overlaid with real ones or if there are virtual characters are in the real environment. The location of the player is tracked with different approaches: a GPS receiver is continuously used, but at the challenge locations a more precise computer-vision based tracking system takes over. The device abstraction (see sec. 4) of our framework supports this approach.

Being in a public space, it is not possible to set up our own WiFi. But public WiFi is unreliable and only available near the hot spots. Even GSM or 3G is sometimes not available. Facing this technical problems of the communication infrastructure (see sec. 3.2), a continuous game play seems only to be possible, if the game engine runs locally at the mobile AR system and only synchronizes from time to time with a game server for on-line orchestration and synchronization in case of multiple players. The game state of co-located players is synchronized via a Bluetooth connection.

The augmentation requires registered 3D visualization, so we decided to use a head-mounted display. But as play takes place in a quite large area we have to provide an additional orientation aid. Thus the player carries a handheld display showing an augmented map of the game area. Augmented information includes their path and current location, nearby time portals, location of team members, and locations of any game items they have found. Furthermore, an augmented video stream is published via the internet for public access. The different types of augmentation (see sec. 3.3) uses the MORGAN render engine and the MORGAN AR/VR Viewer (see sec. 4) to display the virtual game elements on different display.

A non-functional requirement of TimeWarp is the adaptability to different cities. An orchestration interface to adapt to different game environments, e.g. from Cologne to Berlin, and to set up different challenges appropriate for the chosen city, is mandatory. Furthermore game orchestration includes localization of time portals and other game items as well as the set-up of player and team profiles. It must also allow on-line game orchestration, e.g. the dropping of fortune elements and support for moderator intervention.

6. DISCUSSION AND LESSONS LEARNED

In this section we will discuss the use of the individual technologies in our sample pervasive AR games and present the lessons learned so far. An overview of the technologies in regard to the sample game applications can be found in Table 1.

	NetAttack	Epidemic Menace	TimeWarp
Localization			
GPS	YES	YES	YES
CV	YES	NO	YES
Communication			
WiFi	YES	YES	(NO)
GPRS/HSDPA (3G)	NO	YES	YES
CORBA	YES	YES	(YES)
TCP/HTTP	NO	(YES)	(YES)
UDP/Multicast	NO	YES	YES
Speech (GSM/3G)	YES	YES	YES
Speech (VoIP)	(YES)	NO	YES
Crossmedia augmentation			
HMD/HWD	YES	YES	YES
Handheld (TabletPC/PDA)	NO	(YES)	YES
Mobile	NO	(YES)	YES
Devices			
Mouse-like	YES	YES	YES
others	NO	YES	YES
Game engine			
Central game engine	YES	YES	YES
(semi-) autonomous clients	YES	NO	YES
Handling disconnectivity	NO	(YES)	YES
Orchestration			
Pre-game orchestration	NO	YES	YES
On-game surveillance and	NO	YES	YES
Team overview	YES	YES	(NO)

Table 1. Technologies used in sample game applications

All prototypes use GPS as the primary means of localization, CV was used where precision was required, other technologies are not applied due to the additional effort implied. Communication is done via WiFi and/or GSM/3G. All games use CORBA to some extent, but the later developments applied other connectionless services to better handle disconnectivity. Speech communication is used in all prototypes, but primarily done via traditional telephony rather than VoIP (although this will be the standard in future versions). All games use augmentation based on head-

mounted or head-word displays, but other types of augmentation recently evolved and will become more important in the future. All prototypes used mouse-like input devices, Epidemic Menace also used mobile phones and PDA's as input devices and applied special input devices for gaining local weather information. All games have central game engines, but more recent development support limited autonomy of the clients to better deal with disconnections. Game orchestration is missing in *NetAttack*, but is shown to be a central element for the organization and performance of in other gaming contexts.

Based on the pervasive AR games built so far, we can state that the use of a universal format for transferring positional date proved to be very useful. It provided an additional abstraction layer, making it easier to make different devices work together. However it also pointed to the need to avoid complexity be reducing the number and range of devices, in particular with respect to the number of devices being used by one player. Depending on the local game environment alternative tracking solutions based on radio beacons should be examined. However, where precise tracking of items, building details or other players is required, computer vision-based approaches have to be applied as GPS is not sufficient. Additionally, self-reporting of the location may be considered, but has to include features to prevent cheating (i.e. only enabled, when GPS fails, limited to certain area and a certain distance to the last known position). Another important issue to ensure that temporary disconnection does not cause a problem within the game environment. Thus any system should make sure that the local game application (on the player's client device) allows participation in the game, even when data is lost. Further use appropriate (connectionless) networking protocols (Multicast / UDP rather then TCP or CORBA), which do not require setting up the connection again after each disconnection. It is also very important to check the signal quality (GPS, GSM/3G, WiFi) in the desired gaming area before the actual gaming event. If possible this should already be done during the game design phase. However, if the final game area is not known or the game is supposed to be played in various locations, typical areas, which are representatives for the final environment, should be used for measuring and testing. One should also consider the impact weather will have on game play as not all equipment may be rain-proof, other equipment would at least have to be covered, which may reduce connectivity. Moreover thunderstorm clouds can influence signal quality, etc.. Regarding the game engine it is important to avoid sending very large (bulked) status messages, and using appropriate distribution mechanisms to efficiently distribute data to a rather large number of recipients (e.g. using Multicasting or P2P networks – the latter again introducing the risk of cheating). The game engine should be designed for robustness – nevertheless it may become necessary to restart the game – thus it should (in any game situation) be easy to recover the current state. Finally, orchestration proved to be essential for pervasive AR games. Not only that it allows to set-up the game properly, but it is the primary interface for the game moderators (we think that pervasive AR games so far have to be moderated) to influence the game and by that keep the game alive. This may also include the use of cheating by the moderator.

In order to meet the technology changes identified and by that simplifying the development and maintenance of pervasive AR games, the development of a set of rather high-level tools seems to be the logical next step. One of those obviously is the game engine (see sec. 3.5), enhanced by special means to use the technologies identified and to communicate with other pervasive AR tools. Another essential tool is the orchestration tool (see sec. 3.6). While orchestration has been supported in our existing prototypes to some extend, a game independent orchestration tool would allow for easy adaptation to any gaming location and configuration to support individual game items, teams, etc. The different localization techniques (see sec. 3.1) and their limitations require a ubiquitous tracking tool that not only abstracts from a specific technique, but also seamlessly chooses the best (or a combination) available. A tool for game authoring eases the creation of content and the game logic, e.g. by defining triggers, actions and game state transitions, in addition it is essential for adapting a game for new circumstances, such as new game actions or a different story line. It has to interface closely with the game engine. Finally, a logging tool supports an in-depth evaluation of the game play, allowing for playback a gaming session in combination with the orchestration tool.

7. CONCLUSION AND FUTURE WORK

In this paper we introduced the typical technological challenges the developers of pervasive AR games are facing today. The paper focused on the topics localization, communication, crossmedia augmentation, devices, authoring and game engine, and orchestration and surveillance. We further introduced the individual mechanisms provided by our AR framework MORGAN to deal with these challenges. We have discussed three pervasive AR games in detail, and have identified the related critical issues, resulting in an initial set of recommendations for the developer, but also providing the basis for the development of sophisticated pervasive gaming tools.

We expect further feedback regarding the use of the individual technologies for pervasive AR games after the initial test runs of the TimeWarp prototype. We are currently in process of unifying the set of game specific tools discussed. In our future work we plan to use and extend these tools to simplify and improve the overall quality of the development and performing of pervasive AR games. These tools will be built on top of the existing MORGAN framework.

8. ACKNOWLEDGMENT

The authors thank their colleagues at the Collaborative Virtual and Augmented Environments Department at Fraunhofer FIT for their comments and contributions, and especially Rod McCall for proofreading this paper. They further wish to thank their project partners of the IPerG and IPCity projects for their ideas, cooperation, and their support. The authors thank in particular Blast Theory and Sony Netservices for the permission to use images from previous gaming performances for this paper. IPerG (FP6-2003-IST-3-004457) and IPCity (FP6-2004-IST-4-27571) are partially funded by the European Commission as part of the 6th Framework.

9. REFERENCES

[1] Bell, M. Chalmers, M. Barkhuus, L. Hall, M. Sherwood, S. Tennent, P. Brown, B. Rowland, D. ,Benford, S., Capra,M., Hampshire, A. 2006. Interweaving Mobile Games with Everyday Life. In Proc. of ACM CHI, ACM, pp 417-426

[2] Benford, S., Seagar, W., Flintham, M., Anastasi, R., Rowland, D., Humble, J., Stanton, D., Bowers, J., Tandavanitj, N., Adams, M., Row-farr, J., Oldroyd, A., and Sutton, J. 2004. The error of our ways: The experience of self-reported positioning in a l location-based game. In Proceedings of Ubicomp 2004 (Nottingham, Sept. 2004). Springer-Verlag, Berlin.

[3] Benford, S., Magerkurth, C., and Ljungstrand, P. 2005. Bridging the physical and digital in pervasive gaming. Commun. ACM 48, 3 (Mar. 2005), 54-57.

[4] Björk, S., Falk, J., et al. Pirates! – Using the Physical World as a Game Board. In Proceedings of Interact 2001, Tokyo, Japan, 2001, 9-13

[5] Broll, W., Lindt, I., Ohlenburg, J., Herbst, I., Wittkämper, M., and Novotny, T.: An Infrastructure for Realizing Custom-Tailored Augmented Reality User Interfaces, IEEE Transactions on Visualization and Computer Graphics, 11 (6), Nov. 2005, 722-733.

[6] Chalmers, M., Barkhuus. L., Bell, M., Brown, B., Hall, M., Sherwood, S., Tennent, P. 2005. Gaming on the edge: Using seams in pervasive games. In The Second International Workshop on Gaming Applications in Pervasive Computing Environments at Pervasive 2005 (Munich). http://www.pergames.de.

[7] Cheok, A, Sreekumar, A. Lei, C., Thang, L., Capture the Flag: Mixed-Reality Social Gaming with Smart Phones, IEEE Pervasive Computing, 2006, 62-69

[8] Cheok, A., Goh, K. H., Liu, W., Farbiz, F., Fong, S. W., Teo, S. L., Li, Y., and Yang, X. 2004. Human Pacman: a mobile, wide-area entertainment system based on physical, social, and ubiquitous computing. Personal Ubiquitous Comput. 8, 2 (May. 2004), 71-81.

[9] Crabtree, A., Benford, S., Rodden, T., et al., Orchestrating a Mixed Reality Game 'On the Ground'. Proc. ACM CHI, 2004

[10] Falk, J., Ljungstrand, P., et al., Pirates: Proximity-Triggered Interaction in a Multi-Player Game. Proc. ACM CHI, 2001

[11] Flintham, M., Benford, S., Anastasi, R., Hemmings, T., Crabtree, A., Greenhalgh, C., Tandavanitj, N., Adams, M., and Row-Farr, J. 2003. Where on-line meets on the streets: experiences with mobile mixed reality games. In Proceedings of the SIGCHI Conference on Human Factors in Computing Systems (Ft. Lauderdale, Florida, USA, April 05 - 10, 2003). CHI '03. ACM Press, New York, NY, 569-576.

[12] Lewis, M. and Jacobson, J. 2002. Game Engines in Scientific Research. *Commun. ACM* 45, 1 (Jan. 2002), 27-31. DOI= http://doi.acm.org/10.1145/502269.502288

[13] Lindt, I., Broll, W.: NetAttack – First Steps Towards Pervasive Gaming, ERCIM NEWS Special Issue on Games Technology, No. 57 (April 2004,). 49-50

[14] Lindt, I., Ohlenburg, J., Pankoke-Babatz, U., Prinz, W., and Ghellal, S. Combining Multiple Gaming Interfaces in Epidemic Menace. Experience Report, In Proceedings of CHI 2006, Montreal, Canada.

[15] Lindt, I., Ohlenburg, J., Pankoke-Babatz, U., S. Ghellal, L. Oppermann, and M. Adams. Designing Cross Media Games. In Proc. PerGames Workshop at Pervasive 2005.

[16] Magerkurth, C., Memisoglu, M., Engelke, T., and Streitz, N. 2004. Towards the next generation of tabletop gaming experiences. In Proceedings of the 2004 Conference on Graphics interface (London, Ontario, Canada, May 17 - 19, 2004). ACM International Conference Proceeding Series, vol. 62. Canadian Human-Computer Communications Society, School of Computer Science, University of Waterloo, Waterloo, Ontario, 73-80.

[17] Magerkurth, C., Cheok, A. D., Mandryk, R. L., and Nilsen, T. 2005. Pervasive games: bringing computer entertainment back to the real world. Comput. Entertain. 3, 3 (Jul. 2005), 4-4.

[18] Malone, T. W. 1981. Toward a theory of intrinsically motivating instruction. Cognitive Science 4, 13 (1981), 333-369.

[19] Ohlenburg, J., Fröhlich, T., and Broll, W. 2005. Internal and External Scene Graphs: A New Approach for Flexible Distributed Render Engines. In Proceedings of the IEEE Virtual Reality Conference 2005, pp. 293-294, Bonn, Germany.

[20] Ohlenburg, J., Herbst, I., Lindt, I., Fröhlich, T., and Broll, W. 2004. The MORGAN Framework: Enabling Dynamic Multi-User AR and VR Projects. In Proceedings of the ACM Symposium on Virtual Reality Software and Technology (VRST 2004, Hong-Kong).

[21] Ohlenburg, J., Lindt, I., and Pankoke-Babatz, U. 2006. Report about the Crossmedia Game Epidemic Menace. In Proceedings of the 3rd International Workshop on Pervasive Gaming Applications (PerGames 2006, Dublin, Ireland).

[22] Piekarski, W. and Thomas, B. 2002. ARQuake: the outdoor augmented reality gaming system. Commun. ACM 45, 1 (Jan. 2002), 36-38.

[23] Singhal, S. and Zyda, M. Networked Virtual Environments. ACM Press, Addison-Wesley, New York, 1999.

[24] Skrypnyk, I., and Lowe, D.G. 2004. Scene Modelling, Recognition and Tracking with Invariant Image Features. In Proceedings of the 3rd IEEE and ACM International Symposium on Mixed and Augmented Reality (ISMAR) 2004, 110-119, IEEE, Piscataway, NJ.

[25] Stapleton, C. B., Hughes, C. E., and Moshell, J. M. 2002. Mixed reality and the interactive imagination. In Proceedings of the First Swedish-American Workshop on Modeling and Simulation (SAWMAS 02. Orlando, FL, Oct. 30-31, 2002).

[26] Xu, K., Prince, S., Cheok, A., Qiu, Y., and Kumar, K. 2003. Visual registration for unprepared augmented reality environments. Personal Ubiquitous Comput. 7, 5 (Oct. 2003), 287-298.

[27] Wagner, D., Pintaric, T., Ledermann, F., Schmalstieg, D., 2005. Towards Massively Multi-User Augmented Reality on Handheld Devices. Proceedings of the Third International Conference on Pervasive Computing (Pervasive 20055), Munich, Germany