Collaborative Gaming in Augmented Reality

Zsolt Szalavári, Erik Eckstein, Michael Gervautz

Institute of Computer Graphics
Vienna University of Technology
Karlsplatz 13/186/2, A-1040 Vienna, Austria
szalavari|eckstein|gervautz@cg.tuwien.ac.at

Abstract: We introduce a local collaborative environment for gaming. In our setup multiple users can interact with the virtual game and the real surroundings at the same time. They are able to communicate with other players during the game. We describe an augmented reality setup for multiple users with see-trough head-mounted displays, allowing dedicated stereoscopic views and individualized interaction for each user. We use face-snapping for fast and precise direct object manipulation. With face snapping and the subdivision of the gaming space into spatial regions, the semantics of actions can be derived out of geometric actions of the user. Further, we introduce a layering concept allowing individual views onto the common data structure. The layer concept allows to make privacy management very easy by simply manipulating the common data structure. Moreover, assigning layers to spatial regions carefully, a special privacy management is often not necessary. Moving objects from one region into another will automatically change their visibility and privacy for each participant.

We demonstrate our system with two example board-games: Virtual Roulette and Mah-Jongg, both relying heavily on social communication and the need of a private space.

Keywords: augmented reality, interaction techniques, CSCW, virtual gaming

1. Introduction

Gaming is getting more and more one of the major application for computer graphics and related industry is growing very fast. We are interested in a computer-based gaming environment which is suitable for multi-player board-games. These type of games are driven by two major aspects: Social communication and the freedom to maintain individuality in a very private space. The social communication aspect can be clearly observed with non-computer based multi-player board-games like Mah-Jongg, Trivial Pursuit, etc. Computer based games, which gather around some classical computer-game ideas (like Jump-and-Run) often fail to support this type of communication.

The other important aspect is privacy. Parts of the “gaming space” like the table in a board-game are public. Every user has the same visual access to this common information. Parallel to that the game has to maintain a private space for the player. This space provides security for individual strategic decisions.

These two aspects are weighted differently in different kind of games. In a game like Trivial Pursuit for example, the social communication has more weight than the private space. Privacy is only needed to hide answers from the other players. Whereas Mah-Jongg needs a private space for each user.

Looking at computer games, we find them ranging from consoles over PCs, workstations, location based entertainment to multi-user networks. The different technologies are suited more or less for board-gaming. Consoles and PCs with people

sitting in front of one display do not restrict social communication. However there is almost no chance for privacy. On the other hand networked games provide large possibilities for privacy but restrict the social communication channel massively.

Lot of expectation surrounds virtual reality (VR) technology to bring new solutions to gaming with computers. Observations shows results against these expectations, however the technology is appealing and everyone wants a personal experience [Pausch, 1996]. Pausch et. al. came to the conclusion that it is possible to suspend disbelief. Yet to convince players is very hard, even with a familiar content and story presented with high fidelity hardware.

Distributed multi-user VR systems like DIVE [Carlsson, 1993] try to incorporate communication for geographically distant users. Communication distribution is additional to application demands. Depending on the type of information that is communicated (avatar representation, text, audio or even video channel), high network performance and intelligent distribution strategies have to support these tasks.

Local multi-user systems like CAVE [Cruz-Neira, 1993], Responsive Workbench [Krüger, 1995], Shared-Space [Billinghurst, 1996] and Studierstube [Fuhrmann, 1997] focused on topics in scientific visualization and collaborative work. Social communication is provided by these systems naturally, participants are located in one room.

We identified the ability to display different information to each participant as a requirement for visual privacy. The CAVE environment provides one projected view to a group of users, thus privacy could only be on a group basis. The Two-User Responsive Workbench [Agrawala, 1997] extends the previous Responsive Workbench approach to two users. Multiplexed field sequential stereo projection provides independent views for participants. The authors also investigate the possibility to display different information to the two users. However they had problems to identify real world applications for this type of collaborative situations. The Shared Space and the Studierstube approach meets the requirements of customized views best, by providing stereoscopic head mounted displays to each participant.

We describe a technology setup which provides both: unhindered social communication and the possibility of private behavior. As we concentrate on board-games the social communication aspect is very important for us, so we restrict ourselves to a setup where all players are in the same room, so that we do not have to communicate over a computer network with restricted bandwidth. For individual view and interaction with the system each player has to have his own head mounted display and interaction tools. In this way not only individual views but also simultaneously and independent interactions with the game are possible.

In the next sections we describe the hardware and software configuration of our system. Natural interaction is very important to our application domain. We show how to overcome limited precision in an augmented reality environment, and how individual views onto a common scenario are managed.

2. Salon de Jeux - A multi-user augmented gaming environment

2.1 General Concept and Setup

The setup we have chosen is similar to that of *Studierstube* [Fuhrmann, 1997], consisting of private see-trough head mounted displays (Virtual I/O i-glasses!) and a Personal Interaction Panel (PIP) [Szalavári, 1997] for each user. HMDs and interaction devices are tracked in position and orientation with a magnetic tracking system. The see-through HMD does not block the view onto the environment, so additional to verbal communication gestures of other players can be recognized. Using dedicated display devices for each user, display of private information is supported. The Personal Interaction Panel consisting of panel and pen allows powerful two-handed manipulation of virtual objects with high precision. Interaction with the game is independent from other users as the interface serves as personal tool.

The system architecture consists of a game server maintaining the graphical database of the game and running simulation based on the input data from magnetic trackers and input devices. Clients for each user render the customized view on the shared environment. Tracker data and information of the buttons on the pens are distributed over the local network using a tracker server. Using a multi-cast group every participant gets synchronized updates of tracker data for rendering. This approach allows scaleability in the number of users to a certain extent. An overview of the system architecture can be seen in figure 1. Figure 2 shows our VR-hardware components as seen from an observer.

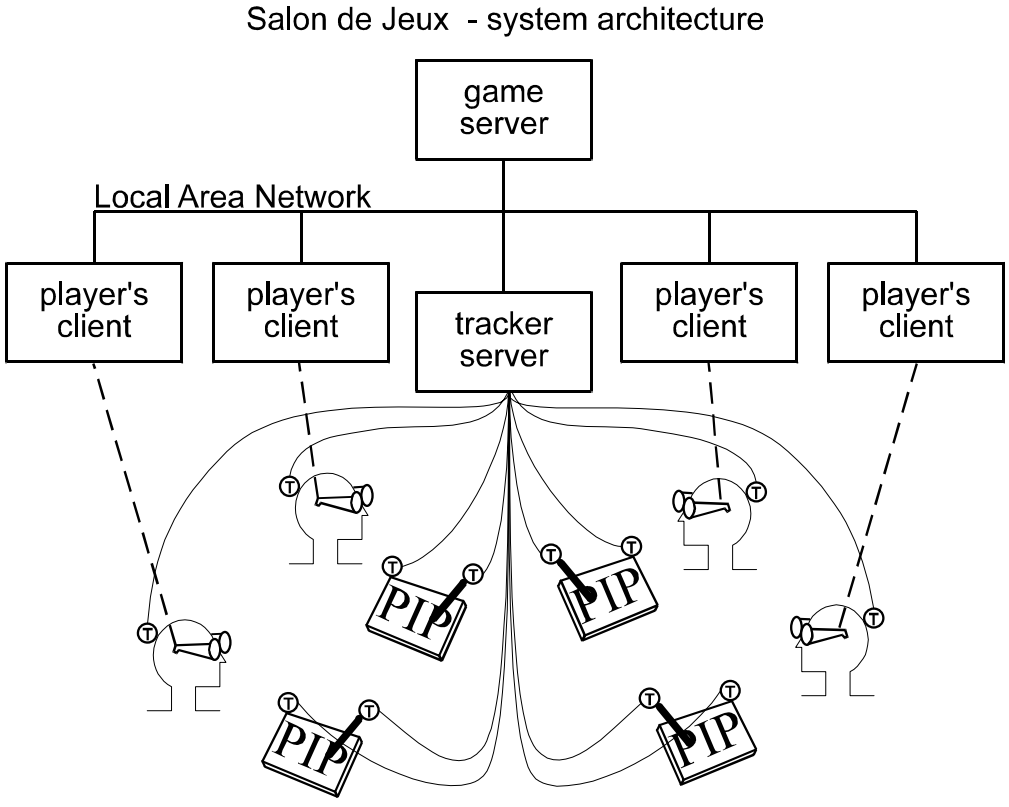

Figure 1. The client-server-tracker architecture of our system.

Figure 2. A view onto the hardware environment visible to the players. Each player wears a see-trough head-mounted display and interacts with a personal interaction panel.

2.2 Benefits of the proposed setup

Our proposed set-up supports the following features, we identified as requirements to multi-user gaming:

- *Augmented reality (AR)*: Compared to virtual reality AR offers a technology to present visual 3D information in a smooth and more ergonomic way. Apart from eye contact social communication channels are not hindered by the technology providing a secure feeling and allowing natural movements. Our experience shows that people wearing see-trough HMDs tend to move more than full immersed users. They do not have the fear to bump into real world obstacles during their movements. Furthermore, compared to VR technology, augmented reality has a lower demand on rendering performance, since only the augmenting information has to be rendered. No additional geometry is needed for way-finding or the creation of immersion like in many VR applications.
- *See-through head-mounted displays*: The stereoscopic display provides a good level of immersion and suspends disbelief on the 3D experience. Furthermore stereoscopy supports the utilized direct manipulation of virtual objects. The individual view onto the environment allows privacy within the common environment. Users can switch on help-engines displaying additional information, without changing the visual representation of the game for other participants.
- *Personal Interaction Panel (PIP)*: We employ a two-handed augmented interaction device for manipulation of virtual objects. The PIP is a combination of a notebook sized panel and a pen. Aligned 3D information is augmented using the HMDs. The simple hardware of the PIP provides good haptic feedback increasing acceptance. The metaphor is based on the everyday knowledge of using pen and paper, not overwhelming players with additional cognitive load. In a similar way other real objects can be incorporated in the game, e.g., a table providing a base for multi-player 3D games.
- *3D-graphics*: Interactive 3D-graphics is a very powerful tool to simulate artificial environments. Using graphics to augment reality, our setup is capable to change the

mood of the surrounding environment intensifying immersion . An appropriate mood can be generated by carefully designing additional geometry to reality.

- *Client-server architecture*: Within a certain range our architecture is scaleable in the number of users. Additional participating players or observers can take part in the game with low additional effort. As simulation is carried out by the server, at any scale only the server has to be powerful enough to run the simulation. Geometric scene complexity has to be scaled to the clients, as they render copies of the common scene.

Summarizing these benefits, the demands of simpler multi-user augmented games on hardware and especially rendering performance are on a level where today's lower end graphics solutions can match the requirements. With our current experience we dare to foresee that our configuration has a good potential to become a multi-user console.

3. Interaction techniques

We augment the real environment with additional information. To avoid context switching from one space to the other, the virtual overlay has to behave like the real world. The less new metaphors we introduce the more natural the interaction will be and the more easy to play.

According to [Hand, 1997] interaction can roughly be categorized into *object manipulation*, *navigation*, and *system control*. Interaction in our gaming applications will cover besides some obvious system controls mostly direct object manipulation. Users in our environment will mainly play the game by manipulating virtual objects (tiles, dices, cards) in front of them. Most of the actions that will occur during direct manipulation are actions such as "Put-That-There" inspired by the motto of [Bolt, 1980]. Our concept relies on a proximity based drag-and-drop model for 3D user interaction. Such actions have in addition to the geometric transformation also semantic meaning.

- *Precise manipulation*: We need to overcome the limited precision of the applied VR hardware. Magnetic trackers provide noisy position and orientation measurements decreasing performance in object manipulation tasks. Instead of applying quality enhancing techniques on raw tracker measurements, like filtering or prediction, we enhance manipulation precision on the scene level. We identified snapping as a very powerful tool for aligning objects precisely, speeding up manipulation tasks.
- *Logic control*: AR technology is closely bound to the real world, as applications run in real space augmenting it. To avoid complicated interaction metaphors overloading users, we needed an interaction technique which is natural and easy to understand. It should be as similar as possible to real world interactions and keep cognitive load on a low level.

3.1 Face-snapping for precise object placement

One of the major problems of virtual reality due to noisy input devices (like magnetic trackers), low resolution HMDs and the lack of haptic feedback is the restricted precision during object manipulation. Objects can not be placed very exactly in space. Especially, moving one object face-aligned onto another which is a very often performed task, is really hard to achieve.

Several solutions have been proposed for that problem. Collision detection [Cohen, 1995], an often used approach in common VR-systems, does not add very much value to that problem. The avoidance of interpenetration does only help a little for the alignment task and is computationally expensive.

In [Mine, 1997] proprioception is exploited to enhance the precision mainly of remote manipulation of objects which are out of reach of the user.

Bier's famous work on snap-dragging in 3D [Bier, 1990] was one of the first solutions for direct 3D manipulation with high precision. Although his method is very intuitive it can not be directly used for direct manipulation, too many additional commands have to be set up during the task.

The whole field of constraint based modeling (summarized in [Juster, 1992]) deals also with high precision. There, constraints are used to automatically keep some kinematic relationships between parts of the scene. Some systems maintain a constraint graph to store this constraints for further use [Fa, 1993]. They describe a method to introduce new constraints into the graph during interactive manipulation.

We use face-snapping, where objects, which are close to each other are aligned automatically by calculating an alignment constraint between the faces – the objects snap onto each other. If an object has to be placed onto another object, the user simply moves it close to the other, face-snapping aligns them automatically.

Our method is similar to the interactive constraint based modeling technique proposed by Fa et. al. [Fa, 1993]. The objective of their work is to built up a constraint graph during object manipulation. A constraint recognition task finds possible constraints during the movement of one object and displays them to the user. If the user does not move further for a certain amount of time, the constraint will be set up in the constraint graph and maintained for further use.

In comparison to Fa's automatic constraint recognition process, our approach does not introduce explicit constraints, which can be inserted and deleted from a data structure. During dragging of an object by the users' pen for each frame all possible snapping conditions are checked and the one with the highest priority is performed immediately. The dragged object is moved according to the geometric constraints defined by the two geometries. Because the detection and calculation process is invoked after every pen movement, the two faces keep snapped until the snapping condition is no longer valid. If the user releases the object during a snap, the object stays aligned with the other, but no constraint is kept for the future.

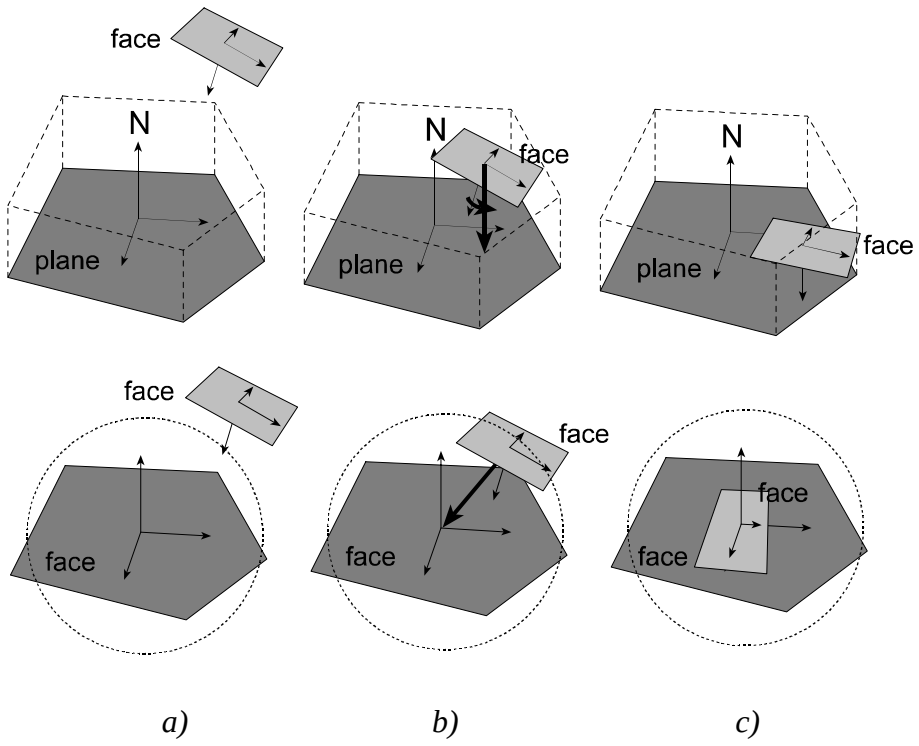

Figure 3. Face-to-plane snapping (upper row) and face-to-face snapping (lower row). A face is approaching a constraint region (a), moves into the region (b), snaps to the plane or face respectively (c).

Looking at the object manipulation requirements of our application (we use mainly put-that-there actions), we find that we need only a very small subset of constraints. We use face-to-plane (binding three DOF) and face-to-face constraints (binding all 6 DOF). The face-to-plane constraint allows 2D translation and 1D rotation on the plane (see figure 3 upper row), the face-to-face constrain fixes the two faces onto each other (see figure 3 lower row). In both cases the snapping condition for the automatic recognition process measures the distance and the relative angles between the faces. The distance is measured between the plane and the center of the face for face-to-plane constraints and between the center of the two faces for the face-to-face constraint, respectively. If both, distance and angle are under a defined limit, the snapping condition is satisfied. To avoid jitter in the snapping, which results from noisy tracker coordinates, we applied a hysteresis on the snapping condition.

In environments with many potential snapping-faces still a high computational effort is needed to find the nearest snapping face pair. This constraint-recognition algorithm must be very efficient to be used in our real time application. We use a axes-aligned bounding-box hierarchy for the static part of the scene, the geometry on each PIP and the dynamic part of the scene. This algorithm is similar to the broad phase algorithms like [Gottschalk, 1996] or [Cohen, 1995] for collision detection. For a small number of dragged objects (each user has just one pen) we can achieve real-time performance.

The result of the constraint recognition is a list of valid snapping face-pairs. For simplicity we only like to process one snapping action at a time. We choose the nearest face pair (i.e. having the smallest difference in angle and distance) to perform the according transformation to the dragged object. This kind of priority selection behaves

also very natural because objects snapping only to near object as it is expected by the user.

Beside the described advantage of fast and precise object placement the snapping movement itself gives feedback for the user that he has placed the object correctly. So there is no additional effort to be done to show that a positioning action has been completed.

3.2 Spatial controlled semantics

Object placement using snapping is not purely a geometric alignment task. This type of manipulations also implicate semantic meaning. The problem is to read the semantic meaning out of the geometric action. Mine et. al. [Mine, 1997] introduced the idea of gestural actions to identify this type of actions and trigger semantic meaning. One nice example was the interpretation of the movement of throwing an object over the shoulder as deleting this object.

Bukowski et. al describe in [Bukowski, 1995] a software framework to populate 3D environments with objects on the "WALKEDIT" desktop based 3D design system. They map 2D cursor positions into 3D space and enhance object placement with pseudo-physical behavior. In a second step they associate objects implicitly based on geometric characteristics, like distance to nearby entities. Associated groups are dynamic and can have hierarchical structure. This work has shown that "magic" - i.e. pseudo-physical rules - can enhance interaction with 2D interfaces and increase productivity.

We introduce the general concept of *regions* as an extension to previous approaches. Regions are dynamic logical groups of objects in a scene. They act as a container to hold groups together, identifying some kind of association of members. To build hierarchical structures regions can also be placed into regions. This allows the hierarchical association of objects contained in this groups. Regions are unambiguous to objects located in them. We assign geometric extent to regions, thus logical groups can be formed by recognizing geometric conditions between this extent and arbitrary objects in the scene.

A specific object is moved from one region into another if it is moved in 3-space out of the area of the one region into the area of the other region. There are three problems with this approach.

- It is very time consuming to calculate geometric intersections between regions and individual objects especially when there are many regions in the environment.
- If regions do not have a visual representation, the user needs a separate feedback for entering and leaving a region.
- If regions overlap, there has to be a simple decision mechanism for each object to which region it belongs.

In our direct manipulation scenario, we identified the geometric condition to be identical with the snapping condition of two faces or a face and a plane. However, a geometric extent is not mandatory to form a group, as described later in the implementation section.

Using this approach, snapping is a mechanism to read out semantic meanings from the geometric actions. The same interaction event can be used to precise direct manipulation and semantic control. Snapping constraints give a visual feedback of docking to the user. In parallel to that a semantic action is triggered to associate the manipulated object to the target, where it was moved to. We call this process *region transition,* assuming that all objects are associated to a region in the beginning. Overlapping geometric conditions are resolved by snapping priorities, assigned knowledge-based. This has to be done in advance, in the design process of the application.

4. The concept of security and privacy

Most multi-user VR applications present the synthetic environment to each user in the same way, albeit from different viewpoint or resolution due to LOD selection algorithms. Like early 2D collaborative systems they simply replicate the common database and show the same visual content.

Smith and Mariani describe in [Smith, 1997] a mechanism to present subjective views in an existing distributed multi-user environment. Query results in the shared database are presented subjectively to users by assigning object modifiers to found entities. Thus relationships and representation of the requested data can be tailored to user specific needs and additional information is not cluttering up the scene for other participants.

Agrawala et. al. present the Two-User Workbench in [Agrawala, 1997] introducing the potential to display customized views to two users. They also propose different partitioning techniques to present information.

While these solutions provide security for visual information, multi-user interaction in shared environments leads to even more complex problems. W. Broll offers in [Broll, 1995] a good overview on what distributed application might implement to handle concurrent object access. The paper identifies locking and the use of master entities as primary solutions for multi-user interaction in VR systems.

We introduce a concept for layering both visual and interaction characteristics of objects. Unlike other approaches privacy information is not stored as object property of each entity. Our investigation has lead to the result, that groups of objects with same security state can be formed. Members of a group inherit security state, transition to another group automatically implies an implicit state change. We identified regions - described in the previous section - as the groups to hold a specific security state. However the approach allows to define a layer to each object, resembling previous work.

In our interactive setting the semantic of security is triggered by geometric actions through following chain (see figure 4): Moving objects in the scene and snapping them into regions implicates a region transition. The transition is a regrouping of the object in the hierarchy of regions. Due to the security inheritance the new logical location implies a new security state both for viewing and interaction as shown in section 3.5.

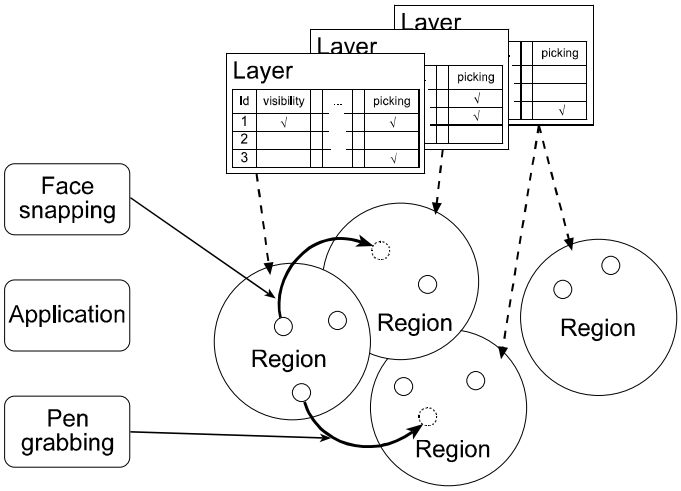

Figure 4. Object transitions between regions is triggered by events coming from snapping, the pen or the application itself. Layers encode privacy information for players and are assigned to regions. Transition of an object (dots in the regions) into regions with different privacy, will change the privacy of the objects automatically.

This approach is simple yet powerful enough to meet most requirements. It is transparent to the application and does not cause additional management load. The number of different security layers is much less, than the number of objects which have to be categorized into these layers. In most cases these layers can be predefined, the application has no additional tasks at runtime for privacy management. It is also possible to change layers apart from moving objects from one region to another.

The application domain of multi-player games is a good test-bed for our security management approach. Depending on the definition of layers, independent subjective versions of the same scene can be presented to participants providing a private view. The private space of one user is protected from other users, while public spaces allow access to everybody. Independent from visual appearance, manipulation or snapping characteristics may be hidden from others.

Implementation of Layers

The simplicity of the layering mechanism can be shown best by a short description of the implementation. We have integrated the layering-concept using the Open Inventor library [Strauss, 1992], which provides a good background due to its architecture. Layers are derived from group nodes acting as general containers for child-nodes. Each layer has a table of bit masks. One bit mask per player is used to enable or disable action traversal of child-nodes for each user. This includes actions like `gl_render_action`, `get_bbox_action` or the `pick_3d_action`-implementation by ourselves. The same scene graph is rendered for each user setting a `player_id` in the root node. This id is passed down in the traversal state, and determines which set of security will be used when rendering or picking objects in the scene. Also bounding-box calculations for snapping can be disabled to prohibit grouping between objects on different layers. It is possible to define more `player_id`'s and sets of security masks than physical users. With this it is possible to switch between predefined security levels, even without changing information in layers at runtime.

A second mechanism to manage region based privacy are *state-switches*. Using additional values passed down in the traversal state, objects lower in the hierarchy can inherit rights from a container. To keep the security management at runtime completely decoupled from objects, we prepare them with state-switches to react on the values in the current state. These values can be set on layers enabled or disabled for specific users. By this approach it is possible to keep privacy management absolutely transparent to the application, if no changes on layer-rights have to be made during runtime. The application has only to be aware of regions, which are statically assigned to layers.

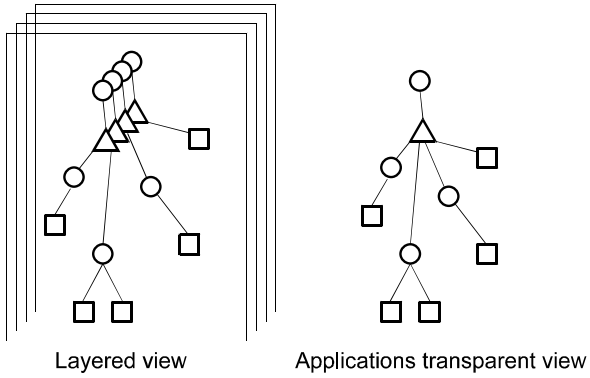

Figure 5. The layered view on the left shows the scene-graph orthogonal structure of layers. Action traversal for different player-ids is split up in the state-switch (triangle). The hierarchy of the scene-graph remains still consistent to the application as presented in the right image.

One can imagine to give regions the ability to hold the user rights, meaning the objects of a region have the rights defined in the region. This approach would result in a lack of flexibility, because all objects of a region automatically have the same rights. The layer-concept is orthogonal to the region-concept (see figure 4 and 5). By putting a region onto a layer, all objects in the region are affected by the layer's rights. But it is also possible to put the objects of a region onto different layers. That means, the objects are in the same region but have different user rights. Our approach allows any combination and resolution of setting rights.

As an additional feature, layers can be organized in a hierarchical structure. Sub layers inherit rights from the super layer. This allows to build groups of layers and to define group rights in an easy way.

Private Help

The layer mechanism can also be used to implement private help. Private help is any kind of help information which can only be seen by the user, who requested the help. Other users can not see the help information and are disturbed by it. Sometimes it is desirable that the application gives a hint to a specific user, which other users should not see, e.g. a teacher – student collaboration, where only the teacher can see the right answer.

To implement private help all the geometry, which form the help information, has to lie on a specific help-layer. Only the user, who should see the help information has rights on the help-layer. Other users do not have rights on this layer and therefore they can not see, nor manipulate the help information.

The technique to layer information is a powerful mechanism, however assignment of rights remains an open question. Using the layer approach regions can be independently enabled to users for viewing and interaction.

5. Implementation and Results

A large number of different games seem to fit excellent into our concept. All kind of non-computer based board games like pictionary, any card game are good candidates but also many of the existing multi-user console games can gain additional benefit of the augmented reality setup. For evaluation of our system we selected two games which are well known around the world and rely on the use of our social communication channel: Roulette and Mah-Jongg.

The current hardware set-up as shown in Figure 1 supports up to four users playing simultaneously in an augmented environment. Rendering is done on a SGI Maximum Impact R10000 with Impact Channel Option (ICO). The ICO is utilized to split the frame buffer into four regions and to convert output to VGA signals. Using converters images are presented in Virtual I/O i-glasses! . Our implementation using Open Inventor can deliver independent line interleaved stereoscopic rendering for four players at about 10 frames per second.

5.1 Virtual Roulette

A good example for the multi-user aspect of gaming is a virtual roulette table with multiple users. The social channel is very important for this game. Because of the almost 50% mean-chance for a win, even with a small amount of players there is almost each turn a winner on the table, giving the whole game a positive mood. A single user setup would make the whole game much less interesting. Individuality is used for the ownership of coins and private help.

Our setup consists of a common roulette-table floating in space, where players can place their coins (see figure 6). The PIP is used as wallet to store individual money of each user. Like in real world players can show or hide their individual money by simply turning the panel away from the other player’s view.

The game allows a couple of users to place their coins onto the table independent of each other. Each user can have his own help displayed individually, without disturbing the others in their task. Money on the table can be seen by every player whereas only each owner can manipulate his money.

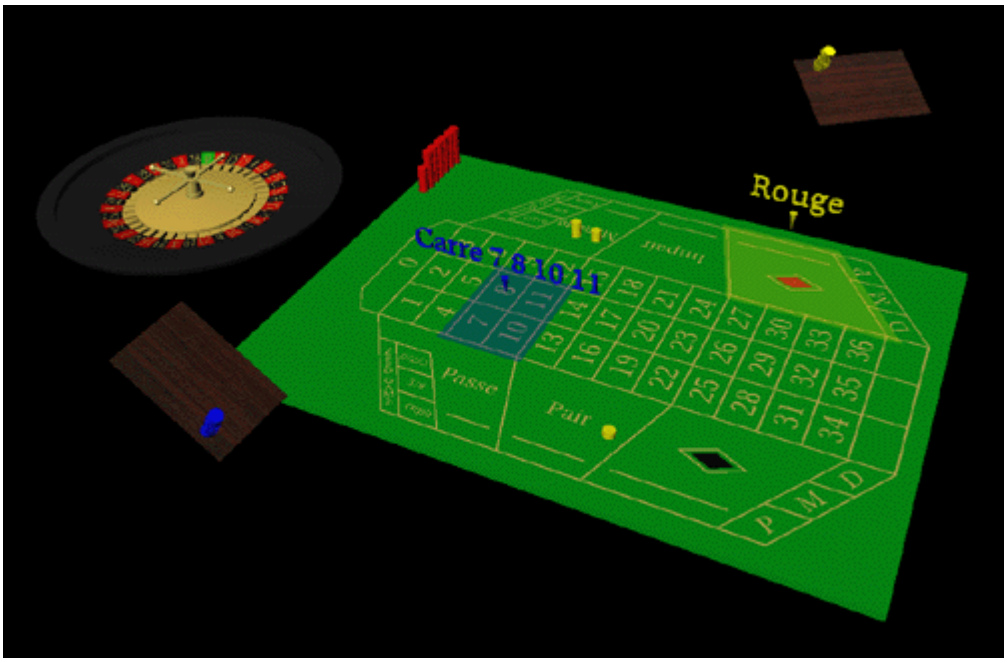

Figure 6. Virtual Roulette, a famous casino game. Our implementation shows the table with multiple users playing. For this snapshot we enabled private help for public by placing it on a public layer.

5.2 Mah-Jongg

The Game

Mah-Jongg is a very old traditional Chinese game, with roots going back into the dust of ancient centuries (figure 7). In the beginning it's been only played by emperors and those in the know, because it's secret. Later public got access to it, and in our century it become popular around the world. The high number of enthusiastic players developed many-many variants of the game, yet the basic rules are still kept.

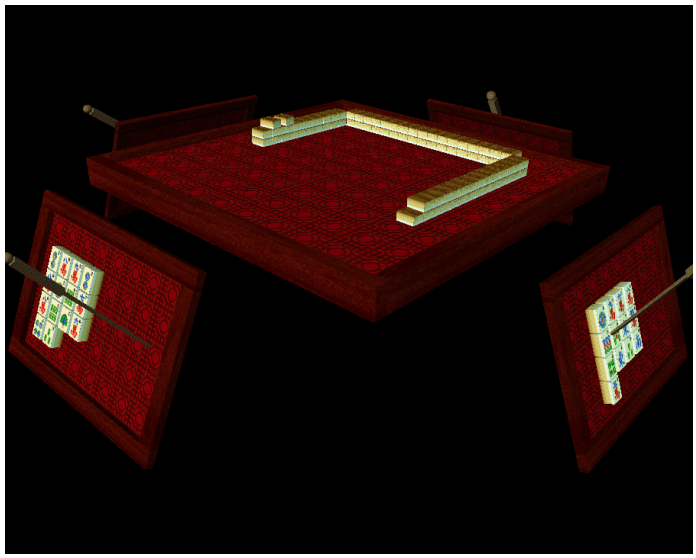

Figure 7. Mah-Jongg. The augmentation overlay showing four PIPs.

As everything in Chinese tradition nothing is left to fortune. Strict rules describe how to take places around the Mah-Jongg table, how to select the first player, and so on. A long procedure of social communication and interaction is happening before the game finally starts. The game is played usually with 136 *tiles* carrying signs and numbers on one side. Preceding the game tiles are put together to form a *wall*, which is a central element in the game. It serves as a common ground where new tiles are drawn from. Each player gets a number of tiles at the beginning. At each step of the game one player draws a new tile, compares it to his tiles, and decides to throw it away or group it with others. The goal of the game is to form groups of tiles, e.g., three tiles with same picture make up a Pung. Different groupings have different scores. The game usually ends when a player finishes a Mah-Jongg. This means that all his tiles are grouped and presented to other players.

We have selected Mah-Jongg as the major implementation example of our concept, because it is known world-wide, the main rules are easy to understand, and many of our system features can be presented with it. Mah-Jongg is played in our setup by sitting around the table wearing see-through HMDs. Each user also has a Personal Interaction Panel to manipulate the augmented game. The panel carries one players tiles, which are manipulated with the stylus.

Spatial controlled semantics

To play Mah-Jongg, mostly tiles have to be transferred from one location to the other. Tiles are being transported from the wall to the users hand, within the users hand from one place to another to form groups, and from the hand to the table. We have defined several regions for the game: the wall region, the table region, the region on the PIP representing the users hand, the region of the pen for the transition of tiles. In contrast to all other regions the pen region has no assigned snap condition geometry. The region transition of tiles into and from the region is triggered by pressing and releasing the button on the pen.

Players can independently manipulate their private tiles. Tiles snap to each other providing visual feedback and to trigger semantics. In this way groups can be formed and braked dynamically by pushing tiles together or moving them apart. Grouping generates additional regions, allowing to add other tiles to that sub-region. Additionally the grouping mechanism gives feedback to the game engine to provide help information and determine game state (see figure 8).

Figure 8. A Grouping sequence. A tile is grabbed with the pen and snapped to another tile to form a group. This action triggers semantics to insert the help-shovel.

Privacy in Mah-Jongg

Naturally the real game relies very much on the honesty of players not to look into other players tiles. This gives the game a secret and mystic touch. In general the rights to see tiles and manipulate the is governed by game rules and tradition. The gaming situation decides what tiles can be picked up by which user. In the real game these conflict situations are solved over the social channel.

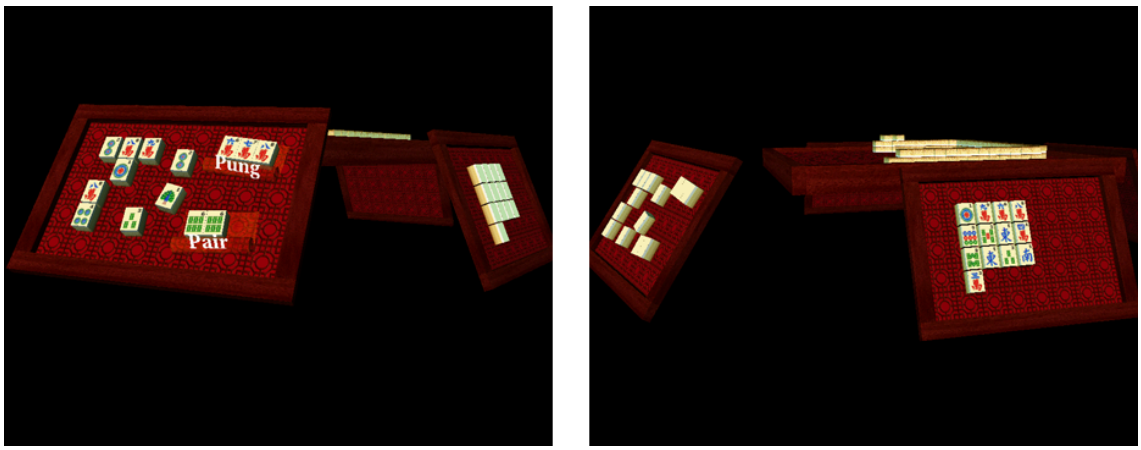

Figure 9. Privacy: The same situation in the game seen by two different players. Characters on the tiles of the opponent as well as help-shovels are not shown.

Our layering concept allows to support this by assigning different security levels to regions. Tiles on a players panel can only be seen and manipulated by himself (see figure 9). The texture containing the tiles sign is switched on and off using a layer switch, inheriting privacy information from the PIP region. Picked up by pointing with the pen inside the tile and pressing the button transfers is to the pen region. The pen region has the same privacy settings like the PIP, so that other players still can't see the texture while manipulating the tile. However the pen has the same security, it was necessary to define this region. The hierarchy of transformations allows thus to define a local pen coordinate system, which is transformed by tracker updates in world coordinates. Moving the tile close enough to the table region, the tile snaps onto the surface, indicating a region transition. Releasing the pen button confirms this transition. As the security definition of the table region enables viewing for each user, the tile texture becomes visible and every player can see it.

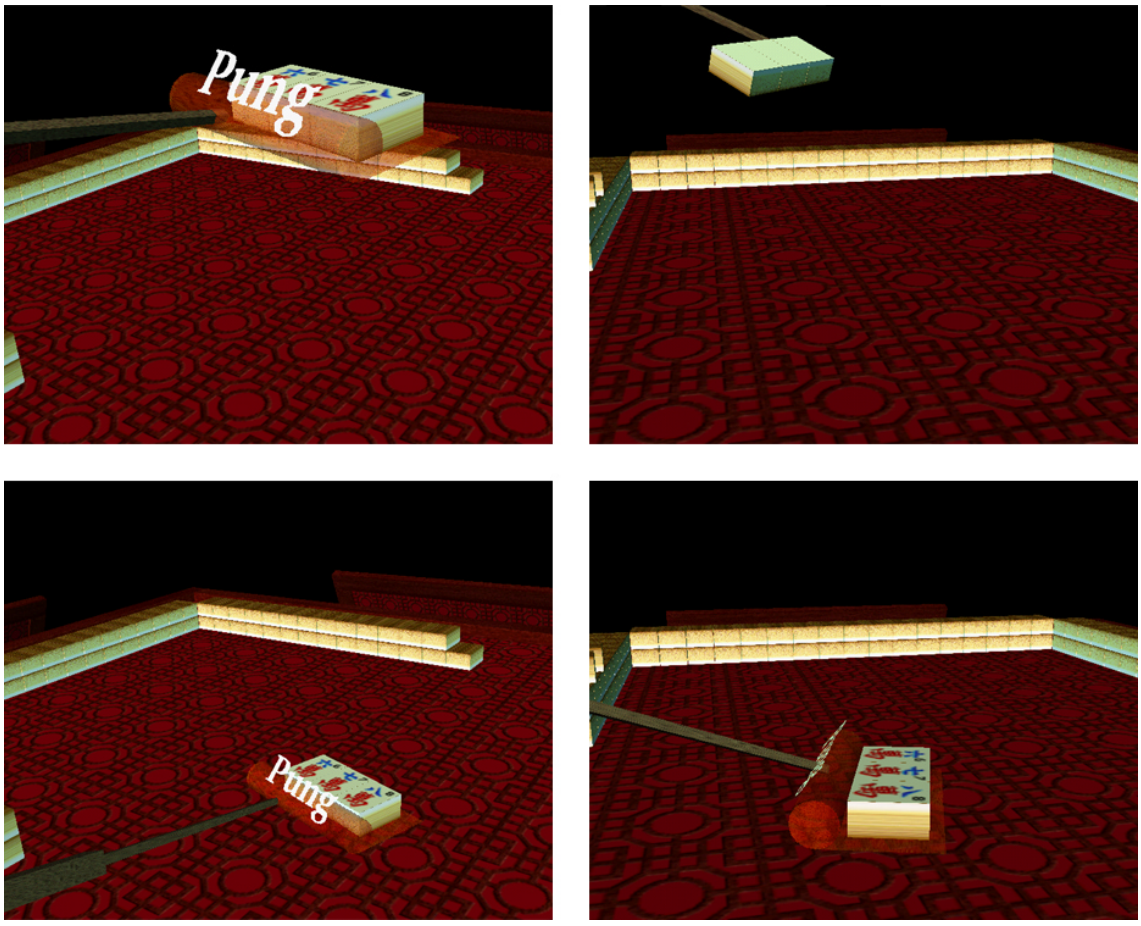

Figure 10. A player is placing a Pung onto the table (left column). A different player is observing this action (right column). This sequence is a good example for a region transition with changing privacy.

Although privacy is supported by information layering, the PIP incorporates simply by its physical properties an additional kind of privacy. Players may hold the panel in a position, so that tiles are visually hidden to other players, however they do not contain any identification in other users views.

Private Help

Additional information is provided to players on private layers. Our Mah-Jongg implementation has a semantic controlled help system supporting visual feedback for understanding and additional geometry for group manipulations. Forming groups of tiles results in the creation of a *shovel* placed under the grouped tiles. Text on the handle of the shovel indicates valid combinations. This kind of help-information can be used for more than status reporting. The owner of the group can also move the whole group by picking the handle of the shovel and moving it to another region. On the table shovels get displayed to every user to easy calculation of scores at the end of the game.

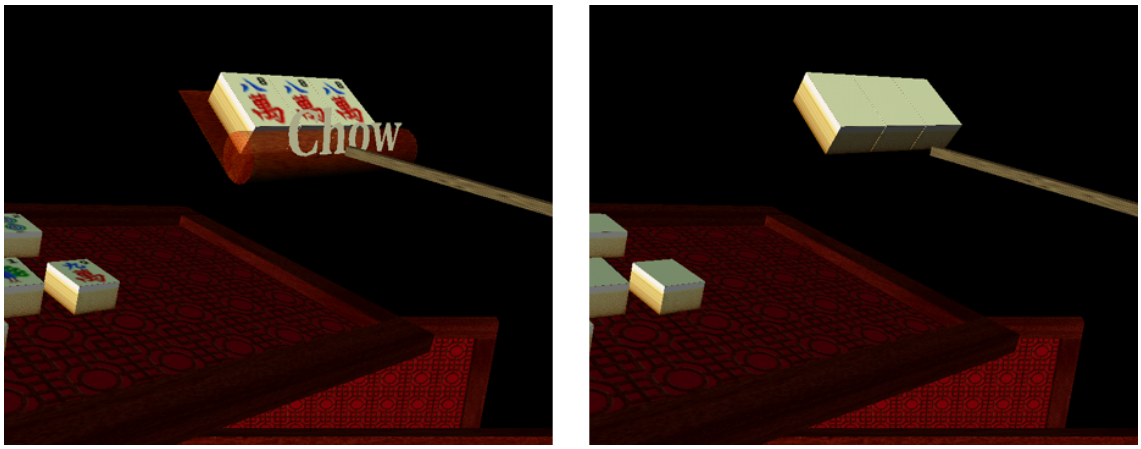

Figure 11. Private help expressed as a shovel. This additional handle helps to manipulate groups together. The same situation from another viewer is shown on the right.

6. Conclusions

We described a multi-user augmented reality gaming system for board games which maintains social communication and provides private space within the common game to allow individualism. We have made prototype implementations of Roulette and Mah-Jongg to test our setup and interaction techniques.

What have we learned? First of all, it makes great fun to play in such an environment. Even very inexperienced users find out very fast how the game is to be played without any additional information on the general use of the system. The shortcomings in precision of the hardware can be compensated by adequate tools like face-snapping. Especially in regions where there are only a few snapping faces, the snapping conditions can be very generous, so that object snap onto each other even if objects are not so close. This allows very fast actions to be performed giving the game the chance to be a little bit dynamic.

If object manipulation is restricted to drag-and-drop like actions, as presented in our examples, snapping is a powerful substitute for collision detection in virtual environments. As objects snap to each when they are near, interpenetration happens very seldom and only in cases which do not disturb the user. Moreover, in some cases collision detection would hinder easy manipulation of objects more than it would help.

Private help is seamlessly integrated in our approach, and is a natural extension of the game itself. In our tests, users have turned on private help all the time.

Currently our system consists of a commercial available standard hardware, but we see a good chance that the system could be produced as a console game for multiple users in future.

A ”game-box” could contain game-server and rendering clients as well as the tracker source. If the game-box is placed on a table, players can sit around that table holding their pen and panel, the game-board can also be displayed on that table which also gives a haptic feedback.

As our hardware-setup is lent from a scientific-visualization system it is only natural to project our interaction techniques and privacy concepts back to that application area. We think that for scientific visualization our whole system-concept fits very well. Multiple scientists discussing one common visualization are able to switch on and off individualized information they like to see personally. Simplifications induces by the gaming domain could be removed to support other type of applications. We see a great potential for our setup to be used also for education- and presentations-system in the near future.

7. Acknowledgments

This work was funded by the Austrian Science Foundation (FWF) under project no. P-12074-MAT and the City of Vienna. Special thanks to Toni Fuhrmann for providing the tracker-implementation, Herbert Buchegger and Reinhard Sainitzer for helping with the implementation of Virtual Casino, and Dieter Schmalstieg for valuable discussions.

8. References

[Agrawala, 1997] Agrawala, M., Beers, A. C., Bernd Fröhlich, Pat Hanrahan, The Two-User Responsive Workbench: Support for Collaboration Trough Individual Views of a Shared Space, *SIGGRAPH 97*, August 1997, pp. 327-332.

[Bier, 1990] Bier, E.A., Snap-Dragging in three Dimensions, *Computer Graphics, Symposium on Interactive 3D Graphics '90*, 24(2), March 1990, pp. 193-204.

[Billinghurst, 1996] Billinghurst, M., Weghorst, S., Furness, T. III. Shared Space: An Augmented Reality Interface for Computer Supported Collaborative Work. *Proceedings of Collaborative Virtual Environments'96*, 1996.

[Bolt, 1980] Bolt, R., A., "Put-That-There": Voice and gesture at the graphics interface, *Computer Graphics (SIGGRAPH '80 Proceedings)*, 14(3), pp. 262-270, July 1980.

[Bukowski, 1995] Bukowski, R. W., Séquin, C. H., Object Associations - A Simple and Practical Approach to Virtual 3D Manipulation, *1995 Computer Graphics, Symposium on Interactive 3D Graphics '95*, April 1995, pp. 131.-138.

[Carlsson, 1993] Carlsson, C., Hagsand, O., DIVE - A Platform for Multi-User Virtual Environments. *Computers & Graphics*, Vol. 17, No. 6., 1993, pp. 663-669.

[Cohen, 1995] Jonathan D. Cohen, Ming C. Lin, Dinesh Machova, Madhav Ponamgi, I_COLLIDE: An Interactive and Exact Collision Detection System for Large-Scale Environments, *1995 Symposium on Interactive 3D Graphics*, 1995, pp. 189-196.

[Cruz-Neira, 1993] Cruz-Neira, C., Sandin, D.J., De-Fanti, T.A., Surround Screen projection-based virtual reality: The design and implementation of

the cave, *Computer Graphics (SIGGRAPH'93 Proceedings)*, Vol. 27, August 1993, pp. 135-142.

[Fa, 1993] Fa, M., Fernando, T., Dew, P.M., Direct 3D Manipulation Techniques for Interactive Constraint-based Solid Modeling, *EUROGRAPHICS' 93 Conference Proceedings*, 12(3), 1993, pp. 237-248.

[Fuhrmann, 1997] Fuhrmann, A., Löffelmann, H., Schmalstieg, D., Collaborative Augmented Reality: Exploring Dynamical Systems, *Proceedings of Visualization 97*, October 1997, pp.459-462.

[Gottschalk, 1996] Gottschalk, S., Lin, M.C., Manocha, D., OBB-Tree: A Hierarchical Structure for Rapid Interference Detection, *SIGGRAPH 96, Conference Proceedings*, 1996, pp 171-180.

[Hand, 1997] Hand, C., A Survey of 3D Interaction Techniques, *Computer Graphics Forum*, 16(5), Dec. 1997, pp. 269-281.

[Juster, 1992] Juster, N.P., Modelling and Representation of Dimensions and Tolerances: a Survey, *CAD*, 24(1), Jannuary 1992, pp. 3-17.

[Krüger, 1995] Krüger, W., Bohn, C.-A., Fröhlich, B., Schüth, H., Strauss, W., Wesche, G., The responsive workbench: A virtual work environment, *IEEE Computer*, July 1995, pp. 42-48.

[Mine, 1997] Mine, M.R., Brooks Jr. ,F. P., Sequin, C. H., Moving Objects in Space: Exploiting Proprioception In Virtual-Environment Interaction, *Computer Graphics, SIGGRAPH '97 Conference Proceedings*, 1997, pp.19-34.

[Pausch, 1996] Pausch, R., Snoddy, J., Taylor, R., Watson, S., Haseltine, E., Disney's Aladdin: First Steps Towards Storytelling in Virtual Reality, *Computer Graphics, SIGGRAPH '96 Conference Proceedings*, 1996, pp.193-203.

[Smith, 1997] Smith, G., Mariani, J., Using Subjective Views to Enhance 3D Applications, *Proceedings of VRST'97*, 1997, pp. 139-146.

[Strauss, 1992] Strauss, P.S., An Object-Oriented 3D Graphics Toolkit, *Computer Graphics, SIGGRAPH 92*, 26(2), July 1992, pp. 341-347.

[Szalavári, 1997] Szalavári, Zs., Gervautz, M., The Personal Interaction Panel - A Two-Handed Interface for Augmented Reality, *Computer Graphics Forum (Proceedings of EUROGRAPHICS'97)* 16(3), September 1997, pp. 335-346.