An audio visual projection system for virtual room inhabitants

Jochen Ehnes*

The University of Edinburgh

Abstract

In this paper we describe a system to project virtual characters that shall live with us in the same environment. In order to project the characters' visual representations onto room surfaces we use a controllable projector. But characters are not just visible projections, they shall be able to speak as well. In order to make the sound seem to appear from the projected character, we built a controllable sound projector consisting of an 'audio spotlight' (highly directional ultrasonic speaker) mounted in a steerable gimbal. We describe in detail how we built this sound projector and how we integrated it with our character simulation and projection system. We also describe the steps necessary to use a commodity controllable projection system and our method to calibrate both the video projector's and the sound projector's position and orientation in the room.

Index Terms: H.5.2 [Information Systems]: Information Interfaces and Presentation—User Interfaces;

1 Introduction

Dialogue systems are applied frequently to automate telephone services. However, it appears to be the case that some people seem to feel uncomfortable to talk to such a system as a voice in the room without a visual point of focus to it. The expectation to be able to see the dialogue partner can be an important factor if we for example intend to create reminder and support systems for elderly people living at home by themselves, one of our future goals. Those people, who may be suffering from certain forms of dementia, could get very confused if there appears to be a voice talking to them but they cannot connect the voice to a person. Obviously this has to be avoided, or the system might do more harm than it provides support.

With our current project we aim at creating a visual representation of virtual characters that shall live in our environment, created by an extended dialogue system. With this system we plan to verify the existence of this effect and investigate how well different types of visual points of focus or virtual characters may help to compensate for it.

While robots are often seen as the natural way to create virtual partners with a physical presence, they bring with them a whole set of other challenges. Using robots as interaction partners always carries the risk of them getting lost or stuck somewhere, stumbling over objects lying on the floor or in the worst case hurting someone. By using projected virtual characters we basically can avoid all these problems and yet produce a visual point of focus for conversations in our environment, which can follow human users or lead them to somewhere else. Of course the fact that we can only project onto surfaces such as walls, the floor, ceiling or table tops has its own drawbacks, but we believe that a smart choice of the type of virtual characters used can make these limitations less apparent and lets those characters be believable despite the limitations.

The basic concept of our approach is to create virtual surfaces that coincide with the real surfaces of the room on which the virtual characters live. The controllable projection system we developed creates the perfect match between the virtual and the projected surfaces even while it moves. This way it keeps up the illusion that the virtual surface and the real surface are the same and thus that the virtual characters are in the same room with us. In order to make these projected characters believable dialogue partners, it is important that the sound of their speech appears to originate from the location of their projected image. We therefor built a "controllable sound projector" consisting of a highly directional ultrasonic speaker in a controllable gimbal.

*e-mail: jehnes@inf.ed.ac.uk

20th International Conference on Artificial Reality and Telexistence (ICAT2010), 1-3 December 2010, Adelaide, Australia
ISBN: 978-4-904490-03-7 C3450 ©2010 VRSJ

2 Related Work

2.1 MR Projection Systems

Mixed reality interfaces involving a combination of physical objects and computer displays (usually projected) have been investigated since the 1990s. Examples include Wellner's DigitalDesk [18], Ishii and Ullmer's metaDESK [15], as well as Underkoffler and Ishii's I/O Bulbs [17, 16].

The Everywhere Displays project [11] uses a controllable mirror and a steerable camera to project interfaces onto different surfaces, dependent on the user's position. This system required a few seconds to switch between different pre-programmed display locations and did not project anything while the mirror moved. So while this system is able to simulate different fixed projection systems in a room, it would not be able to let a virtual character walk across the room in one continuous motion.

The author himself developed an augmented reality (AR) system based on controllable video projectors [4, 5]. This included a roaming architecture that enabled the applications responsible for the augmentation to migrate between different projection units in order to follow the objects and users. User tests were performed to examine the quality of projection as perceived by human subjects dependent on projection distance and angle in order to optimise the selection of the active projection unit at runtime. While this system was able to project content in a reasonable manner while it was moving, it did not take the projector's offset in the gimbal into account properly. It only used an estimated value, which led to some distortions outside of the centre of the projection. The calibration procedure described in this paper provides a way to determine this offset together with the field of view, which results in pixel accurate projection across the whole projection area.

In order to be able to augment the environment with a pan- and tilt-able projector, the projector should ideally be placed so that the centre of projection and the pivoting point coincide. Mitsugami et al built such a projection system and described how to move the projector to exactly the right spot in the gimbal [10]. As we use an off the shelf controllable projection system which does not allow for the projector to be moved within the gimbal, we could not use their method of calibration. Yet we were inspired by their use of two projection surfaces at different distances in order to calibrate the system. Equally, the use of virtual projection surfaces on which our characters move was partially inspired by this work.

For completeness sake we also have to mention efforts in the field of projector camera research on calibration, such as [14, 6]. However, while it is impressive to be able to measure geometrical changes of the projection surfaces with imperceptible structured light or to be able to move an intelligent projection unit, which in turn compensates for that and realigns its projected image with that of its partner units in about ten seconds, it is not really applicable for our application. For once, our controllable projection units are usually mounted in a fixed location. For that reason that location needs to be calibrated only once and even if we move our system once in a while it does not warrant the effort to set up a camera based calibration system. Furthermore, when the projector is rotated, the image it projects has to be adapted instantaneously (in fact even before the projector moves in order to compensate for rendering delays) and a delay of ten seconds would be completely unacceptable. In a similar notion we want our characters to stay out of the way where things are moved around a lot as that can easily create situations where an object placed in front of the projector could block the character from walking on its intended way. Instead we plan to let characters walk mainly on areas which are not as easily obstructed. That said, it may be worth while in the future to track objects to prevent such situations anyway and allow the characters to go beyond the safe areas.

Ashdown et al developed a calibration method for steerable projectors with a controllable mirror using image processing [1]. While we could imagine to adapt this method to the projection system we used, we were looking for a more practical approach to get the projector's field of view as well as the offset of the centre of projection from the pivoting point. We believe that for the type of projection systems we use, our approach (section 4.3) offers a simple and straight forward way to calibrate these two parameters. We do not need a camera or anything more sophisticated than a measuring tape and two surfaces at different distances to project onto.

Kruppa and colleagues [8, 7] developed a system to migrate and project virtual characters in a museum environment. Although these virtual characters were able to guide people around an exhibition, they moved around on users' PDAs and only at certain exhibits they could migrate from the PDA to the wall next to the exhibit and back. Users would interact with these characters only through their PDAs. Furthermore, the virtual characters were created using a fixed set of animations, leading to visible inaccuracies when the virtual room inhabitant walked along the wall, projected by a controllable projector [9]. In order to cope with this, the characters were usually transformed into a more abstract form, such as a circle or ball, that could be moved without animation.

The sound of the virtual room inhabitant talking was delivered via several speakers in the room. Depending on the location of the character and if known the the user, the system selected up to 3 speakers and made them emanate the signal with particular gain values to place the virtual sound between the active loudspeakers.

2.2 Spatial Sound

While we did not have a chance to listen to the spatial sound of Kruppa's system, we did know from experience with similar systems that the locality of the virtual sound source is not so clear unless one uses a large number of speakers and it can be way off if one does not stand in the system's sweet spot.

2.2.1 Wave field synthesis

The principles of Wave Field Synthesis (WFS) were developed by Berkhout et al [3, 2]. The idea is to create an acoustic wave field by superposition of sounds created by a large array of speakers. The sound signals created by different speakers here do not only differ in their amplitudes, but also in their phases. With this technology virtual sound sources can be placed even behind or in front of the speaker array. Also, with this technology the generation of the signal does not require the position of the listener and the sound appears to originate from the same spot for anyone in the sound field. The disadvantage of this technology however is that even to be able to place virtual sound sources in a single plane it requires a large number of speakers. In order to create sound that seems to appear from any location in 3D space, it would require even more speakers, which makes it inapplicable for use in normal environments.

2.2.2 Ultrasonic Directional Speakers – Audio Spotlights

These speakers create ultrasound with the audio signal modulated onto it. Due to the short wavelength of the ultrasound, a focused beam of ultrasound can be created by a relatively small array of ultrasonic transducers integrated in the speaker. The audio signal is demodulated by nonlinear effects of the air and as a result the sound signal becomes audible where the ultrasound hits the boundary between air and a solid object. This may be the human ear, if it is directed at the listener directly, or a room surface. We use this indirect way via a room surface to create the impression that the projected character talks.

Technology leading to the directional ultrasonic speaker we use was originally developed by the US Navy and Soviet Navy for underwater sonar in the mid-1960s. Yoneyama, Fujimoto, Kawamo and Sasabe analysed the use of the system in air to create sound in a highly directional manner [19]. They also coined the term 'Audio Spotlight'. Pompei [12] described a device that reduced audible distortion essentially to that of a traditional loudspeaker. He made the system commercially available with his company under the name Audio Spotlight. Others have since offered comparable products (HyperSonic Sound (HSS) by American Technology Corporation or Sonicast by DigiFi).

3 Virtual Characters that 'live' in the environment

In contrast to the work by Kruppa et al. [7], we wanted to develop virtual characters that exist continuously in the physical environment, moving across the surfaces of the room in a believable way. From our perspective the main weakness of Kruppa et al.'s Virtual Room Inhabitant was that it could not move around the room in a believable way. The two main reasons for that as we see it are:

1. The use of a set of predefined animations. As a result of this the leg motion usually does not fit well with the character's motion as well as the motion of the mouth did not correspond well with the character's speech.

2. The use of a human like character. The way we experience human beings everyday is them standing in the room or next to us, but rarely directly in front of a wall. For that reason alone a flat projection on the wall could hardly stand in for human beings. But it gets even more problematic once the projected character is supposed to move around. A believable human like character would have to walk with its feet on the ground. That however is often not possible along the wall because of objects like tables or cupboards standing there. Similarly an open door creates an empty space where the character cannot be projected and hence characters cannot move across it.

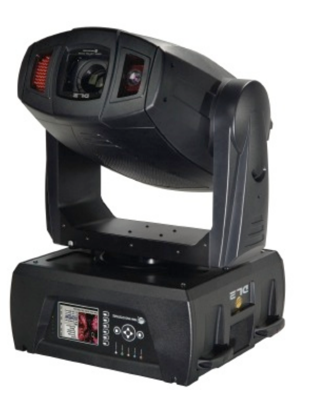
Our system's design addresses both of these points. For once, the motions of the characters are not predefined, but based on physical simulations of the characters as they move and the generated speech as they talk.

And second, as human-like characters and their way of moving around are not very suitable as projected characters, we decided to use characters in the form of cartoon animals, such as a gecko that can walk on walls as well as on the floor, ceiling or tabletop. This way the characters can walk around obstacles in a believable way. Even very cluttered rooms usually have plenty of space for characters to move around on the ceiling. Furthermore, since such animals can usually only be seen from a certain distance and they have a relatively flat appearance to start with, human beings should find their projected images much more acceptable than projected Human like characters. While the uncanny valley, a term coined by Masahiro Mori in 1970, strictly speaking does not apply here, as it only describes what happens when we try to make realistic looking artificial humans, we believe that it can be generalised to all things human beings became accustomed to. Flaws in the artificial version will become more obvious the closer the artificial version gets to the original. At the same time those flaws will also get more obvious the more familiar observers are with the original. In our case, since human beings in general are less familiar with geckos than with other human beings, we expect flaws in our representation of geckos (e.g. that they are a flat projection on the wall) to be much less distracting than the same flaws would be in a human like character.

4 The Video Projection System

While an accurate simulation of foot and body movements are very important to generate the level of believability and presence we aim for, it is equally important that the projection system overlays the virtual surfaces and the characters crawling on them very accurately onto the physical surfaces. A fixed foot would still be drifting if the projection system was not able to stably overlay the virtual objects onto the real surfaces while it moves. For that reason it was crucial to accurately represent and calibrate position and orientation of the projector within the gimbal as well as the projection parameters of its lens in the virtual scene.

While we were considering to build our own controllable gimbal in the beginning as that would have given us the highest level of control over the projector's movement, we realised that this would have been too much effort to be feasible for this project. The mechanical requirements for the construction and building of a gimbal holding and moving the projector with the required accuracy were to high to start from scratch. Instead we decided to go with a commercially available system with the option of replacing its controller should that be necessary. We went with a DL.2 from High End Systems, which can be seen in figure 1(a). In the following we briefly describe how we integrated this commodity hardware. After all, the DL.2 as well as similar products was not designed for this particular purpose. We realise that some of the details may be different with other devices, but in general the approach would be the same. In particular, every system will require a motion simulation to estimate the projector's orientation at the time when the image being rendered is going to be projected. The details of the simulation however may differ as the devices use different controllers and possibly different control algorithms.


(a) DL.2 From High End Systems / Barco

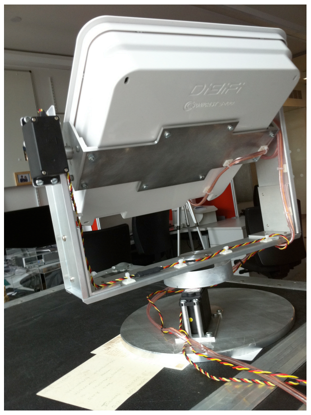

(b) The ultrasonic directional speaker Sonicast S100 in our controllable gimbal

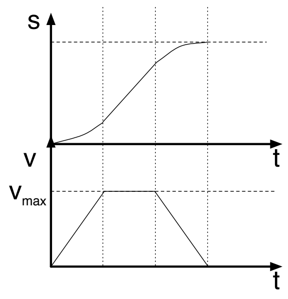
Figure 1: The two controllable pan and tilt units of our system.

4.1 ArtNet interface:

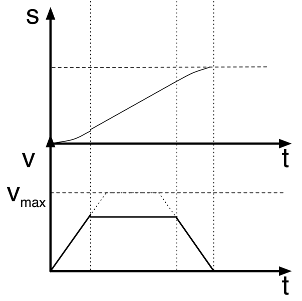
As the DL.2 supports ArtNet, which is basically DMX512 (a standard for controlling lighting equipment) over ethernet, we could make use of that to send control values to the controller. We implemented this interface based on libartnet, an open source implementation of the ArtNet protocol designed for POSIX systems. On top of that interface we created an abstraction layer that accepts angles for pan and tilt and converts them into the DMX values that represent them.

4.2 Motion simulation:

DMX512 and in consequence ArtNet are basically unidirectional protocols intended to control lighting fixtures from a control desk. And although back channels have been introduced in newer versions of these protocols for status messages, the DL.2s, as well as other similar devices we know of, do not provide any feedback about where the projector actually points at while it is moving. In consequence it is necessary to simulate the device's motion in order to be able to create the correct image to be projected. The simulation we developed is based on a speed parameter that can be sent to the projector, which determines the intended duration of the motion to a new position. In the simplest case the system starts from a fixed position accelerates for a while, then moves at constant speed and finally decelerates again to stop at the goal position after the intended duration. Our trials showed that the DL.2 accelerates and decelerates with a constant rate and it has a maximum speed. Without the speed parameter set, the projector would accelerate until it either reaches half its distance or its maximum speed. If it is at half of its distance it switches to decreasing, otherwise it stays at this speed until it reaches the location from where it has to decrease speed in order to come to a stop at the desired distance. However, as we found out from the developer, it does not behave easily predictable in that mode, as they used different speed ramps for different situations/distances. It gets much more predictable when the speed is limited by setting the speed parameter. As the projector times itself to reach the given position after the defined duration (unless that would require it to move faster than it can), it usually accelerates and decelerates for a shorter period of time and moves at a lower constant speed for longer. Figure 2 illustrates that.



(a) Linear acceleration and deceleratio to and from v_{max}



(b) When the distance decreases or the intended duration is extended, the system accelerates to a lower speed, where it stays for longer

Figure 2: Standard accelerate, linear motion, decelerate cycle.

Based on the laws of classical mechanics

$$s = v \cdot t \tag{1}$$

for motion at constant speed and

$$v = a \cdot t \qquad \text{and} \qquad s = \frac{1}{2} \cdot a \cdot t^2 \tag{2}$$

for accelerated motion, and given the time t, distance s and the constant acceleration a (as well as the maximum speed for unusual long distances), we can calculate the times needed to accelerate and decelerate as well as how long the device moves at a constant speed in between.

Of course, it is in no way guaranteed that the projector is in stop state when it receives new position data. In fact it is very likely that it is already moving and this motion has to be taken into account when we calculate the times for acceleration, constant motion and deceleration.

After finding the correct constants for acceleration (110 $degrees/s^2$) as well as maximum speed (101.2 $degrees/s$) of the pan and tilt drives, we got very accurate results. It has to be noted that due to the fact that the device is moved using stepper motors, the motion is very repeatable and predictable. As a result the match of virtual and real surfaces can be kept stable during usual motions. Of course even such good results can be weekend by sending position updates too often, as that constantly starts the new simulation based on old simulation values and consequently small errors sum up to become quite noticeable. We get much better results by reducing the number of updates sent. We reduced it to three to four updates per simulation period (defined by the speed parameter), which is a good compromise. This limits the number of updates of the simulation based on simulated start values, while at the same time it allows the system to generate a smooth and continuous motion instead of moving to the next position and stopping there before moving to the following one. To find the goal positions, we extrapolate where the character will be at the end of the simulation period that is about to begin (see section 7.1), which gives the device exactly the time it needs to get there. In our tests we found that we get the best results with a time parameter defining the duration to slightly more than a second, which made the projector's motion quite smooth while still being able to follow the character on very curvy paths.

4.3 Calibration of projector and virtual camera:

In order to be able to project images of virtual objects placed on surfaces of the virtual room onto the corresponding real surfaces, the virtual camera used to render the projection images has to match the projector mounted in the pan and tilt device exactly. Furthermore the projector should ideally be mounted in the gimbal so that its projection centre falls into the pivot point, which is what Mitsugami et al did in [10], as that avoids possible parallax errors. When using off the shelf hardware however this is not possible. For that reason we had to compensate the offset in software.

As the projector is mounted in a way that puts is centre of gravity into the pivot point, its lens sits clearly in front of that point. Furthermore the projector is usually not perfectly aligned within the gimbal. Consequently the virtual camera has to be moved forward and rotated slightly from the origin of the pan and tilt device, the pivot point.

As the centre of projection or the point that corresponds to the pinhole in the pinhole camera model is not clearly defined for the projector's lens, we had to devise a way to optimise this offset parameter together with the view angle, which has to match the projection angle of the lens. As these two parameters determine the projection, it was impossible to determine any of them by projecting onto a single surface only. Instead one can only determine them by adjusting them for two projection surfaces at different distances.

We calibrated these values by projecting a grid pattern onto two surfaces at different distances from the gimbal's pivot point and adjusting both values until we got the exact grid size on both surfaces (figure 3). In order to do so, we wrote a set of vertex and fragment shaders that renders any triangle or polyhedron overlaid with a grid based on the vertex coordinates. Since we model everything in millimetres, we set the red component to 1.0 if the fragment's

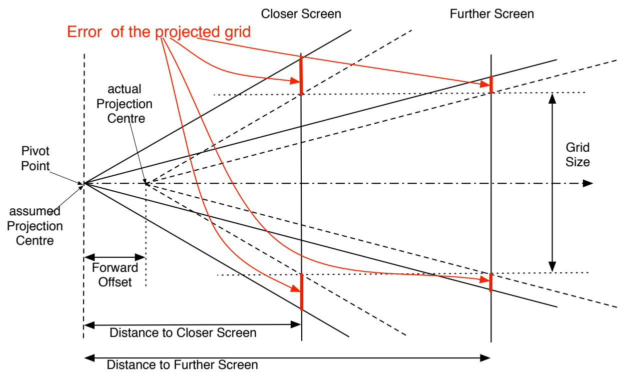

Closer Screen
Further Screen
Error of the projected grid
actual Projection Centre
Pivot Point
Grid Size
assumed Projection Centre
Forward Offset
Distance to Closer Screen
Distance to Further Screen

Figure 3: If the distance between the centre of projection and the two projection screens is is wrong it leads to different errors of the projected grid on both screens. This can be used to find the offset of the centre of projection in relation to the gimbal's pivot point, as the distance between the pivot point and the two screens can be measured accurately. As a wrong value of the field of view parameter affects the projections on both screens to the same degree, it can be distinguished from a wrong forward offset and hence both can be adjusted correctly.

x coordinate modulo 100.0 is smaller than 2.5 and 0.1 otherwise. We do the similar thing for the green and blue components based on the y and z coordinates. As a result everything we project is covered with a 10 cm grid aligned to the object's coordinate system. As the next step we set up two projection surfaces in known distances (the closer one being about half way between the projector and the second one) to the projection system's pivot point and rendered two quads in the same distances from the origin. Then we tweaked both parameters, the field of view as well as the offset of the camera from the origin (i.e. the distance of the projection centre from the pivot point) until the gridlines on both projection surfaces were exactly 10 cm apart. A good way of doing that is to adjust the field of view to the get the 10cm grid on the distant plane. If the grid on the closer surface is smaller than 10 cm, it means that the projector is actually closer than it should be and hence the virtual camera has to be moved closer towards both screens by increasing the offset to compensate for that. Of course this will in turn change the size of the grid on the far surface as well, so the field of view has to be adjusted again and so on. While this may sound complicated, as it involves manual tweaking of two parameters, it can easily be done in less that 10 minutes. As this has to be done only once for a device, if not only once for all devices of the same make, we did not see a need to try and automate this process. However, if large numbers of devices were to be calibrated, this could be automated by visually comparing the projected grid to calibration patterns on the two screens with two cameras

After the distance of the projection centre (the translation along the Z coordinate in the camera's coordinate frame) is known, we have to determine the offset in X an Y direction from the pivot point as well as calibrate the alignment of the projector's optical axis with the camera's Z-axis. In case of the DL.2, we saw that the lens is mounted in the centre, so we assumed the X and Y coordinate offset to be 0.0 to start with, keeping in mind to verify it later. In order to compensate for a slight misalignment between the projector's optical axis and the coordinate frame defined by the gimbal, we project a cross lying on the Z axis in a distance from the camera (we used the average of the distances to the near and far clipping planes, just to be sure it is visible). If we assume that the projector is perfectly aligned with the gimbal and furthermore the projection centre has no offset in X or Y, we can align the optical axis and hence the line of all points where the crosshairs intersect with the pan axis by tilting the projector up 90 degrees. That means that the crosshairs should rotate around their intersection point while the projector rotates around its pan axis. If the intersection point however moves on a circle around a fixed centre point it means that at least one of our assumptions is not true and the alignment is not perfect, or the projection centre is offset sideways from the rotation axis or both. Based on our assumption that the projection centre is not offset sideways, we rotated our virtual camera slightly within the gimbal's coordinate frame to move the crosshairs' intersection point to the centre of rotation. In order to verify that our assumption of no sideways offset was true, we did the same test with a projection screen at a different distance. If this experiment would have shown that the intersection point would have rotated around a different point than itself, it would have shown that correcting the angle just compensated a parallax error stemming from a sideways offset. However, since in our case it still rotated around the intersection point, we proved that our assumption of the X and Y offset being 0 was true and we corrected the small misalignment of the projector inside the gimbal. Furthermore, it means that our crosshairs can be considered to define a ray originating from the device's pivot-point/origin, which made things easier for us in section 6.2.

5 The Sound Projection System

As stated before, the projected characters shall be able to talk to human beings in the room. In order to make the projected characters more believable, the sound of their speech should originate from their mouths.

Ultrasonic directional speakers, or 'audio spotlights' (section 2.2.2) appeared as the best alternative, as they can create a sound source at the spot where the ultrasonic beam hits a surface. The fact that they are relatively small is another advantage, as it makes it easy to move and install them in any room. Ultimately, we may even be able to integrate them with the video projector into one controllable audiovisual projector.

For now however we had to build a separate controllable gimbal for the speaker, as it was not possible to attach it to the DL.2 without seriously limiting its range of movement.

5.1 Controllable Gimbal

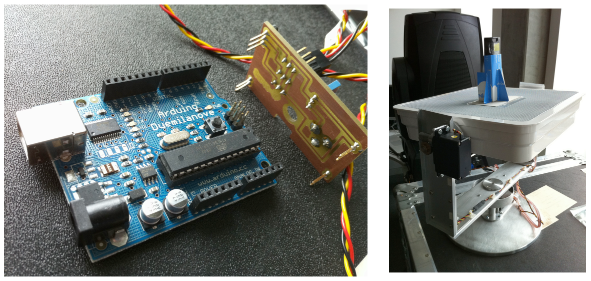
In order to be able to automatically aim the directional speaker (Sonicast S100-50 from DigiFi) at our projected character, we built a controllable gimbal. As the sound beam is very wide and fuzzy compared to a projected pixel, the required precision for the sound projection gimbal is much lower than for the video projector, so we could easily build one ourselves.

We built a gimbal as shown in figure 1(b), using RC-servos to rotate both axes. We chose winch servos, as the range of usual servos (180 degrees) was considered too small, especially for panning the device. While our winch servos were specified to have a range between two and three rotations depending on the used remote control unit, their full range turned out to be eight rotations. In order to increase the accuracy, we decided to use gears to reduce that to two full rotations.

In order to control the RC-servos, we used an Arduino Duemilanove (Atmega328 micro controller). We chose this board for its open design and development tools. It is easy to program the micro controller via the USB connection, which can also be used by the application. With libraries for serial communication as well as servo control, it was trivial to write the software that waits for the values to arrive from the serial port (via USB) and creates the corresponding pulse width modulated signal on two of the six pins available for that purpose on the Arduino board. In order to connect the servos we built a small PCB that fits on top of the Arduino board (figure 4(a)), which provides the servos with power as well as the control signal. The board allows us to supply the servos with power from the Arduino board (powered via the USB port) or an external power adaptor. However, in our tests so far the power provided by the USB port was always sufficient, as the gimbal is well balanced and turns easily.

We defined minimum and maximum values of the pulse width and the corresponding pan and tilt angles in our control application. With these values it is trivial to calculate the necessary pulse width for any valid angle by linear interpolation.

In order to help us to calibrate the sound projector's position and orientation in the room, we built a small holder for a laser pointer which we could stick onto the speaker temporarily (figure 4(b)). The calibration procedure is described in detail in section 6.2.



(a) The Arduino microcontroller board and the board we built to attach the servos.

(b) A laser pointer mounted on the controllable directional speaker to facilitate the calibration of the speaker's position in the room

Figure 4: The controller for the steerable sound projector and calibration aided by a laserpointer.

6 The Simulation Environment

The virtual characters generated by our system shall appear to be living in the same environment as us. Therefor it is important to have a digital representation of the physical environment in which the characters can be simulated. The virtual counterparts of the projection devices have to be placed in this representation at the same place as the real devices in the real environment in order to be able to perform correct projections.

6.1 Modelling of the room surfaces to project onto

Our system offers an environment editor that allows us to define this projection environment. Instead of offering primitives that can be deformed later, it allows us to define points by directly entering their coordinates as we measured them in the real environment. Then these points can be connected to form lines and in turn triangles. While this is a very basic approach, we believe that it is well suited for the task of defining the virtual surfaces that coincide with the real surfaces. As most rooms mainly contain 90 degree angles, coordinates can easily be measured with measuring tape and entered into the system, with one corner of the room being the origin. We may also investigate ways to measure the room's geometry automatically using structured light techniques as in [13] in the future. However, as this model is not only the basis for rendering the projection, but also the environment for the character simulation, some form of editing would still be required. As the characters 'live' on these surfaces, these planes define where characters may walk. The behaviour of the characters can directly be modified by purposefully making only certain areas accessible to the characters. Similarly the connections between different surfaces play an important part for route planning, as they define where characters may cross between surfaces and where not.

Figure 5 shows the triangles we defined in our test setup. In order to make navigation easier, the system allows to define planar surfaces (eg. wall, ceiling) consisting of an arbitrary number of triangles by selecting a start triangle, a point that defines the origin for this surface (in 2D surface coordinates), as well as a direction defining the X axis. The system then recursively adds all connected triangles that lie in the same plane. Ideally we would like to have only one planar coordinate system for the characters to walk in. However, as a three dimensional object can be unfolded by cutting along different edges, which may result in different unfolded planar surfaces, it did not appear to make sense to try to define a unique planar coordinate system across all surfaces. After all the character's path determines which edges need to be unfolded and which need to be cut as it walks along, crossing over unfolded edges.

6.2 Calibrating the projectors' positions in the room

As we described earlier, it is important to accurately align the virtual camera in the virtual space with the projector in the physical space. In section 4.3 we illustrated how to calibrate this alignment within the gimbal, but it is equally important to know exactly where that gimbal (its pivot point to be exact) is located in the room. As this has to be done whenever the pan and tilt device is moved, it is important to be able to do it in a way that is relatively easy and quick, yet accurate. The same is true for the steerable sound projector, even though the requirements in terms of accuracy are not as high.

We therefor select several points entered as part of the room model as calibration points. Then we aim each projector at these points, aided by crosshairs projected in the visual centre (taking into account the slight misalignment described in section 4.3) in case of the video projector or the laser pointer mounted on the directional speaker, and store the pan and tilt values.

In order to calculate the devices' positions and orientations in the room we use these pairs of calibration points and angles and try to find an optimal position and orientation from which the rays sent out under the given angles hit the corresponding points, or at least get as close to them as possible. We do this by first calculating the average of all calibration points to get a starting point, assuming that the calibration points will be chosen on different surfaces/walls of the room the projection device is mounted in and hence the projector will be somewhere within the volume delimited by those points. Then we use a loop during which we optimise the X, Y and Z coordinate, minimising the error or distance by which the rays starting from this point miss the calibration points. We optimise each coordinate by going in the same direction as long as the error decreases and switching the direction as well as cutting the step size in half when it increases until the step size gets smaller than a fraction of a millimetre. This approach based on optimisation instead of solving a set of linear equations lets us use a larger number of calibration pairs to increase accuracy and reduce the effect of measurement/modelling errors. While we do not make use of it currently, it would even allow us to find the calibration pairs with the largest errors, which could then be remeasured to possibly improve our environment model.

We calculate the error, by first calculating a rotation matrix that represents the orientation of the projection system in the room assuming it is located at the current location ($M_{orientation}$) based on the first two calibration pairs. Then we check how well the rotated direction vectors and the vectors from the current (assumed) position (the position of the pivot point) to the corresponding calibration points line up. In detail we do it as follows:

From the first two calibration points (p_1 and p_2) and the current position (pos) we calculate the vectors $v1$ and $v2$.

$$v1 = p_1 - pos \qquad \text{and} \qquad v2 = p_2 - pos \tag{3}$$

The X axis of the new coordinate system shall be defined by $v1$ brought to unit length ($\widehat{v1}$). The Z axis shall be defined by the normal of the two vectors

$$vz = v1 \times v2 \tag{4}$$

and the Y axis by the normal of these Z and X axes

$$vy = vz \times v1 \tag{5}$$

both brought to unit length. The resulting matrix M_{points}, which also contains a translation component to pos is defined as

$$M_{points} = \begin{pmatrix} \widehat{v1} & \widehat{vy} & \widehat{vz} & pos \\ 0 & 0 & 0 & 1 \end{pmatrix} \tag{6}$$

In a similar way we create a matrix $M_{directions}$ based on the first two direction vectors $d1$ and $d2$:

$$M_{directions} = \begin{pmatrix} \widehat{d1} & \widehat{wy} & \widehat{wz} & \begin{pmatrix} 0 \\ 0 \\ 0 \end{pmatrix} \\ 0 & 0 & 0 & 1 \end{pmatrix} \tag{7}$$

$$\text{with} \qquad wz = d1 \times d2 \qquad \text{and} \qquad wy = wz \times d1 \tag{8}$$

The matrix describing the projection device's orientation ($M_{orientation}$), in other words the matrix that aligns the direction vectors with the vectors defined by the assumed position and the calibration points, can now be calculated by multiplying the $M_{directions}$ with the inverse of M_{points}.

$$M_{orientation} = M_{directions} \times M_{points}^{-1} \tag{9}$$

With this matrix we can transfer all calibration points into the space of calibration directions:

$$testVec_i = M_{orientation} \cdot p_i \tag{10}$$

After unitising the resulting vectors we can calculate the angle between them and the corresponding calibration directions by calculating the scalar product of the two unitised vectors, as the scalar product of two unit vectors equals the cosine of the angle between them. As we do not need the actual angles, we calculate the error as the sum of the pairwise dot products.

$$error = \sum_{i=2}^{n} 1.0 - \widehat{testVec_i} \cdot \widehat{d_i} \tag{11}$$

As mentioned before, we try to minimise this *error* by optimising the device's position. We set our system to go 30 times through this optimisation loop, although in our experience it converged to its solution after 3 iterations. Figure 5 shows our calibration points as red dots, the video projector's direction vectors as lines in cyan and the sound projector's direction vectors as lines in magenta after the calibration has been completed.

6.3 Adjusting the focus

With the projection system's position known in the room, we can easily send a ray from there to intersect with the virtual surface the projector aims at. This way we get the distance to set the projector's focus to. As the focus values sent to the projection device do not correspond to a distance directly, we use a table of distances to focus values generated by manually focussing the projector to a screen held at different distances. For distances which are not contained in this table, we interpolate linearly between the next smaller and next larger values.

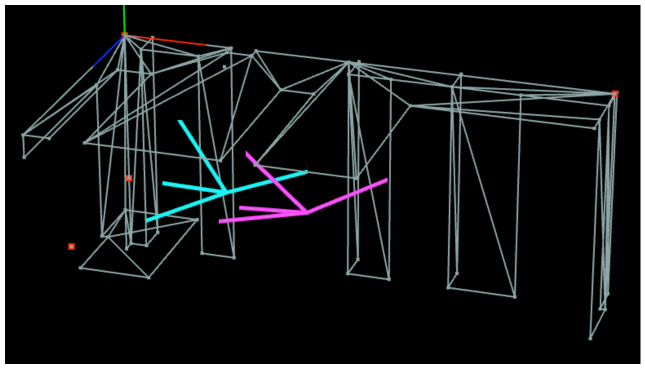
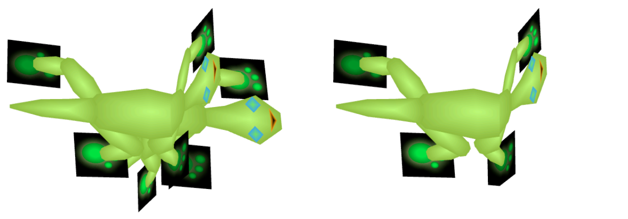


Figure 5: The Environment Editor 3D view: Surface triangles in wireframe, the video projection device is located at the intersection point of the four direction vectors in cyan, aiming at the four calibration points (red) and the sound projector is located at the intersection of the four magenta direction vectors.

6.4 Projection surfaces for Virtual Characters

Projected characters are by nature two dimensional and they walk around in their two dimensional world. This is very obvious and trivial to implement as long as they are on a single surface. However, it does get a bit more complicated when they move from one surface to another. For the simulation of the character's movement we can unfold the edge between the two surfaces to keep the simulation environment flat. But still, the character has to be rendered partially onto both surfaces. In order to support this, we define a projection area for each character, a rectangle which contains all projected parts of the character. As the character and its projection area move across the room's surfaces, the system tests if the projection area crosses any surface boundaries. If so, the system projects the character onto all surfaces it touches, taking care of all transformations to align the character's plane with the surfaces'. In that case it also uses the stencil buffer to ensure that the rendering is limited to each of the surfaces while the character is drawn in that surface's plane. Figure 6 illustrates that.



(a) Character drawn on both surface planes (b) Drawn on both planes but stencilled for each surface

Figure 6: A character crossing projection surfaces. It is rotated around the common boundary.

7 Character Simulation

As discussed in section 3, we decided against a human appearance in favour of cartoon animal style characters. Geckos, which naturally can crawl on walls and ceilings and usually appear quite flat to human observers, seemed to be a natural fit.

7.1 Path of Character Motion and Projector Control

The path the character walks on can be defined as a series of Bezier splines on the surfaces introduced in section 6.1. We calculate the length of each segment to let the character crawl along these paths with a defined speed. We furthermore check if the spline crosses any surface boundaries. If so and if there is another surface connected, we fold the spline along the edge of the two surfaces onto the next surface. We allow splines to cross several boundaries this way. We also allow spline segments to start and end on different surfaces, as long as they lie completely on valid connected surfaces.

Besides driving the crawling simulation of the virtual characters, this spline is also used to calculate the points the projector and speaker have to aim at in order to be able to project the moving character and the sound it creates. An additional benefit of having the walking path defined before the character actually walks on it is that we can anticipate where the character will be about a second ahead of time, which lets us compensate for the fact that the projector's gimbal needs about a second to aim the projector at that position. This way the projector just reaches the spot when the character reaches it as well, keeping the character well centred in the projection area. Besides the pan and tilt of the projector we also constantly adjust the focus based on the projector's distance to the point on the surface the projector is aimed at. Furthermore we aim our sound projector at the same spot.

7.2 First steps (Crawling Simulation)

In order to create the impression of the character crawling across the surface we simulate footsteps according to the motion of the character. We assume that the character always has two legs on the ground and two legs up. More specifically, the front left and back right foot are either up or down together, while the front right and back left foot together are in the opposite state. If the feet are down, they are moved in the opposite direction of the character's motion to make them stay in their fixed location. The lifted feet on the other hand are moved in the same direction as the character's motion, which moves them forward at twice the character's speed. Once any of the feet reaches a certain limit in the distance from its neutral position, the up feet are put down and the down feet are lifted up. In a similar manner we also simulate the rotation of the the feet based on the character's rotation. To visualise the state of the legs, we reduce the brightness of the lifted feet.

7.3 Inverse Kinematics

In order to create a whole character, we implemented a simple inverse kinematic simulation that can move the character's legs to connect the body with the feet. The resulting character can be seen in figure 6.

7.4 Speech output

In order to let the character speak, we included speech synthesis. At the moment we use the speech synthesiser built into Mac OS X. The speech synthesis also informs the character simulation of the phonemes the character is about to speak, which we used to animate the character's mouth. At this stage it is simply opening and closing its mouth, but we may create more detailed lip motion using more control points in the future.

8 Conclusion

The system we developed allows us to model the room as a set of surfaces which represent the physical surfaces on which the characters live. We implemented two simple versions of virtual characters, one that consists only of foot prints on the wall and one that we would characterise as cartoon versions of a gecko or an animal that can crawl on walls in general. In order to increase the believability we simulate their motion, in particular that of their feet, on the surface. This way we can make sure that a foot placed on the surface does not drift, but stays fixed in its place while the rest of the body moves. Even a character consisting of just footprints projected as crawling across the surface this way achieved an astonishing level of presence. These footprints really came alive. A coworker in the same room spontaneously pulled out her phone to film it during one of our early test runs as she was fascinated by these footprints walking across the ceiling. The character with added body and legs which were moved using inverse kinematics based on the feet's motion was also liked when demonstrated to colleagues. However we will need to refine the body's motion to make it more natural as the body consisting of head, torso and tail is stiff at the moment and only the legs move. That appears unnatural for an animal.

When a character crawls across the room, it automatically makes the projection system follow its path. This mechanism not only controls the projector's pan and tilt angles, but also constantly updates its focus depending on the projection distance. We put a lot of effort in getting a very accurate overlay of the virtual onto the real world. In particular we developed and accurate simulation of the projector's motion and calibrated diverse projector parameters. As a result we achieved a projected overlay that firmly stays in its place while the projector moves, much better than we originally thought possible without building our own controller hardware. Mechanical vibrations caused by acceleration or deceleration of the pan or tilt drives lead to slight inaccuracies of no more than a few pixels. The video accompanying this paper should provide some impressions of that. Also, if either of the pan and tilt devices needs to be moved, its position and orientation can be measured quickly by aiming them at a few known points. While four calibration points are usually enough, a larger number can be used in order to increase the accuracy.

In order to let the character talk, we used speech synthesis which also generates events we use to animate the character's lips synchronously to the speech output. We use a directional ultrasonic speaker, which can be described as a spotlight for sound, on a controllable gimbal to make the sound seem to appear from the character's mouth. This works quite well in certain situations, but there are also situations when we would like to improve it. For example if the character is near a corner of the room, the ultrasonic beam gets partially reflected from the wall at the character's position onto the adjacent wall, creating a second sound source that blurs its locality. In certain situations the beam reflected by the second wall may in turn hit the viewer directly, which makes the sound appear to originate mainly from the reflection point on that wall instead of the character's location. Also, the sound image seems to be weaker the more obliquely the ultrasound beam falls onto the surface. It certainly seems necessary to examine the reflections of the ultrasound beam and its effects in more detail. This should also include different surface materials and textures.

9 Future Work

As we are writing this paper, we are working on the integration with a dialogue system developed in our department. This, together with a speech recognition system shall allow us to communicate with the characters verbally. Once the system is complete, we will start to evaluate it more formally. We plan to do many user tests to evaluate if our our observation is true that it is beneficial to have a visual point of focus for a dialogue system. Therefor we also want to investigate what kind of character representation might be suited the best, ranging from very abstract (eg. just a spot of light) over cartoon animals to very detailed and realistic (eg. a gecko textured with an image of a real gecko). We also plan to evaluate these characters with different age groups, and in different settings (private versus public).

References

[1] M. Ashdown and Y. Sato. Steerable projector calibration. In *Proceedings of IEEE International Workshop on Projector-Camera Systems 2005*, 2005.

[2] A. J. Berkhout. A holographic approach to acoustic control. *J. Audio Eng. Soc*, 36(12):977–995, 1988.

[3] A. J. Berkhout, D. de Vries, and P. Vogel. Acoustic control by wave field synthesis. *The Journal of the Acoustical Society of America*, 93(5):2764–2778, 1993.

[4] J. Ehnes and M. Hirose. Projected reality – content delivery right onto objects of daily life. *The International Journal of Virtual Reality*, 5(3):17–23, September 2006.

[5] J. Ehnes, K. Hirota, and M. Hirose. Projected augmentation - augmented reality using rotatable video projectors. In *"ISMAR2004 The Third IEEE and ACM International Symposium on Mixed and Augmented Reality"*, pages 26–35. IEEE Computer Society, 2004.

[6] T. Johnson, G. Welch, H. Fuchs, E. la Force, and H. Towles. A distributed cooperative framework for continuous multi-projector pose estimation. In *VR '09: Proceedings of the 2009 IEEE Virtual Reality Conference*, pages 35–42, Washington, DC, USA, 2009. IEEE Computer Society.

[7] M. Kruppa. *Migrating Characters: Effective User Guidance in Instrumented Environments*. PhD thesis, Saarland Univesity, Saarbrücken, Germany, 2006.

[8] M. Kruppa, A. Krueger, C. Rocchi, O. Stock, and M. Zancanaro. Seamless personalized TV-like presentations on mobile and stationary devices in a museum. In *Proceedings of the International Conference on Hypermedia and Interactivity in Museums (ICHIM)*, Paris, France, 2003.

[9] M. Kruppa, M. Spassova, and M. Schmitz. The virtual room inhabitant - intuitive interaction with intelligent environments. In S. Zhang and R. Jarvis, editors, *Proceedings of the 18th Australian Joint Conference on Artificial Intelligence (AI05)*, Sydney, Australia, 2005.

[10] I. Mitsugami, N. Ukita, and M. Kidode. Multi-planar projection by fixed-center pan-tilt projectors. In *CVPR '05: Proceedings of the 2005 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'05) - Workshops*, page 108, Washington, DC, USA, 2005. IEEE Computer Society.

[11] C. S. Pinhanez. The everywhere displays projector: A device to create ubiquitous graphical interfaces. In *UbiComp '01: Proceedings of the 3rd international conference on Ubiquitous Computing*, pages 315–331, London, UK, 2001. Springer-Verlag.

[12] F. J. Pompei. The use of airborne ultrasonics for generating audible sound beams. *J. Audio Eng. Soc*, 47(9):726–731, 1999.

[13] P. Quirk, T. Johnson, R. Skarbez, H. Towles, F. Gyarfas, and H. Fuchs. Ransac-assisted display model reconstruction for projective display. In *VR '06: Proceedings of the IEEE conference on Virtual Reality*, page 318, Washington, DC, USA, 2006. IEEE Computer Society.

[14] R. Raskar, G. Welch, M. Cutts, A. Lake, L. Stesin, and H. Fuchs. The office of the future: A unified approach to image-based modeling and spatially immersive displays. *Computer Graphics*, 32(Annual Conference Series):179–188, 1998.

[15] B. Ullmer and H. Ishii. The metaDESK: Models and Prototypes for Tangible User Interfaces. In *UIST 97 Banff*, pages 223–232, Alberta, Canada, 1997. ACM.

[16] J. Underkoffler and H. Ishii. Illuminating light: An optical design tool with a luminous-tangible interface. In *CHI*, pages 542–549, 1998.

[17] J. Underkoffler, B. Ullmer, and H. Ishii. Emancipated pixels: real-world graphics in the luminous room. In A. Rockwood, editor, *Proceedings of the 26th annual conference on Computer graphics and interactive techniques*, pages 385–392. ACM Press/Addison-Wesley Publishing Co., 1999.

[18] P. Wellner. The digitaldesk calculator: Tangible manipulation on a desk top display. In *Proc. ACM SIGGRAPH Symposium on User Interface Software and Technology*, pages 107–115., 1991.

[19] M. Yoneyama, J.-i. Fujimoto, Y. Kawamo, and S. Sasabe. The audio spotlight: An application of nonlinear interaction of sound waves to a new type of loudspeaker design. *The Journal of the Acoustical Society of America*, 73(5):1532–1536, 1983.