The Design of an AR Treasure Hunt Game Using a Remote Controlled Car

Stephen Fitchett, Annelies Schippke, Janina Voigt
Department of Computer Science
University of Canterbury
Christchurch, New Zealand
{saf75, asc120, jvo24}@student.canterbury.ac.nz

ABSTRACT

Augmented reality provides new opportunities for game design, where novelty often creates a much richer game experience. We introduce a new remote control car treasure hunt game, which makes extensive use of augmented reality for both input and output. Players use a steering wheel, which is tracked by a camera, to control a remote control car. Using a second camera attached to the car they can see a view as if they were physically driving the car. Markers in the game area are used to overlay hazards for the player to avoid and treasures for the player to collect. This combination of technologies demonstrates the unique and exciting possibilities for augmented reality games.

1 Introduction

Augmented Reality (AR) has been extensively used for the development of realistic and interactive computer games over the last few years. These games often require the interaction of a variety of technology, such as Head Mounted Displays (HDM), Global Positioning Systems (GPS), cameras, laptops or remote controls. Different combinations of these technologies often result in innovative games and insights into the future of augmented reality.

This project involved the development of an AR game using a remote control car with a wireless camera. The design culminated in a treasure hunt game in which players attempt to find treasures to gain points before time runs out. Virtual obstacles impede the car and can cause loss of control. A tangible user interface allows the player to control the car with a cardboard steering wheel, which is tracked by a second camera. Players view the game area from the perspective of a driver inside the car. The software was developed using the OSGART library to reduce development time.

This report begins by discussing related research in section 2, including augmented reality games and windscreen augmentations in cars. It then describes the design and implementation of our game in section 3, including a description of the challenges and limitations of our implementation. Section 4 discusses the lessons learned as part of the project, and section 5 details our conclusions and thoughts about future work.

2 Related Work

While there has been limited research into augmented reality remote control car games, there has been abundant augmented reality research into games and driving interfaces. This research provides insights into the design of an AR remote control game, including both interface design and technological limitations.

Section 2.1 discusses three augmented reality games and provides general insights about AR game design. Section 2.2 describes two AR rac-

(a) Sky Invaders 3D

(b) AR-bowling

(c) Human Pacman

Figure 1: Previously developed augmented reality games

ing games and provides more specific insights into AR remote control car game design. Finally, section 2.3 describes augmented reality driving augmentations, which aid the design of a realistic distraction free game interface.

2.1 AR Games

We investigated three AR games, shown in Figure 1: Sky Invaders 3D (Avery, Piekarski, Warren & Thomas 2006), AR-bowling (Matysczok, Radkowski & Berssenbruegge 2004) and Human Pacman (Cheok, Goh, Liu, Farbiz, Fong, Teo, Li & Yang 2004). These provided insights about indoor and outdoor AR games, user enjoyment, tracking and realism of AR games.

Sky Invaders 3D was developed with the aim of comparing the user enjoyment of an outdoor AR game with a traditional desktop PC game. The goal of the game is to protect the earth by shooting down waves of UFOs. The outdoor version uses optical see-through HMDs to combine reality and computer generated images, whereas the traditional PC version uses a keyboard to control the game. The paper discusses the differing requirements for indoor and outdoor games. Although we decided to develop an application that can only be played inside, it might be possible to later enhance it for outdoor gaming (see section 6 Conclusions and future work). Additionally, the paper also discussed the importance of user enjoyment in creating a successful game, which aided design decisions we made for our game.

AR-bowling aims to provide a realistic augmented reality bowling game experience. It uses a see-through HMD and an optical tracking method using markers. Player interaction is done by hand gestures using specially designed gloves. The game is designed to be playable anywhere and to provide an immersive gaming experience. To achieve this, the player is involved with as many senses as possible and the game aims for realism.

Human Pacman is an augmented reality version of the original Pac-Man game and is played in the real world. In this game, humans control both ghosts and the Pacman character by physically moving around a predefined game area. The game area is a combination of the real, physical world and virtual elements such as pac-dots used to enhance the gaming experience. Players wear a backpack containing a wearable computer and see the world through a head-mounted display.

2.2 AR Racing games

Two previous papers have discussed AR racing games, which provide more specific insights in the design of our game than general AR games. These are shown in Figure 2.

Scorpiodrome (Metaxas, Metin, Schneider, Shapiro, Zhou & Markopoulos 2005) is a mixed reality game involving modified remote-controlled cars (Scorpios). The aim of the game is to drive the Scorpio as fast as possible past checkpoints while collecting virtual diamonds

(a) SCORPIODROME

(b) Oda et al.'s racing game

Figure 2: Augmented reality racing games

and making life harder for the other players. During the game, random events such as plasma storms can occur which hinder the progress of the game. The cars are tracked using motion sensors and the players look at a game board onto which virtual elements are projected by an over-head projector.

This paper introduces the idea of using checkpoints instead of a fixed track the car has to follow. Interestingly, the authors chose to do this because they found that it was too difficult for children to keep a car on the track. In Scorpiodrome, virtual game elements interact with the remote controlled cars, for example causing the cars to spin or drift. This idea later inspired aspects of the design of our game, in particular the interaction between obstacles and cars.

Oda et al. (2007) describe the development of an AR racing game which adds a virtual car to the scene. The scene is captured using several cameras. The game board is made up of fiducial markers on which virtual objects are placed that act as obstacles for the race car. The car driver views the game area through a head-mounted display. His task is to quickly drive the car along a path of specified waypoints while avoiding virtual obstacles. In this paper, the virtual car is controlled through a tangible user interface, in the form of bicycle handle bars.

2.3 AR Driving

We examined three papers that describe windshield augmentations in cars, shown in Figure 3. A key design consideration for these interfaces is that they do not distract the driver from their main driving task. These papers therefore provide insights into interface design for an AR remote control car game which allows players to concentrate on navigating the car.

Kim and Dey (2009) describe a 2.5 dimensional augmented reality system for car windscreens. An outline of the current road appears in the foreground, which blends into a two dimensional map above it which shows streets and other data such as crash sites. This interface is compared to a traditional in-car GPS navigation system using both elderly and younger drivers in a driving simulator. The results show that the augmented reality system decreases cognitive load, reduces divided attention related issues and reduce navigation errors, particularly for the elderly drivers.

Tonnis et al. (2007) present an augmented reality windscreen display which shows a braking bar drawn on the road indicating the car's stopping distance. A second version also shows the car's drive path using curved lines that extend to the braking bar. The system is evaluated in a driving simulator. Results show that partic-

(a) 2.5D in-vehicle navigation system

(b) Braking bars

(c) 3D arrow alerting to imminent dangers

Figure 3: Augmented reality car windshield augmentations

ipants drove faster with the system than without it, although there were more speed oscillations when using it. Additionally, the systems resulted in drivers driving closer to the centre of their lanes. The version displaying only the braking bar and not the driving path performed better on most, but not all, measures.

Tonnis et al. (2005) compare two augmented reality windscreen display systems for alerting car drivers to dangerous situations near the car. The first is a bird's eye exocentric view which draws a small car on the windscreen and draws a red octagon relative to the car to indicate the point of danger. The second is an egocentric view which contains a 3D arrow pointing directly to the point of danger. An evaluation found that the exocentric method unexpectedly performed better for locating the point of danger in terms of both task time and errors. The authors hypothesise that the egocentric view performed badly because the arrow was simulated as being three meters in front of the driver and drivers may have interpreted it as being where they were instead. The egocentric view did, however, result in significantly smaller lane deviations while locating points of danger.

3 Implementation

We implemented a proof-of-concept treasure hunt game to demonstrate the capabilities of an augmented reality remote control car, using ideas from previous research and considering the limitations of the hardware. The aim of the game is to score as many points as possible in a set time limit. This section explains the game in general, and then describes the hardware and software setup. The remainder of the section highlights the limitations of our implementation and the challenges we faced over the course of the project.

3.1 The game

The treasure hunt game includes a number of components and unique features, explained below.

3.1.1 Markers

Before beginning the game, players can set up the gaming area by placing markers on the ground. There are two types of markers: static and dynamic markers. The static markers are used to enhance the background setting for the game and have 3D models such as houses and trees placed on them. They have relatively little influence on the game and serve to visually enhance the gaming area. No points are deducted if a player crashes into them, although when this happens the remote control car is stopped for two seconds as a penalty.

Dynamic markers, however, contain obstacles and treasures during the game. Treasures give

the player points and can be collected by getting close enough to the marker with the car. Obstacles on the other hand will result in points being deducted if the car gets too close to them. In addition, obstacles may affect the car in other ways, for example making it uncontrollable for several seconds, as if it was on ice, or causing it to start turning left. Dynamic markers are numbered to make it easy to distinguish them from static markers when the game is setup.

Models displayed on static markers are permanently fixed. Models displayed on dynamic markers are fixed when not interacting with them, but once a player moves onto one of these markers, the model disappears and the corresponding action is taken (such as awarding points). To prevent depletion of objects over the course of the game, objects are placed on empty dynamic markers after a random period of time. A player can easily tell when the car has gotten close to a marker because a sound will be played in addition to model-specific actions.

Table 1 summarises information about the different markers and models.

3.1.2 Game setup

When setting up the gaming area, the player has total freedom in placing the markers, allowing a slightly different game setup every time the game is played. Figure 4 shows one possible configuration. The markers could even be moved during the game by other players, for example to confuse the person driving the car. We believe that giving the players the opportunity to set up the gaming area before starting the game is very valuable and will enhance their overall experience. When developing an AR game for children, Metaxas et al. found that the players really enjoyed the setup process as well as the game itself (Metaxas et al. 2005) and we believe that this will also be the case for our game.

Apart from the arrangement of markers, the size and shape of the gaming area is also up to the player. Theoretically, the only limitation in the size of the gaming area is the reach of the remote that controls the car. This remote is connected to the computer running our AR application. Due to the limited range of the remote of around a dozen meters, the gaming area size is naturally limited. In addition, the gaming area needs to be close to or around the computer running our application because the remote control is connected to this computer.

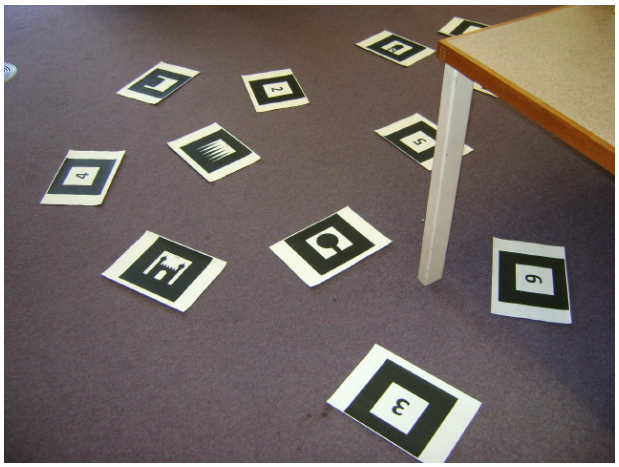


Figure 4: The game setup is controlled by the player

3.1.3 Controlling the game

Once the gaming area has been set up, the game can start. During the game, the player can control the car in two ways: he can either use the keyboard or a steering wheel with a marker on it. Using the keyboard, the car can be controlled using the arrow keys to accelerate, reverse and turn left and right.

When we first started the project, we developed the functionality to control the car using the keyboard but later decided that a tangible user interface, in the form of a steering wheel, would be a more novel approach. Figure 5 shows the interface for this. Our inspiration for the tangible user interface came from the AR racing game developed by Oda et al. (2007). In their game, a virtual car is controlled using a set of bike handle bars with fiducial markers on them. While we liked the idea of having a tangible user interface to control the car, we did not like

Marker	Model	Type	Functionality	Sound	Model Source
1-6	Crown	Dynamic (random)	Finding this treasure adds 100 points to the player's points total.	Coins / Money	http://artist-3d.com
	Ring		Finding this treasure adds 50 points to the player's points total.	Coins / Money	http://www.amazing3d.com
	Fastener		Causes the car to turn left for one second and deducts 5 points.	Car spinning out of control	http://artist-3d.com
	Cone		Stops the driver from being able to control the car for one second and deducts 20 points.	Car spinning out of control	http://artist-3d.com
	Grenade		Causes the car to turn left for 3 seconds and deducts 25 points.	Explosion	http://artist-3d.com
	Man		Stops the driver from being able to control the car for 3 seconds and deducts 10 points.	Scream	http://archive3d.net
Castle Hedge House Tree Grass Tower	Castle Hedge House Tree Grass Tower	Static	Stops the car for 2 seconds if it gets too close to the marker.	Honking	http://www.amazing3d.com http://www.3dmodelfree.com http://artist-3d.com http://www.amazing3d.com http://artist-3d.com http://www.3dm3.com

Table 1: Markers and models

the fact that their interface was based on biking rather than driving a car. We therefore developed a steering wheel user interface, showing in Figure 5, which we felt was more intuitive because it was taken straight from the car domain.

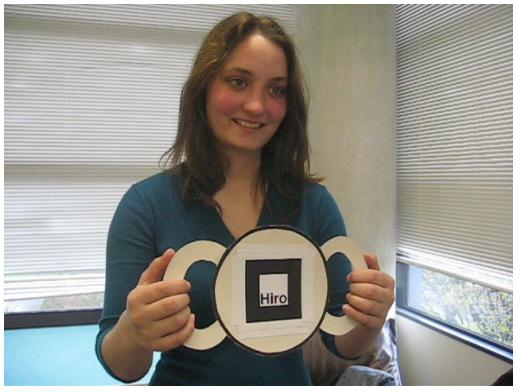


Figure 5: A player using the steering wheel

The steering wheel we created is made up of cardboard and has a single fiducial marker on the side facing away from the player. When using this interface to control the car, the marker has to point in the direction of a camera that is positioned above the monitor which displays the scene.

The speed of the car is controlled by the pitch angle of the steering wheel (and by extension the marker). Tilting the wheel so that the top edge moves away from the player causes the car to accelerate, while tilting it in the other direction causes it to decelerate. Bringing the steering wheel into a vertical position, or up to 20 degrees in either direction in the pitch axis, causes the car to slow down and eventually stop. A tilt angle between 20 and 60 degrees in either direction is linearly mapped to a maximum velocity. Angles exceeding 60 degrees have no additional effect. An illustration of these movements is shown in Figure 6.

Turning the steering wheel to the left or right (in other words, anticlockwise and clockwise rotations in the roll axis) causes the car to turn left

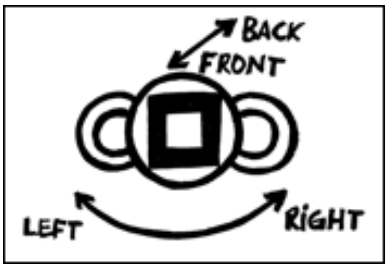


Figure 6: Steering wheel movements

or right respectively. These are binary values: the car is either turning or it is not; there is no control over how quickly it is turning. A combination of these possibilities (e.g. forward and right rotation) provides the player with a natural and realistic driving experience.

In addition to controlling the car, the steering wheel can be used to start, stop and pause the game. A new game will start as soon as the steering wheel is visible to the camera above the monitor and will pause if the steering wheel is subsequently removed from the camera's field of view. This means that the game will only run while the car is being controlled by the player.

The game stops automatically when the game is over and can be restarted by simply bringing the steering wheel back into view of the camera. A game can also be stopped at any time if the player holds up a stop paddle we developed. Every time a game is paused or stopped, a message is displayed on the screen so that the player always knows when a game is actually running.

When playing the game, the video from the wireless camera on the remote controlled car is displayed on a monitor. This means that the user is able to play the game from the perspective of someone driving the car. This is likely to make controlling the game easier, since Oda et al. found in their game that users struggled to control the car from an external viewpoint (2007).

In addition to the video feed from the wireless camera, there is a small window displaying a video feed from the camera installed above the

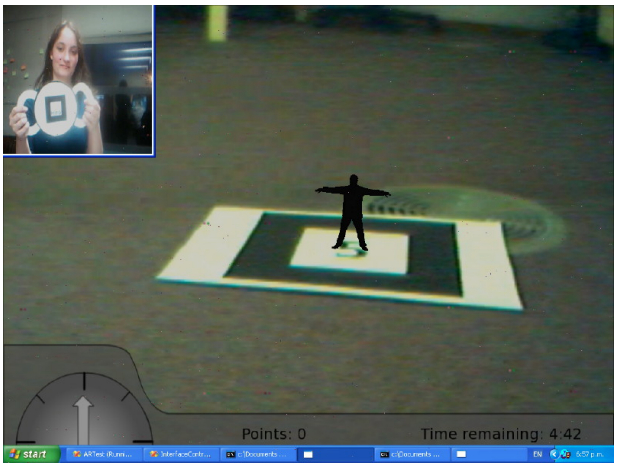


Figure 7: The game interface

monitor. This camera is used to control the car using the steering wheel user interface. This window makes it easier for a user to see when the steering wheel is in the view of the camera and when it is not. Otherwise, it can be hard to know where exactly to hold the steering wheel when playing the game.

The display shown to the player during the game also includes a dashboard at the bottom of the screen, displaying the current speed of the car, the number of points collected so far, and the time remaining until the end of the game. The game interface can be seen in Figure 7.

3.2 Hardware setup

Our system consists of the following hardware, illustrated in Figure 8:

- The basic component for the system is a computer (8), which is required to have a parallel port and anything else required to run OSGART. This computer is connected to all other hardware components and runs both of the applications required for the AR car game (see Section 4.3).
- For the basic input and output, the computer is connected to a monitor (2), keyboard (11) and mouse (3).
- To support the tangible user interface for our system, consisting of the steering wheel (4) and the stop paddle (9), the computer is also

connected to a webcam positioned on top of the monitor (1).

- In order to be able to control the car (7), the computer is connected to a remote control (10) via its parallel port.
- To get the video from the wireless camera on top of the car (5), the computer is connected to a receiver (12). The wireless camera sends the image to this receiver, which is then forwarded to the computer via a USB port connection. The wireless camera enables the detection of markers which lie on the floor (6).

3.3 Software setup

Our system is composed of two applications (see Figure 9): *Interface Controller*, which processes input from the user, and *Car Controller*, which communicates with the car.

The two applications are both written in C++ and use the OSGART library to implement the AR features of our game, including marker tracking. We developed two separate applications rather than a single one because OSGART allows the use of only a single camera per application. However, to be able to control the car using the tangible user interface we developed, a second camera is required in addition to the wireless camera on top of the car. We therefore split our game into two separate applications, each using one of the two cameras. The interface controller application connects to the webcam on top of the monitor to get input from the tangible user interface, including the steering wheel and the stop panel. It detects the marker on the steering wheel and calculates the rotation of that marker relative to the camera. OSGART returns this marker rotation in the form of a quaternion which is then converted by the application into Euler angles. Euler angles provide the rotation of the marker around the x, y and z axes. These rotations are then used by the application to decide if the car should be accelerate, reverse or turn left or right.

In addition to processing input from the tangible user interface, the interface controller is also connected to the keyboard and listens to events from the arrow keys that can also be used to control the car.

The interface controller sends information to the car controller application via pipes. The car controller creates a pipe and listens for data, while the interface controller connects to the pipe to send messages. Supported messages are movement direction (represented as a bit string), pitch angle (used by the car controller application to calculate speed) and special messages with assigned meanings to start, pause or stop the game.

The car controller is connected to the wireless camera on top of the car and thus receives the video feed showing the car's perspective. This video feed is received via the receiver, which is connected to the computer's USB port. Since this video shows the world from the viewpoint of the car, this video may include the markers that were placed in the gaming area, including static and dynamic markers.

When the car controller receives the video from the wireless camera, it places models on any markers that are detected before displaying the video to the user on the monitor. It also calculates the distance between markers and the car to see if the car has gotten close to an obstacle or treasure.

In addition to handling marker detection, the car controller also communicates with the car via the remote control that is connected to the parallel port of the computer. It tells the car to accelerate, reverse and turn left or right by activating and deactivating pins on the parallel port. This information is then sent to the car by the remote control.

One difficulty related to controlling the car is that we wanted to allow for a variable car speed rather than just a binary input. To achieve this, we send oscillating signals to the car at a high frequency to simulate different speeds. At the top speed, $s(max)$, the car is sent a constant 'accelerate' or 'decelerate' signal. For a speed s less than $s(max)$, the car is sent an 'accelerate'

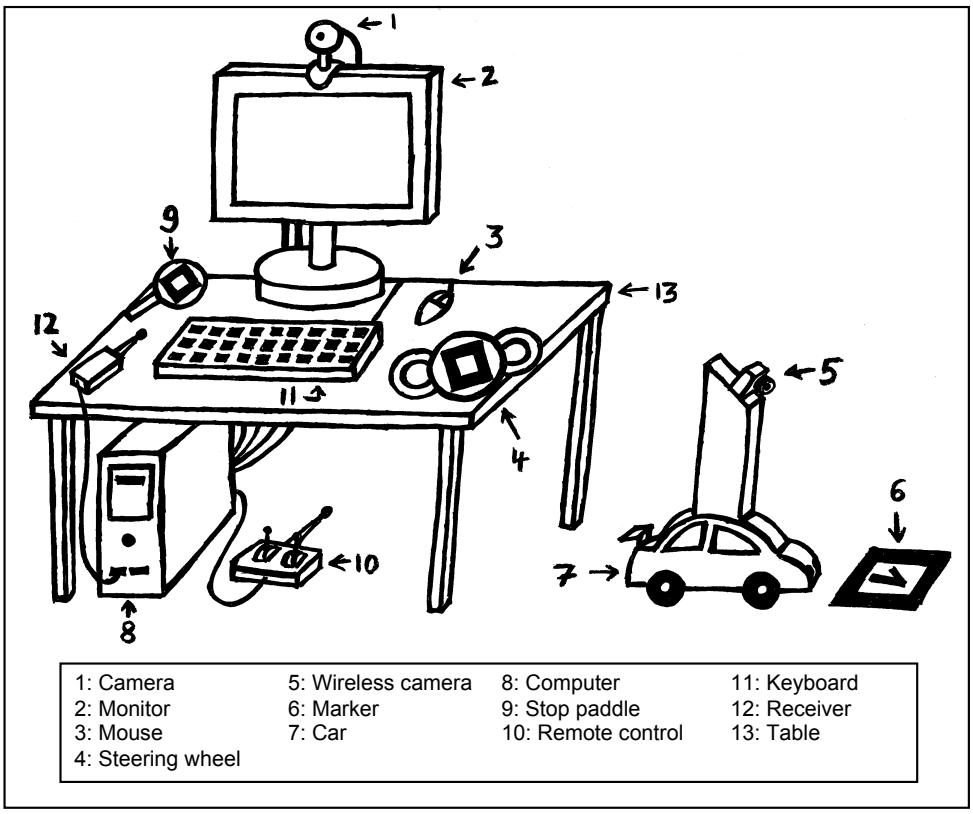


Figure 8: Hardware setup

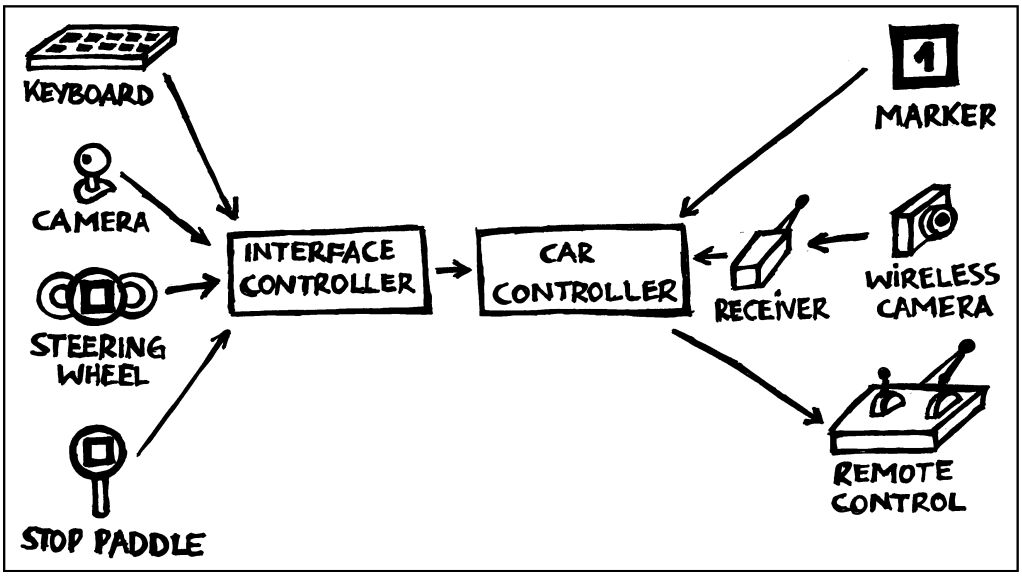


Figure 9: Software setup

or 'decelerate' signal for $\frac{s}{s(max)}\%$ of the time and a 'stop' signal for the remaining time. The signal oscillates between these two values at a rate of 10 Hz, which allows for a relatively consistent motion with limited jerkiness.

Finally, the car controller is responsible for keeping track of information about the current game, including how much time remains until the game finishes and the number of points the player has collected. The application displays this information along with the current speed of the car on the dashboard, which is overlaid on the video feed from the wireless camera.

The dashboard is produced through a combination of TGA images and text. The images are loaded using Open Scene Graph's TGA loading plugin. They are slightly transparent so that the video feed from the camera can be seen through the dashboard.

The speed of the car is displayed on the dashboard using a speed dial constructed from two images: the background of the dial and the arrow. Whenever the speed of the car changes, the speed dial is updated by rotating the arrow image to point to the correct position on the dial.

Both of the applications are multi-threaded to improve the performance of our software. In each application, one thread handles the interprocess communication. In the car controller application, there are several threads for updating the dashboard. One of these threads keeps track of the remaining time, updating that part of the dashboard every second. The second thread continually checks the current speed of the car and updates the speed dial on the dashboard if necessary.

3.4 Challenges and solutions

Wireless camera angle: Because we required the ability for the remote control car to drive over markers, they needed to be flat on the ground. Unfortunately, this resulted in them being almost perpendicular to the forwards facing wireless camera mounted on the car, which prevented them from being detected. To solve this problem, we constructed a styrofoam mount on top of the car and aimed the camera downwards slightly, as shown in Figure 10. This improved the perspective for viewing markers while the camera height allows the player to still see a reasonable distance away. The mount was designed to be light by constructing it with styrofoam and giving it a hollow centre, so that it does not significantly affect the car's stability.

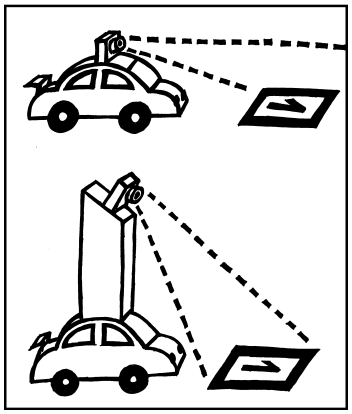
Multiple cameras: A limitation of OSGART meant that we couldn't use multiple cameras within a single application. As explained earlier in the report, we solved this issue by developing separate applications for each camera and having them communicate with each other using pipes.

Latency and frame rate: Low frame rates and camera latency made it difficult to navigate the car by viewing the image on the monitor. This problem can be partially alleviated by reducing the car's speed. This issue is described in more detail in the Limitations section below.

3.5 Limitations

While the AR car game is fully functional, several limitations restrict it to being a proof-of-concept game demonstrating the interaction of a number of technologies. While a number of these limitations are related to the hardware we used, some are more general and harder to fix. The limitations we discovered are listed below.

Lighting: Accurate marker detection is dependent on a suitable level of ambient light. Too little light results in a dark image in which markers cannot be recognised. Too much light or high levels of contrast can result in a bright image or an image with excessive glare that also prevent recognition. The game environment must be carefully set up to ensure that there are suitable levels of light.



(a) Comparison of camera positions

(b) Original position of the wireless camera

(c) Car with styrofoam mount on top

Figure 10: Camera angle has a big effect on marker tracking

Wireless camera: The frame rate of the wireless camera we used is quite low and there is some latency before the image is displayed on the monitor. This makes it difficult to navigate as there is a delay before an object appears on the screen, which reduces the time available to manoeuvre around the object. The time delay between user inputs and the player seeing the car react to their inputs also makes it slightly more difficult to drive the vehicle accurately.

Another problem with our wireless camera was its low resolution combined with noise added to the picture as the receiver receives it. Most significantly, this makes marker detection less reliable, although it also has an effect on driving precision. Bigger markers can be used to partially overcome this problem, but this increases the density of markers in the game area and may require a bigger area as a result. Using a higher resolution camera in the future would help to alleviate this problem.

Wireless camera angle: While a camera mount greatly reduced our problems with marker detection caused by a poor camera angle, detection is still unreliable, especially if the car is travelling quickly. Raising the camera even more is problematic as it reduces the car's stability and the makes the camera's perspective less like that of a driver sitting in the car. A future solution may be to use an additional camera positioned at a height to detect markers as well as the car's position and use the car's camera primarily to show the user an image of a car driver's perspective. This could, however, reduce tracking accuracy

Parallel port: The remote control uses a parallel port to receive signals from the software. Parallel ports have largely been superseded by USB and are no longer commonly found on computers. This limits the distribution of the game.

Remote control range: The range of the remote control is limited to about 12 meters. As a result, the car can easily get out of range. This restricts the location and size of the gaming area.

Multiplayer capabilities: As we only had one complete set of remote control car hardware and a single parallel port we could only develop a single player system. Future work could investigate a multiplayer system using two remote control cars where players compete and their cars can influence each other. This may be feasible using a single computer

for input, or otherwise using a set of networked computers.

4 Lessons learned

Over the course of the project, we have learned several lessons about developing AR applications.

At the start of the project, we spent a lot of time getting the hardware and software we required set up. In particular, we struggled with getting OSGART to work, encountering a number of different errors and exception when trying to run even very basic OSGART programs. We also had some issues towards the end of the project with the wireless camera which stopped working temporarily for no obvious reason. We managed to get it working again after playing around with the camera configuration. All this has taught us that setting up the hardware and software for an AR project is not trivial.

However, once we had set up the hardware and software we needed, using OSGART to create AR applications was surprisingly easy and quick, indicating its usefulness as an AR library.

Our project, and ultimately the success of our game, depends on the reliability of our application, including the reliability of the marker tracking. We have found that this is a significant issue, as marker detection can be quite unreliable at times, in particular when lighting is bad. In addition, the resolution of the wireless camera used in this project is relatively low, compounding the problem of marker detection and tracking.

Apart from complicating marker detection, the low resolution of the wireless camera also becomes apparent when playing the game since we directly display the video feed from that camera to the user. This results in a relatively low resolution image being displayed to the user. In addition to the camera resolution, we have also found that the camera's frame rate is a problem. Because the user is looking at the video feed from the wireless camera while steering the car, a low frame rate makes it harder to avoid obstacles, thus making it harder to steer the car. This has clearly shown us that the choice of hardware and technology is central to the development of AR applications.

5 Conclusions and future work

This report described a novel augmented reality game that makes use of a remote control car and provides a tangible user interface. The combination of equipment and concepts that the system is composed of provide a rich gaming experience.

At this stage of development, the game remains at a proof-of-concept level. There is a large amount of future work that could be conducted to improve the reliability and flexibility of the game, as well as the enjoyment of the game in general.

The use of better quality hardware in the future would improve the quality and reliability of the game. Higher quality or additional cameras could improve marker detection, reduce issues with marker detection and allow for smaller, less obtrusive markers. A remote control car that specifically allows for variable speeds could reduce jerkiness at lower speeds and allow for more precise speed control. A remote control with a larger range and use of a laptop could reduce the limitations related to proximity and size of the gaming area. Finally, additional sets of hardware could enable a multiplayer version of the game.

A number of features and refinements could be added to the game. A number of common features present in other games, such as high scores, have not been implemented due to time constraints and because they are of reduced importance in a proof-of-concept system. More realistic markers and events could be implemented, for example to realistically simulate slipping or losing control on an ice or oil marker. More diversity in marker objects and a larger number of random object locations could also make for more interesting game play.

A version of the game could also be developed using a head mounted display instead of using a computer monitor. Motion sensors could be attached to the player to control the car's movements rather than using a steering wheel. These changes would remove the confinement that the player remains in a particular spot while playing the game and would provide a more immersive experience.

The current game assumes that it will be played indoors. An outdoor version would be more difficult to implement and would have much less predictable conditions, as lighting conditions, weather and power supply need to be considered. Future work could examine these issues.

Despite these limitations, the existing game still provides an enjoyable game experience when set up in the right conditions, and provides valuable insights as a proof-of-concept system.

REFERENCES

Avery, B., Piekarski, W., Warren, J. & Thomas, B. H. (2006), Evaluation of user satisfaction and learnability for outdoor augmented reality gaming, *in* 'AUIC '06: Proceedings of the 7th Australasian User interface conference', Australian Computer Society, Inc., Darlinghurst, Australia, Australia, pp. 17–24.

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

Kim, S. & Dey, A. K. (2009), Simulated augmented reality windshield display as a cognitive mapping aid for elder driver navigation, *in* 'CHI '09: Proceedings of the 27th international conference on Human factors in computing systems', ACM, New York, NY, USA, pp. 133–142.

Matysczok, C., Radkowski, R. & Berssenbruegge, J. (2004), Ar-bowling: immersive and realistic game play in real environments using augmented reality, *in* 'ACE '04: Proceedings of the 2004 ACM SIGCHI International Conference on Advances in computer entertainment technology', ACM, New York, NY, USA, pp. 269–276.

Metaxas, G., Metin, B., Schneider, J., Shapiro, G., Zhou, W. & Markopoulos, P. (2005), Scorpiodrome: an exploration in mixed reality social gaming for children, *in* 'ACE '05: Proceedings of the 2005 ACM SIGCHI International Conference on Advances in computer entertainment technology', ACM, New York, NY, USA, pp. 229–232.

Oda, O., Lister, L. J., White, S. & Feiner, S. (2007), Developing an augmented reality racing game, *in* 'INTETAIN '08: Proceedings of the 2nd international conference on INtelligent TEchnologies for interactive enterTAINment', ICST (Institute for Computer Sciences, Social-Informatics and Telecommunications Engineering), ICST, Brussels, Belgium, Belgium, pp. 1–8.

Tonnis, M., Lange, C. & Klinker, G. (2007), Visual longitudinal and lateral driving assistance in the head-up display of cars, *in* 'ISMAR '07: Proceedings of the 2007 6th IEEE and ACM International Symposium on Mixed and Augmented Reality', IEEE Computer Society, Washington, DC, USA, pp. 1–4.

Tonnis, M., Sandor, C., Lange, C. & Bubb, H. (2005), Experimental evaluation of an augmented reality visualization for directing a car driver's attention, *in* 'ISMAR '05: Proceedings of the 4th IEEE/ACM International Symposium on Mixed and Augmented Reality', IEEE Computer Society, Washington, DC, USA, pp. 56–59.