The Booze Cruise: Impaired Driving in Virtual Spaces

J. R. Parker, N. Sorenson, N. Esmaeili, R. Sicre
Digital Media Laboratory
Faculty of Fine Arts
University of Calgary
1 403 220 6784
parker@cpsc.ucalgary.ca

Lori Shyba, John Heerema, Philippa Sessini
Interdisciplinary Studies Program
University of Calgary
2500 University Dr. N.W.
Calgary, Alberta, Canada T2N 1N4
1 403 220 6784

ABSTRACT

The first course in Canada on Serious Game development was run at the University of Calgary as a joint Fine Arts and Computer Science course. The assessment in the course was based on a project, and the game project used was a simulation of impaired driving called the 'Booze Cruise'. The game was designed in conjunction with the alcohol unit from the Calgary Police Service, and was intended to show how difficult it would be to drive while impaired. A huge media response occurred when the game was announced in October of 2007, including print, radio, and television both nationally and internationally. This is a summary of the design process for that game, relates the stages of development, and summarizes the response by the media, the University, and by NGOs.

Keywords

Serious games, education, interactive media, computer games.

INTRODUCTION

In January of 2007, the first course in Canada on Serious Game Development was offered at the University of Calgary (Sawyer, 2002; Michael and Chen, 2006). It was Computer Science 701.03, officially a high level graduate course, but having participation from arts and interdisciplinary students at senior undergraduate levels. The course evaluation was based on a project, and the game project selected was called the 'Booze Cruise', a simulation of impaired driving. The group consulted with the Calgary Police Service before embarking on the simulation; the Alcohol Unit was keen to help, and game a summary of what sort of accidents are related to alcohol consumption and why those types.

A huge media response occurred when the game was announced in October of 2007, and the game won the best student game category at the Futureplay International Game Competition in that year. This article is a summary of the design process for that game, relates the stages of development, and summarizes the response by the media, the University, and by NGOs.

Some of this response is still taking place. Impaired driving is a serious problem in Canada and is universally seen as having a social stigma connected with it. It is a preventable form of death and injury that has a high profile at the present time. NGOs and police services are always searching for better ways to reach the public with the message that drinking and driving is dangerous and socially irresponsible.

This article is a description of the Booze Cruise project from inception to Discovery Channel interview.

BOOZE CRUISE DESIGN

When the course began there was no predefined project associated with it. The original idea for the Booze Cruise game can as a result of scrums held during the first two weeks of the semester held with the entire team, 4 computer scientists and 2 artists, attending. After proposing and discarding a handful of ideas, the team became focussed on a game about drunk driving for three main reasons: many in the group had some experience with an accident or injury caused by an impaired driver; impaired driving is a high profile problem being addressed by many police groups and NGOs; and finally, a software base was available because the recent book by Parker (2005).

Based on the scrums, a high concept design document was generated outlining the game play, assets, and interface. Then, while three of the team started modifying the *Start Your Engines* (Parker, 2005) code, the remainder of the group made arrangements to meet with the Calgary Police Service (*Alcohol Unit*). The meeting was conceived as a way to keep the game grounded in reality. If there was a group who would understand the effect of alcohol on a driver, surely this would be it.

The overview of the game as described in the high concept is simple: the player wakes up in the trunk of their car after a lot of drinking, and tries to drive the car home. The back story concerning how the player ended up in the trunk is not essential to the game description, but the fact that the player is still legally impaired is essential. The player's drug alcohol is simulated at 0.25, rough three times the legal limit. The player has 90 seconds to get home, and there are pedestrians, traffic, police cars (a check stop) and animals on the road. The chance of even getting home in time is small, based on observations of game play over a six month period.

The Alcohol Unit - Real-World Design Considerations

The police seemed initially amused and perhaps unconvinced at the prospect of a video game based on impaired driving. They were nonetheless very helpful, and provided information essential to the technical game design. In particular, they gave three fundamental effects of alcohol on a driver that could be simulated in a game.

First, the police reported that human reaction time is lower after absorbing alcohol. This effect is predicable, but the detail was essential - people lose 0.15 seconds of reaction time for each standard drink consumed. Based on this information, it was decided to add a delay in the interface. This time delay occurs between the time that a player types a key to control the vehicle and the time that the command is implemented. This is the most obvious game play feature, according to all of the players that have been observed.

The second feature of the game inspired by the alcohol unit interview was a visual halo surrounding bright lights. This was easy to implement, but in order to enhance the effect it was decided that the game would take place at night. This is also a time when impaired drivers are most likely to be on the road.

Finally, the police noted that vision tends to blur as a function of blood alcohol content. This effect is most pronounced in moving objects, especially when the driver moves their head. This turned out to be the most difficult aspect of impaired driving to simulate, and required the use of the pixel shader in OpenGL (Schreiner, 2003) that allows developers to program the graphics card.

Artistic Design

The high concept document specifies generalities about gameplay, but leaves the 'look and feel' of the game for the next stage. While consultation with the police service was taking place, the artistic issues surrounding the game were also being addressed back in the lab.

The game was to be a drive home, in 3D, and was to take place in an urban environment to give a richer visual presentation and to permit a greater number of possible obstacles for the driver. A map was drawn showing the route to be driven and the surrounding spaces, but the group felt that it was difficult to visualize the environment properly in a single 2D rendering. So it was that a physical model of the play area was constructed (*Figure 1*).

Figure 1: The tangible model of the Booze Cruise first level. The designer, Nooshin Esmaeili, is included for scale.

Final Design Nuance

A player can learn how to deal with a fixed time delay in a manner similar to the way players learn how to deal with lag due to latency in network games. In order to make the delay more difficult to handle, we choose to not have it constant. Randomness was added to the delay in the steering and acceleration. It is easy to imagine that depending on the quantity of alcohol absorbed, the driver is less precise in his movement, and that this random effect is therefore a simulation.

Music

The development team had connections in the music industry, and one group in particular was interested in helping with the project. The Edmonton band *Gloom Room* responded positively to having their music used in our game. This was important so as to avoid licensing fees that would forbid the game from being distributed freely on the Internet.

The band has a web presence on myspace:

http://www.myspace.com/gloomroom

It was decided to use OpenAL (Loki, 2000) as the audio display mechanism. Unlike alternatives such as DirectX, (Boer, 2005) OpenAL is not associated with a particular platform, and this should result in a more portable system. An OpenAL install script will be distributed with the game for those players who don't have it installed already, likely the vast majority.

IMPLEMENTATION - SIMPLICITY IS KEY

The software for the game implementation was based on an existing system written for a textbook on game programming [Park05]. The book gives programs for moving a simple object about a 3D surface under user control. What is needed in addition is: collision detection, the user interface, importing the artistic assets, sound, and the splash screens An essential aspect of the software design is simplicity. Because of the rather limited time frame for development, simple and yet effective methods must be devised so as to avoid a long and complex development protocol.

User Interface - *Boozification*

The process for making the user interface appear as if the user had been drinking became known as *boozification* within the group, and was the focus of a good deal of experimentation. What will be described here is the final implementation, not the entire path leading to it.

Instead of being passed to the software immediately, any user commands (key presses) are saved in two circular buffers. One is for steering, and one is for acceleration. The game runs at a fixed 30 frames per second, and so during each iteration all commands in the buffer for the next 0.033 seconds are interpreted, in temporal order. So, when a key is pressed, a function (*drunk_delay*) is called that determines the precise time when the associated command will be applied, and then places that command into the buffer for that time.

Every time the player takes a drink, the variable *delay_base* is incremented - this represented the mean time delay caused by blood alcohol content, 0.15 seconds per drink. This is obviously a simplification. To this delay we add a random value between 50% and 150% of *delay_base*, giving the unpredictable delay to be used for the command.

For example: in the case of one drink, we will have a total delay between 0.075 seconds and 0.225 seconds with a mean at 0.15 seconds (50% to 150% of the *delay_base*). In addition to changing the moment when the command is effected, its value is also changed in a similar way.

Note that we are talking here about seconds, but in fact everything in the program is transformed in number of frames. The values returned by those functions will be a number of frames and each frame is supposed to be 0.033 seconds long. The values are rounded to fit to a frame boundary. Our program runs with different speed on different computers that is why we calculate at each loop the time spent during the last one. That gives a better value of the delay, otherwise It would get really impossible to drive if the computer is a bit slow.

It is not really possible to know if the simulation is true to life, because if players who were actually impaired were to play the game their reports could not be relied upon. However, as is the case in all video games, the issue is to make things appear to be correct rather than to actually be correct.

Camera

The position of the camera compare to the car is not fixed. As with most driving games, the camera is positioned above and behind the car, and the camera's height increases with speed. Unfortunately the random motions introduced by the boozification process causes jerky camera moves which are quite distracting. The solution was to compute the position of the camera based on the average speed of the car over the previous 15 frames. This moving average technique is not uncommon in driving games, but is essential in the Booze Cruise.

Collision Detection

An important aspect of any real-time game is collision detection. For the Booze Cruise we again took a simple approach that uses a lookup table of object locations.

Figure 2: A screen from the base version of the Booze Cruise. When the timer (pink, top) counts down to zero, the ride is over.

Collisions are identified by looking up the vehicle's x and y coordinates in the table. Depending on the value (0 or 1) the car will have collided with a stationary object or not. If a collision takes place, then a simple bouncing mechanism is used to return the car to the course. This algorithm is inspired by methods in emergent computing; it involves checking squares around the vehicle's current location in the look up table until a free square is found or until a maximum radius is reached (5 squares). While this method leaves something to be desired as far as ricocheting off walls realistically etc. We find that for the purposes of our game thus far the method works adequately, and is fast.
The lookup table is populated by reading in a map file when the game starts up. This shows the location of fixed objects as 1 or 0 and is 2D, indexed by position. Loading the map in this way also enables us to easily change the layout of the course just by modifying the map file and the ground texture map (for aesthetic reasons).
One unusual limitation of the collision detection method described is that as the frame rate decreases the collision detection begins to fail. This results from the car moving so far in one frame that it moves outside of the bouncing radius and can no longer find the road. It is possible to keep the frame rates would be sufficiently high this would not cause problems during actual game play.

RESULTS

The combination of simplicity in implementation and practical design was wildly successful. The public release of the game in October 2007 was attended by a dozen media outlets, and the development team was supported by the Calgary Police Service and the Dean of Fine Arts. The media requests continued for a week, and resulted in more than 20 specific requests for copies of the game by police agencies and groups like MADD and SADD.

One interesting issue arose from this media attention. Some of the reporters, and a caller in an open-line radio program, suggested that the game could be used by people attempting to avoid being caught as impaired drivers. The feeling seemed to be that by practice with the game, drivers could avoid being identified as impaired because they would be driving better. Of course if this were so then that would be a good thing in and of itself - if the drivers can be made more competent by practice with our game then we may save a life.

However, this is unlikely. The point of the game, and one missed by most of the players and sponsors, is that the game does not actually involve driving. It involves a playing a driving game, and as such is a very practical metaphor. The target audience knows what it is like to play a driving game, and the Booze Cruise controls are the same as might be found in any typical instance of that genre. The connection is subtle to some: driving game is to Booze cruise as driving is to impaired driving. Indeed, many players of the Booze Cruise will not have driven a real car and will not know what that is like. Many will not have been drunk either, and so it is hard to explain why it is impossi-

Figure 3: A sample of media clips from the release of the Booze Cruise in October, 2007.

ble to drive a car properly while impaired. This game shows what it is like to play a driving game impaired, and that it is not possible to avoid the perception problems connected with the impairment and 'think yourself straight'.

The traditional approach in impaired driving education has been to show consequences, such as bent and twisted vehicles and bleeding, torn bodies. It may well be more effective to show instead that the effect of impairment is unavoidable and un accountable.

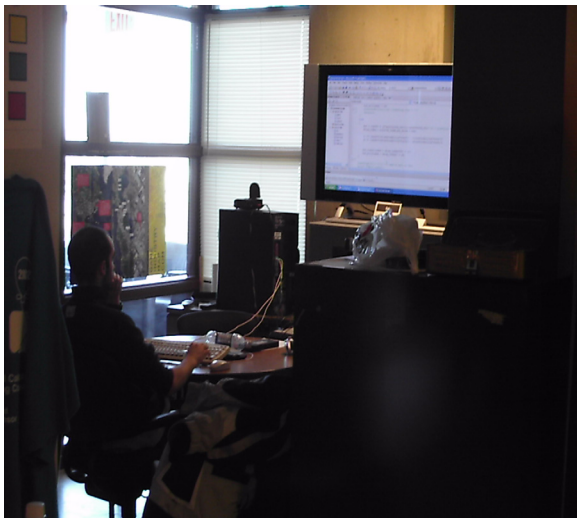
CONCLUSIONS

A student project in serious game development has led to a useful, practical device for use in schools and displays. Key aspects are:

- simplicity in design, so that the game does not try to do too much.
- simplicity in implementation, so that the development does not take too long. Reusing working code and implementing simple, effective algorithms are key here.
- The use of a 3D model is, if not unique, then at least rarely documented and effective.
- Portability is supported by the use of OpenGL and OpenAL.
- The team works best if artists and programmers are involved with each stage of development.
- An intense media response can be more important and more useful than academic publications. The media response to the Booze Cruise visited a more appropriate audience, in terms of who can actually use the technology, than a journal publication. It also was seen by vastly more people.

REFERENCES

1. J. Boer, (2005) **Game Audio Programming**, Charles River Media, Boston, MA.
2. Loki Software, (2000). **OpenAL Specification and Reference**, Ver 1.0. www.openal.org/openal_webstf/specs/oalspecs-specs.pdf
3. Michael, David and Chen, Sande (2006). **Serious Games - Games That Educate, Train, and Inform**, Thomson Course Technology, Boston, MA.
4. Parker, J. R., (2005). **Start Your Engines - Developing Driving and Racing Games**, Paraglyph Press, Scottsdale, AZ.
5. Sawyer, Ben (2002). **Serious Games: Improving Public Policy through Game-Based Learning and Simulation**. Woodrow Wilson Center. White paper.
6. Schreiner, D., Woo, M., Neider, J. and Davis, T. (2003). **OpenGL Programming Guide** (4th Ed.), Addison-Wesley, N.Y.



This work was a team effort, and is the result of a class project done for credit. These sorts of projects can result in extremely valuable results and practical 'real-world' experience. This is especially true when multidisciplinary teams are involved. This kind of activity needs to be promoted at Canadian Universities.