NTNU

Norwegian University of
Science and Technology

Report

IMT4888 Specialisation in Game Technology (2017 Fall)

Magic Sand Buggy

Gjøvik, 16/11/2017

Jan Tiemann

1 Introduction

This is a report about my project work in the course IMT4888 Specialisation in Game Technology. I have never written a report like this in my study career and I hope I meet the expectations.

The aim of this project was to build upon the results of Skotvold, Somby, M´Hamed, Lecoq and extend it to a full multi-player Virtual Reality(VR) / real sandbox game. The main vision was a cooperative racing game with a changing racetrack. One player is in VR driving a car around in a landscape. The landscape is based of the height profile of a real world sandbox. Another player or a group of players are at the sandbox and manipulate the sand. Changes in the real world sand are reflected by the in-game landscape at runtime. The position of the car and other race track elements along with the height profile are projected onto the sand. The aim of the game is to allow the driver to perform specific maneuvers. For example, if the current objective is to jump a long distance. The sandbox players have to form a hill with a long enough runway in-front of it to allow the driving player jump with enough speed off the hill.

Because of lacking manpower and changes in the underlying technology only a proof-of-concept could be realized.

1.1 Sandbox Applications

The foundation for this and the previous project was the Augmented Reality (AR) Sandbox created by researchers at UC Davis[1]. The later used Software *Magic Sand*, which is a partial port of the *AR Sandbox* to the *openframeworks* framework, was mainly written by Thomas Wolf[2]. Both use the same setup. A Microsoft *Kinect* and a projector are installed over a sandbox and face the sand surface. The depth sensors of the *Kinect* allow the application to scan the height of the surface. Afterwards, the projector is used to display this height map back on the sand. Extensive calibration between *Kinect* depth image and back projection is needed to align the projection with the sand surface.

The internal code structure of the *AR Sandbox* project was very opaque and made creating extensions difficult. Individual lines of code were commented but the architecture was not clear and the application was not consistently modularized. It felt like a further developed prototype. At 1/3 of the project time the alternative *Magic Sand* was found and a switch was made. *Magic Sand* was very well modularized. In addition to that, it was build on top of the *openframeworks* framework which made OS-transparent development and preparing the build environment easier.

1.2 Previous Work

The previous work was created for the fulfillment of a bachelor level course's requirement. It consisted of two modules. One server which was an extension to the *AR Sandbox* and a demo

[1] `https://arsandbox.ucdavis.edu`
[2] `https://github.com/thomwolf/Magic-Sand`

client application which was build with the Unity Engine. The server extracted the height map from the *AR Sandbox* and send it upon a request from the client over an TCP connection. The client rendered the height map by sculpting a plane accordingly. The transaction used a simple protocol which employed packets with a short header and a payload tail. To serialize the height map, the float values were truncated to three digits, converted to strings using the actual number chars and then send as ASCII chars over the network. Additionally the client employed meshes without updated colliders to present the height data. However, those colliders are essential if one wants to drive with something on it. The most important contribution to the current project was to locate where it was possible to extract the height data from *AR Sandbox*. This had to be a quite hard task since the quality of the code structure was lacking, as mentioned before.

When the setup was switched from *AR Sandbox* to *Magic Sand*, I decided to abandon the previous project and write my own implementation. I had to do that for the unity party anyway and it would have been more work to understand the existing server and adapt it to *Magic Sand* that to write it anew. Furthermore *openframeworks* offered an TCP server implementation so half of the server was already done. I kept the general protocol structure but more on that in chapter 2.3.

2 Method

The project had the same structure as the previous one. It was split into client and server. The server was an extension to the sandbox software with the main task of extracting and sending the height data. In addition to that, the server had to receive and render the position of the car back onto the sand in this project. The client had to request height data, sculpt terrain according to it, take user inputs, run a racing game and send the position of the car back to the server.

2.1 Server

c++ build uppon magic Sand using image filtering

baseplane + height = true height not shure about texture values scale to baseplane -> guessed half offset of baseplane offset and seems to work really well

2.2 Client

unity c#

Vr simply import into unity from asset store and go

default car physisc made for street driving heightend body and increased dampening to improve behavior smothing and spare

null max height scale to max level not useable in action

2.3 Protocol

too fast -> people are visible in the sand

streaming unity cam not doable in time two camera approach - single client different textures for terrain -> different render runs streaming camera out of unity to streaming service is hard and not openly done before two client solution the whole game was build in single player not enouth experience to split it in short time

using two ports was mistake but no time to fix, maybe not

3 Discussion

3.1 Improvements

Was only a single person

buggy graphic, sprite or drawing instead of dot

smoothing between transitons buggy/sand only do (big) changes of terrain when not in view

pick ups

car physics

raceing game stuff

3.2 Ethical Concerns

vs. natural interaction (traditional sandbox), tech/advancemend allways good?

racial body issues, handicaped people

more and more technologial playground ->higher entrance borders - body, monetary issues -> more exclusive

4 Conclusion

will continue, at least for feature jam fun project will ask pb museum if they want sandbox