Prototyping Report

Group 2

Kwok Hou

Table of Contents

Prototype: Gameplay 2

Prototype - Procedural Level generator 2

Prototype - Level 3

Prototype - Multiplayer..4

Prototype - Models 5

Prototype Menu 6

Prototype - Sound 7

Prototype: Gameplay

The core gameplay is fundamental to a good game. This determines for a large amount how much fun the player has in playing the game. This prototype was used to develop gameplay and specifically the gravity shift system.

Does it work as expected?

Based on playtesting, a lot of changes have been made to the gameplay. This was not because it didn't work as expected, but because the changes improvement the gameplay experience. We had a big problem with the gravity switch; the player lost its orientation if the gravity changed. This was solved, and circumvented by rewriting the code with ray tracing smooth lerp transitions. Camera and movement are based on basic FPS movement. We are still experimenting with the function of the different mouse buttons. We decided to split the function of the bullet to one bullet that can hit other players (Left mouse button) and another bullet to make the gravity switch. (Right mouse button)

There is still a bug with movement. (when you stop moving you go back a little). This is a minor annoyance we were not able to fix yet.

Did it take more time to develop than expected?

Gameplay is something we expect to work on for the rest of the development. We did not experience major setbacks.

Are we satisfied with the prototype?

The game is already fun to play and move around in. This is very positive and we are satisfied with the result we achieved in the first two weeks.

Will we use it in our final game?

This is the starting point we will base our game upon. All gameplay elements are subject to change, but we do not expect to alter a lot.

Does it need improvement, and why?

The multiplayer aspect needs to be merged to do further testing and adjustments. The core movement is fun to play and does not need an improvement as of now.

Prototype - Procedural Level generator

The initial idea was to program a procedural level generator that was not grid based. It would pick a given number of initial spawn places (the user could input how many) and it would spawn a cube there. The cube itself would have a script attached that chooses one of the cube's neighboring positions and spawns another block there (which also has the same script attached, so it's like a recursive algorithm). It would send a parameter to the next block indicating how many blocks should still spawn (could also be adjusted by the user).

Does it work as expected?

The concept worked, it would create levels, but there were quite a few issues.

- It was difficult to prevent the algorithm from growing outside the defined dimensions of the level.
- If 2 spawns grow towards each other, because it isn't grid based, the overlap between spawns may be rather awkward, which would not favor gameplay
- There was a bug where if I tried to implement a "favored direction" into the algorithm (e.g. the spawns are more likely to grow in x-direction) it wouldn't quite work.
- There was a bug where sometimes blocks would be spawned multiple times at the same location.

As we tried to fix these bugs, we realized that it would probably just be better to rewrite the algorithm and use a grid based algorithm that fills up a matrix with 1's (representing blocks), and applying a separated draw function that instantiates blocks at the positions indicated in the matrix. This would certainly solve the problem of blocks being spawned multiple times at the same location (the draw function would only go over all matrix elements once, and thus only draw a block at most once in a certain position). This would also make it easier to force the algorithm to stay within the indicated dimensions of the level, because you simply can't define 1's outside of the matrix space. We also discovered the bug causing the preferred direction algorithm not to work properly and fixed it. Additionally, the new algorithm is a lot quicker than the old one. You don't want people to wait for a minute before the level is created and they can play. The old algorithm used colliders to see if the next position was free to spawn a block, this takes a lot of time. It is much quicker to just check whether an entry in the matrix is 1 or 0.

Did it take more time to develop than expected?

Definitely, especially because we had to rewrite the entire algorithm after realizing the first idea was not a very good idea. Also the amount of bugs was larger than expected.

Are we satisfied with the prototype?

Yes. The levels created are really creative and work well (has been tested in the actual game).

Will we use it in our final game?

Certainly. Procedural level generation will prevent people from becoming to familiar with the levels and always picking spots that work well, you have to always get used to a new level, and in this sense everyone is equal in knowledge about the level.

Does it need improvement, and why?

As of right now there are no things that need to be improved in the algorithm.

Prototype - Level

Besides the procedural generated part the level is made out of the skybox and team bases.

Does it work as expected?

Making a nice skybox in Photoshop was easy enough. Finding out how to add it to Unity wasn't that much trouble either. Although one of the problems with it was that it had a clear seam at one point. We fixed this by horizontally mirroring the skybox in the middle. Another problem was that, as we mapped the skybox texture cylindrical to a sphere, the texture is pinched in the top and the bottom. This looked bad, and so far we fixed it by making this part of the texture black.
But we didn't just want a static skybox. We want a rotating one. This has proven to be much more of

a challenge. For the team bases we created a blender model for the bases and added it to Unity. We gave it a Mesh collider and tried it out. It did not work out as expected as some of the shots were going straight through the building. Eventually we found out this was caused by the way the gravity shift worked and the new implementation of the gravity shift has fixed this.

Did it take more time to develop than expected?

The rotation on the skybox is taking more time than expected. The Team base took more time than expected because we didn't know what was the caused the mesh collider problem at first.

Are you satisfied with your prototype?

Not satisfied with the rotation on the skybox. The Team base is satisfactory.

Will you use it in your final game?

We will certainly use the skybox, but we might make changes to the look of it. We might use the Team base if we decide to make a fitting game-mode for it.

Does it need improvement, and why?

The team base is alright, it just needs a nice texture. And for the rotation of the skybox, we now have it working to rotate around the y-axis, but adding the x and z axes' rotation causes problems which are currently being worked on.

Prototype - Multiplayer

Surreal Gravity is intended as a multiplayer game; single player will most likely be an option, but multiplayer will be the core aspect. Therefore this prototype has been made.

This prototype consists of 2 parts: Networking and Accounts/Account storage. The networking part was finished first; the accounts part second.

Does it work as expected?

Networking

(Note: as of now, this prototype has been tested on 1 PC (thus not tested if 2 PC's can connect to each other)

One game client has the option to either create a server or connect to a server (via the same network, so LAN). If a server has been created, another game client cannot create a server (since the server names are the same). A game client can refresh hosts; if a server has been created, the game client can connect to the server. You can only control 1 'player' on a game client.

It works as expected: if the player moves on one game client, every other game client sees the player move. If one player (doesn't matter which one) comes close to the square, the square turns blue on every game client. There is thus obviously a connection, and via that connection all players can simultaneously move in the game.

Accounts

If a server is created, or a connection with a server has been made, you have to login to an account. If needed, you can first register one, and then log in on that account (or another one).

The accounts are as of now stored locally in a text file (simply username \n password \n username \n password and so on). Our goal is to store the accounts on a webserver (preferably in a database), but right now not much is clear about the webserver which should be provided.

If you are logged in, your player spawns and your username is shown on the top of the screen. The color of your username depends on the team you've chosen. If a game client connects, and logs in, every other connected game client sees the player connecting through both the spawning of the player as well as the manifestation of his username.

It works as expected: you can login to an account (given that the account exists and the password is correct) and everyone knows which player you are due to the networking.

Did it take more time to develop than expected?

This prototype took more time to develop than initially was thought. The networking part was finished quite fast, however the accounts part took more time to finish. More specifically, being able to show your username to every other client was more complex than we had thought. Due to this, it took more time to finish the prototype.

Are we satisfied with the prototype?

We are quite satisfied with this prototype. We should be able to implement this prototype in our final game, with only a few adjustments. The core of multiplayer is already there; you can connect to each other and you can log in to accounts.

Will we use it in our final game?

We will definitely use this prototype in our final game; as previously mentioned, multiplayer is the core aspect of our game.

Does it need improvement, and why?

This prototype needs some improvements.

In this prototype, the accounts are stored locally. This is not preferable when playing via LAN (if 1 client registers an account it will only be stored on that client's pc). We want to store the accounts on a webserver, if possible in a database. We will probably also look into encoding the passwords, to prevent people "hacking" our game.

In this prototype, you have to connect first, then log in and then you are spawned. This is not ideal; hence, we will probably implement a sort of lobby; here you can connect with other players while already logged in to an account. This gives the possibility of playing single player locally, which is important. This also makes sure that the server initializes everything before the game is loaded and everyone joins. Moreover, you can, for example, first choose your character before playing. And lastly, it will make sure the game start when everybody is ready (and not that the server is playing already when he is alone).

Prototype - Models

The current prototypes of the models in our game are:

- Bullet
- Character

- Weapon
- Flag

Does it work as expected?

At first the futuristic bullet was created using a puck as core; however we decided that it would look better if the core was a ball. The bullet was also animated with easing on and had to be turned off. The animation of the character and bullet are now working as expected. For the weapon we created a RPG at first, after some reviewing a futuristic weapon would be more fitting in our game. During the making of the weapon we encountered a strange problem: some parts of the weapon suddenly disappeared when joining the parts. To fix this problem a few redesigns had to be done, now the weapon works as expected. The flag is a pretty simple blender model, so that worked pretty well. The problem with it was with Unity´s built-in interactive cloth. Since the Mesh provided by Unity is a plane, the face of the flag is only seen from one side. For now we have fixed this by making two of these faces with the backs against each other. This is still not a perfect solution so we´ll definitely look into changing this later on.

Did it take more time to develop than expected?

Because of the problems we encountered it took some more time to develop than expected.

Are we satisfied with the prototype?

At this point we are satisfied with the models.

Will we use it in our final game?

Yes, these prototypes will be used in our final game.

Does it need improvement, and why?

The models definitely need some improvements and optimizations, a texture and mapping have still to be added to the character.

Prototype Menu

Do they work as expected?

Yes

Does it take more time to develop than expected?

Due to experience the programming of the level was not particularly hard. We will however use the menu to manage accounts and server/clients. This is not yet implemented and is expected to be more of a challenge.

Are you satisfied with the prototype?

Yes, we are very satisfied. The font used is very appealing, and the technical implementation of the menu buttons allows for efficient programming.

Will you use it in your final game?

Yes, we will use this in our final game.

Does it need improvement, and why?

Graphically, the menu has made good progress but needs to be tweaked still. The menu is also not very dynamic and we are looking for ways to implement that later. Also there are currently no sounds added to the menu. These have to be added later aswell.

Prototype - Sound

For the game, two different kinds of sound types are needed for the game.

First we will need <u>short, recognizable sound effects/samples</u>. This to make the game more realistic and professional, but also to let the player project himself more into the game. We can divide these in 6 different main categories (see bold titles below). All of these categories have a few sub-events which will all need an own sound effect. After searching for the proper effects we downloaded them and put the name of the downloaded files behind the names of the corresponding sub-events.

After choosing the best sounds effect for each item, we will modify the sound effect to fit it better in the game. The following sounds were selected:

1.Gun:

- A sound representing the shot of a gun/launch of a rocket: *Anti-aircraft gun, Chain gun, Bomb explosion!, Winchester 1873 Single Shots!!!, Gun Shot!!!, Gun Fire!!, Barrett M107 Sniper Rifle*
- A sound representing the flying bullet/rocket: *Missile Fire War*
- A sound representing an explosion when a bullet hits a wall: *Super Punch MMA, Explosion And Debris, Bomb Exploding!!, Blast, Grenade!!*
- A sound representing an explosion when a bullet hits another player Normal Explosion + *Blood Splatters!, Blood Squirt,*
- A sound representing the reloading of a gun: *Cracking Peanut Shells, Cocking Gun!!, Beer Can opening!,*

2. Achievements:

- *Reward sound when you hit another player: Evil Laugh Cackle, Evil Laugh!!,*
- Screaming when getting hit by a bullet from another player and die: *Egg Cracking Sound, Evil Laugh Cackle, Breathing Vent, Crowd BOO!, Evil Yelling!, Female Scream*
- Screaming when falling of a level and die: *Screaming Female, Fall And Splat (te kort)> woman screaming!!, Person Screaming NO!!, Tortured Person Screaming!!!, Scream and Die Fx!!!!, Falling Off !!!!!,*

3.Interface sound:

- *A sound when clicking any option in the menu: IM Reply Computer, Click Button 2, Button Click On, Click2!!!,*

- A sound when the game begins: *Bell sound ring, Boxing Bell Start Round, Box Mma Or Wrestling, Metal Gong,*
- A sound when the game is finished and the highscores will be displayed: *Heartbeat*
- A sound when clicking 'Exit game': *Goodbye Female,*
- Aftellen tot het spel begint: *Apollo Moonlanding Sound,*

4. Background sounds while playing:

- Level Background: *Alien recording sound, Light saber turn on, Martian Speak, Time Travel Clip, Metal Gong, Tune Radio, Killer Movie Scene, Evil Ambience, Spooky Wind Howling, Demon Zomie ambiance,*
- Players walking?: *Footsteps walking*

5. Enemies:

- Ufo: *Dc9 Landing, F15 Fighter Jet Flyby,*

6. Sound during menu:

- *Elves laughing High Pitch, Alien Brain Scanner!!, Evil Ambiance,*

Furthermore, we also needed music to create an ambience, as well as encouraging the player, giving him an extra boost. This is:

- Music during the menu. This is the first music the player will hear when starting the game.
- Music during the playing of the game. This will give the player a boost or create an atmosphere in the game

All these sounds can be set off in 'Options' in the menu. There will be an apart button for the music and the sound effects. In this way, the player can choose to disable the music (if they don't like it or don't want to hear it at the moment) but keeping the sound. If you don't want to have any sound at all, you can disable both options: 'sounds' and 'music'.

The soundeffects were downloaded from different internet sites:

- *http://audiojungle.net/category/sound/game-sounds*
- *http://soundbible.com/tags-game.html*
- *http://www.bfxr.net/*

The music will be produced by the music producer of the team.

Does it work as expected?

Yes, they work as expected.

Did it take more time to develop than expected?

Yes, because huge libraries had to be gone through. After finding fitting sound effects, they had to be modified, using Reeverb, Parametric Equalizing, Filtering, Delaying and other Mixing techniques. The background music for during the menu and game still has to be developed.

Are we satisfied with the prototype?

We are not entirely satisfied with the sound effects. There are some sound effects that sound too mechanical for our game. We want them more futuristic and ‘space-likely’.

Will we use it in our final game?

We will use the sounds that we are satisfied with definitely in our game, but there have to be some new sound effects found.

Does it need improvement, and why?

The sound effects that we are not satisfied with have to be modified so they will fit optimally in the context of our game (which is not the case right now). If this is not possible this way, they will need to be replaced with other sound effects.