CS 485: XNA
Team 3: Frank Masby, Anthony Saunders, Bobby Vandiver

Final Game Write Up

Frank Masby:

It was truly a pleasure and learning experience working with the guys of group 3. Everyone contributed greatly to the project. It was chance to learn and implement object oriented programming as a group. We developed functionality independently then merged it into a master branch using version control.

We held four meeting throughout the durations of the semester. During these meetings we had progress updates on previously set goals and task. We were able to share problems and discuss certain debugging solutions. During the meetings we were also able to discuss different ideas of added functionality. We corresponded via email mostly. Checking off task and checking out task.
During the development I tried to be conscious of writing code that was expensive to process. Initially I drew the bombard model on the CPU. I would add multiple models in the world in order to compensate for the functionality that would be added from the other group members. After doing this I noticed some lag in the gameplay. To prevent performance degradation I rendered and textured the models using the GPU. I also decided to randomly place faux asteroids (billboards) in the world to look like multiple asteroids were coming from multiple directions.

Another difficulty I faced was getting the asteroid to be drawn in a way where it was moving slow enough to be viewed with an efficent amount of time to dodge it. However not to slow where it would be to easy to dodge it. I ended up doing a little extra scaling. I made the asteroid larger and quicker as it approached the planet's surface. This effect allowed the model to be drawn in the world long enough to dodge, while being visible within the bounds of the far clipping plane.

Bombard.cs

Bombard()
instantiates all variables, terrain, camera, effect, andsmoke effect

RandomPosition()

gets a random position and moves the object based on milliseconds

ShakeCamera()

Shakes the camera based on how close the asteroid intersection collision is to the camera

NewAsteroid()

generates new asteroid once the asteroid drawn intersects the billboard surface

Update()

calls shake camera, new asteroid, updates time, and changes intensity (based on asteroid position)

spreadCollision()

spreads the billboards out when the asteroid collides with the terrain

Draw()

sets the effect and matrices for the bombard effects
sets the matrices for the camera
draws the model in the mesh using the effect files

Anthony Saunders:

My Contributions to our XNA project were mostly on the ship level. I coded most of the entirety of the level by myself. I had some help debugging certain issues that I had trouble with and I recycled some of the initial code and some of the features from the planet level.

The space camera is restricted on two axes and limited in movement controls. I did this to generate a semi playable level (since you can't leave the asteroid field). It seemed like a reasonable way to approach the level as making asteroids from every direction would have slowed it down a great deal.

My skybox was a basic background (no actual skybox). This helped simulate space in my opinion, and the asteroids were the central focus of the level so I wanted them to govern the levels appearance. And since the direction was limited, a skybox would have been slightly unnecessary (although I did plan to implement a better one)

I modified the HUD from the planet level to be applicable with the space level and using asteroids for positions instead of the alien monsters. I had to make the HUD take parameters for the positioning, but the overall effect came out quite nice after I modded it.

I wrote the following object classes:

Bullet.cs (modified version of asteroid.cs)
Asteroid.cs
Spaceship.cs
Explosion.cs

AsteroidField.cs generated the majority of the space level. It generates the asteroid objects into an array and displays them on the level in appropriate locations and directions based on if the asteroid is destroyed or not. Explosion.cs is a smaller version of the asteroids, meant to be bits and pieces. These are generated on asteroid deletion at the asteroids target location. I really liked this idea (although it didn't work out).

Overall I just tried to incorporate anything I felt was valuable to the games design from the planet level into the space level and to follow Bobby's original object oriented design. Although parts of my level are(and were) bad logic at some points (originally I wasn't using triggers in the update correctly) I think it came out decent and I am semi proud of my design of that level and some of the methods I used.

My group members were great and I got very lucky to get two guys that worked just as hard (if not harder) than I did on this final project.

Bobby Vandiver:

This project gave me the opportunity to see a small video game be built from the ground up and gain experience with the myriad of design decisions that have to be made along the way. I've spent quite some time reading in my free time about how to apply object oriented best practices to video games, so I was anxious to really apply this.

Since I did the initial ground work for this game, I had the chance to layout a general architecture for the game. This was the most interesting aspect of the project for me. With all of the parts involved in a game, there are a multitude of design decisions that must be made to facilitate ease of development and debugging later.

As we had originally intended to have three distinct levels, I tried my best to create a few base classes and a general overall structure so that my teammates could create a few derived classes and implement a few methods and facilitate adding new levels. I also tried something similar in the case of models. In my opinion, the Level and Camera class hierarchies accomplished this goal while the BasicModel class did not. This was due in large part to my own lack of experience working with models and a lack of understanding of exactly what general functionality is required for a model in a 3D game.

I'm really impressed with the enthusiasm and ideas that my team mates brought to the table. I'm equally impressed with the interest they demonstrated in learning more about things like version control that are not covered in the CS curriculum. As an aside, I think this is a huge failing on the part of the CS program. If group projects are going to be emphasized, then there should be some required class or seminar or *something* that focuses on developing the skill sets needed to work effectively in a group in software development. Theory entirely divorced from practice is wasted theory, in my opinion.

Overall, it was a positive experience and I really enjoyed working with Frank and Anthony. While our game does not demonstrate many advanced techniques or really push the envelope of what XNA can do, it does demonstrate what three capable people can accomplish in a short amount of time without relying on third party libraries.