

**ICT 312 Physics Simulation**

**Milestone One**

**Design Document**

**By Arran Ford, Hamish Carrier and Timothy Veletta**

**16/09/2013**

**Contents Page**

1. Title Page

1.1 Coversheet

2.Program Design Philosophy

2.1Model Information

2.2Program Information

3. Physical Design Specifications

**2. Program Design Philosophy**

Our approach to the design and subsequent creation of this project was a 3 step procedure. We initially started out by getting together and thoroughly planningout how we were going to approach the project. It was at this stage that we decided on what we wanted to get out of the project and what would be the best way to do it, as we were all also doing ICT313 (and in two different groups) we decided that the ability to be able to work independently was one of the most important goals. This way depending on the varying group schedules we could each work on the project at times that best fit into our own schedules. To facilitate the ability to work independently we split the first major milestone of the project up into major three categories, graphics development, collision detection and resolution, and asset creation. This kept the amount of points of overlap to a minimum, which minimised the amount of conflict we encountered when merging / sharing each of the three categories.

Once we finished the planning stage we came together as a group to measure out and record the physical design specifications of the area we planned on simulating. In addition to physical measurements we also photographed all the models we planned on creating, and any textures that we would need to replicate / make use of.

After all the physical specifications were recorded we each moved onto the category we had decided on earlier. An advantage of splitting up the tasks into three categories with as few dependencies on the other tasks as possible, was that we were able to enter the development stage and work concurrently. Without ever having to halt development while waiting on another party to finish a part of their section.

As a result of our design philosophy we were able to develop much quicker than we had done in previous group projects, and the number of problems we encountered was also much lower than previously experienced.

**2.1 Model Information**

All of the assets were created in two programs, the 3D models in 3DS Max, and the textures in Photoshop. This was done due to the ease of use of the programs, and due to the ability of Ogre3D to easily render pre-made 3D files. Once the files were created in 3DS Max they were exported in a .mesh format, for which Ogre3D has inbuilt functionality that can be to load it.

**2.2 Program information.**

Our program consists of 3 main parts, classes associated with running the game, classes associated with rendering, and the collision system.

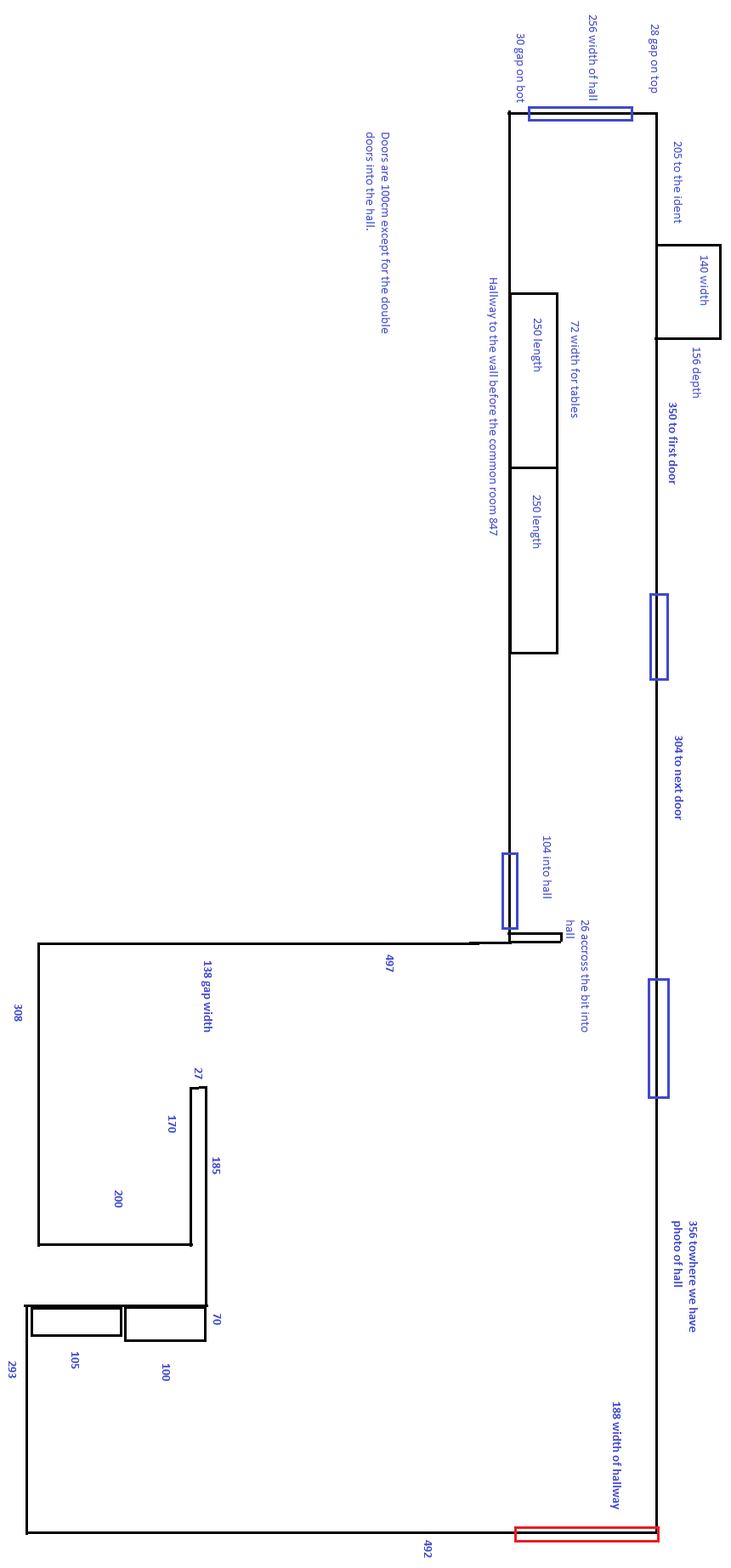
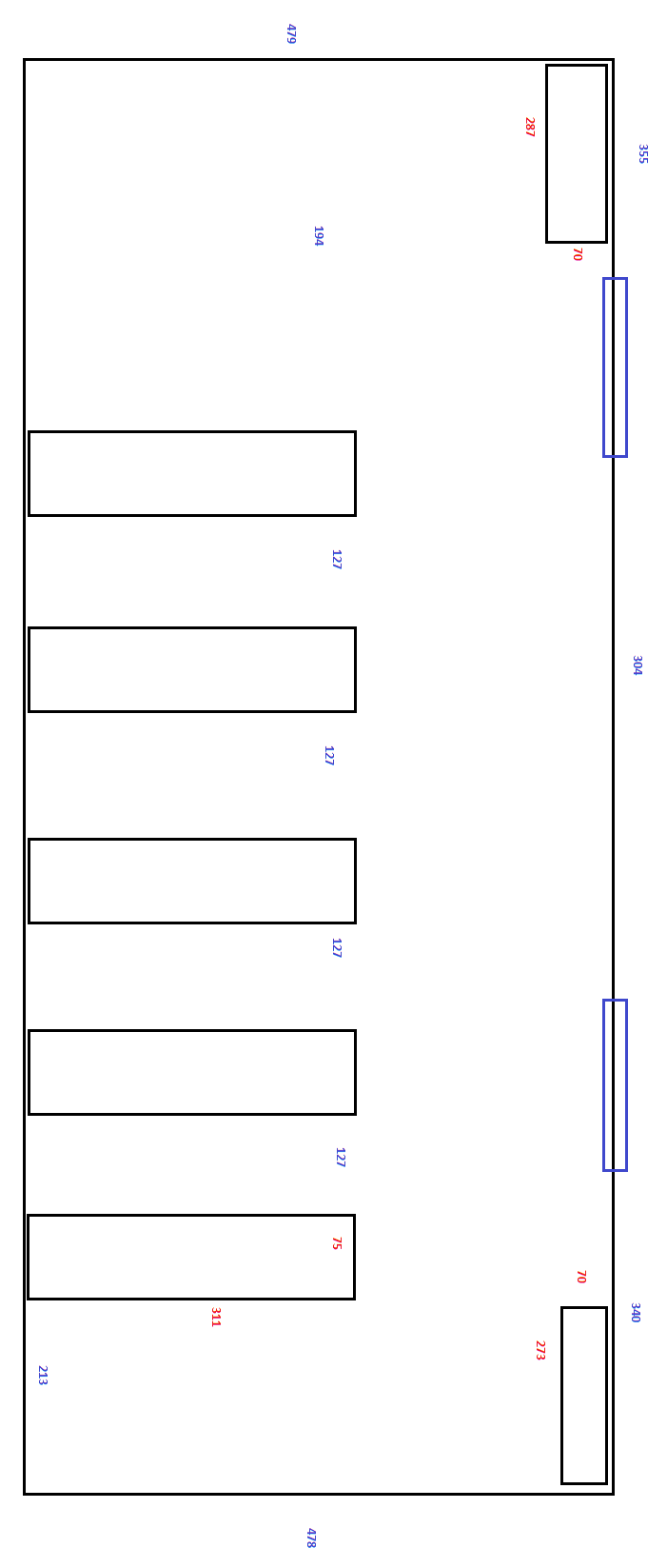
For rendering we use the Graphics API “Ogre” - tim stuff here.

The collision system uses the API “Bullet Physics” to detect collisions, each collision object in the world has a void pointer that points to a custom object we create that is associated with with the collision object, allowing the collision system to trigger collision resolution in the objects that have collided.

For this first milestone, the only object that is affected by collision is the TemporaryPlayerObject which represents the camera. The void pointer system will allow us to later associate NPC and physics objects with collision objects also. When a collision is detected, the camera is moved back to its previous location which is stored in the TemporaryPlayerObject.

**Physical Design Specifications**

The following Diagrams are electronic versions of the original hand drawn specifications we created from measuring the rooms we modelled.



For larger versions of images see:

<http://i.imgur.com/H0yi7Zr.png>

and

<http://i.imgur.com/I7tqwMN.png>

respectively.