# Report – Stage 2

**June 2015**

A major portion of the code for the Silhouette Method implementation has been done. It now works for an obstacle anywhere in the arena.

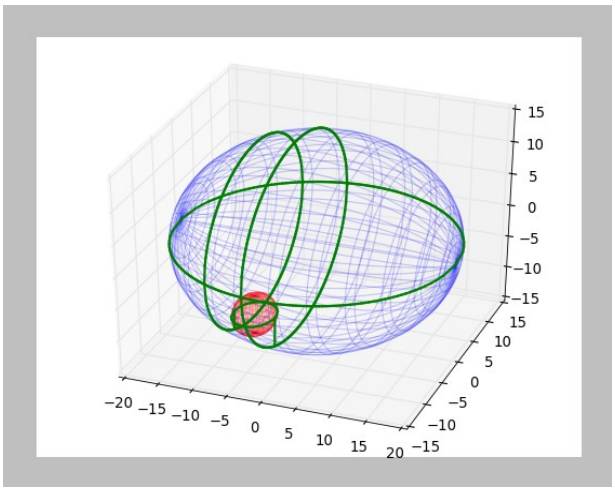
The issues we faced in the previous attempt to solve this problem had forced us to simplify the position of the obstacle.

Now, we no longer face any such restrictions. We can place the obstacles anywhere in the designated area. The obstacles can be of any shape and size. And most importantly, the obstacles can be in any configuration (rotation) in space.

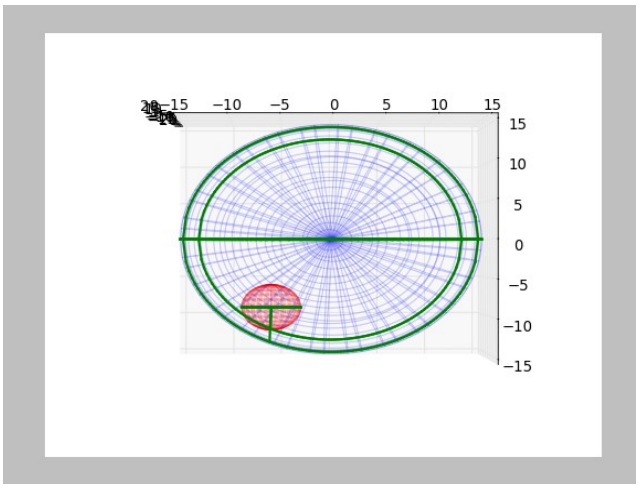
The images that follow display sample Road maps generated for different settings.

The Blue Ellipsoid is the bounding area and the Red ones are obstacles to be avoided.

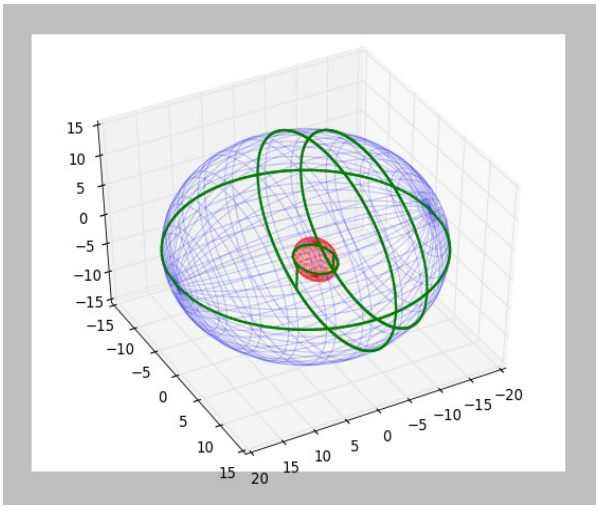
Obstacle 1 : Aligned along the coordinate axes but spatially random

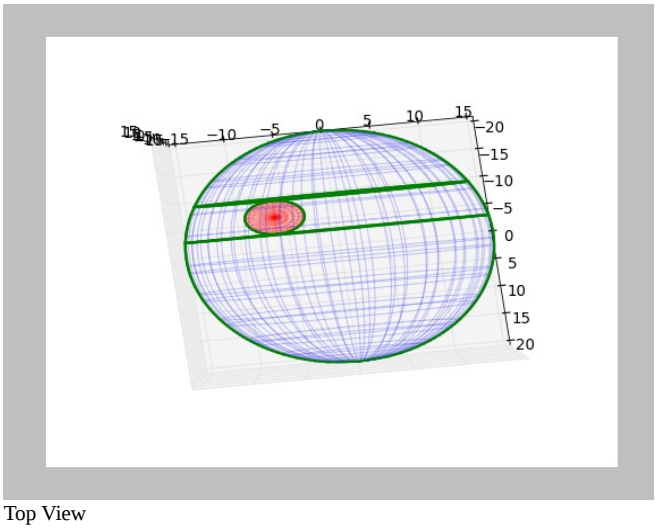


3D View



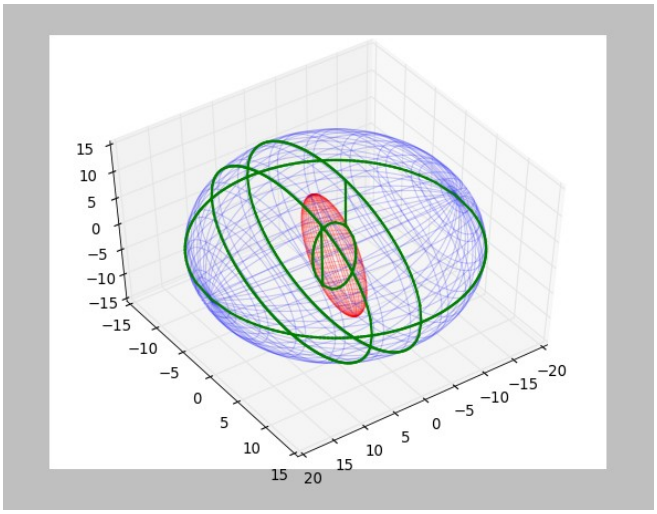
Side View



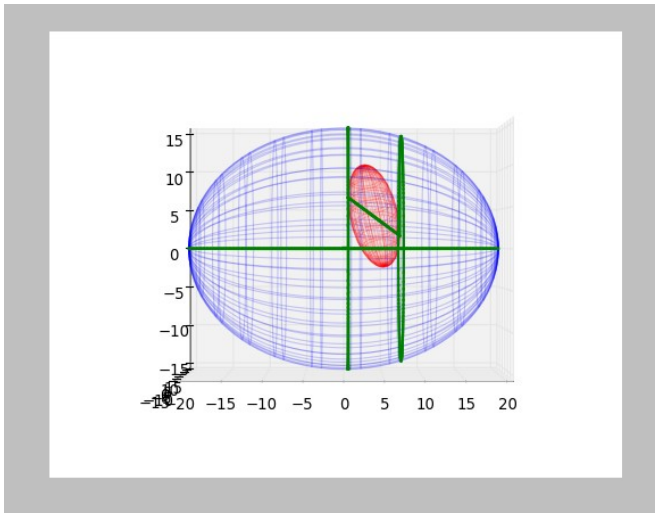


Top View

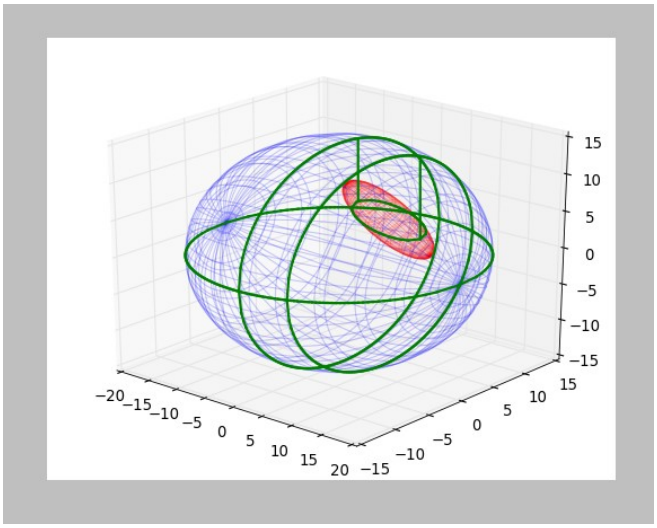
## Obstacle 2 : Arbitrary Configuration

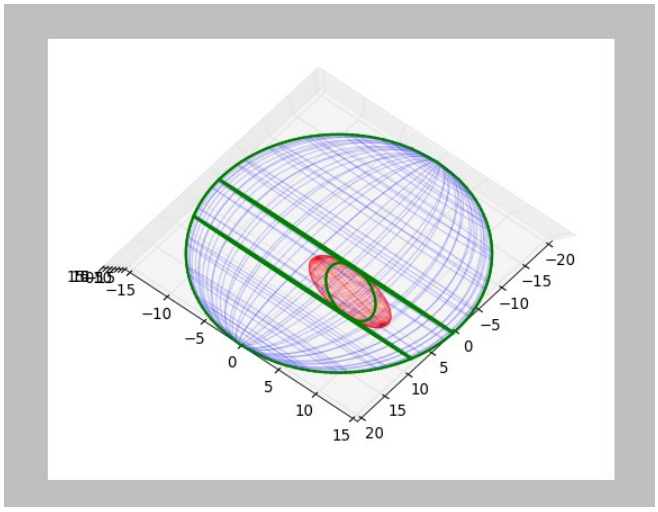


3D View



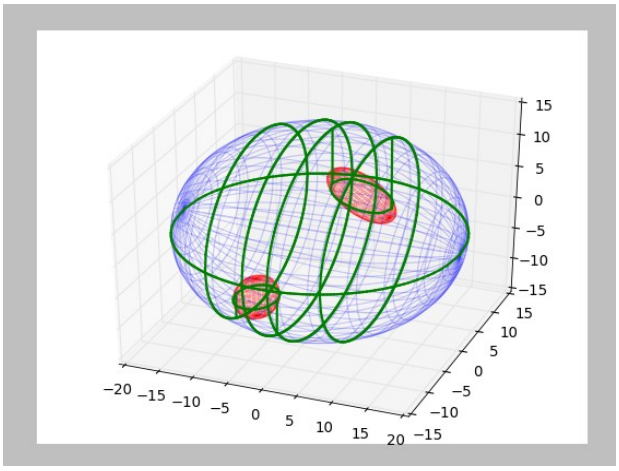
Side View

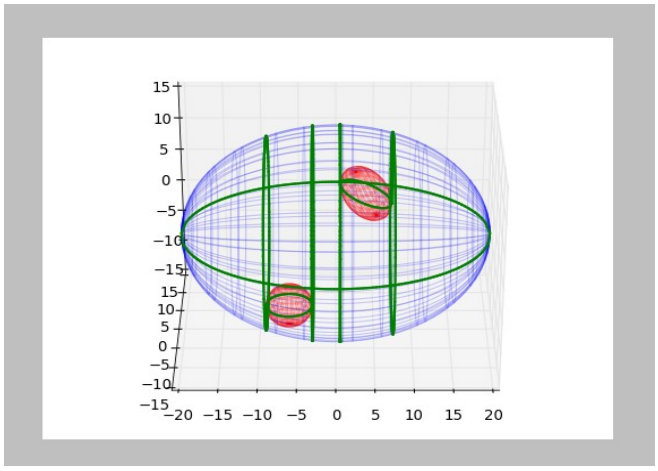


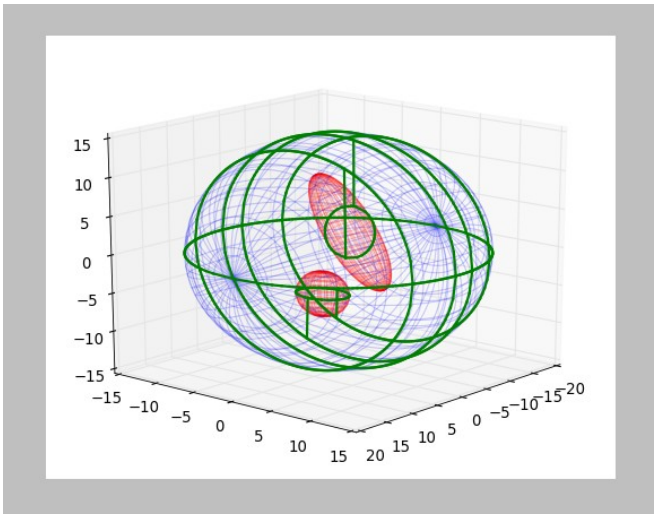
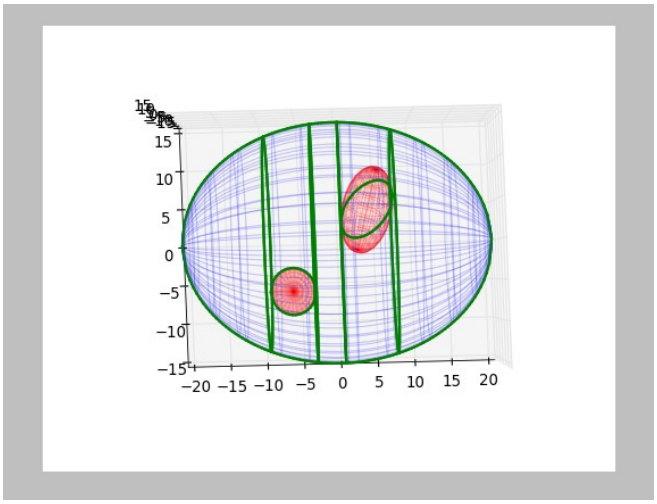


Top View

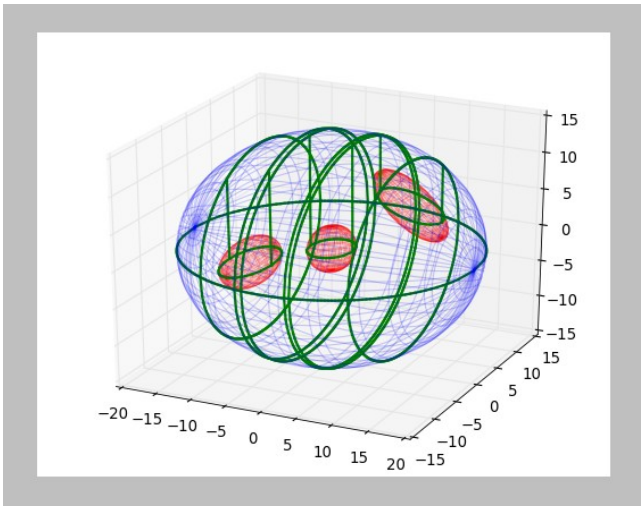
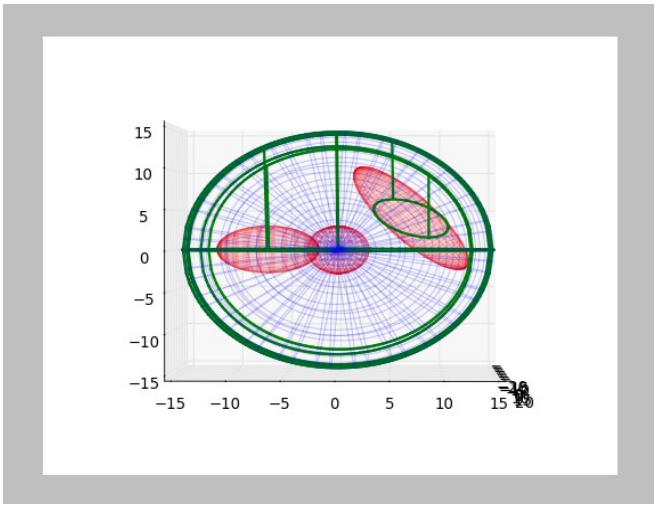
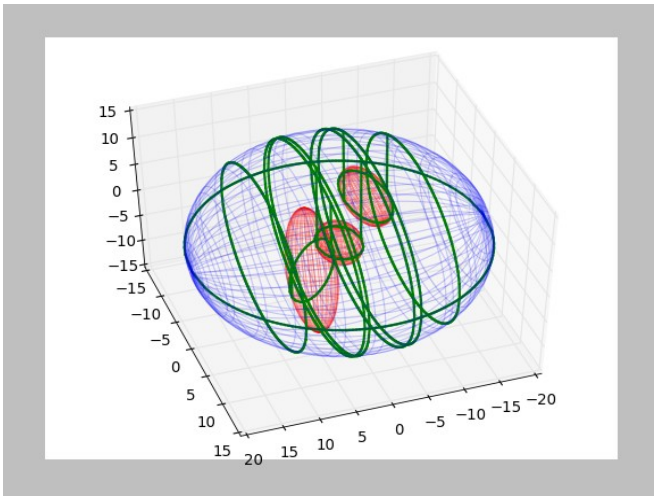
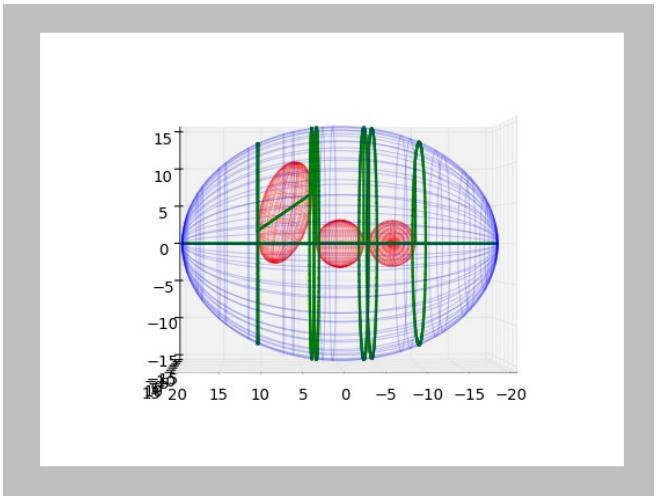
## Combined Obstacles:





| 3D View | Side View |
| --- | --- |
|  |  |
| | Top View |

Three Obstacles :

| | |
| --- | --- |
|  |  |
| 3D View | Side View |
|  |  |
| | Side View 2 |

**Code**

Major Modules:

The following is a short list covering the modules in the project in a rather boring fashion.

Though the entire project has been re-written, the Major Modules that have developed for the first time are marked with a star (*) :

1. Module to **process the input** provided by the user and present it as Ellipsoids and their origins
2. Module for **Road map Creation** at a given dimension. This module is **called recursively** in higher dimensions till we reach the 2D case, the base case.
3. Module to calculate the **intersection point of a Plane and an Ellipsoid** (Tangent Plane). *
4. Module to use the Module in 3. to calculate all the critical points, to be encountered in this dimension. *
5. Module to detect the presence of a Critical Slice and run modules 9 and 10 for recursion on the lower dimension. *
6. Module to link the vectors optimally along the road map as the sweep proceeds to generate the edges.
7. Module to calculate the **intersection along a given axis** for a given ellipse.
8. Module to calculate the ellipses to be connected in this particular slice.
9. Module to **calculate the ellipsoids of a lower dimension** from a given dimension. This module is central to the present system. We had discussed this in our last meeting. *
10. Module to **calculate the center of the ellipsoids** of a lower dimension from a higher dimension. *
11. Module to end the sweep at a particular recursion level and clean it up.

**Issues with my code** (Still in Progress)

The output works best when the obstacles are not too close to each other. This is because there are two modules still missing from our code:

1. Module to detect whether a point on the road map lies in another obstacle.
2. Module to deal with intersecting obstacles

**Note**

I have created the output as in the Silhouette Method discussed in Latombe. The sweep for one recursion level is only along one axis (I made a provision to extend to all axes). My reason is it suffices our needs as a sweep along just one direction ensures completeness. Also, it provides clarity to the Road map in the final plot.

I hope the images clearly depict the results generated.