Path Planning in Ellipsoidal Configuration Space using Silhouette method

September 7, 2016

bstract

This report presents the methodology used for Implementing Silhouette method for Robot motion planning in Configuration spaces in arbitrary dimensions. collection of ellipsoids are used to design and implement this Configuration Space of the robot. The path is then designed based on the Silhouette method as described by Canny [2], which creates semi-free paths in the configuration space and is complete. Djikstra's algorithm is used to connect the paths obtained. This report introduces each of the key areas of the project and follows a step by step implementation of each. To conclude, we discuss the model output by our implementation and future directions of this project.

1 Introduction

1.1 Motion Planning

Robot Motion Planning comprises the problem of finding a valid path for a robot from its start configuration to its goal configuration. robot conf iguration q is a specification which determines the positions of all robot points relative to a fixed coordinate system and the configuration space is the space of all possible configurations. valid configuration is one in which the robot does not "touch" or intersect any obstacle.

valid path implies that at each instant of execution of the path, each configuration should be valid in the configuration space.

Let $W \subseteq R^m$, where m is dimension of the configuration space be the work-space of the robot, $O \subseteq W$ the set of obstacles and (q) the robot in configuration $q \in C$.

$$C_{free} = \{q \in C \quad | \quad (q) \cap O = \phi\}$$

$$q_{valid} \in C_{free}$$

$$C_{obs} = C/C_{free}$$

Silhouette method is classified as a roadmap Method. roadmap is a union of curves in the configuration space C_{free}. roadmap is said to be accessible if there exists a path from q_{start} to q', where $q' \in C_{free}$. roadmap is said to be departable if there

exists a path from q'' to q_{end}, where $q'' \in C_{free}$. Finally, a roadmap is said to be connectable if there exists a path from q' to q''. Silhouette method is a complete method, that is, the method extracts a path from a start configuration to the goal configuration if any exists in the selected roadmap, or outputs failure if no such path exists.

2 Silhouette Method

Let S be a compact subset of R^m, ($S \subseteq R^m$). set S of real numbers is compact if every sequence in S has a subsequence that converges to an element again contained in S.

In Silhouette method, a hyperplane of dimension (m 1) is swept across subset S, perpendicular to a reference axis, say the x_1-axis. The hyperplane is swept through at regular intervals of x_1. t a position, $x_1 = c$, it is denoted by P_c.

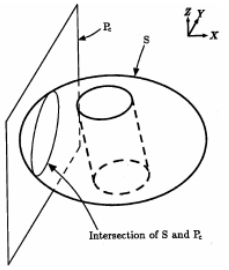

Figure 1: Depiction of P_c, a 2D plane, cutting subset S

t each hyperplane, the extremal points of intersection between the slice and the compact space and the obstacles are calculated along one arbitrary direction, for example, the x_2-axis. s input to our system, we require from the user a function which takes as input any possible configuration of the robot and outputs whether it is a valid configuration or not. It should also output the distance that the point is from the boundary along the chosen direction (the x_2-axis in this case). Using these functions we find the extremal points and since S is compact, these points exist in every direction.

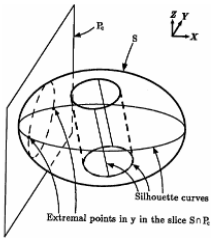

Figure 2: Silhouette Curves traced out by connecting the extremal points

s c varies, the extremal points in the chosen direction are connected to the extremal points of the previous slice and the method traces piece-wise silhouette curves.

2.1 Critical Slices

s long as the number of extremal points in the graph remain consistent, the slicing process continues. However, where the number of extremal points in the slice increases or decreases, the connectivity of the graph changes. In case of an increase in the extremal points the new points are called critical points. In case of a decrease of extremal points, the point where the silhoutte curves meet are called critical points. Slices which pass through these critical points are known as critical slices, and the values c, such that $x_1 = c$ describes the critical slice, are known as the critical values.

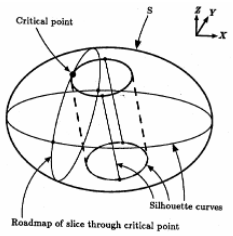

Figure 3: critical point occurs when a new extremal appears

t each of these critical slices, a recursive step is initiated. This procedure treats the intersection of the critical slice and the compact space S as the new compact space, S', and then applies the same method of slicing on S', along a reference axis on the slice. Thus, for an m dimensional space and (m 1) dimensional hyper-space, on recursion a subspace of (m 1) dimensions is described which is the intersection of the subspace S and the slice P_c, ($S \cap P_c$). The method then sweeps across this subspace using a m 2 hyper-space along a third direction, say the x_3-axis and checks for connectivity and critical points.

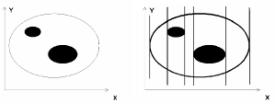

Figure 4: Recursive call at the lowest level

This recursion carries on till either all the critical points in the subspace are connected, or the recursion reaches the lowest level, a 2-dimensional space. t this lowest level, the space is swept across by a line segment and the critical point is simply connected to the closest silhouette curve using a straight line segment.

2.2 ccessibility and Departibility

The term accessibility means that there should be a path from the start configuration q_{start}, of the robot to a valid configuration q', which lies on the silhouette curves.Departibility means that there should be a path from some valid configuration, q'' of the robot which lies on the silhouette curves, to a goal configuration, q_{goal}.

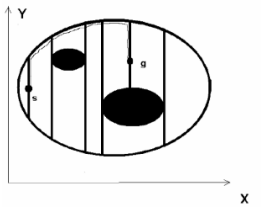

Figure 5: Silhouette path for a 2D space

For providing accessibility and departibility to the system, we make the slices containing the start and the goal configurations part of the set of critical slices, with these two configurations as critical points.

3 Experiments and Calculations

3.1 Test Region

For designing and testing our implementation of the Silhouette method, we have designed and created N-dimensional workspaces described using ellipsoids. Here we discuss the mathematical modeling of these configuration spaces.

The region enclosed by an ellipsoid T, in m dimensions can be described by the equation

$$(X \quad \hat{X})^T * \quad * (X \quad \hat{X}) \leq 1$$

Here $X \in R^m$, $\hat{X}$ describes the center of the ellipsoid and is a positive definite matrix.

In order to create a complex configuration space, we use multiple ellipsoids to describe the space. The first ellipsoid provided as input is used as the primary ellipsoid (PE), that is, the robot configuration is valid only if it is inside or on the primary ellipsoid.

The remaining ellipsoids provided in the input are then used as secondary ellipsoids, (SE) and no configuration that lies inside them is valid. For a secondary ellipsoid, $E, E \in SE$, if it lies completely outside the primary ellipsoid, then it is ignored. If it intersects PE, then it forms a part of the boundary of the compact subspace and if it is completely inside PE, then it is part of the obstacles, $E \subseteq O$.

For the user to be able to view and understand the space and the path described by the robot, we have developed the method to reduce the positive definite matrix for each of the input matrices to view them in 2D and 3D. In each of the images,

primary ellipsoids (PE) are represented by a blue mesh, and secondary ellipsoids (SE) are represented by a red mesh.

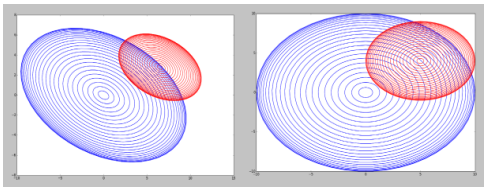

Figure 6: 2D projection onto the x and y axis and the z and w axis of a 4D configuration space

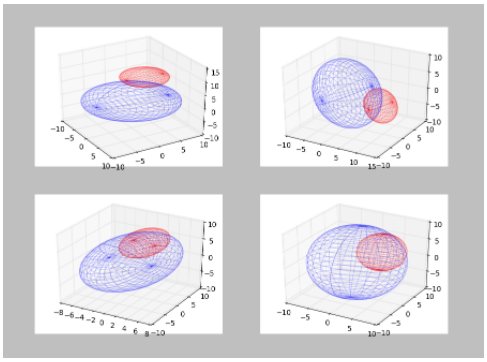

Figure 7: 3D projection onto three of (x,y,z,w) of the same 4D configuration space

The matrix of each ellipsoid is reduced to obtain the projection of the ellipsoids by first identifying vectors normal to which the hyper-plane will be described. Using Gram Schmidt Orthogonalization we generate a basis that will describe the complete plane from which the ellipsoids will be projected.

Let the basis of the plane of dimension n be V and the m vectors normal to the planes be V_{orth} or $p_1 \dots p_m$ and the n m vectors tangential be V_{tang} or $p_{m+1} \dots p_n$ $_m$, then

$$V = [V_{orth} \quad V_{tang}]$$

general equation of a point x in the plane described by vectors $p_1 \dots p_n$ to be projected is $x = \sum_{i=0}^{n}$ $_i p_i + \sum_{i=m}^{n} \beta_i p_i$ and on any ellipsoid it is (using P in place of V_{orth}),

$$[V_{orth} * x \quad \hat{x}]^T \quad [V_{orth} * x \quad \hat{x}] = 1$$

$$[x \quad P^T \hat{x}]^T (P^T \quad P)[x \quad P^T \hat{x}] = 1$$

The Matrix (P^T P) is the required projection matrix onto the plane to be described. Images used in this report have been created using matplotlib, which a python library for 2D and 3D plotting, in order to depict the projection of the workspaces and path generated .

3.2 Implementation Steps

Our system takes as input the number of dimensions, n to be worked on, followed by the number of ellipsoids to be used to create the test region and details of each of these ellipsoid. The first module to run is ProcessInput module which parses the input to generate the positive definite matrix and origin vector for each ellipsoid. The required input format mandates that the primary ellipsoid's (PE) center coincides with the center of the fixed coordinate system and each of it's axes is parallel to one of the axes of the fixed coordinate system.

The next module to run is the RoadMapGeneration module to start the creation of the roadmap (silhouette curves). First the primary axis is chosen arbitrarily as one of the axes of the primary ellipsoid, say the x_1 axis. The range of traversal is subsequently chosen as the end points of the primary ellipsoid along the chosen axis, that is where PE intersects the x_1 axis.

This module utilizes the PlaneEllipseIntersect module. For each of the obstacle ellipsoids, it calculates it's pair of Critical points, the points a along the traversal axis, where $x_1 = a$ is a tangent plane to the obstacle ellipsoid using the following equation :

$$x = c \pm \frac{1}{a \quad c_1} \quad {}^1e_1. \tag{1}$$

Where is the positive-definite matix of in $R^n \times R^n$, c is the center of the ellipsoid, c_1 is the center along the traversal axis and x is the point of intersection, here the critical point. This equation can derived in the following steps :
The equation of the ellipsoid is $\{E(x) = (x \quad c)^t \quad (x \quad c) = 1\}$ and hyperplane is $\{x_1 = a\}$
The gradient of E is the vector field

$$\nabla E(x) = 2 \quad (x \quad c).$$

The gradient is parallel to the first coordinate direction if and only if there is a real number λ such that

$$2 \quad (x \quad c) = \lambda e_1. \tag{2}$$

If the ellipsoid is tangent to the hyperplane $\{x_1 = a\}$ then the first coordinate of x is a. Such a x is necessarily unique since the ellipsoid is strictly convex.

$$(x \quad c)^t e_1 = a \quad c_1 \tag{3}$$

Substituting (2) into equation of the ellipsoid and comparing with (3) gives us

$$\tfrac{1}{2}\lambda(a \quad c_1) = 1. \tag{4}$$

Comparing (4) with (2) leads the conclusion, (1):

$$x \quad c = \frac{1}{2}\lambda \quad {}^1e_1 = \frac{1}{a \quad c_1} \quad {}^1e_1.$$

Following the calculation of all the Critical points, the traversal of the ellipsoid begins. In the highest recursion level the traversal begins from minimum of the two end

points of the primary ellipsoid along the traversal axis. t regular intervals along the traversal axis, we obtain the slice and apply the PlaneEllipseBoundary module. This modules checks if an obstacle intersects with the plane and calculates the boundary points along the chosen arbitrary axis lying on the plane.

For example, at the highest recursion level with x_1 as the traversal axis, $x_1 = a$ will be the slice, c is the center of the ellipsoid and is it's positive definite matrix and along one of the x_i axis of the dimensional plane, such that i 1, say the x_2 axis, the Intersect module will solve the equation for the vector of dimension equal to the dimensions of the plane, $X = [a, x_2, 0 \dots 0]$:

$$(X \quad c)^T * \quad * (X \quad c) = 1$$

t each interval, which is at a regular distance, the LinkToRoadmap module connects the intersection points obtained from the above module to the points and curves obtained from the previous slice. s the traversal proceeds we check if there are any critical points between the present slice and the next. In case there are none, we proceed in this manner till the end point.

If there are Critical Points between the two slices, it implies that there is at least one Critical Slice between the two slices. The system then works on each Critical Slice recursively along with run the PlaneEllipseBoundary and LinkToRoadmap. The system constructs the intersection of the present configuration space C and the slice, $S : x_1 = a$, to obtain workspace formed by the intersection W', $W' = W \cap S$. We utilize the ReduceEllipsoids module to obtain the intersection of each of the ellipsoids and the slice, that is, it returns the origin and the positive definite matrix of the ellipsoid in the intersection space. Taking $\in R^n \times R^n$, it returns the $B \in R^{n\ 1} \times R^{n\ 1}$.

n ellipsoid needs to be reduced when the plane intersects it at more than one point, that is, it is not the critical slice for that ellipsoid. By definition, the primary ellipsoid always needs to be reduced, unless the critical points intersect with its start or end point. The reduction of the ellipsoid proceeds as follows,

$$(x \quad c)^T * \quad * (x \quad c) = 1$$

$$\begin{bmatrix} a & c_1, q & \hat{c} \end{bmatrix}^T * \begin{bmatrix} a_{11}, \hat{a_1}^T \\ \hat{a_1}^T, \hat{} \end{bmatrix} * \begin{bmatrix} a & c_1 \\ q & \hat{c} \end{bmatrix} = 1 \tag{5}$$

Where x is a point in the configuration space of n dimensions, c is the center of the ellipsoid, is it's semi-definite matrix, q is a point in the lower dimension (n 1), c_1 is the center of the ellipsoid along the x_1 axis, $\hat{c}$ is the list of center points excluding the center at the x_1 axis, equation of the plane is $x_1 = a$, a_{11} is the corner most point of and $\hat{a_1}$ is the list of values in the first row of , excluding the first point. (5) leads to the following.

$$q^T \hat{} q \quad 2\hat{c}^T \hat{} q + 2(a \quad c_1)\hat{a_1}^T q = 1 \quad a_{11}(a \quad c_1)^2 \quad \hat{c}^T \hat{} \hat{c} + 2(a \quad c_1)\hat{a}_1^T \hat{c} \tag{6}$$

Now, we wanted an the resulting ellipsoid formed by the Intersection, for which the equation was:

$$(q \quad d)^T * B * (q \quad d) = 1$$

$$q^T Bq \quad 2d^T Bq = 1 \quad d^T Bd \tag{7}$$

Now, the equations (6) and (7) are for the same ellipsoid and by matching terms we obtain the following, which we solve to get B:

$$B = \frac{1 \quad d^T Bd}{1 \quad a_{11}(a \quad c_1)^2 \quad \hat{c}^T \hat{c} + 2(a \quad c_1)\hat{a}_1^T \hat{c}} \hat{} \tag{1}$$

$$B^T d = \frac{1 \quad d^T Bd}{1 \quad a_{11}(a \quad c_1)^2 \quad \hat{c}^T \hat{c} + 2(a \quad c_1)\hat{a}_1^T \hat{c}} (\hat{}^T \hat{c} \quad (a \quad c_1)\hat{a}_1) \tag{2}$$

The second module it uses is to calculate the center of the ellipsoid in the new intersection plane. The line parallel to which the boundary of an obstacle ellipsoid is traced also connects its pair of critical points. When a slice $x_1 = a$ intersects with the ellipsoid, its center in the intersection plane lies on this line. When x_i^{first} and x_i^{second} are the value of the i^{th} axis for the first and the second critical points:

$$x_i^{center} = x_i^{first} + \frac{a \quad x_1^{first}}{x_1^{second} \quad x_1^{first}} \times (x_i^{second} \quad x_i^{first})$$

In the recursive stage the silhouette method proceeds in which the intersection of the primary ellipsoid and the plane forms the primary ellipsoid for recursion and similarly for the obstacles. The traversal axis chosen is the next axis of the primary ellipse and the other steps described above follow. The base case of this recursion is when the space being traversed is 2 dimensional. t this level, the critical points found in the traversal of the 2D space and the ones passed down from higher spaces are connected by straight lines to the closest points on the previous slice. This helps ensure that the path does not cut across an obstacle.

Finally the connections returned from the recursion are linked together with the silhouette curves at the calling level to form the roadmap (graph). We run Djikstra's algorithm on the adjacency list obtained from this graph to describe a path between the start and the end configurations.

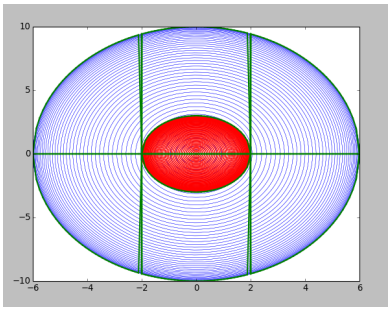

Figure 8: Output path for a 3D configuration space formed along one standard axis

4 Result

We tested our system using test configurations in two, three and four dimensions. We will start with the 2D test cases with one and two obstacles. The roadmap generated in shown in figure [8].

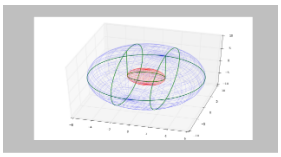

Figure 9: Roadmap generated for one obstacle

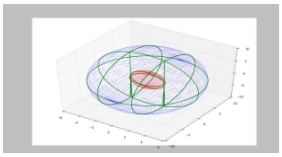

Figure 10: Roadmap generated for one obstacle along all axes

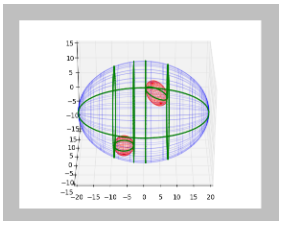

Figure 11: Roadmap generated for two obstacles

Testing on 3 Dimensions, we present the Roadmap generated for one obstacle followed by the roadmap for multiple obstacles. The roadmap created by the method fulfills the rules required for accessibility and departibility as shown [16].

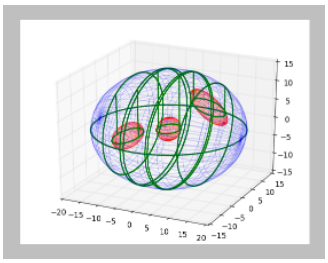

Figure 12: Three obstacles - 3D view

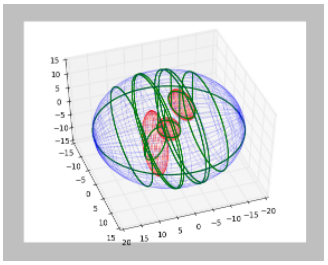

Figure 13: Three obstacles - 3D view

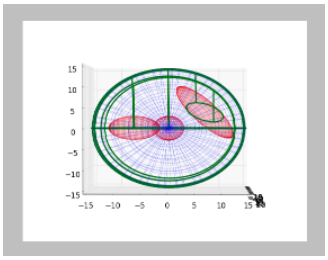

Figure 14: Three obstacles - view along the X&Y axis

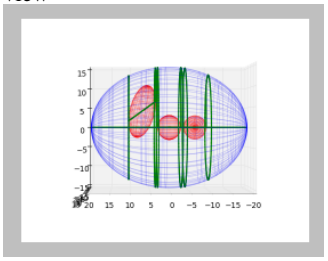

Figure 15: Three obstacles - view along the Y&Z axis

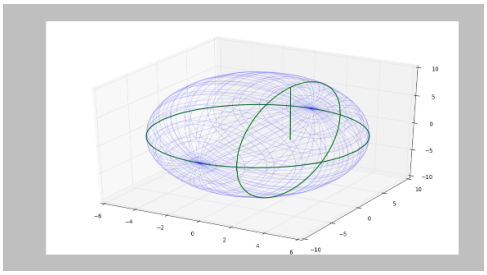

Figure 16: Output path for a 3D configuration space when provided start/end point

The figure [17] is the 2D projections of a roadmap formed for a 3D space. The first two images show the roadmap (green) closely following the boundary of the primary ellipse (blue) and the secondary ellipse. Taking the 3D space to be described by the x, y and z axis, the first two images are projections along the y and z axes. The third image which is a projection along the x-axis is more interesting. The green path does not seem to be along the boundaries of the respective ellipses which is because of the formation of the path in the recursive layer. The path does indeed follow the outer boundary but it does so at di erent intervals and the path formed at those intervals is the projection shown.

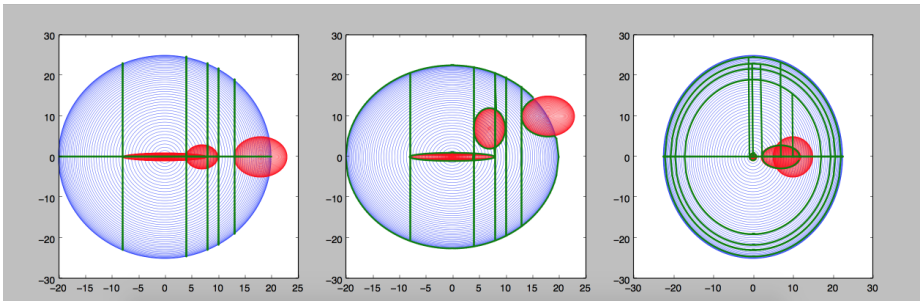

Figure 17: 2D one obstacle

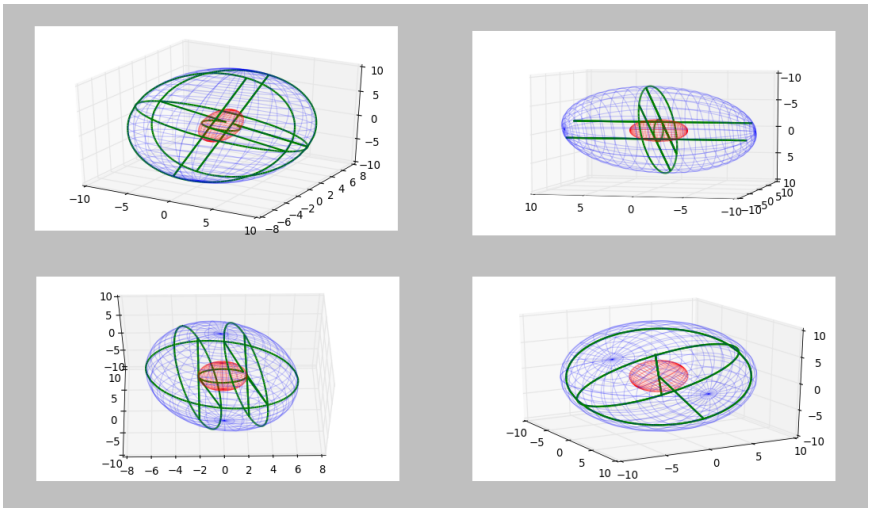

Figure 18: 3D projection of path onto three of (x,y,z,w) of the same 4D configuration space

The output of the model when applied on a 4-dimensional input space is shown in figure [20]. Each image shown is the projection of the model along the three chosen axes of the space. These axes can be chosen in 4 di erent ways and the projection of the space along with the path is shown along each. This final output underscores the result of this project to be able to find a path any arbitrary dimensional space if it exists.

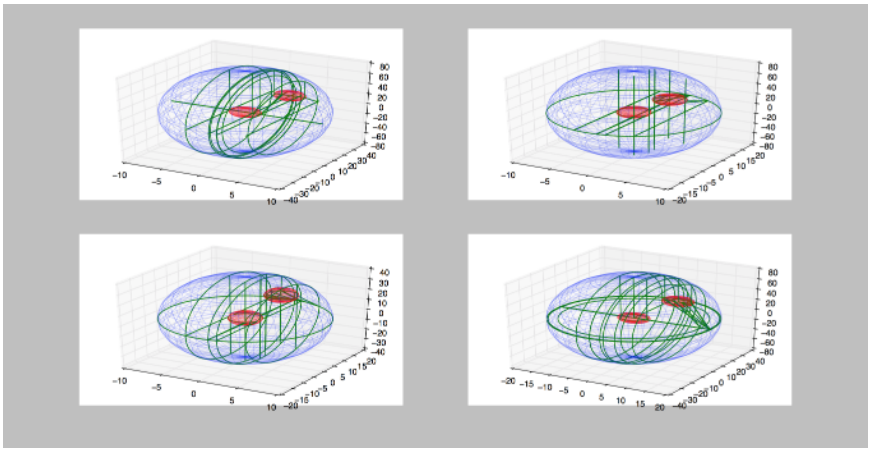

Figure 19: 3D projection of path onto three of (x,y,z,w) of the same 4D configuration space - multiple obstacles

The final path obtained using the Dikstra's algorithm applied on the roadmap from the start configuration of the arena to the goal configuration specified close to the only obstacle present in the arena is shown.

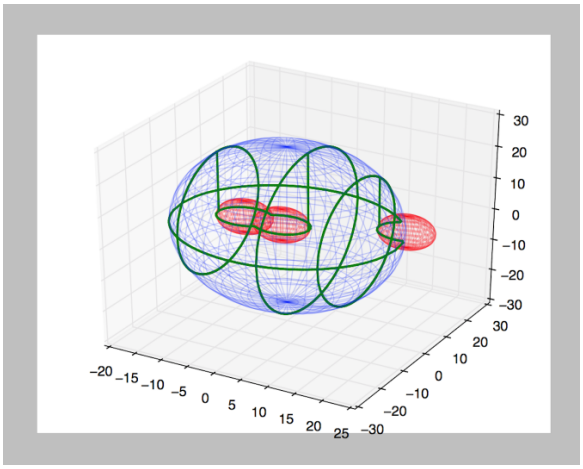

Figure 20: 3D configuration space with intersecting obstacles

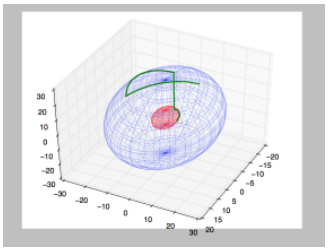

Figure 21: Path from Start to Goal configuration

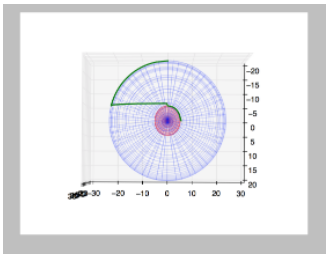

Figure 22: Path from Start to Goal configuration - Top View

5 Conclusion

In this report we have presented a method used for the implementation of the Silhouette method for Robot Motion Planning. The Silhouette method is a roadmap method for robot motion planning which is complete, which means it will find a valid path if any exists. The report initially discusses the concept of the Silhouette method followed by our implementation. The step by step mechanism followed was reviewed. The configuration space we have generated to test the Silhouette method is using ellipsoids described in any arbitrary N dimension. The advantage of using an ellipsoid space is the capability to quickly and easily model obstacles in the ellipsoidal space. Projection of these regions has been discussed, which is utilized when the configuration space is in 3 or greater dimensions. In the result section, the output images for paths obtained in 2, 3 and finally 4 dimensional configuration space are shown. The 4 dimensional output demonstrates the applicability of the Silhouette method in arbitrary dimensional configuration space. Djikstra's algorithm has been applied to find a path between the start and goal configuration, on the obtained roadmap.

References

[1] neeta Bhattacharyya, Ekta Singla and Bhaskar Dasgupta *Robot path planning using silhouette method*, *13th National Conference on Mechanisms and Machines (NaCoMM07), 2007*

[2] C. F. Canny *The Complexity of Robot Motion Planning*, *MIT Press, Cambridge, M , 1988*

[3] J.C. Latombe, *Robot Motion Planning*, *Boston, M : Kluwer cademic Publishers, 1991*

[4] Howie Choset, *Robotic Motion Planning: Roadmap Methods*