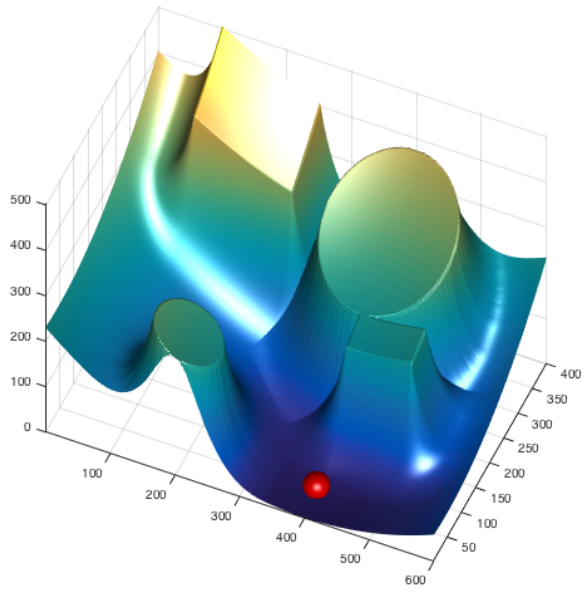
1 Introduction

In this assignment you will be developing code to guide a robot from one location to another in a 2 dimensional configuration space using artificial potential fields. The figure below depicts a plot of the energy surface associated with our sample environment and the state of the robot is modeled by the red sphere which we can think of as rolling down the energy surface towards the goal location.



2 Assignment: Gradient Based Planner [15 points]

Your job is to complete the function named `GradientBasedPlanner` which is used to control the motion of the robot. The signature of the function is given below.

```
1 function route = GradientBasedPlanner (f, start_coords, end_coords, max_its)
```

The input arguments to this function are explained below.

`f` : A 2D array containing the potential function values.

`start_coords` : An array specifying the coordinates of the start location the first entry is the x coordinate and the second the y

`end_coords` : An array specifying the coordinates of the goal the first entry is the x coordinate and the second the y

`max_its` : The maximum number of iterations that should be tried by the planner

The section of code that you are asked to complete should perform the following procedure:

1. On every iteration the planner should update the position of the robot based on the gradient values contained in the arrays `gx` and `gy`.

2. Update the route by adding the new position of the robot to the end of the route array. Note that the distance between successive locations in the route should not be greater than 1.0.

3. Continue the same procedure until the distance between the robot's current position and the goal is less than 2.0 or the number of iterations exceeds the value contained in `max_its`.

The output of this function, `route`, should be an array with 2 columns which depicts how the position of the robot evolves over the state of the simulation. The first column should correspond to the x coordinate and the second to the y coordinate, the first row should correspond to the initial position of the robot at the start and the last row to the location where it ends up.

Once you have written this function you can run the `GradientBasedPlanner` script to test it. You should experiment by changing the positions of the obstacles and the start positions. Note that the procedure will not always work. For some configurations the robot may get stuck in a local minima, for others the gradient update procedure may lead the robot off the workspace though this second problem could be mitigated by adding an obstacle that rings the boundary of the workspace.

2.1 Files in this Assignment

`GradientBasedPlanner.m` - The function used to guide the robot from a given start location to the goal based on the gradient of the function, f

`PotentialFieldScript.m` [*] - The script to set up obstacles in a simulated environment and construct an artificial potential function over the environment.

`evaluate.p` - Code that evaluates your implemented functions

`submit.m` - Script which calls the `evaluate` function for generating submission result

* indicates the files you will need to complete

2.2 Improving the view

In order to improve the look of your 3D plots you should select the Camera Toolbar entry under the View menu on each 3D plot. From this camera toolbar you can turn on lighting by clicking on the lightbulb icon and adjust your view on the plot by selecting the Orbit Camera icon.

2.3 Submission and Grading

To submit your result to our server, you need to run the command `submit` in your MATLAB command window. A script will then evaluate your `GradientBasedPlanner` and generate output file (`GradientBasedPlanner.mat`) to be uploaded to the Coursera web UI. If you pass our test cases, you will get the full score in this assignment. You may submit your result multiple times, and we will count only the highest score towards your grade.

Part	**Submitted File**	**Points**
Potential Gradient Based Planner	`GradientBasedPlanner.mat`	15