

**ECE 5397/6397: Intro to Robotics**

**HW 6, Due Apr 28** Path planning

The path-planning method called Artificial Potential Fields uses gradient descent search to find a collision-free path to a goal configuration.

We provide the function PotentialFieldNavigation.m at

<https://github.com/UH-ECE6397/Assignments/blob/master/PotentialFieldNavigation.m>

% A path planner for an n-link planar robot arm moving among polygonal obstacles.

%

% Based off chapter 5.2 in "Robot Modeling and Control" by Spong, Hutchinson,

% and Vidyasagar

%

% Aaron T Becker, 04-13-2016, atbecker@uh.edu

%

Task 1 (5pts) repulsion from point obstacle: There are point obstacles that represent cutting lasers which must be avoided. Implement the function

function Fvec = frepPt(q, pObstacle, eta, rhoNot)

%computes the forces that repel each DH frame origin from a point

%at positon pObstacle, given by equation 5.6 & 5.7 in RD&C

%q: configuration of the arm

%pObstacle: xy position of the point obstacle

%eta: vector parameter that scales the forces for each degree-of-freedom

%rhoNot: vector that defines the distance of influence of the obstacle

Task 2 (Graduate students 5pts, Undergrads, 5pts E.C.): add a set of floating repulsive control points (one per link)

function Fvec = frepFloatingPt(q, pObstacle, eta, rhoNot)

% computes the forces that repel a point on the link that is closest to any workspace obstacle

% at positon pObstacle, given by equation 5.6 & 5.7 in RD&C

% q: configuration of the arm

% pObstacle: xy position of the point obstacle

% eta: vector parameter that scales the forces for each degree-of-freedom

% rhoNot: vector that defines the distance of influence of the obstacle

Task 3 (5pts) detect a local minimum: using the code on page 181, check if the movments for last three iterations are all less than **epsilon\_min**. If so, set variable **inLocalMinimum** to be true.

Find a value of **epsilon\_min** that will detect if the robot is stuck

**epsilon\_min = \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**

Task 4 (5pts) random walk: if **inLocalMinimum** , then perform a random walk (page 181). Select a **t** value that works well, given the value v that specifies the maximum size of each step.

t = 5; %how many random steps to take?

v = pi/10; % maximum random value at each step

Task 5 (5pts E.C.) polygonal obstacles: Change the lasers to exploding polygonal containers of dynamite that must be avoided. Change frepFloatingPt(q, pObstacle, eta, rhoNot) to compute the closest point on the polygonal obstacle.