README

Ansuhree Singh

To check for any test conditions , run the file pathtestMain directly.

- For Training the I made the following functions:
    - Pathplanning.m : I made this file to crop the main map into smaller maps, resized it. The rest of the file trains the path using the following functions:
        1. read_files.m – function that reads the cropped image that I want to train on.
        2. getFeatures.m – function that is passed with the map (image) and it returns 10 features as per my algorithm that are binary images.
        3. getCostNOW.m – takes in the features and the weight vector to give the cost, (this is the initial raw cost), just for debugging.
        4. gradDescent.m – this function take sin the image, features and the weight, and gives out the final cost map and the cost to go. Also it plots the planned path. This function calls the following functions:
            - getMapCellsFromRay .m– mex file for Bresenham's implementation to get points from line drawing.
            - getCostNOW.m – already explained above.
            - Dijkstra_matrix.m – mex file for calculating ctg.
            - Dijkstra_path .m– mex file to calculate the optimum path.
- After the optimum weights are calculated, for testing , following functions are used:
    - PathtestMain.m : This function reads the main map, resized to ½ half for speed. Loads the features and the optimum weights for the pedestrian path and the car path and shows the optimum output path on the map. It calls the following functions:
        1. getCostNOW .m – same as explained above.
        2. Read_files.m – to read the main map.
        3. getampNOW .m –This function is called twice, to get the paths for pedestrian and car. function that takes in the image, feature, weights and also a variable k, that decides pedestrian path or car path is being calculated and two variables x, y (0,0 in the first case) the ginput , gives out final cost and cost to go.