Assignment 3, Due Dec. 7th

(Worth 15%)

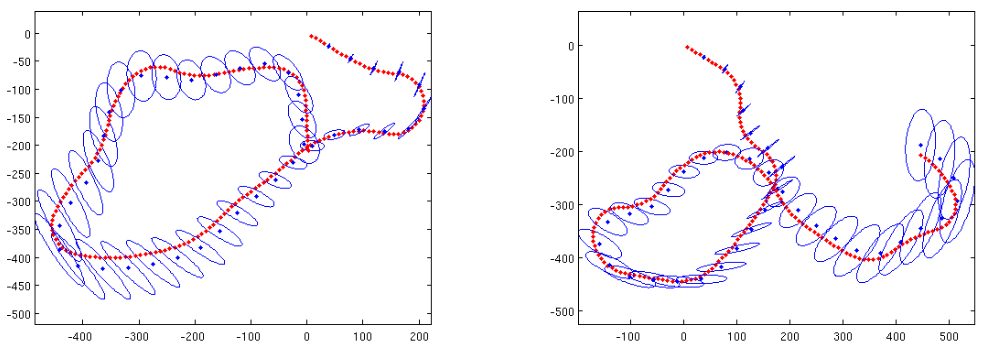
November 20, 2009

1 Question 1: Extended Kalman Filter

Weight 30%

Using the code provided implement the Propagate Function for an Extended Kalman Filter using the equations presented in class. The robot moves with constant velocity v and at regular intervals the angular velocity changes. See Fig. 1 for sample trajectories.

Plot sample trajectories for different values of noise.



(a) (b)

Figure 1: Sample trajectories using propagation only.

Keep in mind there is not update in this implementation.

2 Question 2: Exploration of an unknown environment using the Generalized Voronoi Graph

Weight 70% In assignment 2 you implemented a controller for following different types of trajectories, including, a path in free space; following a wall; and following the edges of the Voronoi Graph. Sample code for wall-following is provided as a guideline. Implement the GVG exploration algorithm we have discussed in class.

First implement and test independently the following functions:

- AccessGVG: this function moves the robot away from the closest obstacle until the robot is equidistant from two obstacles. In order to compensate for the blind range of the sensor (behind the robot) keep in memory the discovered closest obstacle. **(From Assignment 2).**
- OrientAlongEdge: rotate the robot to face along the Voronoi edge.
- FollowEdge: this function guides the robot on a path equidistant to two obstacles.**(From Assignment 2).**
- DetectMeetPoint: this function detects when the robot approaches a location which is equidistant to more than two obstacles. In addition to the assignment 2 implementation if a meetpoint is detected you should add it as a vertex in the Voronoi Graph (use Boost), unless you have already visited that vertex (use the odometry to verify). The inserted vertex should have degree equal to the number of edges that leave from this vertex.
- SelectEdge: If there is an unexplored edge, select it. Otherwise, traverse the GVG (using the graph representation) until the robot reaches a meetpoint (vertex) with unexplored edges. If there are no unexplored edges the algorithm terminates.
- DetectEdgeEnd: when the robot is following an edge and the closest distance drops below a threshold, then this edge can be terminated with a meetpoint (vertex) of degree 1.

It is strongly recommended to use the Boost Graph Library for the implementation of the Voronoi Graph. The BGL can be downloaded from:
`http://www.boost.org/doc/libs/1_41_0/libs/graph/doc/index.html`

Create a program that performs GVG exploration of an unknown environment using the above described functions. Present results from the `cave.cfg`, `autolab.cfg`, and `hospital1.cfg` configuration files.