MAE 145: Intro Robot Planning and Estimation
Extra Credit Problem 1

Due on March 18, 12:00pm

Notice:

For this assigment, you just have to update a python code. All extra credit programming problems (all combined) in the course will count up to an additional 5% of the grade in the course.

Discrete Bayes implementation:

In this exercise you will be implementing a discrete Bayes filter accounting for the motion of a robot on a 1-D constrained world. Assume that the robot lives in a world with 20 cells and is positioned on the 10th cell. The world is bounded, so the robot cannot move to outside of the specified area. Assume further that at each time step the robot can execute either a move forward or a move backward command. Unfortunately, the motion of the robot is subject to error, so if the robot executes an action it will sometimes fail. When the robot moves forward we know that the following might happen:

1. With a 25% chance the robot will not move,
2. With a 50% chance the robot will move to the next cell,
3. With a 25% chance the robot will move two cells forward,
4. There is a 0% chance of the robot either moving in the wrong direction or more than two cells forwards

Assume the same model also when moving backward, just in the opposite direction. Since the robot is living on a bounded world it is constrained by its limits, this changes the motion probabilities on the boundary cells, namely:

1. If the robot is located at the last cell and tries to move forward, it will stay at the same cell with a chance of 100%
2. If the robot is located at the second to last cell and tries to move forward, it will stay at the same cell with a chance of 25%, while it will move to the next cell with a chance of 75%.

Again, assume the same model when moving backward, just in the opposite direction. Implement in Python a discrete Bayes filter and estimate the final belief on the position of the robot after having executed 9 consecutive move forward commands and 3 consecutive move backward commands. Plot the resulting belief on the position of the robot.

Hint: Start from an initial belief of: `bel = numpy.hstack ((numpy.zeros(10), 1, numpy.zeros(9)))` You can check your implementation by noting that the belief needs to sum to one (within a very small error, due to the limited precision of the computer). Be careful about the bounds in the world, those need to be handled ad-hoc.