# Homework 010

# Submission Instructions

The problems involve writing Python code. Please annotate your code with appropriate comments. You don’t have to comment every line, but at least comment major sections.

Please include the answers (graphs, explanations, page scans, etc) in a single PDF file and attach the code in a separate zip file. So one PDF file and one zip file for the code. This will streamline the grading process for the TA’s because they can just leave comments on the PDF.

# Problems

The file RBE500-F17-100ms-constant-vel.csv contains Lidar range data values, z’s , in centimeters, measured every 100 msec. The sensor is on a platform that is moving in a straight line at approximately constant velocity towards a wall. Consider the origin of coordinates to be the wall. Twice during the motion, the sensor reading jumps since it momentarily points at an object that is closer than the target wall. These jumps are real – not bad data.

* 1. Create a Kalman filter with the state and a Gaussian plant noise model with covariance matrix of where and, of course, . Update the process state and error covariance every 0.1 sec. Assume the Lidar measurement noise is Gaussian with a variance . Assume that the initial estimate of position is 2530 cm from the wall and that the initial estimate of velocity is 10 cm/sec toward the wall, and that the initial covariance of position and velocity is . For this first part, run the Kalman filter as if all of the data were valid measurements of distance to the wall. (In this case, since the sensor is moving on a straight line toward the wall, ray casting is equivalent just estimating the position relative to the wall.)

Plot the raw data and the estimated position on the same graph with time in seconds on the x axis and distance in cm on the y axis. On a separate graph plot the estimated velocity in cm/sec on the y axis and time in seconds on the x axis. Also plot the standard deviations of the position and velocity standard deviations and their correlation coefficient versus time in a similar manner. Be sure to label the graphs and their axes (including units). **[3 points]**

* 1. Now rerun the Kalman filter trying to detect and eliminate the anomalous data by comparing its computed innovation , where , to the expected innovation covariance . Perform the comparison by computing the squared deviation (since everything is scalar in this equation). If , corresponding to a point, disregard the data and do not perform an update. Produce the same type of graphs as in section a). **[3 points]**
  2. Next rerun the Kalman filter, trying to detect and eliminate the anomalous data by using a mixture model for the errors in the Lidar data. For this problem, there are no out of range points and we can assume that there is no random noise component. Therefore, there is only a Gaussian component centered at the expected range and an exponential component. Therefore, assume the model takes the form , where the parameters of the model are . Use Bayes’ rule to determine the most likely component that generated the data. If it is the exponential component, disregard the data and do not perform the update. Produce the same type of graphs as in section a). **[3 points]**
  3. Compare the performance of the three approaches. Were any anomalous points processed by mistake? Were any valid points rejected? How can each of these methods fail? **[1 point]**