Fall 2024 – 16-642 Manipulation, Estimation, and Control
Problem Set 3

Due: Wednesday 6 November 2024

GUIDELINES:

- You must *neatly* write up your solutions and submit all required material electronically via Canvas by **11:59 pm of the posted due date.**
- Include any MATLAB scripts that were used in finding the solutions. Also include any plots generated with MATLAB as a part of the question.
- You are encouraged to work with other students in the class, however you must turn in your own *unique* solution.
- Late Policy: You are given **in total 72 hours** of grace period for **ALL** homework across the semester, which you can use to give yourself extra time without penalty. Late work handed in when you have run out of grace will not be accepted.

1. (50 points total) In this problem, you will implement an Extended Kalman filter pose estimate for the differential drive robot. The configuration of the robot can be represented by the vector

$$q = \begin{bmatrix} x_r \\ y_r \\ \theta \end{bmatrix}$$

where (x_r, y_r) is the 2D position in meters of the point midway between the two wheels, and θ is the angle in radians of the body relative to the x-axis. The input vector is

$$u = \begin{bmatrix} v_f \\ \omega \end{bmatrix}$$

where v_f is the forward speed of the body in meters per second, and ω is the angular velocity of the body in radians per second. The kinematic discrete time equation of motion is

$$q[k+1] = \begin{bmatrix} q_1[k] + T(u_1[k] + v_1[k]) \cos q_3[k] \\ q_2[k] + T(u_1[k] + v_1[k]) \sin q_3[k] \\ q_3[k] + T(u_2[k] + v_2[k]) \end{bmatrix}$$

where $T = 0.01$ seconds is the rate at which the input is applied, and $v[k]$ is assumed to zero-mean white Gaussian noise with constant covariance matrix $V \in \mathbb{R}^{2\times 2}$.

The robot receives GPS measurements once every 10 timesteps, i.e., the first y corresponds to $x[10]$, the second y corresponds to $x[20]$, and so on. The output equation is

$$y[k] = \begin{bmatrix} q_1[k] \\ q_2[k] \end{bmatrix} + w[k]$$

where $w[k]$ is assumed to be zero-mean white Gaussian noise with constant covariance matrix $W \in \mathbb{R}^{2\times 2}$.

(a) (10 points) Find the linearized approximation of the system as a function of the current state. In particular, your approximation should look like

$$q[k+1] \approx F(q[k], u[k]) \cdot q[k] + G(q[k]) \cdot u[k] + \Gamma(q[k]) \cdot v[k]$$

where $F(q[k], u[k])$ is a 3×3 matrix, and $G(q[k])$ and $\Gamma(q[k])$ are both 3×2.

(b) (15 points) The file `calibration.mat` contains data generated during a run where an expensive ground truth system was available that could measure the full state of the robot. This data can be used to estimate the noise parameters, namely the noise on the input and on the GPS. The variables in calibration.mat are:

- `t_groundtruth`: a vector of times associated with the input signals.
- `q_groundtruth`: a matrix whose columns are the robot states at the times associated the times in `t_groundtruth`. These states are assumed to be perfect.
- `u`: a matrix whose columns are the robot inputs at the times associated with `t_groundtruth`. This signal is the signal applied to the robot, it does not include the noise terms in $v[k]$.
- `t_y`: a vector of times associated with the GPS measurement.
- `y`: a matrix whole columns are the noisy GPS measurements taken at the times in `t_y`.

Using this information, estimate the process covariance V and measurement covariance W. You may want to start with W since computing it is the simpler of the two. Your solution should include an explanation of how you did the computations, your MATLAB code, and the computed values of V and W.

(c) (25 points) The file `kfData.mat` contains another data set that you can use to demonstrate the effectiveness of a Kalman filter. The variables it contains are:

- `t`: a vector of times associated with the input signals.
- `u`: a matrix whose columns are the robot inputs at the times associated with `t`. This signal is the signal applied to the robot, it does not include the noise terms in $v[k]$.
- `t_y`: a vector of times associated with the GPS measurement.
- `y`: a matrix whole columns are the noisy GPS measurements taken at the times in `t_y`.
- `q_groundtruth`: a matrix whose columns are the robot states at the times associated the times in `t`. These states are assumed to be perfect.
  NOTE: this should **NOT** be used to help you solve the problem. It is meant for **comparison only**.

Note that the noise $v[k]$ is a part of the input, not the state model. This means there are variables factoring into how the process noise covariance affects the state, similar to how there are variables factoring into how the state covariance changes.

Use an initial estimate of

$$\hat{q}[1] = \begin{bmatrix} 0.355 \\ -1.590 \\ 0.682 \end{bmatrix}$$

and assume that the initial covariance matrix is

$$P[1|1] = \begin{bmatrix} 25 & 0 & 0 \\ 0 & 25 & 0 \\ 0 & 0 & 0.154 \end{bmatrix}$$

to write a full Extended Kalman Filter. Submit your code and make a plot that contains a trajectory plot of y_r vs. x_r for the ground truth, a scatter plot (y_r vs. x_r) from the GPS measurements, and a trajectory plot of y_r vs. x_r for your EKF.

2. (35 points total) The objective of this problem is to write a particle filter that estimates the pose of a differential drive robot using measurements from two range beacons. The motion model is the same as for the Kalman filter problem (Equation 1), though the timestep T may be different. $v[k]$ is zero-mean Gaussian noise with unknown covariance. At every timestep, the robot receives noisy range measurements to two fixed beacons that are located at

$$b_1 = \begin{bmatrix} x_{b1} \\ y_{b1} \end{bmatrix} = \begin{bmatrix} 5 \\ 5 \end{bmatrix}, b_2 = \begin{bmatrix} x_{b2} \\ y_{b2} \end{bmatrix} = \begin{bmatrix} 15 \\ 5 \end{bmatrix},$$

and the measurement model is then given by

$$y[k] = \begin{bmatrix} \sqrt{(q_1[k] - x_{b1})^2 + (q_2[k] - y_{b1})^2} \\ \sqrt{(q_1[k] - x_{b2})^2 + (q_2[k] - y_{b2})^2} \end{bmatrix} + w[k] ,$$

where $w[k]$ is zero-mean Gaussian noise with unknown covariance.

The file `pfData.mat` contains another data set that you can use to demonstrate your particle filter. The variables it contains are

- `t`: a vector of times associated with the input signals.
- `u`: a matrix whose columns are the robot inputs at the times associated with `t`. This signal is the signal applied to the robot, it does not include the noise terms in $v[k]$.
- `y`: a matrix whole columns are the noisy GPS measurements taken at every timestep. $y_1[k]$ is the range to b_1 at time k, $y_2[k]$ is the range to b_2.
- `q_groundtruth`: a matrix whose columns are the robot states at the times associated the times in `t`. These states are assumed to be perfect.
  NOTE: this should **NOT** be used to help you solve the problem. It is meant for **comparison only**.

A template you can use to structure your particle filter along with some useful functions for visualizing the particle cloud and robot location is in `pfTemplate.m`. Your job is to modify this code to add a particle filter that visualizes the results at every timestep. Your particle filter should have a constant number of particles and resample them at every step. The particle filter will have several parameters which you will need to tune to get rasonable performance, in particular:

- number of particles in the cloud
- covariance of particle filter process noise
- covariance of paritcle filter measurement noise

At each timestep, you should make a plot containing the following:

- Beacon locations
- Robot ground truth pose
- Robot ground truth trajectory
- Particle cloud
- Robot pose estimate (derived from particle cloud)

Submit a write-up detailing what choices you made in creating the filter, your code, and a `.mp4` movie showing the plots created at every timestep. **Your grade is dependent on your code and your mp4 movie showing how the particles start with an initial uniform distribution which then converges sufficiently close to the ground truth trajectory.**

3. (15 points total) A team of engineers is developing an autonomous underwater robot designed to explore coral reefs and map dangerous underwater terrain. The robot is equipped with a sensor array to help it navigate through narrow reef channels and avoid harmful underwater structures like jagged rocks or sunken debris. Additionally, the robot must be able to detect and avoid strong currents that could sweep it off course or damage it.

   In this environment:
   - 35% of the area contains hazardous underwater structures, like rocks and debris, that the robot needs to avoid.
   - 15% of the area has strong currents that could displace or damage the robot.
   Note: For simplicity, these parts are mutually exclusive, i.e. there is never an area that contains both hazardous structures and strong currents.

The robot's sensor array consists of two main sensors:
Sonar Sensor: This sensor detects hazardous underwater structures. It has a 90% chance of accurately detecting an obstacle but has a 2% false alarm rate when no obstacle is present.
Current Flow Sensor: This sensor detects dangerous underwater currents. It has an 45% chance of correctly identifying a current, with a 4% chance of falsely detecting a current when none is present.
These two sensors are completely independent, i.e. they have no knowledge of each other. It is possible for both sensors to send a warning signal at the same time (but we will not know which one triggered it).

(a) (7 points) If the robot receives a warning signal from its sensor array (triggered from either the Sonar or Current Flow sensor, robot does not know which one), what is the probability that the robot is about to encounter a strong underwater current? Hint: The prior distribution of the environment matters.

(b) (4 points) Assuming the robot and environment are static, how many consecutive measurements from the Current Flow Sensor need to be taken to get the same probability of correctly identifying an imminent danger to the robot as the Sonar Sensor. Hint: The prior distribution of the environment does not matter.

(c) (4 points) If the noise model for the Sonar Sensor is $N(0, 7)$, what is the largest zero-mean Gaussian noise model for the Current Flow Sensor based on the number of measurement as determined in part (b) that must be averaged to achieve the same aggregate $N(0, 7)$ noise. Your solution should be $N(0, ?)$ where you are solving for the variance.