Posting Date: Monday Nov. 18th.

You need not hand in anything. Instead, be prepared to answer any of these problems on an upcoming take-home exam. You may discuss your solutions with classmates up until the time that the exam becomes available.

1. The MATLAB template file "c2dnonlinear_temp.m" contains a template for programming a numerical integration function that computes the right-hand side of a discrete-time dynamics model of the form

   $$\underline{x}(k+1) = \underline{f}[k, \underline{x}(k), \underline{u}(k), \underline{v}(k)]$$

   based on numerical integration of a differential equation of the form:

   $$\dot{\underline{x}}(t) = \underline{f}[t, \underline{x}(t), \underline{u}(k), \underline{v}(k)]$$

   from time t_k to time t_{k+1}. It also should be able to compute the partial derivatives of $\underline{f}[k, \underline{x}(k), \underline{u}(k), \underline{v}(k)]$ with respect to $\underline{x}(k)$ and $\underline{v}(k)$, except that pertinent parts of the code have been removed and replaced with question marks "????".

   Fix the code so that it can compute the required partial derivatives.

   Test your code using the supplied test function "fscript_ts01.m". Test the function by numerically integrating from a random initial condition and using a random process noise vector. Integrate over a time span of 3 seconds, and use 120 4th-order Runge-Kutta numerical integration steps. Compare your results with the exact results, which you can computed as outlined in the initial comments section of "fscript_ts01.m". Test the results again, but this time use only 60 4th-order Runge-Kutta steps for the same inputs. Does the error for this second case change as you expect it to change in comparison with the error for the first numerical integration case?

2. Suppose that you are given a nonlinear discrete-time dynamics model of the form:

   $$\underline{x}(k+1) = \underline{f}[k, \underline{x}(k), \underline{u}(k), \underline{v}(k)]$$

   Suppose, also, that this dynamics function has a functional inverse:

   $$\underline{g}[k, \underline{x}(k+1), \underline{u}(k), \underline{v}(k)] = \underline{f}^{-1}[k, \underline{x}(k+1), \underline{u}(k), \underline{v}(k)]$$

   where the inverse is defined such that

   $$\underline{g}\{k, \underline{f}[k, \underline{x}(k), \underline{u}(k), \underline{v}(k)], \underline{u}(k), \underline{v}(k)\} = \underline{x}(k)$$

   Prove that

$$\frac{\partial \underline{g}}{\partial \underline{x}(k+1)} = \left[\frac{\partial \underline{f}}{\partial \underline{x}(k)}\right]^{-1} \quad \text{and} \quad \frac{\partial \underline{g}}{\partial \underline{v}(k)} = -\left[\frac{\partial \underline{f}}{\partial \underline{x}(k)}\right]^{-1} \frac{\partial \underline{f}}{\partial \underline{v}(k)}$$

where each partial derivative is evaluated at appropriate values of the original functions' input arguments.

Hints: Differentiate the equation that defines the inverse. Differentiate it with respect to $\underline{x}(k)$ or with respect to $\underline{v}(k)$, depending on which relationship you are proving. Make sure that you apply the chain rule correctly.

3. Derive an SRIF version of the extended Kalman filter. It should be equivalent to the normal covariance-based extended Kalman filter, but it should keep track of its state and covariance by using components of an information equation, as with the SRIF for linear problems.

   Hints: The SRIF derivation involves use of inverse dynamics to eliminate $\underline{x}(k)$ from its information equation. Afterwards, the equations get manipulated using QR factorizations. Use the inverse of the linearized dynamics equation to eliminate $\underline{x}(k)$ in the extended SRIF. Also, linearize the measurement equation about an appropriate value of $\underline{x}(k+1)$ before you incorporate it into the usual SRIF calculations. Don't forget to re-normalize the measurement equation on each iteration so that its re-normalized measurement error covariance matrix equals the identity.

4. Derive an SRIF version of the iterated extended Kalman filter. It should be equivalent to the normal covariance-based iterated extended Kalman filter, but it should keep track of its state and covariance by using components of an information equation, as with the SRIF for linear problems.

   Hints: The iterated filter re-linearizes its measurement equation about new estimates of $\underline{x}(k+1)$. This can be done in the context of the SRIF's normalized measurement error equation. Certain of the QR-factorization-based calculations will have to be repeated after each re-linearization.

5. Derive an extended nonlinear smoother based on the same principles as the extended Kalman filter. What smoother equations must change? Do this only for the non-square-root covariance smoother that was derived in class. Develop two alternate versions of the calculations of $\underline{x}^*(k)$ as a function of $\underline{x}^*(k+1)$, as was done in class. The first should use the inverse dynamics function $\underline{f}^{-1}[k, \underline{x}(k+1), \underline{u}(k), \underline{v}(k)]$ and $\underline{v}^*(k)$. The second should involve neither inverse dynamics nor $\underline{v}^*(k)$, but it will have to rely on the dynamics linearization that was used during the forward pass of the extended Kalman filter.

6. Derive a Kalman filter for the following system, which effectively has correlated process noise and measurement noise:

$$\underline{x}(k+1) = F(k)\underline{x}(k) + G(k)\underline{u}(k) + \Gamma(k)\underline{v}(k)$$

$$\underline{z}(k) = H(k)\underline{x}(k) + \Lambda(k)\underline{v}(k) + \underline{w}(k)$$

where $v(k) \sim \mathcal{N}[0,Q(k)]$, $w(k) \sim \mathcal{N}[0,R(k)]$, $\mathrm{E}[v(k)w^{\mathrm{T}}(j)] = 0$, $\mathrm{E}[v(k)v^{\mathrm{T}}(j)] = 0$ if $k \neq j$, $\mathrm{E}[w(k)w^{\mathrm{T}}(j)] = 0$ if $k \neq j$. This problem can be solved by using the techniques of Bar-Shalom Section 8.3.1 if one makes appropriate re-definitions of the process noise and the measurement noise, but do not solve the problem in this way. Instead, create augmented vectors

$$\underline{x}_{aug}(k) = \begin{bmatrix} \underline{x}(k) \\ \underline{v}(k) \end{bmatrix} \quad \text{and} \quad \underline{x}_{aug}(k+1) = \begin{bmatrix} \underline{x}(k+1) \\ \underline{v}(k+1) \end{bmatrix}$$

and create an augmented model that involves matrices $F_{aug}(k)$, $G_{aug}(k)$, $\Gamma_{aug}(k)$, $H_{aug}(k)$, $Q_{aug}(k)$, $R_{aug}(k)$, and $P_{aug}(0)$ and the vectors $\underline{x}_{aug}(k)$, $\underline{u}(k)$, $\underline{v}_{aug}(k)$, $\underline{z}(k)$, and $\hat{\underline{x}}_{aug}(0)$ in a standard Kalman filter model form. The correct definition of $\underline{v}_{aug}(k)$ may seem disallowed at first. If one realizes that $\underline{v}(k)$ and $\underline{v}_{aug}(k)$ are never known exactly, but only in terms of their statistical models before any data have been processed, then the model is seen to be allowable.

Define the needed model with its matrices and vectors in terms of the matrices and vectors of the original problem model.

7. Consider the tricycle cart tracking system that is given in Problem 5 of Problem Set 3. In a modified version of the problem the cart is being driven by an opponent, and you do not know how your opponent is commanding the steer angle or the speed. You have an idea that the opposing driver steers and controls the speed in a manner that can be modeled as two independent Markov processes. Thus, your system differential equation model is:

$$\dot{x}_1(t) = -\frac{x_5(t)\,tan[x_4(t)]}{b}$$

$$\dot{x}_2(t) = x_5(t)\,cos[x_1(t)]$$

$$\dot{x}_3(t) = x_5(t)\,sin[x_1(t)]$$

$$\dot{x}_4(t) = -\frac{1}{\tau_{steer}}x_4(t) + \tilde{v}_{steer}(t)$$

$$\dot{x}_5(t) = -\frac{1}{\tau_{speed}}[x_5(t) - \bar{x}_5] + \tilde{v}_{speed}(t)$$

where $x_1 = \psi$ is the heading angle, $x_2 = y_{1r}$ is the east coordinate of the midpoint between the two rear wheels, $x_3 = y_{2r}$ is the north coordinate of the midpoint between the two rear wheels, $x_4 = \beta$ is the steer angle, and $x_5 = v_r$ is the speed of the midpoint between the two rear wheels. The wheel base is b, and the time-constants of the two first-order Markov models are τ_{steer} and τ_{speed}. The value $\bar{x}_5$ is the mean speed at which the opposing driver seems to like to drive. The measurements are range measurements from two radar located at zero north position and at the east positions ℓ_a and ℓ_b. The particular parameter values that apply to the current system are b = 0.1 m, τ_{steer} = 0.25 sec, τ_{speed} = 0.60 sec, $\bar{x}_5$ = 2.1 m/sec, ℓ_a = -1 m, and ℓ_b = +1 m. The radar measurement noise standard deviations are $\sigma_{\rho a} = \sigma_{\rho b}$ = 0.002 m. The continuous-time process noise intensities are $\tilde{q}_{steer}$ = 0.10 rad^2/sec and $\tilde{q}_{speed}$ = 21.25 m^2/sec^3.

Implement an extended Kalman filter for this system and apply it to the measurement data that can be loaded into the MATLAB work space by executing the script "cart_EKF_meas.m".

You will have to generate $\underline{\hat{x}}(0)$ and $P(0)$ by using a heuristic approach. Use the same approach for generating a guess of the cart initial condition as is spelled out in Assignment #3, except for the following changes: Generate your initial state estimate and covariance matrix at the first measurement sample time, i.e., generate $\underline{\hat{x}}(1)$ and $P(1)$. Therefore, step c) of the Assignment #3 initial guess procedure gets modified to omit the $-\underline{v}_r t_1$ term. Step e) gets modified to set $\hat{x}_4(1) = 0$ because a steer angle of zero produces a turn rate of zero. Assuming that you use this initialization approach to calculate $\underline{\hat{x}}(1)$, you can use the following heuristic estimate of $P(1)$:

$$P(1) = \begin{bmatrix} \dfrac{2\sigma_{\rho a}\sigma_{\rho b}}{\|\underline{v}_r\|^2 (t_2 - t_1)^2} & 0 & 0 & 0 & 0 \\ 0 & (\sigma_{\rho a}\sigma_{\rho b}) & 0 & 0 & 0 \\ 0 & 0 & (\sigma_{\rho a}\sigma_{\rho b}) & 0 & 0 \\ 0 & 0 & 0 & \dfrac{\tilde{q}_{steer}\tau_{steer}}{2} & 0 \\ 0 & 0 & 0 & 0 & \dfrac{2\sigma_{\rho a}\sigma_{\rho b}}{(t_2 - t_1)^2} \end{bmatrix}$$

where the magnitudes of the (1,1), (2,2), (3,3), and (5,5) elements use the geometric mean of the variances of the two radar range measurements and are based on the geometry of the heuristic state initialization. The (4,4) element is the steady-state covariance of the steer angle, which comes from solving the Lyapunov equation that is associated with the covariance dynamics of the Markov steer angle model.

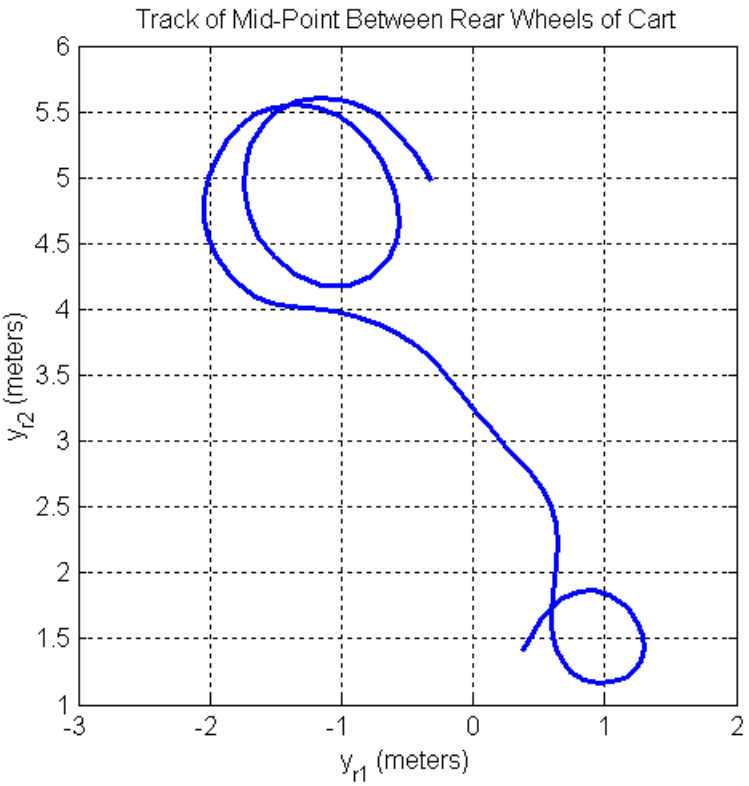
While you are running your Kalman filter also calculate the innovation statistic $\varepsilon_\nu(j)$. If the Kalman filter model is correct and if the EKF linearization assumptions are reasonable, then each $\varepsilon_\nu(j)$ value should be a sample of a Chi-squared distribution of degree 2, and the mean

of these values should be 2 within some tolerance. Check the mean and determine whether the performance of your filter seems to be reasonable by applying some sort of bound test to the mean of the $\varepsilon_\nu(j)$ values.

Be sure to plot your estimated time histories of each state and your estimated ground-track of the mid-point between the cart's rear wheels.

Hints and Help: You should be able to use your code from Problem 1 to transform this continuous-time problem into a discrete-time problem. The required MATLAB continuous-time dynamics function can be developed by filling in the ???? locations in the template file "fscript_cart_temp.m". Likewise, the required MATLAB nonlinear measurement model function can be developed by filling in the ???? locations in the template file "h_cart_temp.m". Be sure to check your derivative calculations in your "fscript_cart.m" and "h_cart.m" functions by using a numerical differentiation test, as was mentioned in lecture earlier in the semester. Do not forget to divide $\widetilde{q}_{steer}$ and $\widetilde{q}_{speed}$ by Δt in order to determine the discrete-time process noise covariances.

The truth ground track of the mid-point between the cart's rear wheels follows the following trajectory in the horizontal plane.



8. Re-do Problem 7 using an Unscented Kalman filter. Initialize your UKF using the same values for $\underline{\hat{x}}(1)$ and $P(1)$ that you used in Problem 7. Compare your results to those from Problem 7.

Also do the following problems from Bar Shalom:

10-3, 10-10