Unknown forcing input u_k

- A common Kalman-filter application is to attempt to localize a stationary or mobile target in some coordinate system.
- The target may be friendly (search and rescue) or hostile (military application), but in neither case do we know the driving input u_k to the target dynamics.
- Therefore, the target-tracking application must treat u_k as a random signal where, arguably, the Gaussian assumption breaks down.
- So, we often rely on very simple models based on the physics of lumped objects.
- The simplified model and unknown u_k cause our position and velocity estimates to be less accurate than they otherwise might be.
- But, covariance still has a geometric interpretation (hyper-ellipsoid) that is helpful in pin-pointing the geographic uncertainty of the object being tracked, and provides hints as to where to point sensors to look for the object at the next time step.

Models used for target tracking

- From Lesson 1.2.6, you learned several generalized model types that are useful for tracking.
- The 2-d nearly constant position (NCP) model is:

$$\begin{aligned} x_{k+1} &= \begin{bmatrix} \xi_{k+1} \\ \eta_{k+1} \end{bmatrix} \\ &= x_k + w_k \\ z_k &= x_k + v_k. \end{aligned}$$

- The 2-d nearly constant velocity (NCV) model is:

$$\begin{aligned} x_{k+1} &= \begin{bmatrix} \xi_{k+1} \\ \dot{\xi}_{k+1} \\ \eta_{k+1} \\ \dot{\eta}_{k+1} \end{bmatrix} = \begin{bmatrix} 1 & \Delta t & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & \Delta t \\ 0 & 0 & 0 & 1 \end{bmatrix} x_k + w_k \\ z_k &= \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix} x_k + v_k. \end{aligned}$$

Models used for target tracking (cont.)

- The 2-d coordinated-turn (CT) model (Ω is rad s^{-1}) is:

$$\begin{aligned} x_{k+1} = \begin{bmatrix} \xi_{k+1} \\ \dot{\xi}_{k+1} \\ \eta_{k+1} \\ \dot{\eta}_{k+1} \end{bmatrix} &= \begin{bmatrix} 1 & \sin(\Omega\Delta t)/\Omega & 0 & (\cos(\Omega\Delta t)-1)/\Omega \\ 0 & \cos(\Omega\Delta t) & 0 & -\sin(\Omega\Delta t) \\ 0 & (1-\cos(\Omega\Delta t))/\Omega & 1 & \sin(\Omega\Delta t)/\Omega \\ 0 & \sin(\Omega\Delta t) & 0 & \cos(\Omega\Delta t) \end{bmatrix} x_k \\ &+ \begin{bmatrix} (1-\cos(\Omega\Delta t))/\Omega^2 & (\sin(\Omega\Delta t)-\Omega\Delta t)/\Omega^2 \\ \sin(\Omega\Delta t)/\Omega & (\cos(\Omega\Delta t)-1)/\Omega \\ (\Omega\Delta t-\sin(\Omega\Delta t))/\Omega^2 & (1-\cos(\Omega\Delta t))/\Omega^2 \\ (1-\cos(\Omega\Delta t))/\Omega & \sin(\Omega\Delta t)/\Omega \end{bmatrix} w_k \\ z_k &= \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix} x_k + v_k. \end{aligned}$$

Extended illustrative example

- In target tracking, KF is often only one solution component.
- Consider an illustrative example: tracking missiles fired at a military aircraft.[1] In each image where a missile is present, we want to compute:
  1. Our best estimate of the missile position in frame k given k observations.
  2. Our best estimate of where the missile position will be in frame $k+1$ given k observations, so we can point the sensor to maintain target lock.
- Assumptions:
  1. We acquire the target when it is still far away using an infrared (camera) sensor.
     - It will appear as a single very bright pixel, perhaps surrounded by several relatively dimmer pixels that are somewhat brighter than the background.
  2. Our camera's frame rate is very fast compared to the kinematics of the target. Thus, the target is capable of only negligible accelerations from frame to frame.

[1] Adapted from J.P. Havlicek's *ECE5283: Kalman Filtering*, Ch. 5, University of Oklahoma, 2003.

An overall target-tracking strategy

- The overall target-tracking strategy will need methods to:
  - Process data from a camera frame to detect pixels or groups of pixels that look like they may indicate targets.
  - There may be multiple targets; since nearby "bright" pixels are probably the same target, we need to average their locations together to make a single observation.
  - We will use separate Kalman filters to estimate the state of each target (one per target) to track their positions based on the measurements.
  - So, we will need ways to associate the set of observations to the set of Kalman filters tracking targets. This introduces several complications:
    - Start new track: What if the observation looks like it comes from a new target?
    - Merge tracks: What do we do if two tracks intersect temporarily or permanently?
    - Coast track: What do we do if the target is obscured?
    - Delete track: What do we do if a target seems to have disappeared?

Detecting exceedances

- When we receive a new image frame from the sensor, we first apply a nonlinear image-processing filter to detect target(s).
- Idea: Near each pixel, formulate a background estimate, which is then subtracted from the pixel.
  - Background estimate $M_{i,j}$ = median of eight nearest neighbors.
  - Filter output $Y_{i,j} = I_{i,j} - M_{i,j}$.
  - Note: (x,y) indexing, not (*row*,*col*) indexing.

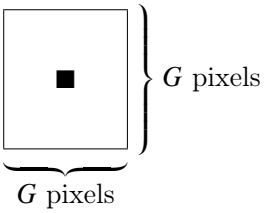
$I_{i-1,j+1}$	$I_{i,j+1}$	$I_{i+1,j+1}$
$I_{i-1,j}$	$I_{i,j}$	$I_{i+1,j}$
$I_{i-1,j-1}$	$I_{i,j-1}$	$I_{i+1,j-1}$

Indexing the image

- The median-filtered image is then thresholded.
  - We could use an absolute threshold, a relative threshold (based, *e.g.*, on the variance of the neighborhood), or both.
  - Pixels that have value greater than the threshold are called exceedances. They are candidate targets, denoted $E_{i,j}$.

Gating the image near target tracks

- Exceedances need to be grouped into possible targets.
- We define a rectangular window having width and height G pixels, centered on the estimated position of an existing track or a candidate new track, to be a "track gate".
- We consider all exceedances within a track gate to originate from the same target.
- Typically, the gate size is varied during tracking.
  - We will stipulate minimum and maximum values for G.
  - For a new track, G typically starts out at $G_{\max}$.



Implications of the measurement-gate size

- Why vary the gate size?
  - For a candidate new track, our knowledge of the true track position is typically poor. So, we want a large, or "loose" gate to be sure we pick up all the exceedances associated with the track.
  - For an existing track, if the measurements have not been agreeing well with the predictions, we also want a large gate for the same reasons as above.
  - For an existing track where the measurements and predictions have been in good agreement, however, we want the gate to be small, or "tight."
  - Since only exceedances within the gate will be included in the observed threshold calculation, a tight gate means less noise in the centroid calculation.
- Note: Henceforth, we consider tracking only in the horizontal direction. Vertical tracking is analogous.

Gating and association: Update of existing tracks

- We filter and threshold frame k to detect exceedances.
- For each existing track, we predict position: $\hat{\xi}_k^- = A\hat{\xi}_{k-1}^+$; we then place the track gate centered at $\hat{\xi}_k^-$ and compute the centroid of the exceedances in the gate. This centroid is the observation z_k for the track.
- The centroid of the exceedances in the track gate is calculated:

$$z_k = \frac{\sum_{m,n\in\text{Gate}} m E_{m,n}}{\sum_{m,n\in\text{Gate}} E_{m,n}} \quad \text{where} \quad E_{i,j} = \begin{cases} Y_{i,j}, & Y_{i,j} \geq \text{threshold;} \\ 0, & \text{else.} \end{cases}$$

  - Measurement noise arises from the discrete nature of the camera, thermal noise, and imperfections in the gating and target extraction processing.
- Note, the standard deviation of the exceedances in the gate is also sometimes used in the calculation of G, the gate size.
- For each existing track, its z_k is used to update its Kalman filter.

New track starts

- In any given frame, there may be exceedances that do not fall near the gate of any existing track ("do not associate.")
- A default-sized gate is placed around such exceedances and they become candidates for new tracks.
  - But, since spurious bright spots may occur due to reflections, sun glints, and so forth, we don't start a new track right away.
  - Instead, we require that the candidate track to persist for some small number of frames before we actually start a new track. This is called the "persistence test."
- If the candidate track passes the persistence test, we use the most recent measurements to initialize a track. For an NCV model, we may choose:

$$\hat{x}_0 = \begin{bmatrix} z_0 \\ \frac{z_0 - z_{-1}}{\Delta t} \end{bmatrix} \quad \text{and} \quad \Sigma_{\tilde{x},0}^{+} = \begin{bmatrix} \Sigma_{\tilde{v}} & \frac{\Sigma_{\tilde{v}}}{\Delta t} \\ \frac{\Sigma_{\tilde{v}}}{\Delta t} & \frac{2\Sigma_{\tilde{v}} + \Sigma_{\tilde{w}}(\Delta t)^4/4}{(\Delta t)^2} \end{bmatrix}.$$

Track coasting

Track coasting:

- Sometimes a target is temporarily obscured; then, there will be no exceedances in the track gate of the existing track.
- When this occurs, we need to "coast" the track.
- This involves two steps:
  1. Open the track gate up to the maximum size.
  2. Allow the KF to evolve with no measurement update (open-loop).
- If the number of missed detections exceeds a set limit, then the track is terminated.
  - This is called "track loss" or "track deletion."
- It may be helpful to restart the Kalman filter upon reacquisition.

Track crossings and merging

Track crossings:

- Sometimes two targets cross paths; their gates will overlap.
- In this case, it may not be possible to resolve the individual targets.
- The best thing may be to coast both tracks and hope to reacquire the targets when they separate; it may be helpful to restart the Kalman gains upon reacquisition.

Track merge:

- Sometimes two tracks will cross and never come apart again; we merge the tracks.
- The simplest way to merge is to keep the track that agrees best with the observations and delete the other track.

Summary

- In this lesson, you learned that in a target-tracking application, Kalman filters often comprise only one component of the solution, which also must include methods to:
  - Detect the presence of targets in the sensor field;
  - Associate targets with the correct Kalman filters tracking those targets;
  - Start the execution of new Kalman filters when a new target appears;
  - Merge, coast, and/or delete tracks when targets disappear.
- Since u_k is unknown, the Kalman filters must use approximate models of motion, such as NCP, NCV, and CT.
- With this background strategy in mind, the rest of this week will focus specifically on topics related to the application of Kalman filters to target tracking.

Credits

- The NCV initialization on slide 10 was derived from: X. Rong Li and Chen He, “Optimal Initialization of Linear Recursive Filters,” *Proceedings of the 37th IEEE Conference on Decision and Control*, Tampa, FL, Dec. 1998, pp. 2335–2340. For additional generality and rigor, see the original paper.