Assignment 1 (20% of the total course mark)

Implement a nearest-neighbor CMKF tracker for the following scenario:

1. Target:
    - Generate a single target that moves with a nearly constant velocity model.
        - Note that the process noise in the x and y axis is independent and not zero.

2. Sensor:
    - Parameters:
        - Position: [1000 500], Stationary (i.e., velocity = [0 0])
        - Sensor Measurements: range and azimuth
        - Error standard deviation: range = 10 m, azimuth = 0.01 rad
        - Sampling time = 2 s
        - Pd = 0.9, generate measurements with a probability of Pd.
        - False alarm density = 1e-4
        - Coverage: range [0 to 10000]m, azimuth [-π to π]

    - ➢ Generate measurements from targets with given Pd and measurement noise.
    - ➢ Generate false alarms: the number of false alarms follows a Poisson distribution with the mean number of events λ:

      $$\lambda = \mathit{false\ alarm\ rate} \times \mathit{range\ coverage} \times \mathit{azimuth\ coverage}$$

      The values of false alarms follow a uniform distribution.
    - ➢ You may need to change the parameter values while performing the testing.

3. Tracker:
    - Assume the track is already initialized.
        - In each Monte-Carlo run, you need to reinitialize the target state vector x_0 and the target state covariance P_0.
        - Pick a P_0 and select a random x_0 based on the actual initial target location and the assumed covariance.
    - Use simple nearest-neighbor data association.
        - Pick the measurement corresponding to the smallest normalized distance in the gate.
    - Use CMKF filtering.
        - Convert the range and azimuth measurements into x and y measurements, then apply the Kalman filter with these converted measurements.

4. Performance Evaluation:
    - Evaluate RMSE
        - Plot position and speed RMSEs separately
    - Evaluate the performance for different target trajectories and sensor parameters

In the report, please include the following:

- Problem description
- Implementation details
- Plots showing the truth and estimated trajectories
- Plots showing the RMSE for different test cases by changing the target and sensor parameters (e.g., target process noise level, target initial state, sensor measurement covariance, etc.)
- Discussion of the results

Upload the Matlab code as a zip file to Avenue to Learn.

Due date:

Task	**Due Date**
Final code	February 20
Presentation	February 20
Updated code (if modified after the presentation) and report	February 24

Suggested intermediate deadlines:

Task	**Due Date**
Target generation & parameter file	January 23
Measurement generation	January 30
Track initialization, Monte-Carlo and time step loops & Data association	February 6
Filtering	February 13
Remaining	February 19