Final Project: Localisation using EKF

SC649: Embedded Controls and Robotics

Team 2: Aayush Prasad (22b0674), Dhriti Maniar (22b2176), Manan Mehta (22b2129)

Indian Institute of Technology, Bombay

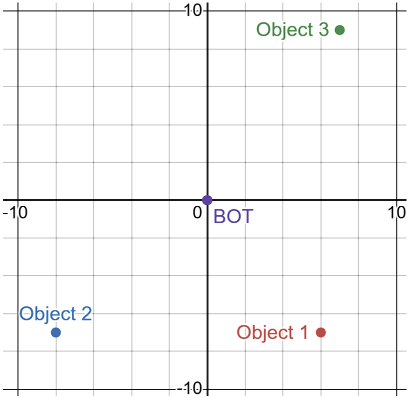
Goal

- The Goal of this project is to build and implement an Extended Kalman Filter (EKF) on a burger Turtlebot3 in the ARMS Lab setting.
- The team will be using the ViCON motion sensing system to gain information about the odometry of the turtlebot.

Trilateration and Triangulation

For our team, the objects are located at (6,-7),(-8,-7),(7,9)

Using the objects, the bot can localise itself by calculating the distances and angles between the objects and itself. Thus, it can find its own pose quite accurately if originally intitialised with its own position as well as the locations of the objects.



Extended Kalman Filter

- The Extended Kalman Filter (EKF) is a powerful algorithm used for state estimation in systems where the process and measurement models are nonlinear.
- It generalises the simple Kalman Filter (KF) to incorporate for non-linearities.

System Description

The system can be described from the following equations:

$$X_t = g(\mu_{t-1}, u_t) + \epsilon_t \text{ [Prediction Step]} \tag{1}$$

$$z_t = h(\overline{\mu_t}) + \delta_t \text{ [Update Step]} \tag{2}$$

where X is a system state; u_t is a system input and z_t is a measurement. ϵ_t and δ_t are zero mean multivariate noise of covariances R and Q respectively. i.e $(\epsilon_t) \sim \mathcal{N}(0,\ R)$ and $(\delta_t) \sim \mathcal{N}(0,\ Q)$

Procedure

1. Start with describing the system equations, which for the turtlebot is:

$$\dot{X}_t = \begin{bmatrix} \dot{x}_t \\ \dot{y}_t \\ \dot{\theta}_t \end{bmatrix} = \begin{bmatrix} v\cos(\theta_t) \\ v\sin(\theta_t) \\ \omega \end{bmatrix}$$

   i.e $\dot{X} = f(X, v)$
2. Initialise the matrices: **R**(process noise), **Q**(measurement noise), **P**(initial random Covariance Matrix), **F**(system matrix for unicycle)
3. Linearise the matrices H from equations (1) and (2): (Thus, we need to take the Jacobian of the matrices:)

$$H_A = \begin{bmatrix} \frac{\partial r_A}{\partial x} & \frac{\partial r_A}{\partial y} & \frac{\partial r_A}{\partial \theta} \\ \frac{\partial \theta_A}{\partial x} & \frac{\partial \theta_A}{\partial y} & \frac{\partial \theta_A}{\partial \theta} \end{bmatrix} = \begin{bmatrix} \frac{-\Delta x_A}{r_A} & \frac{-\Delta y_A}{r_A} & 0 \\ \frac{\Delta y}{r_A} & \frac{-\Delta x}{r_A} & -1 \end{bmatrix}$$

   H will help us determine the current pose of the car from measurements.

Procedure

4. Collect the measurements from objects A, B, and C. Collect multiple measurements and average it out. ($\because$ the noise is 0 averaged.)
5. Using the data collected, we predict the pose of the bot:

$$\begin{bmatrix} x_t \\ y_t \\ \theta_t \end{bmatrix} = \begin{bmatrix} x_{t-1} + v\cos(\theta_t) * \Delta t \\ y_{t-1} + v\sin(\theta_t) * \Delta t \\ \theta_{t-1} + \omega * \Delta t \end{bmatrix}$$

   and update the covariance matrix: $\mathbf{P} = F * \mathbf{P} * F^T + R$
6. Utilise this data to predict new measurements for the entire system. Calculate distances using Eulerian distance and Bearing angles using the atan2 function.
7. Calcuate the residual ($Y_{measured} - \text{Predicted Measurement}$).
   $S = H * P * H^T + Q$
   $K = P * H^T * S^{-1}$
   The Estimated Pose= Predicted Pose + K*Residual

Procedure

8. Now apply a proportional controller on the error between the reference and actual turtlebot. Publish appropriate linear and angular velocities to the turtlebot.

Question 2: Trajectory Tracking

https://drive.google.com/drive/folders/1DZpZicqroM9jUN-pjnqrcMNclfdR7cLO?usp=sharing

Question 2: Trajectory Tracking

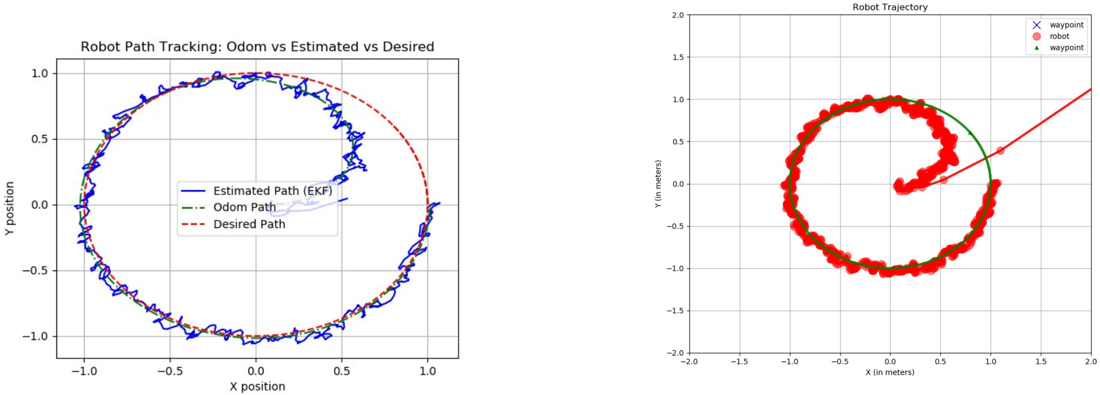


Figure: Estimated and desired Path for $\delta = 0$

Question 3: Variation of MSE with δ

δ	0	0.1	0.2	0.3	0.5	6
MSE	0.3218	0.3119	0.3068	0.2718	0.3256	0.3575

Question 4

Question: Give reasoning for the choice of initial co-variance matrices and analyse your results based on the sampling time variation of the correction step.
Answer: The initial covariance matrix the team has utilised is

$$\begin{bmatrix} 0.1 & 0 & 0 \\ 0 & 0.1 & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

The values 0.1 for x and y suggest the TurtleBot has a relatively accurate estimate of its initial position. This is because the original pose of the turtlebot is known to us (0,0,0). However, we are uncertain about the initial angle of the bot's motion. Thus, the variance of the angle is set comparatively higher. Other values are set 0 to denote that originally, we believe that there might not be a correlation betweeen x, y, θ and hence their mutual Cov is 0.

The End