DEPARTMENT OF MECHATRONICS ENGINEERING

Localization-based navigation is a core concept in **autonomous systems and robotics**, where a robot or agent determines its position within an environment and uses this information to plan and execute movement.

Definition:

Localization-based navigation is the process of using **sensor data** and **map information** to estimate a robot's current location and navigate through an environment accurately and autonomously.

2. Importance of Localization

- Enables **autonomous decision-making** in unknown or dynamic environments.
- Critical for applications such as:
  - Self-driving cars
  - Delivery robots
  - Drones
  - Mobile assistants
- Supports **safe**, **efficient**, and **goal-oriented movement**.

3. Components of Localization-Based Navigation

a. Localization

- Estimating the robot's **position** and **orientation (pose)**.
- Can be:
  - **Global**: Unknown initial position.
  - **Local**: Known initial position, tracking over time.

b. Mapping

- Creating or using a **map** of the environment.
- Types of maps:

- Grid maps
- Topological maps
- Semantic maps

c. Motion Planning

- Determining a path from the current position to the goal.
- Considers obstacles, dynamic elements, and optimality (time, energy, etc.).

d. Control

- Executing the planned path using motor commands while adapting to feedback.

4. Localization Techniques

Technique	Description	Example
Dead Reckoning	Uses odometry (wheel rotations, IMU)	Low accuracy over time due to drift
GPS-based Localization	Uses satellite signals	Effective outdoors; inaccurate indoors
Beacon-based (RFID, UWB)	Uses fixed reference points	High accuracy in controlled environments
Vision-Based (VSLAM)	Uses cameras and computer vision	Common in drones and AR devices
LiDAR-Based Localization	Uses laser sensors to detect surroundings	Highly accurate in structured environments
Sensor Fusion	Combines multiple sensors (e.g., IMU + GPS + LiDAR)	Improves robustness and accuracy

5. Algorithms for Localization

a. Kalman Filter (KF)

In mobile navigation, accurate estimation of a robot's position, orientation, and velocity is crucial for safe and effective movement. The **Kalman Filter (KF)** helps by **fusing noisy sensor data** and

predicting the system state in a mathematically optimal way.

Kalman Filter in Mobile Navigation

- Sensors like **GPS**, **wheel encoders**, **IMUs**, and **lidar** have **noise** and **drift**.
- KF fuses multiple data sources to produce a **more accurate** and **stable** estimate.
- KF provides a **recursive update**, which is computationally efficient for real-time navigation.

System Modeling for Mobile Robots

Assume a robot navigating in 2D space:

🔹 **State Vector (example):**

$$\mathbf{x} = \begin{bmatrix} x \\ y \\ \theta \\ v \\ \omega \end{bmatrix}$$

Where:

- x, y: Position
- θ: Orientation (heading)
- v: Linear velocity
- ω: Angular velocity

🔹 **Control Input:**

$$\mathbf{u} = \begin{bmatrix} a \\ \alpha \end{bmatrix}$$

Where:

- a: Linear acceleration
- α: Angular acceleration

🔹 Motion Model (Prediction):

Using a **discrete-time kinematic model**, the robot's next state can be predicted as:

$$\begin{aligned}
x_{k+1} &= x_k + v_k \cos(\theta_k)\Delta t \\
y_{k+1} &= y_k + v_k \sin(\theta_k)\Delta t \\
\theta_{k+1} &= \theta_k + \omega_k \Delta t \\
v_{k+1} &= v_k + a\Delta t \\
\omega_{k+1} &= \omega_k + \alpha\Delta t
\end{aligned}$$

This is **non-linear**, so we typically use an **Extended Kalman Filter (EKF)**, which linearizes around the current estimate.

Sensor Fusion in KF

Common Sensors:

Sensor	Measures	Limitations
IMU	Acceleration, angular velocity	Drift over time
GPS	Global position	Low frequency, noisy
Odometry	Relative motion	Wheel slip errors
Compass	Orientation	Magnetic interference

Kalman Filter Role:

- Combine IMU (fast but drifty) with GPS (slow but accurate) to **smooth position estimation**.
- Combine odometry with compass to estimate robot's heading and location.

Kalman Filter Workflow for Mobile Navigation

🔵 Prediction Step

1. Predict the next state using motion model:

$$\hat{\mathbf{x}}_k^- = f(\hat{\mathbf{x}}_{k-1}, \mathbf{u}_k)$$

2. Predict the new covariance:

$$\mathbf{P}_k^- = \mathbf{F}_k \mathbf{P}_{k-1} \mathbf{F}_k^\top + \mathbf{Q}$$

Where $\mathbf{F}_k$ is the Jacobian of the motion model.

🟢 Update Step

1. Compute the Kalman Gain:

$$\mathbf{K}_k = \mathbf{P}_k^- \mathbf{H}_k^\top (\mathbf{H}_k \mathbf{P}_k^- \mathbf{H}_k^\top + \mathbf{R})^{-1}$$

Navigation Strategies

a. Map-Based Navigation

- Requires a pre-built or online-generated map.
- Examples: A* algorithm, Dijkstra's, RRT.

b. Mapless (Reactive) Navigation

- Relies on sensor feedback without a global map.
- Examples: Obstacle avoidance, potential fields.

c. Hybrid Approaches

- Combines global planning with local obstacle avoidance.
- Common in real-world autonomous systems.

7. Challenges in Localization-Based Navigation

Challenge	**Description**
Sensor Noise	Inaccurate or delayed sensor readings

Dynamic Environments	Moving people or objects disrupt planning
Drift	Accumulated error in dead reckoning
Loop Closure	Identifying previously visited locations accurately
Localization Failure	Losing track of position due to poor data or featureless environments

8. Applications

- Autonomous Vehicles: Navigation in cities with GPS and LiDAR.
- Warehouse Robots: Navigate aisles using QR codes and SLAM.
- Drones: Localize and fly indoors using VIO (Visual Inertial Odometry).
- Service Robots: Navigate homes using vision-based maps