Assignment 4: Victoria Park_EKF-SLAM

EKF-SLAM Algorithm Overview

EKF-SLAM operates on the same principles as the Extended Kalman Filter (EKF). However, unlike a standard EKF, which typically uses a constant state matrix to track robot features such as position, velocity, and orientation, the EKF-SLAM algorithm dynamically updates its state vector and covariance matrix to reflect the discovery of new landmarks in the environment.

Similar to the EKF, each iteration of EKF-SLAM consists of two main steps:

- **Prediction Step:** Use the motion model and the previous state estimate to predict the current state.
- **Update Step:** Incorporate new sensor measurements (e.g., odometry, GPS, LIDAR) to correct the predicted state using the Kalman Gain.

In addition to these standard EKF steps, EKF-SLAM must solve the *data association* problem—identifying whether an observed feature corresponds to an existing landmark or represents a new one.

To handle this, I implemented a **maximum likelihood estimator** based on the *Mahalanobis Distance*. This metric, unlike Euclidean distance, accounts for the covariance of the observations, making it well-suited for associating noisy measurements with known landmark distributions. Since each landmark (e.g., tree or cone) is modeled as a Gaussian distribution, Mahalanobis Distance provides a principled and effective approach to matching observations with existing map features.

Algorithm 1 : EKF SLAM

while filter running **do**
    $e \leftarrow$ next event to process
    if e is an odometry event **then**
        Perform EKF propagation with e
    else if e is a GPS measurement **then**
        Perform an EKF update with e
    else if e is a laser scan **then**
        Extract tree range, bearing measurements $\{z\}$ from e
        Perform an EKF update with $\{z\}$
    end if
end while

Figure 1: EKF Algorithm

System Overview

This report outlines the implementation of an **Extended Kalman Filter SLAM (EKF-SLAM)** system using data from a victoria SLAM dataset taken from sensor-equipped vehicle. The vehicle is equipped with a wheel encoder, GPS, and LIDAR sensor, and traverses an environment populated with tree landmarks. The motion model and measurement model was taken as instructed in the assignment.

The full state vector is defined as:

$$x = \begin{bmatrix} x & y & \phi & f_1^T & \dots & f_n^T \end{bmatrix}^T$$

where (x, y, ϕ) represents the robot's pose and $f_i = [x_i, y_i]^T$ are 2D positions of the i-th landmark.

EKF-SLAM alternates between *propagation* and *update* steps as sensor data becomes available.

2.1 Odometry Propagation

The vehicle motion model and its Jacobian were implemented based on the given kinematics. The propagation step uses odometry input to predict the next state:

$$x_{t+1} = f(x_t, u_t), \quad P_{t+1} = F_t P_t F_t^T + Q$$

2.2 GPS Update

GPS updates are incorporated using the measurement model:

$$h(x) = \begin{bmatrix} x \\ y \end{bmatrix}$$

To reject outliers, I compute the Mahalanobis distance:

$$d = r^T S^{-1} r$$

Updates are accepted only if $d < 13.8$, based on the χ^2 distribution with 2 degrees of freedom (0.1% threshold).

2.3 LIDAR Update

(a) Range-Bearing Model

The measurement model is defined as:

$$z = h(x) = \begin{bmatrix} \sqrt{(x_L - x)^2 + (y_L - y)^2} \\ \arctan\left(\frac{y_L - y}{x_L - x}\right) - \phi + \frac{\pi}{2} \end{bmatrix}$$

(b) Landmark Initialization

New landmarks are initialized using the inverse of the measurement function and the robot's current pose. The state vector and covariance matrix are expanded accordingly.

(c) Data Association

Measurements are associated with known landmarks using the Mahalanobis distance. A cost matrix is constructed and solved heuristically. Each measurement is:

- Associated with an existing landmark,
- Used to initialize a new landmark, or
- Discarded if ambiguous.

(d) EKF Laser Update

For each valid laser measurement, I perform an EKF update to refine both the robot's pose and the associated landmark positions.

Practical Considerations

- Covariance matrix P is symmetrized after updates to prevent numerical issues.
- If necessary, P is adjusted to remain positive semi-definite by adding a small κI.
- All angles (e.g., ϕ) are normalized to the range $[-\pi, \pi)$.

Conclusion

The EKF-SLAM system integrates asynchronous odometry, GPS, and LIDAR data to simultaneously estimate the vehicle's pose and map the environment. Accurate data association and outlier rejection are crucial for robust and stable SLAM performance.

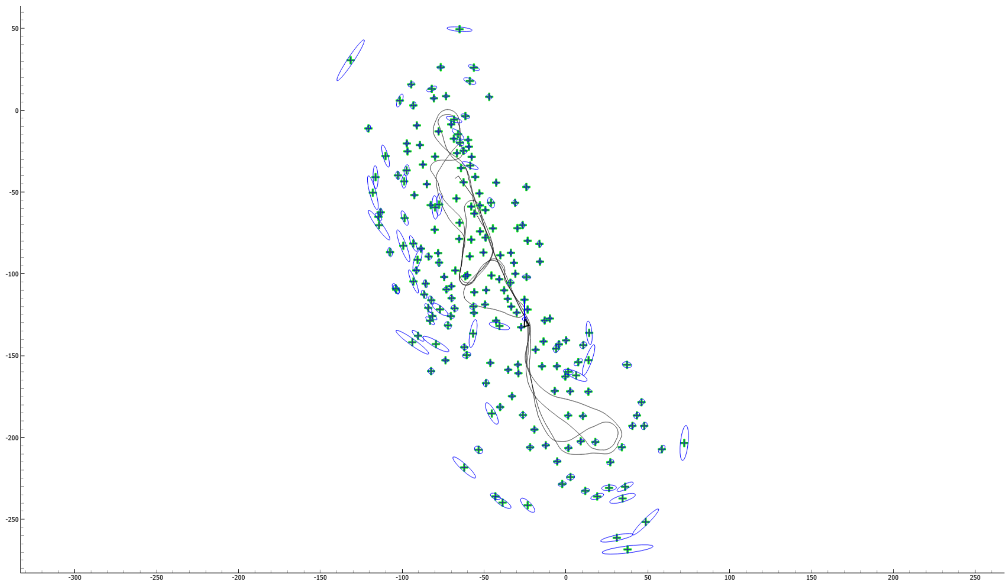

Figure 2: Environment Map

Reference

- EKF-SLAM GitHub Repository: https://github.com/adarshmodh/EKF_SLAM