Extended Kalman Filter Localization

Internship Report

Atreya G Bhat

Abstract

Robot Localization is an essential task in the field of autonomous vehicles wherein the vehicle has to localize in the deployed environment. Extended Kalman Filter (EKF) is a commonly used framework to fuse different sensors in order to localize the robot. In this work, we formulate a sensor fusion framework to combine multiple sensors such as wheel encoders/ IMU/ GPS/ Lidar and determine the robot's pose without a pre-built high-precision map. The performance is tested on standard datasets such as KITTI.

Introduction

Localization means the estimation of the robot's position and orientation in an environment with or without a pre-built map. Robust localization is possible from fusing the measurements from diverse on-board sensors like LIDAR, GPS, IMU and wheel encoder. Intuitively, this is because the noise characteristics of these sensors vary with different operating conditions. For example, Lidar can be accurate even in low-light conditions and under tunnels, while GPS provides reliable estimates in blue-sky scenarios and wheel encoders are very accurate in the short-term whereas an IMU can give out intrinsic parameters of the robot's state with high precision but can drift over time leading to increasing uncertainty. By combining the relative strengths, the estimate of the robot's pose is more refined. A probabilistic approach is used where each sensor gives an uncertainty estimate of the robot's pose. Various sensor estimates are fused asynchronously with the relative weights for each sensor being a measure of confidence of the sensor estimate. This leads to the composite robot pose being more accurate.

In this report, the EKF algorithm is implemented which utilizes LIDAR, GPS, Wheel Encoder and IMU to localize a vehicle based on their availability in the specific datasets being tested on. The two main configurations implemented are fusing LIDAR, GPS and IMU in the KITTI dataset and fusing LIDAR, Wheel Encoder and IMU measurements in the IISc dataset. The report also outlines the implementation of ICP algorithm on the lidar data and Mercator scale conversions for GPS data both to get the relative pose of the robot. This work can also be extended to work on real-time systems where each sensor has a different data output rate.

Extended Kalman Filter

Extended Kalman Filter is an extension of the classic Kalman Filter which was originally formulated with the assumption that the process and measurement models are linear with Gaussian noise models. In real-world situations, these constraints would need to be relaxed. EKF does this by linearizing the non-linear motion or measurement using the Jacobian at the current operating point. Kalman filters work as an iterative two-step process, namely, prediction and correction. The assumed motion model is used to predict the vehicle state at time k, given the previous vehicle state and the current control input. The correction step uses the measurement model by incorporating the observed data from the sensors and the sensor noise covariance to update the predicted state. Higher weight to a given sensor's observation implies greater confidence and hence the estimated robot pose leans closer to this observation.

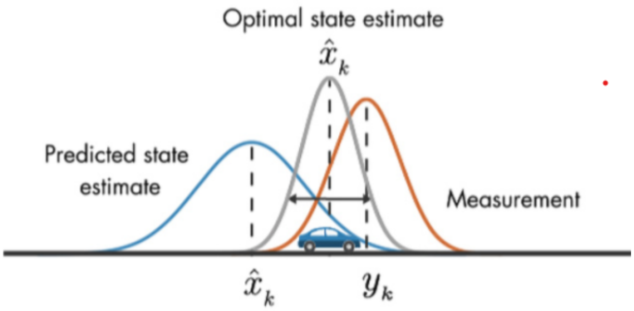

Figure 1: KF representation

The motion and observation models for EKF :

$$X_t = f(X_{t-1}, u_t) + V_t, V_t \sim N(0, Q_t)$$

$$Z_t = h(X_t) + W_t, W_t \sim N(0, R_t)$$

The EKF update algorithm follows the conventional solution structure given below :

Prediction

$$X_k = FX_{k-1} + BU$$
$$P_k = F_k P F_k^T + Q_k$$

Correction

$$K = P_{k-1}H^T(HP_{k-1}H^T + R)^{-1}$$
$$X_k = X_{k-1} + K(Z_k - HX_{k-1})$$
$$P_k = P_{k-1}(I - KH)$$

The notation used in the prediction and correction steps are as follows

- **X** - State Matrix estimate
- **P** - Covariance Matrix estimate
- **B** - Control Matrix
- **U** - Control input
- **F** - State transition matrix
- **Q** - Process noise covariance
- **K** - Kalman Gain
- **Z** - Measurement data
- **H** - Observation matrix
- **R** - Sensor noise covariance matrix

Iterative Closest Point Algorithm

ICP is a popular algorithm to match two lidar point clouds and find the relative translation and rotation matrices. This is used to find the relative change in pose of the robot based on the successive lidar scan data and hence provides the$(\mathbf{x}, \mathbf{y}, \theta)$ of the robot. This is used as the measurement data in the EKF correction step. The point-to-point ICP algorithm is applied on two equal-sized scans. There are two implicit assumptions for convergence

1. Ordering of points in the individual point clouds are identical
2. The change in rotation/ translation across the 2 point clouds are within a small range

Say, P is the reference scan of robot and Q is the new scan of robot. The relative motion is to be determined using ICP. Without loss of generality, each scan has M points, i.e,

$P = \{(x_1^p, y_1^p, z_1^p), (x_2^p, y_2^p, z_2^p), \cdots, (x_M^p, y_M^p, z_M^p)\}$

$Q = \{(x_1^q, y_1^q, z_1^q), (x_2^q, y_2^q, z_2^q), \cdots, (x_M^q, y_M^q, z_M^q)\}$

In case of pure translation, $Q = P + t$, i.e

$$\begin{bmatrix} x_i^q \\ y_i^q \\ z_i^q \end{bmatrix} = . \begin{bmatrix} x_i^p \\ y_i^p \\ z_i^p \end{bmatrix} + \begin{bmatrix} t_x \\ t_y \\ t_z \end{bmatrix} \quad \forall\, i = 1, 2, \cdots, M$$

In case of pure rotation, $Q = RP$, i.e.,

$$\begin{bmatrix} x_i^q \\ y_i^q \\ z_i^q \end{bmatrix} = R(\theta)R(\phi)R(\psi) \begin{bmatrix} x_i^p \\ y_i^p \\ z_i^p \end{bmatrix} \quad \forall\, i = 1, 2, \cdots, M,$$

where (θ, ϕ, ψ) is the roll/ pitch/yaw respectively.

The objective of ICP is to find R, t that minimizes

$\min_{R,t} ||RP + t - Q||^2$

Note that we restrict ourselves to 2-D point clouds in this work.

GPS Transformation

The output parameters of a generic GPS module will be in terms of latitude, longitude and altitude which need to be converted to the Cartesian system in the body frame. The Mercator scale is used for this transformation shown below :

$$x = s \times r \times \frac{\pi lon}{180}$$

$$y = s \times r \times \log\left(\tan\left(\frac{\pi(90 + lat)}{360}\right)\right)$$

where, **r** is the radius of the earth ≈ 6378137 meters and **s** is the scale constant = cos $\left(\frac{lat_0\ \pi}{180}\right)$ These coordinates are again converted to body frame coordinates with the vehicle pose initially at origin.

EKF Sensor Fusion

The sensor fusion was implemented and tested with multiple configurations and motion models. The following are the major configurations :

1. IMU as control input in the process model, while LIDAR and GPS are used in the Measurement model (KITTI dataset)
2. Wheel-encoder and IMU are the control inputs in the Process model, LIDAR in the Measurement model

The motion models are for 2D and 6D pose estimation given below. The latter is based on quaternion formulation for robot pose. X_k denotes the state matrix and the state variables are self-explanatory.

Motion model for 2D pose estimation

$$X_k = \begin{bmatrix} x_k \\ y_k \\ \theta_k \\ v_k \\ \omega_k \end{bmatrix} = \begin{bmatrix} x_{k-1} + v_{k-1}\Delta t \cos\theta_{k-1} \\ y_{k-1} + v_{k-1}\Delta t \sin\theta_{k-1} \\ \theta_{k-1} + \omega_{k-1}\Delta t \\ v_{k-1} \\ \omega_{k-1} \end{bmatrix}$$

Motion model for 6D pose estimation (Quaternion approach)

$$X_k = \begin{bmatrix} p_k \\ v_k \\ q_k \end{bmatrix} \in R^{10}$$

$$\mathbf{p_k} = \mathbf{p}_{k-1} + \Delta t \mathbf{v}_{k-1} + \frac{\Delta t^2}{2}(\mathbf{R}a_{k-1} - \mathbf{g})$$

$$\mathbf{v_k} = \mathbf{v}_{k-1} + \Delta t(\mathbf{R}a_{k-1} - \mathbf{g})$$

$$\mathbf{q_k} = \mathbf{\Omega}[q(\omega_{k-1}\Delta t)]\mathbf{q}_{k-1}$$

where, $\mathbf{p_k}$ is the position vector (3x1) , $\mathbf{v_k}$ is the velocity vector (3x1) , $\mathbf{q_k}$ is the quaternion (4x1), $\mathbf{\Omega}$ is a skew symmetric matrix and **R** is the rotation matrix corresponding to the quaternion.

Algorithm for implementation of the proposed models: We give the psuedocode for the algorithm studied

- Preprocess the lidar frames and GPS data
- Initialize the state, process and sensor noise covariances
- The observation models for different sensors are suitably chosen
- The Jacobian matrices for process/ observation model are computed if the corresponding model is non-linear

FOR each dataset sequence :

- EKF Prediction using IMU and/or Wheel Encoder as the control input
- EKF Correction using the LIDAR ICP pose as measurement input
- If number of GPS satellites > 3 and $v_x > 0.01$ m/s , then use EKF Correction using the GPS pose as measurement input
- Update the estimated states

Results and Analysis

The model was tested on KITTI dataset which is an open source dataset(Outdoor) containing time synced data from GPS, LIDAR, IMU and Stereo cameras along with their relative transformation matrices for calibration.

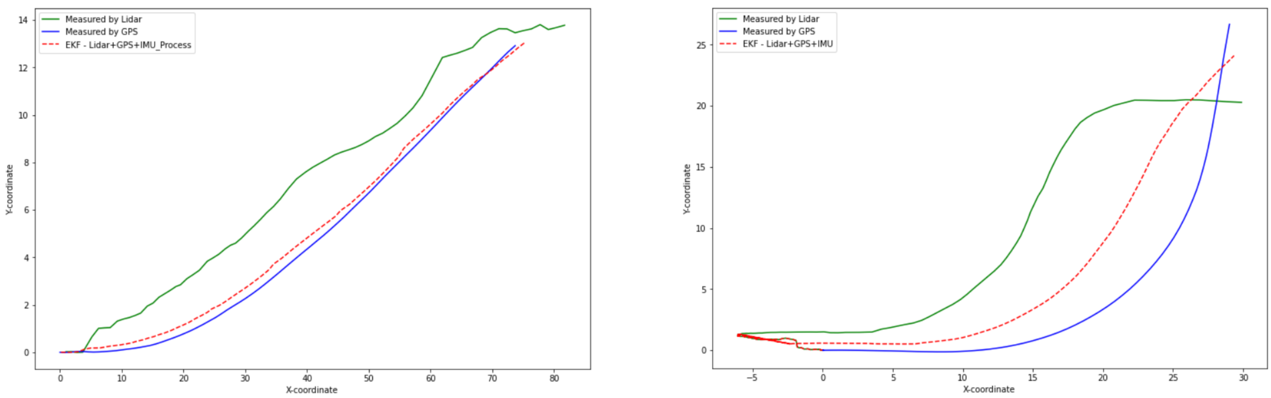

Figure 2: Results of Sequence 0002 and 0018 of KITTI dataset with the Unicycle constant velocity model

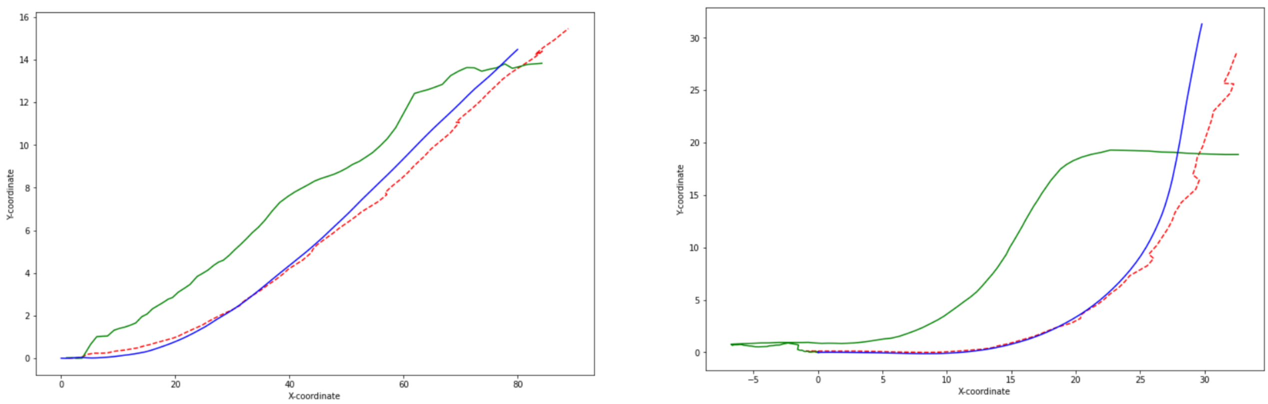

Figure 3: Results of Sequence 0002 and 0018 of KITTI dataset with Quaternion based model

- The Figures 2 & 3 show the result plots of EKF localisation by dead-reckoning on KITTI. However, the unavailability of ground-truth poses in KITTI is a hurdle to quantify the robustness of the algorithm
- The ICP algorithm implemented can be improved using robust tools to get an accurate pose estimate which will further improve the localization
- The IMU drift growth is stabilised by the use of wheel encoders but the biases themselves are ignored and not included in the state matrix
- The results obtained depend on the sensor and process noise parameters and ICP pose accuracy. These can be tuned based on the user requirement and hardware for real time systems.

Future Work

1. This work is based on Dead reckoning localization where a map does not exist beforehand. Whereas most of the localization problems work using a map which can give better results through methods like feature matching with lidar scans. We note that the implemented algorithm can gain better accuracy with a pre-built map.
2. There are more advanced and robust ICP implementations such as point-to-plane or generalized ICP than what is used in this work which can provide more accurate pose outputs.
3. The bias in IMUs has been ignored in this work. The bias even in expensive accelerometers can be about 0.01 degrees/second which can have a huge effect over long time. This can also be included in the state of the filter and also the wheel encoders can slow the growth of this error.
4. A lot of research is going on about the estimation of noise covariance matrices dynamically. This can reduce a lot of time and effort spent in tuning the model and can increase the accuracy based on how different circumstances over time can influence the noise added to the system.

References

1. **"Probabilistic Robotics"** by Dieter Fox, Sebastian Thrun, and Wolfram Burgard
2. **"Sensor data fusion using Kalman filter"** by Sasiadek J Z and Hartana P
3. **"A robust vehicle localization approach based on GNSS/IMU/DMI/LiDAR sensor fusion for autonomous vehicles"** by Meng X, Wang H & Liu B
4. **"State estimation and Localization for Self-driving cars"** , University of Toronto, Coursera
5. Lectures on Mobile Robots by Prof. Dr. Cyrill Stachniss