5 Implementation

Collecting enough statistics pertaining to the different sources of error is a precondition for the implementation of indoor localisation.

5.1 Classification of The Errors

Assume that there is a reference system to quantify the quality of our estimation (*BOSCH Laser measure* used in our experiments), errors can be classified in this way:

- Error belonging to the robot's system, e.g. driving setup and steering setup, marked as e_r
- Error due to the measurement, marked as e_m

They have at least two characteristics. Both e_r and e_m are:

1. measurable
2. explainable using a probability density function (pdf)

5.2 Calculation of process error and measurement error

- Measure the process covariance $\mathbf{Q}$ through observation: Let the robot move forward 2m under the velocity of 0.5 m/s [1] and record its position afterwards, the ground truth coordinate should be exactly 2m forward compared to the starting point. Measure the actual stop position. Repeat e.g. 100 times, which makes it reliable. *Root Mean Square Error* is the standard deviation of the residuals, $rmse = 0.001090280296483436$. Here in order to demonstrate the original calculation result, no round operation was conducted.

[1] 0.5 m/s is a proper velocity, lower velocity will make experiments less efficient.

– As for the measurement covariance **R**, let the robot stay at specific positions for a short time period, e.g. around 10s and gather some coordinate information, its ground truth position can be measured by *BOSCH Laser measure* and the measurement result comes from the *UWB tag*. Through this we can calculate the mean error, the variance of the error and Standard Deviation. *RMSE* is the measurement noise in our case.

Through *describe()* method we can get details about X-coordinates ($expected\_x = 4.045$):

Listing 5.1: describe output

```
count      100.000000
mean         4.045630
std          0.022511
min          3.985000
25%          4.033750
50%          4.051500
75%          4.062000
max          4.086000
Name: x, dtype: float64
```

We can also plot its *Histograms with different binwidths*, as shown in Figure 5.1:
Its density plot is in Figure 5.2, which is essentially a smooth version of Histogram:
Standard deviation can be calculated by:

```python
def variance(data, ddof=0):
    n = len(data)
    mean = sum(data) / n
    return sum((x - mean) ** 2 for x in data) / (n - ddof)
def stdev(data):
    var = variance(data)
    std_dev = math.sqrt(var)
    return std_dev
```

The standard deviation for the process error is: 0.02241880295075522.

According to [Nil18], the spatial RMSE can be calculated separately for X- and Y-axis through:

$$RMSE_i = \sqrt{\frac{1}{n}\Sigma_{m=1}^{n}\Big(Est_i - Actual_i\Big)^2} \tag{5.1}$$

where i is the coordinate axis. And a net RMSE can be calculated through:

$$RMSE_{Net} = \sqrt{RMSE_X^2 + RMSE_Y^2} \tag{5.2}$$

The ground truth of the first three points we chose are:

1. pose1: (x = 3.916, y = 2.465)
2. pose2: (x = 5.143, y = 1.947)
3. pose3: (x = 6.641, y = 4.788)

We gathered their measurements from *UWB tag* and write the coordinate information into csv files and calculate the MSE using *mean_squared_error()* method *from sklearn.metrics*. And the calculation result is: $rmse = 0.01422606083083504$. Their separate and concatenated density plot are in Figure 5.3.

In order to get representative measurement error, 25 poses are chosen across the lab. Their density plot and histogram is in Figure 5.4.

The standard deviation for the measurement error is: 0.07915516240539047.

5.3 Architecture Design

The architecture is shown in Figure 5.5. An EKF is first used to fuse data from internal *Odometry* and *IMU*. And then the output from the EKF is as one input of *Kalman filter* and the other input source is *UWB* data.

5.4 First Order Kalman Filter

5.4.1 One Dimension With Control Input

First order Kalman Filter tracks a first order system, e.g. our state vector is: $x = [x,\ v_x]^T$, which corresponds to X-coordinate and X-velocity. Our exogenous control input is: $u = [v_{con}]$. Assuming there's no delay in the remote control process, so the correlation between position, velocity and time interval is:

$$x = x + v_{con}\boldsymbol{\Delta}t$$

$$v = v_{con}$$

We need to apply these equations in the form of:

$$\overline{X}_k = FX_{k-1} + Bu \tag{5.3}$$

where

- $\overline{X}_k$ is State Prior
- F is Prediction/Transition Matrix
- B is Control Matrix
- u is Control Vector

which represents: **The new best estimate is a prediction from previous best estimate with a correction for known external influences**.

The transformation process is:

$$\begin{bmatrix} x \\ v \end{bmatrix} = \begin{bmatrix} 1 & 0 \\ 0 & 0 \end{bmatrix} \begin{bmatrix} x \\ v \end{bmatrix} + \begin{bmatrix} \boldsymbol{\Delta} t \\ 1 \end{bmatrix} \begin{bmatrix} v_{con} \end{bmatrix}$$

which is also equal to:

$$\begin{bmatrix} 1 & \boldsymbol{\Delta} t \\ 0 & 1 \end{bmatrix} \begin{bmatrix} x \\ v \end{bmatrix}$$

So we can know, in essence the movement through script or remote control are equivalent.

5.4.2 Two Dimensions Without Control Input

State vector for constant velocity model is: $x = [x, \dot{x}, y, \dot{y}]^T$, which corresponds to X-coordinate, X-velocity, Y-coordinate and Y-velocity.

State Transition Function

State transition function is used to derive next state from previous state: $\overline{x} = Fx$. The state equations are:

$$\overline{x} = 1x + \boldsymbol{\Delta} t\dot{x} + 0y + 0\dot{y}$$

$$v_x = 0x + 1\dot{x} + 0y + 0\dot{y}$$

$$\overline{y} = 0x + 0\dot{x} + 1y + \boldsymbol{\Delta} t\dot{y}$$

$$v_y = 0x + 0\dot{x} + 0y + 1\dot{y}$$

Its corresponding linear algebra is:

$$\overline{\begin{bmatrix} x \\ \dot{x} \\ y \\ \dot{y} \end{bmatrix}} = \begin{bmatrix} 1 & \boldsymbol{\Delta} t & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & \boldsymbol{\Delta} t \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x \\ \dot{x} \\ y \\ \dot{y} \end{bmatrix}$$

5.4.3 Two Dimensions With Control Input

State vector for constant velocity model is still the same: $x = [x, \dot{x}, y, \dot{y}]^T$, which corresponds to X-coordinate, X-velocity, Y-coordinate and Y-velocity.

State Transition Function When Local Coordinate System Corresponds To the Global One

State transition function is used to derive next state from previous state: $\overline{x} = Fx + Bu$. The state equations are:

$$\overline{x} = 1x + \boldsymbol{\Delta} t\dot{x} + 0y + 0\dot{y}$$

$$v_x = 0x + 1\dot{x} + 0y + 0\dot{y}$$

$$\overline{y} = 0x + 0\dot{x} + 1y + \boldsymbol{\Delta} t\dot{y}$$

$$v_y = 0x + 0\dot{x} + 0y + 1\dot{y}$$

We need to separate them into *state transition function.* Its corresponding linear algebra is:

$$\overline{\begin{bmatrix} x \\ \dot{x} \\ y \\ \dot{y} \end{bmatrix}} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x \\ \dot{x} \\ y \\ \dot{y} \end{bmatrix} + \begin{bmatrix} \boldsymbol{\Delta} t & 0 \\ 10^{-9} & 0 \\ 0 & \boldsymbol{\Delta} t \\ 0 & 10^{-9} \end{bmatrix} \begin{bmatrix} \dot{x} \\ \dot{y} \end{bmatrix}$$

Its alternative linear algebra is:

$$\overline{\begin{bmatrix} x \\ \dot{x} \\ y \\ \dot{y} \end{bmatrix}} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 \end{bmatrix} \begin{bmatrix} x \\ \dot{x} \\ y \\ \dot{y} \end{bmatrix} + \begin{bmatrix} \boldsymbol{\Delta} t & 0 \\ 1 & 0 \\ 0 & \boldsymbol{\Delta} t \\ 0 & 1 \end{bmatrix} \begin{bmatrix} \dot{x} \\ \dot{y} \end{bmatrix}$$

State Transition Function When Local Coordinate System Doesn't Correspond To the Global One

State transition function is used to derive next state from previous state: $\overline{x} = Fx + Bu$. However, when the local coordinate system doesn't correspond to the global one, before integrating u directly, a rotation matrix R in form of $R = \left(\begin{smallmatrix} \cos\theta & -\sin\theta \\ \sin\theta & \cos\theta \end{smallmatrix}\right)$ is integrated to transform local linear velocity to global one. As time increases, the formula becomes:

$$R_t = \Delta R_t * R_{t-1}$$

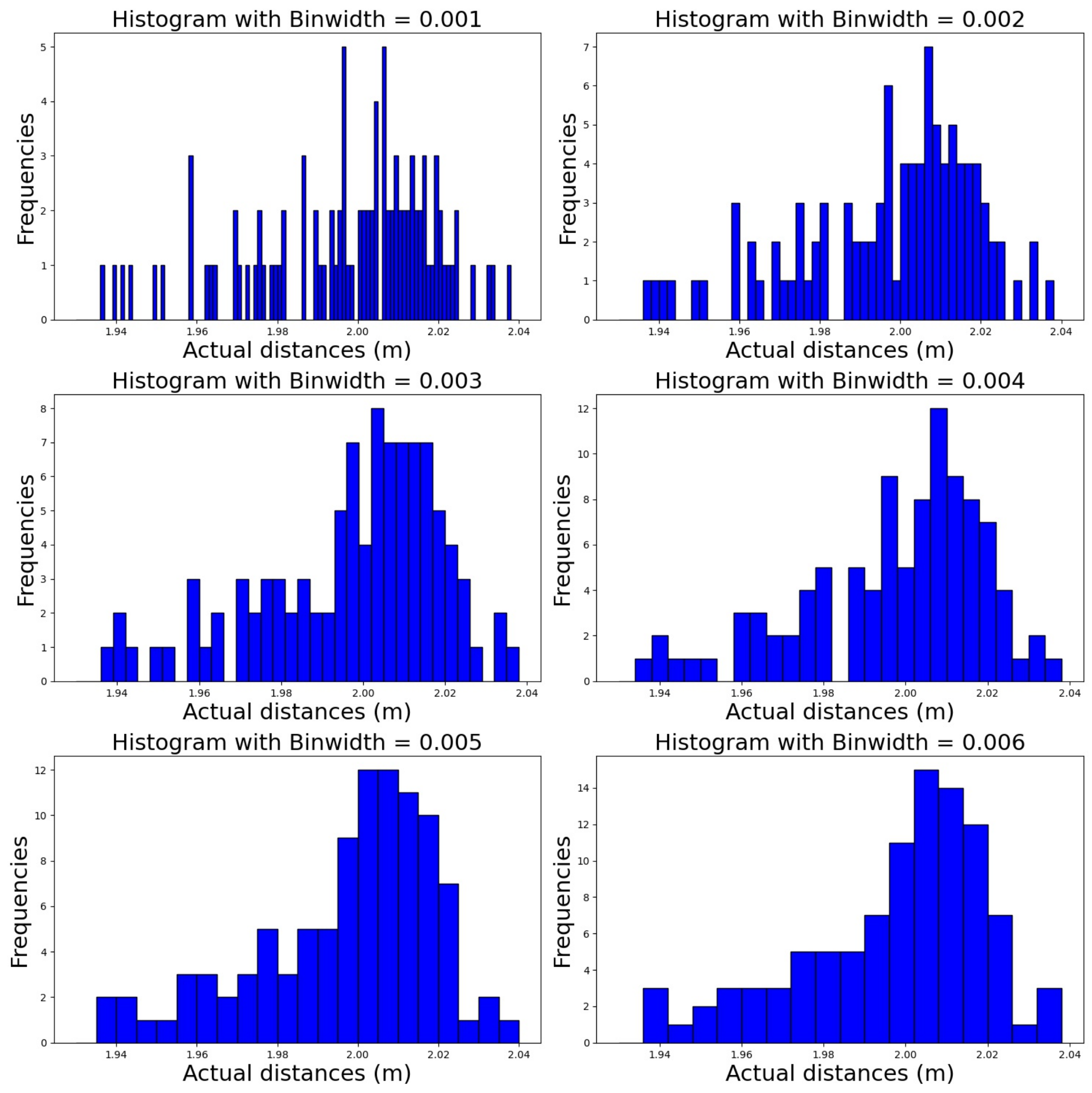
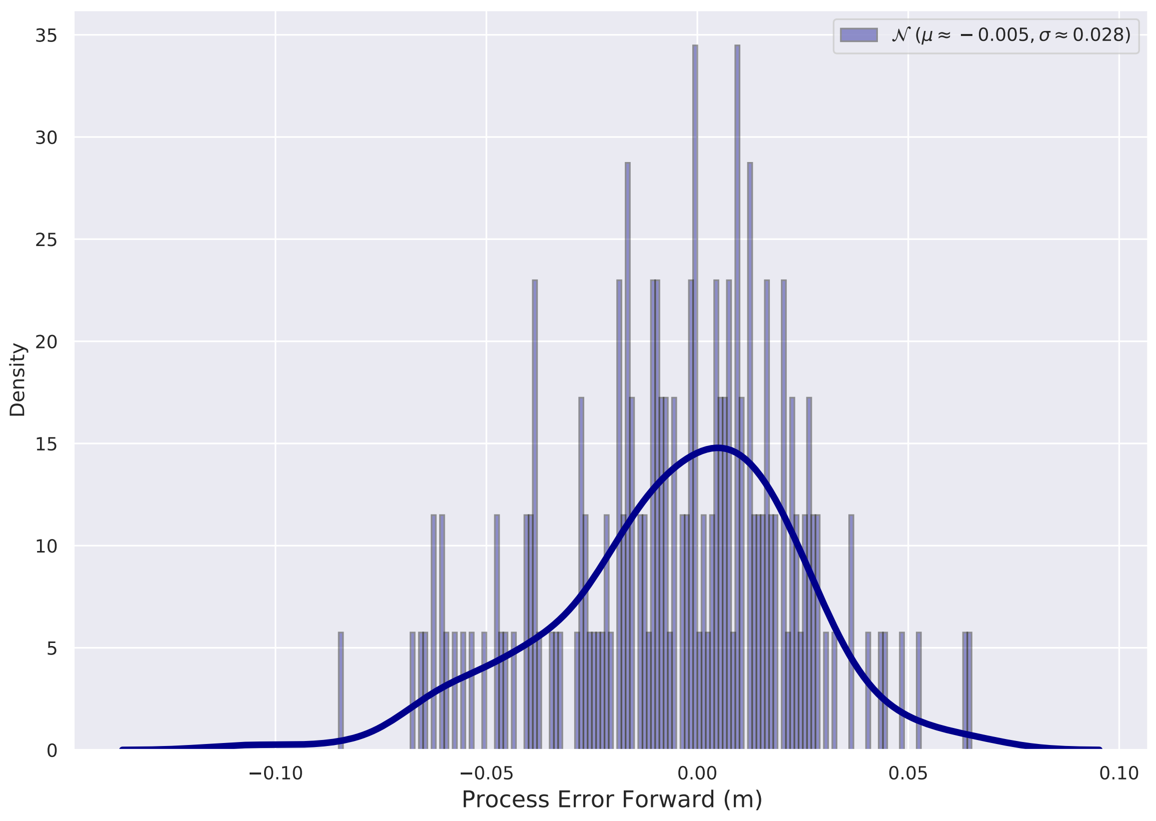


Figure 5.1: Histograms with different bin-widths for 2m movement



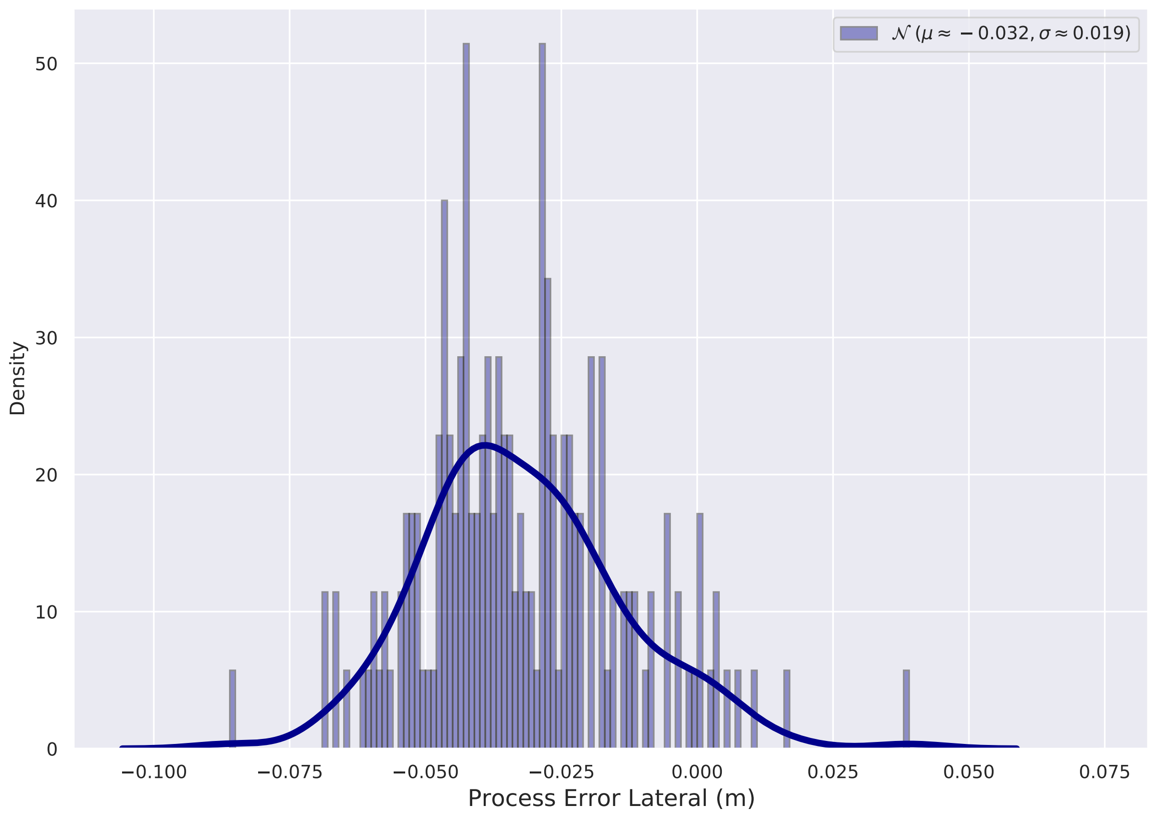
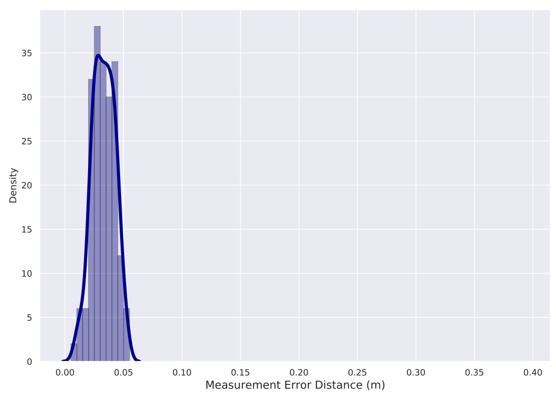
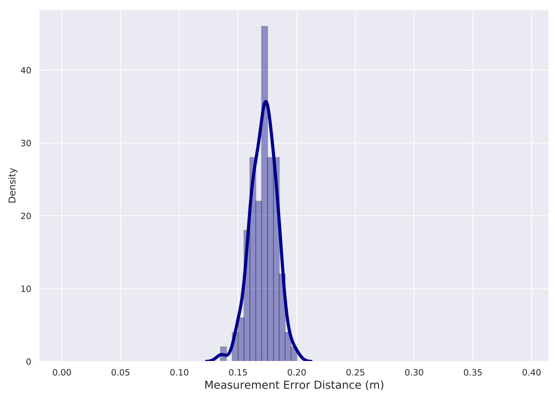


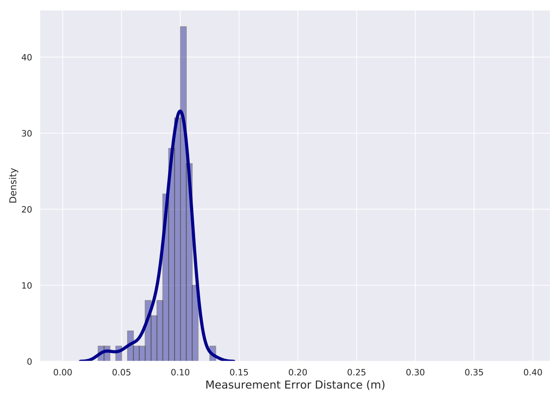
Figure 5.2: Density plot and histogram for process error forward and lateral based on 175 experiments with velocity from 0.5 m/s to 1.1 m/s



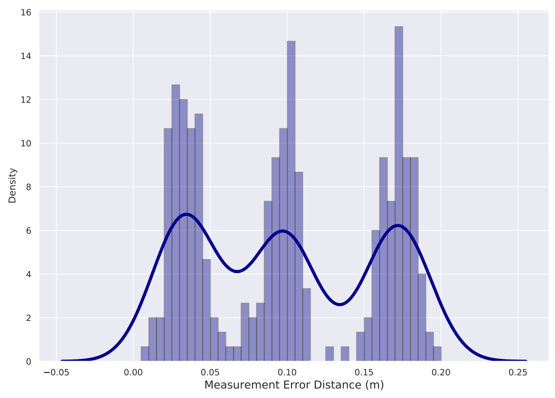
(a) Density plot for pose1



(b) Density plot for pose2

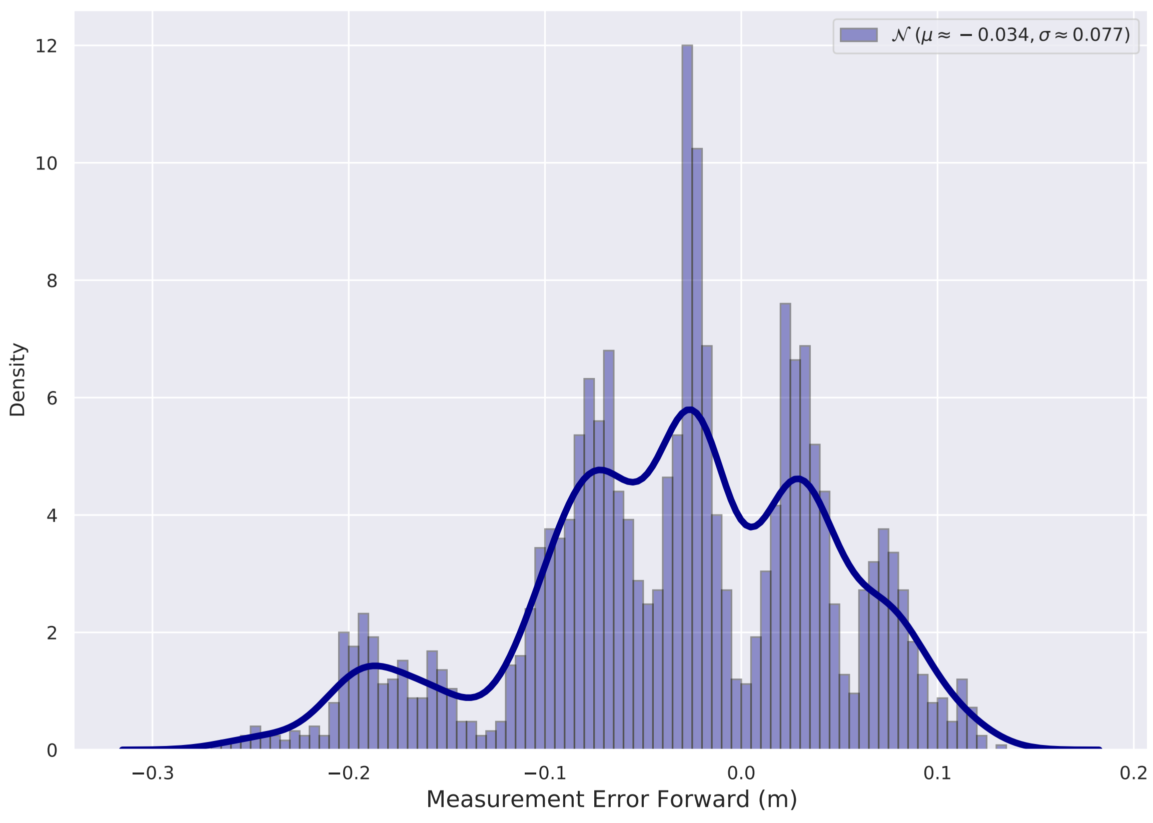


(c) Density plot for pose3



(d) Density plot for all these three poses

Figure 5.3: Density plots for measurement error for UWB tag at three chosen positions



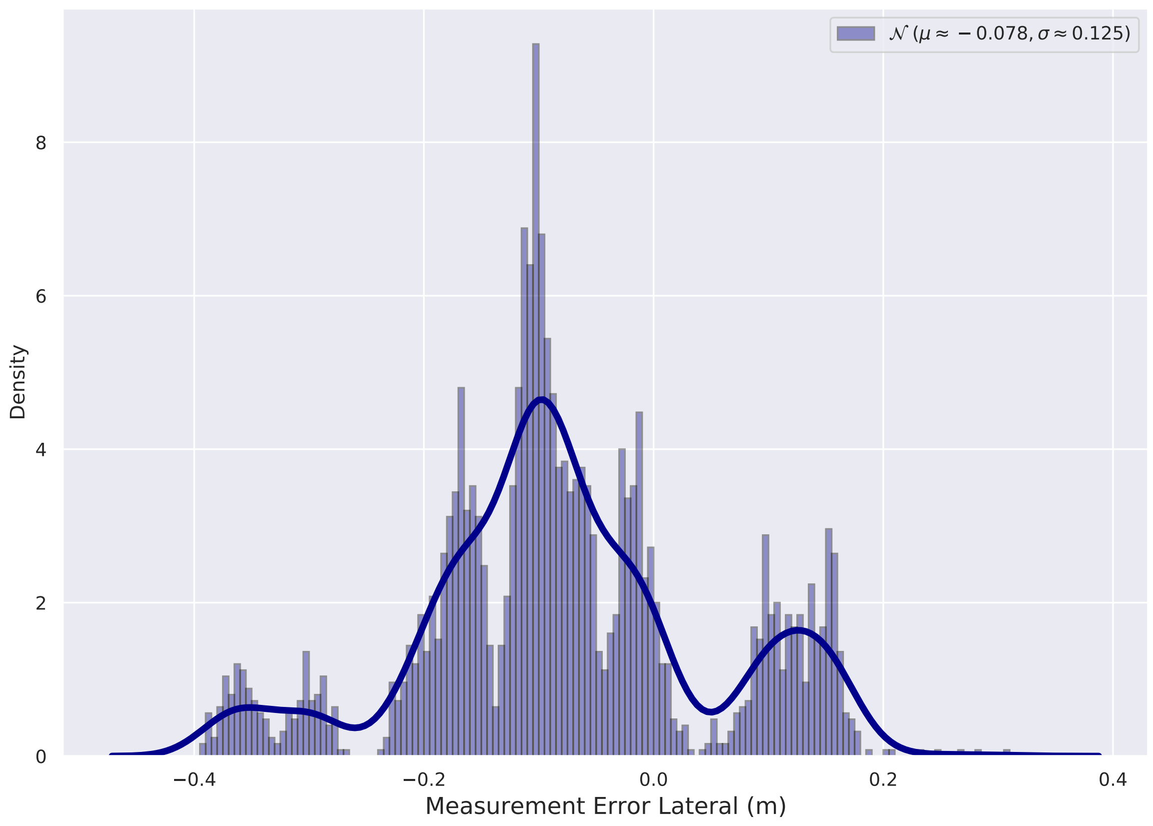


Figure 5.4: Density plot and histogram for measurement error forward and lateral based on 2500 data across 25 poses in the energy lab

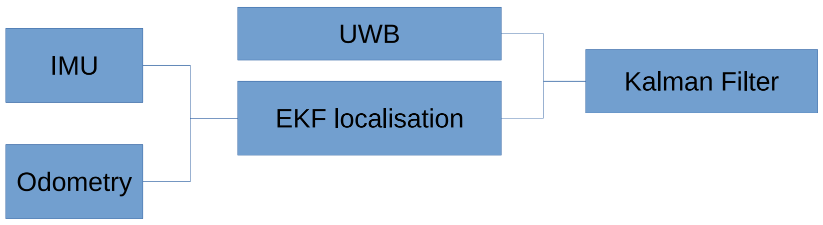


Figure 5.5: Architecture of the indoor localisation algorithm