86739: Mobile Robots
Localization of mobile robots using extended Kalman filter

Date: Mon 0:00, 06.06.16

Prof. G. Garcia, Prof. P. Martinet

Rabbia Asghar, BEng, Ernest Skrzypczyk, BSc

Contents

List of Figures

Chapter 1

Localization of mobile robots using extended Kalman filter

Experimental setup

The problem statement incorporates a robot equipped with sensor array as shown in figure (1.1), made out of eight magnetic binary sensors, located perpendicular to the x_m axis in front of the mobile platform. The reed sensors are displaced by $80mm$ in the x_m axis from the origin of the mobile platform O_m and have a spacing of $10mm$ between them. The robot has two wheel encoders with a resolution of 360 dots per revolution. Inner circuitry returns a byte representing the current state of the sensor array.

The workspace consists of a square array of magnets arranged at $55mm$ distance from each other. Localization is done by using an extended Kalman filter (EKF), as described in the next sections.

System model

The odometry based model for the robot is derived first. With encoder feedback from left and right wheel, the distance travelled by left S_L and right wheel S_R can simply be computed using equations (1.1) and (1.2)

$$S_L = \frac{n_L}{60} 2\pi r_L t \tag{1.1}$$

$$S_R = \frac{n_R}{60} 2\pi r_R t \tag{1.2}$$

where n_L and n_R are the revolutions per minute ($[rpm]$) of left and right wheel, r_L and r_R are the radii of the two wheels and t is time step in seconds.

$\Delta\theta$ represents the angle of travel for both the wheels and can be computed as shown in (1.3) where l represents wheel base, ergo distance between or axle length of two wheels.

$$\Delta\theta = \frac{S_L - S_R}{l} \tag{1.3}$$

The distance ΔD between initial position and final position of the robot can be approximated by taking average of S_L and S_R (1.4).

$$\Delta D = \frac{S_L + S_R}{2} \tag{1.4}$$

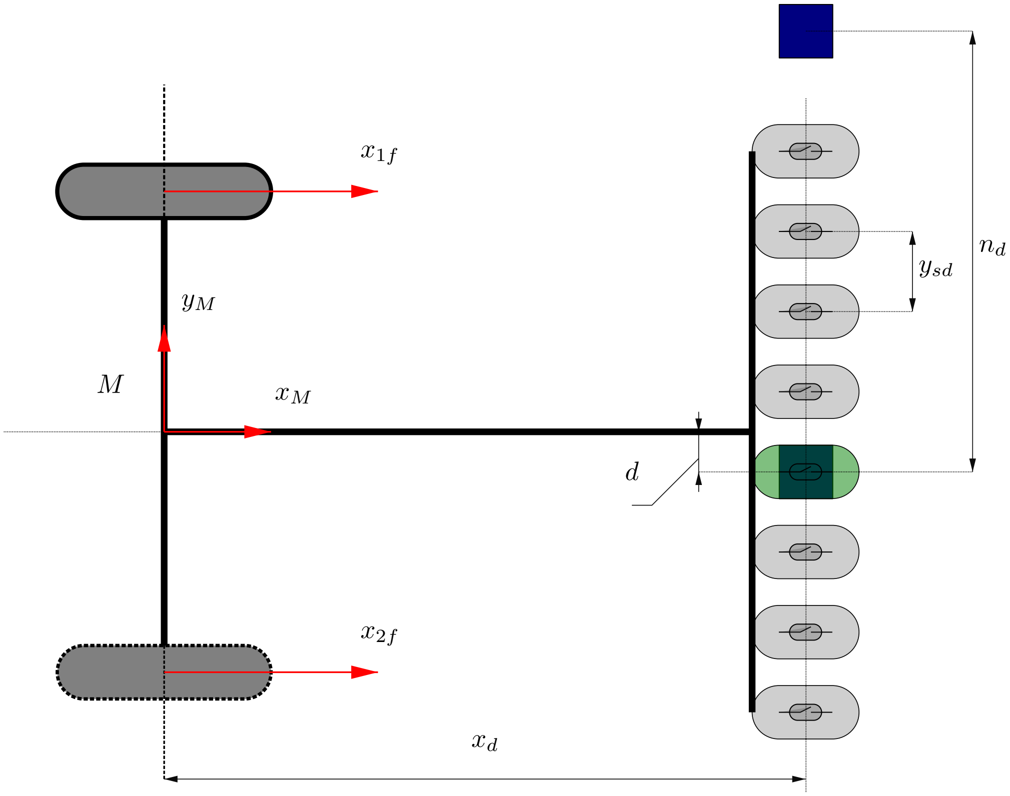


Figure 1.1: (2, 0) type mobile robot equipped with an array of reed sensors, used for localization using EKF and a square array of magnets in the ideal case of orientation $\theta = k\frac{\pi}{2}$.

Location of the robot can be defined at instant k using state variables (1.5), which include position in x and y coordinates and orientation θ.

$$X_k = \begin{bmatrix} x_k \\ y_k \\ \theta_k \end{bmatrix} \tag{1.5}$$

The control input provided to robot consists of left and right wheel speeds. This defines the distance covered by both the wheels in a given unit of time, represented by S_L and S_R. The relative displacement of the robot at instant k can be computed using ΔD and $\Delta\theta$. Thus, control input U_k can be expressed as a vector comprising of ΔD and $\Delta\theta$.

$$U_k = \begin{bmatrix} \Delta D_k \\ \Delta\theta_k \end{bmatrix} \tag{1.6}$$

Given X_k and U_k, the next location of the robot X_{k+1} can be predicted as follows.

$$\begin{bmatrix} x_{k+1} \\ y_{k+1} \\ \theta_{k+1} \end{bmatrix} = \begin{bmatrix} x_k + \Delta D_k \cdot \cos(\theta_k + \frac{\Delta\theta_k}{2}) \\ y_k + \Delta D_k \cdot \sin(\theta_k + \frac{\Delta\theta_k}{2}) \\ \theta_k + \Delta\theta_k \end{bmatrix} \tag{1.7}$$

However for simplicity, following system model equations are implemented.

$$\begin{bmatrix} x_{k+1} \\ y_{k+1} \\ \theta_{k+1} \end{bmatrix} = \begin{bmatrix} x_k + \Delta D_k \cdot \cos(\theta_k) \\ y_k + \Delta D_k \cdot \sin(\theta_k) \\ \theta_k + \Delta\theta_k) \end{bmatrix} \tag{1.8}$$

In the above derivation of the system model it was assumed that there are no noise sources. In the next section *"System model with noise"*, noise has been modelled into the system.

System model with noise

Initially noise was added to the relative displacement with the assumption, that it can be modelled by a random noise vector q_k such that, the noise is a Gaussian distribution with zero mean $\hat{q_k}$ and covariance matrix Q_β.

$$q_k \sim N(\hat{q_k}, Q_\beta) \tag{1.9}$$

where

$$\hat{q_k} = \begin{bmatrix} 0 \\ 0 \end{bmatrix}$$

and

$$Q_\beta = \mathrm{E}(q_k - \hat{q_k})(q_k - \hat{q_k})^T$$

$$Q_\beta = \begin{bmatrix} \sigma^2_{q[\Delta D],k} & \sigma_{q[\Delta\theta],k}\sigma_{q[\Delta D],k} \\ \sigma_{q[\Delta\theta],k}\sigma_{q[\Delta D],k} & \sigma^2_{q[\Delta\theta],k} \end{bmatrix}$$

With the assumption that the noise sources are independent, the off-diagonal elements of the covariance matrix Q_β are equal to zero. The control input, or relative displacement, can be expressed as shown in (1.10).

$$U_k = \begin{bmatrix} \Delta D_k \\ \Delta\theta_k \end{bmatrix} + \begin{bmatrix} q_{[\Delta D],k} \\ q_{[\Delta\theta],k} \end{bmatrix}$$

This makes U_k a random vector. Assuming, that ΔD_k and θ_k are deterministic, uncertainty in U_k equals the uncertainty in the noise term q_k.

The system noise can similarly be modelled by a random noise vector w_k such that, the noise is a Gaussian distribution with zero mean $\hat{w_k}$ and covariance matrix Q_α. This noise source is not directly related to the relative displacement, however it is inherent to the system. Modelling errors, odometry errors, discretization and approximations involved in the derivation of the model all play a part in this noise source. Noise vector w_k is represented in equation (1.10).

$$w_k \sim N(\hat{w_k}, Q_\alpha) \tag{1.10}$$

where

$$\hat{w_k} = \begin{bmatrix} 0 \\ 0 \\ 0 \end{bmatrix}$$

and

$$Q_\alpha = \mathrm{E}(w_k - \hat{w_k})(w_k - \hat{w_k})^T$$

$$Q_\alpha = \begin{bmatrix} \sigma^2_{w[x],k} & \sigma_{w[y],k}\sigma_{w[x],k} & \sigma_{w[\theta],k}\sigma_{w[x],k} \\ \sigma_{w[x],k}\sigma_{w[y],k} & \sigma^2_{w[y],k} & \sigma_{w[\theta],k}\sigma_{w[y],k} \\ \sigma_{w[x],k}\sigma_{w[\theta],k} & \sigma_{w[y],k}\sigma_{w[\theta],k} & \sigma^2_{w[\theta],k} \end{bmatrix}$$

With the assumption that the noise sources are independent, the off-diagonal elements of the covariance matrix Q_α are presumably zero. The system can be expressed as shown below.

$$X_{k+1} = f(X_k, U_k) + w_k \tag{1.11}$$

$$X_{k+1} = \begin{bmatrix} f_x(X_k, U_k) \\ f_y(X_k, U_k) \\ f_\theta(X_k, U_k) \end{bmatrix} + \begin{bmatrix} w_{[x],k} \\ w_{[y],k} \\ w_{[\theta],k} \end{bmatrix} \tag{1.12}$$

Measurement Model

The array of reed sensors mounted in front of the robot provide measurements for the implementation of extended Kalman filter. When a particular sensor detects a magnet, it gets activated. The robot can then determine the position of the magnet by identifying the position of the sensor that sent signal at activation.

It must be noted that several factors, like the sensitivity of the sensor, the magnetic field strength and geometry of the magnet, error in location and orientation as well as others, influence how many sensors are activated at a time instant by the same magnet. Thus, at a given instant multiple sensors may be activated. In the given experimental setup in case of two magnets being detected at the same time two consequential measurements are being conducted.

The position of the array of sensors along x_m is known and fixed, x_d as shown in (1.1). The position of the activated sensor, which is located perpendicular to axis x_m, gives y coordinate of the magnet in the robot frame, d illustrated in (1.1). Therefore the measurement is the position of the magnet in the mobile frame and can be expressed in homogeneous coordinates as shown in (1.13).

$${}^mP_{mag} = \begin{bmatrix} x_d \\ d \\ 1 \end{bmatrix} = \begin{bmatrix} Y \\ 1 \end{bmatrix} \tag{1.13}$$

The position of the magnet with respect to the absolute reference frame can be computed by multiplying ${}^mP_{mag}$ with the transformation matrix ${}^0T_m$.

$${}^0P_{mag} = {}^0T_m \, {}^mP_{mag} \tag{1.14}$$

where

$$^{0}T_{m} = \begin{bmatrix} \cos(x_{[\theta]}) & \sin(x_{[\theta]}) & x_{[x]} \\ -\sin(x_{[\theta]}) & \cos(x_{[\theta]}) & x_{[y]} \\ 0 & 0 & 1 \end{bmatrix} \tag{1.15}$$

The measurement model can therefore be derived by rearranging equation (1.14) into (1.16).

$$\begin{bmatrix} Y_k \\ 1 \end{bmatrix} = \begin{bmatrix} g(X_k) \\ 1 \end{bmatrix} = \begin{bmatrix} \cos(\theta_k) \cdot (^{0}x_{mag} - x_k) + \sin(\theta_k) \cdot (^{0}x_{mag} - y_k) \\ \cos(\theta_k) \cdot (^{0}y_{mag} - y_k) - \sin(\theta_k) \cdot (^{0}y_{mag} - x_k) \\ 1 \end{bmatrix} \tag{1.16}$$

where $^{0}x_{mag}$ and $^{0}x_{mag}$ represent the postion of magnet with respect to the fixed frame. The above model is derived with the assumption that sensors and magnets are point objects and a sensor is activated only if on top of a magnet. However, in praxis the sensor can get activated at varying distance from the magnet. This is taken into consideration by definition of noise in the measurement model.

Measurement model with noise

To deal with the uncertainty in the location of a certain magnet, it is assumed that noise m in measurement of the magnet's location is Gaussian distributed according to

$$m_k \sim N(\hat{m_k}, Q_{\gamma,k}) \tag{1.17}$$

where $\hat{m_k}$ is the zero mean value of the distribution with covariance $Q_{\gamma,k}$.

$$Q_{\gamma,k} = \begin{bmatrix} \sigma^2_{m[x],k} & \sigma_{m[y],k}\sigma_{m[x],k} \\ \sigma_{m[x],k}\sigma_{m[y],k} & \sigma^2_{m[y],k} \end{bmatrix} \tag{1.18}$$

Assuming that the noise sources are independent, the off-diagonal elements of the covariance matrix $Q_{\gamma,k}$ are presumably zero. The final measurement model with noise equation derived from the argument above is shown (1.19)

$$Y_k = g(X_k) + m_k \tag{1.19}$$

Kalman filter

Kalman filter is an algorithm that uses a series of measurements observed over time, containing statistical noise and other inaccuracies, and produces estimates of unknown variables that tend to be more precise than those based on a single measurement alone[1].

A physical system is driven by a set of external inputs or controls. Its outputs are evaluated by sensors such that, the knowledge on the system's behaviour is solely given by the inputs and the observed outputs. For example, in the presented case, the physical system is a mobile robot and given is the problem of localizing it. The control, or the input, is defined by the velocity command given to the left and right wheels, which determines the relative displacement of the robot. The observed output, ergo measurement, is the position of the magnets detected by the robot in absolute coordinate frame. The observations convey the

[1] e-Study Guide for: Introduction to Probability Theory and Stochastic Processes

errors and uncertainties in the process, namely the sensor noise and the system errors. Based on the available information (control inputs and observations), Kalman filter computes an estimate of the system's state by minimizing the variance of the estimation error. From a theoretical standpoint, the main assumptions of the Kalman filter are that the underlying system is a linear dynamical system and that all error terms and measurements have a Gaussian distribution.

Kalman filtering is a recursive analytical technique involving two main steps: prediction and innovation. Prediction is also known as *time update* and refers to prediction of state variables using information from previous state and control input. This step does not involve using measurement. Innovation which is also known as *correction* or *measurement update* refers to estimation of state variable using measurement.

Kalman filter is broadly used in robotics for solving the localization problem, because of its efficiency and accuracy. Some of its characteristics are:

1. Very easy implementation
2. Computationally efficient, updates filter when providing new measurements to the existing data set
3. Uncertainty estimates are provided as part of the filter
4. Recursive algorithm, which makes it suitable for real time applications using only the present input measurements and the previously calculated state.

All members of Kalman family of filters like extended Kalman filter and unscented Kalman filter, comply with a structured sequence of six steps per iteration:

1. **State estimate time update:** An updated state prediction $\hat{x}_{k|k-1}$ is made, based on a priori information and the system model.
2. **Error covariance time update:** The second step is to determine the predicted state-estimate error covariance matrix $P_{k|k-1}$ based on a priori information and the system model.
3. **Estimate system output:** The third step is to estimate the system's output $\hat{z}_k$ corresponding to the timestamp of the most recently received measurement using present a priori information.
4. **Estimator gain matrix:** The fourth step is to compute the estimator gain matrix H_k.
5. **State estimate measurement update:** The fifth step is to compute the a posteriori state estimate $\hat{x}_{k|k}$ by updating the a priori estimate using the estimator gain and the output prediction error.
6. **Error covariance measurement update:** The final step computes the a posteriori error covariance matrix $P_{k|k}$.

Kalman filter equations for linear system

In order to implement Kalman filter first represent the linear system as follows.

$$\begin{aligned} X_{k+1} &= A_k X_k + B_k U_k + w_k \\ Y_k &= C_k X_k + v_k \end{aligned} \tag{1.20}$$

where $w_k \sim N(0, Q_\alpha)$ and $v_k \sim N(0, Q_\gamma)$ as described in System model with noise on page 3 and Measurement Model on page 4 respectively.

For Kalman filter A, B, C, Q_α and Q_γ can be time variant. However, for simplicity it is assumed that they are time invariant for the provided system. Given initial state estimate $\hat{X}_{0|0}$ and initial state covariance matrix $P_{0|0}$ the Kalman filter can be computed using following set of equations.

Prediction (time update)

$$\begin{aligned} \hat{X}_{k+1|k} &= A\hat{X}_{k|k} + BU_k \\ P_{k+1|k} &= AP_{k|k}A^T + BQ_\beta B^T + Q_\alpha \end{aligned} \tag{1.21}$$

Innovation (measurement update)

$$\begin{aligned} H_{k+1} &= P_{k+1|k}C^T {(CP_{k+1|k}C^T + Q_\gamma)}^{-1} \\ \hat{X}_{k+1|k+1} &= \hat{X}_{k+1|k} + H_{k+1}[Y_{k+1} - C\hat{X}_{k+1|k}] \\ P_{k+1|k+1} &= [I - H_{k+1}C]P_{k+1|k} \end{aligned} \tag{1.22}$$

Kalman filter equations for non-linear system

Consider a non-linear extension to the Kalman filter *Extended Kalman Filter (EKF)*. The EKF implements Kalman filter for non-linear system dynamics that result from the linearization of the original non-linear filter dynamics around the previous state estimates.

A non linear system can be represented as follows.

$$\begin{aligned} X_{k+1} &= f\left(X_k, U_k\right) + w_k \\ Y_{k+1} &= g\left(X_{k+1}\right) + v_k \end{aligned} \tag{1.23}$$

where $w_k \sim N\left(0, Q_\alpha\right)$ and $v_k \sim N\left(0, Q_\gamma\right)$.

Another assumption is that $f(\cdot)$, $h(\cdot)$, Q_α and Q_γ are time invariant for the provided non-linear system. $f(\cdot)$ and $h(\cdot)$ are linearized about the prior best estimate of the states at each instant of time by finite difference method in order to compute covariances. Given initial state estimate $\hat{X}_{0|0}$ and initial state covariance matrix $P_{0|0}$ the extended Kalman filter can be computed using following set of equations.

Prediction (time update)

$$\begin{aligned} \hat{X}_{k+1|k} &= f\left(\hat{X}_{k|k}, U_k\right) \\ P_{k+1|k} &= J_A P_{k|k} J_A^T + J_B Q_\beta J_B^T + Q_\alpha \end{aligned} \tag{1.24}$$

Innovation (measurement update)

$$\begin{aligned} H_{k+1} &= P_{k+1|k} J_C^T {(J_C P_{k+1|k} J_C^T + Q_\gamma)}^{-1} \\ \hat{X}_{k+1|k+1} &= \hat{X}_{k+1|k} + H_{k+1}[Y_{k+1} - g\left(\hat{X}_{k+1|k}\right)] \\ P_{k+1|k+1} &= [I - H_{k+1}J_C]P_{k+1|k} \end{aligned} \tag{1.25}$$

where J_A, J_B and J_C are the Jacobian matrices.

J_A is the Jacobian matrix containing the partial derivative of the system function $f(\cdot)$ with respect to the state X, evaluated at the last state estimate $\hat{X}_{k|k}$ and control input U_k.

J_B is the Jacobian matrix containing the partial derivative of the system function $f(\cdot)$ with respect to the control input U_k, evaluated at the last state estimate $\hat{X}_{k|k}$ and control input U_k.

J_C is the Jacobian matrix containing partial derivatives of the measurement function $g(\cdot)$ with respect to the state X, evaluated at the prior state estimate $\hat{x}_{k+1|k}$.

Extended Kalman filter equations for the given system

After deriving physical model of the provided system for System model and understanding the extended Kalman filter in Kalman filter equations for non-linear system, preparation for the EKF equations for odometry can be done.

$$\begin{bmatrix} x_{k+1} \\ y_{k+1} \\ \theta_{k+1} \end{bmatrix} = \begin{bmatrix} f_x(X_k, U_k) \\ f_y(X_k, U_k) \\ f_\theta(X_k, U_k) \end{bmatrix} + \begin{bmatrix} w_{[x],k}) \\ w_{[y],k}) \\ w_{[\theta],k}) \end{bmatrix} \tag{1.26}$$

$$\begin{bmatrix} x_{k+1} \\ y_{k+1} \\ \theta_{k+1} \end{bmatrix} = \begin{bmatrix} x_k + \Delta D_k \cdot \cos(\theta_k) \\ y_k + \Delta D_k \cdot \sin(\theta_k) \\ \theta_k + \Delta\theta_k) \end{bmatrix} + \begin{bmatrix} w_{[x],k}) \\ w_{[y],k}) \\ w_{[\theta],k}) \end{bmatrix} \tag{1.27}$$

Prediction (time update)

$$\begin{bmatrix} \hat{x}_{k+1} \\ \hat{y}_{k+1} \\ \hat{\theta}_{k+1} \end{bmatrix} = \begin{bmatrix} \hat{x}_k + \Delta D_k \cdot \cos(\hat{\theta}_k) \\ \hat{y}_k + \Delta D_k \cdot \sin(\hat{\theta}_k) \\ \hat{\theta}_k + \Delta\theta_k) \end{bmatrix} \tag{1.28}$$

The above equation was implemented in the Evoltionmodel.m.

$$\begin{aligned} J_{A,k} = & \quad \frac{\partial f(X)}{\partial X}\bigg|_{X=\hat{X}_{k|k}, U=U_k} \\ = & \begin{bmatrix} \frac{\partial f_x}{\partial x} & \frac{\partial f_X}{\partial y} & \frac{\partial f_x}{\partial \theta} \\ \frac{\partial f_y}{\partial x} & \frac{\partial f_y}{\partial y} & \frac{\partial f_y}{\partial \theta} \\ \frac{\partial f_\theta}{\partial x} & \frac{\partial f_\theta}{\partial y} & \frac{\partial f_\theta}{\partial \theta} \end{bmatrix}_{X=\hat{X}_{k|k}, U=U_k} \\ = & \begin{bmatrix} 1 & 0 & -\Delta D_k \cdot \sin(\theta_k) \\ 0 & 1 & +\Delta D_k \cdot \cos(\theta_k) \\ 0 & 0 & 1 \end{bmatrix}_{X=\hat{X}_{k|k}, U=U_k} \end{aligned} \tag{1.29}$$

$$\begin{aligned} J_{B,k} = & \quad \frac{\partial f(X)}{\partial U}\bigg|_{X=\hat{X}_{k|k}, U=U_k} \\ = & \begin{bmatrix} \frac{\partial f_x}{\partial \Delta D} & \frac{\partial f_X}{\partial \Delta \theta} \\ \frac{\partial f_y}{\partial \Delta D} & \frac{\partial f_y}{\partial \Delta \theta} \\ \frac{\partial f_\theta}{\partial \Delta D} & \frac{\partial f_\theta}{\partial \Delta \theta} \end{bmatrix}_{X=\hat{X}_{k|k}, U=U_k} \\ = & \begin{bmatrix} \cos(\theta_k) & 0 \\ \sin(\theta_k) & 0 \\ 0 & 1 \end{bmatrix}_{X=\hat{X}_{k|k}, U=U_k} \end{aligned} \tag{1.30}$$

$$P_{k+1|k} = J_A P_{k|k} J_A^T + J_B Q_\beta J_B^T + Q_\alpha \tag{1.31}$$

Innovation (measurement update)

$$\begin{aligned}
J_{C,k} &= \frac{\partial g(X)}{\partial X}\Big|_{X=\hat{X}_{k+1|k}} \\
&= \begin{bmatrix} \frac{\partial g_x}{\partial x} & \frac{\partial g_x}{\partial y} & \frac{\partial g_x}{\partial \theta} \\ \frac{\partial g_y}{\partial x} & \frac{\partial g_y}{\partial y} & \frac{\partial g_y}{\partial \theta} \end{bmatrix}_{X=\hat{X}_{k+1|k}} \\
&= \begin{bmatrix} \sin(\theta_{k+1}) & -\cos(\theta_{k+1}) & -\sin(\theta_{k+1}) \cdot ({}^0y_{mag} - y_{k+1}) - \cos(\theta_{k+1}) \cdot ({}^0x_{mag} - x_{k+1}) \end{bmatrix}_{X=\hat{X}_{k+1|k}}
\end{aligned} \tag{1.32}$$

Using the above derived Jacobian matrix (1.32) for computation of Kalman gain matrix H_k, a posteriori state estimate and a posteriori error covariance matrix following concludes.

$$\begin{aligned}
H_{k+1} &= P_{k+1|k} J_C^T (J_C P_{k+1|k} J_C^T + Q_\gamma)^{-1} \\
\hat{X}_{k+1|k+1} &= \hat{X}_{k+1|k} + H_{k+1}[Y_{k+1} - g\left(\hat{X}_{k+1|k}\right)] \\
P_{k+1|k+1} &= [I - H_{k+1} J_C] P_{k+1|k}
\end{aligned} \tag{1.33}$$

Initialization of extended Kalman filter

Proceeding with the EKF model for odometry, initialization of the model is necessary before practical implementation, which requires following components to be initialized:

1. initial state estimate $\hat{x}_{0|0}$,
2. initial state covariance matrix $P_{0|0}$,
3. process covariance matrices Q_α and Q_β,
4. measurement covariance matrix Q_γ.

Initial state estimate $\hat{X}_{0|0}$

This is the initial estimate of the state vector required for the first iteration of the EKF. For each run, placement of the robot was conducted manually with the initial posture $X_{0|0} = 0$ with the exception of runs diagonal45degrees.txt and line2magnets.txt. There is a certain tolerance in the initial posture, because of human error.

Initial state covariance matrix $P_{0|0}$

Initial state covariance matrix is based on the initialization error of the state. If the initial state estimate is very close to the actual state, this matrix will contain very small values. With the assumption that noise sources for different state elements are independent, the off diagonal elements of the covariance matrix can be assumed to equal zero.

Usually if this matrix is unknown identity matrix is used, since state covariance matrix is updated in every iteration of EKF. For a stable EKF this matrix should be converging over time. An initial estimate closer to the actual state will offer faster convergence of the Kalman filter.

The noise in placement of the mobile platform at its initial position is assumed to be normally distributed with a mean value of zero and variance of $5mm$ represented by σ_x, σ_y. A position error greater than $5mm$ would have been easily noticeable and corrected. For the initial orientation the distribution is also assumed to be normal with a mean value of zero and variance of $18° = 5\% \cdot 360°$ and represented as σ_θ.

$$P_{0|0} = \begin{bmatrix} \sigma^2_{x,0|0} & 0 & 0 \\ 0 & \sigma^2_{y,0|0} & 0 \\ 0 & 0 & \sigma^2_{\theta,0|0} \end{bmatrix} = \begin{bmatrix} 5 & 0 & 0 \\ 0 & 5 & 0 \\ 0 & 0 & \pi/10 \end{bmatrix} \tag{1.34}$$

Process noise covariance matrix Q_α and input noise covariance matrix Q_β

These are the error covariance matrix of the process and control input and give an estimate of uncertainty in the state equations. The uncertainty in the process could be a result of various sources of error, including modelling errors, odometry errors, discretization and approximations involved in the derivation of the model.

In the presented case all system errors are assumed to be on the level of the wheels with a deterministic relation between input covariance matrix Q_β (1.9) and wheel assembly in its entire structure with respect to kinematics.

Direct Kinematic model of a differential drive robot is considered for this. Cartesian velocities of the robot, V and ω can be expressed in terms of right and left wheel velocities $\dot{\varphi_R}$ and $\dot{\varphi_L}$.

$$\begin{bmatrix} V \\ \omega \end{bmatrix} = \begin{bmatrix} r/2 & r/2 \\ r/l & -r/l \end{bmatrix} \begin{bmatrix} \dot{\varphi_R} \\ \dot{\varphi_L} \end{bmatrix} \tag{1.35}$$

where l is the wheel base and r is the radius of wheels. This relation can be discretized as follows.

$$\begin{bmatrix} \Delta D \\ \Delta\theta \end{bmatrix} = \begin{bmatrix} r/2 & r/2 \\ r/l & -r/l \end{bmatrix} \begin{bmatrix} \Delta q_R \\ \Delta q_L \end{bmatrix} \tag{1.36}$$

The noise variances for ΔD and $\Delta\theta$ are indirectly determined by finding the noise variance for the wheels σ_{wheels}. It is assumed that σ_{wheels} is same for both the wheels and is time invariant guaranteeing a deterministic relation, resulting in equation (2.9).

$$Q_\beta = \begin{bmatrix} r/2 & r/2 \\ r/l & -r/l \end{bmatrix} \begin{bmatrix} \sigma^2_{wheels} & 0 \\ 0 & \sigma^2_{wheels} \end{bmatrix} \begin{bmatrix} r/2 & r/l \\ r/2 & -r/l \end{bmatrix} \tag{1.37}$$

σ_{wheels} was determined by trial and error and tuning of extended Kalman filter. It was tuned to the value of 0.045.

Q_α is assumed to be zero, making the approach simpler.

Measurement noise covariance matrix Q_γ

This is the error covariance matrix Q_γ of measurement sensor and provides a measure of how uncertain the measurement is. To get a better approximate of noise in measurement, recorded log files were analyzed based on generated plots showing the number of magnets detected at each step and state of reed sensors.

It was observed that at any time instant no more than two reed sensors have been activated by one magnet (1.2). This suggests that the error localization of the magnet was within ±$5mm$. The probability

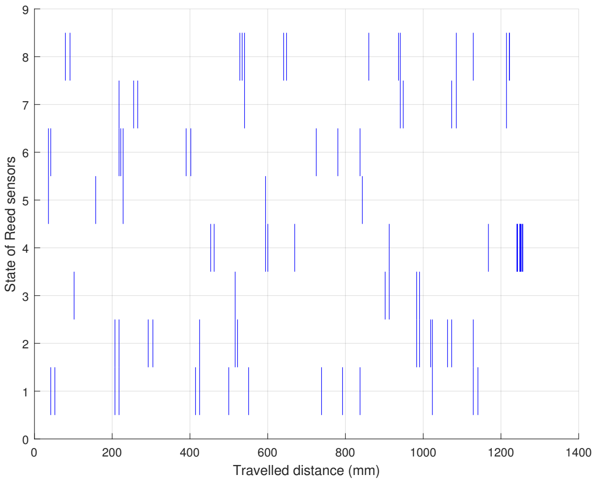


Figure 1.2: State of reed sensors in test run *oneloop* .

of a sensor being activated within a point in this range was equal, representing uniform distribution. The standard deviation for error in y axis $\sigma_{ym} = \frac{\text{reed sensor spacing}}{\sqrt{12}} = \frac{10}{\sqrt{12}}$. For the x axis the standard deviation was determined from the recorded data by measuring the longest active distance using the log files 1line.txt and 2line.txt, resulting in $\sigma_{xm} = \frac{max(\text{longest active distance})}{\sqrt{12}}$. The measurement covariance matrix Q_γ is then defined according to equation (1.38).

$$Q_\gamma = \begin{bmatrix} \sigma_{xm}^2 & 0 \\ 0 & \sigma_{ym}^2 \end{bmatrix} \tag{1.38}$$

Mahalanobis distance

In order to guarantee that the magnet detected by the robot's sensor array has the correct coordinates estimated by the filter with respect to robot posture and is not a result of noise or random errors, Mahalanobis distance (MD) threshold is being used. The determination of the magnet's position in the mobile frame has been already discussed in section *"Measurement Model"*. Knowing the robot's estimated posture, the position of the magnet in the absolute reference frame is estimated using equation (1.16). Since the spacing between the magnets n_d in the square array is known and defined as $n_d = 55mm$, the absolute coordinates of all magnets, and hence also the coordinates of the closest magnet, which is of particular interest here, can be easily determined.

Mahalanobis distance[2] to the magnet detected is computed as shown in equation (1.39) and basically represents how many standard deviations the coordinates of the detected magnet are from the mean of distribution of magnet candidates coordinates. MD is unitless, scale-invariant and sensitive to correlations of the data set.

$$Mahalanobis\ distance = \sqrt{[Y_{k+1} - g\left(\hat{X}_{k+1|k}\right)]^T (J_C P_{k+1|k} J_C^T + Q_\gamma)^{-1} [Y_{k+1} - g\left(\hat{X}_{k+1|k}\right)]} \tag{1.39}$$

Innovation update in the EKF was only carried if the MD to the candidate magnet did not exceed the threshold. Correct tuning of the EKF and identification of the magnet was confirmed by determining if the

[2] Mahalanobis distance

Mahalanobis distance to the four neighbouring magnets did not exceed the threshold.

Mahalanobis distance threshold was computed using the inverse of the chi-square cumulative distribution function[3]. For the selected criteria of possibly rejecting 10% of correct measurements and reject 90% of incorrect ones, the value of $P = 0.9$ is used in the chi2inv for the measurement vector of dimension 2.

Tuning of the extended Kalman filter

Initially all covariance matrices are set to zero in order to run the script and study plots without implementation of extended Kalman filter. This helped with determining measurement covariance matrix Q_γ as previously discussed in the section *"Measurement noise covariance matrix Q_γ"* in detail.

Once all the variances were initialized, the extended Kalman filter was tuned by varying σ_{wheels} in accordance to the algorithm presented in pseudocode:

1. Initialize covariance matrices Q_α, Q_γ and initial state covariance matrix $P_{0|0}$ with values determined in section *"Initialization of extended Kalman filter"*.

2. Set the Mahalanobis distance threshold to the appropriate value taking into consideration the dimension of the measurement vector Y.

3. Tune the σ_{wheels} *parameter* using interval halving method[4] starting with a selected initial value

    (a) While Mahalanobis distance threshold is above all closest magnets for all data sets, halve the σ_{wheels} parameter

    (b) If a closest magnet is rejected, use the difference between last and current step, change computation direction and repeat until Mahalanobis distance threshold is above all closest magnets again as well as below neighbouring magnets.

A value of 0 assigned to σ_{wheels} will lead to zero input covariance matrix Q_β. Since, the process covariance matrix Q_α is also kept zero, this will imply that the odometry based localisation is fully reliable and it can accurately compute the robot posture. However, this was not the case and the measurement was needed to improve robot state estimate.

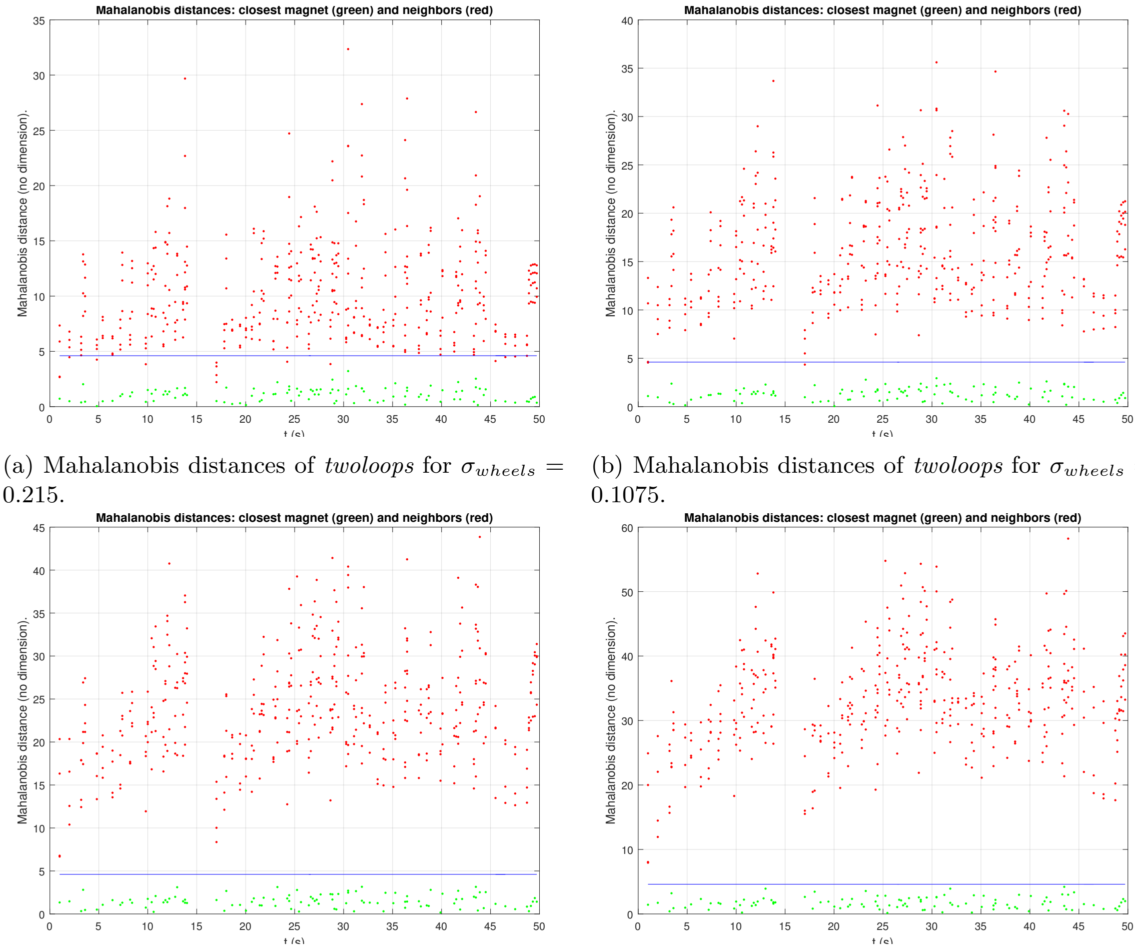
Three main criteria were taken into account for tuning of σ_{wheels}.

1. The Malahanobis distance for all detected magnets during the run should be less than the Malahanobis distance threshold defined *"Mahalanobis distance"* while their neighbouring magnet's Malahanobis distance should be larger than the threshold. This verifies that all measurements are taken into account. In practice the first detected magnet might be used for convergence of the EKF.

2. The estimated path of the robot based on EKF should not contain jumps. However, this is based on the assumption that the skidding and slipping of the robot was negligible during the runs.

3. The estimated path of the robot based on EKF should reach the final position as noted during the experiments. For the test runs oneloop and twoloops the robot returned to its initial position in the end.

Initially, σ_{wheels} was assigned value of 0.215 that corresponded to 1% of the wheel radius. This improved the robot estimated path significantly when compared to the odometry based path. The paths for oneloop and twoloops returned to the robot's initial position in the end. However, for many test runs, the neighbouring magnets distance was found below the Mahalanobis threshold (1.3a). Table 1.1 shows the results for the given value of σ_{wheels}. The 9.09% of closest magnets rejected in diagonal45degrees represent the one magnet that was rejected at the very first iteration of EKF. Since this magnet is rejected before any innovation update, it is not taken into account in analysis of EKF.



(a) Mahalanobis distances of *twoloops* for $\sigma_{wheels} = 0.215$.

(b) Mahalanobis distances of *twoloops* for $\sigma_{wheels} = 0.1075$.

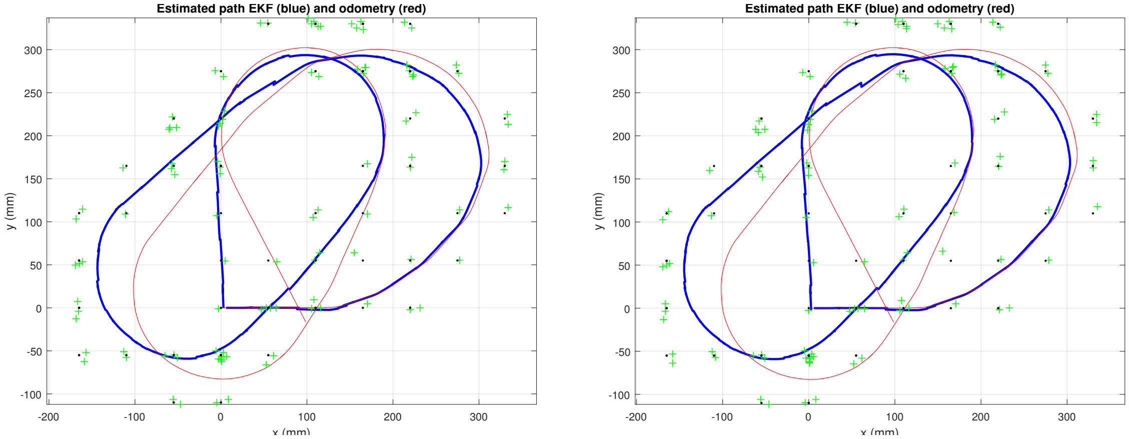
(c) Mahalanobis distances of *twoloops* for $\sigma_{wheels} = 0.0537$.

(d) Mahalanobis distances of *twoloops* for $\sigma_{wheels} = 0.02685$.

Figure 1.3: Mahalanobis distances of *twoloops* using different σ_{wheels}.

Table 1.1: Mahalanobis distacne threshold conditions not satisfied for σ_{wheels} value 0.215

Test run	Percentage of closest magnets rejected	Percentage of neighbour magnets under threshold
circles	0	1.0135
diagonal45degrees	9.0909	4.5455
line1magnet	0	0
line2magnets	0	0.7813
oneloop	0	6.1644
twoloops	0	3.5047



(a) Path estimate of *twoloops* for $\sigma_{wheels} = 0.0537$.

(b) Path estimate of *twoloops* for $\sigma_{wheels} = 0.02685$

Figure 1.4: Path estimate of *twoloops* using different σ_{wheels}.

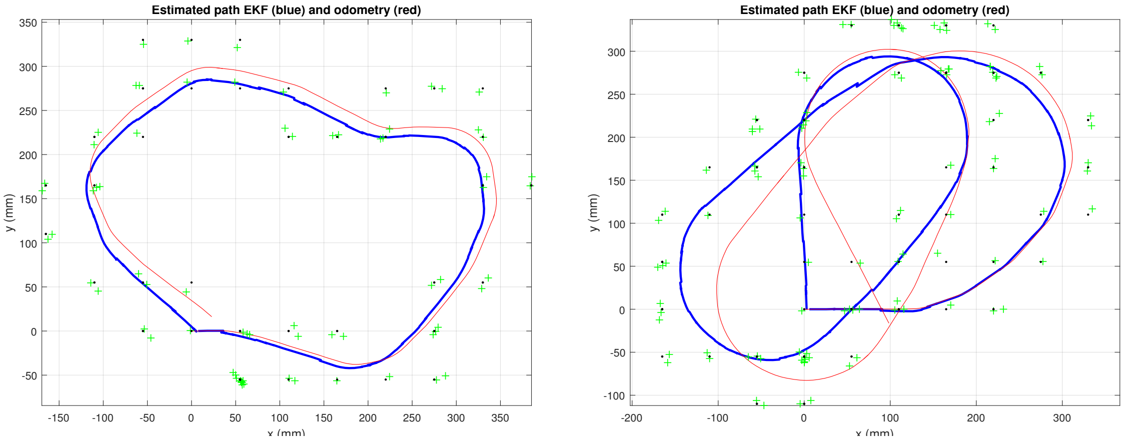
For the next iteration the value for σ_{wheels} is halved to 0.1075. Table 1.2 shows the result for cases the Mahalanobis distance threshold was not satisfied. As can be seen (1.3b), the results are improved by using a smaller value of σ_{wheels}, however there are still neighbouring magnets that do not exceed the threshold limit.

Table 1.2: Mahalanobis distance threshold conditions not satisfied for σ_{wheels} value 0.1075

Test run	Percentage of closest magnets rejected	Percentage of neighbour magnets under threshold
circles	0	0
diagonal45degrees	9.0909	2.2727
line1magnet	0	0
line2magnets	0	0.7813
oneloop	0	0.3425
twoloops	0	0.4673

The value for σ_{wheels} was further halved to 0.0537 and run for the test run logs. No closest magnets were rejected and no neighbouring magnets were found below the Mahalnobis threshold (1.3d). The path estimates using EKF showed consistent response in returning to original position of the robot in oneloop and twoloops tests however small jumps were observable in the path (1.4a). More iterations were performed to find the optimum values for σ_{wheels}.

[3] Chi-square cumulative distribution function
[4] Bisection method



(a) Path estimate of *oneloop*.

(b) Path estimate of *twoloops*.

Figure 1.5: Path estimate of *oneloop* and *twoloops* for tuned value $\sigma_{wheels} = 0.045$.

Further dividing the σ_{wheels} by 2 to value 0.02685 resulted in moving closer to the threshold (1.4b). Also the jumps in the twoloops became more significant (1.4b). This showed decline in the performance of EKF from when the σ_{wheels} was taken to be 0.0537.

Further iterations were carried out for σ_{wheels} between 0.0537 and 0.02685. The plots showing Mahalanobis distances were studied. Reducing σ_{wheels} resulted in moving the neighbour magnets far from the threshold but on the downside, this moved the closest magnets detected closer to the threshold. This risked in closest magnets being rejected in new test runs. A compromise was made between the two conditions to find balance where both the closest magnets and the neighour magnets were not too close to the Mahalanobis distance threshold Final tuned value for σ_{wheels} was taken to be 0.045. Table 1.3 shows the result for cases the Mahalanobis distance threshold was not satisfied for $\sigma_{wheels} = 0.045$.

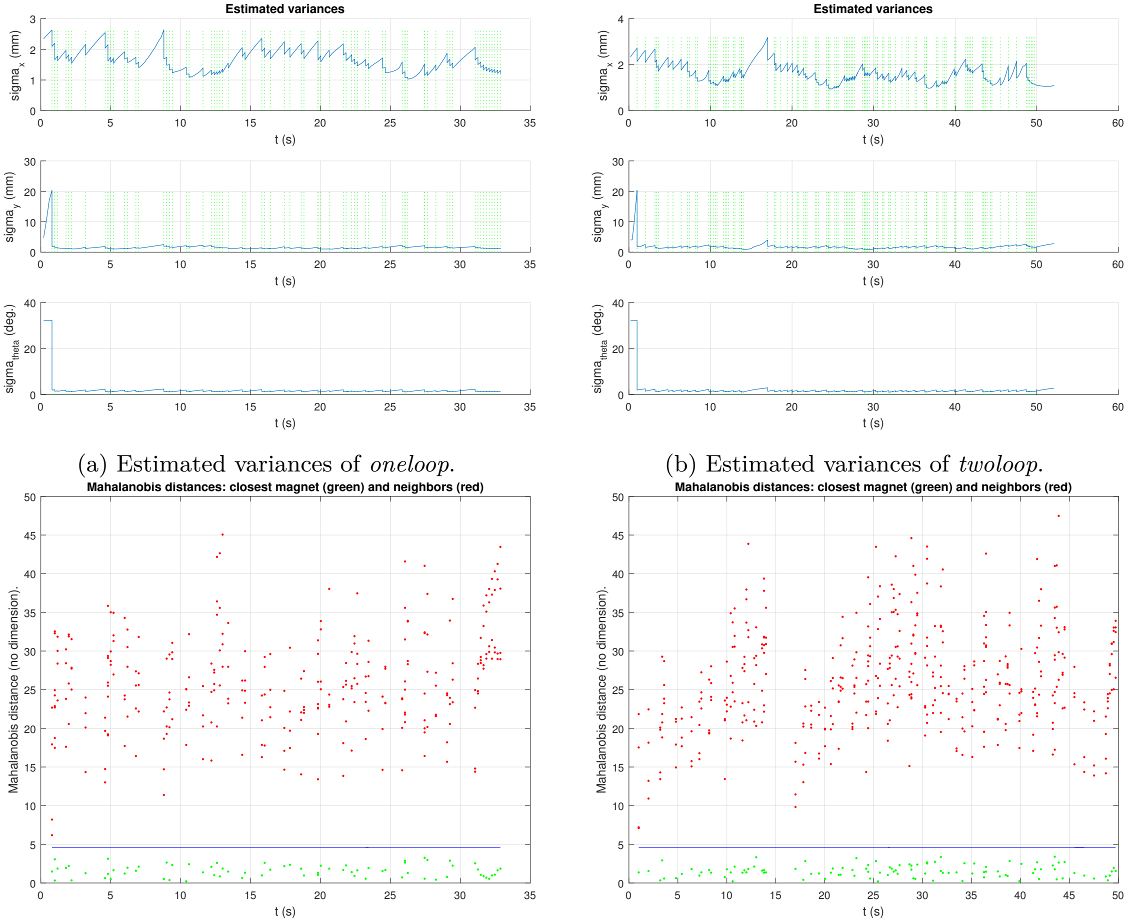
Table 1.3: Mahalanobis distance threshold conditions not satisfied for $\sigma_{wheels} = 0.045$

Test run	Percentage of closest magnets rejected	Percentage of neighbour magnets under threshold
circles	0	0
diagonal45degrees	9.0909	0
line1magnet	0	0
line2magnets	0	0
oneloop	0	0
twoloops	0	0

Localization

Figure (1.5) shows the path estimates for both oneloop and twoloops test runs for the tuned EKF. It can be seen that for both the runs the estimated path returns to the initial position of the robot.

Figure (1.6) shows the variances and Mahalanobis distances for test runs oneloop and twoloops. None of the detected magnets were rejected based on the Mahalanobis distance and all the neighboring magnets exceeded the MD threshold. It can be seen that the variances start to rise when no measurement has been taken for some time period. This is because the odometry generates noise in the position of the robot. The variance for the orientation θ was initialised high but it converges to zero. This shows that error in the value of orientation is very small.



(a) Estimated variances of *oneloop*.

(b) Estimated variances of *twoloop*.

(c) Mahalanobis distances of *oneloop*.

(d) Mahalanobis distances of *twoloops*.

Figure 1.6: Estimated variances and Mahalanobis distances of *oneloop* and *twoloops* for tuned value $\sigma_{wheels} = 0.045$.

Chapter 2

Localization of mobile robots using extended Kalman filter with online state identification

Introduction

Modifications to the problem statement have been introduced. Should a parameter, that is significant for localization, slowly evolve over time or be difficult, maybe even impossible, to measure, another approach is required to solve the localization problem. One technique involves adding the parameter to the state vector in order to identify it online, however practical applications might have difficulties with observability. This requires a new evolution equation incorporating the identified parameters, which is assumed to be constant for very slowly changing parameters. With initial and state noise the parameter can then evolve iteratively. In this case the variables of interests are the wheel radii, which are assumed to be slowly time variant. There are several phenomena that could introduce such changes, like differences between wheels in tire pressure, temperature, wear and other mechanical aspects.

System model and extended Kalman filter equation

In order to identify the parameters of the robot online, in this case the radii of both wheels, the parameters are added to the state vector. State variables are defined in (2.1)

$$X_k = \begin{bmatrix} x_k \\ y_k \\ \theta_k \\ r_{R_k} \\ r_{L_k} \end{bmatrix} \tag{2.1}$$

The control input for the improved system consists of wheel velocities of the robot. This is done so that the radius of the wheels become part of th function to estimate next posture of the robot and thus can be updated by the EKF.

$$U_k = \begin{bmatrix} \Delta q_{Rk} \\ \Delta q_{Lk} \end{bmatrix} \tag{2.2}$$

ΔD and $\Delta\theta$ can be computed from the control input as shown in (2.3).

$$\begin{bmatrix}\Delta D\\ \Delta\theta\end{bmatrix} = \begin{bmatrix}\frac{r_R}{2} & \frac{r_L}{2}\\ \frac{r_R}{l} & -\frac{r_L}{l}\end{bmatrix} U \tag{2.3}$$

Given X_k and U_k, the next location of the robot X_{k+1} can be predicted as follows.

$$\begin{bmatrix}x_{k+1}\\ y_{k+1}\\ \theta_{k+1}\\ r_{R_{k+1}}\\ r_{L_{k+1}}\end{bmatrix} = \begin{bmatrix}x_k + (\frac{r_{R_k}}{2}\Delta q_{R_k} + \frac{r_{L_k}}{2}\Delta q_{L_k})\cdot\cos(\theta_k)\\ y_k + (\frac{r_{R_k}}{2}\Delta q_{R_k} + \frac{r_{L_k}}{2}\Delta q_{L_k})\cdot\sin(\theta_k)\\ \theta_k + (\frac{r_{R_k}}{l}\Delta q_{R_k} - \frac{r_{L_k}}{l}\Delta q_{L_k})\\ r_{R_k}\\ r_{L_k}\end{bmatrix} \tag{2.4}$$

For EKF prediction update the Jacobian matrices are derived again.

$$\begin{aligned} J_{A,k} = & \left.\frac{\partial f(X)}{\partial X}\right|_{X=\hat{X}_{k|k},U=U_k}\\ = & \begin{bmatrix}1 & 0 & -\Delta D_k\cdot\sin(\theta_k) & \frac{\cos(\theta_k)\Delta q_{R_k}}{2} & \frac{\cos(\theta_k)\Delta q_{L_k}}{2}\\ 0 & 1 & +\Delta D_k\cdot\cos(\theta_k) & \frac{\sin(\theta_k)\Delta q_{R_k}}{2} & \frac{\sin(\theta_k)\Delta q_{L_k}}{2}\\ 0 & 0 & 1 & \frac{\Delta q_{R_k}}{l} & -\frac{\Delta q_{L_k}}{l}\\ 0 & 0 & 0 & 1 & 0\\ 0 & 0 & 0 & 0 & 1\end{bmatrix}_{X=\hat{X}_{k|k},U=U_k}\end{aligned} \tag{2.5}$$

$$\begin{aligned} J_{B,k} = & \left.\frac{\partial f(X)}{\partial U}\right|_{X=\hat{X}_{k|k},U=U_k}\\ = & \begin{bmatrix}\frac{r_{R_k}\cos(\theta_k)}{2} & \frac{r_{L_k}\cos(\theta_k)}{2}\\ \frac{r_{R_k}\sin(\theta_k)}{2} & \frac{r_{L_k}\sin(\theta_k)}{2}\\ \frac{r_{R_k}}{l} & -\frac{r_{L_k}}{l}\\ 0 & 0\\ 0 & 0\end{bmatrix}_{X=\hat{X}_{k|k},U=U_k}\end{aligned} \tag{2.6}$$

The measurement equation of the system remains the same. However, the Jacobian matirx J_C needs to be recomputed.

$$J_{C,k} = \begin{bmatrix}-\cos(\theta_{k+1}) & -\sin(\theta_{k+1}) & -\sin(\theta_{k+1})\cdot({}^0x_{mag} - x_{k+1}) + \cos(\theta_{k+1})\cdot({}^0y_{mag} - y_{k+1}) & 0 & 0\\ \sin(\theta_{k+1}) & -\cos(\theta_{k+1}) & -\sin(\theta_{k+1})\cdot({}^0y_{mag} - y_{k+1}) - \cos(\theta_{k+1})\cdot({}^0x_{mag} - x_{k+1}) & 0 & 0\end{bmatrix}_{X=\hat{X}_{k+1|k}} \tag{2.7}$$

Initialization of extended Kalman filter

Initial state estimates for the position and orientation of the robot were kept same as in the previous section. The initial state estimates of the radii of the wheels were assigned wheel radius used in the simulation by default plus $0.25mm$.

For initial state covariance matrix $P_{0|0}$ the variances for noise in position and orientation were kept unchanged. Variances for the noise in radii of the wheels were tuned. $P_{0|0}$ is now defined as follows.

$$P_{0|0} = \begin{bmatrix} \sigma^2_{x,0|0} & 0 & 0 & 0 & 0 \\ 0 & \sigma^2_{y,0|0} & 0 & 0 & 0 \\ 0 & 0 & \sigma^2_{\theta,0|0} & 0 & 0 \\ 0 & 0 & 0 & \sigma^2_{r_R,0|0} & 0 \\ 0 & 0 & 0 & 0 & \sigma^2_{r_L,0|0} \end{bmatrix} \tag{2.8}$$

Input noise covariance matrix Q_β is changed since the control input has been changed. Q_β is now defined as follows.

$$Q_\beta = \begin{bmatrix} \sigma^2_{wheel_R} & 0 \\ 0 & \sigma^2_{wheel_L} \end{bmatrix} \tag{2.9}$$

σ_{wheel_R} and σ_{wheel_L} are same as represented by σ_{wheels} in section *"Process noise covariance matrix Q_α and input noise covariance matrix Q_β"*. They are assigned the same value tuned previously and are not tuned again in this application.

Process covariance matrix Q_α was assigned value of zero previously. Now, with the introduction of robot parameters in the state vector, Q_α is defined as follows.

$$Q_\alpha = \begin{bmatrix} 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & \sigma^2_{\alpha_{r_R}} & 0 \\ 0 & 0 & 0 & 0 & \sigma^2_{\alpha_{r_L}} \end{bmatrix} \tag{2.10}$$

The values for $\sigma^2_{r_R}$ and $\sigma^2_{r_L}$ were tuned. Measurement noise covariance matrix Q_γ remains unchanged.

Tuning of the extended Kalman filter with online state identification

First, all variances were set to zero and the plots were generated to confirm that the path estimate based on EKF overlayed on the odometry based path estimate. This confirmed that the EvolutionModel was correctly defined.

Second, all the variances determined and tuned using offline tuning methods were implemented. This included values for measurement covariance matrix Q_γ, $\sigma^2_{x,0|0}$, $\sigma^2_{y,0|0}$ and $\sigma^2_{\theta,0|0}$ for the initial state covariance matrix $P_{0|0}$ as well as σ_{wheels} ($\sigma_{wheels} = \sigma_{wheel_R} = \sigma_{wheel_L}$). The initial noise in the wheel radii, $\sigma^2_{r_R,0|0}$ and $\sigma^2_{r_L,0|0}$ in $P_{0|0}$ and $\sigma^2_{\alpha_{r_R}}$ and $\sigma^2_{\alpha_{r_L}}$ in Q_α were kept zero. Plots were generated and it was verified that the same results were achieved as in the offline method. This confirmed that the new Jacobian matrices were defined correctly.

Next, the tuning of EKF with robot parameters is conducted. Proceeding with the variances set above, $\sigma^2_{r_R,0|0}$ and $\sigma^2_{r_L,0|0}$ in $P_{0|0}$ were assigned the value of 20% of the wheel radius. They were set to relatively high value so that initial estimate of error was significant and convergence was modelled. It was observed that the values in state covariance matrix were initially large however with time they nearly converge to zero. The convergence of values for $\sigma^2_{r_R,0|0}$ and $\sigma^2_{r_L,0|0}$ in $P_{0|0}$ can be observed in plots in figure (2.1).

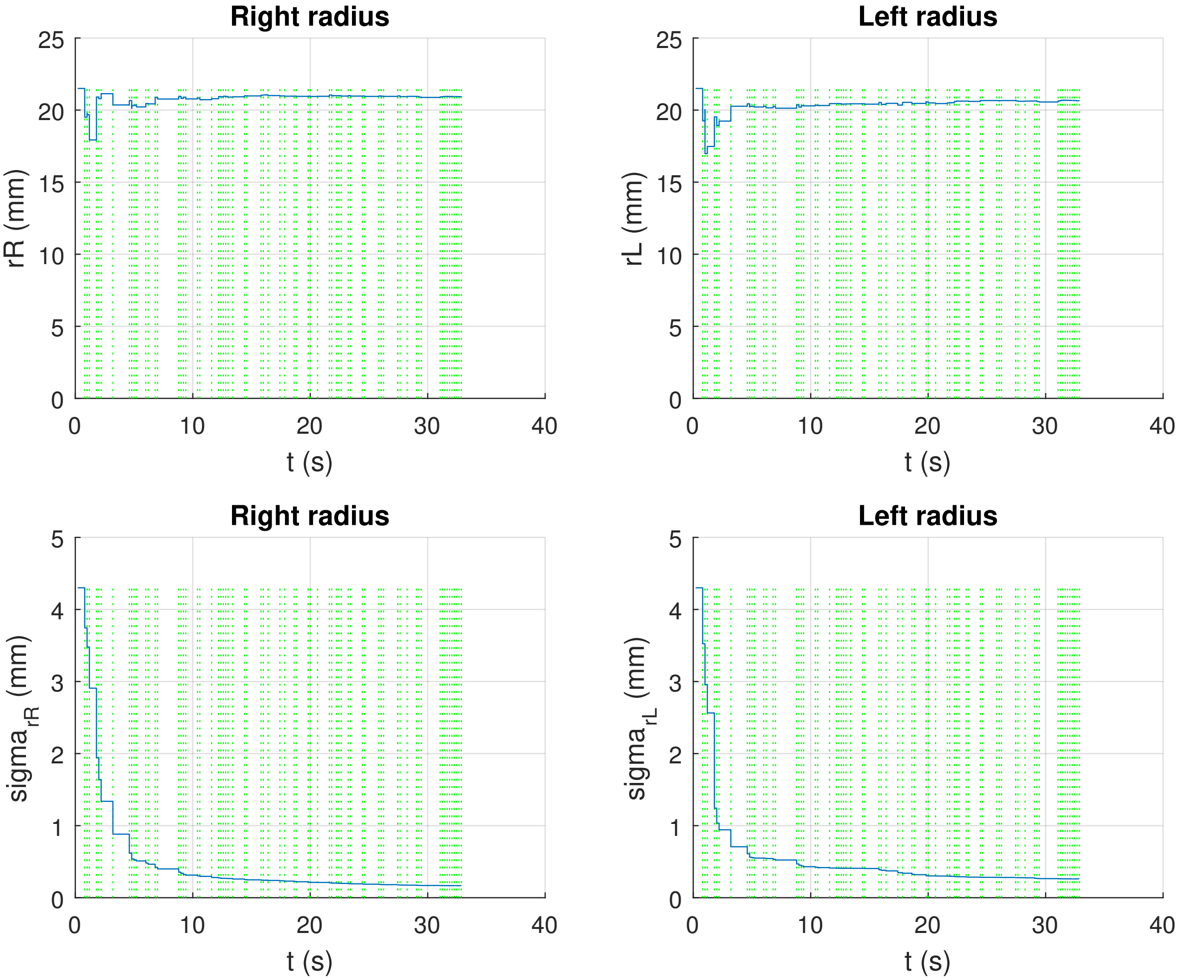


Figure 2.1: Evolution of right and left wheel radii and σ_{r_R} and σ_{r_L} when $\sigma^2_{r_R,0|0}$ and $\sigma^2_{r_L,0|0}$ are set high and nominal values are taken for the wheel radius in test run *oneloop* .

It was observed that the values in state covariance matrix were initially large however with time they nearly converge to very small value. The convergence of values for σ_{r_R} and σ_{r_L} in P can be observed in plot (2.1) for oneloop test run. The change in variance for the wheel radii resulted in convergence of radius for both wheels. The plots were studied for all six test runs and it was observed that the both right and left wheel radii converged to values in range $20.75 \div 21.75mm$. Based on this, a small error was added to right and left wheel radii for the nominal wheel radius was given $21.5mm$.

Finally $\sigma^2_{\alpha_{r_R}}$ and $\sigma^2_{\alpha_{r_L}}$ in Q_α were tuned. Because of the previous experience, trial and error method was mostly based on the intuition and observation. However, it was also conducted with help of bisection method. The same criteria was set for tuning as listed in section *"Tuning of the extended Kalman filter"* and is not repeated here.

During initial tuning, it was observed that for first few iterations of EKF, Mahalanobis distance of some neighbouring magnets did not exceed the threshold. This is illustrated in one of the plots for twoloops run shown in (2.2). This was due to the high value in initial state noise of wheel radii. Thus the values for $\sigma^2_{r_R,0|0}$ and $\sigma^2_{r_L,0|0}$ in $P_{0|0}$ was reduced to 0.5.

For the small error in the wheel radii, it was observed that the system performed best for $Q_\alpha = 0$. In

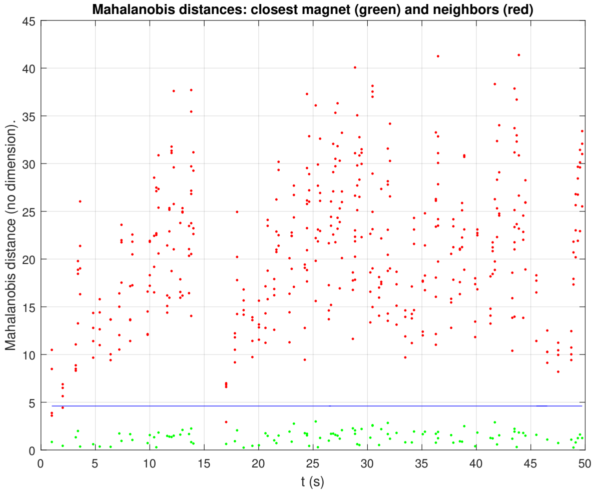


Figure 2.2: Mahalanobis distances for $\sigma_{wheels} = 4.3$ in test run *twoloops*.

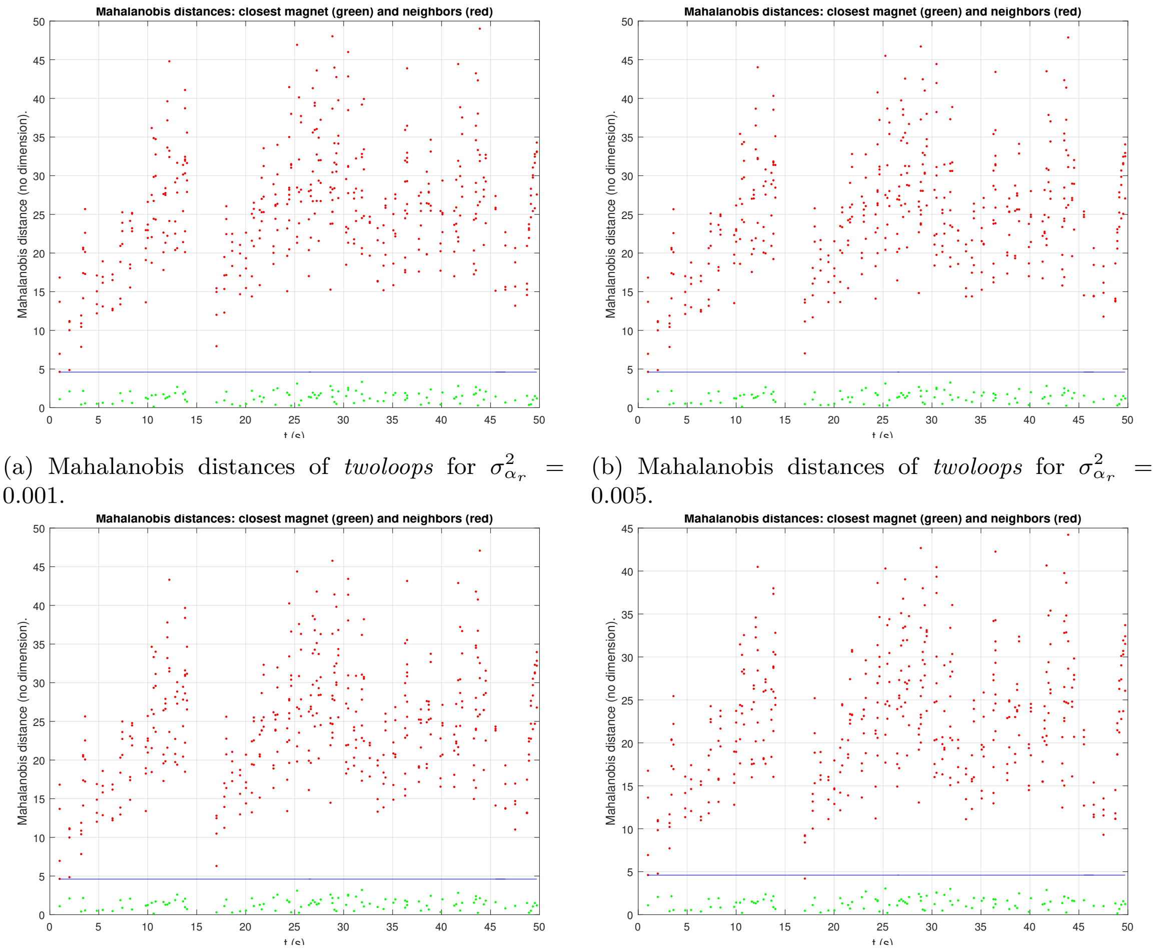
order to simulate tuning of the system the initial wheel radii were taken as follows:

- Right wheel radius = nominal wheel radius $+3 = 24.5mm$
- Left wheel radius = nominal wheel radius $-3 = 18.5mm$

Figure (2.3) shows Mahalanobis distances of twoloops using different $\sigma^2_{\alpha_r}$ for the above wheel radius. It is visible that the small value of $\sigma^2_{\alpha_r}$ shows the best result. Final tuned value for it was kept to be 0.001.

Figure (2.4) shows the path estimates for both oneloop and twoloops test runs for the tuned EKF with robot parameters. Comparing these plots with ones resulted with offline tuning method, it can be observed that initially the path estimate contains jumps. However, with time, with the tuning of wheel radii, the path becomes smoother with very few jumps.

Figure (2.5) shows the wheel radii, wheel radii variances and Mahalanobis distances for test runs oneloop and twoloops. Performance of the system has reduced in context of Mahalanobis distances. The neighbouring magnets are closer to the threshold in this system than with offline tuning method. Figures (2.5b) and (2.5b) show that the error in the wheel radii converge to a steady value witin first 10 seconds. This demonstrates the tuning of wheel radii using extended Kalman Filter.



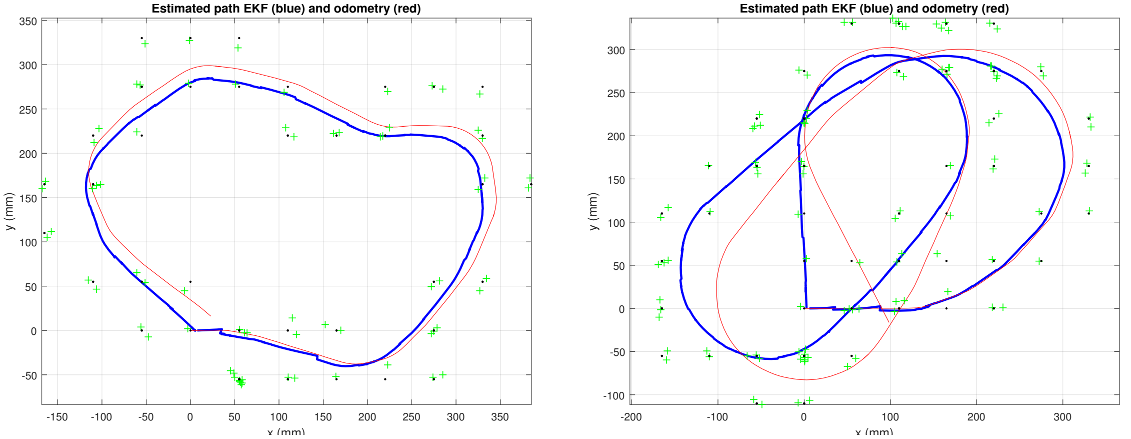
(a) Mahalanobis distances of *twoloops* for $\sigma^2_{\alpha_r} = 0.001$.

(b) Mahalanobis distances of *twoloops* for $\sigma^2_{\alpha_r} = 0.005$.

(c) Mahalanobis distances of *twoloops* for $\sigma^2_{\alpha_r} = 0.01$.

(d) Mahalanobis distances of *twoloops* for $\sigma^2_{\alpha_r} = 0.05$.

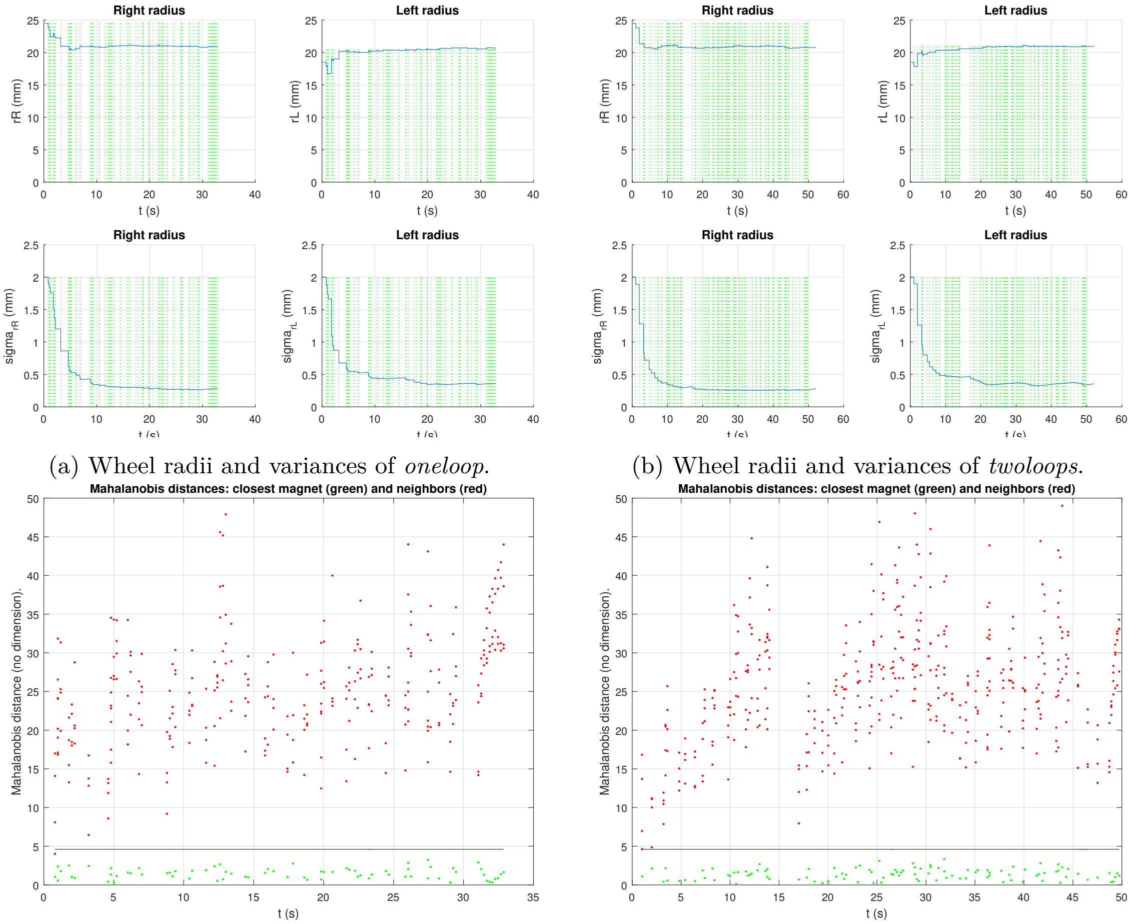
Figure 2.3: Mahalanobis distances of *twoloops* using different $\sigma^2_{\alpha_r}$.



(a) Path estimate of *oneloop*.

(b) Path estimate of *twoloops*.

Figure 2.4: Path estimate of *oneloop* and *twoloops* for tuned value $\sigma_{wheels} = 0.045$.



(a) Wheel radii and variances of *oneloop*.

(b) Wheel radii and variances of *twoloops*.

(c) Mahalanobis distances of *oneloop*.

(d) Mahalanobis distances of *twoloops*.

Figure 2.5: Wheel radii and variances and Mahalanobis distances of *oneloop* and *twoloops* for tuned system.

Conclusions

From the validation and simulation parts it can be concluded that localization using the extended Kalman filter can achieve quite satisfiable results and gives a possibility to deal with time variant or difficult to measure parameters.

In the first part of the laboratory no robot parameters were introduced to the state vector. The odometry model, as expected, accumulated the systematic and random errors, which resulted in a significant error of the posture, making reliance on the odometry alone for the localization problem a poor choice. The extended Kalman filter provided correct results after proper tuning of σ_{wheels} using different criteria of Mahalanobis distances, smoothness of the EKF estimated path and the final position in test runs with a loop path as described in detail in *"Tuning of the extended Kalman filter"*.

In the second part of the laboratory state noise was used. The initial tuning of EKF was performed in the same way as previously. Section *"Tuning of the extended Kalman filter with online state identification"* provides greater detail on the tuning process. The final values of the state vector X compared to the initial ones $X_{0|0}$, for all test runs using the tuned values for EKF, are almost equal for the test run twoloops. The longer the EKF worked, the more the state values for the radii converge towards equal values, because of a higher number of updates, which is the reason for using twoloops. This indicates that the experimental mobile platform had little difference in the wheel radii. In fact for different initial state values for the wheel radii within range of ±5% of as well as $\pm 3mm$ to the nominal value, the state vector evolved to values $X(4) = r_{R_k} = 20.8281$ and $X(5) = r_{L_k} = 20.8105$.

Concluding presented results, it can be stated that the extended Kalman filter is a very reliable tool for localization of mobile robots.