Mobile Robot Localisation for Indoor Environment

Lim Chee Wang

Dr Lim Ser Yong

A/Prof Marcelo H Ang Jr

(Mechatronics Group, 2002)

1 BACKGROUND

A mobile robot must know its whereabout in order to navigate reliably. Therefore, the ability to estimate their position is one of the basic preconditions for the autonomy of mobile robots. The Nomad XR4000 is able to report its relative and global position at any instance. The positioning method used in Nomad XR4000 is based on counting the revolutions of the wheel, this method is commonly known as *odometry*. Unfortunately, wheel slippage and drift occurs in the mobile robot, hence this method produces position errors that are incremental. Section 3.2.1 discusses some of the known positioning errors experienced by the mobile robot.

This problem is, in fact inevitable and common to most mobile robot but to different extents. Various methods are explored by researchers in the past to address the position error problem. The solution to this problem can be classified into two main groups: *odometry error correction* [1, 2, 3] and *environment/landmark observation* [4, 5, 6, 7, 8, 9].

2 OBJECTIVE

The results reported here is a portion of the work in mobile robot motion planning. The main objective in mobile robot motion planning is to achieve high level of autonomy in the robot. The capability of the mobile robot to localise itself is an essential building block in motion planning in mobile robotics.

This report presents the implementation of the *Kalman Filter Localisation* algorithm to an existing mobile base robot, the Nomad XR4000. This implementation allows the robot to localise or correct its position based on the environmental features that it senses, as well as some *priori* knowledge of the environment.

3 METHODOLOGY

3.1 Environment Feature Extraction

Almost all localisation methods are based on matching the actual (detected by sensors)environment feature surrounding the robot to the map known prior by the robot. This map can be a static predetermined map, or a history map generated by the robot while moving. The type of environmental features to be observed by the robot depends on what type of mapping method used, i.e. *Geometric* or *Topological* mapping. In the case of *Topological* map, the map is built using features like walls, doors, hallways, etc. Likewise, it is obvious that the features to be observed by the robot are those features described by the map. Conversely, in *Geometric* mapping methods, the map is only described by the certainty of an obstacle occupying a grid cell, i.e. the map is independent and ignorant of the type of features surrounding the robot. Therefore, the features to be observed by the robot can be anything. Corners and straight lines are most commonly used.

3.1.1 Detecting Corners around the Robot

As *Geometric* mapping method is implemented in this work, corners in the environment are used to localise the robot. Corners are usually perceived as discontinuity in straight lines, and intersection of two straight lines with different gradients. In most indoor environments, the difference in gradients of these two intersecting lines is generally about 1 (or 90 degrees). Figure 1 shows an example of corners observed by the robot.

It should be noted that not all discontinuity are corners, some could be table legs, people, wire trunking on walls or chairs. These discontinuity are undesirable, as they are usually not reflected in the map. To avoid this mistake, the following criteria have to be met in order for an obser-

Keywords: Mobile Robot; Localisation; Kalman Filter; Feature Detection

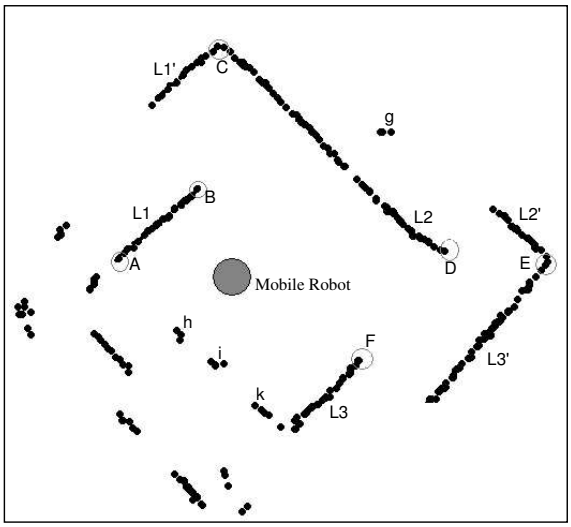


Figure 1. Points that are circled are corners observed

vation to be considered a corner:

a. Only discontinuity between long straight lines would be considered. The length of the straight lines depend on the environment as well as the resolution and accuracy of the sensors used to observe the surrounding.

b. Discontinuity between two same straight lines (i.e. the line equations of both straight lines are the same or very close) would not be considered as corners. These discontinuities are usually noise or simply a small gap between cabinets. (Point g in Figure 1 illustrates this condition)

c. When a discontinuity occurs between two *different* straight lines (for example L1 and L1', L2 and L2', and L3 and L3' in Figure 1), the point nearer to the robot is considered a corner; while the point further away is treated to be unknown. This is due to the fact that the "real" corner belonging to the further straight line (represented by L1', L2' and L3' in Figure 1) might not be in the line of sight of the robot's sensors. Point B, D and F in Figure 1 illustrate this condition.

d. Note that Point A in Figure 1 does not fit the previous condition, as the discontinuity does not occur between two straight lines. A straight line that is discontinued with the subsequent points further away from the robot (same concept as the previous condition) and does not form another straight line can be perceived as the end of a straight line, thus a corner too.

The following section describes the method used here in obtaining continuous straight lines from the environment using a laser range finder.

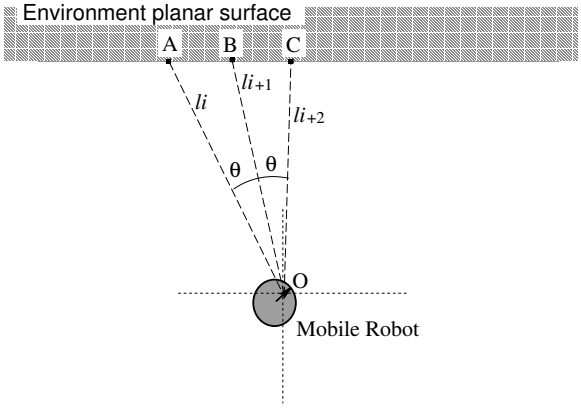


Figure 2. Relationship between successive laser range readings when scanning a planar surface

3.1.2 Detecting Continuous Straight lines in the Environment

Consider the general planar surface shown in Figure 2 and the corresponding sensed data points which would result from a perfect 2D line of sight laser sensor. Application of the Sine and Cosine Rules to the two triangles OAB and OBC in Figure 2, and using the fact that the three points A, B and C all lie within a vertical plane (assuming all the laser rangefinder scans a horizontal plane), it is possible to show the relationship between successive range readings l_i, when the laser beam is incident upon a planar surface (or wall). This is given by:

$$l_{i+2} = \frac{l_i l_{i+1}}{(2l_i cos\theta) - l_{i+1}} \qquad (1)$$

where θ is the constant angle (in bear-

ing) between successive samples of the sensor as it rotates about its vertical axis. Note that this formulation is valid for a perfect sensing device, in actual implementation an uncertainty of about $\pm$5mm is considered.

3.1.3 Corners Matching

The corners recognition methods discussed above applies to the corners in the map as well. In order to achieve localisation based on corners in the environment, it is important to match the observed corners to the predetermined corners correctly. Incorrect matching will result in disastrous errors in localising the robot.

The *corners matching* method is similar to the detection method for corners in the actual environment. As discussed, the determination of corners depends on the straight lines detected from the environment; it will therefore be useful to associate a corner with a straight line. In Figure 3, we will only consider corners A, B, C, D and E, which are within the sight of the robot laser sensor. Points a', b', c', d' and e' are corresponding corners in the predetermined map. For consistency, the corners are associated to the straight lines on their left whenever possible. Note that in the case of corner a', this is not possible, therefore the straight line l_1 is shared by corners a' and b'.

The localization work here assumes that the robot is near to the predicted location based on odometry. Thus, besides matching a corner (for example c') on the map to an observed corner (for example C) by the sensor in its vicinity, the matching procedure will also match the associated straight line and planar surface located by the laser sensor (l_2 and L2 respectively). This straight line matching thus verifies that the corners matching between c' ad C is correct.

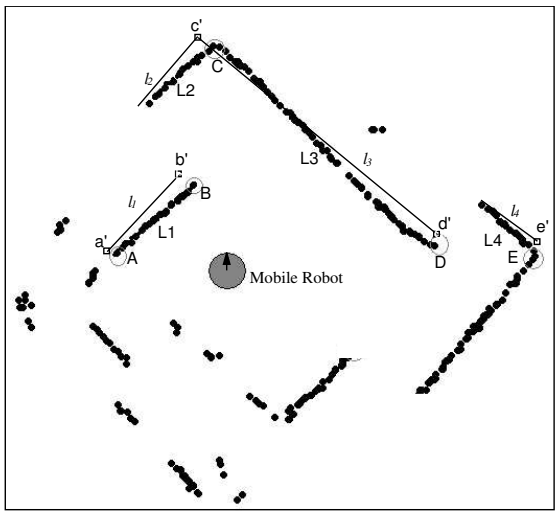


Figure 3. Matching detected corners to corners in predetermined map

3.2 Extended Kalman Filter Localisation

The Kalman filter [10] is useful for estimating the state of a system based on recursive measurement of noisy data. In fact, one can think of the noisy data as a multidimensional signal plus noise, with the system states being the desired unknown signals. The Kalman filter then filters the noisy measurements to estimate the desired signals. The estimates are statistically optimal in the sense that they minimize the mean-square estimation error.

Figure 4 illustrates a simple Kalman Filter algorithm. $\hat{x}(k+1|k)$ is the state vector of the system, where the hat denotes an estimate, and this estimate is one prior to incorporation of the measurements. $P(k+1|k)$ is the error covariance of the state estimate. $z(k+1)$ is the new measurements or observation obtained by the sensors. $H(k+1)$ is the *measurement matrix* or *observation matrix* relating $x(k+1)$ to $z(k+1)$ in the absence of measurement noise. In the updated state estimates in Figure 4, the difference $[z(k+1) - H(k+1)\hat{x}(k+1|k)]$ is call the measurement *innovation*, or the *residual*. This residual reflects the discrepancy between the predicted mea-

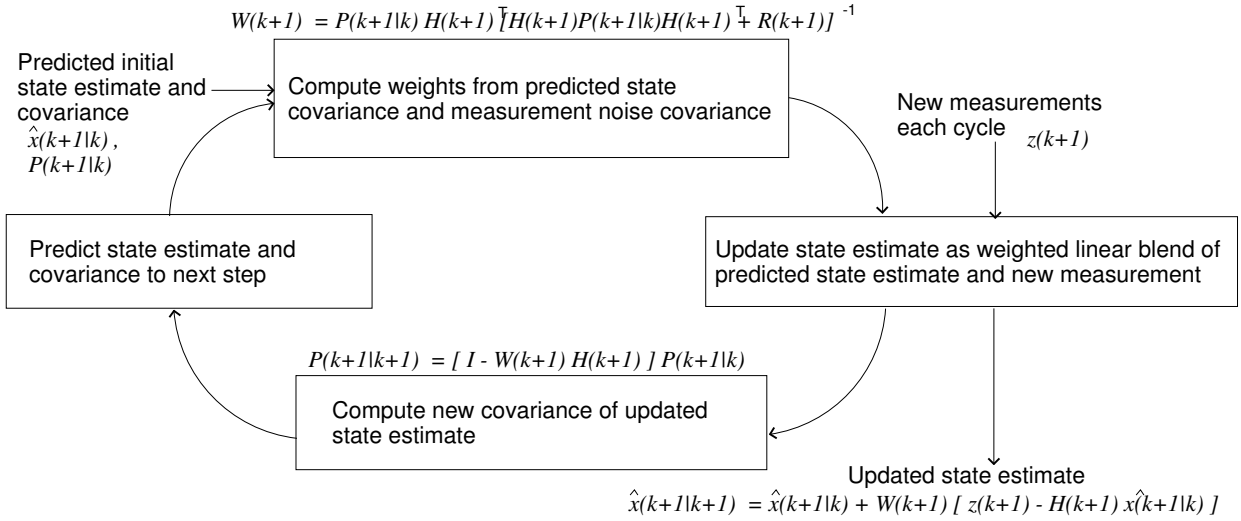


Figure 4. The Kalman filter algorithm

surement $H(k+1)\hat{x}(k+1|k)$ and the actual measurement $z(k+1)$. At the initial stage, a state estimate and its covariance are predicted/estimated. The state estimate should converge to a better estimate as the cycle runs, as illustrated in Figure 4.

The *Kalman gain* used in the algorithm is represented by $W(k+1)$, which is a recursive weighted least square method used to optimize the estimation. This optimization of estimates refers to the minimizing of error covariance, P [10]. $R(k+1)$, the error covariance associated with the measurement $z(k+1)$, is also one of the weighting factor used in computing the *Kalman gain*. To understand more on the weighting principle of the *Kalman gain*, we look at the *gain* function and realized that as the measurement error covariance R approaches zero, the gain W weights the *residual* more heavily, as shown below

$$\lim_{R(k+1)\to 0} W(k+1) = H(k+1)^{-1} \quad (2)$$

On the other hand, as the state error covariance P approaches zero, the *gain* W weights the *residual* less heavily, as shown below

$$\lim_{P(k+1|k)\to 0} W(k+1) = 0 \quad (3)$$

In a nut shell, the weight of the Kalman gain is determined by the measurement and predicted errors. When the measurement error reduces, the actual measurement $z(k+1)$ is “trusted" more, while the predicted measurement $H(k+1)\hat{x}(k+1|k)$ is “trusted" less. In contrast, as the predicted error reduces, the actual measurement is “trusted" less, while the predicted measurement is “trusted" more.

3.2.1 Mathematical Formulation

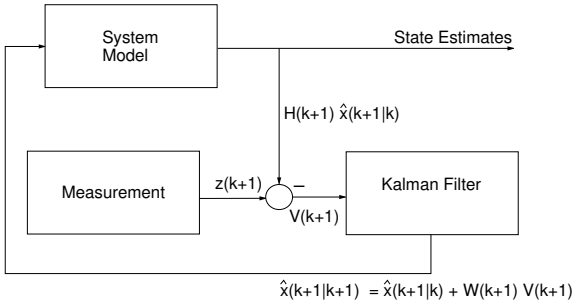


Figure 5. The Extended Kalman Filter can be visualized as a feedback configuration shown in this block diagram

Since the measurement model is nonlinear (discuss in detail in the following section), therefore a *Linear Kalman Filter* cannot be implemented [11, 10], an *Extended Kalman Filter* (EKF) algorithm [12, 13, 10] is used instead. The only dif-

ference between the EKF and the LKF is that the Kalman filter linearizes about both the current mean and covariance. In our work, only the measurement relationship to the state is non-linear, therefore linearizing is needed only for the measurement error covariance. In the remaining of this section the formulation of various models needed for the computation of the Kalman gain as well as the prediction results are presented.

System Model

The system model describes how the vehicle's position $x(k)$ changes with time in response to a control input $u(k)$ and a noise disturbance $v(k)$. It has the form:

$$x(k+1) = F(x(k), u(k)) + v(k) \quad (4)$$

where $F(x(k), u(k))$ is the state transition function. The mobile robot used in this work is *Holonomic*, therefore the state transition is a linear one. Thus, the control input is simply,

$$u(k) = \begin{bmatrix} \triangle\theta(k) \\ \triangle x(k) \\ \triangle y(k) \end{bmatrix} \quad (5)$$

and the transition function has the form,

$$F(x(k), u(k)) = \begin{bmatrix} \theta(k) + \triangle\theta(k) \\ x(k) + \triangle x(k) \\ y(k) + \triangle y(k) \end{bmatrix} \quad (6)$$

Observation Model

The robot uses the environment information around it to deduce its geometrical location relative to a certain feature/features. In our work, we use corners to help us localise the robot.

The robot uses a laser range-finder to scan its surrounding. Every data scan obtain by the robot is described as range and bearing relative to the robot itself. Figure 6 shows the robot's perceptions of a corner in the environment at different positions; α is the bearing of the corner with respect to the heading direction of the robot, and d represent the distance or range of the corner to the robot.

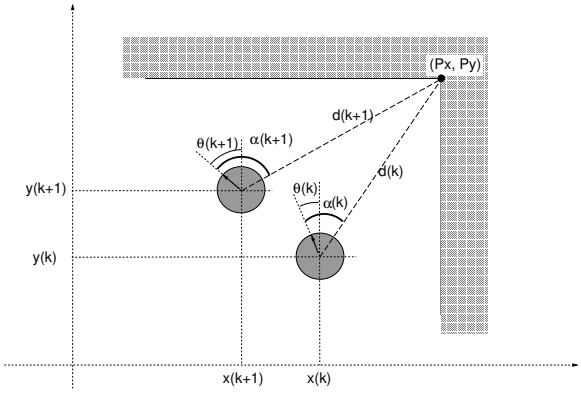


Figure 6. Mobile robot's perception of a corner in the environment at different state configuration

Identifying the corners using the range-finder yields a set of observation results:

$$z(k) = \{z_j(k) \mid 1 <= j <= n_0\} \quad (7)$$

where n_0, j and i denote the number of observation, the actual, and the predicted observation respectively. The observation model describes how the measurement $z_j(k)$ are related to the mobile robot's position and has the form:

$$\begin{array}{rcl} z_j(k) & = & h(x(k), p_t) + w_j(k), \\ w_j(k) & \approx & N(0, R_j(k)) \end{array} \quad (8)$$

In a two-dimensional case, p_t represent the *target* state vector and has the form (p_x, p_y), i.e. the *x, y* coordinates of a corner (Figure 6). The measurement function $h(x(k), p_t)$ expresses an observation $z(k)$ from the sensor to the target as a function of the robot's location $x(k)$ and the target coordinates p_t; taking into consideration the mounting position of the sensor on the robot as well. Figure 7 shows the position of the laser range-finder mounted on the robot. The range-finder covers a 180 degree fan-shaped scan of its front.

Thus, the measurement function is ex-

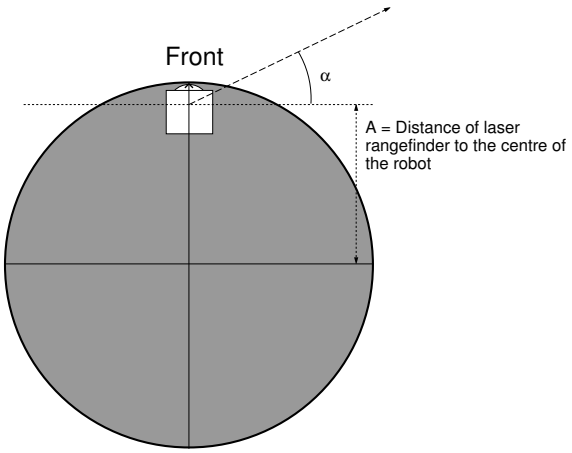


Figure 7. Mounting position of the laser range-finder on the robot

pressed as:

$$h(x(k), p_t) = \begin{pmatrix} d \\ \alpha \end{pmatrix} = \begin{bmatrix} sqrt((p_x - x(k) + A\sin\theta(k))^2 + \\ (p_y - y(k) + A\cos\theta(k))^2) \\ \\ \arctan(\frac{p_y - y(k) - Acos\theta(k)}{p_x - x(k) + Asin\theta(k)}) - \theta(k)) \end{bmatrix} \quad (9)$$

Each measurement is assumed corrupted by a zero-mean, Gaussian noise disturbance $w_j(k)$ which has the covariance $R_j(k)$, associated with the j th detected feature.

Measurement Jacobian

As mentioned previously, the mobile robot used is *holonomic*, therefore the state function is hence linear. The measurement function however is non-linear, as shown in Equation (9). Therefore, $h(x(k), p_t)$ is to be linearised about the state vector $x(k) = (x, y, \theta)$. $h(x(k))$ can be replaced by the Jacobian:

$$H = \nabla h = \begin{bmatrix} \nabla h_{11} & \nabla h_{12} & \nabla h_{13} \\ \nabla h_{21} & \nabla h_{22} & \nabla h_{23} \end{bmatrix} = \begin{bmatrix} \frac{\partial d}{\partial x} & \frac{\partial d}{\partial y} & \frac{\partial d}{\partial \theta} \\ \frac{\partial \alpha}{\partial x} & \frac{\partial \alpha}{\partial y} & \frac{\partial \alpha}{\partial \theta} \end{bmatrix} \quad (10)$$

The partial derivative of each element in the measurement Jacobian are:

$$\nabla h_{11} = \frac{\partial d}{\partial x} = \frac{-p_x + x - A\sin\theta}{\sqrt{(p_x - x + A\sin\theta)^2 + (p_y - y - A\cos\theta)^2}} \quad (11)$$

$$\nabla h_{12} = \frac{\partial d}{\partial y} = \frac{-p_y + y + A\cos\theta}{\sqrt{(p_x - x + A\sin\theta)^2 + (p_y - y - A\cos\theta)^2}} \quad (12)$$

$$\nabla h_{13} = \frac{\partial d}{\partial \theta} = A\frac{(p_x - x)\cos\theta + (p_y - y)\sin\theta}{\sqrt{(p_x - x + A\sin\theta)^2 + (p_y - y - A\cos\theta)^2}} \quad (13)$$

$$\nabla h_{21} = \frac{\partial \alpha}{\partial x} = \frac{p_y - y - A\cos\theta}{(p_x - x + A\sin\theta)^2 + (p_y - y + A\cos\theta)^2} \quad (14)$$

$$\nabla h_{22} = \frac{\partial \alpha}{\partial y} = \frac{-p_x + x - A\cos\theta}{(p_x - x + A\sin\theta)^2 + (p_y - y + A\cos\theta)^2} \quad (15)$$

$$\nabla h_{23} = \frac{\partial \alpha}{\partial \theta} = (2 - \frac{A^2 - (p_x - x)^2 - (p_y - y)^2}{A(A - (p_y - y)\cos\theta + (p_x - x)\sin\theta)})^{-1} - 1 \quad (16)$$

Hence, the linearised observation model is $\delta z(k+1) = H\delta x(k)$.

Uncertainty model

Often there is no knowledge of correlation of error sources, and both the process noise covariance matrix Q and the measurement error covariance matrix R are assumed to be diagonal. Q and R are expressed as follow.

$$Q = diag[\sigma_x^2 \ \sigma_y^2 \ \sigma_\theta^2] \quad (17)$$

$$R = diag[\sigma_d^2 \ \sigma_\alpha^2] \quad (18)$$

The state or predicted error covariance matrix is in general, given by:

$$P(k+1|k) = P(k|k) + Q(k) \quad (19)$$

hence P will evolve into the diagonal terms as the localisation cycle runs. The state uncertainty model represents the disturbances which excite the linear sys-

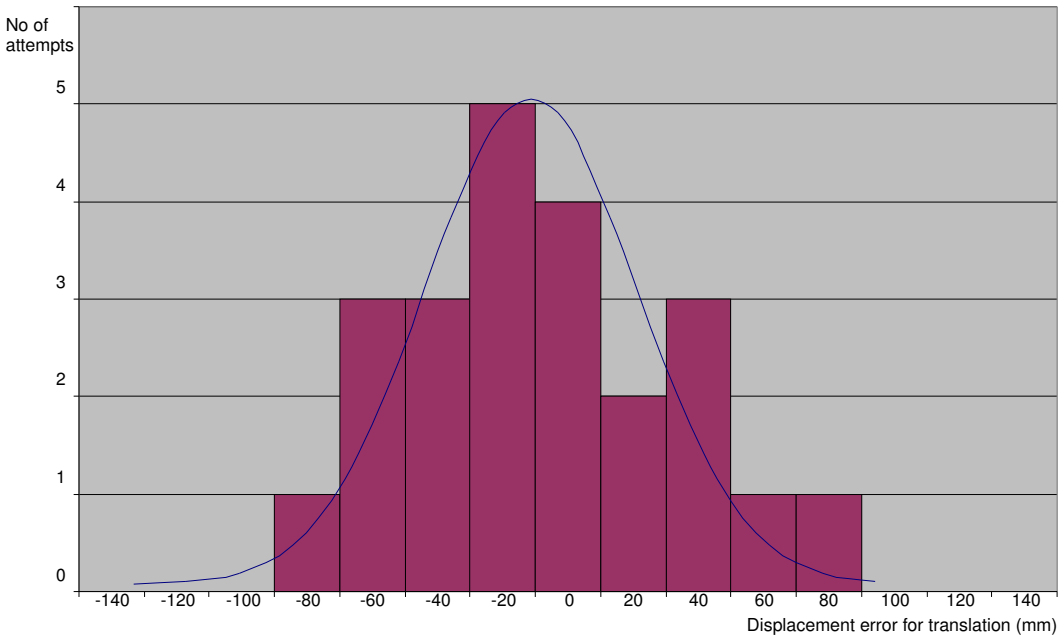


Figure 8. Displacement error for desired distance of 1000mm, with all wheels in line with the heading direction

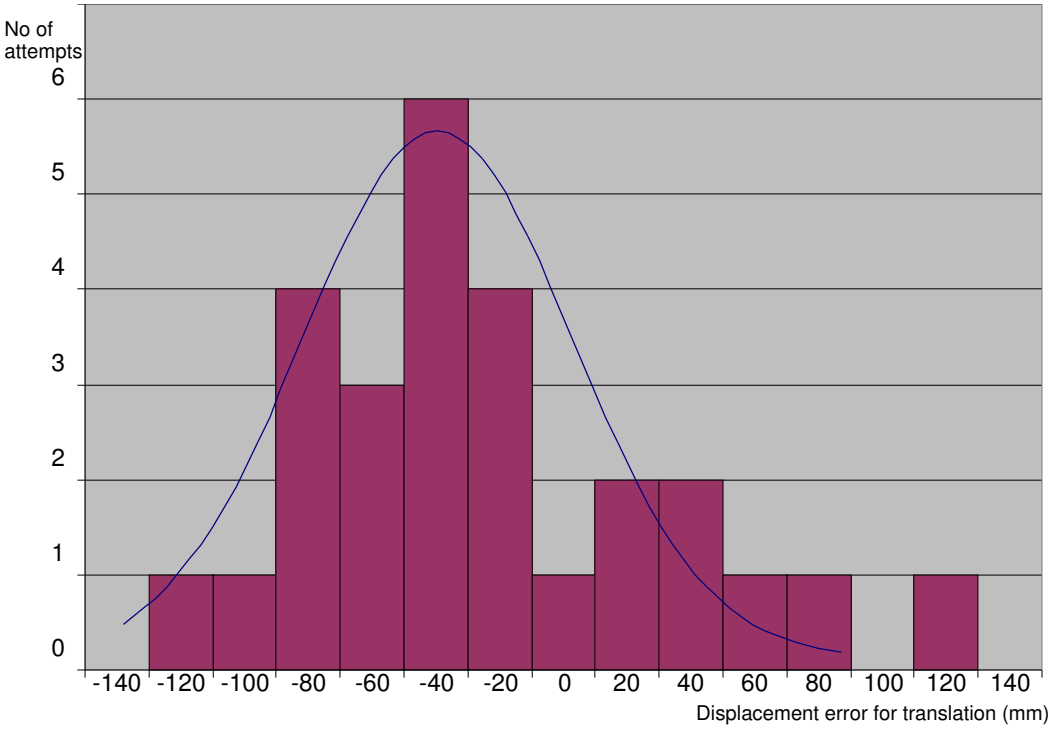


Figure 9. Displacement error for desired distance of 1000mm, with all wheels direction at 180 degree relative to the heading direction

tem. It estimate how bad things can get when the system is run open loop (i.e. with no sensors) for a given period of time. To find the variance σ^2 of each axis, the robot is made to move a certain distance in a particular axis, and the actual distance travelled is measured. This experiment is repeated many times (about 30 times), and when the number of tests is plotted against the displacement error should give a *Guassian* curve or *Normal distribution*. The variance of this curve corresponds to the error covariance σ^2 term in the error covariance matrix Q. This is repeated for the rest of the axes. Figure 8 shows the *Normal distribution* of displacement error against the numbers of attempts. These tests are carried out with the wheels of the robot align with the heading direction of the position command before moving off. A similar curve is shown in Figure 9, but with the wheel direction at 180 degree relative to the heading direction before moving off. The effect of initial wheel orientation will be discussed below.

Wheel slippage in a holonomic *wheel-driven* robot is extremely random. Besides wheel slippage occurring in a holonomic mobile robot, drift will also occur especially when the wheels change direction. The following wheel configuration belongs to the Nomad XR4000.

The steering of the wheel is coupled to the driving, but not vice-versa [14]. This is to facilitate easier steering of the wheel as it overcomes the frictional resistance during the changing of direction, especially when the robot is static. Figure 11 shows the actual motion of the wheel when the robot is changing its direction. The illustration in Figure 11 has been exaggerated. The radial path of the wheel (in Figure 11b) is attributed to the simultaneous driving and steering. The steering angular velocity is much higher than the driving velocity, hence reducing the radius to an insignificant value. The error arises due to a single direction change, is therefore negligible. Moreover, the errors

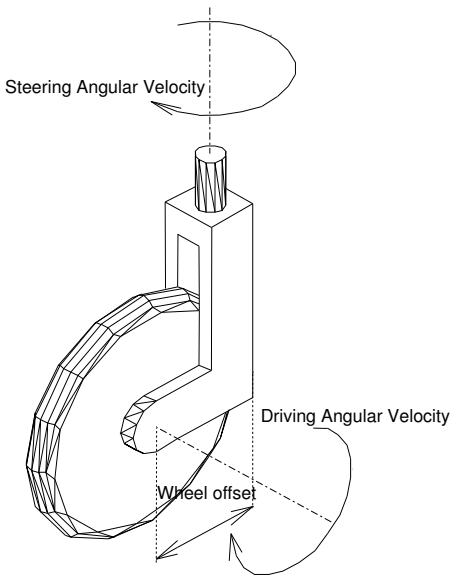


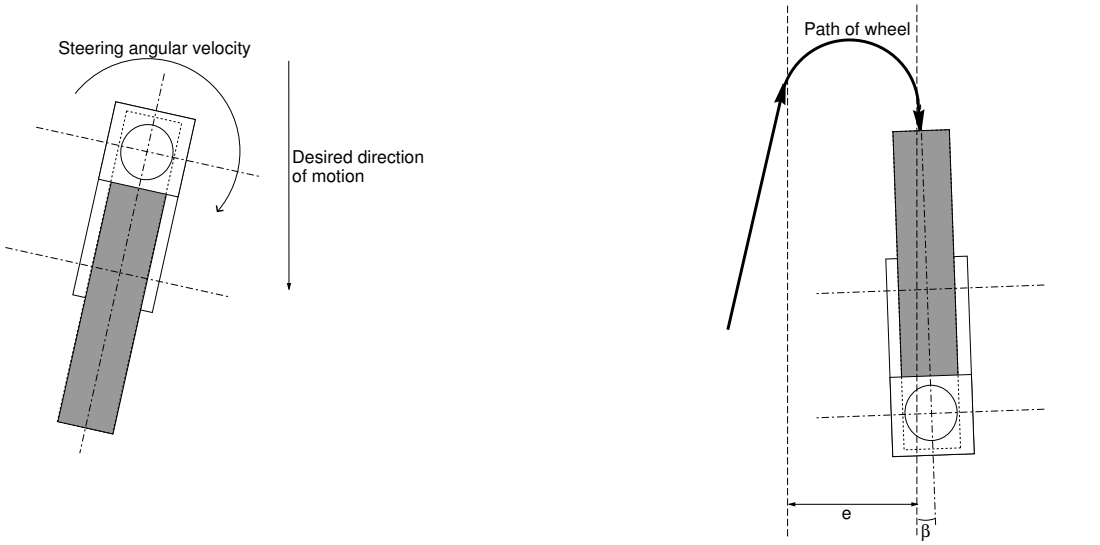
Figure 10. Wheel configuration Nomad XR4000 mobile robot

(e and β in Figure 11b) are incremental and will become relatively significant over time. This is especially when the robot is required to move in a manner which requires frequent change in direction and sudden movement for a substantiate period of time or distance.

The positioning error is, thus caused by the combination of this behaviour in the four wheels of the robot. Although the error is caused by a known behaviour of the wheels, it is still tedious to model such uncertainty. At this stage, our work emphasis is not to investigate such complex model, rather it is on monitoring the change in heading direction, $\triangle\phi$ of the robot. Note that the heading direction is not necessary the same as the orientation of the mobile robot. The error is also proportional to the distance travelled, $\triangle x$, and the standard deviation of the positioning error for x axis can be computed dynamically at every localisation cycle using an appropriate proportional factor E,

$$\sigma_x(k+1) = E\ \triangle x(k)\ \triangle\phi(k) \qquad (20)$$

subsequently performing the same computation for the other axes y, θ will yield the process/positioning error covariance matrix Q (17).



a) Initial orientation of wheel

b) Orientation and path of actual wheel movement resulting in a linear and angular error

Figure 11. Wheel motion during changing of robot's heading direction

The error standard deviation for the laser range-finder is given in its manual. A similar test as the position error is also carried out to verify the value in the manual. σ_α is not stated in the manual and it was found to be negligible, i.e. $\sigma_\alpha \approx 0$.

3.2.2 Localisation Cycle using Kalman Filter

With knowledge of the models required for the computation of the Kalman gain, explanation of the localisation procedure becomes easy. The overview of the cyclic localisation procedure is summarised in the flow chart in Figure 12. The algorithm consists of the following steps: position prediction, observation, measurement prediction, matching, and finally estimation. We now proceed to discuss each of this steps in detail.

Robot Position Prediction

Obtaining the position prediction is the very first step of the localisation cycle. Since we neither have the knowledge of Nomad XR4000 robot's wheel encoder readings, nor the computation of the integrated positioning of the robot; we shall treat the position feedback from the robot as position prediction $\hat{x}(k+1|k)$. The associated covariance $P(k+1|k)$ is also needed, and is obtained through Equation (19). During the initial stage, the variances $P(0|0)$ are given estimated values of a few centimetres/degrees. In practice, the accuracy of the estimated variances is not important, as P should converge to a more correct value as the localisation cycle runs.

Actual Observation

Next, the observations of the corners in the environment is obtained using the *Sick* laser range-finder. The observation set $z_j(k+1)$ is comprised of n_0 observed corners. The subscript 0 in n_0 represents observation.

Observation/Measurement Prediction

Using the predicted robot position $\hat{x}(k+1|k)$ and the current *map* with known corners position, we can generate the predicted observations of each corner/target p_t. This is done using the *measurement function*, which yield the predicted observations,

$$\hat{z}_i(k+1) = h(\hat{x}(k+1|k), p_t), i = 1, 2, ..., n_p \quad (21)$$

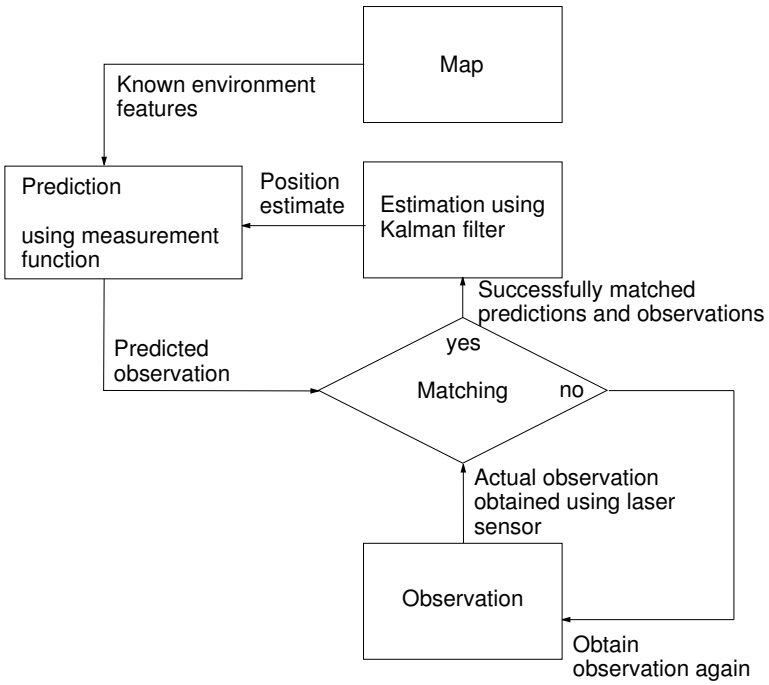


Figure 12. Localisation algorithm

note that here, the subscript p in n_p represents prediction. The *measurement Jacobian* H_i (in equation 10) is then computed using the predicted state $\hat{x}(k+1|k)$ and the corners coordinates p_x, p_y.

Matching

Matching of predicted and actual observation is the next step. The *matching method* is discussed in detail in Section 3.1.3. For each successfully matched predicted and actual observation, *innovation* $v_{ij}(k+1)$ is computed.

$$v_{ij}(k+1) = z_j(k+1) + \hat{z}_i(k+1) \quad (22)$$

Since both $z_j(k+1)$ and $\hat{z}_i(k+1)$ observations are associated with covariances, it is therefore appropriate to compute the *innovation covariance* as well. This covariance $S_{ij}(k+1)$ is expressed as,

$$S_{ij}(k+1) = H_i\ P(k+1)\ H_i^T + Ri(k+1) \quad (23)$$

The numbers of matched observations, n, at time $k+1$ can all be combined in a stacked manner. The innovation can be stacked into a single vector,

$$v(k+1) = \begin{bmatrix} v_{11}(k+1) \\ v_{22}(k+1) \\ . \\ . \\ v_{nn}(k+1) \end{bmatrix} = \begin{bmatrix} z_1(k+1) + \hat{z}_1(k+1)) \\ z_2(k+1) + \hat{z}_2(k+1) \\ . \\ . \\ z_n(k+1) + \hat{z}_n(k+1) \end{bmatrix} \quad (24)$$

The innovation covariance $S_{ij}(k+1)$ can also be stacked to form a *composite* innovation covariance $S(k+1)$ using stacked noise vectors $P(k+1)$, $R(k+1)$, and stacked measurement Jacobian matrix H.

Estimation

Finally, the estimated state $\hat{x}(k+1|k+1)$ result is computed, using the *Kalman gain*,

$$W(k+1) = P(k+1)\ H^T\ S^{-1}(k+1) \quad (25)$$

The updated robot position estimate is

thus,

$$\hat{x}(k+1 \quad |k+1) = \hat{x}(k+1|k) + W(k+1)\, v(k+1) \tag{26}$$

It is also important to compute the updated state covariance, which would be used in the next cycle.

$$P(k+1 \quad |k+1) = P(k+1|k) + W(k+1)\, S(k+1)\, W^T(k+1) \tag{27}$$

The state estimate $\hat{x}$ and its covariance P should converge to a better result as the localisation cycle runs.

4 RESULTS

To assess the localisation capability of the mobile robot, a simple test is carried out. With its initial position marked out on the floor, the Nomad XR4000 is made to move in a two by two metres square path ending at the same position it started off. The square path movement repeats for about 5 cycles. In the case where no localisation is implemented, the end position of the robot will be offset from the initial starting position by a significant value (usually about $\pm$200mm in both x and y axes). In order to be able to localise or verify the reported robot position, a prior knowledge of the environment is needed. We assumed that the first reported position is one that is perfect. A scan of the surrounding will then determined the global positions of some of the corners around the mobile robot. This initial *corners information* is therefore treated as one that is based on accurate positioning. Figure 13 shows the corners identified by the robot before executing the *square path movement*. *Kalman Filter localisation* using the preceding *environmental information* is performed at every corners of the square path. Every localisation cycle will iterate until the correction is less than $\pm$30mm in x and y axes, and less than $\pm$35mRad in orientation of the robot. At the end of 5 cycle of the square path, no localisation is performed

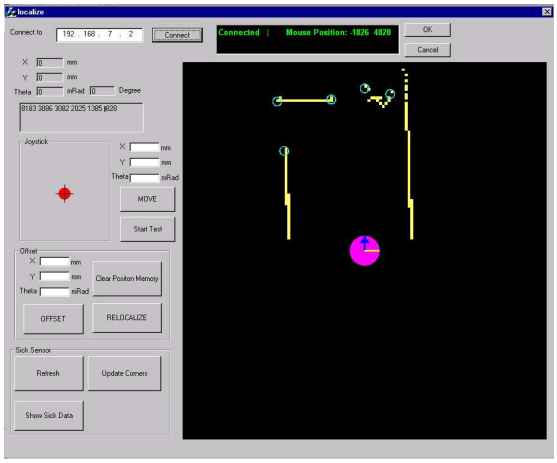


Figure 13. Corners detection in *Square Path Test*

at the ending position. The error of the ending and initial position is found to reduce to about $\pm$50mm in x and y axes, and about $\pm$85mRad in orientation.

Wheel slippage causes a drift in the mobile robot can be quite disastrous in mobile robot navigation. Figure 14 shows the implementation of Nomad XR4000 navigating from *Room B* to *Location A*. Position corrections are performed at predesignated locations in the environment, resulting in a relatively accurate path corresponding to the given environmental layout. On the other hand, another path recorded which is supposed to move from *Location A* back to *Room B*, is controlled manually by a joystick. With pure odometry and no localization, the robot seems to cross over the walls which is obviously not possible. This shows the severity of the error caused by the robot drifts.

The two tests demonstrated the effectiveness of the Kalman Filter localisation. It also shows that localisation is an essential issue to be addressed in autonomous mobile robotics.

5 CONCLUSION

The localisation problem is addressed by

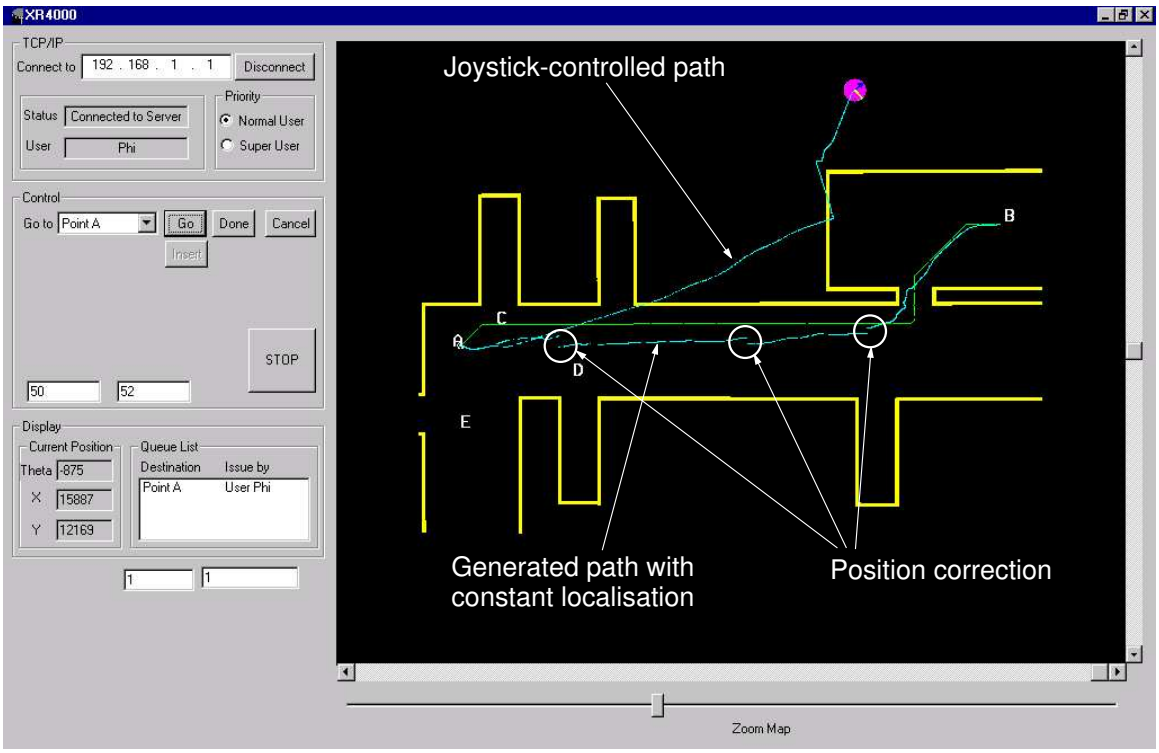


Figure 14. Implementation of Localisation in Office environment. Joystick controlled path without localization give a grossly wrong information

verifying the robot's position with known features in the environment. Kalman Filter localisation is investigated and implemented. Corners in the environment are the features used to verify the robot's position. Corner detection method which extracts the global positions of corners in the environment using the laser rangefinder is implemented. Kalman filter is used to estimate the optimal robot's position based on noisy sensors data (corners detected) and inaccurate positioning from the robot. The localisation process proved to be able to enhance the robot positioning capabilities, consequently improving the mapbuilding and navigating functions as well.

6 INDUSTRIAL SIGNIFICANCE

The localisation issue is part of the development of a mobile base robot. This mobile base platform can be deployed in manufacturing shop floor as an automated transport system. Compared to conventional Automated Guided Vehicle (AGV), this mobile platform needs no external guiding installation (such as magnetic tape, reflector or beacons). The mobile robot being a free-ranging vehicle will possess ability to path planning as well as real-time obstacle avoidance enhanced by the localisation capabilities reported in this article. A manipulator can be integrated onto the mobile platform, thus expanding its applications beyond transportation of materials and parts.

REFERENCES

[1] J. Borenstein and L. Feng, "Measurement and correction of systematic odometry errors in mobile robots," *IEEE Tansactions on Robotics and Automation*, vol. 12, pp. 869–880, December 1996.

[2] J. Borenstein and L. Feng, "Umbmark: A benchmark test for measuring odometry errors in mobile robots," *Proceedings of the 1995 SPIE Conference on Mobile Robots*, pp. 569–574, October 1995.

[3] B. Barshan and H. F. Durrant-Whyte, "An inertial navigation system for mobile robot," *IEEE Tansactions on Robotics and Automation*, vol. 11, pp. 328–342, June 1995.

[4] J. Borenstein and Y. Koren, “Hierarchial computer system for autonomous vehicle,” *Proceedings of the 8th Israelic Convention on CAD/CAM and Robotics, Tel-Aviv, Israel*, December 1986.

[5] C. Cohen and F. Koss, “A comprehensive study of three object triangulation,” *Proceedings of the 1993 SPIE Conference on Mobile Robots, Bosten, MA*, pp. 95–106, November 18-20 1992.

[6] M. Betke and L. Gurvits, “Mobile robot localization using landmarks,” *1994 International Conference on Intelligent Robots and Systems, Munich, Germany*, pp. 135–142, September 12-16 1994.

[7] I. Cox, “Blanche - an experiment in guidance and navigation of an autonomous mobile robot,” *IEEE Tansactions on Robotics and Automation*, vol. 7, no. 3, pp. 193–204, 1991.

[8] J. L. Crowley, “World modelling and position estimation for a mobile robot using ultrasonic ranging,” *Procceedings of the IEEE International Conference on Robotics and Automation, Scottsdale, AZ*, pp. 674–680, May 14-19 1989.

[9] M. D. Adams and A. Kersteins, “Tracking naturally occuring indoor features in 2d and 3d with lidar range/ amplitude data,” *International Journal on Robotics Research*, vol. 17, pp. 907–923, September 1998.

[10] G. Welch and G. Bishop, “An introduction to the kalman filter,” paper TR 95-041, Department of Computer Science, University of North Carolina at Chapel Hill, Chapel Hill, NC 27599-3175, February 1995.

[11] A. Kelly, “A 3d state space formulation of a navigation kalman filter for autonomous vehicles,” tech. rep., The Robotics Institute, Carnegie Mellon University, 5000 Forbes Avenue, Pittsburgh, PA 15213, May 1994.

[12] J. J. Leonard and H. F. Durrant-Whyte, *Directed Sonar Sensing for Mobile Robot Navigation*. Kluwer Academic Publishers, 1992.

[13] M. D. Adams, *Sensor Modelling, Design and Data Processing for Autonomous Navigation*, vol. 13 of *Robotics and Intelligent Systems*, ch. 7-8. World Scientific, 1999.

[14] R. Holmberg and O. Khatib, “Development of a holonomic mobile robot for mobile manipulation tasks,” *Procceedings of International Conference on Field and Service Robotics, FSR’99, Pittsburgh*, pp. 268–273, August 1999.