Mobile Robot Map Building from an Advanced Sonar Array and Accurate Odometry

Kok Seng CHONG
kok.seng.chong@eng.monash.edu.au

Lindsay KLEEMAN
lindsay.kleeman@eng.monash.edu.au

Intelligent Robotics Research Center (IRRC)
Department of Electrical and Computer System Engineering
Monash University, Victoria 3168 Australia
http://calvin.eng.monash.edu.au/IRRC/IRRCHomePage.html

Abstract

This paper describes a mobile robot equipped with a sonar sensor array in a guided feature based map building task in an indoor environment. The landmarks common to indoor environments are planes, corners and edges, and these are located and classified with the sonar sensor array. The map building process makes use of accurate odometry information that is derived from a pair of knife edged unloaded encoder wheels. Discrete sonar observations are incrementally merged into partial planes to produce a realistic representation of environment that is amenable to sonar localisation. Collinearity constraints among features are exploited to enhance both the map feature estimation and robot localisation. The map update employs an Iterated Extended Kalman Filter (IEKF) in the first implementation and subsequently a comparison is made with the Julier-Uhlmann Kalman Filter (JUKF) which improves the accuracy of covariance propagation when non-linear equations are involved. The map accounts for correlation among features and robot positions. Partial planes are also used to eliminate phantom targets caused by specular reflection of the sonar. Unclassifiable sonar targets are integrated into the map for the purpose of obstacle avoidance. The paper presents simulated and experimental data.

1 Introduction

The objective of this work is to implement an autonomous mobile robot capable of navigating in an *a priori* unknown indoor environment using a sonar sensor. To this end, the robot requires the capability to build a map of the environment, which is a cyclic process of moving to a new position, sensing the environment, updating the map and planning subsequent motion. Map building and navigation is a complex problem because map integrity cannot be sustained by odometry alone due to errors introduced by wheel slippage and distortion. Exteroceptive sensing, such as sonar sensing as employed in this paper, is necessary, but any sensing is also subject to random errors. Hence, neither odometry nor matching sensory data to the map gives flawless estimation of the robot’s position, yet this position estimate becomes a reference for the integration of new features in the map. Consequently, with time errors in robot position influence errors in the map and map errors influence the position estimation.

This paper employs sonar sensing in the map building process for many reasons. Sonar has the property that the data is sparse and naturally selects useful landmarks, such as walls, wall moldings and corners. This alleviates the data processing compared to dense ranging devices such as laser range finders and stereo vision systems. Sonar also offers a high degree of ranging and bearing accuracy in an array configuration as deployed in this paper. Since most robots today employ some form of sonar due to the cost and power consumption advantages, there is considerable interest in its application.

Sonar sensing has some important properties that need to be carefully understood in order to properly exploit the sensing data. Firstly, sonar transducers have a significant angular spread of energy known as the beamwidth. In many systems, the beamwidth gives rise to large angular uncertainty in measurement. Some researchers have attempted to deal with this uncertainty by employing grid based maps and repetitive measurements, as in the work by [16]. Grid map update schemes range from Bayesian [13], evidential [20] to fuzzy [19] and rely on viewing targets from many locations. Localisation with a grid map can be complex. A grid map based localisation scheme has been developed in [6], but it is suitable for laser rangefinder systems only. Other researchers do not consider localisation necessary in their applications [4, 18]. Features based mapping schemes have become more commonplace [5, 21] after Kuc and Siegel [11] presented a method for discriminating planes, corners and edges using sonar data gathered at two positions. Later Kleeman and Kuc [10] developed a sonar sensor which allows target discrimination at one position and target localisation with high precision.. Hong and Kleeman [7] have successfully demonstrated the localisation capability of a mobile robot with a sonar array in a known environment using 3D features. Data fusion methods associated with feature based mapping include the Kalman Filter [1, 15, 22], maximum likelihood estimation [14] and heuristic rules [4].

The second important property of sonar systems is the appearance of phantom targets that are due to multiple specular reflections. For example, a sonar sensor will see a virtual image of a corner due to the reflection from the wall in the outwards and return paths in Figure 1. A credibility count [5] has been used to identify these phantom targets, however this approach fails when the phantom target appears consistently from different positions as is the case in the example of Figure 1. A physically based solution is presented in this paper. The third important property of sonar systems is that, when sensing a planar wall, the sensor can only see the part of the wall which is orthogonal to the line of sight - like phantom targets, this property results from specular reflection. Therefore, if the robot navigates along a wall, the robot sees the wall not as an entity but as a set of discrete, approximately collinear planar elements. Postulates need to be made about the relationships between various sonar features during map matching. Furthermore, to reduce the risk of wrongly associating two features, the robot has to be refrained from moving a long distance between successive scanning points during map building.

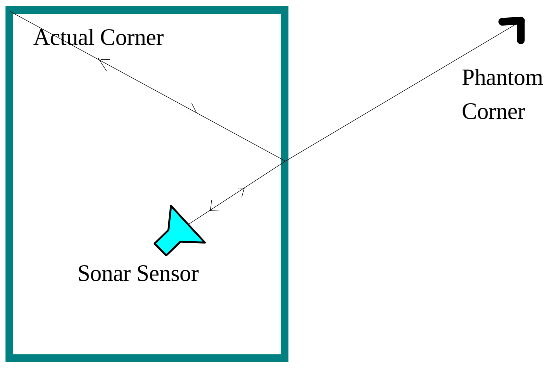


Figure 1 : Phantom Target Example

In the authors' opinion, prolonged navigation can best be achieved when map feature errors are systematically generated from the sensor and odometry errors. It is convenient, and usually justifiable, to assume that errors are approximately Gaussian in their distribution, and to represent the errors with covariance matrices, since robust noise filtering tools use this form of representation. The authors favour the Kalman Filter because the basis of the Kalman Filter is the Bayesian formula and the principle of minimum mean square error [2] that are well understood and physically acceptable. The Kalman Filter reduces the uncertainties of the parameters of interest by weighting the initial estimation of errors with the errors associated with the new information (known as *observations*) about the parameters. In the context of sonar map building, the parameters to be estimated are the robot's position and the features in the map. The observations are the postulates about the relational constraints among the new features and the existing features. The Kalman Filter makes available knowledge about the uncertainties of map features that is important for path planning that avoids obstacles and localisation within the map.

The Kalman Filter is based on a linear system model. To overcome this limitation, the Extended Kalman Filter (EKF) can be employed and is founded on the assumption that for small noise, first order linearisation of the system model is sufficient for propagating the noise covariance. The problem with discarding higher order terms is that bias can accumulate after repeated estimation. Two approaches have been proposed to deal with this bias: The first approach, called the Iterated Extended Kalman Filter (IEKF), iteratively estimates the parameters of interest by repetitively linearising the system equations about the new estimates under little change results. The second approach, which will be referred to as the Julier-Uhlmann method (JUKF) [9], is based on generating a set of data points using the error covariance of the input parameters, propagating the data points and computing the resulting error covariance, thus obviating the need to manually evaluate various Jacobian matrices. This paper compares the accuracy of the two methods.

The mapping strategy presented here is feature based and has the following attributes:

1. All three types of primitive features recognisable by our advanced sonar sensor are processed to become part of a map: Discrete planar and corner elements gathered by the sonar sensor at various stages are merged incrementally to form *partial planes*. Planar elements are only merged to the adjacent partial planes to avoid falsely closing a gap, such as a doorway. Discrete edge elements do not partake in the process of forming partial planes, but they are still used to enhance localisation accuracy and map integrity.
2. Not only does 'plane to plane', 'corner to corner' and 'edge to edge' matching occur as in other approaches [5,11], but the relational constraint between a corner and two intersecting planes is exploited to further improve the fidelity of map. Relational constraints are described in [1] and are used by [17] for a known environment.
3. The partial planes are used to distinguish and subsequently eliminate phantom corner targets and edge targets caused by specular reflection.

4. Two implementations based on the two filters, JUKF and IEKF, are used to evaluate state transition equations, generate state-measurement cross covariance and propagate error covariance matrices. The two approaches are compared.

This paper is organised as follows. The robot processing, locomotion, odometry and sonar sensor are described in Section 2. Section 3 presents a summary of the IEKF and JUKF filters. In section 4 the map environmental model is presented and formulated as a statistical optimisation problem that is solvable with a Kalman Filter. Two sets of equations, one for the IEKF and one for the JUKF, are derived. These equations are evaluated in Section 5 for different map growth scenarios. Simulation results for the IEKF and JUKF methods are presented in Section 6, while the results of four experiments are shown in Section 7. Finally the conclusion summarises the mapping technique and the findings of the comparison between the IEKF and JUKF filters and also presents future directions for the research.

2. Robot Architecture

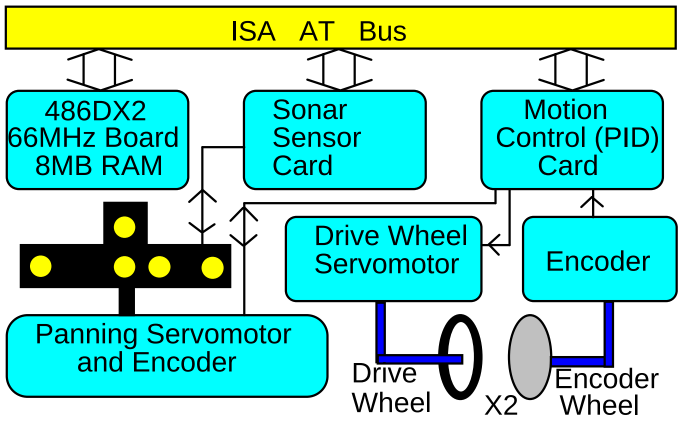


Figure 2 : The robot system architecture

As shown in Figure 2, the communication backbone of the robot is an ISA AT Bus with a 486DX2-66MHz processor board controlling a custom sonar sensor card and a custom servo motion control card. The sensor control card sends transmit pulses and captures entire echoes from three receiving transducers. The transmit pulse is generated from a 10 µs 300 V - 0 V - 300 V voltage pulse and the echo waveform is sampled with a 12 bit ADC at 1 Mhz. The motion control card contains a MC1401 chip which provides PID control to the four DC motors, two in the pan tilt mechanism and two for the drive wheels. For every motor, an encoder is mounted on the actuation shaft (*ie* after a gear box) to generate feedback information that is not corrupted by backlash in the gearbox.

2.1 The Sonar Array

The sonar array illustrated in Figure 3 has a multiple transducer configuration which makes it possible to classify common indoor features into planes, 90° concave corners and edges. The sonar array accurately estimates specular target ranges to within 0.2 mm and elevation and azimuth angles to within 0.02° for ranges to 5 m within the sensor beamwidth [10]. At every scanning point, the sensor first simultaneously fires TR1 while scouting anticlockwise at 90°/sec to locate the directions of potential targets from the echoes on the three receivers. Then, it pans clockwise at the same speed, only slowing down at the directions of the potential targets found earlier and fires T0 followed by T2. If classification is unsuccessful, the target is tagged as unknown but range and bearing are still recorded to unknown objects.

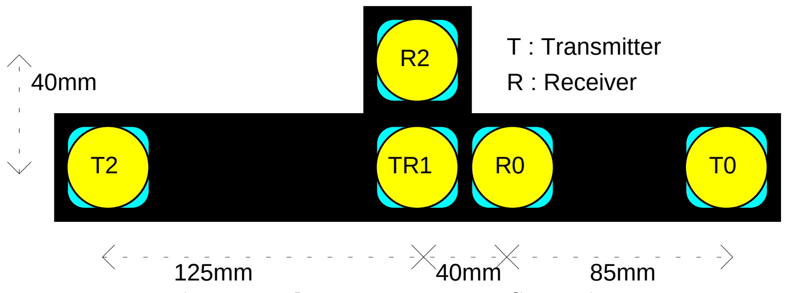


Figure 3: The sonar array configuration

2.2 The Locomotion and Odometry System

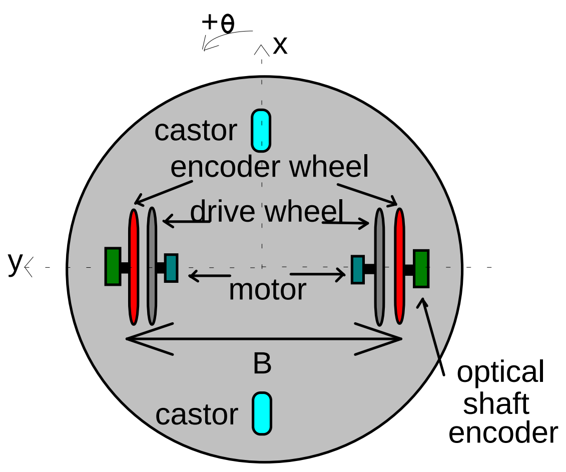


Figure 4 : The odometry system

The locomation and odometry system shown in Figure 4 consists of drive wheels and separate encoder wheels that generate odometry measurements from optical shaft encoders. The encoder wheels are made with O-rings contacting the floor so as to be as sharp-edged as practically possible to reduce wheelbase uncertainty, and are independently mounted on linear bearings to allow vertical motion, and hence minimise problems of wheel distortion and slippage. This design greatly improves the reliability of odometry measurements. The odometry error model used to propagate error covariance and odometry benchmarking can be found in [3].

3 Summary of the Iterated Extended Kalman Filter (IEKF) and the Julier-Uhlmann Kalman Filter (JUKF)

The section begins by introducing Kalman Filter in a general context. Before proceeding, the notation used will be explained. A circumflex above a random variable, $\hat{\mathbf{S}}(k+1)$, is used to indicate the *estimator* of the random variable, whereas a bar over a random variable, $\overline{\mathbf{S}(k+1)}$, is used to indicate the *mean* of the random variable. The partial derivative operator is denoted by ∇ and is defined by (1).

$$\nabla_{\mathrm{x}} = \frac{\partial}{\partial \mathbf{x}} = \begin{bmatrix} \frac{\partial}{\partial x_1} & \frac{\partial}{\partial x_2} & \cdots & \frac{\partial}{\partial x_n} \end{bmatrix} \quad \text{where} \quad \mathbf{x} = \begin{bmatrix} x_1 & x_2 & \cdots & x_n \end{bmatrix} \tag{1}$$

Suppose the state vector **S**(*k*) contains all the randomly distributed parameters of interest which evolve with discrete time according to the state transition equation

$$\mathbf{S}(k+1) = \mathbf{F}(\mathbf{S}(k), \mathbf{U}(k+1)) \tag{2}$$

where **U**(*k*) is the input vector. At stage *k+1*, these random parameters can be *observed* with a set of measurements **M**(*k*) via the observation model

$$\mathbf{G}(\mathbf{S}(k+1), \mathbf{M}(k+1)) = \mathbf{0} \tag{3}$$

The estimation of **S**(*k+1*) with equation (2) and (3) is inherently imperfect because of the noise in **S**(*k*), **U**(*k+1*) and **M**(*k+1*). The goal of optimisation is to generate a new state estimate $\hat{\mathbf{S}}(k+1)$ that minimises the *mean square error* of the parameters **S**(*k+1*) conditioned on all the past observations which is equal to the mean of the parameters conditioned on all the past observations [2]. Let $\mathbf{Z}^j$ be all the observations gathered up to stage *j*, and $\hat{\mathbf{S}}(i|j)$ be the minimum mean square error estimate of **S**(*i*) conditioned on $\mathbf{Z}^j$, then

$$\hat{\mathbf{S}}(i|j) = \arg \underset{\hat{\mathbf{S}}}{\min} E\left\{\left(\mathbf{S}(i) - \hat{\mathbf{S}}\right)\left(\mathbf{S}(i) - \hat{\mathbf{S}}\right)^T | \mathbf{Z}^j\right\} = E\left\{\mathbf{S}(i) | \mathbf{Z}^j\right\} \tag{4}$$

where *E{.}* is the expectation of a random variable. Associated with this estimator is the error covariance matrix

$$\mathbf{P}_{ss}(i|j) = E\left\{\left(\mathbf{S}(i) - \hat{\mathbf{S}}(i|j)\right)\left(\mathbf{S}(i) - \hat{\mathbf{S}}(i|j)\right)^T | \mathbf{Z}^j\right\} \tag{5}$$

Suppose $\hat{\mathbf{S}}(k|k)$ exists at stage *k*. Upon transition to stage *k+1* and prior to making an observation, the parameters at stage *k+1* conditioned on the observations up to *k* only, can be predicted via

$$\hat{\mathbf{S}}(k+1|k) = E\left\{\mathbf{F}(\mathbf{S}(k), \mathbf{U}(k+1) | \mathbf{Z}^k\right\} \tag{6}$$

A set of measurements $\mathbf{M}(k+1)$ about **S**(*k+1*) can also be gathered at stage *k+1*. Due to the noise in both $\mathbf{S}(k+1)$ and $\mathbf{M}(k+1)$, equation (3) does not hold exactly. A *residual* vector can be defined as

$$\mathbf{z}(k+1) = -E\left\{\mathbf{G}(\mathbf{S}(k+1), \mathbf{M}(k+1)) | \mathbf{Z}^k\right\} \tag{7}$$

With the residual vector, the Kalman Filter can be invoked to generate a better estimate of **S**(*k+1*), namely $\hat{\mathbf{S}}(k+1|k+1)$ based on [2],

$$\hat{\mathbf{S}}(k+1|k+1) = \hat{\mathbf{S}}(k+1|k) + \mathbf{P}_{sz}(k+1|k)\mathbf{P}_{zz}^{-1}(k+1|k)\mathbf{z}(k+1) \tag{8}$$

and the error covariance is also reduced to

$$\mathbf{P}_{ss}(k+1|k+1) = \mathbf{P}_{ss}(k+1|k) - \mathbf{P}_{sz}(k+1|k)\mathbf{P}_{zz}^{-1}(k+1|k)\mathbf{P}_{xz}^{T}(k+1|k) \tag{9}$$

Where $\mathbf{P}_{sz}(k+1|k)$ is the cross-covariance between $\hat{\mathbf{S}}(k+1|k)$ and $\mathbf{z}(k+1)$, and $\mathbf{P}_{zz}(k+1|k)$ is the covariance of $\mathbf{z}(k+1)$, defined in a similar fashion.

In practice, the state transition equation and the observation equation are non-linear. Some methods are required to estimate the covariance and cross-covariance matrices required by Kalman Filter. The IEKF filter and the JUKF are introduced for this purpose.

3.1 The IEKF Method

The IEKF method is an extension of the Extended Kalman Filter (EKF) which is discussed first. With the EKF method,

$$\hat{\mathbf{S}}^{IEKF}(k+1|k) \approx \mathbf{F}(\hat{\mathbf{S}}(k|k), \hat{\mathbf{U}}(k+1)) \tag{10}$$

$$\begin{aligned} \mathbf{z}^{IEKF}(k+1) &\approx -\mathbf{G}\left(\hat{\mathbf{S}}(k+1|k), \hat{\mathbf{M}}(k+1)\right) \\ &\approx \nabla_{\mathbf{S}}\mathbf{G}\left\{\mathbf{S}(k+1|k+1) - \hat{\mathbf{S}}(k+1|k)\right\} + \nabla_{\mathbf{M}}\mathbf{G}\left\{\mathbf{M}(k+1) - \hat{\mathbf{M}}(k+1)\right\} \end{aligned} \tag{11}$$

The noise associated with all random vectors is assumed small, so that applying a first order Taylor's expansion about the estimator is reasonable for propagating the error covariance through non-linear equations. Suppose the error of $\hat{\mathbf{S}}(k+1|k)$ is not correlated with $\mathbf{U}(k)$, then

$$\mathbf{P}_{ss}^{IEKF}(k+1|k) \approx \nabla_{\mathbf{S}}\mathbf{F}\mathbf{P}_{ss}(k|k)\nabla_{\mathbf{S}}\mathbf{F}^{T} + \nabla_{\mathbf{U}}\mathbf{F}\mathbf{Cov}(\mathbf{U}(k+1))\nabla_{\mathbf{U}}\mathbf{F}^{T} \tag{12}$$

where $\nabla\mathbf{F}$ is the Jacobian matrix of **F**() evaluated around **S**(*k*|*k*) or **U**(*k+1*), as indicated by the subscript. $\nabla\mathbf{F}$ is also known as the *state transition matrix*. **Cov**(**U**(*k+1*)) is the error covariance of the input vector **U**(*k+1*). In a similar manner,

$$\mathbf{P}_{zz}^{IEKF}(k+1|k) \approx \nabla_{\mathbf{S}}\mathbf{G}\mathbf{P}_{ss}(k+1|k)\nabla_{\mathbf{S}}\mathbf{G}^{T} + \nabla_{\mathbf{M}}\mathbf{G}\mathbf{Cov}(\mathbf{M}(k+1))\nabla_{\mathbf{M}}\mathbf{G}^{T} \tag{13}$$

$$\mathbf{P}_{sz}^{IEKF}(k+1|k) \approx \mathbf{P}_{ss}^{IEKF}(k+1|k)\nabla_{\mathbf{S}}\mathbf{G}^{T} \qquad (14$$

The IEKF method improves the performance of the Extended Kalman Filter by linearising the measurement equation about the new estimate $\hat{\mathbf{S}}(k+1|k+1)$, and attempts to iteratively draw $\hat{\mathbf{S}}(k+1|k+1)$ closer to the true mean, in a way similar to solving a non-linear algebraic equation using Newton Raphson algorithm. Let $\eta_i = \hat{\mathbf{S}}^{IEKF,i}(k+1|k+1)$, with $\eta_0 = \hat{\mathbf{S}}^{IEKF}(k+1|k)$, the pseudo code is as follows,

```
set $i \leftarrow 0$
repeat {
        $\eta_{i+1} = \hat{\mathbf{S}}(k+1|k) + \mathbf{P}_{sz}(k+1|k+1;\eta_i)\mathbf{P}_{zz}^{-1}(k+1|k+1;\eta_i)$
            $\left[\mathbf{z}(k+1;\eta_i) - \nabla_{\mathbf{S}}\mathbf{G}(k+1|k+1;\eta_i)\left(\hat{\mathbf{S}}(k+1|k) - \eta_i\right)\right]$
        $i \leftarrow i+1$
        } while ($|\eta_i - \eta_{i-1}| > \varepsilon_\eta$)
$\mathbf{P}_{ss}^{IEKF}(k+1|k+1) = \mathbf{P}_{ss}(k+1|k) - \mathbf{P}_{sz}(k+1|k;\eta_i)\mathbf{P}_{zz}^{-1}(k+1|k;\eta_i)\mathbf{P}_{sz}^{T}(k+1|k;\eta_i)$
```

where the notation '$;\eta_i$' means 'evaluated at the new estimator η_i', and ε_η is a threshold vector. Further details on the IEKF can be found in [2, 8].

3.2 The JUKF Method

Julier and Uhlmann [9] have developed a method for accurately propagating a covariance matrix through non-linear equations while reducing the bias associated with the result. This section summarises and generalises the JU method in the context of this paper. Examples of how this method can be used with the Kalman Filter (hence the JUKF) are given at the end of this section.

Supposed that $\hat{\mathbf{S}}(k|k)$ of size $n_s \times 1$ is an estimate of a particular random vector $\mathbf{S}(k)$, and associated with the estimate is an error covariance matrix $\mathbf{P}_{ss}(k|k)$ of size $n_s \times n_s$, then a set of *sigma points* σ_j are generated from the $2n_s$ columns of

$$\begin{aligned}\pm\sqrt{n_s\mathbf{P}_{ss}(k|k)} &= \pm\left[\sigma_0 \quad \sigma_1 \quad \cdots \quad \sigma_{n_s-1}\right] \\ &= \pm\left[v_0 \quad v_1 \quad \cdots \quad v_{n_s-1}\right] diag(\sqrt{\lambda_j})\left[v_0 \quad v_1 \quad \cdots \quad v_{n_s-1}\right]^T\end{aligned} \tag{15}$$

where v_j's and λ_j's are all the normalised eigenvectors and eigenvalues of $n_s\mathbf{P}_{ss}(k|k)$, respectively, and *diag(.)* is the diagonal matrix formed from the arguments on the diagonal. The inner product between any two sigma points, $<v_m,v_n>$, is δ_{mn} the Kronecker delta function because any two eigenvectors of a symmetrical matrix, such as a covariance matrix, are orthogonal. A set of $2n_s$ data points can be formed,

$$\mathbf{S}_i(k|k) \in \bigcup_{i=0}^{n_s-1}\left\{\hat{\mathbf{S}}(k|k)+\sigma_i, \hat{\mathbf{S}}(k|k)-\sigma_i\right\} \tag{16}$$

Let $\mathbf{X}$ and $\mathbf{Y}$ be non-linear $n_x \times 1$ and $n_y \times 1$ functions of $\mathbf{S}$,

$$\mathbf{X}(k+1) = \mathbf{X}(\mathbf{S}(k)) \qquad \mathbf{Y}(k+1) = \mathbf{Y}(\mathbf{S}(k)) \tag{17}$$

The following quantities can be calculated with $\mathbf{S}_i(k|k)$

$$\hat{\mathbf{X}}(k+1|k) = \tfrac{1}{2n_s}\sum_{i=0}^{2n_s-1}\mathbf{X}(\mathbf{S}_i(k|k)) \tag{18}$$

$$\hat{\mathbf{Y}}(k+1|k) = \tfrac{1}{2n_s}\sum_{i=0}^{2n_s-1}\mathbf{Y}(\mathbf{S}_i(k|k)) \tag{19}$$

$$\mathbf{P}_{XX}(k+1|k) = \tfrac{1}{2n_s}\sum_{i=0}^{2n_s-1}\left[\mathbf{X}(\mathbf{S}_i(k|k)) - \hat{\mathbf{X}}(k+1|k)\right]\left[\mathbf{X}(\mathbf{S}_i(k|k)) - \hat{\mathbf{X}}(k+1|k)\right]^T \tag{20}$$

$$\mathbf{P}_{YY}(k+1|k) = \tfrac{1}{2n_s}\sum_{i=0}^{2n_s-1}\left[\mathbf{Y}(\mathbf{S}_i(k|k)) - \hat{\mathbf{Y}}(k+1|k)\right]\left[\mathbf{Y}(\mathbf{S}_i(k|k)) - \hat{\mathbf{Y}}(k+1|k)\right]^T \tag{21}$$

$$\mathbf{P}_{XY}(k+1|k) = \tfrac{1}{2n_s}\sum_{i=0}^{2n_s-1}\left[\mathbf{X}(\mathbf{S}_i(k|k)) - \hat{\mathbf{X}}(k+1|k)\right]\left[\mathbf{Y}(\mathbf{S}_i(k|k)) - \hat{\mathbf{Y}}(k+1|k)\right]^T \tag{22}$$

The equations (20) to (22) for obtaining the covariance and cross-covariance are considered suboptimal [9] at the expense of ensuring positive (semi)definiteness. To simplify subsequent discussion, the computational details are encapsulated into the following functions:

1. $\Omega(k, \mathbf{X}(\mathbf{S}), \mathbf{P}_{SS})$ takes the transformation equations, $\mathbf{X}(\mathbf{S})$ and the covariance matrix $\mathbf{P}_{ss}$ of the random vector **S** and generates the means of $\mathbf{X}(\mathbf{S})$.
2. $\Lambda(k, \mathbf{X}(\mathbf{S}), \mathbf{Y}(\mathbf{S}), \mathbf{P}_{SS})$ takes the transformation equations, $\mathbf{X}(\mathbf{S})$ and $\mathbf{Y}(\mathbf{S})$ and the covariance matrix $\mathbf{P}_{ss}$ of the random vector **S** and generates the cross-covariance between $\mathbf{X}(\mathbf{S})$ and $\mathbf{Y}(\mathbf{S})$. $\Lambda(k, \mathbf{X}(\mathbf{S}), \mathbf{P}_{SS})$ returns the covariance of $\mathbf{X}(\mathbf{S})$.

In both functions, *k* is the stage specifier for all the independent parameters.

The computation of the square root of a matrix involves solving for eigenvalues and eigenvectors is computationally expensive and simplication is desirable. For example, if $\mathbf{P}_{ss}$ has a diagonal structure, that is, $\mathbf{P}_{ss}$= diag($\mathbf{P}_i$), then

$$\pm\sqrt{n_s\mathbf{P}_{SS}} = \pm\sqrt{n_s}\sqrt{\mathbf{P}_{SS}} = \pm\sqrt{n_s}\,diag\left(\sqrt{\mathbf{P}_i}\right) \tag{23}$$

The JU method can now be applied to a Kalman Filtering problem:

$$\hat{\mathbf{S}}^{JU}(k+1|k) \approx \Omega\left(k|k, \mathbf{F}(\hat{\mathbf{S}}, \hat{\mathbf{U}}), diag\left(\mathbf{P}_{ss}, \mathbf{Cov}(\mathbf{U})\right)\right) \tag{24}$$

$$\mathbf{z}^{JU}(k+1) \approx -\Omega\left(k+1|k, \mathbf{G}(\hat{\mathbf{S}}, \hat{\mathbf{U}}), diag\left(\mathbf{P}_{ss}, \mathbf{Cov}(\mathbf{M})\right)\right) \tag{25}$$

$$\mathbf{P}_{ss}^{JU} \approx \Lambda\left(k|k, \mathbf{F}(\hat{\mathbf{S}}, \hat{\mathbf{U}}), diag\left(\mathbf{P}_{ss}, \mathbf{Cov}(\mathbf{U})\right)\right) \tag{26}$$

$$\mathbf{P}_{zz}^{JU} \approx \Lambda\left(k+1|k, \mathbf{G}(\hat{\mathbf{S}}, \hat{\mathbf{M}}), diag\left(\mathbf{P}_{ss}, \mathbf{Cov}(\mathbf{M})\right)\right) \tag{27}$$

$$\mathbf{P}_{sz}^{JU} \approx \Lambda\left(k+1|k, \mathbf{S}, \mathbf{G}(\hat{\mathbf{S}}, \hat{\mathbf{M}}), diag\left(\mathbf{P}_{ss}, \mathbf{Cov}(\mathbf{M})\right)\right) \tag{28}$$

where *diag(.)* here is the matrix formed by the argument matrices on the diagonal.

4 Map Building Formalism

The problem of map building can be treated as an optimisation problem and solved with Kalman Filter if

1. The parameters to be optimised are identified and error characteristics properly represented.
2. The relationships between the parameters and the information for optimising these parameters are available and the quality of the information is known.

This section is further subdivided into nine parts. Section 4.1 contains a discussion of the environment model and pinpoints the parameters to be optimised. Section 4.2 details all mapping scenarios that must be considered in order to grow the map primitives. Section 4.3 presents map building as a statistical optimisation problem and formulates solutions in the context of Kalman Filter. Two formulations are presented: The classical Global approach and the Relocation-Fusion approach taken by [15]. The author's formulations bear resemblance to their work, but are more general in the sense that they extend beyond feature-to-feature matching in order to tackle the more complex scenarios faced by sonar mapping. Section 4.4 explains why a corner should be merged to two intersecting partial planes, not one. Section 4.5 describes how a collinearity constraint should be validated. Section 4.6 to section 4.9 focus on the discussion and formulae development for other important map management details, namely, discrimination of phantom targets, incorporation of new measurement, as well as mergence and removal of existing primitives.

4.1 Map Primitives

The environmental model comprises two types of primitives:

Partial Plane is characterised by its state parameters $x_i(k) = [a_i(k) \quad b_i(k)]^T$ from the line equation $ax + by = a^2 + b^2$, the Cartesian coordinates of its approximate endpoints, and a status associated with each endpoint, indicating whether it is terminated with another partial plane to form a corner. When a wall is first detected, it is registered as a partial plane with only one endpoint. It is then grown to have two endpoints and extended as the robot moves along the wall.

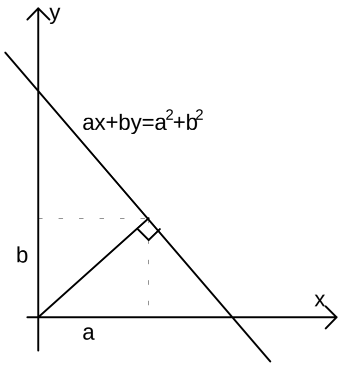


Figure 5 : Parameterisation of partial plane

Corner is characterised by its Cartesian coordinates $x_i(k) = [x_i(k) \quad y_i(k)]^T$ only. The sonar sensor provides no indication of its orientation.

Edge is similarly characterised by its Cartesian coordinates $x_i(k) = [x_i(k) \quad y_i(k)]^T$ only. The sonar sensor provides no indication of its orientation.

In addition, the covariance and cross-covariance among these features are also kept [15]. Each time a new primitive is added, it will expand the number of system state parameters by two. The current strategy also records the unclassifiable features as unknown. In the future, clusters would be formed to assist in obstacle avoidance path planning.

4.2. Growing Map Primitives

Since the robot is operating indoor, discrete feature elements are assumed to come from a few planes, so that they can be merged using some *collinearity constraint* to give a more realistic representation of the environment.

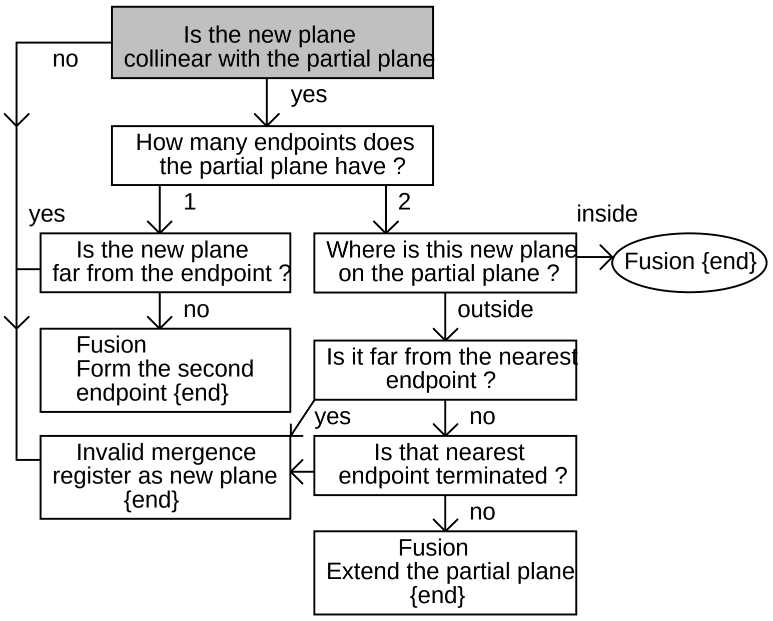

Is the new plane collinear with the partial plane
no
yes
How many endpoints does the partial plane have ?
1
2
yes
inside
Is the new plane far from the endpoint ?
Where is this new plane on the partial plane ?
Fusion {end}
no
outside
Fusion Form the second endpoint {end}
Is it far from the nearest endpoint ?
yes
no
Invalid mergence register as new plane {end}
Is that nearest endpoint terminated ?
no
Fusion Extend the partial plane {end}

Figure 6 : Conditions for growing map primitives with a plane measurement

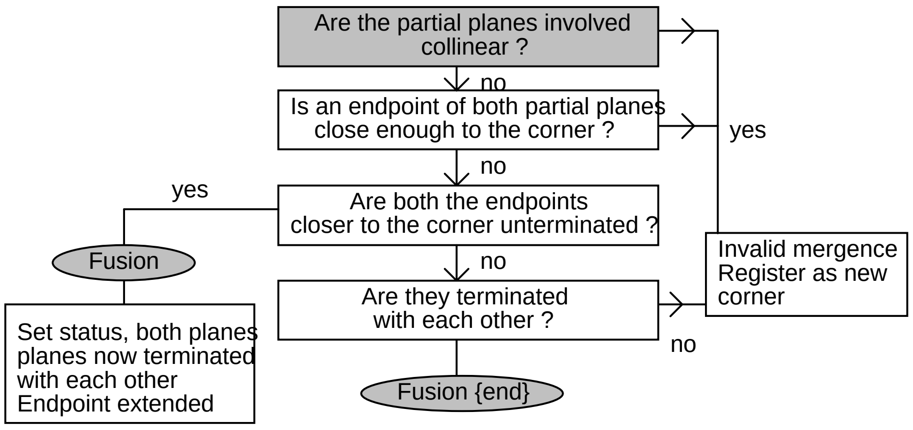

Are the partial planes involved collinear ?
no
Is an endpoint of both partial planes close enough to the corner ?
yes
no
yes
Are both the endpoints closer to the corner unterminated ?
Fusion
no
Invalid mergence Register as new corner
Are they terminated with each other ?
Set status, both planes planes now terminated with each other Endpoint extended
no
Fusion {end}

Figure 7 : Conditions for growing map primitives with a corner measurement

A planar measurement would be fused to a partial plane if it satisfies the conditions depicted in Figure 6. A corner measurement would be fused to an existing corner feature if it is close enough to it, otherwise it would be fused to two existing intersecting planes if it satisfies the conditions depicted in Figure 7. In a typical real environment, edges are produced by the artifacts on the walls such as moldings. While being excellent stationary landmarks for map building and localisation, they cannot be considered as collinear with the nearby walls. Therefore an edge is only fused to an existing edge if they are in the proximity of each other. For all greyed condition boxes in the figures, χ^2 tests (to be described later) are applied. Every time a re-observation of a feature/relation occurs, the state of every map feature would be updated because of their correlation. The unterminated endpoints of partial planes are projected to the new gradient determined by the new state parameters, whereas the terminated endpoint are re-calculated from the intersections of all pairs of partial planes marked as terminated with each other.

4.3 The Kalman Filter Formulation of Map Building Problem

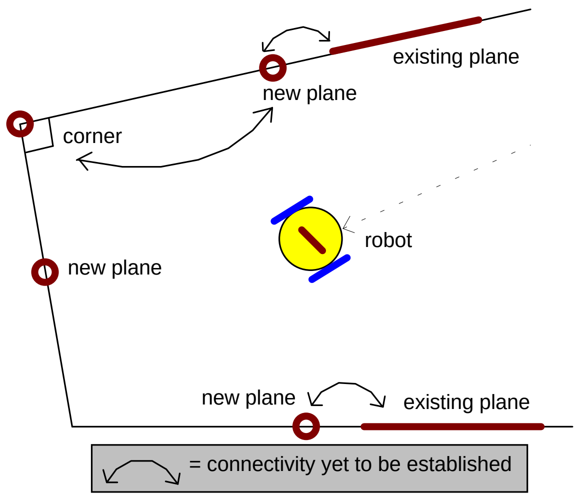


Figure 8 : Status of map and data fusion process at stage *k+1*

Under this section, the map building problem is first formulated according to the classical Global approach, a step considered fundamentally critical if a complete picture is to be gained and modifications in this paper to be fully comprehended. A few equations are then highlighted and modified according to the concept of the Relocation-Fusion approach introduced by [5]. All these are done in the specific context of the sonar mapping. After embedding IEKF or JUKF, the result is more general than the original formulation.

To begin, a snapshot of the map building scenario at stage *k+1* is depicted in Figure 8. The robot has just moved to a new position and sensed a few new features. It is now ready to use some features for localisation, and add the remaining features to the map.

The two dimensional coordinates and orientation (collectively known as the *state*) of the robot, as well as the speed of sound, at stage k is denoted by the random vector $\mathbf{x}_0(k) = [x(k) \quad y(k) \quad \theta(k) \quad c_s(k)]^T$ with respect to a global reference frame. Further assume that a partial map already exists, and the random parameter vectors of the existing features $\mathbf{x}_i(k)$ are concatenated with $\mathbf{x}_0(k)$ to form the *global state vector* $\mathbf{S}(k)$. $\mathbf{S}(k)$

contains all the parameters to be optimised, and $\mathcal{S}$ is the set of all state vectors to be optimised.

$$\mathbf{S}(k) = [\mathbf{x}_0(k) \quad \mathbf{x}_1(k) \quad \mathbf{x}_2(k) \quad \ldots \quad \mathbf{x}_n(k)]^T \tag{29}$$

$$\mathcal{S} = \bigcup_{i=0}^{n} \{x_i\} \tag{30}$$

At stage *k+1*, the robot travels to a new destination. The intermediate state of the robot $\hat{\mathbf{S}}(k+1|k)$ can be predicted as a function of its preceding state $\hat{\mathbf{S}}(k|k)$ and the input vector **U**(*k+1*) using the state transition equation (10) or (24). In this case **U**(*k+1*) is specified by the distance travelled by the left wheel and right wheel. Strictly speaking, the time history of wheel rotations is required to compute the intermediate state (i.e. *L* and *R* are both a function of time). In this experiment, the motion types are confined to linear translation and on the spot rotation only. If the motion is a translation, *L* and *R* should have equal sign; Likewise, if the motion is a rotation, *L* and *R* should have opposite sign.

$$\mathbf{U} = \mathbf{U}(k+1) = \begin{bmatrix} L(k+1) \\ R(k+1) \end{bmatrix} \tag{31}$$

$$\mathbf{Cov}(\mathbf{U}(k+1)) = \begin{bmatrix} k_L^2 \left|\hat{L}(k+1)\right| & 0 \\ 0 & k_R^2 \left|\hat{R}(k+1)\right| \end{bmatrix} \tag{32}$$

Since a new model has been developed in [3] for propagating random odometry errors, equation (12) and (26) are replaced by

$$\mathbf{P}_{ss}^{IEKF}(k+1|k) = \nabla_{\mathbf{S}}\mathbf{F}\mathbf{P}_{ss}(k|k)\nabla_{\mathbf{S}}\mathbf{F}^T + \mathbf{Odom}(\hat{\mathbf{U}}, \mathbf{Cov}(\mathbf{U})) \tag{33}$$

$$\mathbf{P}_{ss}^{JU}(k+1|k) = \Omega(k|k, \mathbf{F}(\hat{\mathbf{S}}, \hat{\mathbf{U}}), \mathbf{P}_{ss}) + \mathbf{Odom}(\hat{\mathbf{U}}, \mathbf{Cov}(\mathbf{U})) \tag{34}$$

where **Odom**() represents the new odometry error model developed in [3] that takes in the robot's wheel covariance matrix **Cov**(**U**(*k+1*)) and wheel turns $\hat{\mathbf{U}}$ and outputs the propagated covariance matrix.

Since the motion will only affect $\mathbf{x}_0(k|k)$, equation (10), (24), (33) and (34) can be simplified further. For the IEKF method,

$$\hat{\mathbf{x}}_0^{IEKF}(k+1|k) = \mathbf{F}(\hat{\mathbf{x}}_0(k|k), \hat{\mathbf{U}}(k+1)) \tag{35}$$

$$\mathbf{P}_{00}^{IEKF}(k+1|k) = \nabla_{\mathbf{x}_0}\mathbf{F}\mathbf{P}_{00}(k|k)\nabla_{\mathbf{x}_0}\mathbf{F}^T + \mathbf{Odom}(\hat{\mathbf{U}}, \mathbf{Cov}(\mathbf{U})) \tag{36}$$

$$\mathbf{P}_{j0}^{IEKF}(k+1|k) = \nabla_{\mathbf{x}_0}\mathbf{F}\mathbf{P}_{00}(k|k) \qquad \forall j > 0 \tag{37}$$

and for the JUKF method,

$$\hat{\mathbf{x}}_0^{JU}(k+1|k) = \Omega\Big(k|k, \mathbf{F}(\hat{\mathbf{x}}_0, \hat{\mathbf{U}}), diag\big(\mathbf{P}_{00}, \mathbf{Cov}(\mathbf{U})\big)\Big) \tag{38}$$

$$\mathbf{P}_{00}^{JU}(k+1|k) = \Lambda\Big(k|k, \mathbf{F}(\hat{\mathbf{x}}_0, \hat{\mathbf{U}}), \mathbf{P}_{00}\Big) + \mathbf{Odom}(\hat{\mathbf{U}}, \mathbf{Cov}(\mathbf{U})) \tag{39}$$

$$\mathbf{P}_{0j}^{JU}(k+1|k) = \Lambda\left(k|k, \mathbf{F}(\hat{\mathbf{x}}_0, \hat{\mathbf{U}}), \hat{\mathbf{x}}_j, \begin{bmatrix} \mathbf{P}_{00} & \mathbf{P}_{0j} \\ \mathbf{P}_{j0} & \mathbf{P}_{jj} \end{bmatrix}\right) \qquad \forall j > 0 \tag{40}$$

A measurement vector consists of a time of flight r_i and a direction Ψ_i to a target, and is denoted by

$$\mathbf{M} = \mathbf{M}_i(k+1) = \begin{bmatrix} r_i(k+1) & \psi_i(k+1) \end{bmatrix}^T \tag{41}$$

Every new measurement is tested against all the possible collinearity constraints set out in section 4.2, in order to grow the map primitives. A typical constraint would take the form

$$\mathbf{G}\big(\mathbf{S}(k+1), \mathbf{M}_i(k+1)\big) = \mathbf{0} \tag{42}$$

Based on this, the residual vector (also known is innovation in some literature) can be computed for each measurement,

$$\mathbf{z}_i^{IEKF}(k+1;\eta_r) = -\mathbf{G}\big(\hat{\mathbf{S}}(k+1|k), \hat{\mathbf{M}}_i(k+1)\big) \tag{43}$$

$$\mathbf{z}_i^{JU}(k+1) = -\Omega\big(k+1|k, \mathbf{G}(\hat{\mathbf{S}}, \hat{\mathbf{M}}_i), diag\big(\mathbf{P}_{ss}, \mathbf{Cov}(\mathbf{M}_i)\big)\big) \tag{44}$$

with error covariance,

$$\mathbf{P}_{zz}^{IEKF}(k+1|k;\eta_r) = \nabla_{\mathbf{S}}\mathbf{G}\mathbf{P}_{ss}(k+1|k)\nabla_{\mathbf{S}}\mathbf{G}^T + \nabla_{\mathbf{M}_i}\mathbf{G}\mathbf{Cov}(\mathbf{M}_i(k+1))\nabla_{\mathbf{M}_i}\mathbf{G}^T \tag{45}$$

$$\mathbf{P}_{zz}^{JU}(k+1|k) = \Lambda\big(k+1|k, \mathbf{G}(\hat{\mathbf{S}}, \hat{\mathbf{M}}_i), diag\big(\mathbf{P}_{ss}, \mathbf{Cov}(\mathbf{M}_i)\big)\big) \tag{46}$$

Where η_r is the rth $\hat{\mathbf{S}}(k+1|k+1)$ generated by IEKF, triggered with $\eta_0 = \hat{S}(k+1|k)$. Since the noise incurred on these residuals are not correlated, block processing is not necessary [2] (that is, they can be processed one at a time). Each residual vector $\mathbf{z}_i(k+1)$ is just a function of $\hat{\mathbf{x}}_0(k|k)$, the measurement $\hat{\mathbf{M}}_i(k+1)$ and the corresponding 'matched' map features, therefore there are significant zero submatrices in the Jacobian matrix on which simplification can be made. The following formulation involves only one feature, $\hat{\mathbf{x}}_i(k|k)$. Formulation involving two states (for example, fusing a new corner measurement to two existing intersecting partial planes) is similar so will not be detailed.

$$\mathbf{z}_i^{IEKF}(k+1;\eta_r) = -\mathbf{G}(\hat{\mathbf{x}}_0(k+1|k), \hat{\mathbf{x}}_i(k+1|k), \hat{\mathbf{M}}_i(k+1)) \tag{47}$$

$$\mathbf{z}_i^{JU}(k+1) = -\Omega\left(k+1|k, \mathbf{G}(\hat{\mathbf{x}}_0, \hat{\mathbf{x}}_i, \hat{\mathbf{M}}_i), diag\left(\begin{bmatrix} \mathbf{P}_{00} & \mathbf{P}_{0i} \\ \mathbf{P}_{i0} & \mathbf{P}_{ii} \end{bmatrix}, \mathbf{Cov}(\mathbf{M}_i)\right)\right) \tag{48}$$

and its error covariance,

$$\mathbf{P}_{zz}^{IEKF}(k+1|k;\eta_r) = \begin{bmatrix}\nabla_{\mathbf{x}_0}\mathbf{G} & \nabla_{\mathbf{x}_i}\mathbf{G}\end{bmatrix}\begin{bmatrix}\mathbf{P}_{00}(k|k) & \mathbf{P}_{0i}(k|k)\\ \mathbf{P}_{i0}(k|k) & \mathbf{P}_{ii}(k|k)\end{bmatrix}\begin{bmatrix}\nabla_{\mathbf{x}_0}\mathbf{G}^T\\ \nabla_{\mathbf{x}_i}\mathbf{G}^T\end{bmatrix} + \nabla_{\mathbf{M}_i}\mathbf{G}\mathbf{Cov}(\mathbf{M}_i(k+1))\nabla_{\mathbf{M}_i}\mathbf{G}^T \tag{49}$$

$$\mathbf{P}_{zz}^{JU}(k+1|k) = \Lambda\left(k+1|k, \mathbf{G}(\hat{\mathbf{x}}_0, \hat{\mathbf{x}}_i, \hat{\mathbf{M}}_i), diag\left(\begin{bmatrix}\mathbf{P}_{00} & \mathbf{P}_{0i}\\ \mathbf{P}_{i0} & \mathbf{P}_{ii}\end{bmatrix}, \mathbf{Cov}(\mathbf{M}_i)\right)\right) \tag{50}$$

The covariance of the measurement should account for the imperfect polygonal world assumption. For example, not all walls are strictly flat. It has a form depicted by equation (51) but more about the matrix values is presented later.

$$\mathbf{Cov}(\mathbf{M}_i(k+1)) = \begin{bmatrix}\sigma_{r_i}^2 & \sigma_{r_i\psi_i}\\ \sigma_{r_i\psi_i} & \sigma_{\psi_i}^2\end{bmatrix} \qquad (51$$

Kalman Filter equations require that the cross-covariance between the observation matrix and the state matrix be evaluated with equation (14) and (28). After that, the state and the error covariance matrix of the map features together with the robot’s position can be updated with equation (8) and (9). Once again, efficiency can be improved by processing the covariance matrix in disparate blocks. To update the state of $x_j \in \mathcal{S}$,

$$\mathbf{P}_{jz}^{IEKF}(k+1|k;\eta_r) = \mathbf{P}_{j0}(k+1|k)\nabla_{\mathbf{x}_0}\mathbf{G}^T + \mathbf{P}_{ji}(k+1|k)\nabla_{\mathbf{x}_i}\mathbf{G}^T \tag{52}$$

$$\mathbf{P}_{jz}^{JU}(k+1|k) = \Lambda\left(k+1|k, \hat{\mathbf{x}}_j, \mathbf{G}(\hat{\mathbf{x}}_0, \hat{\mathbf{x}}_i, \hat{\mathbf{M}}_i), diag\left(\begin{bmatrix}\mathbf{P}_{00} & \mathbf{P}_{0i} & \mathbf{P}_{0j}\\ \mathbf{P}_{i0} & \mathbf{P}_{ii} & \mathbf{P}_{ij}\\ \mathbf{P}_{j0} & \mathbf{P}_{ji} & \mathbf{P}_{jj}\end{bmatrix}, \mathbf{Cov}(\mathbf{M}_i)\right)\right) \tag{53}$$

For IEKF, let $\eta_{j,r}$ be the iterator for $\mathbf{x}_j$ only, $\hat{\mathbf{x}}_j(k+1|k+1)$, after the r^{th} iteration, starting with the initial value $\eta_{j,0} = \hat{\mathbf{x}}_j(k+1|k)$

$$\eta_{j,r+1} = \eta_{j,0} + \mathbf{P}_{jz}(k+1|k;\eta_r)\mathbf{P}_{zz}^{-1}(k+1|k;\eta_r)\left[\mathbf{z}_i(k+1;\eta_r) - \nabla_S G(\eta_0 - \eta_r)\right] \tag{54}$$

For JUKF, $\hat{\mathbf{x}}_j(k+1|k+1)$ is simply found with the following classical equation,

$$\hat{\mathbf{x}}_j(k+1|k+1) = \hat{\mathbf{x}}_j(k+1|k) + \mathbf{P}_{jz}(k+1|k)\mathbf{P}_{zz}^{-1}(k+1|k)\mathbf{z}_i(k+1) \tag{55}$$

Both cases share the same covariance update formula. For all combinations of state *m* and *n*,

$$\mathbf{P}_{mn}(k+1|k+1) = \mathbf{P}_{mn}(k+1|k) - \mathbf{P}_{mz}(k+1|k;\eta_r)\mathbf{P}_{zz}^{-1}(k+1|k;\eta_r)\mathbf{P}_{zn}(k+1|k;\eta_r) \tag{56}$$

Extra care has been taken when forming the covariance matrices required by JUKF. For example, in equation (52), if *j*=0 or *j*=*i*, then the composite covariance matrix passed into the JUKF function should only include $\mathbf{P}_{00}$, $\mathbf{P}_{0i}$, $\mathbf{P}_{i0}$ and $\mathbf{P}_{ii}$ only. Otherwise, a redundant (hence singular with zero determinant) covariance would be formed which triggers a fatal computer run-time error.

The process is then repeated until all observations have been processed. Since IEKF is also an extremely computationally demanding implementation, simplification becomes essential. The original algorithm is modified such that it terminates after exactly three iterations. Under this simplification, the iterator should only contain the states which affect all the matrix terms appearing in the IEKF algorithm, namely $\mathbf{x}_0$ and $\mathbf{x}_i$. The pseudo code is summarised as follows.

```
set r←0
set $\eta_{0,r} = \mathbf{x}_0(k+1|k)$, $\eta_{i,r} = \mathbf{x}_i(k+1|k)$
repeat {
        evaluate $\mathbf{z}_i, \nabla_{\mathbf{x}_0}\mathbf{G}, \nabla_{\mathbf{x}_i}\mathbf{G}, \mathbf{P}_{zz}, \mathbf{P}_{0z}, \mathbf{P}_{iz}$ at $\eta_{0,r}$ and $\eta_{i,r}$
        $\eta_{0,r+1} = \eta_{0,0} + \mathbf{P}_{oz}\mathbf{P}_{zz}^{-1}\left(\mathbf{z}_i - \nabla_{\mathbf{x}_0}\mathbf{G}(\eta_{0,0} - \eta_{0,r}) - \nabla_{\mathbf{x}_i}\mathbf{G}(\eta_{i,0} - \eta_{i,r})\right)$
        $\eta_{i,r+1} = \eta_{i,0} + \mathbf{P}_{iz}\mathbf{P}_{zz}^{-1}\left(\mathbf{z}_i - \nabla_{\mathbf{x}_0}\mathbf{G}(\eta_{0,0} - \eta_{0,r}) - \nabla_{\mathbf{x}_i}\mathbf{G}(\eta_{i,0} - \eta_{i,r})\right)$
        r←r+1
    } while (r<3)
$\forall \mathbf{x}_j \in \mathcal{M} \setminus \{\mathbf{x}_0, \mathbf{x}_i\}$, evaluate $\mathbf{P}_{jz}$ and update $\mathbf{x}_j$
$\forall \{\mathbf{x}_m, \mathbf{x}_n\} \subset \mathcal{M}$, update $\mathbf{P}_{mn}$
```

After the fusion of all measurements associated with the reobserved features, the remaining features are considered new and are simply incorporated into the global state. More information about fusing new observations is contained in section 4.4.

The Relocation-Fusion approach formulated by [15] makes a minor variation on the Global approach. The measurement error is first used to update the robot's state $\mathbf{x}_0$ ONLY. The improved $\mathbf{x}_0$ is then used to re-calculate the residual vector and all the related Jacobian matrices, which are then used to update the remaining map features. Stepwise, after $\mathbf{z}_i$ is computed, $\mathbf{x}_0$ and $\mathbf{P}_{00}$ and the cross-covariance between $\mathbf{x}_0$ and all other map feature $\mathbf{x}_n$ can be found,

$$\mathbf{x}_0(k+1|k+1) = \mathbf{x}_0(k+1|k) + \mathbf{P}_{0z}(k+1|k)\mathbf{P}_{zz}^{-1}(k+1|k)\mathbf{z}_i(k+1) \tag{57}$$

$$\mathbf{P}_{0n}(k+1|k+1) = \mathbf{P}_{0n}(k+1|k) - \mathbf{P}_{0z}(k+1|k)\mathbf{P}_{zz}^{-1}(k+1|k)\mathbf{P}_{zn}(k+1|k) \quad \forall \mathbf{x}_n \in \mathcal{M} \tag{58}$$

Now with the improved $\mathbf{x}_0$, $\mathbf{z}_i$, ($\nabla_{\mathbf{x}_0}\mathbf{G}$ and $\nabla_{\mathbf{x}_i}\mathbf{G}$ in case of IEKF), $\mathbf{P}_{zz}$, and $\mathbf{P}_{jz}$ can be re-generated, in this particular order, by applying equation (47) to (50). This is followed by the update of the states of all other map features, all covariance and cross-covariance, excluding $\mathbf{x}_0$ and $\mathbf{P}_{00}$, using equation (55) to (56). To embed the IEKF algorithm, iteration is first performed on $\mathbf{x}_0$. The matched feature $\mathbf{x}_i$ is included in the iteration after three runs. After this, the remaining states follow.

```
set r←0
set $\eta_{0,r} = \mathbf{x}_0(k+1|k)$, $\eta_{i,r} = \mathbf{x}_i(k+1|k)$
repeat {        /* Relocation with IEKF */
```

```
        evaluate $\mathbf{z}_i, \nabla_{\mathbf{x}_0}\mathbf{G}, \nabla_{\mathbf{x}_i}\mathbf{G}, \mathbf{P}_{zz}, \mathbf{P}_{0z}$ at $\eta_{0,r}$ and $\eta_{i,0}$
        $\eta_{0,r+1} = \eta_{0,0} + \mathbf{P}_{oz}\mathbf{P}_{zz}^{-1}\left(\mathbf{z}_i - \nabla_{\mathbf{x}_0}\mathbf{G}\left(\eta_{0,0} - \eta_{0,r}\right)\right)$
        r←r+1
      } while (r<3)
    $\forall \mathbf{x}_j \in \mathcal{M} \setminus \{\mathbf{x}_0\}$, evaluate $\mathbf{P}_{jz}$
    $\forall \mathbf{x}_j \in \mathcal{M}$, update $\mathbf{P}_{0j}$ and $\mathbf{P}_{j0}$
    set r←0
    repeat {          /* Fusion with IEKF */
        evaluate $\mathbf{z}_i, \nabla_{\mathbf{x}_0}\mathbf{G}, \nabla_{\mathbf{x}_i}\mathbf{G}, \mathbf{P}_{zz}, \mathbf{P}_{iz}, \mathbf{P}_{0z}$ at $\eta_{0,2}$ and $\eta_{i,r}$
        $\eta_{i,r+1} = \eta_{i,0} + \mathbf{P}_{iz}\mathbf{P}_{zz}^{-1}\left(\mathbf{z}_i - \nabla_{\mathbf{x}_i}\mathbf{G}\left(\eta_{i,0} - \eta_{i,r}\right)\right)$
        r←r+1
      } while (r<3)
    $\forall \mathbf{x}_j \in \mathcal{M} \setminus \{\mathbf{x}_i\}$, evaluate $\mathbf{P}_{jz}$
    $\forall \mathbf{x}_j \in \mathcal{M} \setminus \{\mathbf{x}_0, \mathbf{x}_i\}$, update $\mathbf{x}_j$
    $\forall \{\mathbf{x}_m, \mathbf{x}_n\} \subset \mathcal{M}$, update $\mathbf{P}_{mn}$ (exclude $\mathbf{P}_{00}$)
```

The Relocation-Fusion [15] approach has been shown to be less sensitive to position bias introduced by non-linearities and non-ideal odometry model, at the expense of optimality caused by the explicit removal of position information from map features update. In the implementation here, this approach is applied hand in hand with the IEKF and JUKF to achieve the maximal effect.

4.4 Why Should a New Corner be Fused to Two Intersecting Partial Planes ?

If a corner measurement fails to be fused to one of the standalone corners, an attempt will be made to fuse it to two intersecting planes. The reason behind fusing a corner to two intersecting planes is that, a corner measurement vector $[a_i(\mathbf{x}_0,\mathbf{Mi})\ b_i(\mathbf{x}_0,\mathbf{Mi})]^T$ has a size 2×1. If it sets up a collinearity constraint with one partial plane $[a\ b]^T$ only, then the residual vector formed would have a size 1×1, that is, $\mathbf{z}_i=[a_i a+b_i b-a^2-b^2]$. After fusion, three options for dealing with the corner are available:

- Discard the corner measurement. This wastes some information.
- Reparameterise the partial plane to create some kind of ‘plane-corner’ entity, which should have size(partial plane vector) + size(corner vector) - size(residual vector) = 3 state parameters. This leads to a series of avalanche effects. For example, it would later lead to some ‘plane-corner-plane’ entity and so forth. This complicates the map management process many folds.
- Register the corner as a new map feature too. However, since it has been used to fuse a partial plane before, any composite covariance matrix involving both of these features, like the state covariance matrix, would carry redundancy (not full rank), hence suffer the risk of singularity (zero determinant). Once again, this choice calls for complex map management scheme, as instances like this increase.

On the contrary, if it is not used at all, then there wouldn’t be redundancy to consider. If two intersecting partial planes could be found, the constraint would be ‘the corner should be collinear with two partial planes', so the residual vector would have a size of 2×1. This means the corner measurement can be discarded after fusion without wasting any information. On top of that, the two partial planes would be marked as terminated with each other, so the corner position could always be generated if required by subsequent path planning. The test to ensure that the two partial planes are not collinear is vital because if not, later if the two partial planes are merged into one, the resultant partial plane would have a 'hanging' terminated endpoint.

4.5 Validity of Collinearity Constraint

To access the validity of a constraint relationship, Mahalabonis distance test (or also known as χ^2 test) is applied to the residual vector :

$$\|\mathbf{z}_i\|^2_{\mathbf{P}_{zz}} = \mathbf{z}_i^T(k+1)\mathbf{P}_{zz}^{-1}(k+1|k)\mathbf{z}_i(k+1) \tag{59}$$

where $\|\mathbf{z}_i\|^2_{\mathbf{P}_{zz}}$ is the normalised sum of square of all the vector components. If the residual vector is assumed to be jointly Gaussian, then the expression will have a χ^2 distribution with degree of freedom determined by the rank of $\mathbf{P}_{zz}$. A one-sided acceptance interval is chosen to establish a 90% probability concentration ellipsoid in the distribution. A new measurement whose $\|\mathbf{z}_i\|^2_{\mathbf{P}_{zz}}$ falls in this acceptance interval is assumed to have satisfied the collinearity constraint set up with the existing feature(s). In this work, all residual vectors have a size of 2×1, so the degree of freedom is 2, and the acceptance interval is < 5.991.

To improve computational efficiency, $\mathbf{z}_i$ which is considerably different from **0** is rejected without going through the test, to avoid the series of matrix operations. At this stage, the issue of features falling into more than one validation gates has been temporarily put aside. This problem can arise either when the position covariance is too large, or when two existing map features are very close together but are not yet merged. This difficult issue will be investigated in the future.

4.6 Distinguishing Phantom Targets

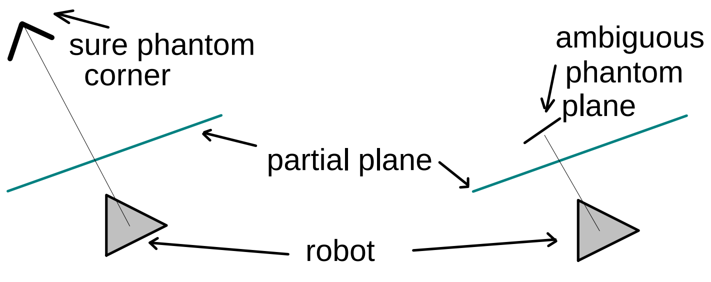


Figure 9 : Example of treatment of phantom targets

Local maps are preserved. Each feature in the local map has a parameter indicating which state it has been fused to. Therefore, the knowledge of where a particular map primitive was observed, is available. When the map is sufficiently complete, many phantom targets caused by specular reflection can be eliminated by checking whether the line of sights from the positions they were observed are blocked by some partial planes. If the phantom targets are too close to some partial planes they are considered ambiguous and would not be eliminated. Experimentally it has been validated that, due to specularity, corners and edges are more likely to cause phantom targets than planes.

4.7 Fusion of the Remaining New Features

After localisation, the fusion of the remaining features will make use of the estimated robot position. Each new feature **x***i* is a function **H**() of the robot's position **x***0* and a measurement vector **M**i. For each new feature, the error covariance can be calculated :

$$\hat{\mathbf{x}}_i^{IEKF}(k+1|k+1) = \mathbf{H}\left(\hat{\mathbf{x}}_0(k+1|k+1), \hat{\mathbf{M}}_i(k+1)\right) \tag{60}$$

$$\mathbf{P}_{ii}^{IEKF}(k+1|k+1) = \nabla_{\mathbf{x}_0}\mathbf{H}\mathbf{P}_{00}(k+1|k+1)\nabla_{\mathbf{x}_0}\mathbf{H}^T + \nabla_{\mathbf{M}_i}\mathbf{H}\mathbf{Cov}(\mathbf{M}_i(k+1))\nabla_{\mathbf{M}_i}\mathbf{H}^T \tag{61}$$

$$\hat{\mathbf{x}}_i^{JU}(k+1|k+1) = \Omega\left(k+1|k+1, \mathbf{H}(\hat{\mathbf{x}}_0, \hat{\mathbf{M}}_i), diag\left(\mathbf{P}_{00}, \mathbf{Cov}(\mathbf{M}_i)\right)\right) \tag{62}$$

$$\mathbf{P}_{ii}^{JU}(k+1|k+1) = \Lambda\left(k+1|k+1, \mathbf{H}(\hat{\mathbf{x}}_0, \hat{\mathbf{M}}_i), diag\left(\mathbf{P}_{00}, \mathbf{Cov}(\mathbf{M}_i)\right)\right) \tag{63}$$

and all the cross-covariance among the new features and the existing features are also generated. Let *j* denote the objects already in the map,

$$\mathbf{P}_{ij}^{IEKF}(k+1|k+1) = \nabla_{\mathbf{x}_0}\mathbf{H}\mathbf{P}_{0j}(k+1|k+1) \qquad \forall j \neq i \tag{64}$$

$$\mathbf{P}_{ij}^{JU}(k+1|k+1) = \Lambda\left(k+1|k+1, \mathbf{H}(\hat{\mathbf{x}}_0, \hat{\mathbf{M}}_i), \hat{\mathbf{x}}_j, diag\left(\begin{bmatrix}\mathbf{P}_{00} & \mathbf{P}_{0j} \\ \mathbf{P}_{j0} & \mathbf{P}_{jj}\end{bmatrix}, \mathbf{Cov}(\mathbf{M}_i)\right)\right) \tag{65}$$

As a reminder, if *j*=0, the composite matrix passed to the JUKF function should comprise **P***00* and **Cov**(**M***i*) only. By symmetry,

$$\mathbf{P}_{ji}(k+1|k+1) = \mathbf{P}_{ij}^T(k+1|k+1) \tag{66}$$

all of which are then inserted into $\hat{\mathbf{S}}(k+1|k+1)$ and $\mathbf{P}_{ss}(k+1|k+1)$.

4.8 Simultaneous Encounter of Collinear Features

There would be occasions when two or more collinear features are encountered at the same stage. For instance, this situation would occur if the robot reaches a corner and observes the corner and the walls that form the corner for the first time. When this situation arises, the planar feature is first incorporated into the global state vector as a new feature. The corner feature is then regarded as the observation for that new feature.

4.9 Removal of Redundant Primitives

Removal of redundant primitives would occur when

1. Two existing partial planes are actually collinear and adjacent to each other.
2. Two existing corners are the same.
3. Two existing edges are the same.
4. An existing corner appears to be located at the intersection of two existing partial planes.

In such cases, internal fusion is performed by forming residual vectors in a manner similar to section 5. For the last three cases, the map primitve growth assessment/produre is similar to that depicted in Figure 7, except that when invalidity occurs, integration of the new feature need not be carried out. However, to merge two partial planes, the procedure depicted in Figure 10 is followed.

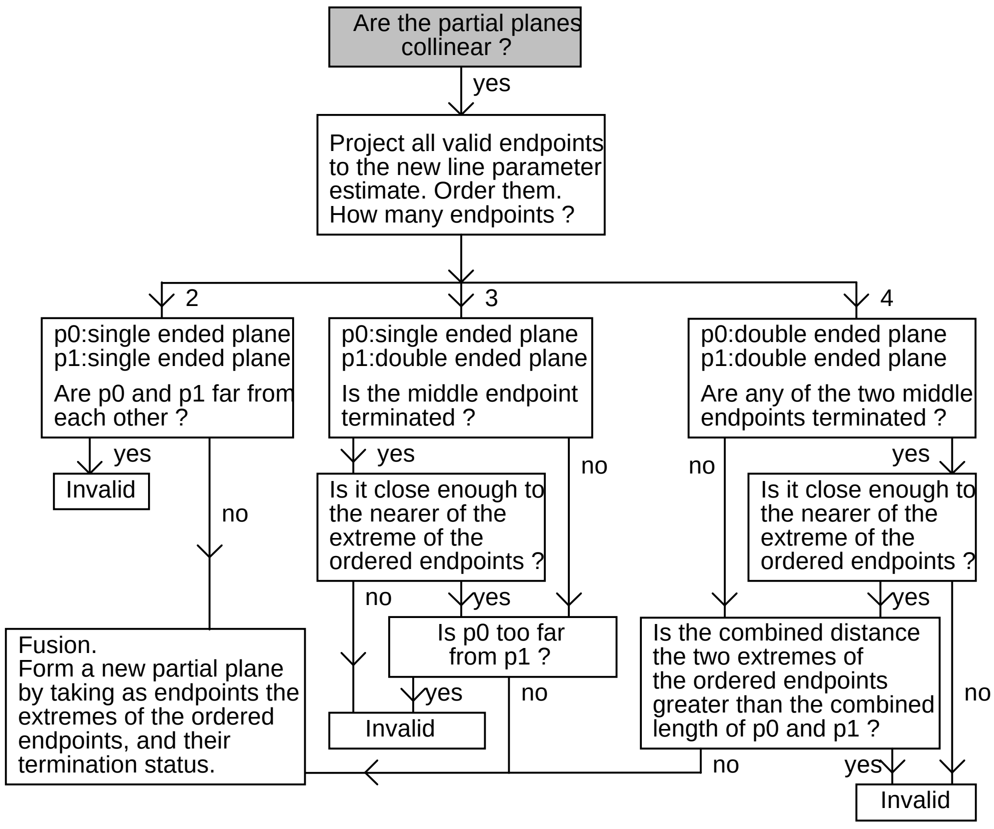

Are the partial planes collinear ?
yes
Project all valid endpoints to the new line parameter estimate. Order them. How many endpoints ?
2
3
4
p0:single ended plane
p1:single ended plane
Are p0 and p1 far from each other ?
yes
Invalid
no
p0:single ended plane
p1:double ended plane
Is the middle endpoint terminated ?
yes
no
Is it close enough to the nearer of the extreme of the ordered endpoints ?
no
yes
Is p0 too far from p1 ?
yes
no
Invalid
p0:double ended plane
p1:double ended plane
Are any of the two middle endpoints terminated ?
no
yes
Is it close enough to the nearer of the extreme of the ordered endpoints ?
yes
no
Is the combined distance the two extremes of the ordered endpoints greater than the combined length of p0 and p1 ?
no
yes
Invalid
Fusion.
Form a new partial plane by taking as endpoints the extremes of the ordered endpoints, and their termination status.

Figure 10 : Conditions for merging two existing partial planes in map

As an illustration, an example involving two states is given here: When two existing features, with states $\mathbf{x}_i(k)$ and $\mathbf{x}_j(k)$ respectively, are to form a collinearity constraint, redundancy can be removed by enhancing the estimation of $\mathbf{x}_i(k)$ with $\mathbf{x}_j(k)$, and discarding $\mathbf{x}_j(k)$ afterwards. Suppose $\hat{\mathbf{x}}_i(k|k)$ is to be enhanced by $\hat{\mathbf{x}}_j(k|k)$

Let

$$\mathbf{G}'(\mathbf{x}_i(k|k),\mathbf{x}_j(k|k)) = \mathbf{0} \tag{67}$$

$$\mathbf{z}^{IEKF}(k|k) = -\mathbf{G}'(\hat{\mathbf{x}}_i(k|k),\hat{\mathbf{x}}_j(k|k)) \tag{68}$$

$$\mathbf{z}^{JU}(k|k) = -\Omega\left(k|k,\mathbf{G}'(\hat{\mathbf{x}}_i,\hat{\mathbf{x}}_j),\begin{bmatrix}\mathbf{P}_{ii} & \mathbf{P}_{ij}\\ \mathbf{P}_{ji} & \mathbf{P}_{jj}\end{bmatrix}\right) \tag{69}$$

then the new estimate of state m is

$$\hat{\mathbf{x}}'_m(k|k) = \hat{\mathbf{x}}_m(k|k) + \mathbf{P}_{mz}(k|k)\mathbf{P}_{zz}^{-1}(k|k)\mathbf{z}(k|k) \qquad \forall \mathbf{x}_m \in \mathcal{S} \setminus \{\mathbf{x}_j\} \tag{70}$$

and the new cross covariance between any two states *m* and *n* can then be re-estimated

$$\mathbf{P}'_{mn}(k|k) = \mathbf{P}_{mn}(k|k) - \mathbf{P}_{mz}(k|k)\mathbf{P}_{zz}^{-1}(k|k)\mathbf{P}_{zn}(k|k) \qquad \forall\{\mathbf{x}_m, \mathbf{x}_n\} \subset \mathcal{S} \setminus \{\mathbf{x}_j\} \tag{71}$$

where

$$\mathbf{P}_{mz}^{IEKF}(k|k) = \mathbf{P}_{mi}(k|k)\nabla_{\mathbf{x}_i}\mathbf{G}'^{T} + \mathbf{P}_{mj}(k|k)\nabla_{\mathbf{x}_j}\mathbf{G}'^{T} \tag{72}$$

$$\mathbf{P}_{mz}^{JU}(k|k) = \Lambda\left(k|k, \hat{\mathbf{x}}_m, \mathbf{G}'(\hat{\mathbf{x}}_i, \hat{\mathbf{x}}_j), \begin{bmatrix} \mathbf{P}_{ii} & \mathbf{P}_{ij} & \mathbf{P}_{im} \\ \mathbf{P}_{ji} & \mathbf{P}_{jj} & \mathbf{P}_{jm} \\ \mathbf{P}_{mi} & \mathbf{P}_{mj} & \mathbf{P}_{mm} \end{bmatrix} \right) \tag{73}$$

and the measurement error covariance is

$$\mathbf{P}_{zz}^{IEKF}(k|k) = \begin{bmatrix} \nabla_{\mathbf{x}_i}\mathbf{G}' & \nabla_{\mathbf{x}_j}\mathbf{G}' \end{bmatrix} \begin{bmatrix} \mathbf{P}_{ii}(k|k) & \mathbf{P}_{ii}(k|k) \\ \mathbf{P}_{ji}(k|k) & \mathbf{P}_{jj}(k|k) \end{bmatrix} \begin{bmatrix} \nabla_{\mathbf{x}_i}\mathbf{G}'^{T} \\ \nabla_{\mathbf{x}_j}\mathbf{G}'^{T} \end{bmatrix} \tag{74}$$

$$\mathbf{P}_{zz}^{JU}(k|k) = \Lambda\left(k|k, \mathbf{G}'(\hat{\mathbf{x}}_i, \hat{\mathbf{x}}_j), \begin{bmatrix} \mathbf{P}_{ii} & \mathbf{P}_{ij} \\ \mathbf{P}_{ji} & \mathbf{P}_{jj} \end{bmatrix} \right) \tag{75}$$

and by symmetry,

$$\mathbf{P}'_{mn}(k|k) = \mathbf{P}'^{T}_{nm}(k|k) \tag{76}$$

After these operations, all $\mathbf{x}_j$ related terms in **S** and $\mathbf{P}_{ss}$ can be removed. The three-run IEKF algorithm is applied in a similar fashion so it will not be elaborated further. Also if the constraint involves three states, as in the case of fusing a corner to two intersecting partial planes, the formulation is similar.

It is also possible to exploit the orthogonality constraint among partial planes. However, not all intersecting walls in today's indoor environments are strictly perpendicular, so the idea has not been implemented even though it can be accommodated. If implemented, a χ^2 test would also be applied to assess orthogonality.

5 Implementation Details

This section evaluates the equations given in the last few subsections for all scenarios the current implementation accounts for. Let *B* denotes the effective wheelbase, after moving, the robot's new position can be computed from the following state update equations:

$$\hat{\mathbf{x}}_0(k+1|k) = \begin{bmatrix} \hat{x}(k|k) + \hat{r}(k+1)\left(\sin\hat{\theta}(k|k) - \sin\left(\hat{\theta}(k|k) + \frac{\hat{R}(k+1)-\hat{L}(k+1)}{B}\right)\right) \\ \hat{y}(k|k) + \hat{r}(k+1)\left(\cos\left(\hat{\theta}(k|k) + \frac{\hat{R}(k+1)-\hat{L}(k+1)}{B}\right) - \cos\hat{\theta}(k|k)\right) \\ \hat{\theta}(k|k) + \frac{\hat{R}(k+1)-\hat{L}(k+1)}{B} \\ c_s(k|k) \end{bmatrix} \tag{77}$$

where

$$\hat{r}(k+1) = \frac{B}{2}\left(\frac{\hat{L}(k+1)+\hat{R}(k+1)}{\hat{L}(k+1)-\hat{R}(k+1)}\right) \tag{78}$$

The Jacobian matrices with respect to $\hat{\mathbf{x}}_0(k|k)$ and $\hat{\mathbf{U}}(k+1)$ can be found in [3] hence will not be reproduced here. From this point onwards, all circumflexes and suffices are dropped to alleviate viewing.

To match a planar measurement to a partial plane, let $\alpha_i = \psi_i + \theta$

$$\mathbf{z} = -\begin{bmatrix} \frac{x}{2}(1+\cos 2\alpha_i) + \frac{y}{2}\sin 2\alpha_i + r_i c_s \cos\alpha_i \\ \frac{x}{2}\sin 2\alpha_i + \frac{y}{2}(1-\cos 2\alpha_i) + r_i c_s \sin\alpha_i \end{bmatrix} + \begin{bmatrix} a_i \\ b_i \end{bmatrix} \tag{79}$$

$$\nabla_{\mathbf{M}_i}\mathbf{G} = \begin{bmatrix} c_s\cos\alpha_i & -x\sin 2\alpha_i + y\cos 2\alpha_i - r_i c_s \sin\alpha_i \\ c_s \sin\alpha_i & x\cos 2\alpha_i + y\sin 2\alpha_i + r_i c_s\cos\alpha_i \end{bmatrix} \tag{80}$$

$$\nabla_{\mathbf{x}_o}\mathbf{G} = \begin{bmatrix} \cos^2\alpha_i & \sin\alpha_i\cos\alpha_i & -x\sin 2\alpha_i + y\cos 2\alpha_i - r_i c_s\sin\alpha_i & r_i\cos\alpha_i \\ \sin\alpha_i\cos\alpha_i & \sin^2\alpha_i & x\cos 2\alpha_i + y\sin 2\alpha_i + r_i c_s\cos\alpha_i & r_i\sin\alpha_i \end{bmatrix} \tag{81}$$

$$\nabla_{\mathbf{x}_i}\mathbf{G} = -\mathbf{I}_{2\times 2} \tag{82}$$

To match a corner feature to an existing corner, or an edge feature to an existing edge feature,

$$\mathbf{z} = -\begin{bmatrix} x + r_i c_s\cos\alpha_i \\ y + r_i c_s \sin\alpha_i \end{bmatrix} + \begin{bmatrix} a_i \\ b_i \end{bmatrix} \tag{83}$$

$$\nabla_{\mathbf{M}_i}\mathbf{G} = \begin{bmatrix} c_s\cos\alpha_i & -r_i c_s\sin\alpha_i \\ c_s\sin\alpha_i & r_i c_s\cos\alpha_i \end{bmatrix} \tag{84}$$

$$\nabla_{\mathbf{x}_0}\mathbf{G} = \begin{bmatrix} 1 & 0 & -r_i c_s\sin\alpha_i & r_i\cos\alpha_i \\ 0 & 1 & r_i c_s\cos\alpha_i & r_i\sin\alpha_i \end{bmatrix} \tag{85}$$

$$\nabla_{\mathbf{x}_i}\mathbf{G} = -\mathbf{I}_{2\times 2} \tag{86}$$

To match a corner feature to two partial planes $\mathbf{x}_i$ and $\mathbf{x}_j$,

$$\mathbf{z} = -\begin{bmatrix} a_i(x + r_i c_s\cos\alpha_i) + b_i(y + r_i c_s\sin\alpha_i) - a_i^2 - b_i^2 \\ a_j(x + r_i c_s\cos\alpha_i) + b_j(y + r_i c_s\sin\alpha_i) - a_j^2 - b_j^2 \end{bmatrix} \tag{87}$$

$$\nabla_{\mathbf{M}_i}\mathbf{G} = \begin{bmatrix} c_s(a_i\cos\alpha_i + b_i\sin\alpha_i) & r_i c_s(-a_i\sin\alpha_i + b_i\cos\alpha_i) \\ c_s(a_j\cos\alpha_i + b_j\sin\alpha_i) & r_i c_s(-a_j\sin\alpha_i + b_j\cos\alpha_i) \end{bmatrix} \tag{88}$$

$$\nabla_{\mathbf{x}_0}\mathbf{G} = \begin{bmatrix} a_i & b_i & r_i c_s(-a_i\sin\alpha_i + b_i\cos\alpha_i) & r_i(a_i\cos\alpha_i + b_i\sin\alpha_i) \\ a_j & b_j & r_i c_s(-a_j\sin\alpha_i + b_j\cos\alpha_i) & r_i(a_j\cos\alpha_i + b_j\sin\alpha_i) \end{bmatrix} \tag{89}$$

$$\nabla_{\mathbf{x}_i}\mathbf{G} = \begin{bmatrix} x + r_i c_s\cos\alpha_i - 2a_i & y + r_i c_s\sin\alpha_i - 2b_i \\ 0 & 0 \end{bmatrix} \tag{90}$$

$$\nabla_{\mathbf{x}_i}\mathbf{G} = \begin{bmatrix} 0 & 0 \\ x + r_i c_s \cos\alpha_i - 2a_j & y + r_i c_s \sin\alpha_i - 2b_j \end{bmatrix} \tag{91}$$

For fusion of a new planar measurement,

$$\mathbf{x}_i = \left(x\cos\alpha_i + y\sin\alpha_i + r_i c_s\right)\begin{bmatrix}\cos\alpha_i \\ \sin\alpha_i\end{bmatrix} \tag{92}$$

$$\nabla_{\mathbf{x}_o}\mathbf{H} = \begin{bmatrix} \cos^2\alpha_i & \sin\alpha_i\cos\alpha_i & -x\sin 2\alpha_i + y\cos 2\alpha_i - r_i c_s\sin\alpha_i & r_i\cos\alpha_i \\ \sin\alpha_i\cos\alpha_i & \sin^2\alpha_i & x\cos 2\alpha_i + y\sin 2\alpha_i + r_i c_s\cos\alpha_i & r_i\sin\alpha_i \end{bmatrix} \tag{93}$$

$$\nabla_{\mathbf{M}_i}\mathbf{H} = \begin{bmatrix} c_s\cos\alpha_i & -x\sin 2\alpha_i + y\cos 2\alpha_i - r_i c_s\sin\alpha_i \\ c_s\sin\alpha_i & x\cos 2\alpha_i + y\sin 2\alpha_i + r_i c_s\cos\alpha_i \end{bmatrix} \tag{94}$$

and for a new corner measurement,

$$\mathbf{x}_i = \begin{bmatrix} x + r_i c_s\cos\alpha_i \\ y + r_i c_s\sin\alpha_i \end{bmatrix} \tag{95}$$

$$\nabla_{\mathbf{x}_0}\mathbf{H} = \begin{bmatrix} 1 & 0 & -r_i c_s\sin\alpha_i & r_i\cos\alpha_i \\ 0 & 1 & r_i c_s\cos\alpha_i & r_i\sin\alpha_i \end{bmatrix} \tag{96}$$

$$\nabla_{\mathbf{M}_i}\mathbf{H} = \begin{bmatrix} \cos\alpha_i & -r_i c_s\sin\alpha_i \\ \sin\alpha_i & r_i c_s\cos\alpha_i \end{bmatrix} \tag{97}$$

Finally for removal of redundancy, if the feature type of $\mathbf{x}_i$ and $\mathbf{x}_j$ are the same, such as fusing a partial plane to a partial plane, fusing a corner to a corner, or fusing an edge to an edge,

$$\mathbf{z} = -\begin{bmatrix} a_i \\ b_i \end{bmatrix} + \begin{bmatrix} a_j \\ b_j \end{bmatrix} \tag{98}$$

$$\nabla_{\mathbf{x}_i}\mathbf{G}' = \mathbf{I}_{2\times 2} \tag{99}$$

$$\nabla_{\mathbf{x}_j}\mathbf{G}' = -\mathbf{I}_{2\times 2} \tag{100}$$

If $\mathbf{x}_i$ is a corner and $\mathbf{x}_j$, $\mathbf{x}_k$ are two partial planes,

$$\mathbf{z} = -\begin{bmatrix} a_i a_j + b_i b_j - a_j^2 - b_j^2 \\ a_i a_k + b_i b_k - a_k^2 - b_k^2 \end{bmatrix} \tag{101}$$

$$\nabla_{\mathbf{x}_i}\mathbf{G}' = \begin{bmatrix} a_j & b_j \\ a_k & b_k \end{bmatrix} \tag{102}$$

$$\nabla_{\mathbf{x}_j}\mathbf{G}' = \begin{bmatrix} a_i - 2a_j & b_i - 2b_j \\ 0 & 0 \end{bmatrix} \tag{103}$$

$$\nabla_{\mathbf{x}_j}\mathbf{G}' = \begin{bmatrix} 0 & 0 \\ a_i - 2a_k & b_i - 2b_k \end{bmatrix} \tag{104}$$

6 Simulation Results

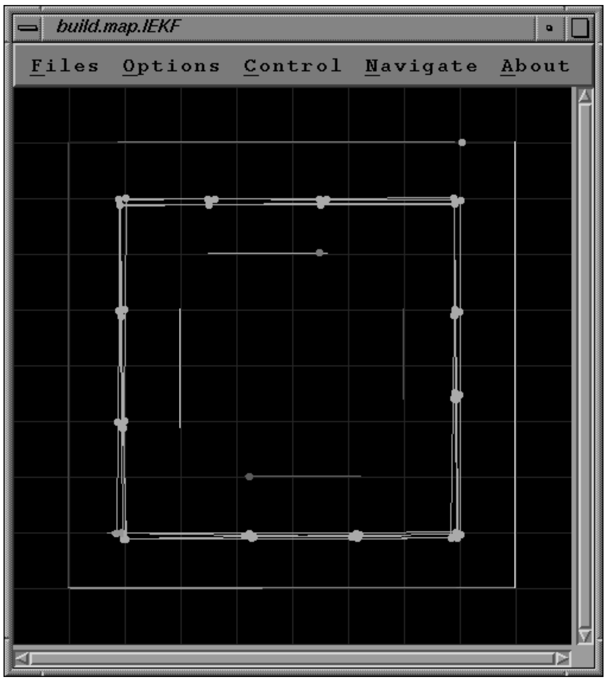


Figure 11 : Map building simulation experiment using IEKF and JUKF

IEKF and JUKF have first been tested with simulation. Starting at (2.0m, -8.0m, 0 rad), a virtual robot was driven around a virtual square corridor four times. The walls in the artificial environment are denoted by the equations x=1, x=3, x=7, x=9, y=-1, y=-3, y=-7 and y=-9 respectively. Other data include wheelbase, B=0.4m, k_R=1.2×10^{-3}m$^{1/2}$, k_L=10^{-3}m$^{1/2}$, $\sigma^2_{r_i} = \sigma^2_{\psi_i} = 10^{-6}\,\mathrm{m}^2$ and $\sigma_{r_i\psi_i} = 0\,\mathrm{m}$. In each round, the robot stops a total of 12 times to rescan the environment. The total distance travelled is 4×8×4=128 metres, and the total number of scanning points is 4×12=48. The comparison of the pre-filtering position errors and post-filtering position errors at all stops is show in Figure 12.

Both methods yield approximately the same performance, with JUKF showing more ‘jitters’ at some scanning points. At the end of the second round, both IEKF simulation and JUKF simulation register a total of 9 partial planes, but at the end of the fourth round, the IEKF simulation registers a total of 13 partial planes whereas the JUKF simulation registers a total of 18 partial planes. By single stepping the program, it has been confirmed that the only reason for the appearance of redundant partial planes is the failure in passing the χ^2 test. The algorithms for growing map primitives have also been verified as working correctly and satisfactorily.

Running on a SGI INDY and code compiled with GNU GCC 2.7.0, the speed required by JUKF to complete the simulation is approximately 5 times that required by IEKF.

7 Experimental Results

Experiments have been carried out in four artificial environment erected with cardboard boxes and they are shown from Figure 13(a) to Figure 16(a). The odometry of the robot has been calibrated to reduce systematic errors, and the parameters required by the non-systematic error model have been obtained in [3] prior to experiment.

Since the cardboard boxes were being lined up manually taking the gridlines on the parquetry floor as reference, the variance associated with the time of flight measurement and angular measurement were set larger than that achievable by the sonar sensor [8] in order for the collinearity constraints to hold. The initial value of speed of sound was set to 342.5 m/s, which is the mean value at the time. In fact, for all four experiments, the following tentative values were used:

standard deviation of time of flight = 1.6×10^{-5} s
standard deviation of direction = 2.4°
initial standard deviation of c_s = 0.18 m/s

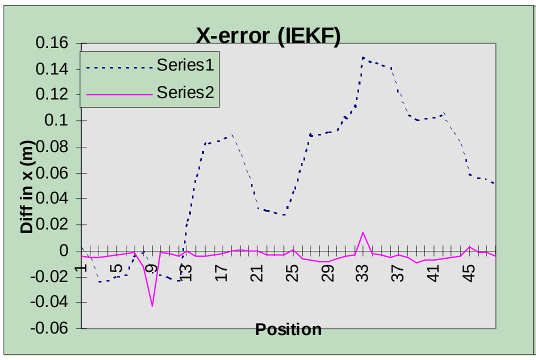
The resultant maps are shown in Figure 13(c)(d) to Figure 16(c)(d). The (b) subfigures show the raw sonar measurements detected at various positions (before position correction) being superimposed onto the same diagrams, and the ‘scan lines’ from one of the position indicate the typical number of features the sonar sensor can capture at any one time. The grid spacing is 1 metre. It has been noticed that the sensor detects the gaps between the cardboard boxes as edges. They should not be regarded as some artificial aids to the mapping process as in a real environment, wall moldings are often found to give rise to the same phenomenon.

In the first environment, the robot navigated around the enclosure once, first clockwise (Figure 13) then counterclockwise (Figure 14). The maps generated with JUKF and IEKF are very similar except that for the clockwise experiment, IEKF does not merge the two partial planes on the right which are supposed to belong to the same ‘wall’. Also, IEKF does not fuse several corners to the intersecting planes in both runs. Their fusion to the planes are found to be hindered by the χ^2 tests. A comparison of the covariance generated by IEKF and JUKF for a few features indicates that JUKF in general tends to generate larger covariance. As a result, the error ellipses for ‘related’ features are more likely to overlap and more mergence can be observed. Despite the minor imperfection, all post-filtering maps show that only one partial plane is generated for each wall, and most of the corners have been successfully fused to two intersecting partial planes, hence well defined intersections can be observed. Also, repetitive observations of the same edge are all successfully merged into one edge map feature. All unterminated endpoints of partial planes have also been properly projected to the line parameters. In the counterclockwise run, a phantom target can be observed (refer to (b)) but it has been eliminated by its neareast partial plane. Overall, the maps produced by IEKF and JUKF are very similar.

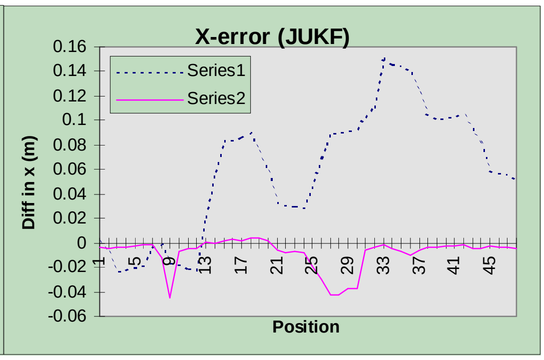
The third experiment (second environment) is more challenging. The robot was programmed to repetitively enter, make a 180° turn, exit an enclosure four times to investigate the long term performance of both filters. Once again, both filters have remained consistent throughout the navigation. JUKF produces a map with all features correctly merged. IEKF’s performance closely matches that of JUKF, with only one corner not fused to two partial planes and two edge features not identified as belonged to the same physical edge. Three phantom corners are retained in the raw data map, but are subsequently eliminated in the post-filtering maps by the partial planes blocking their lines of sight. Once again, the maps produced by IEKF and JUKF are very similar, but the speed of JUKF is significantly slower.

In the fourth experiment (third environment), the robot was programmed to follow a rectangular path four times. One side of the 'wall' was indented by about 0.5 metre. The observations made about this experiment are virtually the same as those made in the third experiment, so no repetition is necessary. At first glance, it might seem wasteful not to extend the partial planes with the edge features by exploiting another collinearity constraint. This idea is found to be impractical in real world for two important reasons:

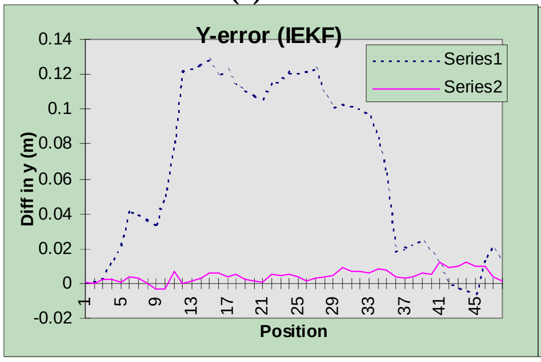
- Most edge reflections are generated by artifacts which are not necessarily collinear with the wall, such as wall moldings.
- Even if the edges are really collinear with the partial planes, once they have been merged into the partial planes, they have to be eliminated, much like merging a corner to two partial planes. Unlike the corner, the fused edge cannot be regenerated when required. This in fact reduces the number of useful landmarks for localisation!



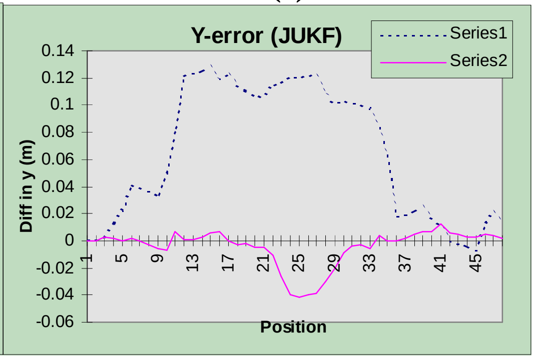
(a)



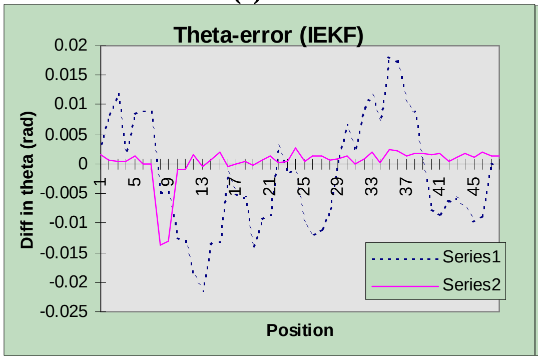
(b)



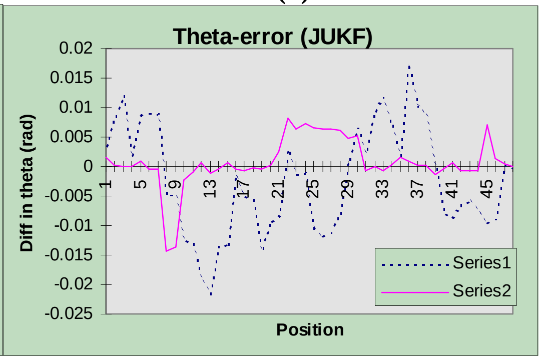
(c)



(d)

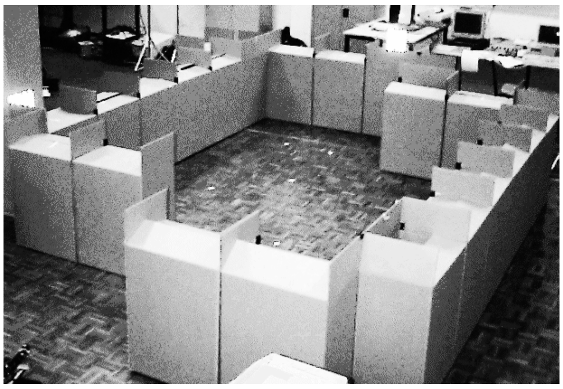


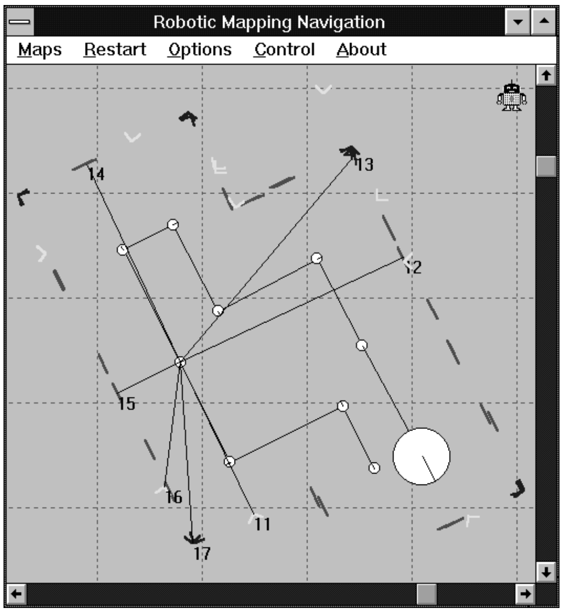
(e)

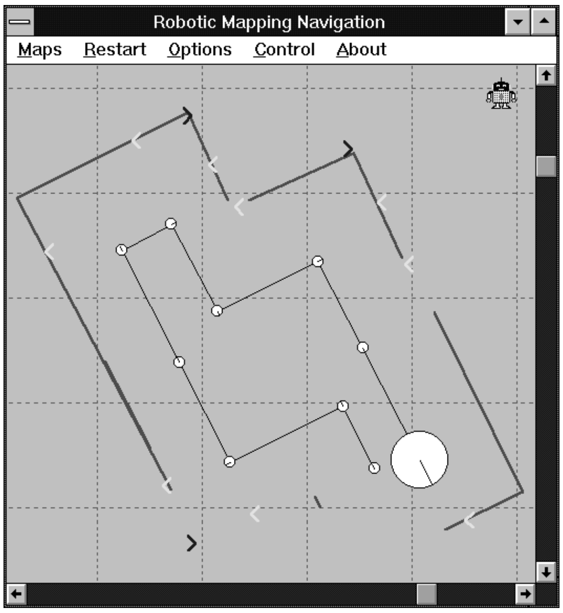


(f)

Figure 12 : Comparison of position and orientation errors, before and after IEKF and JUKF. The solid lines represent post-filtering errors, whereas the dotted lines represent pre-filtering errors


(a) Actual environment


Robotic Mapping Navigation
Maps Restart Options Control About
14
13
12
15
16
11
17
(b) Raw Sensor Data


Robotic Mapping Navigation
Maps Restart Options Control About
(c) IEKF

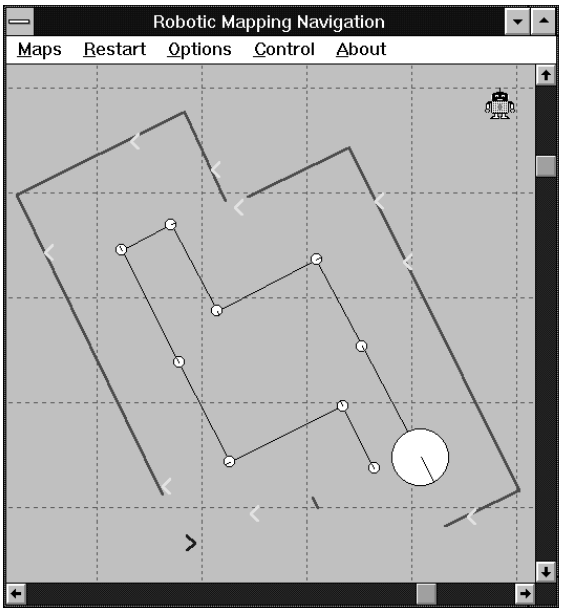
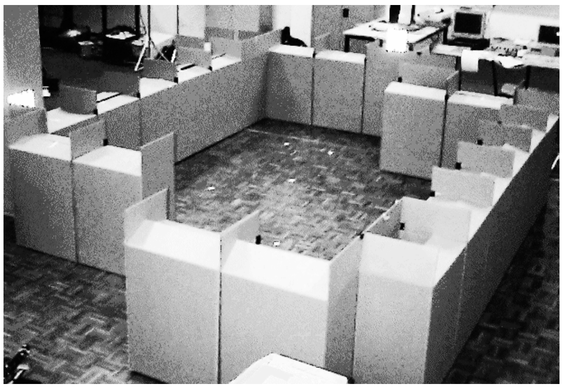
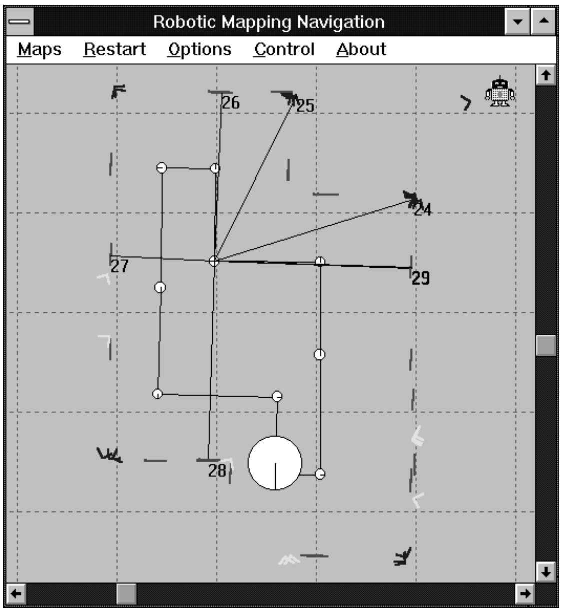

Robotic Mapping Navigation
Maps Restart Options Control About
(d) JUKF

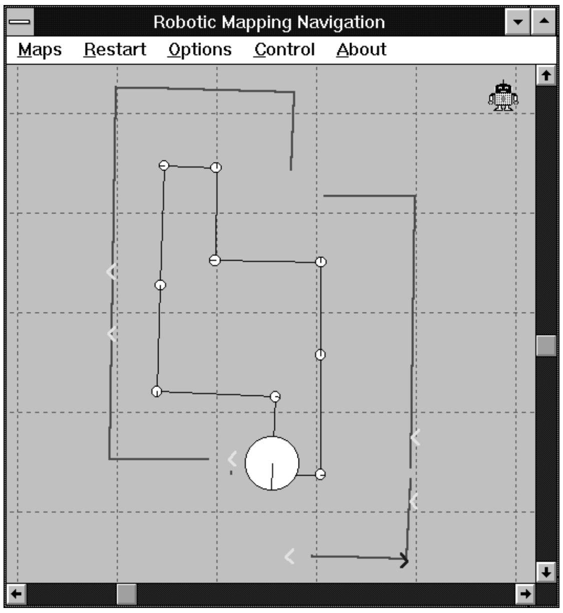
Figure 13 : The first environment, with the robot navigating clockwise once



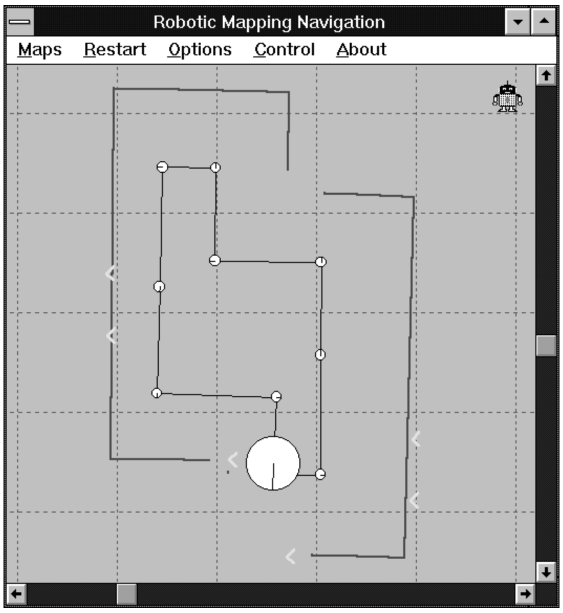
(a) Actual environment


Robotic Mapping Navigation
Maps Restart Options Control About
26
25
24
27
29
28

(b) Pre-filtering Perception

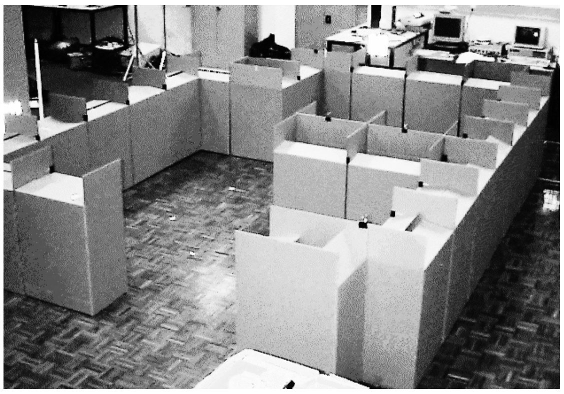

Robotic Mapping Navigation
Maps Restart Options Control About

(e) IEKF

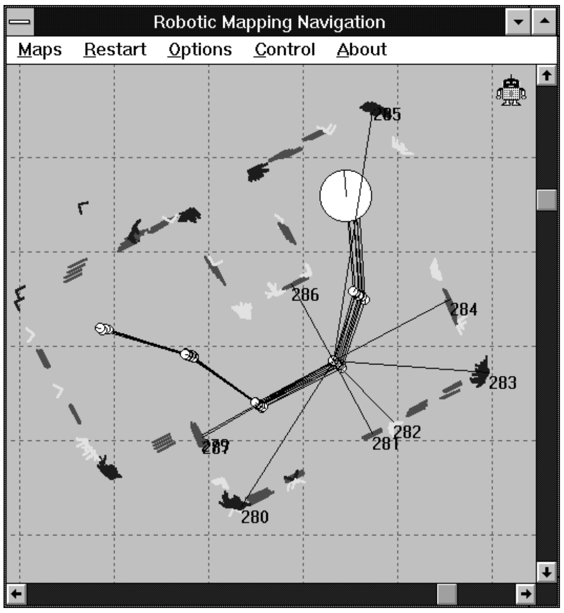

Robotic Mapping Navigation
Maps Restart Options Control About

(d) JUKF

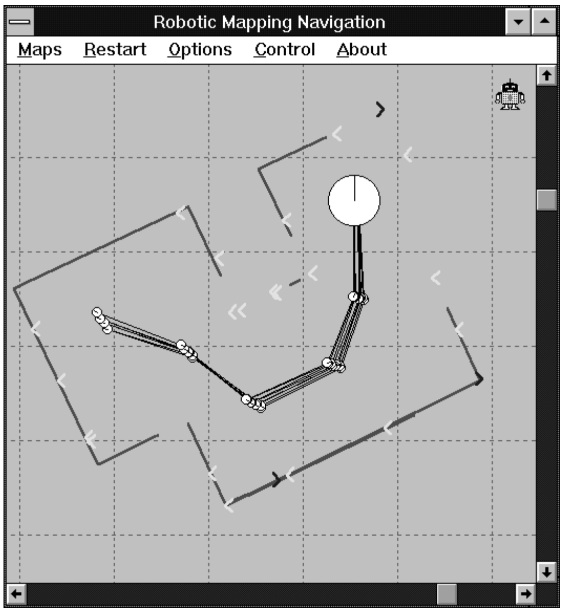
Figure 14 : The first environment, with the robot navigating counterclockwise once

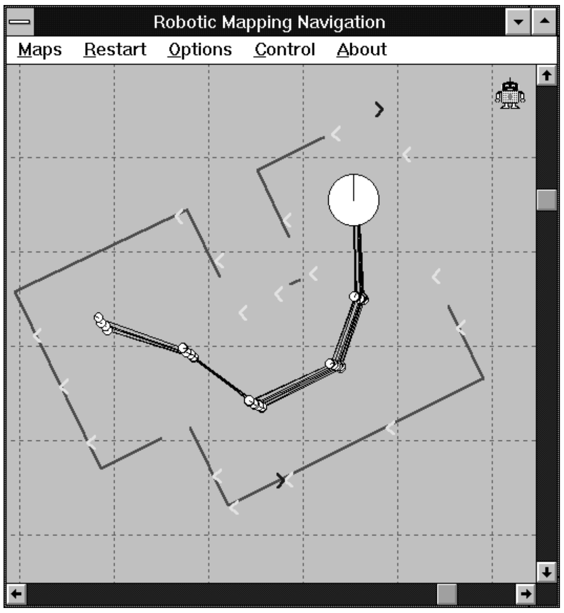


(a) Actual environment


Robotic Mapping Navigation
Maps Restart Options Control About
285
286
284
283
282
281
287
280

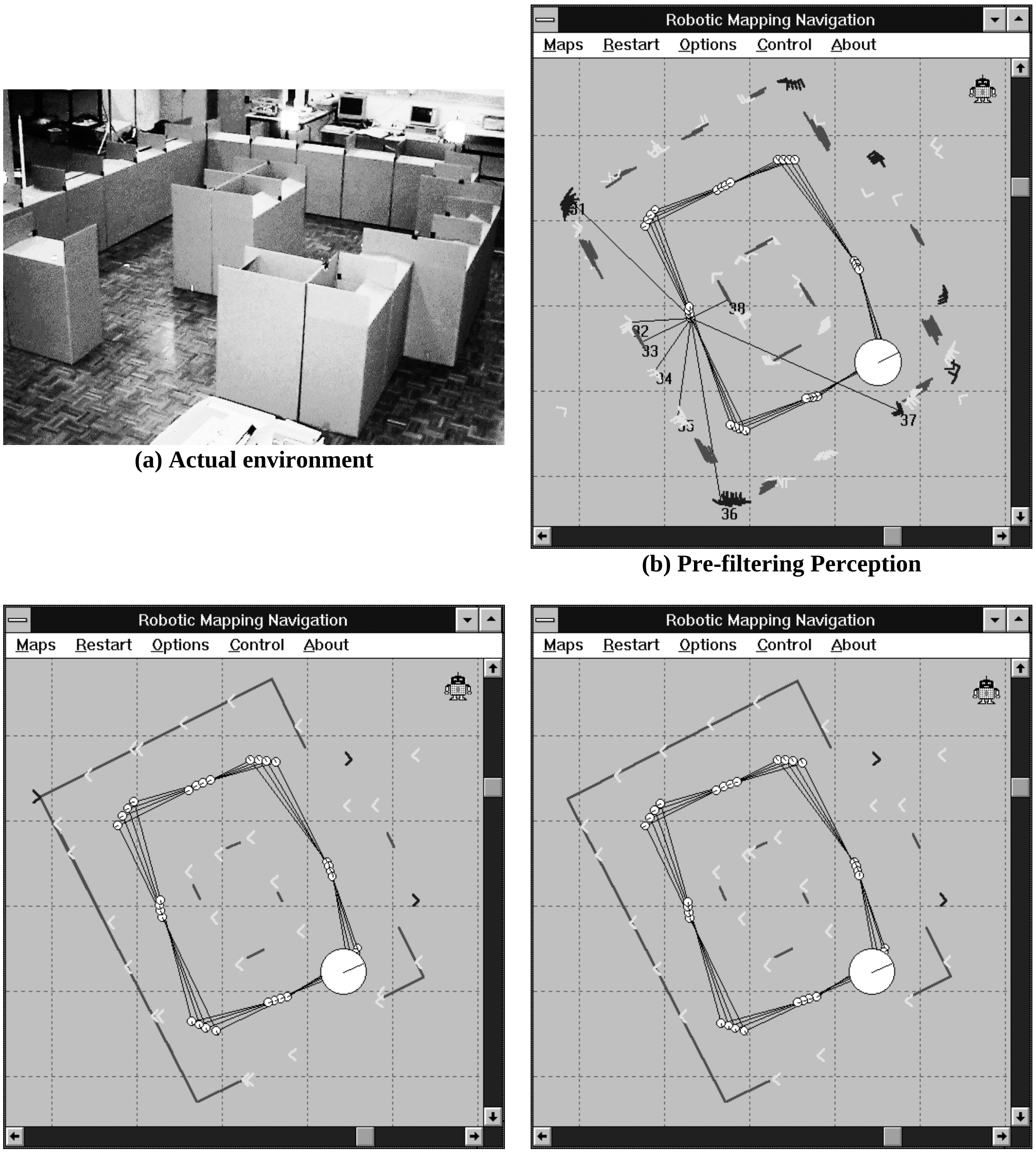
(b) Pre-filtering Perception


Robotic Mapping Navigation
Maps Restart Options Control About


Robotic Mapping Navigation
Maps Restart Options Control About

(e) IEKF **(d) JUKF**

Figure 15 : The second environment, with the robot navigating into and out of the enclosure four times


Robotic Mapping Navigation
Maps Restart Options Control About
31
32
33
34
35
36
37
38

(a) Actual environment

(b) Pre-filtering Perception

(c) IEKF

(d) JUKF

Figure 16 : The third environment, with the robot repeating a retangular path four times

8 Conclusion

The capability of autonomous navigation by mapping of our mobile robot system in some simple environments has been demonstrated. IEKF and JUKF have been employed to deal with the problem of covariance propagation through nonlinear transformation, and their strengths and weaknesses with regards to accuracy and speed have been compared with simulated and real data. It has been shown that the accuracy demonstrated by IEKF is comparable to that by JUKF and is in fact sufficient in practice. While eliminating the tedium of deriving Jacobian matrices, JUKF is less efficient compared to IEKF. The algorithm is now being intensively upgraded to enhance its robustness and efficiency. Current research focal points include the elimination of the storage and update of the covariance between two features if it is found to be small, in order to improve the speed and memory requirement of the algorithm. Also under investigation is a map matching strategy to re-establish robot’s position when its uncertainty is too large or when the accumulation of position bias becomes significant.

9 Acknowledgment

Mr. Greg Curmi’s assistance in the design of the robot and sonar implementation is gratefully acknowledged.

10 References

[1] Ayache, N. and Faugeras, O.D. "Maintaining Representation of the Environment of a Mobile Robot", IEEE Transactions on Robotics and Automation, Vol 5, No 6, Dec 1989, pp.804-819.

[2] Bar-Shalom, Y. and Li, X.R. "Estimation and Tracking: Principles, Techniques and software", Boston, London: Artech House Inc., 1993.

[3] Chong, K.S. and Kleeman, L. "Accurate Odometry and Error Modelling for a Mobile Robot", to appear in Proceedings of the 1997 IEEE International Conference on Robotics and Automation.

[4] Dudek, G. et al "Just-in-time Sensing: Efficiently Combining Sonar and Laser Range Data for Exploring Unknown Worlds", Proceedings 1996 IEEE International Conference on Robotics and Automation, pp.667-672.

[5] Durrant-Whyte, H.F. and Leonard, J.J. "Simultaneous Map Building and Localisation for an Autonomous Robot", IEEE/RSJ International Workshop on Intelligent Robots and Systems IROS ' 91, Nov 3-5, 1991, pp.1442-1447.

[6] Gonzalez, J. "An Iconic Position Estimator for a 2D Laser RangeFinder" Proceedings 1992 IEEE International Conference on Robotics and Automation, Vol 3, pp.2646-2651.

[7] Hong, M.L. and Kleeman, L. "A Low Sample Rate 3D Sonar Sensor for Mobile Robots", IEEE International Conference on Robotics and Automation, 1995, pp.3015-3020.

[8] Jazwinski, A.H. "Stochastic Processes and Filtering Theory", New York: Academic Press, 1970.

[9] Julier, S. and Uhlmann, J. "A General Method for Approximating Nolinear Transformations of Probability Distributions", 1995, WWW.

[10] Kleeman, L. and Kuc, R. "Mobile Robot Sonar for Target Localization and Classification", The International Journal of Robotics Research, Vol. 14, No 4, August 1995, pp.295-318.

[11] Kuc, R. and Siegel, M.W. "Physically Based Simulation Model for Acoustic Sensor Robot Navigation", IEEE Transactions on Pattern Analysis and Machine Intelligence, Vol PAMI-9, No 6, Nov 1987, pp.766-778.

[12] Leonard, J.J. and Durrant-Whyte H.F. "Dynamic Map Building for an Autonomous Mobile Robot", International Journal of Robotics Research, August 1992, Vol 11, pp.286-298.

[13] Lim, J.H. and Cho, D.W. "Experimental Investigation Of Mapping and Navigation Based on Certainty Grid Using Sonar Sensors", Robotica, Jan-Feb 1993, Vol 11, Iss:part 1, pp.7-17.

[14] Lu, F. and Milios, E.E. "Optimal Global Pose Estimation for Consistent Sensor Data Registration", Proceedings 1995 IEEE International Conference on Robotics and Automation, pp.93-100.

[15] Moutarlier, P. and Chatila, R. "Stochastic Multisensory Data Fusion for Mobile Robot Location and Environment Modeling", 5th International Symposium on Robotics Research, Tokyo, 1989, pp.85-94.

[16] Moravec, H.P. and Elfes, A. "High Resolution Map From Wide-Angle Sonar", Proceedings 1985, IEEE International Conference on Robotics and Automation, pp.116-121.

[17] Neira, J. et al “Multisensor Mobile Robot Localisation” Proceedings 1996 International Conference on Robotics and Automation, pp.673-679.

[18] Ohya, A. et al "Exploring Unknown Environment and Map Construction Using Ultrasonic Sensing of Normal Direction of Walls", pp.485-492.

[19] Oriolo, G. et al "On-Line Map Building and Navigation for Autonomous Mobile Robots", IEEE International Conference on Robotics and Automation, 1995, pp.2900-2906.

[20] Pagac, D. et al “An Evidential Approach to Probabilistic Map-Building”, Proceedings 1996 IEEE International Conference on Robotics and Automation, pp. 745-750.

[21] Rencken, W.D. "Concurrent Localisation and Map Building for Mobile Robots Using Ultrasonic Sensors", Proceedings 1993 IEEE International Conference On Robotics and Automation, Vol 3, pp.2192-2197.

[22] Smith, R.C. and Cheeseman, P. "On the Representation and Estimation of Spatial Uncertainty", The International Journal of Robotics Research, Vol 5, No 4, 1986, pp.56-68.