Simultaneous Localisation and Map Building for Autonomous Guided Vehicles

Stephen Borthwick and Hugh Durrant-Whyte
Robotics Research Group
Department of Engineering Science
University of Oxford
Oxford, OX1 3PJ United Kingdom

Abstract

As Autonomous Guided Vehicles claim wider applications, the need for a flexible navigation system increases. Previously developed systems designed to incorporate localisation and environmental modelling have suffered from poor operation speed, often depending upon a "stop look update" policy. We describe a multi-track Extended Kalman Filter navigation system which offers real time localisation while simultaneously constructing a map consisting of geometric features, each in turn described by an extended Kalman Filter. Dynamic operation is achieved by utilising the increased scan rate and data quality of infra-red scanning to overcome the need for a multi-hypothesis framework or other such computationally expensive paradigm.

1 Introduction

Since the introduction of wire guided Autonomous Guided Vehicles (AGVs) into industrial environments, the importance of flexibility in their operation has been recognised. The development of navigation techniques utilising either artificial or naturally occurring beacons has improved the performance of AGVs insofar as they are no longer constrained to fixed paths. Despite this improvement however, they have still depended upon their operating environment having been mapped *a priori*. In practice of course, few environments are static so navigation systems have been developed which update a stored description of their environment or even model the environment without any *a priori* information. Many of the original map building techniques assumed accurate knowledge of the sensor location [2], thus requiring only the problem of feature localisation to be addressed. However, this artificial requirement reduced the effectiveness of such systems. To overcome this, techniques have been developed which simultaneously produce a map of the environment and achieve localisation.

The simultaneous map construction and localisation technique developed by Moutarlier and Chatila [5] utilises the correlations between features within a unique reference frame to maintain a consistent model of the environment. The data association problem is addressed by the implementation of a *nearest neighbour* validation technique. However, this method inevitably produces "false alarms" which are not accounted for within the system's framework. Also, by resolving the correlations between each feature at every update, the imposed computation burden prevents real time operation.

The system developed by Leonard and Cox [3] is capable of simultaneous localisation and environment mapping. This uses information obtained from Polaroid sonar sensors and a Bayesian Multiple Hypothesis algorithm to achieve localisation and environmental modelling. This technique does however, suffer from poor operation speed: sonar carrier characteristics impose a long propagation delay (resulting in a complete scan of an environment taking in the order of several seconds to complete). Also, the Multiple Hypothesis algorithm imposes a large computation demand when the environment includes a realistic number of features and the sensing device employed supplies ambiguous information. This latter problem also affected the system developed by Ayache and Faugeras [4].

In order that a flexible AGV may be realised, it must be capable of dynamic operation. That is, the vehicle must navigate and model its environment without the need to apply a "stop, look, update" process. In order to facilitate this, we employ an *Amplitude Modulated Continuous Wave* infra-red range scanner developed by Brownlow [6] at Oxford University. The advantages this sensor has over sonar for example, include a significantly higher data acquisition rate (typically 2 scans per second), an extended operational range (0.3m to 15m) and non specular reflection characteristics.

The navigation algorithm we have developed is a multi-track, multi-target Extended Kalman Filter based approach, updated with observations from the infra-red range scanner. This system enables an AGV to localise within an unknown environment with respect to its initial state (an arbitrary origin) while modelling its environment. Localisation and map construction and maintenance is achieved in real time without the requirement of pausing to obtain observations or undertake computation.

In Section 2, we outline the development of the theoretical basis to the localisation, validation, environment modelling and map maintenance techniques. In Section 3, we show experimental results obtained from the system and discuss their implications.

2 AGV Localisation and Environment Modelling System

In this section we show how the AGV and the features in its environment are modelled and related to each other through a Extended Kalman Filtering (EKF) formalism.

The basis of the unified system is a multi-track, multi-target EKF. The individual filters are of the same format as those advocated by Leonard and Cox [3]. In their system however, these are integrated into a multi-hypothesis framework whereas we propose maintaining a single hypothesis concerning the world state. We acknowledge that errors may occur within this framework and account for them by utilising the increased information content in the infra-red scanning technique. That is, the scan rate and data quality of the infra-red sensor provides a high density of accurate observations with which the EKFs in the system can be updated. The rapid update with good quality observations enables features that have been located erroneously to be identified and removed through map maintenance techniques.

2.1 Vehicle and Environment Geometry

At the outset of operation, the environment origin is arbitrarily set to be the location of the AGV. Similarly, the 0^o bearing is set to the initial heading of the AGV. At any proceeding point in time, the AGV state, $\mathbf{x}(k) = [x, y, \theta]^T$ is referenced with respect to this origin:

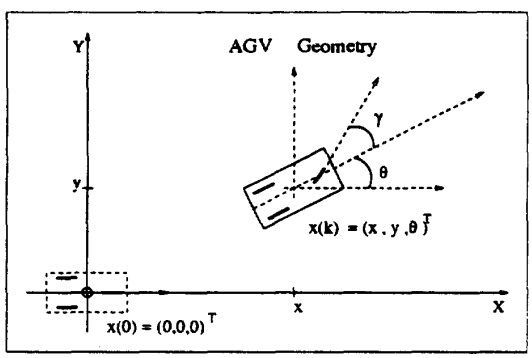

Figure 1: System Geometry

The input to the system $\mathbf{u}(k)$ consists of two components: AGV velocity V and the bearing of the steering wheel γ. The system is described according to the "truth" model:

$$\begin{aligned} \dot{x}(t) &= V\cos(\theta(t)+\gamma) \\ \dot{y}(t) &= V\sin(\theta(t)+\gamma) \\ \dot{\theta}(t) &= \frac{V}{B}\sin(\gamma) \end{aligned} \tag{1}$$

where B, the base line of the vehicle is unity.

The AGV's environment is assumed to consist of features described by three geometric primitives: line sections, corners and edges. Other features are able to be included (such as curved segments) however these are not apparent in the environments that the Oxford Navigator vehicle (on which the system is to be based) operates. The features are described in a minimalistic manner. Corners and edges are described by a Cartesian point co-ordinate $\mathbf{p} = [p_x, p_y]^T$ (with respect to the AGV origin). Line segments by a *normal* representation $\mathbf{p} = [p_R, p_\theta]^T$ as shown in Figure 2.

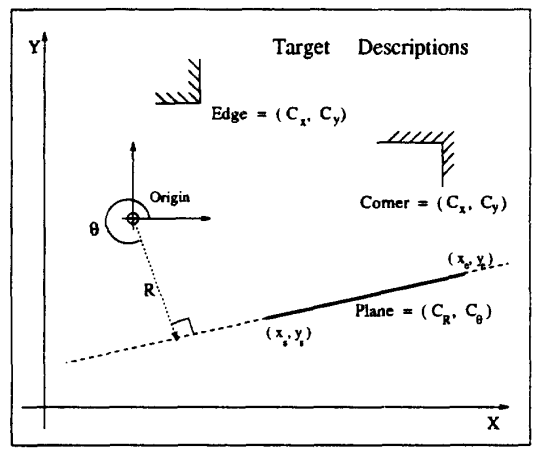

Figure 2: Target Geometry

These features are held within a map structure, consisting of a linked list of each feature type:

$$\begin{aligned} corner_0 &= (p_x, p_y) & edge_0 &= (p_x, p_y) \\ corner_1 &= (p_x, p_y) & edge_1 &= (p_x, p_y) \\ &\vdots & &\vdots \\ corner_n &= (p_x, p_y) & edge_n &= (p_x, p_y) \end{aligned}$$

The feature observations are obtained from processing the range scans of the infra-red scanner. A *range, bearing* pair $[R, \phi]^T$ is produced for each corner, edge and line targets observed. The observation model for the system is:

$$\mathbf{z} = \mathbf{h}(\mathbf{p}, \mathbf{x}) + \mathbf{w} = \begin{bmatrix} R \\ \phi \end{bmatrix} \tag{2}$$

where $\mathbf{w}$ is the motion noise (assumed to be Gaussian with zero mean), R is the range to the feature and ϕ is the angle from the vehicle heading θ to the target.

2.2 Navigation System

The system achieves simultaneous vehicle localisation and map construction by applying the following simplified algorithm:

IF a feature is observed
- Calculate time from last update Δt
- Predict AGV position (and variance)
- Match observation with targets held in map
- IF matched:
  - Update AGV position
  - Update target position
- ELSE:
  - Initialise a new target
  - Update AGV filter
  - Assign any end-points found

From the algorithm, it can be seen that localisation and map maintenance occurs upon each observation rather than at the end of each scan. This enables the computation overhead to be distributed throughout operation. It also however, prevents using traditional techniques of data association i.e. optimally (by some criteria) allocating observations to known features. Instead, we use the increased information obtained from infra-red scanning to minimise the errors from a nearest neighbour validation approach (as described in Section 2.4).

2.3 AGV Localisation

The AGV's position (or state) is obtained using an EKF to recursively update the optimal state estimate with respect to a minimum mean square error criteria. The initial stage of filtering, prediction of the AGV state is obtained by use of the state transition equation shown in Equation 4:

$$\begin{aligned}\mathbf{x}(k+1) &= \mathbf{f}(\mathbf{x}(k),\mathbf{u}(k)+\mathbf{v}(k)) \qquad (3)\\ &= \begin{bmatrix} x(k \mid k)+\Delta t V\cos(\theta(k \mid k)+\gamma) \\ y(k \mid k)+\Delta t V\sin(\theta(k \mid k)+\gamma) \\ \theta(k \mid k)+\frac{V}{B}\Delta t\sin(\gamma)\end{bmatrix}+\mathbf{v}(k)\end{aligned}$$

where $\mathbf{v}(k)$ is the motion noise (assumed to be Gaussian with zero mean).

The predicted variance $\mathbf{P}(k+1 \mid k)$ is calculated according to:

$$\mathbf{P}(k+1 \mid k)=\nabla\mathbf{f}_{|x}\mathbf{P}(k \mid k)\nabla\mathbf{f}_{|x}^{T}+\nabla\mathbf{f}_{|u}\mathbf{Q}\nabla\mathbf{f}_{|u}^{T} \qquad (4)$$

where $\nabla\mathbf{f}_{|x}$ is the Jacobian of the state transition equation with respect to $[x,y,\theta]^T$

$$\nabla\mathbf{f}_{|x}=\begin{bmatrix}1 & 0 & -\Delta t V\sin(\theta(k)+\gamma)\\ 0 & 1 & \Delta t V\cos(\theta(k)+\gamma)\\ 0 & 0 & 1\end{bmatrix} \qquad (5)$$

and $\mathbf{Q}$, the expected value of the motion noise, $E[\mathbf{v}\mathbf{v}^T]$ is defined as follows

$$\mathbf{Q}=\begin{bmatrix}\sigma_V^2 & 0\\ 0 & \sigma_\gamma^2\end{bmatrix} \qquad (6)$$

and the Jacobian $\nabla\mathbf{f}_{|u}$ is the state transition equation differentiated with respect to the input $\mathbf{u}(k)$:

$$\nabla\mathbf{f}_{|u}=\frac{d\mathbf{x}}{d\mathbf{u}}=\begin{bmatrix}\Delta t\cos(\theta+\gamma) & -\Delta t V\sin(\theta+\gamma)\\ \Delta t\sin(\theta+\gamma) & \Delta t V\cos(\theta+\gamma)\\ \Delta t\frac{1}{B}\sin(\gamma) & \Delta t\frac{V}{B}\cos(\gamma)\end{bmatrix} \qquad (7)$$

From Equation 7 it can be seen that $\nabla\mathbf{f}_{|u}$ is a rotation component which translates the motion noise $\mathbf{Q}$ into *skid* and *slide* error components [8]. These are applied to the vehicle in the direction corresponding to γ and its perpendicular.

Update

When an observation is validated against a feature held in the map, the AGV filter is updated with the observation according to the standard EKF update equations. The calculation of the observation matrix $\mathbf{h}$ however, depends upon which type of target is validated.

For line targets, the observation prediction is a function of the predicted state and a mapped line feature $\mathbf{p}=[p_R,p_\theta]^T$:

$$\mathbf{z}(k)=\mathbf{h}(\mathbf{p},\mathbf{x}(k))+\mathbf{v}(k)=\begin{bmatrix}p_R-x(k)\cos(p_\theta)-y(k)\sin(p_\theta)\\ p_\theta-\theta(k)\end{bmatrix} \qquad (8)$$

Hence the Jacobian of the predicted observation of a line feature (with respect to the AGV state):

$$\nabla\mathbf{h}_{|x}=\begin{bmatrix}-\cos(p_\theta) & -\sin(p_\theta) & 0\\ 0 & 0 & -1\end{bmatrix} \qquad (9)$$

For corners and edges, the predicted observation is a function of the predicted state and a mapped corner or edge feature $\mathbf{p}=[p_x,p_y]^T$:

$$\mathbf{z}(k)=\mathbf{h}(\mathbf{p},\mathbf{x}(k))+\mathbf{v}(k)=\begin{bmatrix}\sqrt{(p_x-x)^2+(p_y-y)^2}\\ \arctan(\frac{(p_y-y)}{(p_x-x)})-\theta\end{bmatrix} \qquad (10)$$

Therefore its Jacobian is

$$\nabla\mathbf{h}_{|x}=\begin{bmatrix}-\frac{(p_x-x)}{d} & -\frac{(p_y-y)}{d} & 0\\ \frac{(p_y-y)}{d^2} & \frac{-(p_x-x)}{d^2} & -1\end{bmatrix} \qquad (11)$$

where d is the upper element of the matrix in Equation 10.

These are used to calculate the innovation covariance $\mathbf{S}(k+1)$ of the AGV filter in the following equation.

$$\mathbf{S}(k+1)=\nabla\mathbf{h}_{|x}\mathbf{P}(k+1 \mid k)\nabla\mathbf{h}_{|x}^{T}+\mathbf{R}+\mathbf{S}_{Target}(k+1) \qquad (12)$$

where $\mathbf{R}$ is the noise associated with the observation.

The target's predicted variance is accounted for by incorporating the target innovation covariance $\mathbf{S}_{Target}(k+1)$. In order to include this though, it must be calculated within the vehicle filtering process and passed to the validated target ready to be incorporated into it's update procedure. Again, the manner in which the target variance is calculated, depends upon which type of target is being matched to:

For line targets, the variance is calculated using the Jacobian of the observation prediction Equation 8 (differentiated with respect to the target's state $\mathbf{p}=[p_R,p_\theta]^T$)

thus:

$$\nabla \mathbf{h}_{|\mathbf{p}} = \begin{bmatrix} 1 & x(k)\sin(p_\theta) - y(k)\cos(p_\theta) \\ 0 & 1 \end{bmatrix} \quad (13)$$

but for corners and edges, the Jacobian of the edge and corner observation prediction (Equation 10) is differentiated with respect to $\mathbf{p} = [p_x, p_y]^T$:

$$\nabla \mathbf{h}_{|\mathbf{p}} = \begin{bmatrix} \frac{(P_x - x)}{d} & \frac{(P_y - y)}{d} \\ \frac{-(P_y - y)}{d^2} & \frac{1}{d^2} \end{bmatrix} \quad (14)$$

Hence the innovation covariance of the validated target is calculated using:

$$\mathbf{S}_{Target}(k+1) = \nabla \mathbf{h}_{|\mathbf{p}} \mathbf{P}_{Target}(k+1 \mid k) \nabla \mathbf{h}_{|\mathbf{p}}^T \quad (15)$$

The update of the vehicle filter then continues according to the standard EKF procedure [1].

2.4 Validation

Validation of observations with mapped features is achieved in a two stage process: if an observation passes a preliminary (rapid) check against a target, then a nearest neighbour test is applied (using the Mahalanobis distance measure).

Preliminary Match

Observations are initially validated by simply comparing the predicted target state to each mapped target of a similar type. For line targets, the predicted state is calculated by rearranging Equation 8 to produce

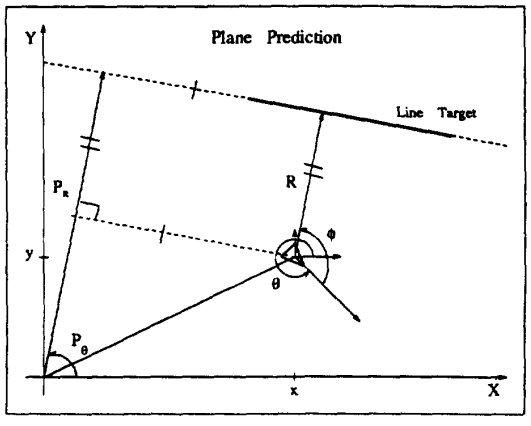

Figure 3: Line Target Preliminary Validation

$$\hat{\mathbf{p}} = \begin{bmatrix} \hat{p}_R \\ \hat{p}_\theta \end{bmatrix} = \begin{bmatrix} R + x(k+1 \mid k)\cos(P_\theta) + y(k+1 \mid k)\sin(P_\theta) \\ \theta(k+1 \mid k) + \phi \end{bmatrix} \quad (16)$$

Predicted corner and edge states are calculated by rearranging Equation 10:

$$\hat{\mathbf{p}} = \begin{bmatrix} \hat{p}_x \\ \hat{p}_y \end{bmatrix} = \begin{bmatrix} x(k+1 \mid k) + R\cos(\theta(k+1 \mid k) + \phi) \\ y(k+1 \mid k) + R\sin(\theta(k+1 \mid k) + \phi) \end{bmatrix} \quad (17)$$

Any feature found to lie within error bounds (arbitrarily set) of the actual observation is then passed to the nearest neighbour validation procedure.

Validation Gate

The target that is ultimately matched to the observation is chosen by means of a *nearest neighbour* technique. A standard validation gate (Equation 18) is used to obtain the association which minimises the Mahalanobis distance.

$$\mathbf{D}(k+1) = \nu^T(k+1)\mathbf{S}^{-1}(k+1)\nu(k+1) \quad (18)$$

where $\nu(k+1)$ is the *innovation*

$$\nu(k+1) = \mathbf{z}(k+1) - \mathbf{h}_x(\mathbf{x}(k+1 \mid k))$$

and $\mathbf{S}(k+1)$ is the innovation covariance:

$$\mathbf{S}(k+1) = \nabla \mathbf{h}_{|x} \mathbf{P}(k+1 \mid k) \nabla \mathbf{h}_{|x}^T + \mathbf{R} + \mathbf{P}_{Target}(k+1 \mid k) \quad (19)$$

where

$$\mathbf{P}_{Target}(k+1 \mid k) = \nabla \mathbf{f}_{Target} \mathbf{P}_p(k \mid k) \nabla \mathbf{F}_{Target}^T \quad (20)$$

Equation 20 does not include a noise component as the targets are assumed to be stationary and therefore no change in position is expected.

False alarms are minimised through the higher accuracy of infra red and the rapidity of update (hence odometric errors are minimised). Any errors that do occur have a brief and minor effect upon the system as the density of correct validations causes the system to rapidly disregard any erroneous decisions made through false alarms or erroneous associations.

2.5 Map Construction

The map held in the system is constructed from a number of *target* filters. Each filter describes a feature and is updated upon each occasion it is observed. The map is constantly maintained by several processes: *end-pointing* assigns start and end points to line segments, *target prediction* provides a means of determining when targets are no longer valid and *target filtering* is responsible for updating target state and initialising new features.

2.5.1 Target Filtering

As has been seen, feature filters can be subdivided into two categories; corner and edge filters and line filters. These only differ in their observation matrices and that which their state matrices represent. On the occasions that observations are not validated against any features held within the map, a new feature is initialised and installed within the map. This feature is assigned a probability (arbitrarily set, but accounting for the likelihood of correct observations from infra-red scanning) which will be increased upon each re-observation. The target state is initialised using Equation 16 and Equation 17 for lines

and corners/edges respectively. As the features are assumed to be stationary, the transition is assumed to be constant hence the Jacobian $\nabla \mathbf{f}_{Target}$ is an identity matrix.

Upon a feature being validated against an observation, it is updated with the information obtained with the new information. From Section 2.3 it can be seen that much of the filtering process for the target filter has been completed by the validation and AGV filtering processes. In fact all that is required in the target filtering process is to calculate the predicted observation according to Equation 8 for line features and Equation 10 for corners and edges. The update of the targets' position then continues according to the standard EKF procedure.

2.5.2 End-Point Fitting

For the line targets in the map to be fully defined, they must be assigned start and end points. Whenever a new target is found, the entire map is scanned for end-point assignment according to an algorithm similar to that developed by Duda and Hart [7].

2.5.3 Target Prediction

In order to identify the occasions when targets held within the map have been removed from the environment, a prediction is made as to what targets should be observed throughout the AGV's trajectory. This is achieved by applying a ray trace to the mapped targets to obtain a list of targets that are visible. The techniques employed in target prediction are designed to maximise speed of computation. Also, the prediction occurs at each observation thereby distributing the computation overhead over the entire scan.

Line Targets

Each line target within the map is tested for ray trace interception in order to identify both line targets (the ray striking at 90° to the line) and occlusions of other targets.

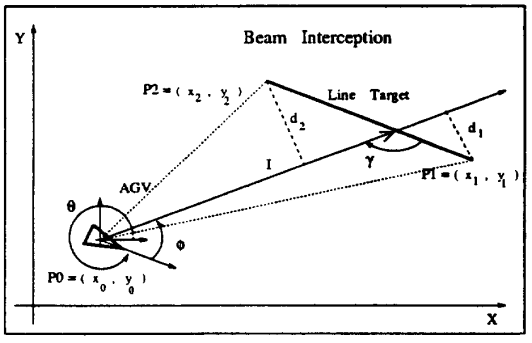

Figure 4: Ray Trace Interception

By vector geometry, if d_1 and d_2 differ in sign an interception is found i.e if:

$$d_1 d_2 \leq 0$$

where

$$d_1 = (-x_1 \sin(\theta + \phi) + y_1 \cos(\theta + \phi) + r) \quad (21)$$

$$d_2 = (-x_2 \sin(\theta + \phi) + y_2 \cos(\theta + \phi) + r) \quad (22)$$

and

$$r = x_0 \sin(\theta + \phi) - y_0 \cos(\theta + \phi)$$

The distance to an interception is found by calculating the vector I:

$$I = \overline{X0X2} + \alpha \overline{X2X1}$$

i.e.

$$I = \begin{bmatrix} x_2 - x_0 \\ y_2 - y_0 \end{bmatrix} + \begin{bmatrix} x_1 - x_2 \\ y_1 - y_2 \end{bmatrix} \cdot \frac{d_2}{d_2 - d_1} = \begin{bmatrix} x' \\ y' \end{bmatrix} \quad (23)$$

The angle to the interception is found using the following geometric relationship

$$\cos(\gamma) = \frac{\begin{bmatrix} x' \\ y' \end{bmatrix} \cdot \begin{bmatrix} x_1 - x_2 \\ y_1 - y_2 \end{bmatrix}}{\left\| \begin{bmatrix} x' \\ y' \end{bmatrix} \right\| \cdot \left\| \begin{bmatrix} x_1 - x_2 \\ y_1 - y_2 \end{bmatrix} \right\|} \quad (24)$$

If the angle is 90° (± error margin) it is stored as a target (so being there is no target occluding it).

Point Targets

A corner or edge is detected as a potential target by the following geometric test: If the feature lies on the left of

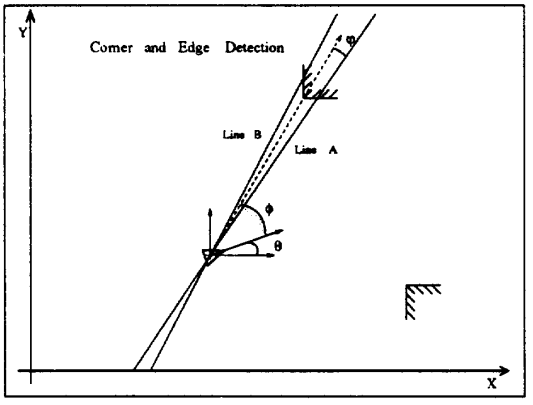

Figure 5: Point Target Interceptions

Line B and on the right of Line A, an intercept is recorded.

$$((p_y - y)\cos(\lambda) - (p_x - x)\sin(\lambda)) \times$$
$$((p_y - y)\cos(\kappa) - (p_x - x)\sin(\kappa)) < 0 \quad (25)$$

where $\lambda = (\theta + \phi + \psi)$ and $\kappa = (\theta + \phi - \psi)$.

Upon an intercept, its distance is calculated and compared to any existing interceptions. The closest target is stored as the predicted feature.

Should a predicted feature fail to correspond with an observation, the feature's likelihood probability is reduced. Subsequent failures to observe the target will result in its probability falling below a preset threshold, at which point it will be removed from the map.

3 Experimental Results

From the experiments we carried out, we show that the technique developed in Section 2 does indeed achieve simultaneous localisation and map construction.

A comprehensive series of tests were carried out on simulated data (which was subject to additive white noise). The AGV was modelled as having a constant velocity (of 50cm per second). It following a trajectory around a hand measured model of the AGV laboratory (Figure 6 at Oxford University. The data gathered from the tests included the map state and the innovation sequence from the AGV filter. The map state (shown in Figures 9 to 20) shows graphically both the AGV trajectory and the feature states held within the map (though line segments are only displayed if they have both end and start points assigned).

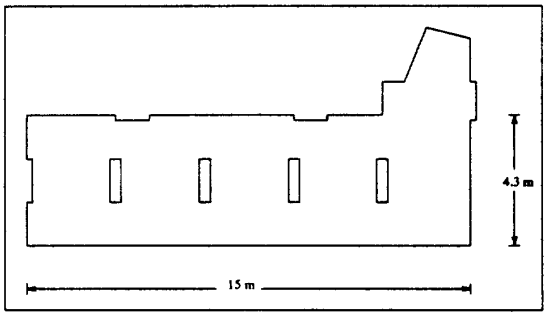

Figure 6: Operational Environment

Initially, the vehicle remains stationary to enable a complete first scan to be made. Figure 9 shows the map state after the complete scan. Although only one wall is displayed, two are held within the map data structure. As the end-points of one of the walls have not yet been found, the wall section is not displayed. Figure 10 shows that the origin has been located at the initial location of the vehicle. Figures 11 and 12 show the vehicle moving through the laboratory and observing more features which are then added to the map. Figure 13 and 14 illustrate the operation of the end-point fitting process. Once a line is found (corresponding to the base of the pillar, the map is scanned for corner and edge features that lie on the line. The two that are found (in opposing directions from the point the line was defined) are assigned. As subsequent observations are made, a new edge feature is discovered. As this lies upon a line feature in the map and is found to be closer to the observed point on the line (compared to the previous end-point), it replaces the old feature as the line's end-point.

Figures 15 and 16 provide a further example of end-point fitting at a later stage of operation. Figures 17 to 20 show the manner in which the environment model continues to be built over time. The similarity with the hand measured model in Figure 6 can now be clearly seen. It is also important to note that the AGV has adhered to the simulated trajectory. Throughout the entire testing procedure, the computation of the necessary information was achieved within the limits required for continuous operation of the AGV.

The AGV localisation filtering was verified by applying a whiteness test to the innovation sequence of a test consisting of 2200 updates. The innovation sequence consists of *range* and *bearing* innovations. Each was autocorrelated to produce the sequences shown in Figures 7 and 8. As can be seen, 95% of both sequences lie within the 2σ bounds, indicating that they represent white noise. This, together with the trajectory paths of Figures 9 to 20, show that the localisation filter is performing according to theory.

4 Conclusions

We have shown that real time localisation and map construction can be achieved using a multi-filter navigation system. By use of the higher data rate and quality of information provided by infra-red sensing, a robust system may be achieved without computationally expensive processes such as multi hypothesis frameworks.

Upon the replication of the prototype infra-red scanner, we will implement the navigation system on the Oxford Navigator robot. Further modifications to the system will include a regionalisation approach which will isolate regions of the constructed map appropriate to the AGV, thus allowing continuous real time operation regardless of the increases in size of the environmental model.

[1] Y. Bar-Shalom and T. E. Fortmann *Tracking and Data Association* Academic Press, 1988

[2] A. Zilensky *Environment Mapping with a Mobile Robot Using Sonar* AI Proc. Australian Joint Artificial Intelligence Conference, Nov. 1988.

[3] J. J. Leonard and I. J. Cox *Modelling a Dynamic Environment Using a Bayesian Multiple Hypothesis Approach* AAAI Journal of Artificial Intelligence 1993

[4] N. Ayache and O. D. Faugeras *Maintaining Representations of the Environmnent of a Mobile Robot* IEEE Transactions on Robotics and Automation 1989

[5] P. Moutarlier and R. Chatila *Stochastic Multisensory Data Fusion for Mobile Robot Location and Environment Modelling* Proc. 5^{th} Int. Symposium on Robotics Research, Tokyo 1989.

[6] M. Brownlow *Mobile Robot Localisation Using A Time-Of-Flight Optical RangeFinder* D.Phil. Thesis, University of Oxford. 1993.

[7] R. O. Duda and P. E. Hart *Pattern Classification and Scene Analysis* John Wiley ans Sons 1973

[8] H. F. Durrant-Whyte *Uncertain Geometry in Robotics* IEEE J. Robotics and Automation, 4(1):23-31, 1988.

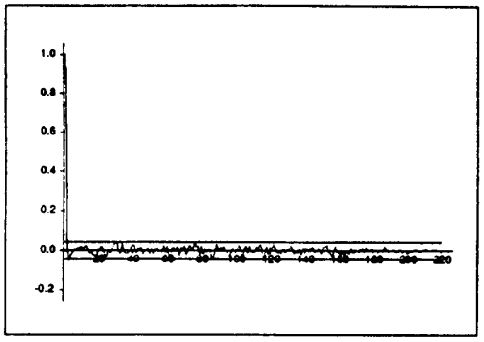

Figure 7: R Innovation Autocorrelation

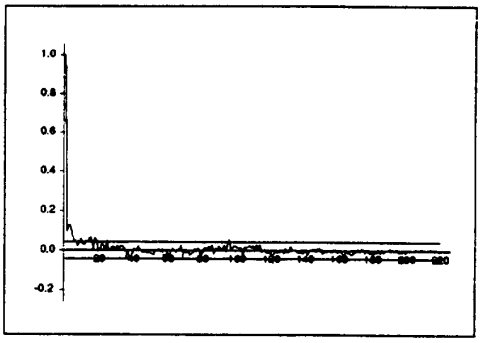

Figure 8: ϕ Innovation Autocorrelation

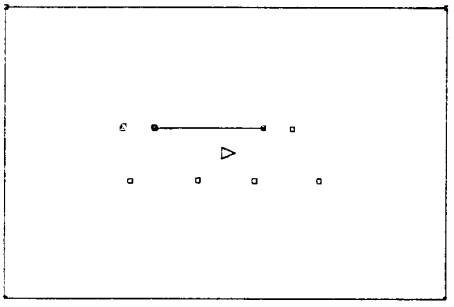

Figure 9: Initial Scan

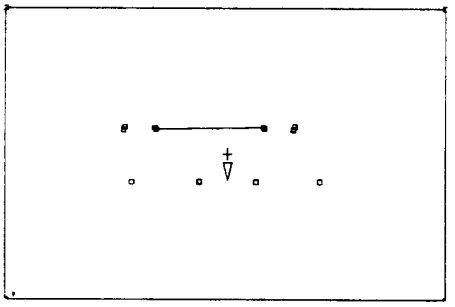

Figure 10: Origin Set

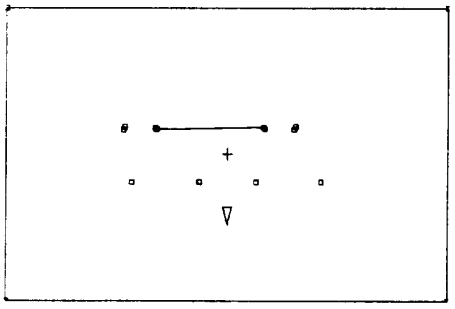

Figure 11: Wall Detection

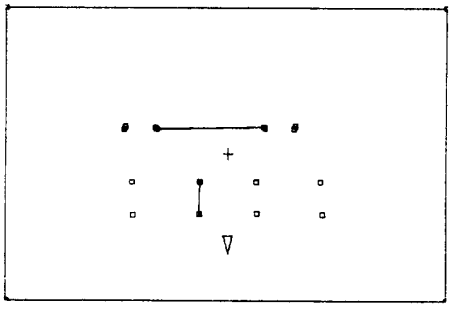

Figure 12: Wall Display

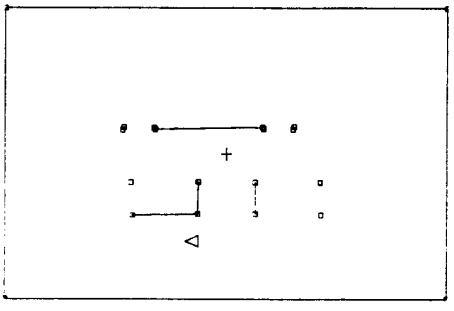

Figure 13: End-Point Fitting

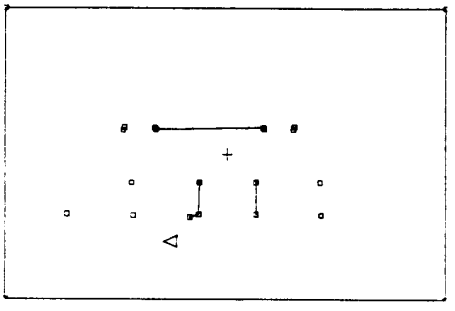

Figure 14: End-Point Update

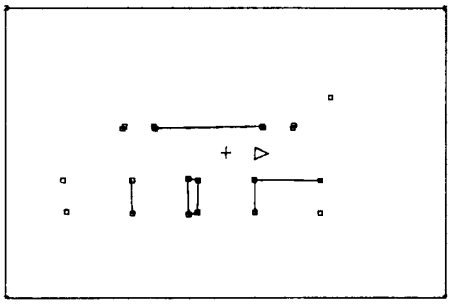

Figure 15: End-Point Fitting 2

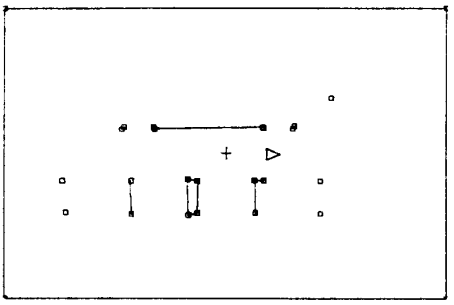

Figure 16: End-Point Update 2

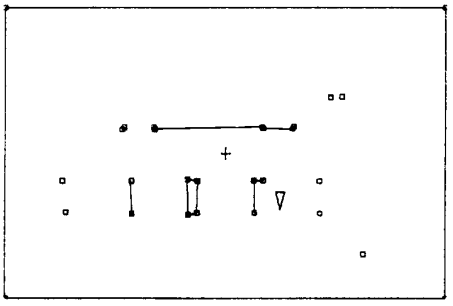

Figure 17: Map Construction

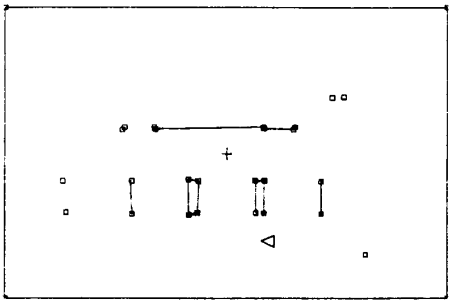

Figure 18: Map Construction

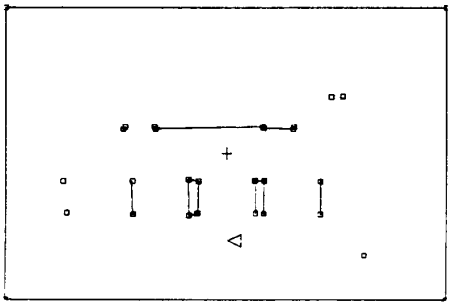

Figure 19: Map Construction

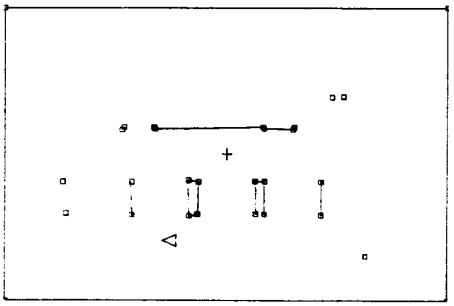

Figure 20: Map Construction