AUV NAVIGATION USING A FORWARD LOOKING SONAR

Yvan Petillot, Ioseba Tena Ruiz, David Lane
Ocean Systems Laboratory
Department of Computing and Electrical Engineering
Heriot-Watt University, Edinburgh, EH14 4AS
Scotland, UK
htt://www.cee.hw.ac.uk/oceans/
email:(ceeyrp,tena,dml)@cee.hw.ac.uk

1 Introduction

Many commercial systems exist for finding the *absolute* position of a submersible vehicle, these include *acoustic positioning* systems; *acoustic super short*, *short* and *long baseline navigation*. The two former methods are only suitable for local area positioning, over relatively short distance, less than one nautical mile. Long baseline provides positioning to within one to ten meters and covers an area of up to seven nautical miles. The need for the submersible to be within an envelope covered by the above mentioned systems makes them unsuitable for exploration of new terrain using AUVs. Standard practice has been to achieve positioning by means of integrating the inertial navigation readings with an absolute positioning measurement provided by an acoustic positioning system which communicates with a surface vessel. The use of such system involves a considerable cost. The development of an inexpensive and yet reliable positioning system is therefore desirable, to achieve this goal one must break the existing link between surface vessels and UVs. We propose a system that builds a map of features in the environment, localises it-self with respect to these features, and avoids obstacles. The system uses standard *dead-reckoning* sensors and a mechanical forward looking sonar.

Our previous research [1, 2, 3, 4, 5] presented different approaches for tracking returns from a forward looking sonar, for the purpose of classification and obstacle avoidance. Our current research tracks the position of objects in the environment and that of the vehicle itself, with respect to the world reference frame, to build an stochastic map [6] and allow the vehicle to navigate in this map whilst avoiding obstacles. The returns from the sonar are used to build images, standard image processing techniques are used to segment these images and extract the position w.r.t. to the vehicle of possible obstacles. These obstacles are used to create a local map of the environment that will be used by the obstacle avoidance algorithm to plan a trajectory. They will also be used to find an *absolute position* estimate of the vehicle which will be integrated with the vehicle's dead-reckoning sensors.

2 Principles

The following section will describe the theoretical principles used by the underlying algorithms. The potential field approach used for avoiding obstacles has been widely used in the field of robotics for obstacle avoidance and path planning [7, 8, 9, 10] and will be briefly outlined in section 2.1. The following section 2.2 describes the theory behind simultaneous localisation and mapping.

2.1 Artificial potential field

Obstacle avoidance can be broadly divided into two classes of techniques: global and local. global methodologies rely on the description of the obstacles in the configuration space of the vehicle [7] while local techniques generally uses *artificial potential field* or related techniques [11]. Global techniques reduce the problem of planning the motion for a manipulator or vehicle to a problem of planning the motion of points representing the configuration of the vehicle/manipulator in the proper configuration space (which takes into account all the constraints of the vehicle/manipulator). Therefore such algorithm have two main steps. The first step consists in transforming the available description of the obstacles in the real world (Cartesian coordinates) to a suitable representation in the configuration space. The second step consists in determining the free space of the configuration space and to search for the optimal path (with respect to a given criterion, i.e. time to goal, energy minimisation,...). These algorithms produce a collision free path if one exists. However, they are computationally intensive and require a priori knowledge of the environment.

Local techniques rely on the use of artificial potential functions and provide a fast and viable option to global techniques, especially for real-time applications and for applications where the environment can change rapidly or is only partially known. In local techniques, the obstacles are represented by a repulsive potential while the goal point is represented by an attractive potential. The super-imposition of the two potential provides the desired workspace energy topology. For any starting point in the modeled workspace it is possible to follow a path decreasing the potential and reaching the goal. However, the workspace created can present local minima and a robust algorithm should be able to escape local minima while still avoiding obstacles. In order to do that the navigation function as described in [8] is used.

2.1.1 The potential model

The potential model is generated in two stages. The first stage consists in defining the attractive well which draws the robot towards the goal point. The next stage is the definition of the repulsive field, which pushes the robot away from the obstacles. The potential field is represented as a discrete approximation of the analog field. This approximation is not a limitation as the data representing the objects is in general discrete and the discretisation process can be tailored to match the resolution needed by a given application.

The attractive field should have only one minimum point, the goal. A standard way of defining it is to use a parabolic well defined as:

$$U_{att}(q) = \frac{1}{2}\eta(q - q_{goal})^2 \tag{1}$$

where q is a point of the workspace and η a scaling factor.

The force applied to the robot at any point q of the space driving it towards the goal is therefore:

$$F_{att}(q) = -\eta(q - q_{goal}) \tag{2}$$

This form of potential has some useful properties as the force derived from it converges linearly to zero. Hence, the robot can be made to decelerate smoothly as it approaches the goal point.

The repulsive potential is defined as:

$$U_{rep}(q) = \begin{cases} \frac{1}{2}\mu\left(\frac{1}{\rho(q)} - \frac{1}{\rho(q_0)}\right)^2 & \text{if } \rho(q) \leq \rho_0 \\ 0 & \text{otherwise} \end{cases} \tag{3}$$

where μ is a positive scaling factor and $\rho(q)$ is the smallest distance between the configuration q and any obstacle. ρ_0 is a positive constant defining the *distance of influence* of the obstacle. It can be seen that the force exerted by the obstacle is zero when the robot configuration lies outside the distance of influence and tends to infinity as the robot approaches the obstacle preventing any collisions.

2.1.2 Planning algorithm

The classical technique uses standard gradient descent. Starting from the current robot configuration, the neighbour with the smallest potential amongst the ones having a smaller potential than the current configuration is selected. The process is iterated until the goal is reached or no neighbour can be selected. In the latter case, the current configuration is a local minimum.

In order to avoid the local minimum problem, we use a tree searching algorithm which is complete. It is guaranteed to find a path in the free space if one exists. Again it operates by examining the neighbours of the current configuration with the following assumptions:

- A path between adjacent configurations lies in the free space.
- The configuration space lies in a bounded rectangular area.

A tree T of possible paths is built with the starting configuration q_{init} as the root of the tree. At every iteration, the leaf of T with the lowest potential is selected. The neighbours of that configuration whose potential are less than a given threshold (high in general but less than inside any object) and which are not already in the tree are selected. They are installed in the tree as successors of the current leaf. The algorithm terminates when q_{goal} has been reached or any attainable configuration from q_{init} has been installed in the tree. If q_{goal} has been found, a path can be tracked back from q_{goal} to q_{init}. Effectively it is the same as following the gradient descent technique until a local minimum has been found, in which case the algorithm acts by filling the well of the local minimum until a saddle point is found.

2.2 Simultaneous localisation and mapping

The first consistent solution to the simultaneous localisation and mapping problem was put forward by Smith *et al* [6]. The research presented by Smith *et al* has been extensively used by other researches, leading to exciting new developments [12, 13, 14, 15]. In [16] a thorough explanation on the underlying fundamentals is given. The algorithm is an augmentation on the extended Kalman filter [17, 18]. The filter now holds the relevant states of the vehicle and that of the features used in the mapping and localisation process. The new state vector $\mathbf{x}(\cdot)$ assumes the following form,

$$\mathbf{x}(k) = \begin{bmatrix}\mathbf{x_v}(k) & \mathbf{x_1}(k) & \mathbf{x_2}(k) & \ldots & \mathbf{x_n}(k)\end{bmatrix}^T \tag{4}$$

where $\mathbf{x_v}(k)$ holds the state of the vehicle and $\mathbf{x_1}(k), \mathbf{x_2}(k), \ldots, \mathbf{x_n}(k)$ hold the states of the n features. The estimated error covariance (approximated mean-square error) for this system,

$$\mathbf{P}(k) = \begin{bmatrix} \mathbf{P_{v\,v}}(k) & \mathbf{P_{v\,1}}(k) & \mathbf{P_{v\,2}}(k) & \ldots & \mathbf{P_{v\,n}}(k) \\ \mathbf{P_{1\,v}}(k) & \mathbf{P_{1\,1}}(k) & \mathbf{P_{1\,2}}(k) & \ldots & \mathbf{P_{1\,n}}(k) \\ \mathbf{P_{2\,v}}(k) & \mathbf{P_{2\,1}}(k) & \mathbf{P_{2\,2}}(k) & \ldots & \mathbf{P_{2\,n}}(k) \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ \mathbf{P_{n\,v}}(k) & \mathbf{P_{n\,1}}(k) & \mathbf{P_{n\,2}}(k) & \ldots & \mathbf{P_{n\,n}}(k) \end{bmatrix} \tag{5}$$

where the sub-matrices $\mathbf{P_{v\,v}}(k)$, $\mathbf{P_{v\,i}}(k)$ and $\mathbf{P_{i\,i}}(k)$ are the vehicle-to-vehicle, vehicle-to-feature and feature-to-feature covariances respectively.

The different steps followed by the algorithm are now summarised:

2.2.1 Propagate the vehicle state

Stochastic mapping assumes fixed features and the resulting state propagation will become,

$$\hat{\mathbf{x}}_\mathbf{v}(k) = \mathbf{f_v}[\hat{\mathbf{x}}_\mathbf{v}(k-1), \mathbf{u}(k), \mathbf{0}, k] \tag{6}$$

where $\hat{\mathbf{x}}_\mathbf{v}(\cdot)$ is the vehicle's state and $\mathbf{f_v}[\hat{\mathbf{x}}_\mathbf{v}(\cdot), \mathbf{u}(k), \mathbf{0}, k]$ is the vehicle's dynamic model. Its associated covariance, strictly speaking the approximated mean-square error as $\hat{\mathbf{x}}(k)$ is not the exact conditional mean, is obtained as follows,

$$\mathbf{P}(k) = \begin{bmatrix} \mathbf{f_{x_v}}\mathbf{P_{v\,v}}(k)\mathbf{f_{x_v}}^T & \mathbf{f_{x_v}}\mathbf{P_{v\,1}}(k) & \ldots & \mathbf{f_{x_v}}\mathbf{P_{v\,n}}(k) \\ \mathbf{P_{1\,v}}(k)\mathbf{f_{x_v}}^T & \mathbf{P_{1\,1}}(k) & \ldots & \mathbf{P_{1\,n}}(k) \\ \vdots & \vdots & \ddots & \vdots \\ \mathbf{P_{n\,v}}(k)\mathbf{f_{x_v}}^T & \mathbf{P_{n\,1}}(k) & \ldots & \mathbf{P_{n\,n}}(k) \end{bmatrix} + \begin{bmatrix} \mathbf{f_{w_v}}\mathbf{Q}(k)\mathbf{f_{w_v}}^T & \mathbf{0} & \ldots & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & \ldots & \mathbf{0} \\ \vdots & \vdots & \ddots & \vdots \\ \mathbf{0} & \mathbf{0} & \ldots & \mathbf{0} \end{bmatrix} \tag{7}$$

where $\mathbf{f_{x_V}}$ is the Jacobian of the vehicle model with respect to the vehicle state, used to linearise the state of the vehicle error $\tilde{\mathbf{x}}_\mathbf{v}(k-1)$, and $\mathbf{f_{w_v}}$ is the Jacobian of the vehicle model with respect to the process noise.

2.2.2 Predict position of features w.r.t. the vehicle frame

For a vehicle using a range sensor returning range and angle w.r.t. the vehicle frame the observation vector for a single feature will be $\mathbf{z_i}(k) = [r \;\; \theta]^T$, the full feature vector will be of the form,

$$\mathbf{z}(k) = \begin{bmatrix}\mathbf{z_1}(k) & \mathbf{z_2}(k) & \ldots & \mathbf{z_n}(k)\end{bmatrix}^T \tag{8}$$

The prediction for feature $\mathbf{x_i}(k)$ will be,

$$\hat{\mathbf{z}}_{\mathbf{i}}(k) = \begin{bmatrix}\sqrt{\bar{x}_i(k)^2 + \bar{y}_i(k)^2} \\ tan^{-1}(\bar{y}_i(k)/\bar{x}_i(k)) + \phi_v(k)\end{bmatrix} \tag{9}$$

where $\phi_v(\cdot)$ is the orientation of the vehicle w.r.t. the world frame and $\bar{x}_i(\cdot)$, $\bar{y}_i(\cdot)$ are respectively

$$\bar{x}_i(k) = x_i(k) - x_v(k) \tag{10}$$

$$\bar{y}_i(k) = y_i(k) - y_v(k) \tag{11}$$

The coordinates w.r.t. to the world frame of feature i and of the vehicle being $[x_i(k)\, y_i(k)]$ and $[x_v(k)\, y_v(k)]$ respectively.

2.2.3 Find features w.r.t. the vehicle frame

Not all features will be observed by the vehicle at each iteration. Those which are observed must be associated to the features in the stochastic map state vector $\mathbf{x}(k)$. This is known as data association. Data association has been a subject of major research. A good description of various methods can be found in [18]. The simplest method is the gated nearest neighbour approach. This method is performed in innovation space and takes into account the uncertainty in both the sensor and the stochastic map state. Each observed feature is associated to an already existing feature if the following criteria is satisfied,

$$\upsilon_{\mathbf{i}}{}^T \mathbf{S_i}{}^{-1} \upsilon_{\mathbf{i}} \leq \gamma \tag{12}$$

where the innovation $\upsilon_{\mathbf{i}}$ is defined as

$$\upsilon_{\mathbf{i}} = \hat{\mathbf{z}}_{\mathbf{i}}(k) - \mathbf{z_i}(k) \tag{13}$$

with innovation covariance matrix $\mathbf{S_i}$ and γ is the threshold parameter obtained from the χ^2 distribution.

If more than one observation meets this criteria, with respect to the same feature, than the one with the lowest value, i.e. the most probable will be chosen. Once all measurements have been associated the stochastic map can be corrected. Observations that are not associated to any objects will be added to the map as explained in section 2.2.5.

2.2.4 Correct the position of the vehicle and features w.r.t. the world frame

The new set of measurements is used to update the state of the stochastic map and its associated covariance. The algorithm maintains the correlations between the errors of the vehicle and all the features. The state is updated according to the following equation

$$\hat{\mathbf{x}}(k+1) = \hat{\mathbf{x}}(k) + \mathbf{K}(k)v(k) \tag{14}$$

where $\mathbf{K}(k)$ is found to be

$$\mathbf{K}(k) = \mathbf{P}(k)\mathbf{H}^T(k)\mathbf{S}^{-1}(k) \tag{15}$$

The innovation matrix, $\mathbf{S}(k)$, is defined as

$$\mathbf{S}(k) = \mathbf{H}(k)\mathbf{P}(k)\mathbf{H}^T(k) + \mathbf{R}(k) \tag{16}$$

where $\mathbf{H}(k)$ is a matrix that stacks the Jacobians of the observed features and $\mathbf{R}(k)$ is a stack matrix of the measurement error covariances. The stochastic map's covariance will be updated according to

$$\mathbf{P}(k+1) = \mathbf{P}(k) - \mathbf{K}(k)\mathbf{H}(k)\mathbf{P}(k) \tag{17}$$

2.2.5 Add new features to the map in the world frame

Observations that were not associated to an existing feature will be added to the stochastic map state and covariance. The new observation $\mathbf{z_{new}} = [r \;\; \theta]^T$ is estimated with respect to the vehicle's reference frame,

$$\mathbf{x_{n+1}}(k) = \begin{bmatrix} x_v(k) + r\cos(\phi+\theta) \\ y_v(k) + r\sin(\phi+\theta) \end{bmatrix} \tag{18}$$

The new map state and associated covariance will be

$$\mathbf{x}(k) \leftarrow \begin{bmatrix} \mathbf{x}(k) \\ \mathbf{x_{n+1}}(k) \end{bmatrix} \tag{19}$$

$$\begin{aligned} \mathbf{P_{n+1\,n+1}}(k) &= \mathbf{L_{x_v}}\mathbf{P_{v\,v}}(k)\mathbf{L_{x_v}}^T + \mathbf{L_{z_{new}}}\mathbf{R}(k)\mathbf{L_{z_{new}}}^T \\ \mathbf{P_{n+1\,v}}(k) &= \mathbf{P_{v\,n+1}}^T(k) = \mathbf{L_{x_v}}\mathbf{P_{v\,v}}(k) \end{aligned} \tag{20}$$

where $\mathbf{L_{x_v}}$ and $\mathbf{L_{z_{new}}}$ are the Jacobian of equation 18 with respect to the robot vehicle state $\hat{\mathbf{x}}_\mathbf{v}$ evaluated at $\hat{\mathbf{x}}_\mathbf{v}(k)$ and to the new observation $\mathbf{z_{new}}$ evaluated at $\mathbf{z_{new}}$.

3 Navigation

The system has been designed with RAUVER in mind. RAUVER is an ROV designed by the Ocean Systems Laboratory as a testbed for algorithm development. The aim is to develop a suitable algorithm which will than be easily integrated onto an existing AUV framework. The motivation for using a mechanically scanned sonar lies in its proven reliability, low cost and above all compact size and light weight. The Tritech DFS forward looking sonar presents itself as a versatile option and has been integrated into our framework. The obstacle avoidance algorithm will also be integrated in the ARAMIS skid. ARAMIS is a European Commission Directorate General XII funded project. This skid will be mounted onto ROMEO [19], supplied by CNR-IAN, and VICTOR 6000 [20], supplied by IFREMER.

The operation of the system will now be described. Section 3.1 presents an overview, and the following sections will look at each building block in closer detail.

3.1 Overview

The overview of the system can be seen Figure 1. The data from the sonar is segmented and the obstacles are fed into the stochastic map, the dead-reckoning sensors are used to improve the accuracy of the map. The obstacle avoidance algorithm is fed with a binary map where the regions that are occupied, and therefore must be avoided, are represented as ones and the empty regions are represented as zeros. The output from the obstacle avoidance algorithm is a path, this path is fed into the vehicle controller. The vehicle controller will in turn feed the stochastic map with the thrust values used at each iteration.

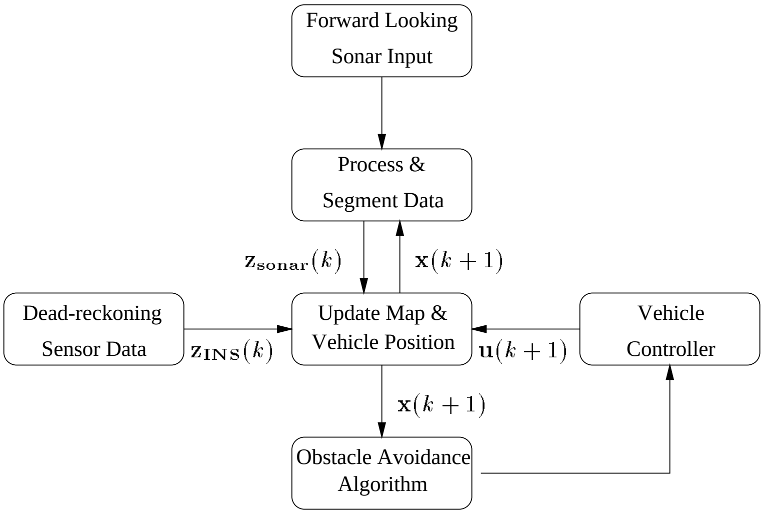
Forward Looking Sonar Input
Process & Segment Data
$\mathbf{z}_{\mathbf{sonar}}(k)$
$\mathbf{x}(k+1)$
Dead-reckoning Sensor Data
$\mathbf{z}_{\mathbf{INS}}(k)$
Update Map & Vehicle Position
$\mathbf{u}(k+1)$
Vehicle Controller
$\mathbf{x}(k+1)$
Obstacle Avoidance Algorithm

Figure 1: Overview of the System

3.2 Forward looking sonar input

The Tritech DFS has two possible operating frequencies, the chosen frequency for the algorithm is 325 kHz. Which is a good compromise between resolution and speed. Typical working ranges will be 25 and a sector of size 60^o will be scanned.

3.3 Data processing and segmentation

As the TRITECH sonar has a 30^o vertical beamwidth, in typical operation, 1 to 5 meters above the seabed, the seabed is in its field of view. The returns from the bottom should be removed to avoid high false alarm rates. In order to do that, the first significant returns are detected for each sonar beam and the bottom mean and variance is estimated. This estimate is continuously updated using a sliding window. If the parameters of the sonar are changed, the estimation is restarted from the first next image. A threshold is then derived from the mean and variance of the bottom, $Mean_{bottom} + 2 * Var_{bottom}$. The image is then segmented using this threshold. If an object is in the part of the image where the bottom estimation is performed (close to the object)?, its effect will be minimal because of the sliding window used in the bottom estimation. Once the image has been segmented the range and angle of the obstacles w.r.t. to the vehicle is extracted and fed to the mapping and localisation module. Knowledge of the obstacles predicted position can also be used to relax the image processing constraints, as was done in [4].

3.4 Dead-reckoning sensor data

The sensor suite will mainly consist of off-the-shelf products. Some of these have not been purchased at the time of writing. The sensors that have been purchased are,

- AOSI EZ-COMPASS-3: Tilt compensated magnetometer compass
- muRata Piezoelectric vibrating gyroscope: Single axis gyroscope

Other sensors that will be included are,

- Depth pressure sensor
- Echosounder
- Two Flowmeters
- Shatf Encoders for the thrusters

The data from these sensors will be integrated and used to update the vehicle state in the Stochastic map. The system will benefit from better sensor technology. Integration of further sensors is made easy under the Kalman filter structure.

3.5 Updating map and vehicle position

The stochastic map, as explained in section 2.2, holds the state of the vehicle and obstacles. The Laboratory is in the process of developing a dynamic model of RAUVER, this model will be integrated in the stochastic map. In the mean time a linear model is used. This model has been found to work well and can be used with any vehicle. The state of the vehicle takes the following form,

$$\mathbf{x_v}(k) = \begin{bmatrix} x & \dot{x} & \ddot{x} & y & \dot{y} & \ddot{y} & \phi & \dot{\phi} & \ddot{\phi} \end{bmatrix}^T \tag{21}$$

with the following dynamic model,

$$\mathbf{F_v}(k) = \begin{bmatrix} \mathbf{F_{v_x}}(k) & \mathbf{0} & \mathbf{0} \\ \mathbf{0} & \mathbf{F_{v_y}}(k) & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & \mathbf{F_{v_\phi}}(k) \end{bmatrix} \tag{22}$$

where

$$\mathbf{F_{v_x}}(k) = \mathbf{F_{v_y}}(k) = \mathbf{F_{v_\phi}}(k) \begin{bmatrix} 1 & T & \frac{1}{2}T^2 \\ 0 & 1 & T \\ 0 & 0 & 1 \end{bmatrix} \tag{23}$$

Each obstacle has a state vector with states for its position and size,

$$\mathbf{x_o}(k) = \begin{bmatrix} x & y & s \end{bmatrix}^T \tag{24}$$

The updating of the state is done according to the procedure outlined in section 2.2.

3.6 Obstacle avoidance algorithm

All obstacles are stored in a global map referenced to a known starting position of the vehicle or any georeferenced position. The first step of the obstacle avoidance algorithm is to update the map of obstacles based on new data coming from the sonar. Once the data has been segmented as described above, each beam position is calculated using:

- The time when the beam was collected.
- The stored timestamped positions of the vehicle during data collection.

The two closest positions of the vehicle at the time the beam was collected are computed and interpolated to give the best estimate of the real position when the beam was emitted. The new data is then added to the global map. The new global map can now be used for path planning. If the positioning system of the vehicle is not accurate, the relative position between obstacles detected during different sonar scans will drift in time and eventually become meaningless. It is an option of the system to clean the map of objects detected in previous scans. It is even possible to use only the last scan when no positioning system is available.

The global map consist of a collection of points belonging to obstacles in the real world reference frame. The workspace is then discretised using a given cell resolution. In each cell, the number of obstacles points is counted and if this number is above a threshold (corresponding at a number of points per square meter), the cell is considered as a obstacle and creates a repulsive potential around it as explained in section 2.1.

The attractive map is added to the repulsive map and the algorithm described in section 2.1 is applied to derive the path.

3.7 Vehicle controller

Once a path has been derived it is fed into the vehicle controller which in turn will output the thruster values required to follow the desired path. These values, along with RAUVER's dynamic model, will be used in the future by the stochastic mapping algorithm.

4 Results

The obstacle avoidance system has been tested in the Oceans Systems Laboratory's tank. Figure 2 shows the true dimensions of the tank. The obstacle avoidance algorithm starts operation, Figure 3, and builds a local map of the environment. The image on the right shows a grid where each element is a meter square in size. The sonar is in the middle of the image. The goal is two meters to the left and three meters forwards. The image in the left are the actual returns from the sonar. The map is updated and the obstacles are found as the sonar moves, Figure 4. Figure 5 shows the full map of the tank and the path which will be fed to the vehicle controller.

Figure 6 shows the output from the stochastic mapping. The sequence of sonar scans, displayed on the top of the image, was generated using a forward-looking sonar simulator [21] generated in Heriot-Watt as an extension to previous work which created a realistic sidescan simulator [22]. The bottom image shows the map as it is being created. The sector of seabed observed by the sonar, the positions of the features and the vehicle and the certainty (covariance) on these results are all displayed. Figure 7 shows the outcome of the algorithm for an eighty meter mission. The average position error for the vehicle in x is of 0.16 meters, equivalent to 0.2% of the distance traveled, and the average position error for the vehicle in y is of 0.08 meters, equivalent to 0.1% of the distance traveled. To obtain these results the sonar was aided by a compass simulation of the AOSI EZ-COMPASS-3.

5 Conclusions

The potential of the navigation algorithm has been demonstrated by the results. The obstacle avoidance algorithm is a promising tool that will allow for a degree of automatisation even in controlled ROV missions. The stochastic map offers accurate position fixes of both the obstacles and the vehicle itself. These results should be expected to improve as further sensors and a vehicle model are integrated in the system. A limitation of the algorithm which needs to be addressed is the fact that the maps grow without bounds. Future work will consider the structure of these maps in closer detail.

6 Acknowledgments

This research is being supported by the European community under the MAST project CT97-0083 (ARAMIS).

References

[1] D. M. Lane, M. J. Chantler, and D. Dai. Robust tracking of multiple objects in sector-scan sonar image sequences using optical flow motion estimation. *IEEE Journal of Oceanic Engineering*, 23(1):31–46, January 1998.

[2] D. M. Lane, M. Chantler, and D. Y. Dai. Segmentation and tracking of multiple objects in multi-beam forward looking sonar image sequences. In *Proceedings of the 1998 Oceanology International Conference*, pages 51–62, Brighton, UK, March 1998.

[3] D. M. Lane, M. Chantler, D. Y. Dai, and I. Tena Ruiz. Tracking and classification of multiple objects in multi-beam sector scan sonar image sequences. In *Proceedings of the 1998 International Symposium on Underwater Technology*, pages 269–273, Tokyo, Japan, April 1998.

[4] Y. Petillot, I. Tena Ruiz, D. M. Lane, Y. Wang, E. Trucco, and N. Pican. Underwater vehicle path planning using a multi-beam forward looking sonar. In *OCEANS'98 IEEE Proceedings*, pages 1194–1199, Nice, France, September 1998.

[5] I. Tena Ruiz, Y. Petillot, D. M. Lane, and J. M. Bell. Tracking objects in underwater multibeam sonar images. In *IEE Colloquium on Motion Analysis and Tracking*, pages 11/1–11/7, London, UK, May 1999.

[6] R. Smith, M. Self, and P. Cheeseman. Estimating uncertain spatial relationships in robotics. In I. Cox and G. Wilfong, editors, *Autonomous Robot Vehicles.* Springer-Verlag, 1990.

[7] O. Khatib. *Commande dynamique dans l'espace opérationnel des robots manipulateurs en présence d'obstacle.* in French, Ecole Nationale Supérieure de l'Aéronautique et de l'Espace, Toulouse, 1980.

[8] J. C. Latombe. *Robot motion planning.* Kluwer Academic Publishers, Boston, 1991.

[9] O. Khatib. Real-time obstacle avoidance for manipulators and mobile robots. *International Journal of Robotic Research*, 5:90–98, 1986.

[10] Y. Zhang and K. P. Valavanis. A 3-D potential panel method for robot motion planning. *Robotica*, 15:421–434, 1997.

[11] Y. Wang and D. M. Lane. Subsea vehicle path planning using nonlinear programming and constructive solid geometry. *IEE Proceedings on Control Theory Applications*, 144:143–152, 1997.

[12] H. J. S. Feder. *Simultaneous Stochastic Mapping and Localization.* PhD thesis, Massachusetts Institute of Technology, 1999.

[13] R. N. Carpenter. Concurrent mapping and localization with FLS. In *1998 Workshop on Autonomous Underwater Vehicles*, pages 133–148, Cambridge, Massachusetts, USA, August 1998.

[14] S. Clark and H. Durrant-Whyte. Autonomous land vehicle navigation using millimeter wave radar. In *Proceedings of the 1998 IEEE International Conference on Robotics and Automation*, volume 4, pages 3697–3702, Leuven, Belgium, May 1998. IEEE.

[15] J. K. Uhlmann. *Dynamic Map Building and Localization: New Theoretical Foundations.* PhD thesis, University of Oxford, 1995.

[16] M. Csorba. *Simultaneous Localisation and Map Building.* PhD thesis, University of Oxford, 1997.

[17] P. S. Maybeck. *Stochaistic models, estimation, and control. Volume 2*, volume 141 of *Mathematics in Science and Engineering.* Academic Press, 1982.

[18] Y. Bar-Shalom and T. E. Fortmann. *Tracking and Data Association*, volume 179 of *Mathematics in Science and Engineering.* Academic Press, 1988.

[19] R. Bono, Ga. Bruzzone, Gi. Bruzzone, M. Caccia, E. Spirandelli, and G. Veruggio. ROMEO goes to Anctartica. In *OCEANS'98 IEEE Proceedings*, pages 1568–1572, Nica, France, September 1998.

[20] JF. Cardiou, S. Coudray, P. Léon, and M. Perrier. Control architecture of a new deep scientific ROV : VICTOR 6000. In *OCEANS'98 IEEE Proceedings*, pages 492–497, Nice, France, September 1998.

[21] J. M. Bell. Synthesising sonar images and resulting textures using physical based models. In *Eurel meeting on radar and sonar signal processing*, pages 20/1–20/2, Peebles, July 1998.

[22] J. M. Bell. and L. M. Linnett. Simulation and analysis of synthetic sidescan sonar images. *IEE Proceedings - radar, sonar and navigation*, 144(4):219–226, August 1997.

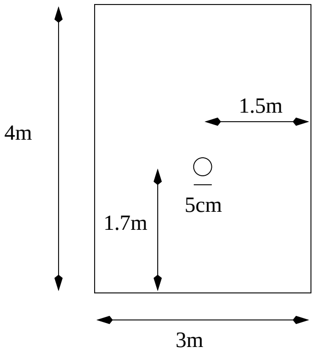

Figure 2: Tank used for obstacle avoidance experiments

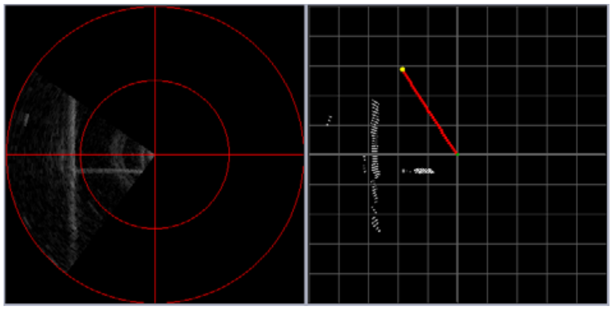

Figure 3: First iteration of the obstacle avoidance algorithm

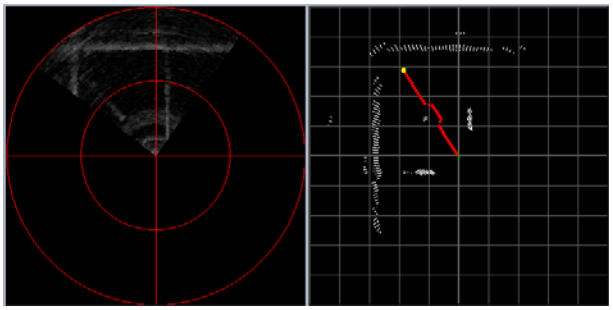

Figure 4: Obstacle has been found by the obstacle avoidance algorithm

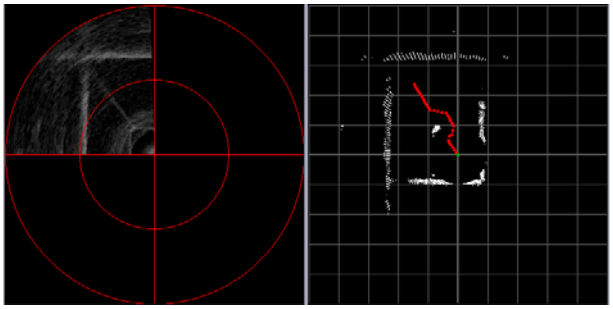

Figure 5: Full map of the tank

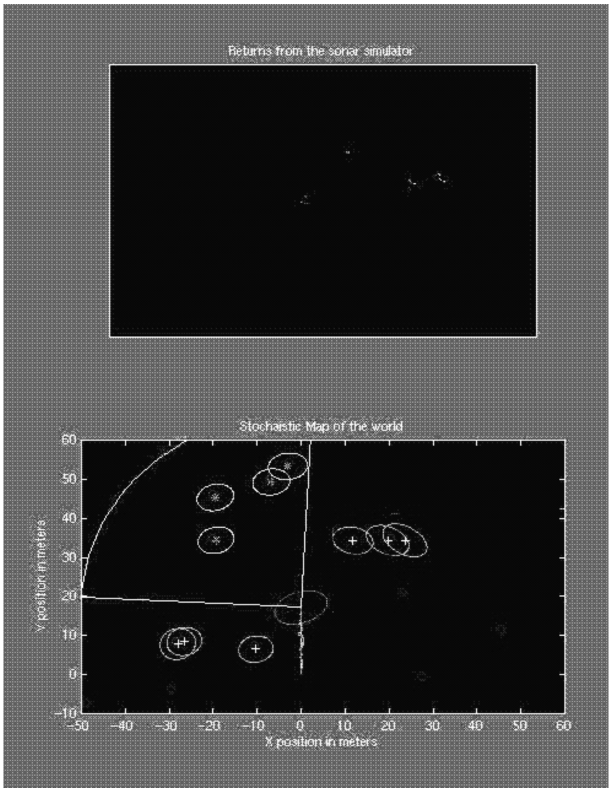

Figure 6: Simulateneous mapping and localisation

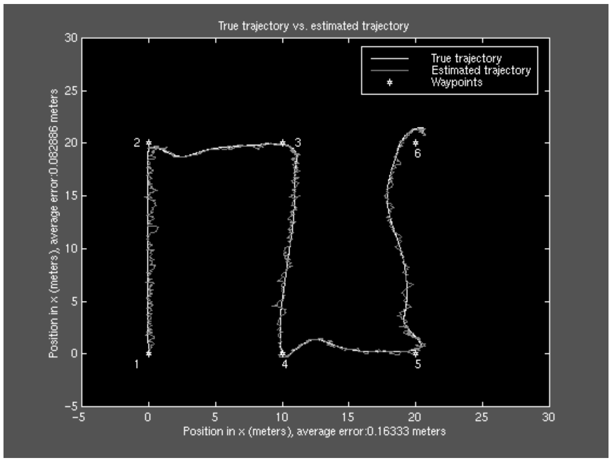

Figure 7: Estimated vs. true trajectory