Underwater Vehicle Obstacle Avoidance and Path Planning Using a Multi-Beam Forward Looking Sonar

Yvan Petillot, Ioseba Tena Ruiz, and David M. Lane, *Member, IEEE*

***Abstract*—This paper describes a new framework for segmentation of sonar images, tracking of underwater objects and motion estimation. This framework is applied to the design of an obstacle avoidance and path planning system for underwater vehicles based on a multi-beam forward looking sonar sensor. The real-time data flow (acoustic images) at the input of the system is first segmented and relevant features are extracted. We also take advantage of the real-time data stream to track the obstacles in following frames to obtain their dynamic characteristics. This allows us to optimize the preprocessing phases in segmenting only the relevant part of the images. Once the static (size and shape) as well as dynamic characteristics (velocity, acceleration, …) of the obstacles have been computed, we create a representation of the vehicle's workspace based on these features. This representation uses *constructive solid geometry* (CSG) to create a convex set of obstacles defining the workspace. The tracking takes also into account obstacles which are no longer in the field of view of the sonar in the path planning phase. A well-proven nonlinear search (sequential quadratic programming) is then employed, where obstacles are expressed as constraints in the search space. This approach is less affected by local minima than classical methods using potential fields. The proposed system is not only capable of obstacle avoidance but also of path planning in complex environments which include fast moving obstacles. Results obtained on real sonar data are shown and discussed. Possible applications to sonar servoing and real-time motion estimation are also discussed.**

***Index Terms*—Obstacle avoidance, path planning, segmentation, sonar, tracking, underwater robotics.**

I. Introduction

WE ADDRESS the general path planning and obstacle avoidance problem for an underwater vehicle using high resolution, real-time sonar sensory data. Although related problems such as two-dimensional (2-D) map building, environment modeling [1], [2] and motion estimation could be tackled in the framework of the presented system, we will mainly focus on obstacle avoidance and path planning.

Until recently, most obstacle avoidance systems used low resolution or low frame rate sonar sensors yielding inaccurate estimations of the obstacles positions and movement. These systems were suitable for reactive obstacle avoidance ("reflex behavior") but not for real path planning in a complex and changing environment [3][1], [4][2].

With the recent development of reliable, high resolution, multi-beam sonars, a new range of methods have emerged, which allow for a more detailed description of the environment and have broadened the spectrum of techniques that can be used [5]–[9]. It is now possible to get a real-time update of the sensed environment leading to a better understanding of the scene and the capability to handle complex and changing environments [10].

A. Target Application

The target of this research is to develop an obstacle avoidance system for the Advanced ROV Package for Automatic Mobile Investigation of Sediments (ARAMIS) tool-skid, where ROV stands for Remotely Operated Vehicle. The ARAMIS project (MAST-CT97-0083) provides a geological/scientific tool-skid which will be mounted on two different ROVs, VICTOR from IFREMER (France) and ROMEO from CNR-IAN (Italy), operating at a close distance from the seabed (2 meters) at depths ranging from 50 to 2000 m. The main missions of the ROVs are geological and biological surveys of the seabed and water column, including benthic and pelagic missions which could last up to 72 h. The need for an automated piloting system, or at least an aided piloting system, is clear. The cruising speed for both ROVs is around 1 kn and the movements of the ROVs are measured by several on-board sensors feeding the obstacle avoidance system with position, speed and orientation of the vehicle in world coordinates. The positioning system will consist in a long baseline system (LBL), an super short baseline system (SSBL) and of various inertial navigation sensors (INS), depending on the vehicle such as compasses, gyro-meters and Doppler velocity log (DVL). Although at a depth of up to 2000 m, the baselines systems are likely to be imprecise, and INS sensors will drift with time, the obstacle avoidance module does not require long-term accuracy as it is creating a *local* map of the environment. The drift of the INS sensors should be negligible on short periods and ensure the successful creation of a precise local map.

Manuscript received January 19, 1999; revised November 7, 2000. This work was supported by the European community MAST Project under Grant CT97-0083.

The authors are with the Ocean Systems Laboratory, Department of Computing and Electrical Engineering, Heriot-Watt University, Riccarton, Edinburgh EH14 4AS, U.K.

Publisher Item Identifier S 0364-9059(01)03491-4.

[1]See Y. Petillot. A local navigation technique with obstacle avoidance, called adaptive navigation, is proposed for mobile robots in which the dynamics of the robot are taken into consideration. The only information needed about the local environment is the distance between the robot and the obstacles in three specified directions, The navigation law is a first-order differential equation and navigation to the goal and obstacle avoidance are achieved by switching the direction angle of the robot, The effectiveness of the technique is demonstrated by means of simulation examples.

[2]See Y. Petillot.

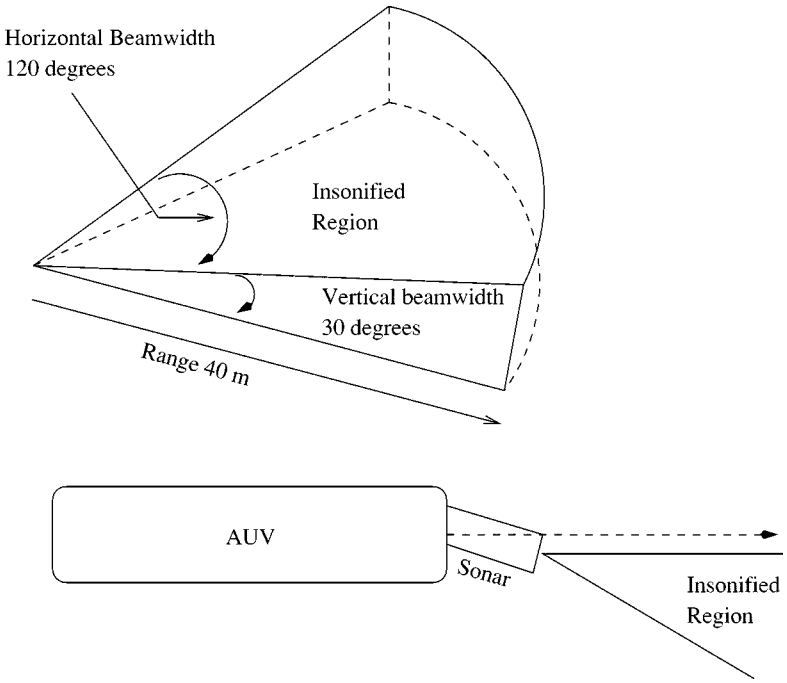


Fig. 1. FAU sonar characteristics and mounting configuration.

B. Sonar Data Collection

The sonar used for collecting the data used throughout this paper is the in-house built multibeam sonar of Florida Atlantic University (FAU) [11][3]. This sonar was mounted on the Ocean Explorer AUV looking forward and slightly downward as depicted in Fig. 1.

It has the following characteristics:

- number of beams: 120;
- vertical beam-width: 30°;
- horizontal beam-width: 1°;
- sector: 120°;
- operating frequency: 600 kHz ;
- operating range: 40 m.

C. System Overview

When one wants to address the problem of obstacle avoidance and path planning in a partially sensed environment, the main problem encountered is the extraction of information from the input data to create a representation of the environment that is as close as possible to the "*ground truth*" scene and can be interpreted in terms that are suitable for computation.

In our case, the first aim is to detect and avoid obstacles. Therefore, the segmentation of the scene is the first and crucial task. It is also useful to know how these obstacles are moving with respect to the vehicle in order to take movement into account in the obstacle avoidance/path planning process. We have introduced a tracking module to perform this operation. Once the obstacles' static and dynamic characteristics have been computed from the input data flow, we create a workspace of obstacles surrounding the vehicle. This workspace uses constructive solid geometry (CSG) to create a convex representation of the obstacles, which eases convergence of the path planning algorithm. The map of obstacles surrounding the vehicle takes into account the obstacles currently in the field of view of the vehicle but also obstacles that have gone out of the field of view, which have been tracked in the near past, and whose position is still critical to the definition of a safe path for the vehicle.

The system we have designed (see Fig. 2) is modular in nature. Modularity is seen as a way to handle different needs within the same framework.

1) Segmentation: The purpose of this module is to identify the interesting regions of the image containing obstacles. Considering the very nature of multi-beam sonar images, we have decided to discard the certainty grid approach [12] often used in air ultrasonic sensor based motion planning and to focus on an object oriented description of the workspace. As the vehicle is moving close to the seabed, high backscatter seabed returns are expected. This backscatter must be estimated and removed when possible in the segmentation process.

2) Feature Extraction: Once the image has been segmented, potential obstacles and their features (position, moments, area) are computed. These features will be used later to discard false alarms and track the obstacles and the vehicle.

3) Tracking: This module provides a dynamic model of the obstacles. Moreover, considering the amount of data to be processed, the tracking drives the segmentation and reduces the computational cost.

4) Workspace Representation: From obstacles and features extracted from the current image, we can build a symbolic representation of the vehicle's surrounding. Combining this representation with previous instances of the vehicle's environment, a dynamic workspace is built and constantly updated. It forms the basis for the path planning algorithm. In this workspace, each object is represented by its current position, shape and estimated velocity. The objects shape is assumed to be elliptic and real objects are represented as ellipses. This workspace can be seen as

[3]See Y. Petillot.

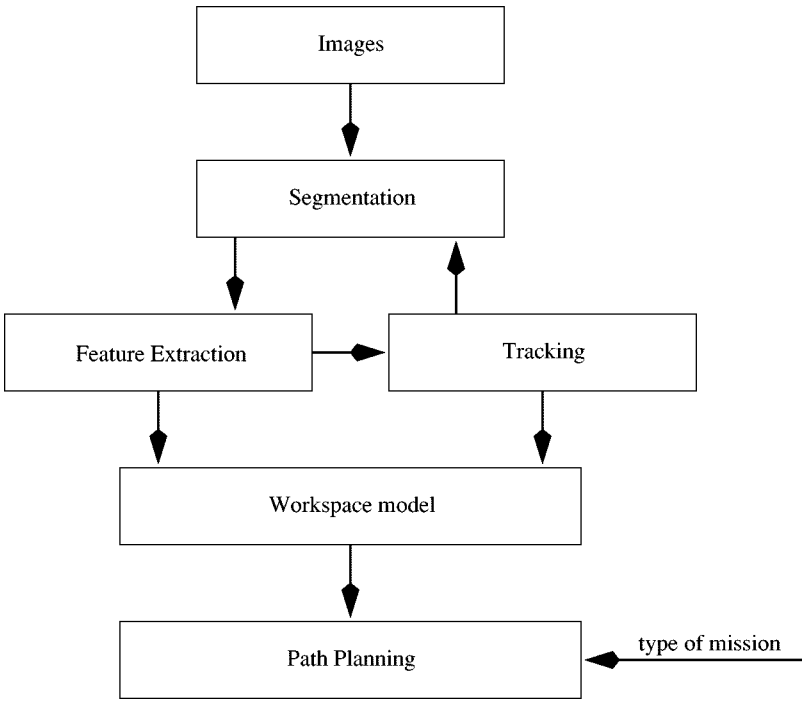


Fig. 2. Architecture of sonar-based, real-time path planning.

a *local* map of symbolic objects with their associated estimated static (shape, position) and dynamic (velocity) properties.

5) Path Planning: A nonlinear programming technique based on a CSG representation of the obstacles is used for path planning. Each obstacle in the workspace is represented as a constraint that has to be met in the search space (that is, the path must not cross the obstacle) while minimizing the Euclidean distance to the goal. This approach takes forward some of our previous work [7], [8].

6) Plan of the Paper: The plan of the paper is as follows. Section II details the segmentation technique we have developed. Section III reviews the feature extraction and tracking modules. Section IV explains the workspace model building procedure and the path planning algorithm. Results are shown for each module and the general path planner is demonstrated in Section V using real sonar data. Section VI draws a summarizing picture of our achievements so far while Section VII investigates possible applications of the system to motion estimation and sonar servoing.

II. Segmentation

A. Introduction

Much work has been done on the segmentation of side-scan sonar images, but not so much on forward-looking sonar images. In general, segmentation is performed on still images, or separates the moving and static parts of the images using fast Fourier transform (FFT) techniques [6], [13].

Multi-beam sonar images are generally noisy and need to be filtered. The noise is mainly due to backscatter from either the sea surface or the sea bottom. Two kinds of segmentation procedures can be envisaged: 1) still image segmentation, where each image is segmented independently and 2) image flow segmentation, where the images are segmented taking into account the results of the segmentation of the previous frames. Sonar images are very difficult to segment using a single return from the sonar. As we have a flow of acoustic images at the input of the system at a sufficient frame rate, it is of more interest to design a segmentation procedure that takes into account several frames. Moreover, good segmentation requires very time consuming operations. For this reason, we focus on segmentation of areas of interest on the image, namely the: 1) first layer segmentation: segmentation of areas where a very basic and fast segmentation algorithm indicates that there is a new object and 2) second layer segmentation: segmentation of areas where an object has already been detected in the past, or, if the object is moving with respect to the vehicle, areas where the object is expected to be at the time of the image acquisition. To do this, objects must be tracked throughout a sequence of consecutive frames in order to estimate their current location.

B. First Layer Segmentation

A common segmentation procedure for sonar images consists of median filtering followed by thresholding [6]. Filtering for noise smoothing is an absolute necessity in the case of sonar images as backscatter is common, especially in the case of multibeam sonar images. The filtering part is also generally very time consuming. We have tried several filtering techniques (mean, median, Gaussian) and found that a good compromise between quality and speed was reached using the following scheme.

- **Filtering:** The filter used to remove the backscatter noise is a 7×7 Gaussian filter which yields results almost as good as the median filter even on noisy images but at a reduced computational cost [14].
- **Threshold:** A single, fixed threshold, generally gives results which are highly dependent on the background level. We have used an adaptive thresholding technique based

on the image histogram which is independent of the actual signal level. The idea is to estimate the noise probability density function assuming that the histogram of the image is a good estimate of it (thus assuming that new objects to be detected are small in the image). It must be noted that the calculation of the histogram is done on the original image and not the filtered image.

The predicted locations of previously identified objects are removed from histogram calculation. A false alarm rate (FAR) is then fixed and used in conjunction with the histogram to derive the threshold value. If an object is part of the image on which the histogram has been derived, it contributes, most probably, to the higher part of the histogram and will be selected even with a high false alarm rate, while most of the noise will be rejected. The noise that should be tolerated corresponds to backscatter returns which are at the same level as the objects present in the image. Removing this kind of noise would also remove the objects or part of them.

Special cases where the images are mainly composed of obstacles (with high returns) or with a lot of backscatter noise from the seabed can easily be detected as they show a high variance. The process can then be adapted to these special cases. This technique runs in real time (Sparc Ultra-10 with code in Matlab 5.2) and represents a compromise between real adaptive filtering, where the threshold value is derived *locally* with respect to the surrounding pixels and fixed thresholding.

This scheme is used on a subsampled image to quickly detect potential obstacles in the scene while the second layer segmentation uses more elaborate techniques on selected parts of the scene.

C. Second Layer Segmentation

The second layer takes advantage of the tracking module (see Section III) which tracks objects from frame to frame and predicts their dynamic characteristics (including their next location) with respect to the vehicle.

The process can be decomposed in two distinct parts: 1) selection of the areas of interest in the image and 2) segmentation of these regions.

1) Selection of the Areas of Interest: The tracking is based on Kalman filters. A filter is associated to each object detected in the scene. Let's call $\{O_i\}$ the set of objects currently present in the scene. This set can be decomposed in two subsets $\{NewO_i\}$ and $\{TrackO_i\}$ which, respectively, represent the objects which just appeared on the scene and the objects tracked from previous frames. For each object in $\{NewO_i\}$, an area of interest is set which matches exactly the labeled object resulting from the first layer segmentation. For each object in $\{TrackO_i\}$, an area of interest is set which matches the previously tracked object and is positioned using the Kalman filter prediction. The size of the area of interest is increased by the uncertainty measures on the position and area of the tracked object given by the Kalman filter associated with the object. Once the areas of interest have been set, they can be segmented.

2) Segmentation: The segmentation algorithm is again based on the histogram of the original image, previously computed, to set the thresholds. The image is filtered in the areas of interest using a 7×7 Gaussian filter and a double threshold is applied. The first (higher) threshold selects the parts of the image that is to be taken into account while the second threshold (lower) selects the areas above it which are connected to the regions selected by the higher threshold by a continuous chain of pixels above the lower threshold. Eight or four-connectivity can be used to define the neighborhood of a pixel. We have used eight-connectivity. Fig. 3 gives a simple example of this procedure.

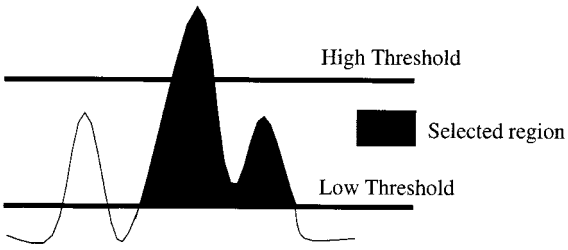


Fig. 3. Example of application of the double threshold algorithm to a simple 1D curve. The regions selected by the algorithm are in grey.

The first interest of this algorithm is to discard middle value peaks, not connected to high returns and generally due to noise, which would be kept by a simple thresholding technique. This algorithm also keeps relatively low intensity pixels connected to high returns which correspond to less reflective parts of an object. An example of segmentation is shown in Fig. 4. The results shown take into account objects which have been tracked for a few frames.

The second layer segmentation algorithm runs in 0.1 s on an Ultra-5 Workstation including the Gaussian filtering, while the first layer algorithms runs in 0.3 second under MATLAB 5.2. The gain in speed for the second layer (while performing a more complex operation) is due to the smaller amount of data processed using the areas of interest. A real-time version of the code is under development.

III. Feature Extraction and Tracking

A. Feature Extraction

Once segmented, the different regions representing the obstacles in the image are labeled using a standard labeling algorithm, and the features for each obstacle are extracted. These features are:

- position in the image (the position is calculated as the centroid of the object);
- area of the object in pixels;
- perimeter of the object in pixels.

B. Tracking

Tracking in forward-looking sonar images has often been tackled using pixel-based techniques such as optical flow associated to tracking trees or multiple hypothesis tracking [9], [15]. These techniques, combined with a segmentation of the scene perform well when the noise level is small but they often require knowledge of the vehicle movement.

We have based our algorithm on a combination of segmentation and feature extraction. This technique does not require the knowledge of the vehicle motion.

The extracted features are the basis of the tracking algorithm. The tracker has two main functions: 1) to reduce the computational cost of the segmentation; 2) to extract the dynamic char-

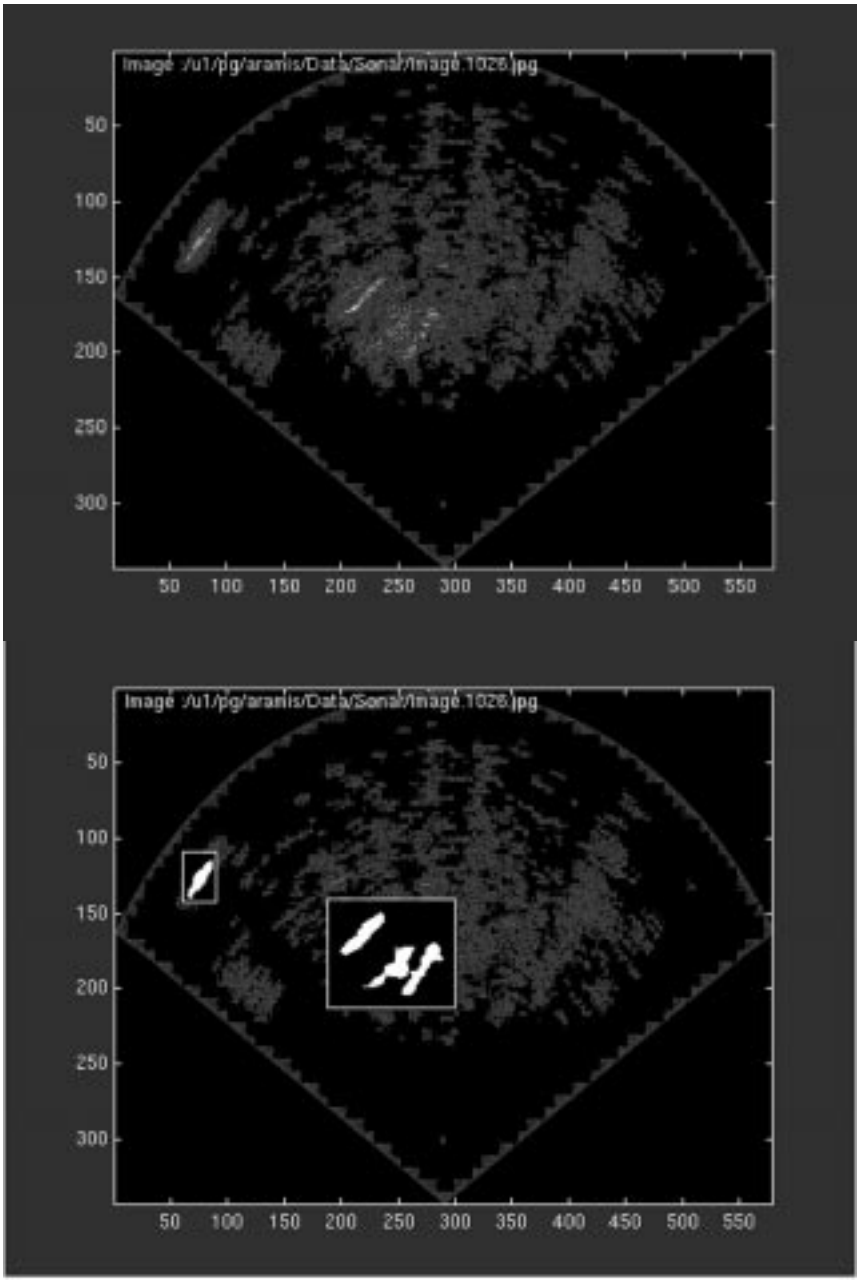


Fig. 4. Segmentation of multi-beam high-resolution sonar images using a Gaussian filter and a double threshold algorithm. Scans were obtained from FAU multi-beam sonar (Courtesy of FAU). The range of the sonar was 40 m, the operating frequency 600 kHz. The x and y labels on the image corresponds to the pixels of the image.

acteristics of the objects for the path planning. We have chosen to use Kalman filters as the core element of our tracking scheme. As mentioned earlier, the noisy nature of sonar images strongly limits the range of tracking techniques that can be used. For instance, pixel based techniques such as correlation or optical flow are very computationally expensive and can yield poor results in very noisy environment. The Kalman filter is a good candidate since:

- it is a fast algorithm when the state vector is small;
- it gives the uncertainty on the estimated parameters. This is extremely useful for path planning purposes as the uncertainty measure can be included in the obstacle avoidance module to increase the safety of the system;
- a model of the vehicle dynamics can be easily included using an extended Kalman filter;
- fusion from different sensors can be performed in a very natural way.

1) Kalman Filter Design: When implementing a Kalman filter, there is always a choice to make between the complexity of the model and the number of states of the filter on the one hand, and the computational cost associated with a given model on the other hand. Moreover, the denser the data flow, the simpler the model can be, and the sparser the data flow, the more accurate the model must be. Multi-beam sonars provide a high frame rate of sonar images (typically between 4–30 frames/s). A simple model can therefore be used. In our tracking algorithm,

a Kalman filter is associated to each object and we track the objects with respect to the vehicle. The Kalman filter used has the following characteristics: the state vector, X, is composed of the position in x and y coordinates, the area a of the object and their associated first and second derivatives

$$X = [x \quad \dot{x} \quad \ddot{x} \quad y \quad \dot{y} \quad \ddot{y} \quad a \quad \dot{a} \quad \ddot{a}]^T. \tag{1}$$

This model handles piecewise linear motions. If the frame rate is high enough, one can expect to cope with almost any smooth general motion interpolated as a succession of piecewise linear motions.

Note that we could have designed a single Kalman filter integrating all the objects to take into account the correlation between them. However, the Kalman filter complexity is in $O(n^3)$ where n is the number of states. This would have lead to very computationally expensive tracking.

The state model is given by

$$X_{k+1} = MX_k + W_k \tag{2}$$

while the measurement model is given by

$$Z_k = HX_k + V_k \tag{3}$$

where the random processes W_k and V_k represents respectively the process and the measurement noises which are assumed independent (of each other), white, and with normal probability distributions of respective covariance matrices Q and R. Our model is a simple linear model where the matrix M can be expressed as

$$M = \begin{bmatrix} 1 & \Delta t & \frac{1}{2}\Delta t^2 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & \Delta t & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & \Delta t & \frac{1}{2}\Delta t^2 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & \Delta t & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & \Delta t & \frac{1}{2}\Delta t^2 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & \Delta t \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \end{bmatrix} \tag{4}$$

while the observation is directly the position and area of the object giving:

$$H = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \end{bmatrix}. \tag{5}$$

The initialization of R and Q are standard and can be found in [16]. The initialization of P_k is also a critical factor as it drives the data association. It is initialized at a high value to avoid data association problems.

The Kalman recursion is then applied as usual to:

- cCompute the gain (blending factor) as

$$K_k = P_k^- H^T \left(HP_k^- H^T + R\right)^{-1} \tag{6}$$

- update estimate

$$\hat{X}_k = \hat{X}_k^- + K_k \left(Z_k - H\hat{X}_k^-\right) \tag{7}$$

- update the covariance matrix

$$P_k = (I - K_k H)P_k^- \tag{8}$$

- get the prediction

$$\hat{X}_{k+1}^- = M\hat{X}_k \tag{9}$$

- estimate the predicted covariance matrix

$$P_{k+1}^- = MP_k M^T + Q \tag{10}$$

If we assume the position in x and y and the area a to be independent, then the Kalman filter can be split into three smaller filters, one for each variable x, y and a, resulting in faster processing. However, keeping variables together allows to take into account the interdependency of the parameters via the covariance matrix P_k. We have used the larger filter version.

2) Data Association: In order to track objects in a sequence of images, the first step to perform once the Kalman filters are created and initialized is to associate new observations (objects) in the current image with previously identified objects. This is known as the *data association phase*. Multi-target tracking is a difficult problem and has been extensively studied in the past [16]. A compromise must be found between performance and computing time as our system has to run in real time. We have therefore decided to use a nearest neighbor algorithm. The algorithm uses position and area to perform the association and has been modified to handle the merge of two objects and the split of one object into two distinct objects. This scenario is very likely to occur as the segmentation process is automatic and the intensity of the returns highly variable in sonar images.

The data association algorithm has the following structure.

1) Calculate the distances between all the observations and the predicted positions of the tracked objects.
2) For each tracked object, select the observations that:
   - Either falls within the validation gate of the Kalman filter for position in x and y. The validation gate is defined as the uncertainty on a state of the Kalman filter given by the covariance matrix of the filter.
   - Or intersect with the tracked object.
3) For all the selected observations, verify if the areas are compatible.
4) Prune the selected observations as follows.
   - If a single observation falls within the validation gate of the tracked object and its area is compatible, keep it and discard the others.
   - If two or more observations fall in the validation gate of the tracked object and are compatible in area, take the closest one in terms of position.
   - If no observation falls in the validation gate matches the area of the tracked object, there might be a split or a merge. Try to aggregate observations to have

a compatible resulting area. In case of failure, take the closest observation.

- There is no observation in the validation gate of the tracked object. An error in position estimation might have occurred. Select the observations that are intersecting with the tracked object. If they are not a best choice for another tracked object, consider them, and repeat the merge and split test as previously described.

3) Tracking Update: Once the data association has been applied, we can update the tracking. Three cases are then possible:

- There is a new observation matching the predicted position. The Kalman filter recursion is applied, a new state vector derived and new internal values computed.
- No new observation matches the prediction. The obstacle prediction is updated using the Kalman filter internal values which are not updated. If no match is found between the observations and a given tracked object on a predefined number of frames, the tracked object is discarded as a false alarm.
- An observation is not associated with any tracked object, a new object is created and its corresponding Kalman filter initialized.

C. Tracking Results

This scheme has been successfully used with real sonar data provided by FAU. The sonar used was the FAU multi-beam sonar [11]. The data was acquired using Ocean Explorer Autonomous Underwater Vehicle (AUV) moving at around 1 knot. The range of the sonar was set to 40 meters. The data was sampled at 9 frames/s but processed at 3 frames/s. Hence real-time was not achieved for the tracking. However, an optimized implementation of the algorithm would allow real-time processing at 9 frames/s. Fig. 5 shows the estimated trajectory of the tracked objects with respect to the vehicle on a sequence of 300 frames with a 9-frames/s frame rate. Images where subsampled by a factor of 2, in both x and y, to speed up the procedure. The resulting images had a resolution of 550×400 pixels. The latest estimate speed vector is also displayed on the figure. Fig. 5 contains four examples of the tracking at different stages. The image number (on the figure) indicates what image of the sequence is considered.

The position and heading given by the inertial navigation system are available, enabling benchmarking our results. The inertial sensors cannot take currents into account leading to uncertainties in the position while the heading remains accurate. We have used the heading values as a benchmark for our tracking.

Assuming the objects are still, the tracking in fact detects the motion of the vehicle with respect to the seabed. Estimating the rotation of the objects between frames in the image yields an estimation of the vehicle rotation in the world reference frame. Therefore, we can compare the readings of the INS heading difference between two frames and the corresponding estimated heading difference using the tracking.

The inertial sensors refresh rate is about 1 s while our estimation is given at 9 frames/s. Therefore, differences between the two measures can also be due to the latency in the refresh of the inertial sensor value. In order to minimize this effect we have linearly interpolated the INS readings when they were not available for a given image.

Table I shows the estimated values of the heading difference compared to the values given by the inertial sensors for different frames. The trajectory of the best tracked object (visual assessment) was used to compute the estimated headings. For instance, for objects too close to the sonar head (less than 10 meters), near field effects corrupts the results.

These results seems to indicate a good match between our estimation, based on image processing alone and the inertial sensor measures of the vehicle. The tracker could therefore be used as a secondary navigation sensor which offers the following advantages over classical inertial sensors.

- It has a high frame rate.
- It can take into account currents as it tracks objects in the seabed.

The application of these results to navigation are beyond the scope of this paper although preliminary encouraging results have been obtained. We are planning to investigate this issue in more details in the near future.

IV. Workspace Representation and Path Planning

The choice of a workspace representation is intimately linked with the path planning technique used. Most path planning algorithms assume a convex representation of the obstacles to ensure that the goal will be reached. When dealing with a changing environment which is sensed on the fly, it is advisable to use a reactive path planning technique which does not need a complete description of the workspace between the current position and the goal. The reasons for this are as follows.

- Only partial information is available (due to the limitations of the sensor).
- New obstacles can appear in the workspace at any time.
- The precision in the representation of the obstacles will change with their respective distance to the vehicle.

Global path planning techniques need a complete description of the workspace as they define the complete path from the starting point of the vehicle to the goal, generally using visibility graphs (see [12] for a review on the subject) while local path planning only defines a partial path toward the goal given a possibly incomplete representation of the neighboring workspace [17]–[19]. Global path planning is not advisable here as the environment is sensed while moving and therefore the workspace is continuously changing and only partially known.

We have chosen to use a local path planning technique based on some of our previous work [7], [8] where only the next step of the path leading toward the goal is calculated. The central idea of the method is to represent the free space of the workspace as a set of inequality constraints of a nonlinear programming problem. The goal point is designed as a unique global minimum of the objective function. The initial configuration of the vehicle is treated as the starting point of the nonlinear search. CSG is used to represent the free space of the robot as a set of inequalities.

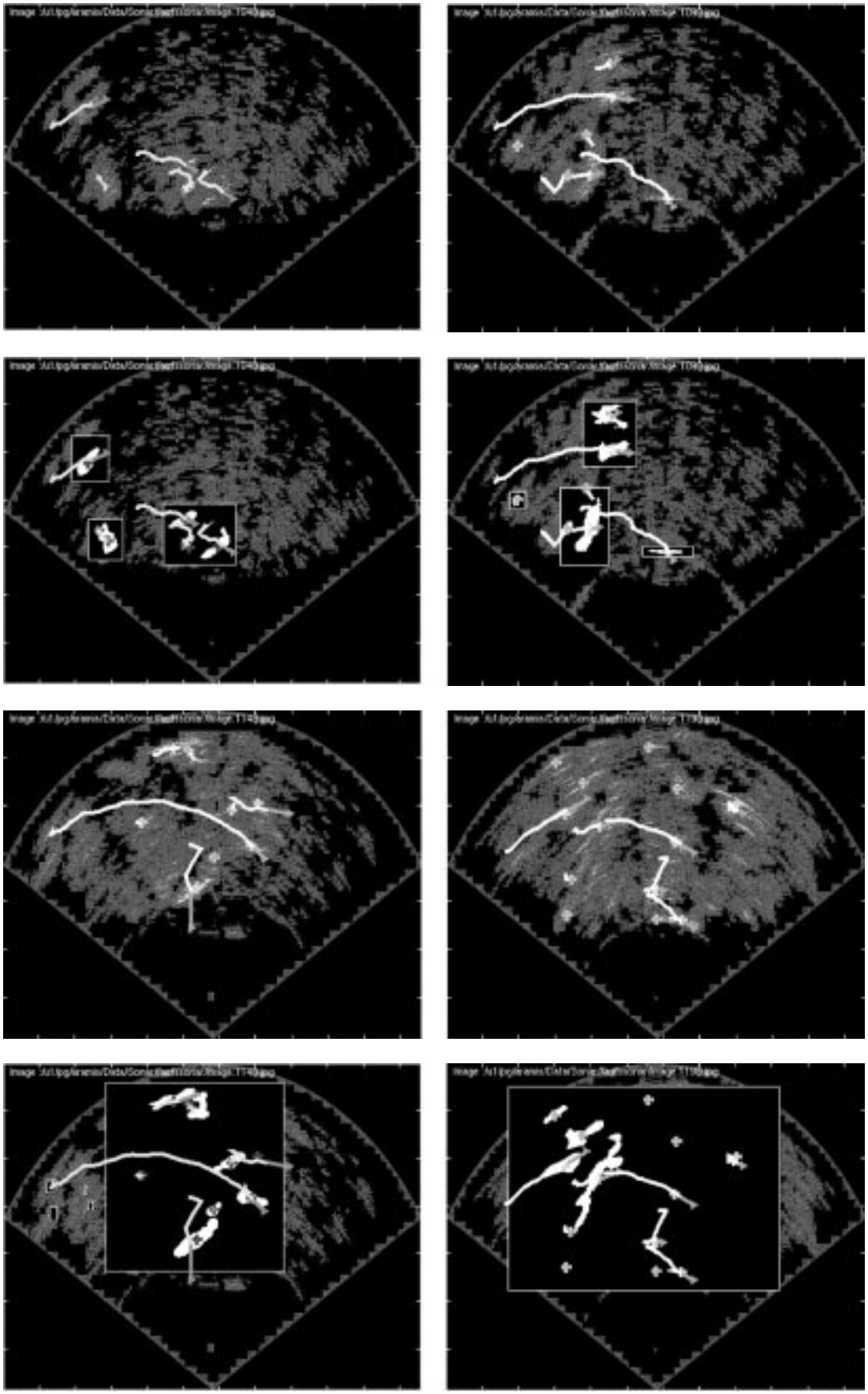


Fig. 5. Example of tracking results on a 300 images sequence of multi-beam sonar images. The tracked trajectory is denoted as a yellow line while the center of the object is denoted as a red cross and the estimated speed vector as a green arrow.

TABLE I
COMPARISON OF THE HEADING ESTIMATES USING THE TRACKER AND THE READINGS OF THE INERTIAL SENSORS ON THE VEHICLE. THE LEFT COLUMN CORRESPONDS TO THE INTERPOLATED READINGS OF THE INERTIAL SENSORS TO TAKE INTO ACCOUNT THE LOWER REFRESH RATE

Frame Numbers	Estimated differential Heading	Inertial differential Heading	Inertial differential Heading (interp.)
1048 - 1148	73.5°	75°	73.2°
1098 - 1148	36.17°	35.08°	35.08°

A. Workspace Representation Using CSG

In the following section, we operate in the configuration space of the vehicle. It means that in this space, which integrates both the kinematics and the link geometry of the vehicle, the vehicle can be represented as a point.

The choice of CSG to represent obstacles is driven by the fact that classical surfaces such as spheres, cylinders, and half-spaces are CSG primitives that can be very easily combined.

Each obstacle in the workspace is given a mathematical representation. Let S be the 2-D or three–dimensional (3-D) surface of an Euclidean space E representing the obstacle, and let's denote its interior points by I, its boundary points by B and its exterior points by T in a topological sense

$$I \cup B \cup T = E$$
$$I \cap B = B \cap T = I \cap T = \emptyset. \tag{11}$$

The nonnegative function f on E is called a defining function (in the CSG sense) of the obstacle S if

$$\begin{aligned} &\forall p \in I, && 0 < f(p) < 1 \\ &\forall p \in B, && f(p) = 1 \\ &\forall p \in T, && f(p) > 1. \end{aligned} \tag{12}$$

As an example, the defining function of an ellipse when $E = \mathbb{R}^2$ is

$$\forall p \in \mathbb{R}^2, \qquad f(p) = (x/a)^2 + (y/b)^2 \tag{13}$$

where a and b are the half-axes of the ellipse and p is the point of coordinates (x, y) in the plan.

One of the major interests of CSG lies in the fact that complex objects can easily be constructed from simple canonical objects using the union and intersection operations

$$\forall p \in E, \qquad f^I(p) = \max(f_1(p), f_2(p), \ldots, f_n(p)) \tag{14}$$

defines the intersection of n objects whose respective defining functions are $f_1, f_2, \ldots, f_n$ while

$$\forall p \in E, \qquad f^U(p) = \min(f_1(p), f_2(p), \ldots, f_n(p)) \tag{15}$$

defines the union of the same objects. These functions are difficult to obtain in practice and are replaced by the following approximation:

$$f^I = (f_1^m + f_2^m + \cdots + f_n^m)^{1/m} \tag{16}$$

approximates the intersection of n objects whose respective defining functions are $f_1, f_2, \ldots, f_n$ while

$$f^U = \left(f_1^{-m} + f_2^{-m} + \cdots + f_n^{-m}\right)^{(-1)/m} \tag{17}$$

approximates the union of the same objects. m is any positive real number. m can be used to control the accuracy of the smooth approximation and can be used to obtain convex unions and intersections.

Here, for the sake of simplicity and without loss of generality, we have decided to represent the obstacles as ellipses. More general representation are possible, including polygonal ones [8]. From the real obstacles contours, their convex hull is extracted from the pixel-based representation of the objects and an optimal elliptic fitting algorithm is applied to obtain the representation of a given obstacle. Multi-beam sonars generally have between 64 to 240 receiving beams whose aperture is in the range [0.5°, 2°] horizontally and [15°, 30°] vertically. The FAU sonar has 120 beams with a 1° horizontal beam-width and a 30° vertical beam-width. The 2-D resulting image is a projection of a 3-D insonified region. The detected obstacles could therefore be above or below the vehicle vertical position. This problem can be addressed by adapting the vertical beam-width to the typical operating distance of the vehicle to the seabed in order to take only the objects in the water column into account. Bottom detection can also be performed and used to determine if an object is lying on the seabed or not. Finally, the shadow created by an object can be detected and processed to extract its 3-D characteristics. These techniques are widely used in side-scan sonar image processing but very little has been done in this field concerning forward-looking sonar images. This issue in itself is a wide subject for future research and falls beyond the scope of this paper.

B. Path Planning Algorithm with Static Objects

In this case we assume all the objects are static objects in the world reference frame. All obstacles O_i ($i \in [1, n]$) of the workspace are defined as ellipses whose defining functions g_i in a 2-D Euclidean space are defined in equation (13). The free space of the vehicle with respect to obstacle O_i is defined as

$$\{p \in E \mid 1 - g_i(p) < 0\}. \tag{18}$$

Therefore the complete free space of the vehicle can be represented as

$$\{p \in E \mid \forall i \in [1, n],\ \ 1 - g_i(p) < 0\}. \tag{19}$$

Let us now define the objective function f representing the practical problem to be solved, in our case, the minimum distance from the start to the goal point in the configuration space as

$$\forall p \in E, \qquad f(p) = (p - p_g)^T (p - p_g) \tag{20}$$

where p_g designs the goal point and ^{T} is the transpose operation.

We now have completely defined our path planning problem

Optimize f under the constraints $g_i, i \in [1, n]$.

This is a classical problem in optimization. As in the general case the defining functions are nonlinear, we use well-proven numerical nonlinear programming techniques to solve the path planning problem. This approach generates very smooth paths

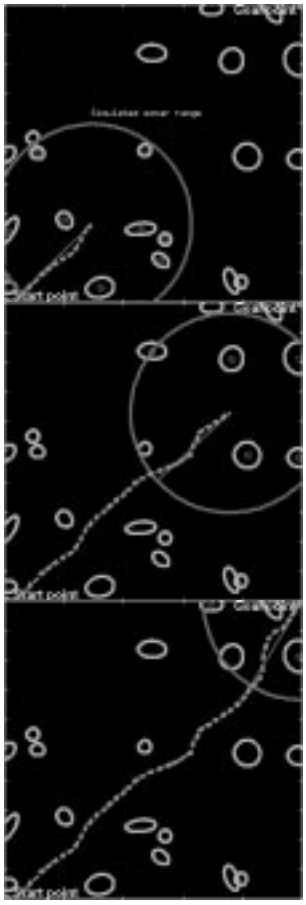


Fig. 6. Example of generated path on a simulated workspace. The Starting point is the bottom left corner while the goal point is the top right corner. The vehicle speed is limited to 5 m/s. The simulated sonar's field of view is defined by the circle. The objects in the field of view and currently used for the path planning process are denoted with crosses (+ sign) while the ones that will be used in the next iteration of the path planning algorithm are denoted with squares.

compatible with feasible vehicle motion. The effect of each constraint can be clearly seen while it is often hidden in a single objective function in other optimization techniques such as potential fields where the careless definition of the potential functions can easily lead to local minima. Finally, the CSG modeling of the obstacles offers a lot of flexibility in the representation of the workspace. An example of a generated path on a simulated virtual workspace containing ellipses of random size and position is given in Fig. 6. Some limitations of the vehicle such as maximum speed or maximum rotation can also be taken into account using additional nonlinear constraints. In this simulation, the vehicle speed has been limited to 5 m/s. This is added as a new constraint. Other vehicles' dynamic limitations can be added if they can be stated as nonlinear constraints.

C. Path Planning Algorithm with Moving Objects

Dealing with moving objects is a natural and desirable property of any obstacle avoidance system. Most systems use a static representation of the objects which if updated at a high frame rate allows to take the movement of the obstacles into account. However, this approach has several limitations. First, the objects can never be found at their real position during the planning process (between two updates of the workspace) and second, it can lead to very suboptimal paths when the robot is pushed away from its trajectory by a moving object. The best way to take into account moving objects is to go from a 2-D workspace to a 3-D workspace, the third dimension being time [20], [21].

The workspace representation of the objects presented in the previous sections does not take into account their dynamic properties which were extracted by the tracker. It is straightforward to include these properties in the description of the objects using CSG. The parameter p describing the position of the robot in the configuration space will include time as a new variable and the defining function will also depend on time.

As an example, the defining function of an ellipse (obstacle) in $E = \mathbb{R}^3$ will be

$$\forall p \in \mathbb{R}^3, \qquad f(p) = (x(t)/a(t))^2 + (y(t)/b(t))^2 \qquad (21)$$

where a and b are the half-axes of the ellipse and p is the point of coordinates (x, y, t) in $\mathbb{R}^3$.

The workspace should represent the objects in a world reference frame and only objects which are moving in this frame (*and not with respect to the vehicle*) should be considered as moving objects. If the motion of the vehicle is not known, one can assume the mean motion of the objects (with respect to the vehicle) as the estimated vehicle motion.

The path planning problem can then be reformulated. All the equations derived for the static case are still valid. The only difference is that now, p not only is a function of the position of the robot in the configuration space but also a function of the time t. This is equivalent to saying that the configuration space has one more dimension.

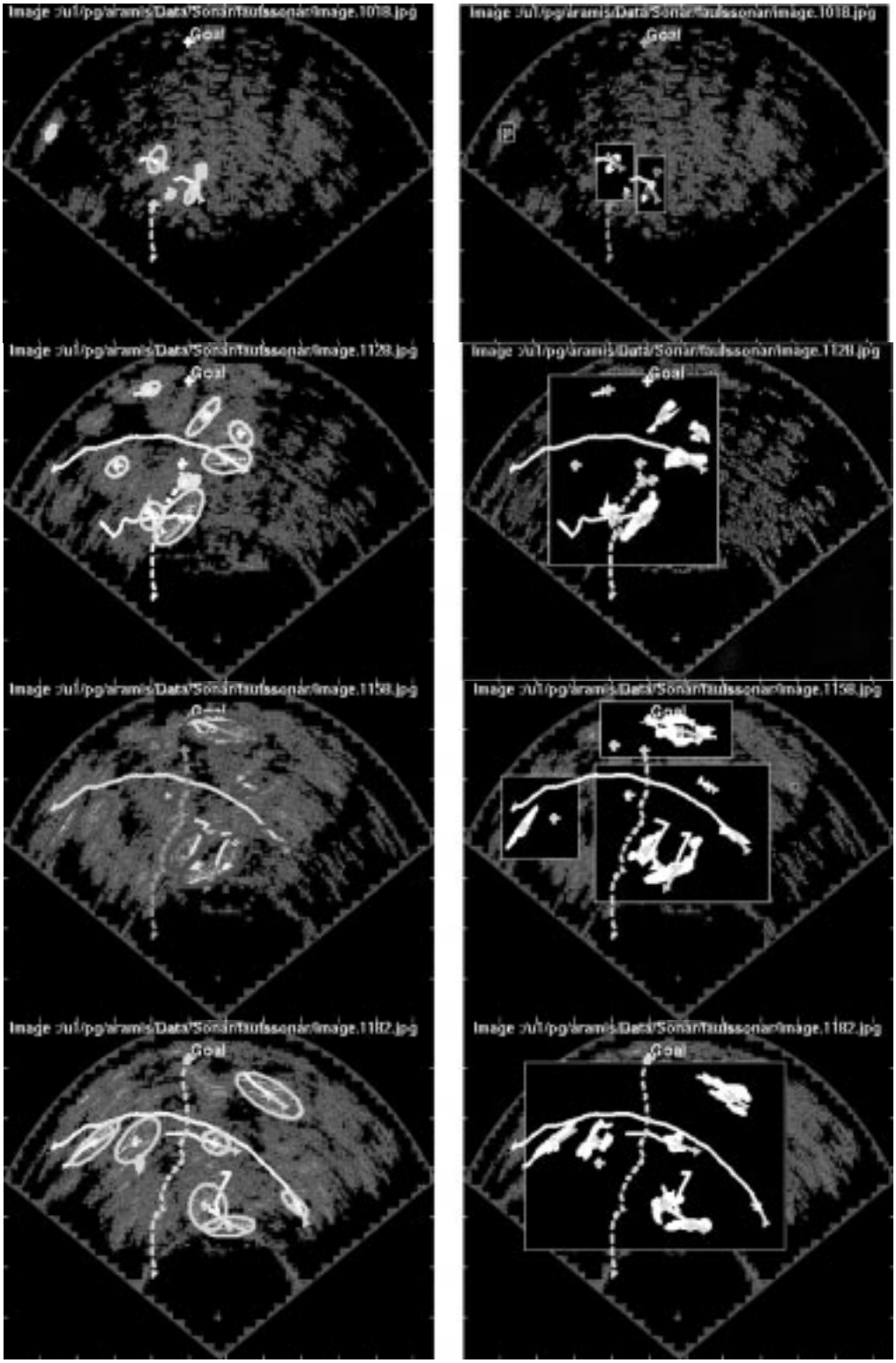
However, time is a variable which has special properties. For instance it must be strictly positive and it must always increase. These constraints have to be added in the list of constraints in the nonlinear optimization. The other implication of this reformulation is that the goal is now defined in time as well. However, it is impossible to know *a priori* when the vehicle will reach the goal as the path is unknown. The best way to tackle this problem is to assume that the vehicle will reach the goal in a straight line (no obstacles) at its maximum speed and to state the goal in time according to this assumption. Then, at each iteration of the path planning algorithm, the new time to goal is computed and a new goal is set. Its physical position remains unchanged but the time of arrival of the vehicle to the goal is changed according to the path followed by the vehicle. This technique ensures a feasible solution at each iteration.

This time varying representation of the workspace can easily integrate the tracking information extracted from the objects to yield a more reliable path planning algorithm. The results obtained using this scheme on real-sonar data are shown in the next section.

V. Path Planning Results

We have tested the combination of the segmentation, the feature extraction, the tracking and the path planning modules on real sequences of sonar data. In order to test the algorithm, we used sequences provided by FAU taken from the forward looking sonar mounted on the front of the Ocean Explorer AUV.

As we cannot close the loop to control the vehicle movement, we have simulated the movement of a "blind" ROV, driven ac-



(a) (b)

Fig. 7. (a) Example of path planning on a sequence of real images with the elliptic representation of the objects. (b) Same example with the segmented images. The blue cross represents the goal. The objects are represented as red ellipses and the trajectory is depicted as a dashed green line. Tracking legends are the same as the ones used previously.

cording to the data received from the sonar. This does not alter the validity of the approach. The path planning is performed in the vehicle reference frame and not in the real world reference frame to simulate moving objects. This is equivalent to having a still sonar looking at a moving environment while in the real mission we should expect a moving vehicle in a mostly still environment (mostly still features and maybe a few moving objects).

An animated MPEG version of the results displayed in Fig. 7 can be found on our Web page at: http://www.cee.hw.ac.uk/~aramis/resources/.

The goal is set so that the generated path crosses the path of the moving obstacles. On Fig. 7, the left image is the original image while the right image is the segmented image showing the identified obstacles. On the segmented images, the obstacles contour are displayed. The planned path is also drawn on both sequences of images.

In order to achieve faster processing time, the original images (1200×700 pixels) were subsampled by a factor of 2 in both directions. Using Matlab 5.2 on a Sun Ultra-10, the whole process (segmentation, workspace representation and path planning) takes 3 s/frame. Considering a frame rate of a few images per second and using optimized code, a real-time system is certainly achievable using the framework presented here.

VI. Conclusion

We have presented here a general framework for performing 2-D obstacle avoidance and path planning for underwater vehicles based on a multi-beam forward looking sonar sensor. This scheme has been shown to be expendable to moving obstacles.

The ability of the system has been demonstrated on real sonar data. The sequence used corresponds to a real trial, and the ability of the system to perform obstacle avoidance is demonstrated. Compared to other methods, our system generates very smooth paths, can handle complex and changing workspaces and presents no local minima as we use a convex representation for the obstacles.

This system can also be used for motion estimation using the tracking module and some applications such as sonar servoing, simultaneous localization and mapping can be handle within this framework.

VII. Future Work

We are currently building a small ROV (http://www.cee.hw.ac.uk/kelvin/rauver_index.html) which will integrate an obstacle avoidance module in closed loop to test the validity of our approach during see trials. We will investigate the possible application of the tracking module to motion estimation and vehicle localization. Applications to sonar servoing and automatic docking will also be studied.

Acknowledgment

The authors wish to thank Dr. Y. Wang for useful discussions on path planning and obstacle avoidance, and Prof. J. Cushieri, Ocean Engineering Department, Florida Atlantic University, for providing the sonar data.

References

[1] J. J. Leonard and H. F. Durrant-Whyte, *Directed Sonar Sensing for Mobile Robot Navigation*, ser. The Kluwer International Series in Engineering and Computer Science. Norwell, MA: Kluwer , 1992.

[2] R. Bono, M. Caccia, and G. Veruggio, "Reconstructing 2-D maps from multiple sonar scans," in *Proc. IEEE Oceans Conf.*, vol. 1, 1994, pp. 164–169.

[3] A. Fujimori, P. N. Nikiforuk, and M. M. Gupta, "Adaptive navigation of mobile robots with obstacle avoidance," *IEEE Trans. Robot. Automat.*, vol. 13, pp. 596–602, Aug. 1997.

[4] A. J. Healey, D. B. Marco, R. B. McGhee, D. P. Brutzman, F. A. P. R. Christi, and S. H. Kwak, "Tactical/execution level coordination for Hover control of the NPS AUV II using onboard sonar servoing," in *Proc. Symp. Autonomous Underwater Vehicle Technology*, 1994, pp. 129–138.

[5] L. Henriksen, "Real-time underwater object detection based on electrically scanned high-resolution sonar," in *Proc. IEEE Symp. Autonomous Underwater Vehicle Technology, AUV'94*, Cambridge, MA, July 1994.

[6] M. J. Chantler and J. P. Stoner, "Automatic interpretation of sonar image sequences using temporal feature measurements," *IEEE J.Oceanic Eng.*, vol. 22, pp. 47–56, Jan. 1997.

[7] Y. Wang and D. M. Lane, "Subsea vehicle path planning using nonlinear programming and constructive solid geometry," *Proc. IEE Control Theory Appl.*, vol. 144, pp. 143–152, 1997.

[8] Y. Wang and D. M. Lane, "Path planning for underwater vehicles using constrained optimization," in *Proc. Oceanology Int. Conf.* , Brighton, Mar. 1998, pp. 175–186.

[9] 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 J.Oceanic Eng.*, vol. 23, pp. 31–46, Jan. 1998.

[10] 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 *Proc. OCEANS'98*, vol. 2. Nice, Italy, Sept. 1998, pp. 1194–1199.

[11] L. LeBlanc, J. M. Cushieri, P. Beaujean, and M. Singer, "Electronically steered and focused forward-looking scan sonar," in *Proc. Symp. Autonomous Underwater Vehicle Technology*, Monterey, CA, June 1996.

[12] J. C. Latombe, *Robot Motion Planning*. Boston, MA: Kluwer Academic, 1991.

[13] D. M. Lane and J. P. Stoner, "Automatic interpretation of sonar imagery using qualitative feature matching," *IEEE J. Oceanic Eng.*, vol. 19, pp. 391–405, 1994.

[14] R. Gonzalez and R. Woods, *Digital Image Processing*. Reading, MA: Addison-Wesley, 1992.

[15] J. Cushieri and S. Negahdaripour, "Use of forward scan sonar images for positioning and navigation by an AUV," in *Proc. OCEANS'98*, vol. 2, Nice, Italy, Sept. 1998, pp. 752–756.

[16] Y. Bar-Shalom and T. E. Fortmann, "Tracking and data association," in *Mathematics in Science and Engineering*. New York: Academic , 1988, vol. 179.

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

[18] ——, "Real-time obstacle avoidance for manipulators and mobile robots," *Int. J. Robot. Res.*, vol. 5, pp. 90–98, 1986.

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

[20] J. Gil de Lamadrid and J. Zimmerman, "Avoidance of obstacles with unknown trajectories: Locally optimal paths and path complexity—Part I," *Robotica*, vol. 11, pp. 299–308, 1993.

[21] ——, "Avoidance of obstacles with unknown trajectories: Locally optimal paths and path complexity—Part II," *Robotica*, vol. 11, pp. 403–412, 1993.

Yvan Petillot was born in Saint-Etienne, France. He received the École Nationale Superieure des Telecommunications de Bretagne (ENSTE) degree, and the Ph.D degree, both from the University de Bretagne Occidentale, France, in 1991, and 1997, respectively.

From 1997 to 2000, he was a Research Associate at the Heriott Watt University, U. K., and worked in the field of sonar image processing and motion planning. His current research interests are in sonar image processing for classification, oceanographical systems, and multi-sensor function.

Ioseba Tena Ruiz was born in La Laguna, Santa Cruz de Tenerife, Spain. He received the Bachelor's degree in electronic engineering from the Heriott Watt University, U. K., in 1996, and is working toward the Ph.D. degree at the same University.

He is also a Research Associate at the Heriott Watt University. His research interests are in processing sonar returns, classifying sonar returns, concurrent mapping, and localization for autonomous underwater-vehicles and obstacle avoidance and navigation.

David M. Lane (M'98), photograph and biography not available at the time of publication.