Mosaicing of 3D Sonar Data Sets - Techniques and Applications

Rolf Kahrs Hansen
CodaOctopus OmniTech AS, Norway

Umberto Castellani, Vittorio Murino, Andrea Fusiello
Department of Computer Science, University of Verona, Italy

Enrico Puppo, Laura Papaleo
Department of Computer and Information Sciences, University of Genova, Italy

Massimiliano Pittore
e-magine IT srl, Italy

Marco Gobbi, Luigi Bisone
Centro Elettrotecnico Sperimentale Italiano, Italy

Kurt Kleppe
Fugro Survey Norge AS, Norway

Martin Hall
General Robotics Ltd., UK

Abstract - **This paper presents a system for registration of 3D sonar data sets based on the Echoscope 1600 3D sonar. Although a 3D sonar image provides instantaneous volumetric data sets, many applications require the data sets to be mosaic'ed into larger data sets. The lack of navigation and motion sensors being able to produce mosaics with accuracies comparable to the 3D sonar resolution necessitates the use of registration techniques. Techniques for filtering of 3D sonar data, mesh generation, registration and geometric fusion applied to both on-line and offline mosaicing will be discussed.**

On-line mosaicing is done at a rate of 5 to 10 data sets per second, a rate that is acceptable in most practical situations.

The techniques will find a number of applications, such as scanning harbors and seabed for potentially dangerous objects as well as seabed surveys. As the registration provides the sensor position relative to an object, object related positioning and navigation is also feasible, either as a stand alone underwater navigation system or as a system that improves current techniques like LBL, SBL and others.

I. INTRODUCTION

The 3D reconstruction of the object space is done by adopting an automatic modeling approach [1]. Very few systems address the reconstruction of the underwater environment and as far as we know, none of them are able to operate real-time [2]. Real-time operation is necessary in order to facilitate use as a navigation sensor. Kamgar-Parsi [3] proposed an acoustic lens technique for 3D data acquisition from which 3D models are recovered by using a standard volumetric approach. Negahdaripour [4] focused on computer vision techniques such as shape from stereo and video and 2D mosaicing from optical camera images. Several works regarding non-underwater applications have been described in the literature [1]. In particular, some examples of real time systems are proposed in [5] and [6]. Rusinkiewicz [5] introduced a complete model acquisition system that permits the user to see the update of the model as the object is scanned. The real time performance is achieved by the use of a simple and fast structured-light range scanner and by speeding up the performance of the registration phase. A similar approach is proposed by Koninckx et.al. [6]. The real time acquisition is guaranteed by adapting the code of the projected lines. The reconstruction is carried out in one-shot by allowing the modeling of deformable surfaces.

We have developed a data processing pipeline that starts with data acquisition and includes a network for data transmission, processing and display is a 3D graphic user interface. The following components are included:

a. Data capture. The acquisition of 3D sonar data.
b. Single frame reconstruction. The processing of a data set in order to obtain a triangle mesh.
c. Motion tracking. The sonar motion is estimated on the basis of alignment between successive data sets.
d. Geometric fusion. New data sets are merged into the 3D mosaic.

e. Data distribution. Data are distributed via Ethernet to various users, including an ROV pilot's interface and a surveyor's interface.
f. Rendering. Rendering and display of 3D data on a graphical user interface.

Some of these steps will be discussed in the following and some experiment data are included.

II. 3D SONAR DATA GENERATION

3D sonar data are obtained with a high-resolution 3D sonar [7]. The scene is insonified with a 50 by 50 degree beam using a 300 kHz pulse. Beam-forming is done using a 40 by 40 array of hydrophones where amplitude and phase are detected simultaneously on all 1600 channels. 64 by 64 beams are generated and the typical lateral resolution is 1.8 degree while the range resolution in these experiments is 5 cm. One echo is detected per beam based on a user defined threshold. Each echo is represented by beam number (i,j), range and intensity - which is a function of acoustic target strength. The typical update rate with the 1600-model is between 1 and 10 data sets per second.

The sonar data contain noise, side-lobes and multi-path reflections, hence filtering of the data is essential in order to form a basis for object reconstruction and mosaicing. Filtering has to be adapted to the application. For these experiments, we are interested in surfaces of and not individual objects like fish etc.

III. SINGLE FRAME RECONSTRUCTION

Acoustic range images are noisy and sometimes data points are missing even if a surface is detected. This can be due to roughness in the surface, variations in the surface target strength, noise as well as shadow effects from objects above or in front of the surface. In order to eliminate undesired holes in the final mesh, we extended the classical approach of triangle mesh reconstruction. The extended method tests for usual adjacencies and also adjacencies in a 3 x 3 window that surrounds each vacancy in the input. In order to avoid connecting points that have adjacent entries but lie far in 3D space (i.e. at different ranges), a potential adjacency generates an edge only if the radial distance of points is smaller than a given threshold.

Triangles are obtained by considering cycles of three edges. A topographic data structure is built on the fly which allows us traversing the mesh by triangle adjacencies in order to identify connected components in the mesh. Connected components having sizes smaller than a given threshold are filtered out. The output is a mesh of triangles with surface normals for each vertex. This is called a single frame mesh.

IV. 3D REGISTRATION AND MOTION TRACKING

The proposed method is based on the alignment of sequential pair-wise frames. In particular, since we assume static scenarios (only the sonar can move) the transformation that brings the frame *N* to the frame *N+1* is given by the sonar motion. The global motion (i.e. the motion with respect to the global reference system) is computed by accumulating the pair-wise transformations. Reliable pair-wise registration allows us to:

- build 3D models of the environment,
- navigate relative to the environment.

The operations on the acoustic data are:

- **Accelerations techniques:** a fast Iterative Closest Point (Fast ICP or FICP) algorithm has been developed based on the characteristics of the acoustic data.
- **Accuracy techniques:** a robust method for outliers rejections has been introduced. Furthermore, some iterations of accurate ICP algorithm are carried out in order to improve the convergence of the fast registration.
- **Parameters definitions:** several parameters need to be adjusted in order to improve the quality of results. Some of them are empirically estimated from the already stored data, and some of them can be directly modified on-line by the pilot.

A. Iterative Closest Point Algorithm (ICP)

Besl and McKay [8] proposed the Iterative Closest Point algorithm as a general purpose method for registration of rigid 3D shapes.

Suppose we have two sets of 3D points corresponding to a single shape, but expressed in different reference frames. Call one of these sets *the model set X* and the other *the data set Y*. The problem is to find a 3D transformation that minimizes the distance between the two point sets. The problem can be stated as

$$\min_{R,t} \sum_{i=1}^{N} \left\| x_i - (R y_i + t) \right\|^2 \qquad (4.1)$$

where

$$G_a = \begin{bmatrix} R & t \\ 0 & 1 \end{bmatrix} \qquad (4.2)$$

R is a 3 x 3 rotation matrix and t is a 3 x 1 translation vector. G_a is thus the 4 x 4 motion matrix. i refers to corresponding elements of the sets X and Y. Efficient non-iterative solutions to this problem were compared in [9]

and the one based on Singular Value Decomposition was found to be the best.

Suppose we have two sets, X and Y, corresponding to a single shape and that the transformation between X and Y is unknown. For each point y_i in the set Y, there exists at least one point in X that is closer to y_i than all the other points in X. This is the closest point x_i. The basic idea behind the ICP is that under certain conditions, the point correspondences provided by sets of closest points are reasonable approximations for the true point correspondences. Besl and MacKay proved that, if the process of finding the closest-point sets and the solving equation (4.1) is iterated, the solution is guaranteed to converge to a local minimum.

As the ICP is only guaranteed to convert to a local minimum and not the global minimum, the starting point of the iterative process is critical. We also need to make sure that the two data sets that are to be compared shows the same features, i.e. we need a good estimate of the overlapping region. Hence we want to make use of all a priori information as possible, as

- orientation sensors in the sonar itself,
- motion sensors available on the sonar platform (such as ROV gyros, LBL systems etc.),
- depth sensors,
- altitude sensors,
- doppler velocity logs,
- and least but not last, the history of motion between successive data sets (and hence the motion of the sonar platform).

Only some of these will typically be available. This calls for a flexible approach with the ability to weight the importance of different sensor inputs and to predict the next position of the sensor. In order to achieve this we use a Kalman filter.

The fast version of the ICP takes the data geometry into consideration so that the amount of data points taking part in the ICP is reduced as much as possible.

B. Kalman Filter

The Kalman filter addresses the general problem of trying to estimate the state $x \in R^n$ of a discrete-time controlled process that is governed by the linear stochastic difference equations:

$$x_k = Ax_{k-1} + Bu_{k-1} + w_{k-1} \tag{4.3}$$

with a measurement $z \in R^m$ that is

$$z_k = Hx_k + v_k . \tag{4.4}$$

The random variables w_k and v_k represent the process and measurement noise (respectively). They are assumed to be independent (of each other), white, and with normal probability distributions

$$p(w) \sim N(0, Q), \tag{4.5}$$

$$p(v) \sim N(0, R) . \tag{4.6}$$

The process noise covariance Q and measurement noise covariance R matrices are assumed to be constant.

The $n \times n$ matrix A in the difference equation (4.3) relates the state at the previous time step k - 1 to the state at the current step k, in the absence of either a driving function or process noise. Note that in practice A might change with each time step, but here we assume it is constant. The $n \times l$ matrix B relates the optional control input $u \in R^l$ to the state x. The $m \times n$ matrix H in the measurement (4.2) relates the state to the measurement z_k. In practice H might change with each time step or measurement, but here we assume it is constant.

1) Notation

We define $\hat{x}^- \in R^n$ (note the "super minus") to be our a priori state estimate at step k given knowledge of the process prior to step k, and $\hat{x} \in R^n$ to be our a posteriori state estimate at step k given measurement z_k. We can then define a priori and a posteriori estimate errors as

$$e_k^- \equiv x_k - \hat{x}_k^- \tag{4.7}$$

$$e_k \equiv x_k - \hat{x}_k \tag{4.8}$$

The a priori estimate error covariance is then

$$P_k^- = E\left[e_k^- e_k^{-T}\right] \tag{4.9}$$

and the a posteriori estimate error covariance is

$$P_k = E\left[e_k e_k^T\right] \tag{4.10}$$

In deriving the equations for the Kalman filter, we begin with the goal of finding an equation that computes an a posteriori state estimate $\hat{x}_k$ as a linear combination of an a priori estimate $\hat{x}_k^-$ and a weighted difference between an actual measurement z_k and a measurement prediction $H\hat{x}_k^-$ as shown below in (4.11).

$$\hat{x}_k = \hat{x}_k^- + K\left(z_k - H\hat{x}_k^-\right) \tag{4.11}$$

The difference $\left(z_k - H\hat{x}_k^-\right)$ in (4.11) is called the measurement innovation, or the residual. The residual

reflects the discrepancy between the predicted measurement $H\hat{x}_k^-$ and the actual measurement z_k. A residual of zero means that the two are in complete agreement.

The $n \times m$ matrix K in (4.11) is chosen to be the gain or blending factor that minimizes the a posteriori error covariance (4.10). This minimization can be accomplished by first substituting (4.11) into the above definition for e_k, substituting that into (4.10), performing the indicated expectations, taking the derivative of the trace of the result with respect to K, setting that result equal to zero, and then solving for K. One form of the resulting K that minimizes (4.10) is given by

$$K_k = P_k^- H^T \left(H P_k^- H^T + R \right)^{-1} \qquad (4.12)$$

From (4.12) we see that as the measurement error covariance R approaches zero, the gain K weights the residual more heavily.

Specifically,

$$\lim_{R \to 0} K_k = H^{-1} \qquad (4.13)$$

On the other hand, as the a priori estimate error covariance P_k^- approaches zero, the gain K weights the residual less heavily. Specifically,

$$\lim_{P_k^- \to 0} K_k = 0 \qquad (4.14)$$

As the measurement error covariance R approaches zero, the actual measurement z_k is "trusted" more and more, while the predicted measurement $H\hat{x}_k^-$ is trusted less and less. On the other hand, as the a priori estimate error covariance P_k^- approaches zero the actual measurement z_k is trusted less and less, while the a priori estimate $\hat{x}_k^-$ is trusted more and more.

2) The Probabilistic aspect of the Filter

The justification for (4.11) is based on the probability of the a priori estimate $\hat{x}_k^-$ conditioned on all prior measurements z_k (Bayes' rule). The Kalman filter maintains the first two moments of the state distribution,

$$E[x_k] = \hat{x}_k \qquad (4.15)$$

$$E\left[(x_k - \hat{x}_k)(x_k - \hat{x}_k)^T \right] = P_k \qquad (4.16)$$

The a posteriori state estimate (4.11) reflects the mean (the first moment) of the state distribution - it is normally distributed if the conditions of (4.5) and (4.6) are met. The a posteriori estimate error covariance (4.10) reflects the variance of the state distribution (the second non-central moment). In other words,

$$\begin{aligned} P &\geq (x_k \mid z_k) \\ &\sim N\left(E[x_k], E\left[(x_k - \hat{x}_k)(x_k - \hat{x}_k)^T \right] \right) \\ &= N(\hat{x}_k, P_k) \end{aligned} \qquad (4.17)$$

3) The Discrete Kalman Filter Algorithm

The Kalman filter estimates a process by using feedback control. The filter estimates the process state at some time and then obtains feedback in the form of (noisy) measurements. As such, the equations for the Kalman filter fall into two groups: time update equations and measurement update equations. The time update equations are responsible for projecting forward (in time) the current state and error covariance estimates to obtain the a priori estimates for the next time step. The measurement update equations are responsible for the feedback - i.e. for incorporating a new measurement into the a priori estimate to obtain an improved a posteriori estimate.

The time update equations can also be thought of as predictor equations, while the measurement update equations can be thought of as corrector equations. Indeed the final estimation algorithm resembles that of a predictor-corrector algorithm for solving numerical problems as shown below in Figure 1.

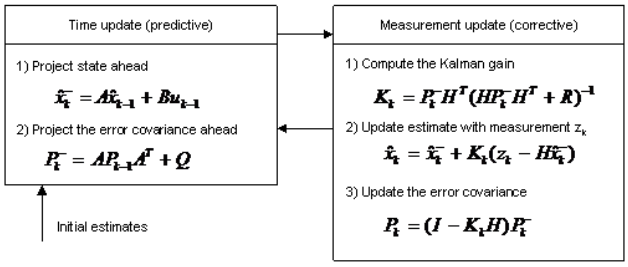

Figure 1. Operation of the Kalman filter .

Note how the time update equations in Figure 1 projects the state and covariance estimates forward from time step $k-1$ to step k. A and B are from (4.3), while Q is from (4.5). Initial conditions for the filter are discussed in the references.

The first task during the measurement update is to compute the Kalman gain, K_k. Notice that the equation given here is the same as (4.3). The next step is to actually measure the process to obtain z_k, and then to generate an a posteriori state estimate by incorporating the measurement.

The final step is to obtain an a posteriori error covariance estimate.

After each time and measurement update pair, the process is repeated with the previous a posteriori estimates used to project or predict the new a priori estimates.

4) Filter Parameters and Tuning

In the actual implementation of the filter, the measurement noise covariance R is usually measured prior to operation of the filter. Measuring the measurement error covariance R is generally practical (possible) because we need to be able to measure the process anyway (while operating the filter) so we should generally be able to take some off-line sample measurements in order to determine the variance of the measurement noise.

The determination of the process noise covariance Q is generally more difficult as we typically do not have the ability to directly observe the process we are estimating. Sometimes a relatively simple (poor) process model can produce acceptable results if one "injects" enough uncertainty into the process via the selection of Q.

In either case, whether or not we have a rational basis for choosing the parameters, often times superior filter performance (statistically speaking) can be obtained by tuning the filter parameters Q and R. The tuning is usually performed off-line, frequently with the help of another (distinct) Kalman filter in a process generally referred to as system identification.

We note that under conditions where Q and R are in fact constant, both the estimation error covariance P_k and the Kalman gain K_k will stabilize quickly and then remain constant (see the filter update equations in Figure 1). If this is the case, these parameters can be pre-computed by either running the filter off-line, or for example by determining the steady-state value of P_k.

It is frequently the case however, that the measurement error (in particular) does not remain constant. Also, the process noise Q is sometimes changed dynamically during filter operation - becoming Q_k - in order to adjust to different dynamics. In such cases Q_k might be chosen to account for both uncertainty about the user's intentions and uncertainty in the model.

5) The Implementation

The complete process of the operations of the Kalman filter is described in Figure 1. According to this picture, it is clear that the filtering algorithm has three procedures: initialization, time update and measurement update, while the time update and the measurement update are defined in five equations. However, in the initialization, several matrixes and vectors, such state vector, process noise covariance, etc. must be defined.

The state: In (4.3), $\boldsymbol{x}$ is used to describe the state of objects processed in Kalman filter. In the case of sonar motion, the state the sonar can be defined as follows:

Translation:
- position: x, y, z
- speed: v_x, v_y, v_z
- acceleration: a_x, a_y, a_z

Rotation (Euler angles):
- position: R_x, R_y, R_z
- speed: v_{rx}, v_{ry}, v_{rz}
- acceleration: a_{rx}, a_{ry}, a_{rz}

Thus, the state vector $\boldsymbol{x}$ is:

$$\boldsymbol{x} = [x, y, z, v_x, v_y, v_z, a_x, a_y, a_z, R_x, R_y, R_z, v_{rx}, v_{ry}, v_{rz}, a_{rx}, a_{ry}, a_{rz}]^T$$

The motion model: in (4.3), matrix A is used to predict the state at time k from time k - 1. Thus, matrix A represents the motion model of the sonar. In general the sonar motion can be modeled as a constant acceleration movement. For example, the motion model on the sonar translation on X axis is:

$$x_{k+1} = x_k + v_k t + \frac{1}{2} a_k t^2 \tag{4.18}$$

where x is the position of the object on the X axis, v is the speed and a is the acceleration, which is a constant value. In the case of rotation, the similar model for the motion angles is:

$$r_{k+1} = r_k + v_k t + a_k t^2 \tag{4.19}$$

where, r_k is always set to zero in case of Euler angles.

The process noise covariance Q: as it is mentioned above that to determine the process noise covariance Q is generally more difficult as we typically do not have the ability to directly observe the process we are estimating. Thus, Q matrix used in our filtering algorithm is defined by Y. Bar-Shalom and T. E. Fortmann [10].

V. NAVIGATION AND MOTION TRACKING EXPERIMENTS

In order to evaluate the capability of the proposed system in tracking the sonar motion, a comparison of the tracking performance obtained with different settings has been carried out. Figure 2 shows an example of the estimated trajectory for a portion of a quayside in Bergen, Norway. Three lines are displayed. The red line shows the trajectory estimated by using only the measures coming from a Long BaseLine (LBL) system. Note that the LBL system was used under difficult reverberant conditions. This can be seen as jumps in the positions. The blue line shows the trajectory estimated by using only the FICP and

discarding the LBL data completely. The trajectory is smoother but, due to the accumulation error, the positions observed at the final steps deviate from the positions observed form the motion sensors. Finally, the trajectory obtained by merging the two measurements (in green) is more reliable.

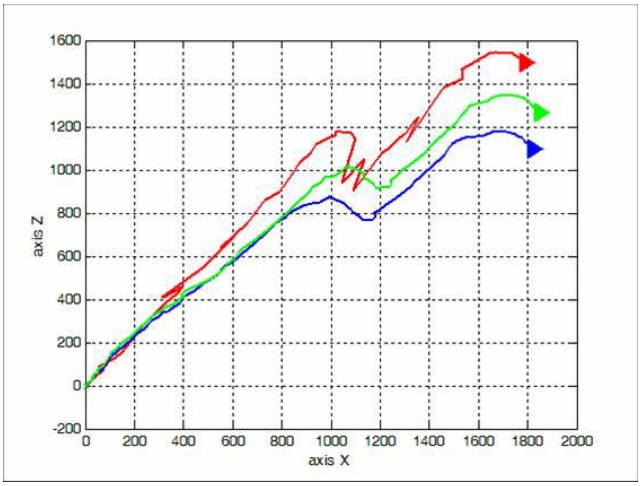

Figure 2 Navigation experiment. FICP based (blue) compared with LBL (red) and the two combined (green). Axes in centimeters.

Another experiment - station keeping - tested the ability to provide stable position data while maintaining an ROV in a stable position relative to a 3 m by 2 m by 2 m object on the seabed. The distance to the object was about 5 m. Figures 4 to 6 show the results with ICP based data compared to LBL based data. Easting, northing and depth are shown. Again the jumps in LBL data are due to reverberation, but the ICP based data are very stable and the noise is in fact less then the basic resolution of the 3D sonar (5 cm).

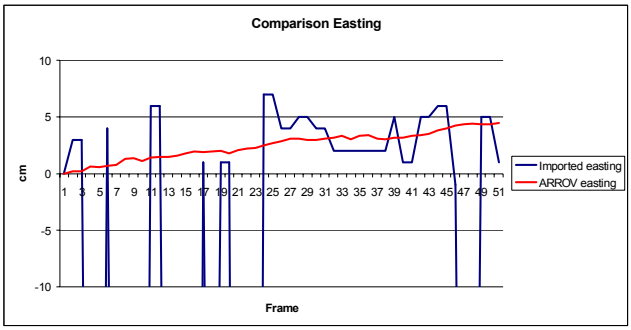

Figure 3. Easting. The red line shows the data generated by the ICP and the blue line is from the LBL system.

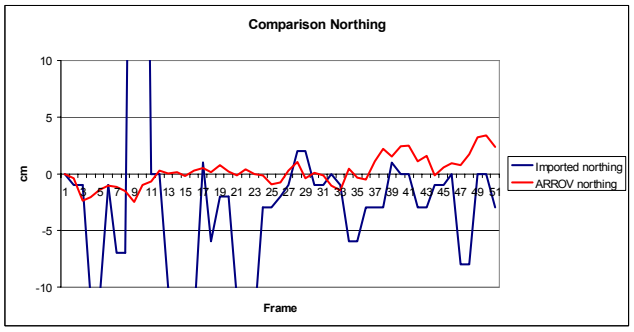

Figure 4. Northing. The red line shows the data generated by the ICP and the blue line is from the LBL system.

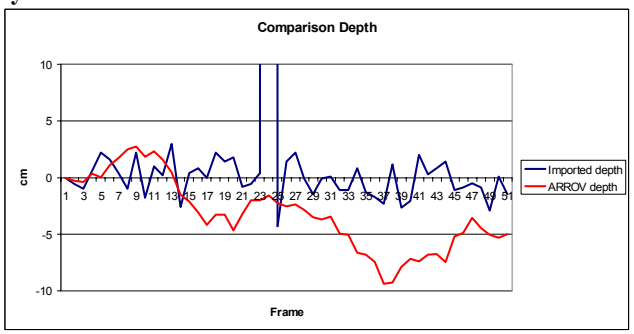

Figure 5. Depth. The red line shows the data generated by the ICP and the blue line is from the LBL system.

VI. GEOMETRIC FUSION

Different single frame meshes registered in a common coordinate system contribute to form the 3D mosaic. Such mosaic must be built and visualized on-line, while frames are acquired from the sensor. See Castellani et.al. ([11] and [12]) for a detailed discussion of this process. In short, registration is performed as described above and active cells are updated online. By active cells we mean new cells and cells that need to be modified.

The geometric fusion technique is particularly useful for seabed surveys. The data shown below comes from a survey at Elsesro near Bergen, Norway. The survey was carried out with the Echoscope pole mounted to a surface vessel using GPS for navigation, along with pitch and roll sensors internal in the Echoscope. The area of interest contains many ship wrecks, there are three wrecks stacked on top of each others in one location.

The water depth was about 10 meters and we used 300 kHz frequency and a 50 by 50 degree coverage. A typical 3D snapshot is shown in figure 6. The motion sensor uncertainties translate to error in the position of the data sets. As we transform all data to a global frame, two consecutive data sets should be perfectly aligned in the same global position and orientation if we had no motion data errors. But as figure 7 shows, the misalignment is more than 1 meter. If we would perform terrain gridding based on these data, all details would be smeared out and useful information would disappear.

The use of FICP enables us to align the data sets online. Figures 8 and 9 show the results.

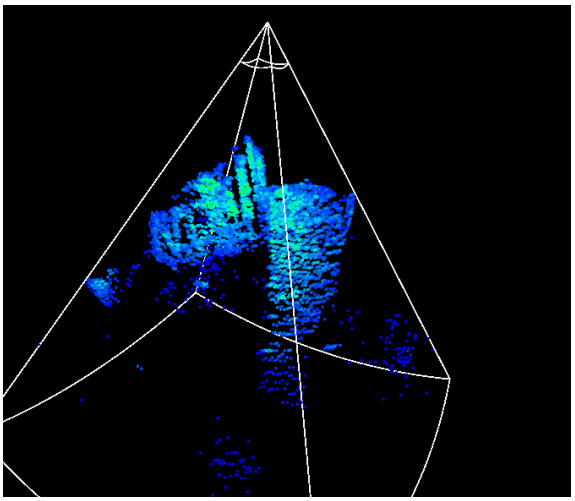

Figure 6. Snapshot of an Echoscope 3D data set. 11 m water depth, 300 kHz frequency, 50 by 50 degree opening angle. Part of ship wreck is shown.

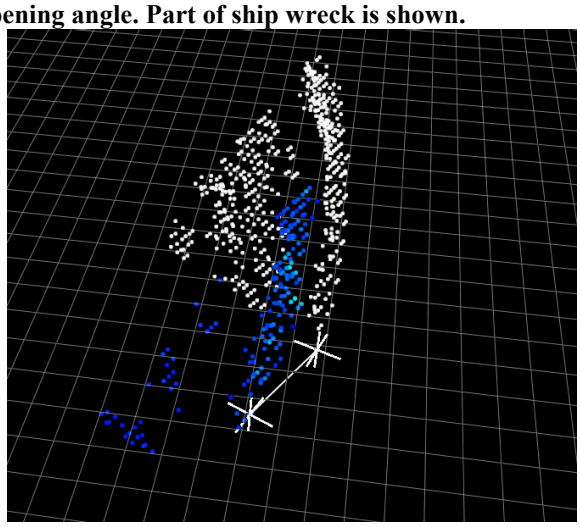

Figure 7. Misalignment between consecutive data sets. The detail here has moved 1.4 m between two data sets.

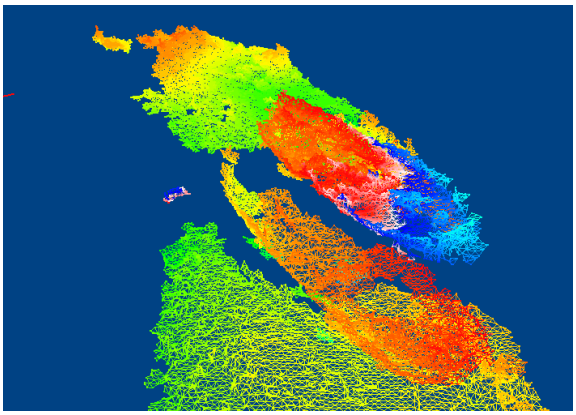

Figure 8. After mosaicing the data using ICP to correct for misalignment. Three ship wrecks and seabed.

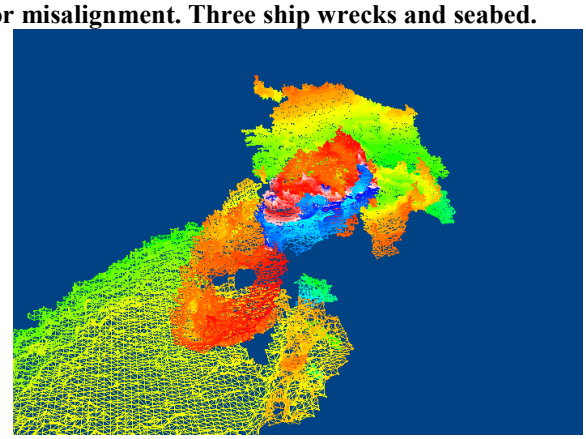

Figure 9 Same data as figure 8, but seen from aft of the wreck.

A. Speed and Accuracy

A sequence with 10 data sets has been analyzed with both classic ICP and our fast ICP. The results are given in table 1.

Table 1. Speed and accuracy of classical vs. Fast ICP.

Data sets	Classic ICP		Fast ICP	
	Time *[sec]*	Accuracy *[cm]*	Time *[sec]*	Accuracy *[cm]*
2 to 1	7.8	19.7	0.18	23.5
3 to 2	12.3	21.5	0.12	25.0
4 to 3	14.1	17.5	0.12	22.3
5 to 4	16.7	18.2	0.11	25.4
6 to 5	23.6	17.4	0.11	28.4
7 to 6	16.3	22.5	0.44	24.1
8 to 7	12.9	21.6	0.27	25.3
9 to 8	13.8	21.6	0.42	21.2
10 to 9	10.16	22.6	0.34	19.5
Mean	**14.2**	**20.3**	**0.23**	**23.8**

As table 1 shows, the FICP is considerable faster than the conventional ICP while maintaining the overall accuracy. Accuracy means average distance between corresponding points.

VIII. CONCLUSION

We have presented a method for real-time alignment of 3D sonar data that allows us to use the 3D real time sonar for object related navigation and positioning as well as for production of 3D survey data. The fast ICP that was developed includes a Kalman filter where motion data from external sensors can be included and be weighted according to their quality. The resulting mesh is updated using a lazy update technique so that the load successive transmission bandwidth and processing is reduced to a minimum.

Real world trials demonstrate the implementation of the methods.

Acknowledgments

This work was supported by the European Commission under project G3RD-CT2000-00285. The implementation of the ICP is partially due to Linmin Tao.

References

[1] R. Hartley and A. Zisserman. *Multiple view geometry in computer vision*. Cambridge University Press, 2000.

[2] H. Singh, X. Tang, E. Trucco and D. Lane. Guest editorial special issue on underwater image and video processing. *IEEE Transactions on Oceanic Engineering,* 28(4):569-776, 2003.

[3] B. Kamgar-Parsi, L.J. Rosenblum and E.O. Belcher. Underwater imaging with a moving acoustic lens. *IEEE Transactions on Image Processing*, 7(1):91-99,1998.

[4] S. Negahdaripour and H. Madjidi. Stereo imaging on submersible platforms of 3-D mapping of benthic habitats and sea-floor structures. *IEEE Transactions on Oceanic Engineering*, 28(4):625-650,2003.

[5] S. Rusinkiewicz, O. Hall-Holt and M. Levoy. Real-time 3D model acquisition. In *Proceedings of Siggraph (SIGGRAPH 2002)*, pp. 438-446, 2002.

[6] T.P. Koninckx, A. Griesser and L. Van Gool. Real-time range scanning of deformable surfaces by adaptively coded structured light. In *Proceedings of 3-D Digital Imaging and Modelling (3DOM 2003)*, pp. 203-300, 2003.

[7] R. K. Hansen and P. A. Andersen. A 3-D underwater Acoustic Camera - Properties and Applications, *Acoustical Imaging*, ed. P.Tortoli and L.Masotti, pp. 607-611, Plenum Press 1996.

[8] P. Besl and N. McKay. A method for registration of 3-D shapes. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 14(2):239-256, February 1992.

[9] A. Lorusso, D.W.Eggert and R.B. Fisher. "A comparison of four algorithms for estimating 3-D rigid transformations," *Machine Vision and Applications*, Vol. 9. pp. 272-290, 1997.

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

[11] U. Castellani, A. Fusiello and V. Murino. Registration of multiple acoustic range views for underwater scene reconstruction. *Computer Vision and Image Understanding*, 87(3); 78-79, July 2002.

[12] U. Castellani, A. Fusiello and V. Murino. Underwater environment modeling by fast 3D mosaicing. In *Model-based Imaging, Rendering, Image Analysis and Graphical special Effects (MIRAGE-2005)*, Rocquencourt, France, March 2005.