# Triangulation based Fusion of Ultrasonic Sensor Data

O. Wijk
S3 Automatic Control
Kungliga Tekniska Högskolan
SE-100 44 Stockholm, Sweden

P. Jensfelt
S3 Automatic Control
Kungliga Tekniska Högskolan
SE-100 44 Stockholm, Sweden

H.I Christensen
Autonomous Systems
Kungliga Tekniska Högskolan
SE-100 44 Stockholm, Sweden

**Abstract**

*Ultrasonic sensors are still one of the most widely used sensors in mobile robotics. A notorious problem in the use of sonar data is the lack of good spatial resolution, which typically results in a high uncertainty in the resulting map of the environment. In this paper a triangulation technique is used for filtering of data so as to obtain an improved grid map of the environment. The basic technique is described and it is outlined how it can be used for identification of natural landmarks.*

## 1 Introduction

Almost all mobile platform are equipped with ultrasonic sensors (sonars) that are used for local mapping of the environment. Examples of such work is illustrated in [1, 2, 3, 4, 5, 6].

It is however characteristic for much of this work that the limited spatial resolution of the sonars results in 'fuzzy' maps, that require a large number of sonar readings or significant post-processing.

In this paper a filtering technique that enable robust estimation of features in the environment is presented. The technique relies on triangulation of sonar readings recorded from different positions for estimation of the location of structures in the environment. The triangulation technique provides a fast method for estimation of structures. When combined with a voting method the result is an improved spatial resolution and a more robust map of the environment.

Initially the physics of the sensors is briefly reviewed, the underlying assumptions are explained together with the terminology used. The basic technique is then developed in section 3 and 4. Section 5 then describes how we can apply the method when building grid maps. Finally it is explained how the technique can be used for robust estimation of features in section 6.

## 2 The sensor

The ultrasonic sensor transmits a short ultrasonic pulse, and the time for return is then measured. Based on knowledge of the transmission speed of sound in air ($\approx 330 m/s$) and the time of flight it is trivial to compute the distance to the target that reflected the pulse. The sound pulse does not propagate along a straight line, but is propagated within a cone. The most frequently used sonar sensor, the Polaroid environmental transducer, has a beam width of 12° (the half beam width). This propagation pattern must be taken into account when the sensor is used for mapping of the environment, as it is impossible from a single sensor reading to determine the position of the target within the beam.

**Assumptions** Throughout the rest of the paper the following assumptions will be made: (i) when the robot is within a small radius the same structure in the environment will give rise to the reflection (i.e., only a single major target returns the signal), (ii) the size of the target is small (i.e., it is the same point on the target that returns the signal), (iii) the beam of the sonar is assumed to be a conic. Only a two dimensional map of the environment is constructed and it is therefore assumed that the shape of the beam is a circular arc (of 24°).

**Terminology** It is assumed that the mobile platform has $m$ sonars that are distributed on the robot at known locations. We use the following terminology in the algorithms and the derivation of the underlying equations.

$T$ Estimated target data represented as $T = \{(x_T, y_T), n_t, (x_s, y_s)\}$, where $(x_T, y_T)$ is the estimated location of the target, $n_t$ is the number of triangulations that have been done to obtain $(x_T, y_T)$ and $(x_s, y_s)$ is one of the $n_t + 1$ sensors positions that have been used in the triangulation of the target.

$R_i$ Reading from sonar $i$ represented as $R_i = (x_{s_i}, y_{s_i}, \gamma_i, r_i)$ where $\gamma$ is the direction of the sonar with respect to the x-axis of a local coordinate system, and $r$ is the range reading.

## 3 Triangulation

Given the assumption that the same target will return a signal if two sonars are fired from almost the same position, it is possible to estimate the target location using simple triangulation. Given two sensors readings recorded from different positions, a situation as shown in figure 1 arise.

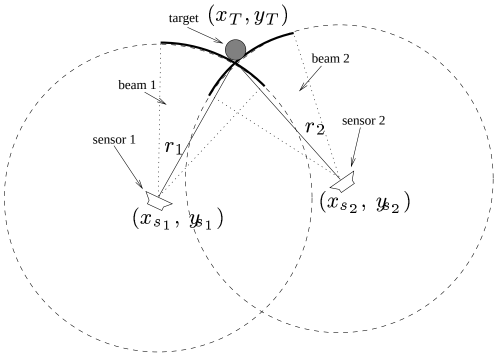


Figure 1: The idea behind triangulation. The intersection between the arcs specify the location of the common target.

The intersection between the arcs defined by the readings specify the location of the common target. This intersection can be found as the solution of the following system of equations:

$$
\begin{aligned}
(x_T - x_{s_1})^2 + (y_T - y_{s_1})^2 &= {r_1}^2 \\
(x_T - x_{s_2})^2 + (y_T - y_{s_2})^2 &= {r_2}^2 \\
(x_T, y_T) \in \text{beam 1,2}
\end{aligned}
$$

The solution (including false roots) can be derived as:

$$
\begin{aligned}
y_T &= y_{s_1} + \frac{1}{c^2}(bd \pm \mid a \mid \sqrt{r_1^2 c^2 - d^2}) \\
x_T &= x_{s_1} \pm \sqrt{r_1^2 - (y_T - y_{s_1})^2}
\end{aligned}
$$

where
$$
\begin{aligned}
a &= x_{s_1} - x_{s_2} \\
b &= y_{s_1} - y_{s_2} \\
c &= \sqrt{a^2 + b^2} \\
d &= \frac{r_2^2 - r_1^2 - c^2}{2}
\end{aligned}
$$

In the algorithm described in this paper the first solution that satisfies all three equations is chosen.

## 4 Hypothesis generation and verification

The equations allow estimation of the position of a single target. It is, however, well known that sonar readings are noisy and it is consequently of interest to be able to integrate the resulting estimate into a hypothesis verification scheme.

This can easily be achieved if multiple sensors readings from several positions are integrated. To achieve this, a time window of sonar readings is maintained. The time window of readings is shown in figure 2. Each column contains a time instance of readings from all $m$ sonars. A new set of readings is only inserted into the table when the robot has moved a certain minimum distance (we have used 5 cm in our experiments). The time window contains a total of $n$ samples. The oldest samples are in the first column. In practical terms the window is implemented as a circular buffer with $n$ entries.

| | | | | | | | |
|---|---|---|---|---|---|---|---|
| Sonar 1 | $R_{11}$ | $R_{12}$ | $R_{13}$ | $\cdots$ | $R_{1,n-2}$ | $R_{1,n-1}$ | $R_{1n}$ |
| Sonar 2 | $R_{21}$ | $R_{22}$ | $R_{23}$ | $\cdots$ | $R_{2,n-2}$ | $R_{2,n-1}$ | $R_{2n}$ |
| Sonar 3 | $R_{31}$ | $R_{32}$ | $R_{33}$ | $\cdots$ | $R_{3,n-2}$ | $R_{3,n-1}$ | $R_{3n}$ |
| Sonar 4 | $R_{41}$ | $R_{42}$ | $R_{43}$ | $\cdots$ | $R_{4,n-2}$ | $R_{4,n-1}$ | $R_{4n}$ |
| Sonar 5 | $R_{51}$ | $R_{52}$ | $R_{53}$ | $\cdots$ | $R_{5,n-2}$ | $R_{5,n-1}$ | $R_{5n}$ |
| | $\vdots$ | $\vdots$ | $\vdots$ | $\ddots$ | $\vdots$ | $\vdots$ | $\vdots$ |
| Sonar m-2 | $R_{m-2,1}$ | $R_{m-2,2}$ | $R_{m-2,3}$ | $\cdots$ | $R_{m-2,n-2}$ | $R_{m-2,n-1}$ | $R_{m-2,n}$ |
| Sonar m-1 | $R_{m-1,1}$ | $R_{m-1,2}$ | $R_{m-1,3}$ | $\cdots$ | $R_{m-1,n-2}$ | $R_{m-1,n-1}$ | $R_{m-1,n}$ |
| Sonar m | $R_{m1}$ | $R_{m2}$ | $R_{m3}$ | $\cdots$ | $R_{m,n-2}$ | $R_{m,n-1}$ | $R_{m,n}$ |

time $\longrightarrow$

Figure 2: Temporal window for storage of sonar readings. A new set of readings is only inserted into the table when the robot has moved a certain minimum distance.

Using the temporal window in combination with a simple voting scheme, it is possible to generate a set of stable target hypotheses. For each hypothesis we store the location and the number of votes received.

The hypothesis generation and verification is shown in the algorithm below

$$
\begin{array}{lr}
\textit{for } i = 1 \rightarrow m & (1) \\
\quad n_t = 0 & (2) \\
\quad \bar{x}_T = x_{s_{in}} + r_{in} \cdot \cos(\gamma_{in}) & (2) \\
\quad \bar{y}_T = y_{s_{in}} + r_{in} \cdot \sin(\gamma_{in}) & (2) \\
\quad \textit{for } j = (n-1) \rightarrow 1 & (3) \\
\qquad \textit{for } k = (i-2) \rightarrow (i+2) & (3) \\
\qquad \textit{if } (\bar{x}_T, \bar{y}_T) \in \text{beam of } R_{kj} & (4) \\
\qquad\quad r_e = \sqrt{(\bar{x}_T - x_{s_{kj}})^2 + (\bar{y}_T - y_{s_{kj}})^2} & (4) \\
\qquad\quad \textit{if } \mid r_e - r_{kj} \mid < \textit{tol} & (4) \\
\qquad\qquad (x_T^{\text{cand}}, y_T^{\text{cand}}) = \text{triang}(R_{in}, R_{kj}) & (5) \\
\qquad\qquad \text{if } (x_T^{\text{cand}}, y_T^{\text{cand}}) \in \text{beams of } R_{in} \;\&\; R_{kj} & (5) \\
\qquad\qquad\quad \bar{x}_T = \frac{1}{n_t+1}(n_t \bar{x}_T + x_T^{\text{cand}}) & (6) \\
\qquad\qquad\quad \bar{y}_T = \frac{1}{n_t+1}(n_t \bar{y}_T + y_T^{\text{cand}}) & (6) \\
\qquad\qquad\quad n_t = n_t + 1 & (6) \\
\quad \textit{if } n_t \geq 2 & (7) \\
\qquad \text{Store } T_i = \{(\bar{x}_T, \bar{y}_T), n_t, (x_{s_{in}}, y_{s_{in}})\} & (7)
\end{array}
$$

The algorithm operates as follows: (1) For all sensors, (2) initialize a new hypothesis at the centre of the beam with count 0, (3) loop over the part of the table that can generate stable hypotheses (example: if for instance $i = 0$ in the algorithm, the interesting part of the table is the shaded area in figure 2), (4) if the hypothesis is within the beam corresponding to reading $R_{kj}$ and the *expected* range reading $r_e$ does not differ to much from the actual range reading $r_{kj}$, (5) triangulate the measurements and check if the target location is inside the beams of the readings $R_{in}$ and $R_{kj}$, (6) if so, update the hypothesis (recursive mean) and increment the hypothesis count (see also figure 3), (7) finally store the hypothesis that received multiple votes.
It should be pointed out that the *tol* variable occurring in the code is dependent on $n_t$. It is more generous when we have a 0-hypothesis, and then it decreases as $n_t$ increases. The algorithm is efficient as it only involves less than 10 operations for the check of beam membership, and less than 100 operations for the triangulation.

## 5 Map building

The algorithm in section 4 has been evaluated for map generation for a robot. The mapping of the environment uses an occupancy grid [2]. In the original formulation by Elfes, the environment is tessellated into a Cartesian grid, where each cell contains a number between 0 and 1. Zero indicates that the grid location is free, while a one indicates that the grid location is occupied. Many have used this framework for map generation, where a probabilistic model for the sonars is used for updating of the map, see for example [7, 2, 8].

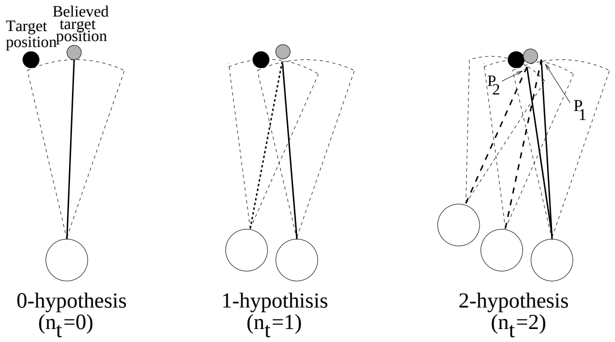


Figure 3: The idea of fusing sonar measurements which are likely to have come from the same target. A 0-hypothesis (*left*) is just a way to get started and is replaced if we can find a sonar measurement to create a 1-hypothesis (*middle*). A 2-hypothesis (*right*) can be created from three sonar measurements by taking the mean position from two triangulations ($\frac{P_1+P_2}{2}$).

We have chosen to use a modified version of the occupancy grid. The map is updated using the following rules:

1. The map is initialized to 0.

2. When a new hypothesis $T_i$ is generated by the algorithm in section 4, the corresponding cell $(\bar{x}_T, \bar{y}_T)$ is set to $n_t$. The number $n_t$ is obviously a measure of the belief that the cell is occupied. If the corresponding cell already holds a value larger than 0, the larger of the new and the already stored value is used.

3. Cells on the line from the target $(\bar{x}_T, \bar{y}_T)$ to the sensor position $(x_{s_{in}}, y_{s_{in}})$ supporting the target hypothesis are set to zero.

This technique, at least in its present state, is best suited for static environment. Moving objects does not accumulate enough readings to build a strong hypothesis. This implies that moving objects will be filtered out from the map, making the extraction of static landmarks easier. Landmarks can be detected in the map as cells holding clusters of large values.

## 6 Evaluation

To evaluate the method for map construction, we have implemented the algorithm on a Nomad 200. The robot has 16 sonars, distributed evenly around the top of the robot. The temporal window size is therefore $16 \times n$. In our experiments we used $n = 7$, but the number could be changed in response to the environment being mapped. The technique was initially evaluated in a corridor setting. A picture of the corridor is shown in figure 4.

We have chosen to compare the algorithm to the widely used algorithm by Borenstein [8, 3]. The raw sensors readings are shown in figure 5. The results obtained with the Borenstein algorithm is shown in figure 6. Finally the results obtained with the technique presented in this paper is shown in figure 7. In the grid maps we have used a cell size of 5 cm.

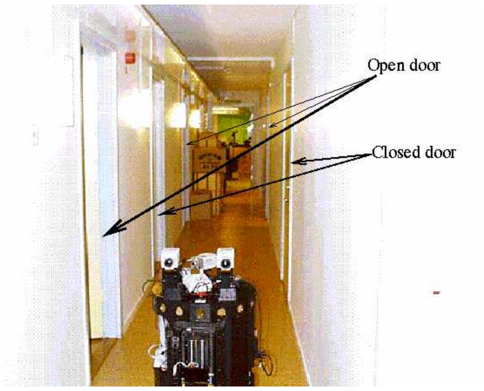


Figure 4: Picture of the corridor. During the experiment the robot followed the corridor passing five doors (two of them were closed) until it reached the boxes on the left side. The speed of the robot was 7 inches/sec.

From the raw sonar map it is obvious that the data contains a fair number of out-layers, and it is difficult to determine the precise location of the doorways. In the results obtained with the Borenstein algorithm it can clearly be seen that many of the outliers have been removed, but the positions of the door posts are still uncertain (fuzzy clusters of high confidence).

In the results with the proposed technique, it is apparent that more outliers have been removed and the location of the door-posts is more accurate. Each presented integer in the map corresponds to a $n_t$-value for a cell in the map.

For landmark detection the high values in the map can be used. Figure 8 shows the highest values in the

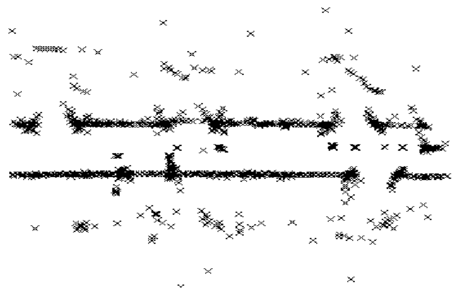

Figure 5: Raw data from the corridor run. Each range reading is here presented under the assumption that the target was hit in the center of the sonar beam.

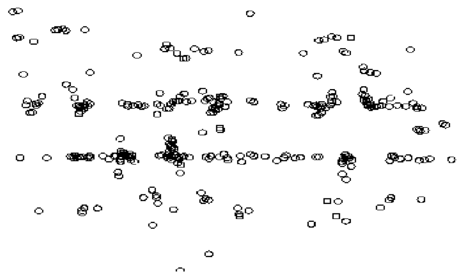

Figure 6: Corridor grid map when applying the Borenstein method [8]

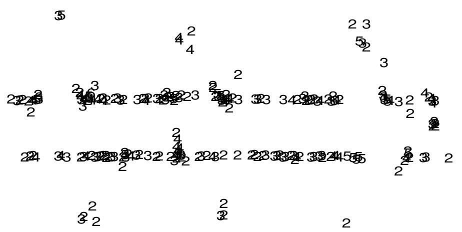

Figure 7: Corridor grid map when applying the technique in section 4. Each presented number is the associated strength value $n_t$ to a specific cell.

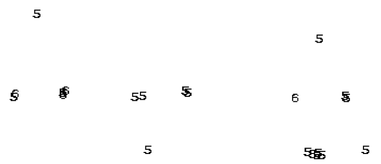

Figure 8: The best triangulations from figure 7. They all correspond to door posts, with the exception of the two top ones, which correspond to a good ultra sonic reflector within a room associated with an open door in the corridor.

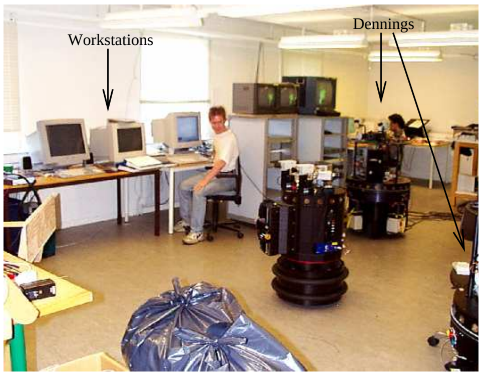


Figure 9: View 1 of the laboratory.


Figure 10: View 2 of the laboratory.

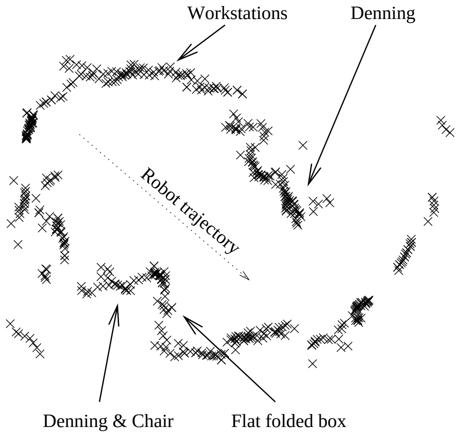


Figure 11: Raw data from the laboratory run. Each range reading is here presented under the assumption that the target was hit in the center of the sonar beam. The speed of the robot was 7 inches/sec.

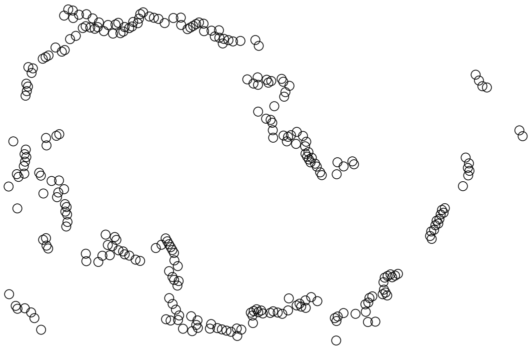

Figure 12: Laboratory grid map when applying the Borenstein method [8]

Figure 13: Laboratory grid map when applying the technique in section 4. Each presented number is the associated strength value $n_t$ to a specific cell.

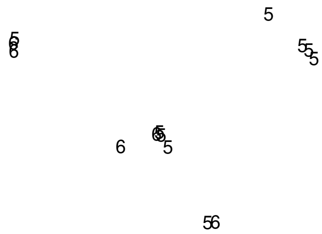

Figure 14: The best triangulations from figure 13. An interesting question is if we can interpret this data as positions of natural landmarks to be used for navigation and localization?

map, and as can be seen in this case they all correspond to door posts, with the exception of the two top ones, which correspond to a good ultra sonic reflector within a room associated with an open door in the corridor.

The corridor represents a highly structured environment that is well suited for the technique presented here. A far more unstructured environment was also tested, our robot laboratory. In figures 9 and 10 the lab environment is shown from two different view points. The major objects have been marked with arrows for easier interpretation of the map presented below.

In figure 11 the raw-data from the run is shown, together with the robot trajectory and the major objects. Figures 12 and 13 show the result of filtering the raw-data with the Borenstein method and our technique, respectively. Both maps contain the essential data from the raw data picture, however, the triangulation map is more sparse. In figure 14 we have filtered out the highest $n_t$-values in the map. For instance we note that the both Denning robots (Rob & Cop) seen in figures 9 and 10 are quite good ultra sonic reflectors. An appealing idea would be to store the positions of the best $n_t$-values and use them as natural landmarks for localization and navigation. We are planning to evaluate this idea by experiments in the future, and hopefully we will have the opportunity to come back upon this matter.

## 7 Summary and Conclusions

A triangulation based technique for clean-up of sonar maps has been introduced. In combination with a simple voting scheme it is possible to generate maps that allow for accurate estimation of high visibility sonar targets, like walls and doorways. The technique has been evaluated on a Nomad platform, where good results have been obtained. In comparison to other techniques, an improved spatial localization has been achieved. In present experimental work the method is being evaluated in the context of several different settings to allow determination of the generality and robustness of the technique.

## Acknowledgments

The research has been sponsored by the Swedish Foundation for Strategic Research under the "Centre for Autonomous Systems" contract.

## References

[1] J. L. Crowley. World modeling and position estimation for a mobile robot using ultrasonic ranging. *IEEE Conference on Robotics and Automation*, 1989.

[2] A. Elfes. Sonar-based real-world mapping and navigation. *IEEE Journal of Robotics and Automation*, RA-3(3):249–265, June 1987.

[3] J. Borenstein and Y. Koren. The vector filed histogram - fast obstacle avoidance for mobile robots. *IEEE Transactions on Robotics and Automation*, 7(3):278–288, June 1991.

[4] J. Borenstein, H. R. Everett, and Liqiang Feng. *Navigating Mobile robots.* A K Peters, Wellesley, Massachusetts, 1996.

[5] J. J. Leonard and H. F. Durrant-Whyte. *Directed Sonar Sensing for Mobile Robot Navigation.* Kluwer Academic Publisher, Boston, 1992.

[6] Phillip John McKerrow. Echolocation - from range to outline segments. *Robotics and Autonomous Systems*, 11(4):205–211, 1993.

[7] Hans P. Moravec. Sensor fusion in certainty grids for mobile robots. *AI Magazine*, 9(2):61–74, Summer 1988.

[8] J. Borenstein and Y. Koren. Histogram in-motion mapping for mobile robot obstacle avoidance. *IEEE Transactions on Robotics and Automation*, 7(4):535–539, August 1991.