Application of Multi-Target Tracking to Sonar-based Mobile Robot Navigation*

John J. Leonard and Hugh F. Durrant-Whyte
Department of Engineering Science
University of Oxford
Oxford, U.K.

Abstract

This paper describes an approach to mobile robot navigation that unifies the problems of obstacle avoidance, position estimation and map building in a common multi-target tracking framework. We view model-based navigation as a process of tracking naturally-occurring geometric targets, or “beacons”. Targets that have been predicted (expected) from the environment map are tracked to provide vehicle position estimates (localization). Targets that are observed, but not predicted, represent unknown environment features or obstacles, and cause new tracks to be initiated, classified, and ultimately integrated into the map. A good sensor model is a crucial component of this approach, and is used both for predicting expected observations and classifying unexpected observations.

This navigation framework is being implemented on a mobile robot that employs sonar as the principle navigation sensor. We present an implementation of model-based localization that achieves robust position estimation in a known environment. We present preliminary results in obstacle identification and map building that lead us to believe that a complete navigation system, encompassing localization, obstacle avoidance, and map building, can be implemented exclusively with sonar.

*Presented at the 29th IEEE Conference on Decision and Control (CDC), Hawaii, U.S.A., December 1990.
1 Navigation by Tracking Geometric Targets

Autonomous mobile robot navigation has been an important goal in robotics for a number of years. There has been considerable success in understanding and solving different aspects of the overall navigation problem. Methods for accomplishing obstacle avoidance using either occupancy grids\cite{18}\cite{12} or potential fields\cite{5}\cite{22} have been successfully demonstrated. Methods for position determination using \textit{a priori} models have been implemented\cite{7}\cite{10}. Kalman filtering techniques have been successfully applied in a variety of applications, including autonomous highway driving\cite{9} and visual map building and motion estimation\cite{1}.

In general, the term navigation is usually taken to mean a combination of abilities including obstacle avoidance, position determination, and map building. Traditionally different aspects of the navigation problem have been treated separately. For example, methods developed for obstacle avoidance are typically of no use in solving the problem of position determination, and the algorithms used for globally-referenced position estimation are not of any direct help in building a map of the environment. In this paper, we present a unified approach to the navigation problem which combines obstacle avoidance, position determination, and map building functions in a common framework. This approach to navigation builds on our earlier work in applying the Extended Kalman Filter (EKF) to the problem of mobile robot position estimation \cite{11}. Here, we extend this work by showing that obstacle avoidance can be treated as a complementary problem to position estimation, and that both problems can be considered in the larger context of building and maintaining maps of the environment.

We refer to the task of determining the robot's position as the process of localization. Localization is a top-down, expectation-driven competence; it is a process of "looking" for and tracking \textit{expected} events. In contrast, obstacle avoidance is a bottom-up, data-driven competence; it is a process of detecting and explaining \textit{unexpected} events. An event is "expected" if it can be predicted from either an internal model of the environment or from previously observed events. It is possible to estimate the motion of a vehicle by observing the motion of these expected events \cite{13}. Logically, an "unexpected" event is one that has not been predicted. Such events may arise as the result of spurious sensor readings or as the result of observing an unmodeled object. In a probabilistic sense, all sensor data is either expected or unexpected. Building maps of the environment from sensor information involves maintaining a coherent map of these expected events while explaining and incorporating new, unexpected, events.

Two important observations can be made about about this process:

- Tracking expected events and explaining unexpected events are two complementary parts of the same problem. When an event or target is observed, predictions based on an environment model or previous observations can be used to suggest possible interpretations or explanations of the data. If a target can be explained through correspondence with prior predictions, then it may be tracked to provide positional information. The rejects from this process are those events that can not be explained. If an observation can not be explained, then a new target can be initiated and subsequently tracked to provide an explanation for the information.

- Map building is a dynamic process that involves providing an interpretation of observed sensor information in terms of physical features in the environment. The prediction of expected events relies on the fact that a correct interpretation has been found for targets that are repeatably observed. Further, tracks that are initiated to explain unexpected events are an attempt to develop such an interpretation over time. Thus the map is built up as new events are observed and explained, and is refined by reobserving events that have been correctly interpreted.

Figure 1 presents an overview of our approach. Utilizing a sensor model, predictions are made, and features are extracted from sensor data. For sonar, our sensor model tells us that these expected and observed features take the form of Regions of Constant Depth (RCDs). Observed and predicted RCDs are
Figure 1: A unified approach to navigation by multi-target tracking.
matched using a “validation gate”[4]. Matched RCDs provide innovations (normalized errors) to update the vehicle location and confirm the presence of prior known targets. The “rejects” from the validation gate represent unexpected data, and need to be explained. From our model of sonar, a given unexpected RCD can be caused by a variety of geometric targets; multiple hypotheses need to be attached to each possible alternative, and new data acquired from different locations to disambiguate between these alternatives. This gives us a way of precisely learning the geometry of a scene by observing unknown targets from multiple positions as the vehicle moves.

We begin our development of this navigation framework by presenting a detailed model of the Polaroid ultrasonic ranging system [19] which is used in all our experiments. In Section 2 we describe this model and show how it can be used to extract and interpret information obtained from the sensor data. Section 3 describes how expected events can be tracked to provide positional information to the robot. An implementation of this is described which employs eight static sonars to localize a “Roboter” vehicle by tracking corner, planar, and cylindrical targets. Section 4 deals with the bottom-up interpretation of sonar observations to provide obstacle detection and map building capabilities. Experimental results demonstrate successful map building with real sonar data.

2 Target Extraction and the Sensor Model

Our model of sonar is inspired by the work of Kuc and Siegel [15]. Their work describes a physically-based model of sonar which considers the responses of corners, walls, and edges in a specular environment. One key conclusion from their work is that corners and walls produce responses that cannot be distinguished from a single scan. The responses from these specular targets take the form of a sequence of headings over which the range value measured is very accurate (within 1 cm.)

The applicability of this model might seem limited by the fact that most environments present a complex mixture of both diffuse and specular targets. However, the vast amount of sonar data that we have taken in our research has led us to conclude that almost all measurements obtained by the off-the-shelf, unmodified Polaroid ranging system in typical scenes (e.g. offices, corridors) are in fact the result of specular reflections. Figure 2 shows a typical, densely-sampled (612 range readings), sonar scan obtained in an uncluttered office scene. Figure 3 shows a range-orientation plot of this information. With this high sampling density, it becomes clear that the scan is composed of sequences of headings at which the range value measured is essentially constant. We refer to such sequences as Regions of Constant Depth (RCDs).

To apply Kuc and Siegel’s model to typical indoor scenes, we briefly put forth an extension to their model. Figures 2 to 4 show that in a typical indoor scene, many “false” range readings are produced by the system when the beam is oriented at high angles of incidence to planar targets. At these high angles of incidence, the sound energy emitted from the side-lobes of the beam that strikes the wall perpendicularly is not of sufficient strength to exceed the threshold of the receiving circuit. As a result, the first echo detected by the system is a multiple reflection by some other part of the beam reflecting specularly off the wall to some other target and then back. These multiple reflections have been observed by many other researchers, and have commonly been referred to in the literature as “specularities”[8]. We feel this term is misleading, because Kuc’s model shows that most accurate range measurements produced by planes and corners are in fact due to specular reflections.

To alleviate this confusion we introduce some terminology: we define the order of a range measurement as the number of surfaces the sound has reflected from before returning to the transducer. Orienting the transducer perpendicular to a planar surface such as a wall produces a 1st-order range reading. Corners produce 2nd-order range readings, because the sound has reflected specularly off two surfaces before returning back to the transducer. Multiple reflections will produce 3rd and higher-order range readings. A crucial
Figure 2: A typical sonar scan.

Figure 3: A plot of Range vs. Transducer orientation for this same scan. 612 measurements equally-spaced in angle were taken. The x axis shows the transducer orientation in degrees. The y axis shows the range in meters. This plot shows that a large proportion of the sonar scan consists of angular regions in which adjacent measurements have nearly the same range, and hence form horizontal line segments (circular arcs in cartesian coordinates.) We refer to these features as Regions of Constant Depth (RCD).
Figure 4: Regions of Constant Depth (RCDs) of width $\beta \geq 10$ degrees extracted from this sonar scan, superimposed on a model of the room. We can see 4 1st-order RCDs from the four walls of the room, 3 2nd-order RCDs from corners, and a single 4th-order RCD resulting from a multiple reflection off the top wall into the lower right-hand corner of the room. There is a single 0th-order RCD resulting from a weaker reflection from the edge in the lower right-hand region of the room.
task in interpretation is to eliminate these higher-order reflections which, if taken to be the distance to the nearest object, yield false range readings.

Another conclusion Kuc and Siegel reach is that (convex) edges give rise to diffuse echoes that will be weaker in intensity than reflections from walls or corners. In a recent paper Kuc [14] shows that the primary task for obstacle avoidance is to “look” for edges since these are the most difficult to detect. This implies that for the purpose of localization, edges are less useful as beacons. To incorporate diffuse edges in our terminology, we refer to diffuse reflections from edges as 0th-order range readings.

We define $\beta$ as the visibility angle of a given target, corresponding to the angles over which the RCD is observed. Figure 4 shows the result of extracting RCDs of width $\beta \geq 10$ degrees from the scan in Figure 2 using a simple thresholding algorithm, superimposed on a line segment model of the room. Here we can see RCDs corresponding to planes, convex and concave corners, and higher-order targets.

Plane, Corner, and Cylinder Target Models

Qualitatively, the shape of the sonar beam pattern dictates that the width of an RCD is determined by a target’s ability to reflect acoustic energy—the stronger the target, the wider the RCD. However, to compute in general the precise visibility angle $\beta_i$ of target $p_i$, is a complicated process involving a multitude of factors, including the complex pattern of the beam (including side-lobes), limitations in the ranging system hardware, and occlusion by other targets. We are addressing these issues in our current research, but for our purposes in this paper, the important conclusion is that planes, cylinders, and corners are the strongest targets, and hence we can obtain precise range measurements to these targets over a wide range of viewing angles. We proceed now to present measurement models for plane, corner, and cylinder targets.

1. Corner

A Corner is a point target in 2-D, and is defined by the parameter vector $p_c = (p_x, p_y)$ where $p_x$ and $p_y$ are the x and y coordinates of the corner defined in global coordinates.

2. Plane

A Plane is a line in 2-D, and is defined by the parameter vector $p_l = (p_R, p_\theta)$. We represent the line in hessian normal form [2]: $p_R$ is the minimum distance from the (infinite) line to the origin of the global coordinate frame, and $p_\theta$ is the angle with respect to the x axis of a perpendicular drawn from the line to the origin.

3. Cylinder

A Cylinder is a circle in 2-D, and is defined by the parameter vector $p_{cy} = (p_x, p_y, p_R)$ where $p_x$ and $p_y$ are the x and y coordinates of the center of the circle in global coordinates and $p_R$ is the radius of the circle.

We define $\hat{r}_i(k)$ as the true range and $\phi_i(k)$ as the true bearing to beacon $p_i$ from $x(k)=[x(k), y(k), \theta(k)]^T$, the vehicle position at time $k$. We compute $\hat{r}_i(k)$ differently for each type of target using the beacon parameterization vector $p_i$. For corners,

$$\hat{r}_i(k) = \sqrt{(p_x - x(k))^2 + (p_y - y(k))^2} \quad (1)$$

$$\tan(\phi_i(k) + \theta(k)) = \frac{p_y - y(k)}{p_x - x(k)} \quad p_x \neq x(k) \quad (2)$$

For planes,

$$\hat{r}_i(k) = |p_R - x(k)\cos(p_\theta) - y(k)\sin(p_\theta)| \quad (3)$$

6
\[ \phi_i(k) = p_\theta - \theta(k) \]

And for cylinders,

\[ \hat{r}_i(k) = \sqrt{(p_x - x(k))^2 + (p_y - y(k))^2} - p_R \]

\[ \tan(\phi_i(k) + \theta(k)) = \frac{p_y - y(k)}{p_x - x(k)} \quad p_x \neq x(k) \]

We define range measurement \( j \) as an ordered pair \((\alpha_j(k), r_j(k))\), where \( \alpha_j(k) \) is the orientation of the sensor and \( r_j(k) \) is the range detected by the sensor. In the absence of occlusion from other targets and neglecting asymmetries in the beam and side-lobe effects, the range measurement \((r_j(k), \alpha_j(k))\) corresponds to target \( i \) if

\[ \phi_i(k) - \frac{\beta_i}{2} \leq \alpha_j(k) \leq \phi_i(k) + \frac{\beta_i}{2}. \]

If this correspondence criterion has been met, then the range value \( r_j(k) \) will approximately equal \( \hat{r}_i(k) \), the true range to target \( i \).

### 3 Localization: Tracking Expected Targets

At the heart of our approach to the navigation problem is the ability of the robot to determine its position by observing and tracking expected "geometric beacons" whose locations are stored in a map. We use the word "target" to refer to any object feature in the environment which can be observed by one of the robot's sensors. A "geometric beacon" is a special class of target which is useful for localization: stable environment features such as walls and corners. We stress that the map is just a set of beacon locations, not an exhaustively detailed world model. For the localization results presented here, the map is provided \textit{a priori} to the algorithm.

With reference to Figure 5, we denote the position and orientation of the vehicle at time step \( k \) by the state vector \( x(k) = [x(k), y(k), \theta(k)]^T \) comprising a cartesian location and a heading defined with respect to a global coordinate frame. At initialization the robot starts at a known location, and has an \textit{a priori} map of the locations of geometric beacons \( p_i \). Each beacon is assumed to be precisely known. At each time step, observations \( z_j(k) \) of these beacons are taken. Our goal in the cyclic process is to associate measurements \( z_j(k) \) with the correct beacon \( p_i \) to compute an updated estimate of vehicle position.

The Kalman filter relies on two models: a \textit{plant model} and a \textit{measurement model}. The plant model describes how the vehicle’s position \( x(k) = (x(k), y(k), \theta(k)) \) changes with time in response to a control input \( u(k) \) and a noise disturbance \( v(k) \)

\[ x(k+1) = f(x(k), u(k)) + v(k), \quad v(k) \sim N(0, Q(k)) \]

where \( f(x(k), u(k)) \) is the (non-linear) state transition function.

The measurement model expresses a sensor observation in terms of the vehicle position and the geometry of the beacon being observed, and has the form:

\[ z_j(k) = h_i(p_i, x(k)) + w_j(k), \quad w_j(k) \sim N(0, R_j(k)) \]

The measurement function \( h_i(p_i, x(k)) \) expresses an observed measurement \( z_j(k) \) as a function of the vehicle location \( x(k) \) and beacon location \( p_i \), and takes a different form for each type of target (corner, plane, or cylinder) in accordance with equations 1, 3, and 5. This observation is assumed corrupted by a zero-mean, gaussian noise disturbance \( w_j(k) \) with variance \( R_j(k) \).
Figure 5: Localization by Tracking Geometric Beacons. The vector $\mathbf{x}(k) = (x(k), y(k), \theta(k))$ is the vehicle's position and orientation at time $k$. Four geometric beacons are in view to an ultrasonic sensor at time $k$ and time $k + 1$: plane $p_1$, corner $p_2$, plane $p_3$, and cylinder $p_4$. The sonar measurements $z_1(k)$ and $z_3(k)$ are the shortest distance from the vehicle to planes $p_1$ and $p_3$ at time $k$. The measurement $z_2(k)$ is the distance from the vehicle to corner $p_2$ at time $k$. Measurement $z_4(k)$ is the distance to the central axis of cylinder $p_4$ less the radius of the cylinder.
The goal of the cyclic computation is to produce an estimate of the location of the robot $\hat{x}(k + 1 | k + 1)$ at time step $k + 1$ based on the estimate of the location $\hat{x}(k | k)$ at time step $k$, the control input $u(k)$ and the new beacon observations $z_j(k + 1)$. The algorithm employs the standard Kalman filtering steps of prediction, observation, matching, and estimation. We state the extended Kalman Filter equations without derivation, and refer the reader to [4] for more detail.

Prediction

First, using the plant model and a knowledge of the control input $u(k)$, we predict the robot's new location at time step $k + 1$:

$$\hat{x}(k + 1 | k) = f(\hat{x}(k | k), u(k))$$

(10)

We next compute $P(k + 1 | k)$, the variance associated with this prediction:

$$P(k + 1 | k) = \nabla f \cdot P(k | k) \cdot \nabla f^T + Q(k)$$

(11)

where $\nabla f$ is the Jacobian of $f(\cdot, \cdot)$ obtained by linearizing about the updated state estimate $\hat{x}(k | k)$. Next, we use this predicted robot location to generate predicted observations of each geometric beacon $p_i$:

$$\hat{z}_i(k + 1) = h_i(p_i, \hat{x}(k + 1 | k)), \quad i = 1, \ldots, N_k$$

(12)

Observation

The next step is to actually take a number of observations $z_j(k + 1)$ of these different beacons, and compare these with our predicted observations. The difference between a prediction $\hat{z}_i(k + 1)$ and an observation $z_j(k + 1)$ is termed the innovation, and is written as

$$\nu_{ij}(k + 1) = [z_j(k + 1) - \hat{z}_i(k + 1)]$$

(13)

The innovation covariance can be found by linearizing Equation 9 about the prediction, squaring, and taking expectations

$$S_{ij}(k + 1) = \nabla h_i \cdot P(k + 1 | k) \cdot \nabla h_i^T + R_j(k + 1)$$

(14)

where the Jacobian $\nabla h_i$ is evaluated at $\hat{x}(k + 1 | k)$ and $p_i$.

Matching

Around each predicted measurement, we set up a validation gate in which we are prepared to accept beacon observations:

$$\nu_{ij}(k + 1) S_{ij}^{-1}(k + 1) \nu_{ij}^T(k + 1) \leq g^2$$

(15)

This equation is used to test each sensor observation $z_j(k + 1)$ for membership in the validation gate for each predicted measurement. When a single observation falls in a single validation gate, we get a successful match. Measurements which do not fall in any validation gate are simply ignored for localization. More complex data association scenarios can arise when a measurement falls in two or more validation regions, or when two or more measurements fall in a single validation region. At this stage, such measurements are simply ignored by the algorithm, as outlier rejection is vital for successful localization. This has proven acceptable thus far, but we intend to investigate the use of probabilistic data association techniques[4] for this purpose in the future.
Estimation

The final step is to use successfully matched predictions and observations to compute \( \hat{x}(k+1 | k+1) \), the updated vehicle location estimate. To do so we use a parallel update procedure [23]. We first stack the validated measurements \( z_j(k+1) \) into a single vector to form \( z(k+1) \), the composite measurement vector for time \( k+1 \), and designate the composite innovation \( v(k+1) \). Next, we stack the measurement jacobians \( \nabla h_i \) for each validated measurement together to form the composite measurement jacobian \( \nabla h \). Using a stacked noise vector \( R(k+1) = \text{diag}[R_i(k+1)] \), we then compute the composite innovation covariance \( S(k+1) \) as in Equation 14. We then utilize the standard result that the Kalman gain can be written as

\[
W(k+1) = P(k+1 | k) \nabla h^T S^{-1}(k+1)
\]

(16)

To compute the updated vehicle position estimate

\[
\hat{x}(k+1 | k+1) = \hat{x}(k+1 | k) + W(k+1) v(k+1)
\]

(17)

with associated variance

\[
P(k+1 | k+1) = P(k+1 | k) - W(k+1) S(k+1) W^T(k+1)
\]

(18)

Experimental Results

This localization algorithm has been implemented on several different robots. Our initial implementation, described in detail in [11], only used planar targets. On-the-fly localization was achieved at typical speeds of 30 cm per second under the restriction that the vehicle follow paths nearly perpendicular to the environment. Our work in progress is extending this system to track all three types of targets described in Section 2 – corners, cylinders, and planes. The results we show here are from a "stop, look, move" run using a fixed ring of 8 sonar sensors mounted on a Robuter mobile robot. When using a ring of fixed sonars, the advantage is that a high sampling speed can be obtained, but the disadvantage is that interpreting the data is made more difficult because each range measurement has no local support, as in the densely-sampled scan of Figure 2. Our remedy to this situation is to rely on sonar's high range accuracy while not using the sensor orientation \( \alpha_j \) directly in the filter. We use the sensor orientation to determine if a sonar return might possibly have originated from a particular target using equation 7. Updates to the orientation of the vehicle come implicitly via range measurements from the different sensors around the vehicle perimeter.

Figure 6 shows various stages of a run in a room with large pillars whose rounded corners define good cylindrical navigation beacons. Figure 7 shows estimated and odometric positions for each stage of the complete run. The room model was measured by hand. The vehicle is 1 meter long and 0.7 meters wide, and has one sensor facing forwards, one sensor facing backwards, two sensors on each side, and one sensor on each of the front corners oriented 30 degrees with respect to straight ahead. The vehicle started from a known location in the lower right-hand corner of the room.

Each time a set of eight range readings are acquired, a reading from the vehicle's odometry system is made. The odometry measurement is used to provide a prediction (using Equation 10) of the vehicle's location at which these sonar range readings were acquired. The matrix \( Q(k) \) was given values to reflect 5 cm of position error and 4 degrees of orientation error for each meter of translation or 90 degrees of rotation. (These values were chosen by an empirical observation of dead-reckoning's performance.) At this predicted location, predicted range measurements are generated for each target in the map using the target models presented in Section 2. We assume a maximum value of \( \beta \) for all targets of 26 degrees. If the sensor orientation is not within 13 degrees of the true bearing to a target, a NULL prediction that cannot be matched is generated.
These predictions are matched to the observed range values using Equation 15, and correct matches are used to update position. The innovation variance \( S(t+1) \) is computed using \( \sigma_r = 1 \) cm for the standard deviation of range measurement error. A value of 2 was used for the validation gate “number of sigmas”[4]. Figure 6 shows the current validated measurements at various stages in the run. 37 percent of the 976 range measurements taken during the run were validated, for an average of just under 3 matched readings per time step. This system works because successful matches prevent small errors in position and orientation from accumulating over time. This reduces the likelihood of incorrect matches between predictions and observations, preventing the vehicle from becoming lost.

4 Obstacle Avoidance and Map Building: Explaining the Unexpected

Current obstacle avoidance systems do not differentiate between known objects and unidentified obstacles. Occupancy grids[12] and potential field methods[22], for example, treat all observed objects as obstacles, and are unable to distinguish between objects which are already known and those that are new. We think of obstacles as unexplained targets, or “rejects” from the localization process presented above in section 3. This has the advantage that our obstacle avoidance system can concentrate on those events which are truly unexpected. In this discussion the environment is assumed static, i.e. all targets are stationary. We refer the reader to [17] for a discussion of the dynamic map building problem.

4.1 Initializing Tracks for Unknown Targets

We view explaining an unknown RCD as a process of track initiation. The first requirement for track initiation is target identification. Kuc and Siegel show that corners and walls will appear the same in a scan from a given location [15]. A cylinder is a 1st-order target that appears similar to a plane or corner from a single location. Higher-order multiple reflections also produce RCDs that if considered in isolation from a single location could be interpreted as any of these targets. After identification, we need to compute \( \mathbf{p}_t \), the target geometric parameterization.

The motivation for our track initiation procedure comes from the Multiple Hypothesis Tracking (MHT) algorithms which have been presented in the literature[20][3][16]. These approaches use Bayes’s theorem to calculate probabilities for multiple data association hypotheses based on known probabilities of detection and assumed densities of clutter and new targets. However, our algorithm does not calculate proper target probabilities using Bayes’s theorem at this stage.

The first approach we considered was to use the unknown RCD to initialize parallel filters, each based on a different assumed target: plane, corner, or cylinder of a given radius. Subsequent matches could then be made with the same machinery as used for localization, by matching observed RCDs with RCDs predicted from the hypothesized target. Hopefully, several subsequent measurements would be matched to the correct filter, while incorrect filters would not receive any matches, and could be pruned away. However, we encountered difficulties with this approach because the full target state is not observable from a single RCD, as a result of sonar’s high angular uncertainty. With densely-sampled data and in the absence of occlusion, the true bearing to the target can be taken as the center of the RCD. However, in practice all but the closest targets in a scene are usually partially occluded. Further, it is desirable to have a method that will work for single range measurements acquired by a moving vehicle. To meet these requirements, we use the motion of the vehicle to take observations of the unknown geometric feature from different locations. Figure 8 illustrates this process.
Figure 6: A localization run in the Oxford AGV laboratory, showing the validated sonar range measurements used to update the vehicle position at various stages in the run. The triangle shows the current position as estimated solely by odometry. The rectangle shows the \textit{a posteriori} position estimate produced by the algorithm.
The primary step of our track initiation procedure consists of evaluating \textit{data-to-data} association hypotheses for RCDs observed from multiple vehicle locations. In contrast, the matching procedure used for localization evaluates \textit{data-to-target} associations. The simple rule for our data-to-data association logic is that RCDs which correspond to a plane (or cylinder) will all be tangent to the plane (or cylinder), while RCDs which correspond to a corner will all intersect in a point, at the corner.

4.2 Building and Maintaining Maps

In the general setting of an unknown environment, this track initiation capability that is used to learn a single obstacle feature will need to be used to learn all the targets in the environment. This should be no more difficult in principle than learning a single unknown object if slow, densely-sampled scans are taken in a "stop, look, move" operating scenario, which would be acceptable when learning. The objective of learning is simply to produce a "map" that is a list of target locations. The extent and detail of this model of the environment is then only just enough to allow the robot to "look" for these targets for "on-the-fly" localization in a normal mode of operation.

To test the use of our track initiation procedure for map-building, a grid of sonar scans were taken from precisely known positions in an uncluttered office scene. Figure 9 shows triangles at each vehicle location for 28 sonar scans, and a hand-measured model of the room in which the scans were taken. The scans were processed off-line in a spiral sequence, starting from the upper left of the figure. Figure 10 shows the map of plane and corner target locations produced by the algorithm. Figure 11 shows this map of target locations superimposed over the hand-measured model, revealing the close correspondence between the built-up map and the actual geometry of the room. Figures 12 and 13 show RCDs matched to typical corner and plane targets. (These results are described in more detail in [17].)
Figure 8: Using the motion of the vehicle to disambiguate between multiple hypotheses for an unexpected event. We track the unknown RCD as we move to determine whether it came from a corner, plane, cylinder, or higher-order multiple reflection. RCDs which correspond to a plane (or cylinder) will all be tangent to the plane (or cylinder). RCDs which correspond to a corner (or edge) will all intersect in a point, at the corner (or edge). RCDs for higher-order multiple reflections can be distinguished because they follow unpredictable trajectories as the vehicle moves.
Figure 9: Hand-measured map of the room, with triangles at each of 28 locations where sonar scans were taken. The shaded triangles indicate the start and finish of the run. The room is 3 meters wide, with a closed door in the upper right hand region of the picture.
Figure 10: Localization map of the room produced by the algorithm. 8σ error ellipses are shown for point (corner) targets.

Figure 11: Learned map superimposed over room model.
Figure 12: RCDs matched to a typical corner target.

Figure 13: RCDs matched to a typical planar target.
5 Conclusion

We have presented a common framework for achieving the dual objectives of localization and obstacle avoidance by classifying sensor data as "expected" and "unexpected"; we use "expected" sensor data to determine the robot's absolute position with respect to a global reference frame while explaining unexpected sensor data to detect obstacles and changes in the environment. We have treated the problem in this manner because the ultimate aim of our research is to endow the robot with an ability to learn an unknown environment and use its learned knowledge to operate efficiently while still detecting transient unexpected events and updating its learned model in response to more permanent changes in the environment. The implementations presented here each address a limited subset of the general problem: localization in a static environment known a priori, and map building from precisely known vehicle locations.

Acknowledgements

This work is supported in part by ESPRIT 1560 (SKIDS) and SERC-ACME GRE/42419. The implementations presented here used software developed with Jean-Michel Valade of Matra MS2I and Chris Brown[6]. The authors thank Yaakov Bar-Shalom for valuable comments.

References


