Dynamic World Modeling for an Intelligent Mobile Robot Using a Rotating Ultra-Sonic Ranging Device

James L. Crowley

CMU-RI-TR-84-27

The Laboratory for Household Robotics
The Robotics Institute
Carnegie-Mellon University
Pittsburgh, Pennsylvania 15213

December 1984

This research was sponsored by Commodore Business Machines, Inc., Denning Mobile Robotics, Inc., and the Commonwealth of Pennsylvania.

Table of Contents

List of Figures

Abstract

A system which performs task-oriented navigation for an intelligent mobile robot is described in this paper. This navigation system is based on a dynamically maintained model of the local environment, called the "Composite Local Model." The Composite Local Model integrates information from a rotating sonar sensor, the robot's touch sensor and a pre-learned Global Model as the robot moves through its environment.

Techniques are described for constructing a line segment description of the most recent sensor scan (the Sensor Model). and for integrating such descriptions to build up a model of the immediate environment (the Composite Local Model). Model integration is based on a process of reinforcing the confidence in consistent information while decaying the confidence in inconsistent information. The estimated position of the robot is corrected by the difference in position between observed sensor signals and the corresponding symbols in the Composite Local Model. This system is useful for navigation in a finite, pre-learned domain such as a house, office, or factory.

1 Introduction

A system for dynamically maintaining a description of the external environment of a mobile robot using a focussed rotating ultra-sonic ranging device is described in this paper. This system is designed to support autonomous navigation by an intelligent mobile robot in a previously learned floorplan.

In the first section of this paper, the problems of world modeling, position estimation and navigation are introduced, and solutions for each of these problems are summarized. We then present the computational framework for this world modeling and navigation system. This is followed by sections with techniques for

- constructing an abstract description of the robot's environment (The Sensor Model) using a rotating sonar ranging device,
- matching this description to a model of the immediate environment (The Composite Local Model),
- detecting and correcting errors in the robot's estimated position and orientation using the correspondnence between the Sensor Model and the Composite Local Model, and
- updating the contents of the Composite Local Model based on the contents of the Sensor Model.

The Composite Local Model is at the heart of this navigation system; it is used for learning, path planning, and path execution.

The techniques described in this paper are part of an effort to develop a low-cost "Intelligent Mobile Platform" or IMP. By the term "intelligent" we mean that the navigation is "task-oriented" and that it is based on dynamically sensing and modeling the external world. The IMP is designed to respond to commands of the form "Go To <place>" where <place> is a pre-learned location in a network of "learned places." The IMP is able to use its network of places to plan a path to <place>. It is then able to use its sensing, modeling and navigation abilities to execute this plan and to modify the plan dynamically in reaction to unexpected events. The IMP is to serve as a foundation for mobile household, business, and factory robots which require intelligent navigation.

1.1 Navigation and World Modeling: the Problem

The task of a navigation system is to plan a path to a specified goal and to execute this plan, modifying it as necessary to avoid unexpected obstacles. The path planning problem can be broken down into a global planning problem and a local planning problem. Global path planning requires a pre-learned model of the domain which may be a somewhat simplified description of the real world, and might not reflect recent changes in the environment. The IMP cannot be everywhere at once. Local path planning carries out the steps in the global plan, correcting for changes in the world that are encountered during the plan execution. Whereas global navigation may operate on a pre-stored model, local navigation requires a model which reflects the state of the environment including changes, as the plan is being executed. We refer to such a model as the Composite Local Model.

The Composite Local Model is built up by integrating recent information from different sensors, taken from different viewing angles. When available, information from a pre-learned Global Model may also be integrated into the Composite Local Model. The construction and maintenance of a Composite Local Model involves:

1. building an abstract description of the most recent sensor data (a Sensor Model),

2. matching to determine the correspondence between the most recent Sensor Models and the current contents of the Composite Local Model,

3. modifying the components of the Composite Local Model, and reinforcing and decaying the confidences to reflect the results of matching.

Having the correspondence between the Sensor Model and the Composite Local Model also makes it possible to measure and correct errors in the estimated position and orientation of the robot due to wheel slippage.

1.2 Summary of Solution

In the system described below, global path planning is based on a pre-learned "network of places." The network of places is learned in a special "active learning mode" in which the robot explores its environment. Automatic learning greatly simplifies the practical problem of giving the system an accurate model of the world. Each place in the network is connected to a set of adjacent places by "legal highways." Global navigation is a process of choosing a set of legal highways which will carry the robot from its current location to the specified goal. Traversing each legal highway is the job of the local navigator. Techniques for active learning, global path planning, and local obstacle avoidance have recently been described in [2].

In the current system, a legal highway consists of a straight line path, and local navigation is accomplished by means of a finite state process. Each path is tested for blocking obstacles using the raw sensor data, the Sensor Model and the Composite Local Model. If an obstacle is detected, a recursive obstacle avoidance procedure plans a new sequence of straight line paths to the next local goal. This recursive obstacle avoidance procedure is based on the current contents of the Composite Local Model.

The Composite Local Model, the Sensor Model, and the Global Model are represented in terms of line segments in a two dimensional "floor-plan" world. All three models are expressed in a world centered coordinate system so that they can be matched invariant to the robot's position. The line segments that compose the Sensor Model are constructed using a variation of the recursive line splitting algorithm which is often used to find edges in images [3]. The confidence of line segments in the Composite Local Model is represented by a finite set of states. A relatively simple state transition mechanism is used to reinforce and decay the confidence in line segments. Segments in the Composite Local Model are "grown" by an update process that extends the segments whenever there is a partial overlap with Sensor Model segments.

1.3 Problem Context

An engineering prototype of the IMP contains the functional components shown in figure 1. At the top of the IMP is a rotating depth sensor which senses the distance to external surfaces with a beam with a starting diameter of approximately 3 inches and a beam spread of approximately 5°. The sensor is mounted at a height of 30 inches, which is about the level of most tables. The sensor is turned by a stepper motor in steps of 3°. Approximately 10 seconds are required to obtain the 120 depth readings given by a complete revolution. With each reading, the sensor returns the distance to the nearest surface within 25.6 feet to a resolution of 0.10 feet. As the IMP travels, rotary position encoders mounted on its power wheels are used to maintain an instantaneous estimate of the IMP's position in a cartesian coordinate system.

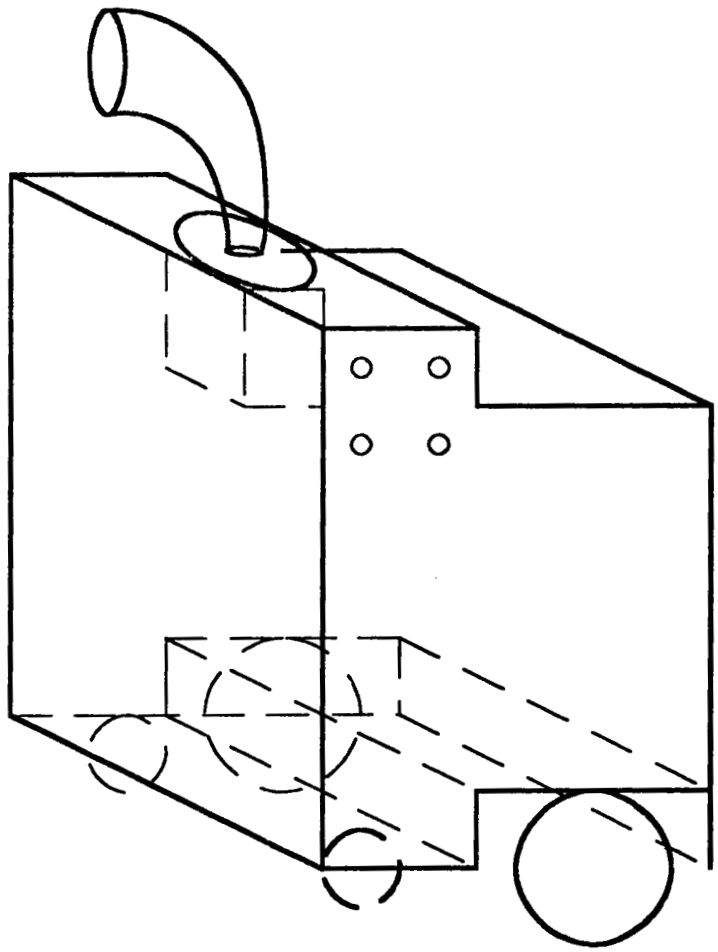

Figure 1: The functional components of the IMP

The world modeling and navigation procedures for the IMP have been implemented and refined using an interactive mobile robot simulation program. These techniques have been re-implemented on a prototype of the IMP which contains two 16 bit micro-processors. Similar techniques have recently implemented for a security robot which uses 24 ultra-sonic ranging devices arrayed in a ring, in place of the rotating focussing horn.

2 The Computational Framework

2.1 The Composite Local Model

The navigation system of the IMP is based on the computational framework shown in figure 2. At the core of this framework is a dynamic model of the surfaces and obstacles in the immediate environment of the IMP called the Composite Local Model. "Local" refers to the fact that only information in the local environment of the IMP is represented. "Composite" refers to the fact that this model is composed of information obtained over time from multiple sensors and from many views.

The Composite Local Model plays two fundamental roles in this computational framework.

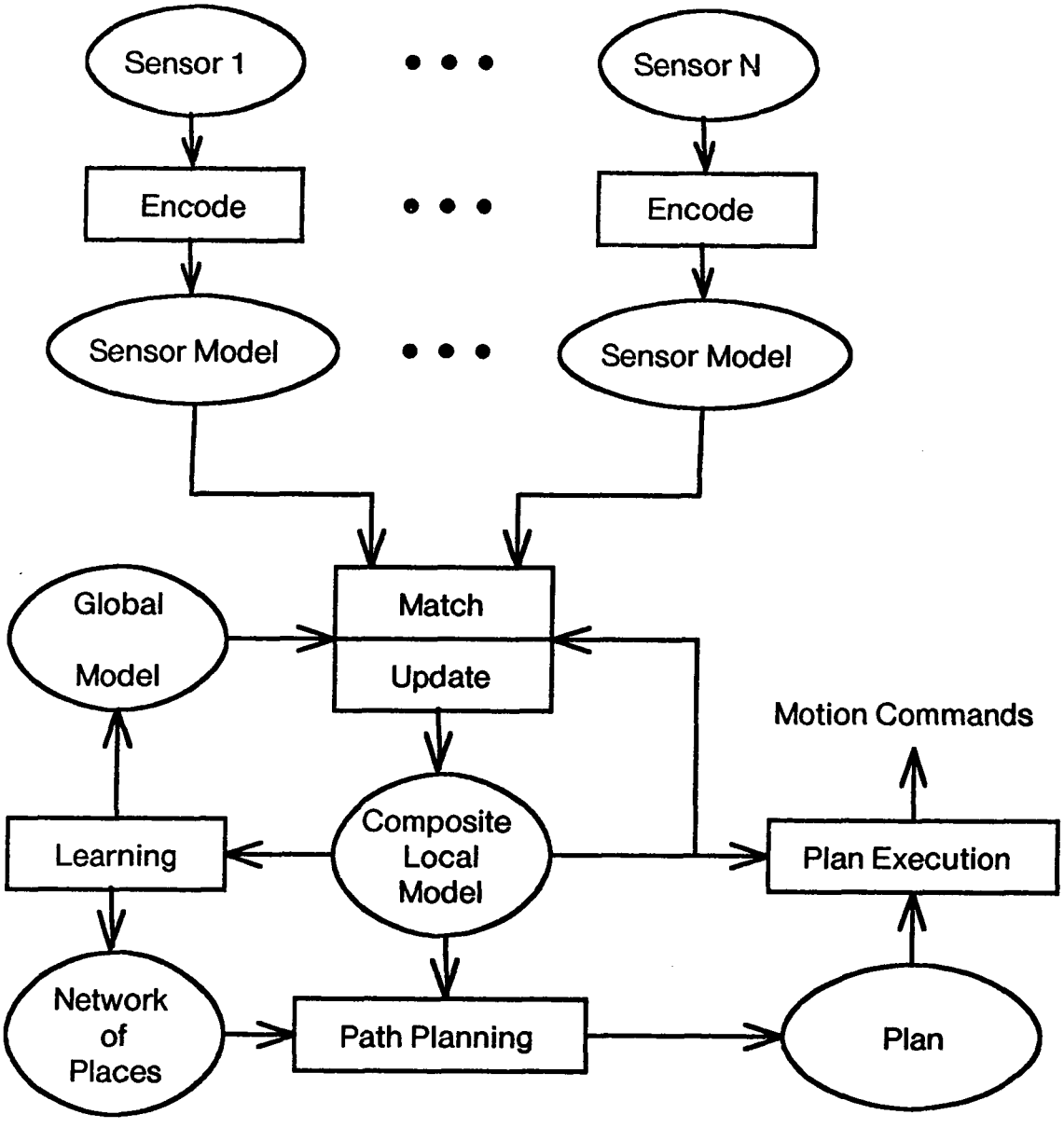

Figure 2: Framework for an Intelligent Navigation System

1. It is the structure in which potentially conflicting information from diverse sensors is integrated with recently observed information and information recalled from long term storage (the Global Model).

2. It is the structure on which processes for local path planning, path execution, learning, object tracking, object recognition, and other "higher level" processes are based.

Because of the nature of the navigation task and the sensors that are employed, the Composite Local Model in the IMP is implemented with a relatively simple 2-D representation. The IMP models the world and plans paths in a 2-D "flat-land" universe. Because the rotating range sensor is mounted at a height of 30 inches, the robot is able to detect and represent most of the furniture that it encounters. Surfaces and obstacles are represented as connected sequences of line segments. Thus a table and a wall have the same structure; both

appear as a barrier with an infinite (or unknown) extent in vertical dimension. The Composite Local Model must include the ability to represent the uncertainty of information. In the IMP, this ability is provided by a state transition mechanism. The line segments which compose the Composite Local Model include a "state" attribute which represents the degree of confidence. Consistent line segments are reinforced and extended while inconsistent line segments are decayed and eventually removed from the model.

2.2 The Sensor Models

Sensors typically produce large amounts of information. Before the information from a sensor can be integrated into the Composite Local Model, surface information must be abstracted from it. The Sensor Model may be viewed as a form of "Logical Sensor" which provides the sensor information in a standard form which may be integrated into the Composite Local Model.

In the first version of the IMP, the sensors are a set of contact sensors on a skirt and the rotating sonar sensor. In each case, the Sensor Model is an abstract description expressed as line segments which represent surfaces in the real world.

2.3 Match and Update

The module labeled "Match" determines the correspondence between the line segments which compose the Sensor Model and the line segments which compose the Composite Local Model. The correspondence is then used to determine errors in the estimated position and to update the position, length and confidence of the segments in the Composite Local Model. Special procedures also exist for detecting and tracking moving objects.

The module labeled "Update" integrates the information from the Sensor Models with the current Composite Local Model. This module adjusts the position, size, connectivity and confidence of the segments in the Composite Local Model to reflect the results of correspondence matching. This Update process also removes segments for which the confidence is low or for which the distance is too far. The process does not remove nearby surfaces which are not currently visible.

3 Constructing the Sensor Model

Depth readings from the rotating sonar are converted into abstract line segments by a sequence of four steps:

1. Project the reading to a Cartesian world coordinate system.

2. Segment measured points into line segments by detecting "discontinuities" and then applying a recursive line fitting process.

3. Compute the line equations of the points from the most reliable interior points.

4. Recompute the segment end-points as the intersection points with neighboring line segments.

These processes are described below.

3.1 Projection to Cartesian World Coordinates

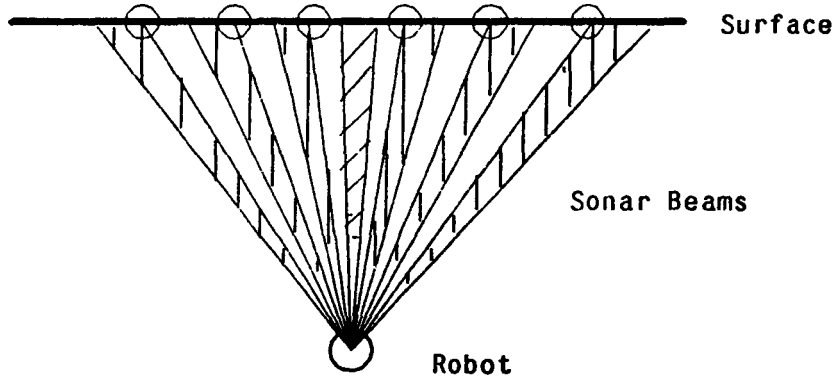

Figure 3: The edges of a sonar beam are projected to world coordinates

Depth readings are obtained from the rotating sonar in cylindrical coordinates, i.e. as depth at a particular angle. As each depth reading is made, the current estimated position and orientation of the IMP is affixed to it. This permits the system to project the reading into a world coordinate system at a later time, even if the data were taken while the IMP was moving.

As a consequence of the detection mechanism in the sonar, the depth reading refers to the depth to the nearest reflecting surface anywhere in the sonar beam's circular footprint. When the beam reflects from a flat surface at a non-perpendicular angle, the sonar returns the distance along the short edge of the beam. Knowledge of this physical process is used in interpreting the sonar depth readings.

As each sonar reading is obtained, it is converted from robot-centered cylindrical coordinates to a world-centered Cartesian coordinate system. This is done by projecting a line by the specified depth at the specified angle, as illustrated in figure 3. If the readings are decreasing as the sonar rotates in a counter-clockwise turn, the angle is adjusted to be the left edge of the beam by adding the estimated half-angle of the sonar beam. If the depth is increasing the angle is adjusted to the right by subtracting the estimated half angle. The difference in depth between the adjacent readings to the right and to the left is computed and affixed to the projected beam as a quality measure.

3.2 Segmenting the Points Into Line Segments

The points are first grouped into a sequence of roughly co-linear readings such that the distance between each adjacent pair of points is less than a tolerance. This tolerance is selected as a compromise between the maximum distance at which the depth readings can be taken, and the smallest gap between objects that the system can detect. For a difference in orientation of α degrees per reading, the minimum distance gap size, G_{min}, is determined by the desired maximum range, R, by considering the difference of beam edges for a perpendicular surface. Such a geometry gives a relationship:

$$G_{min} > RTan(\alpha)$$

In our system, $\alpha = 3^{\circ}$ and R = 25.6 feet, giving G_{min} of 1.34 feet. We have found a value of G_{min} of 1.5 feet to be satisfactory. Thus, the points are scanned to detect any points where the distance between adjacent

readings is greater than G_{min}. Such points are called break points or discontinuity points. Break points mark the boundaries of collections of points that are passed to a recursive line fitting procedure.

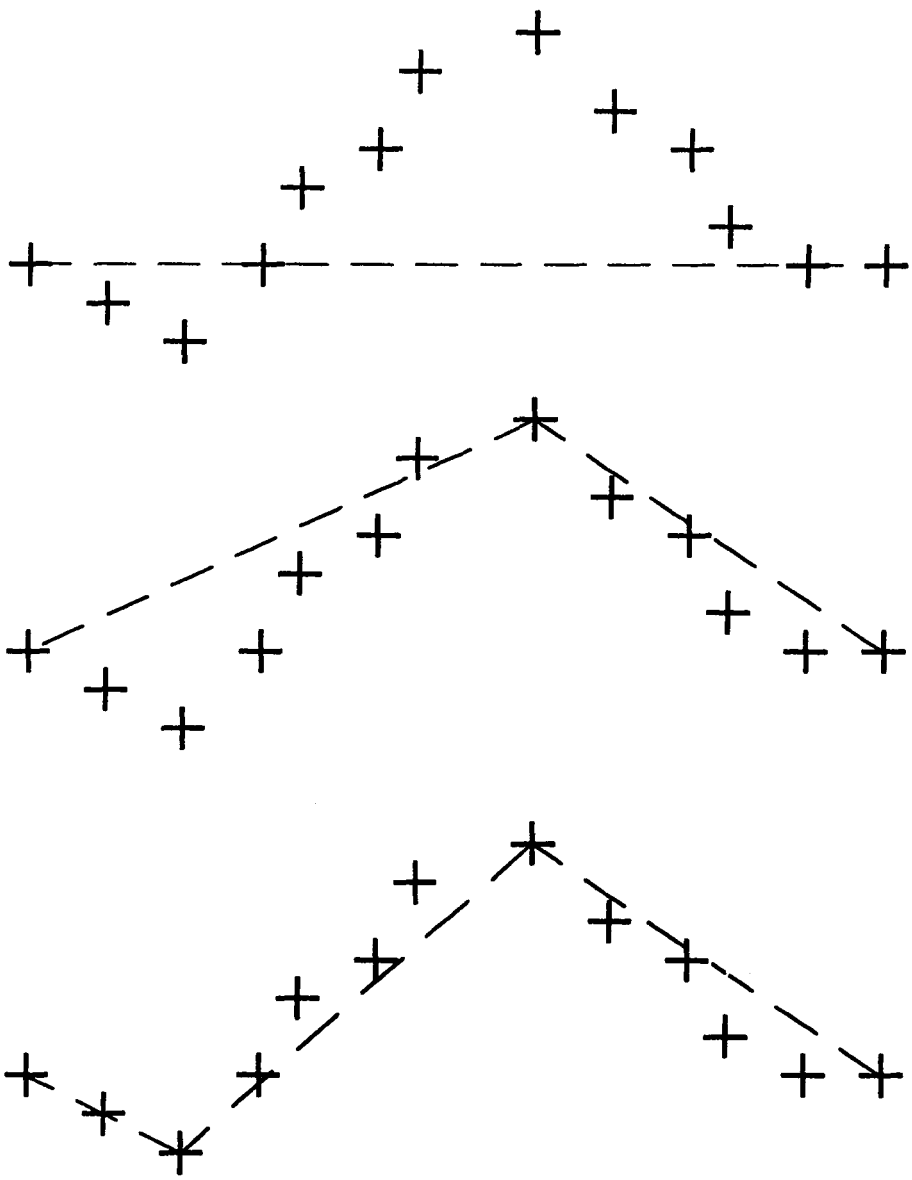

Figure 4: Recursive Line Fitting

Recursive line fitting has been used for years to fit lines to edge points in images [3]. The algorithm is illustrated in figure 4. A line equation of the form

$$Ax + By + C = 0$$

is computed between the two endpoints in the collection of points. If the coefficients A and B are normalized so that the sum of their squares is one, then evaluating the line equation at the location of a point, (x, y), gives the perpendicular distance from the computed line equation. The points in the group between the end points are tested to determine the point where the perpendicular distance is largest. If this largest perpendicular distance is below a tolerance, then the line is accepted as representing the points. Otherwise, the collection of

points is divided into two groups at the point where the perpendicular distance was largest. The line fitting procedure is then evaluated recursively for each of these two groups. The result is a collection of line segments which represents the collection of points.

3.3 Refining the Line Equation

When a sonar beam measures depth near a corner, the depth measurement is often corrupted by reflections. Yet these points give the breakpoints which are used for recursive line fitting. Thus it is desirable to re-calculate the equation of each line segment from interior points.

In our early experiments, we observed that the most reliable sonar points are those to either side of the measurement which is perpendicular to the surface. The perpendicular measurement may be detected as a smooth minimum in the depth readings. However, if a line equation is fit to points which are too close to each other, the equation is very sensitive to small errors in position. To compromise between these conflicting constraints, the difference in depth between sonar readings is used as a quality measure. For each point, a first difference operator (a discrete derivative) is computed from the difference in depth of the point to left and to the right. That is, if the depth readings are denoted by a sequence, $D(k)$, then the quality measure, $Q(k)$ is given by:

$$Q(k) = |D(k-1) - D(k+1)|$$

This quality measure has an inverse sense; values near zero are "good" quality, while larger magnitudes are "less good."

For each line segment, a scan is made for the depth points which are furthest apart, and have a quality measure below a threshold. For the sonar described above, a threshold difference in depth of 1/2 foot was found to work well. This tolerance was selected based on geometric calculations which are beyond the scope of this paper. If two such points are found, then the line equation for the segment is recomputed using these two points. These new line equations then permit a readjustment of the vertex locations between adjacent line segments.

3.4 Adjusting the Vertex Locations

Vertices which are shared between two line segments are referred to as "connected" vertices. Although sonar beams are very poor at measuring the position of corners, the location of a corner can be determined with good precision by computing the intersection of connected line segments.

Whenever the recursive line fitting procedure divides a group of points, the resulting pair of line segments will share a common endpoint. In such a case, the position of the shared vertex is computed from the intersection of the line equations which express the two lines. This has been found experimentally to yield corner locations whose position accuracy is close to the depth resolution of the sonar ranging device. It is important to accurately detect the locations of corners because these points are used to correct for errors in the estimated position of the IMP that arise due to wheel slippage.

The result of this sequence of operations is a list of line segments. These line segments comprise the Sensor Model which is used to verify that the IMP is not about to collide with an obstacle, to correct errors in the IMP's estimated position and to update the Composite Local Model.

4 The Composite Local Model

As noted above, the Composite Local Model is at the core of the world modeling and navigation system. Three functions based on the Composite Local Model have been found to have very wide utility throughout the navigation system. These functions are:

Visible	Is the point P visible from the location L? If not, what is the index of the nearest Composite Local Model line segment which blocks it, and what is the location of the intersection point between this line segment point and the line segment from L to P?
FreePath	Is it possible for the IMP to pass from location L to location P? If not, what is the index of the line segment that gives the nearest collision? FreePath is implemented as sequence of calls to Visible along parallel lines.
Correspond	What is the index of the line segment in the Composite Local Model which corresponds to a given line segment?

4.1 Representing the Composite Local Model

The Composite Local Model is represented as a list of directed line segments. Each line segment contains two vertices which are ordered in a counter-clockwise direction. Each vertex is labeled as concave, convex or disconnected. If the vertex is shared with another line segment, a pointer is given to that line segment. To save time in calculations, the line equation and the angle of the vertex are also stored in the structure.

In addition to the line segment information, each segment also has a State and a Type. The State represents the confidence that the system has in the existence of that segment. At the current time, the State is represented by integers ranging from 1 (transient) to 5 (stable and connected). The Type number represents the source of the segment. There is a precedence between sources of segments to resolve the type when a segment is given by more than one source.

4.2 Matching the Sensor Model to the Composite Local Model

The correspondence between line segments in the Sensor Model and line segments in the Composite Local Model is needed to correct errors in the estimated position and to update the Composite Local Model. Correspondence matching is also used in introducing line segments into the Composite Local Model from the Global Model and from the contact sensor, for keeping track of the "current" line segment during active learning, and for a variety of other spatial reasoning functions.

The Sensor Model is matched to the Composite Local Model in two stages. In the first stage, the best correspondence is found for each line segment in the Sensor Model by making a call to the function "Correspond." This list is then scanned to determine the Sensor Model line that has the best correspondence to each segment in the Composite Local Model. This second correspondence list, from the Composite Local Model to the Sensor Model, is then used for updating the Composite Local Model.

4.3 The Correspond Function

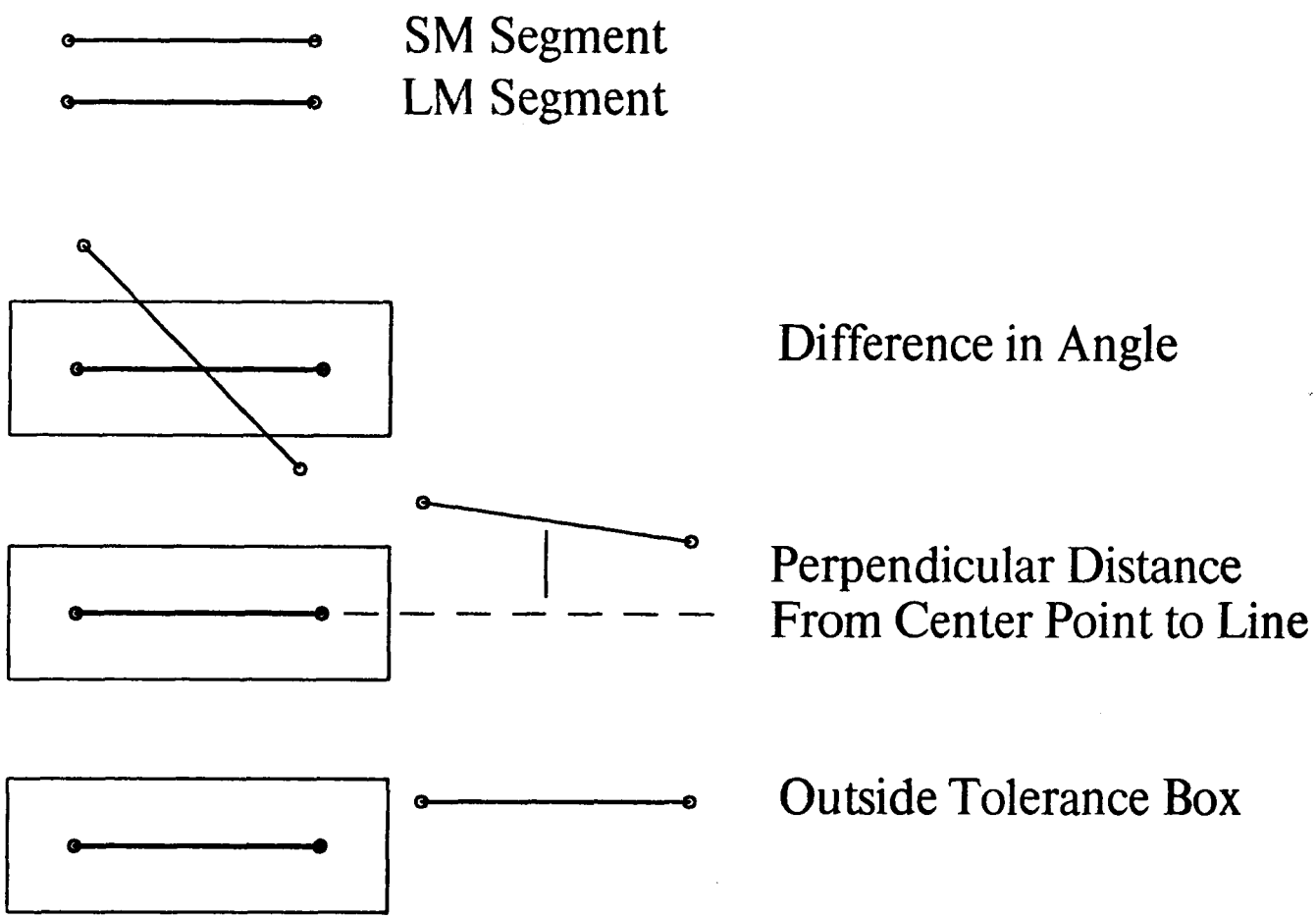

Figure 5: The tests used in determining correspondence

The function "Correspond" is a general purpose function for determining which line segment in the Composite Local Model has the best correspondence with a given line segment. A call to Correspond is made for each segment in the Sensor Model. The Correspond function is organized as a sequence of tests of increasing cost based on the attributes of orientation, position, and length. The correspondence problem is made very simple by assuming that the position and orientation of a segment are known within some tolerance. In the case of the Sensor Model, this assumption is justified because the IMP has kept track of its estimated position using wheel encoders as it moves. The required error tolerance in the estimated position can be reliably estimated.

The sequence of tests used by the correspondence function are illustrated in figure 5. In this figure, LM denotes Composite Local Model, while SM denotes the line segment for which correspondence is sought. For a given segment SM the following test are computed for each segment LM in the Composite Local Model. If a segment LM fails any test, then the process advances to the next segment in the Composite Local Model.

1. Is the difference in angle between SM and LM less than a tolerance (currently 15°)?

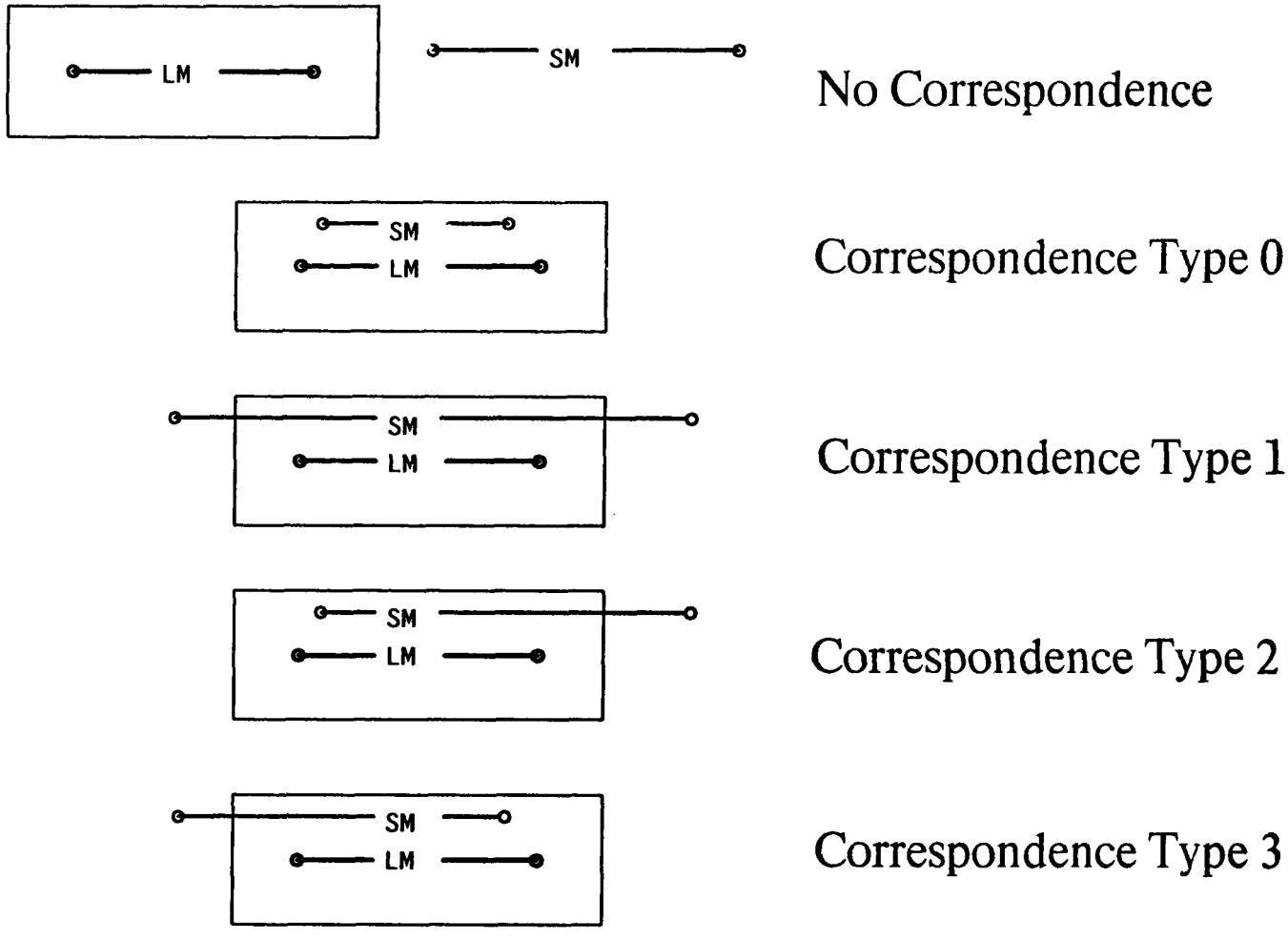

Figure 6: The correspondence types

2. Is the perpendicular distance from the center of SM to the line equation of LM less than a distance tolerance (currently 2.0 feet)?

3. Does SM pass through a box formed around LM? This is a fattened box, formed by adding a tolerance (0.5 feet) to the largest x and y coordinates of the segment LM. There are 5 possible outcomes, illustrated in Figure 6:

   a. no overlap (segment rejected),

   b. both endpoints of SM inside box (Correspondence Type 0),

   c. both endpoints of SM extend outside box (Correspondence Type 1),

   d. first endpoint of SM extends outside of box (Correspondence Type 2), and

   e. second endpoint of SM extends outside of Box (Correspondence Type 3).

The correspondence types are used to extend the Composite Local Model segment during the Update process.

4. Is the segment LM the longest found so far?

The correspondence function provides both the index of the corresponding segment and the type of correspondence.

4.4 The Update Process

The update process is the mechanism by which line segments from the Sensor Model, Global Model and contact sensor enter and refine the Composite Local Model. During each sonar scan, segments from the Sensor Model are matched to the current Composite Local Model, and the result of matching is used to update the Composite Local Model. In addition, as the IMP approaches a pre-learned place, segments which are visible in the Global Model from that place are also matched to the Composite Local Model, and the result is used to update the Composite Local Model.

The main functions of the update process are:

1. to increase the confidence state of transient segments for which there is a corresponding segment in the Sensor Model,

2. to decay the confidence of segments which should be visible, but for which there is not a corresponding segment in the most recent Sensor Model,

3. to add newly observed Sensor Model segments and segments recalled from the Global Model to the Composite Local Model,

4. to refine the vertex position of segments which are "reinforced" by the Sensor Model.

4.5 Marking the Visible Segments in the Composite Local Model

Segments in the Composite Local Model for which there is no correspondence are only modified under two conditions:

- The segment was marked as visible during construction of the Sonar model.

- The nearest point on the line segment is more than a given distance from the IMP's current position.

The second condition is a simple mechanism by which segments "fall of the end of the world" as the IMP moves away from them. The actual distance is relatively unimportant as long as it is beyond the sonar range and the current area of local navigation. Of course, the larger this distance, the more "extra" segments the system has to consider on each calculation.

The first condition establishes a "visible horizon" for the IMP. As each point is added to the Sensor Model, the function Visible is called for a point at the direction of the beam and the range of the sonar, to determine which segment in the Composite Local Model should be visible. If a segment is found, the difference in angle between the beam and that segment is computed. If this difference in angle is not small (< 15 degrees) then that Composite Local Model segment is marked as visible. Segments for which the angle of incidence of the sonar beam is very small are not detected reliably by the sonar and are thus not marked as visible.

4.6 Updating the Vertex Positions and Segment States

The rules for updating the confidence states of segments in the Composite Local Model have evolved by trial and error during experiments with the sonar data. The following principles that have emerged during these experiments:

- Extend unconnected vertices when there is a correspondence of types 1, 2 or 3.
- The position of connected vertices has precedence over the position of an unconnected vertex.
- A segment is more stable when both its vertices are connected.

The actual rules for state updates are implemented as case statements based on the current state and then on the correspondence type.

After the vertex positions and states of the segments in the Composite Local Model have been updated, segments from the Sensor Model, for which there was no correspondence in the Composite Local Model are added to the Composite Local Mode in the lowest confidence state (state 1). A relabeling process is then used to connect adjacent segments for which the vertices are very close.

5 Correcting the Estimated Position

Local path execution, learning and updating the Composite Local Model all depend critically on maintaining an accurate estimate of the IMP's current position. An instantaneous estimate of the IMP's position is maintained from the rotary position encoders on the IMPs wheels. This estimated position is monitored and corrected by a process based on comparing the Sensor Model to the Composite Local Model. Before the Composite Local Model is updated from the Sensor Model, the correspondence between the Sensor Model and the Composite Local Model is used to detect and correct any systematic error in the estimated position of the IMP.

As each Sensor Model line segment is obtained, the correspondence is found to the most likely line segment in the Composite Local Model. The difference in angle between these segments is then computed. When a Sensor Model has been constructed from a complete scan of the rotating depth sensor, the average error in angle is computed. This average error is computed from the difference in orientation between Sensor Model segments and the corresponding Composite Local Model segments, as illustrated by figure 7. The Sensor Model is then rotated around the position of the IMP by this average error in angle, as illustrated by figure 8, and the average error is subtracted from the estimated orientation.

Next the average error in position is computed by computing the average x and y errors between connected vertices in the rotated Sensor Model and the corresponding vertices in the Composite Local Model. This average error in position is then subtracted from the estimated position and from the position of the line segments in the Sensor Model. The segments in the Composite Local Model are then updated to include the results of matching to the Sensor Model.

For both position and orientation, the average error is computed in a two stage process in which the overall average error is first computed, and then the average is recomputed using only points which are near the average. In both cases, at least three points (or segments) near the overall average must be found or the correction is not made.

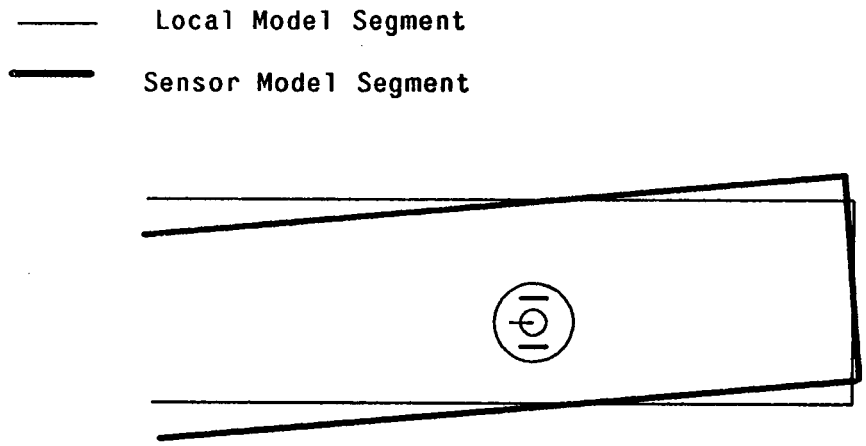

Figure 7: The orientation error is given by the average difference in angle between the sensor model segments and the corresponding local model segments.

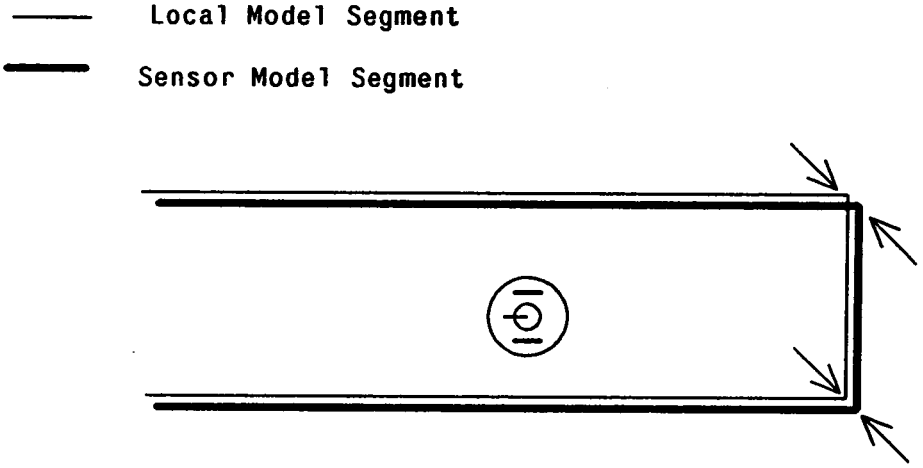

Figure 8: The position error is given by the average difference in position between the connected vertices in the sensor model and the corresponding connected vertices in the local model.

6 Navigation

The navigation system for the IMP is described in detail in [2]. Some of the salient features are described here for reference.

There are three modes in which the IMP may travel:

Learn Mode — Limited exploratory movements or user specified movements in an unfamiliar environment, with the purpose of learning the environment.

Manual Mode — User specified motion executed by local navigation.

Automatic Mode — Autonomous movement to a goal point in a familiar environment.

Learn mode is designed to permit the IMP to learn the network of places and the Global Model. Automatic mode is designed to permit the IMP to execute navigation tasks in a learned environment. Manual mode is a default mode in which the user may drive the IMP in an arbitrary manner without the restrictions required by learn mode and automatic mode.

When the IMP is in Learn mode it is particularly important that it learn an accurate Composite Local Model. This is because this model will serve as the basis for generating the convex segments for the network of places. It will also provide the segments for the Global Model. Thus, in Learn Mode, the IMP stops at short intervals (two feet) to update its Composite Local Model. Each time the IMP stops, it builds a complete Sensor Model for all of the surfaces around it.

6.1 Global Path Planning and Execution

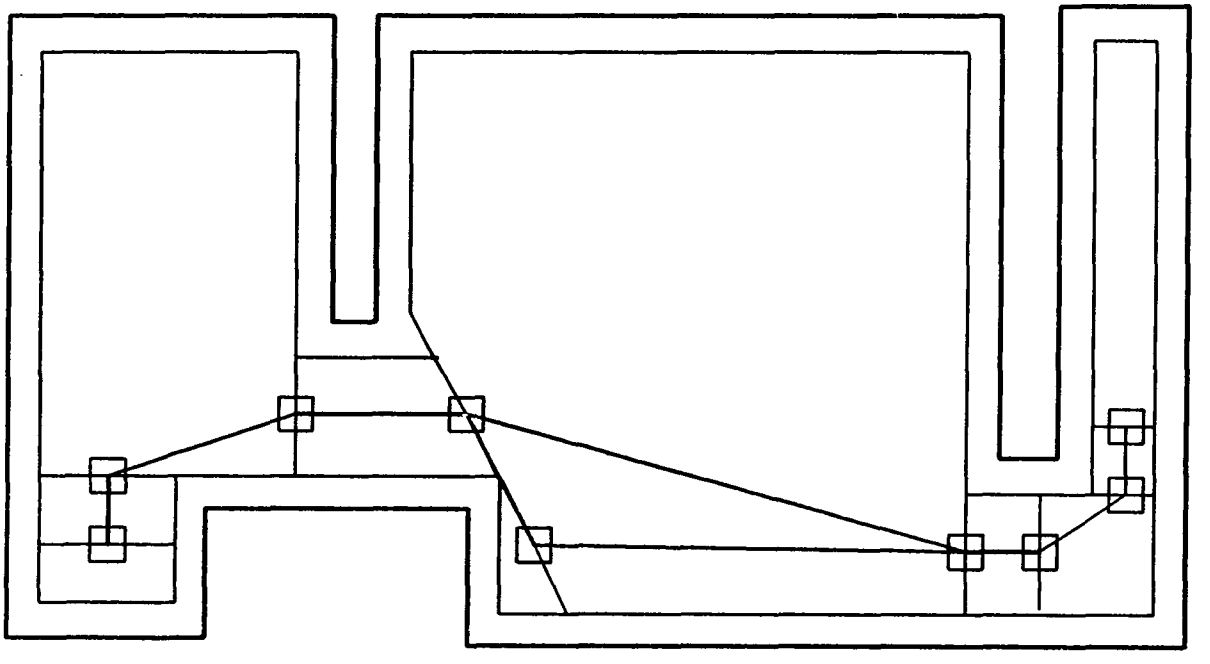

Figure 9: A set of shrunken convex regions and adits.
Adits are shown as boxes.

The pre-learned domain for the IMP is represented in two related data structures: the "Global Model" and the "network of places." Both of these data structures are acquired by the IMP in learn mode. The Global Model is the collection of line segments observed by the Composite Local Model while making a tour of the domain in learn mode. The Global Model permits the IMP to recall the surfaces which it should observe at any location in the known world.

The network of places is a three level structure which serves as a basis for global path planning. The places are obtained by dividing the free space in the Global Model into convex regions. A pair of places (or "Adits") are created for each cut that is made to partition freespace. Any two points within a convex region may be connected by a straight line entirely within the region. The user may also define named places within each convex region. The convex regions serve as "legal highways" for planning paths to any named place, as illustrated in figure 9. The adits are displaced from the cut to have the robot pass through the cut at a roughly perpendicular angle. This protects the robot from grazing the edges of door ways and tight spaces.

A global path is planned as a sequence of adits which will take the robot from its current convex region to the convex region in which the goal point is contained. The shortest path through the network of adits is determined using a version of Dijkstra's algorithm [1] which halts when a path to the desired goal place has been found. If the start (or the end) of this path leads through two adits in the same region, the first (or last) adit is dropped from the path. Global path execution is then reduced to a three step process in which the IMP moves to the first adit in the path, moves from each adit on the path to the next, and then moves from the last adit to the goal place.

6.2 Local Path Planning

The purpose of Local Path Planning is to plan a sequence of straight line paths which will take the IMP around an unexpected obstacle. A very simple recursive process is used, based on the segments in the Composite Local Model. This process plans two paths, one to the left of the obstacle and one to the right. Test are then made to see if a free path exists in the Composite Local Model from this point to the goal, and from this point to the current position. If either path is blocked, the procedure is called recursively to see if it is possible to get around the blockage. The recursion is not continued beyond three levels.

7 Summary and Conclusion

A dynamic world modeling system for an intelligent mobile robot is a system which maintains a dynamic model of the external world (the Composite Local Model) using a rotating depth sensor. The dynamically maintained Composite Local Model supports the functions of path planning, learning, path execution, and object tracking; It provides the core of a task-oriented navigation system.

Described in this paper are the processes by which the Composite Local Model is constructed and maintained using a rotating sonar, a contact sensor, and a pre-learned Global Model. The depth readings from the rotating sonar are projected into a system of cartesian world coordinates and then expressed as line segments. These line segments are matched to the Composite Local Model. The resulting correspondence is used to estimate and correct the estimated position of the IMP, and to update the contents of the Composite Local Model. These techniques form the basis for an inexpensive navigation system which is suitable for pre-learned indoor environments.

References

[1] Aho, A. V., J. E. Hopcroft, and J. D. Ullman.
Data Structures and Algorithms.
Addison Wesley Publishing Co., 1983.

[2] Crowley, J. L.
Navigation for an Intelligent Mobile Robot.
In *IEEE Conference on Applications of AI.* December, 1984.

[3] Duda, R. O. and P. E. Hart.
Pattern Classification and Scene Analysis.
Wiley, New York, 1973.