A Robust Qualitative Method for Robot Exploration and Map-Learning

Benjamin J. Kuipers
Yung-Tai Byun

April 1988

AI88-73

* Support for this work is provided by NASA, under grant #NAG9-200. A short version of this paper has been submitted to AAAI88.
ABSTRACT

We present a robust qualitative method for a mobile robot to explore an unknown environment and learn a map. Procedural knowledge for the movement, topological model for the structure of the environment, and metrical information for geometrical accuracy are separately represented in our method, whereas traditional methods describe the environment mainly by metrical information. The topological model consists of nodes and arcs corresponding to distinctive places and local travel edges linking nearby distinctive places. A distinctive place is defined as the local maximum of some measure of distinctiveness appropriate to its immediate neighborhood, and is found by a hill-climbing search. Local travel edges are defined in terms of local control strategies required for travel. How to find distinctive places and follow edges is the procedural knowledge which the robot learns dynamically during exploration stage and guides the robot in the navigation stage. An accurate topological model is created by linking places and edges, and allows metrical information to be accumulated with reduced vulnerability to metrical errors. The method, inspired by the study of cognitive maps which humans use, can be robust in the face of various possible errors in the real world, and the map description can be more useful than traditional approaches to the robot, as well as people. We describe a working simulation in which a robot, NX, with range sensors and two tractor-type chains explores a variety of static 2-D environments and we give its successful results under varying levels of random sensor error.
1. Introduction

Traditional approaches, mentioned below, to the robot exploration, navigation and map-learning, based on the accumulation of accurate metrical descriptions of the environment, are highly vulnerable to metrical inaccuracy in sensory devices and movement actuators. Brooks [1985], Kuipers and Byun [1987], and Levitt et al. [1987] discuss some of difficulties and limitations of the traditional approaches. Recent work taking a more qualitative approach [Kuipers and Byun, 1987; Levitt et al., 1987] shows great promise of overcoming the fragility of purely metrical methods. Humans perform very well at spatial learning in spite of sensory and processing limitations [Kuipers, 1979]. Many scientists [Lynch, 1960; Piaget and Inhelder, 1967; Siegel and White, 1975] observed that a cognitive map, which humans create and use, is organized into successive layers, and suggested that the basic element of a useful and powerful description of the environment is a topological description. The layered model consists of the identification and recognition of landmarks and places from sensory information processing, procedural knowledge of routes from one place to another, a topological model of connectivity, order, and containment, and metrical information of shapes, distance, direction, orientation, and distorted local and global coordinate systems. It appears that the successive layered structure of the cognitive map is responsible for humans' robust performance. Our approach attempts to apply the methods to the problem of robot exploration and map-learning.

The central description of the spatial environment in our qualitative approach is a topological model as in the TOUR model [Kuipers, 1978]. The model consists of a set of nodes and arcs, where nodes represent distinctively recognizable places in the environment, and arcs represent travel edges connecting them. The nodes and arcs are defined procedurally in terms of the sensorimotor capabilities of the robot. Metrical information is added on top of the topological model.

A place in the environment corresponding to a node in the topological model must be locally distinctive within its immediate neighborhood by one geometric criterion or another. We introduce locally meaningful "distinctiveness" measures defined on a subset of the sensory features, by which some distinctive features can be maximized at a distinctive place. We define the signature of a distinctive place to be the subset of features, the distinctiveness measures, and the feature values, which are maximized at the place. A hill-climbing search is used to identify and recognize a distinctive place when the robot is in its neighborhood. When exploring, both the signature and the local maximum must be found. When returning to a known place, a robot is guided by the known signature.

Travel edges corresponding to arcs are defined by local control strategies which describe how the robot can follow the link connecting two distinctive places. This local control strategy depends on the local environment and there may be several possible strategies. For example, in one environment, following the midline of a corridor may be reasonable; in another environment, maintaining a certain distance from a single boundary on one side is appropriate.

We have implemented and tested successfully our approach with a working simulator NX-SIM. We will discuss related research, our methods and implementation in detail, simulation results, and further
1.1 Related Research: Spatial Representation

A variety of methods to represent the structure of known environments have been developed for robot navigation, and methods to build models for unknown environments through exploration have also been studied for robot exploration and navigation. Generally speaking, there are two categories of methods used to represent an environment. Methods of the first category intend to describe the environment mainly by the location and shape of objects, whereas methods of the second category try to describe the environment mainly by free space for navigation.

Turchan and Wong [1985], Moravec and Elfes [1985], Elfes [1986], and Lozano-Perez [1981] use methods of the first category. Under the assumption of only straight line segments in the environment, accurately measured distances, and knowing the exact location of the robot, Turchan and Wong [1985] project walls and objects onto a Cartesian plane and occluded regions are known by the integration of information from several places. An attribute graph is constructed from the model. Moravec and Elfes [1985] and Elfes [1986] use a rasterized map, rather than a map using only points and lines, onto which moderately high resolution spatial information is projected by integrating several range readings, and each square of the map can be represented as probably occupied, probably unoccupied, or unknown area. Sensory errors are incorporated in this research. But they still have assumed that the correct positional information of a robot is given every time their program analyzes readings to decide values of each square. In the Configuration Space method [Lozano-Perez, 1981], a moving object is shrunk to be just a point, the "reference point", whereas obstacles are expanded depending on the shape of a moving object with respect to the reference point. However, it is computationally expensive to deal with the rotation of an object and reconstruct the obstacle expansion if it gives a different shape as the result of rotation. This method was applied to an ALV by T.S.Chang et al. [1986] to avoid an obstacle and for collision-free navigation.

Methods of the second category describe the free space of given environments in order to find a path from one place to another. Three methods of this category are Skeleton, Generalized Cone, and Convex Polygon. A skeletal graph [Rosenberg and Rowat, 1981; Meng, 1987] using Skeleton has been studied to find a path from one location to another. The path in this representation method can correspond to one of local navigation strategies in robot-exploration research, called Aisle-center [Kadonoff et al.1986], Passage-on-the-middle [Kuipers and Byun, 1987], and Safest [Miller, 1985]. In the ALV world, Lowrie et al. [1985] and Wallace et al. [1985] have developed methods for an ALV to navigate on the center line of the street.

Brooks [1982] uses Overtapping Generalized Cones to represent free space. In this method, it is assumed that the moving object is a convex polygon and obstacles are represented as unions of convex polygons. Every sweepable volume of free space is represented by parameters of generalized cones, and finding a path is done by comparing this sweepable volume of free space with the swept volume of a
moving object. Brooks [1985] also proposes to use freeways for free space as one of the map primitives. The freeways are elongated regions of free space which describe a large class of collision-free straight line motions of the object to be moved. Kadonoff et al. [1986] use this as one of their models to describe the world.

The Non-overlapping Convex Polygons method is used to represent each partition of free space where each partition is constructed as a convex polygon [Laumond, 1983; Giralt, 1983; Chatila and Laumond, 1985]. In this method, they also assume that only polyhedral objects are in the given environment. However, they have also considered building a semantic model from the connectivity graph in which a node corresponds to one convex polygon. Kuan et al. [1985] use both Generalized Cone representation for channels and Convex Polygon representation for passage regions.

Besides the methods mentioned above, Iyengar et al. [1985], Rao et al. [1986] and Weisin [1987] build a map which is based on points and paths connecting points. The spatial graph consists of edges (travel paths) and nodes (stopping points, turning points, or path intersections). The graph is incrementally learned as traversals (navigation) are made. Each polygon constructed by the graph is described by an obstacle-free polygon, an obstacle polygon, or an unexplored polygon. They also obtain a Voronoi diagram [Lee and Preparata, 1984] from the spatial graph to find a node from which it can follow an edge in order to go to the destination. Kadonoff et al. [1986] use several methods together: the Grid Model [Moravec and Elfes, 1985; Elfes, 1986], the Segment Model [Crowley, 1985], the Freeway Model [Brooks, 1985], and the Polygon Model [Miller, 1985], although only two of them are implemented yet. Brooks [1985] discusses the problems of the traditional methods, and proposes to use a relational map as a model of the world rather than using 2-D coordinate system. The relational map is not yet fully specified, but includes such features as metrical range bounds of distances between pairs of places.

1.2 Related Research: Exploration Strategy

There is not much literature discussing both how to explore and the problems resulting from the movement. This is because most researchers in developing their methods assume that they know the current location of a robot exactly in terms of (X, Y) in a 2-D coordinate system. But it is not allowed to make this assumption in the real world practically. In the ALV research, how to follow a street or how to avoid collisions have been studied, but neither how to explore an unknown world nor how to clearly build a map has been studied yet.

Chatila and Laumond [1985] address the problem of inaccurate metrical information of movement and try to solve it by introducing a "fading" function. Brooks [1985] develops the same technique independently. But we wonder if it is absolutely necessary to know all the time the location of a robot in terms of X and Y coordinates. People do not use a global coordinate system, but show good performance in exploration and navigation.

Kadonoff et al. [1986] use several local navigation strategies to avoid unexpected obstacles along a
path with no knowledge of the robot’s position in a known world. Several sensors are used to perform the
local navigation strategies which are Obstacle Avoider, Path Follower, Beacon Tracker, Wall Follower,
Aisle Centerer, and Vector Summer. One of them is dynamically chosen at any time by an arbitor using a
production system. This is similar to our approach for exploration. Since they use several sensors, the
metrical information could be reliable in a sense that their purpose of local navigation strategies is merely
the obstacle avoidance along a path. But local navigation strategy information is neither used in describing
the world nor is saved for later use. It must be computed and decided every time. By knowing the robot’s
position using several different position estimation algorithms, it can plan a path and move to a known
place. Although dynamic navigation strategies are used, it is only for obstacle avoidance, as in the ALV
world, and the spatial representation of the world is still based on only metrical information.

In the layered control system proposed by Brooks [1986], our qualitative exploration approach and
building of a map correspond to level 2, Explore, and level 3, Building maps. Level 0, Avoid Objects,
corresponds to Obstacle Avoider in the work of Kadonoff et al. [1986], and it is naturally implemented in
our approach by choosing and performing a proper local control strategy for a path and a hill-climbing
search for a distinctive place. However the map representation method and exploration strategy in Brooks’
work are different from ours. Because his method is still based mainly on metrical information. There is no
further discussion about local exploration strategies as in our approach and the approach of Kadonoff et al.
[1986].

Levitt et al. [1987] propose qualitative methods of place definition and navigation based on visual
landmark recognition. They argue the weakness of traditional navigation techniques and show the pos-
sibility of navigation and guidance using a coordinate-free model of visual landmark memory, but not using
an accurate map or metrical information. Although they have developed the methods for navigation and we
have developed our method for learning the structure of the environment separately, there are several
similarities. But their place definition is based on a region, whereas our place definition is based on the
distinctive place and its neighborhood. Their methods are most appropriate in environments where several
point-like landmarks are easily observable.
2. A Topological Model with Procedural and Metrical Information

The basic structure of a map, in our approach, is the topological model of which nodes are distinctive places and arcs are travel edges. We discuss how to define distinctive places and travel edges, and their procedural and metrical descriptions with a robot instance, NX.

2.1 A Robot Instance NX

We hypothesize that our approach is supported by any sensorimotor system that provides sufficiently rich sensory input, and takes sufficiently small steps through the environment. For the simplicity and concreteness, we currently define a specific instance of a robot NX which has sixteen sonar-type distance sensors covering 360 degrees with equal angle difference between adjacent sensors, two tractor-type chains for movement, and an absolute compass for global orientation. Thus the input to NX is a vector of time-varying, real-valued functions \([S_1(t), S_2(t), ..., S_{16}(t), \text{Compass}(t)]\). Although we use NX to test our qualitative method, our approach does not depend critically on the choice of sensors and movement actuators.

2.2 Distinctive Places

![Figure 1. Distinctive points in a neighborhood](image)

In order to have the nodes of the network-structured topological model we need to look for distinctive places (DPs). If we consider the geometry of a simple 2-D local neighborhood in Figure 1, we can argue that the dotted lines define a set of places that are qualitatively distinctive for one reason or another. There is clearly a place which is the most distinctive compared to its surroundings. Our approach attempts to find a suitable criterion for defining a maximally distinctive place in any given neighborhood. In environments dominated by obstacles and extended landmarks, we believe that a map based on DPs and connecting edges provides a more robust topological representation than, for example, regions related by adjacency. In an environment dominated by remote, point-like landmarks, the reverse may be true [Leviitt et al., 1987].
In order to formulate locally meaningful "distinctiveness" measures, we need to determine which sensory characteristics provide the distinguishing features by which a place becomes locally distinctive. We hypothesize that any reasonably rich sensory system will have distinctiveness measures that can be defined in terms of low level sensory input. Note that it is not necessary for a place to be globally distinctive; it is only necessary to be distinguished from other points in its immediate neighborhood.

A set of production rules is used to decide whether NX is in the neighborhood of a DP and what distinctive features can be maximized in the neighborhood. Each rule consists of assumptions and a decision for the distinctive features. Here is an example:

(defrule DP-r10
  (if (>= (number-of-objects) 3)
      (not (all-objects-far-away))
      (not (there-is-wide-open-space)))
  (then (am-I-in-neighborhood-dp is 'dp-syrm-equal))
)

Once NX knows what distinctive features can be maximized locally in the neighborhood of a distinctive place, NX performs a hill-climbing search around the neighborhood looking for the point of maximum distinctiveness (e.g., minimizing differences of distances to near objects, if DP-r10 is true). The distinctive place can be defined in the topological model by local maxima of the distinctiveness measures. When the distinctive place is identified, it is added to the topological model with the distinctiveness measures, connectivity to edges, and metrical information.

The individual distinctiveness measures are an open-ended, domain- and sensor-specific set of measures. For our current robot, the measures we can define include the following.

- Extent of distance differences to near objects.
- Extent and quality of symmetry across the center of the robot or a line.
- Temporal discontinuity in one or more sensors, given a small step.
- Number of directions of reasonable motion into open spaces around the robot.
- Temporal change in number of directions of motion provided by the distinct open spaces, with a small step.
- The point along a path that minimizes or maximizes lateral distance readings.

In the current simulation, we have considered some of these. Several distinctive places listed below have been found by one or more of the distinctiveness measures.

- A place which minimizes the distance differences to near objects.
- A place which maximizes local symmetry across the center.
- An ending place (only one direction of motion).
- A place which is defined by a temporal discontinuity (change of angles to nearest objects).
- A decision place (three or more directions of motion).
- A place of a minimum lateral distance to an object in a wide open space.

We summarize the levels of description of DPs: (An example is given in Section 3.)
- **Procedural knowledge** for a DP: Ability to recognize the neighborhood, knowledge of what features can be maximized in the neighborhood, and ability to perform the hill-climbing search to get to the DP. Learned in the exploration stage and used in the navigation stage.
- **Topological descriptions** of a DP: A node in the topological model, connected to edges and other DPs. Added to the topological model when it is found and possibly updated during the process of constructing the model.
- **Metrical information** about a DP: Local geometry like directions to OPEN-SPACE, shape of near objects, distances and directions to objects, etc. Continuously accumulated in the exploration and navigation stage and averaged to minimize metrical error.

### 2.3 Travel Edges

Travel edges are defined in terms of local control strategies (LCS). Once a DP has been identified, the robot moves to another place by choosing an appropriate control strategy. While following an edge with a chosen strategy, the robot continues to analyze its sensory input for evidence of new distinctive features. Once the next place has been identified and defined, the arc connecting the two DPs is defined procedurally in terms of the LCS required to follow it.

![Figure 2. Movement with Error](image)

The edges followed during exploration are defined by some distinctiveness criterion that is sufficient to specify a one-dimensional set of points. Therefore, following our control strategies, the robot will follow the midline of a corridor, or walk along the edge of a large space, but will not venture into the interior of a large space, where the points have no qualitatively distinctive characteristics.

As shown in Figure 2, when the robot is following a known edge from one node to another, it starts by
using the hill-climbing algorithm to locate itself at the DP corresponding to the first node. It then follows the LCS associated with the arc and ends up somewhere in the neighborhood of the second place. Then the hill-climbing algorithm brings it to the DP corresponding to the second node. This method uses continuous sensory feedback to eliminate cumulative error.

A set of production rules to decide a proper LCS depending on the current sensory information is given to NX. An example of a rule is given below.

(defrule LCS-rule10 ()
  (if (>= (number-of-objects) 2)
      (two-walls-are-near-to-each-other)
      (two-walls-are-almost-opposite-directions))
  (then (proper-LCS is 'pass-on-the-midline)))

The current local control strategies are:
- Follow-Midline
- Walk-along-Object-Right
- Walk-along-Object-Left
- Blind-Step

In summary for edges: (An example is given in Section 3.)
- **Procedural knowledge**: Ability to choose and perform a proper LCS and knowledge of which control strategy defines the edge. Learned in the exploration stage and used in the navigation stage.
- In the Topological model: An edge with direction, connected to two end-places. Added to the topological model when the second end-place is found.
- **Metrical information**: Curvature, distance, change of orientation, lateral width while traveling, etc. Continuously accumulated in the exploration and navigation stage and averaged to minimize metrical error.

### 2.4 Position Referencing Problem

While NX explores the given environment, it needs to know the current position in the map. In traditional approaches, the current position is represented by \((X, Y)\) in their coordinate frame. As noted in Section 1, it is not easy to get correct coordinates.

In our method, the current position is described topologically rather than metrically (The metrical representation can be added when the correct topological model has been built and rich metrical information has been accumulated). When NX is at a distinctive place, the current position is described by the current place name, the current orientation in degrees, and a travel edge through which NX has come to the
current place from the previous place. When NX is on an edge, the current position is described by the previous place name, the current orientation, and the current edge. If the edge is unknown yet, it uses just an indication "ON-EDGE".

2.5 Matching Process to Determine the Current Position

When NX reaches a place during its exploration, the identification of the place is the most important task. If a place has been visited before and NX comes back to that place, NX should recognize it. A new place must be recognized as new, even if it is very similar to one of the previously visited places. Our matching process is done topologically as well as metrically.

While NX explores, it uses an exploration agenda to keep the information about where and in which direction it should explore further to complete its exploration. If (Place1 Direction1) is in the exploration agenda, it means that NX has previously visited Place1 and left it in some direction(s) other than Direction1. Therefore, in order to delete (Place1 Direction1) from the exploration agenda, NX should either visit Place1 later and leave in the direction Direction1, or return to Place1 from the opposite direction.

When NX gets to a place in the exploration stage, the exploration agenda can be either empty or not empty. If the exploration agenda is empty, it means that there is no known place with directions which require further exploration. Therefore the current place must be new, unless NX has intentionally returned to a previously known place through a known edge. If the exploration agenda is not empty, the current place could be one of the places saved in the exploration agenda. This is only possible when the current place description is similar to that of a place saved in the exploration agenda, and the difference between the current orientation and the direction saved on the agenda is approximately 180 degrees.

The current and stored place descriptions are compared metrically, allowing a certain amount of looseness of match to provide robustness in the face of small variations in sensory input. But mismatching is possible. If there is any possibility, the topological matching process is initiated. From the topological model and procedural knowledge of edges and nearby DPs, the rehearsal procedure [Kuipers 1985] is activated to test the hypothesis that the current place is equal to a previously known place. NX constructs routes between the known place and adjacent DPs. It then tries to follow the routes and return to the current place. If the routes performed as predicted, then the current place matches the previously known one, and NX has identified the current place. If not, then the current place must be a new place with the same sensory description as the old one.

For any fixed search radius of this topological match, it is possible to construct an environment that will yield a false positive match. However, if there are a reference place that is somehow marked so as to be globally unique (e.g., "home"), false positives can be eliminated.
2.6 Robustness

Metrical matching of the current sensory information with the saved information in the traditional approaches is not a simple task. There are several problems. Geometrical description methods used currently for randomly shaped objects have limitations in terms of accuracy and computation costs. It is almost impossible to get metrically correct information if there are errors. And therefore it is hard to build a metrically accurate map of an unknown environment through exploration. Without a metrically accurate map, the matching cannot be done properly with traditional methods. Although a metrically accurate map is given to the robot, it is not easy for it to know all the time the accurate location of itself in the frame in which the map is given. With noisy data, it is not easy to extract the accurate shape of objects in the environment unless restrictions on the shapes of objects are given (e.g., only allow straight lines). When what is to be compared in the saved information with the current information becomes unknown for some reason, it will cost a lot to find out what is to be compared with the current sensory data.

In our approach, once NX is located in the neighborhood of a distinctive place, the hill-climbing search takes it to the most distinctive place in the neighborhood. Although there are sensory and movement errors, continuous sensory feedback to movement will not fail to bring NX very near to a distinctive place. It is not necessary for NX to be located at the same \((X, Y)\) coordinates. It is enough to be located where some distinctiveness measures can have almost maximum values compared to the values when NX is only in the neighborhood of a distinctive place. As we can see in Section 3 where several near locations are classified as one distinctive place.

If there was a movement error where two encoders count the rotation of chains and one encoder is miscalibrated, a purely metrical map could be completely distorted or absolutely wrong, and therefore the result would make a robot get lost wherever it is. In our method, a proper local control strategy guides NX to avoid collisions and helps it move reasonably depending on the local geometry. Although there are sensory and movement errors, it will take NX to the neighborhood of a distinctive place. And it remembers what LCS is appropriate in that region. Since our model is based on this qualitative description, metrical errors do not impact on the model so seriously as on the traditional approaches. Once NX is in the neighborhood, the hill-climbing search guides NX to find another distinctive place.

If a local control strategy and set of distinctiveness measures could be chosen uniquely with a particular set of sensory data and could be performed properly, the topological model of an environment could be defined uniquely. We will give the results of several simulations with several different environments in Section 3. The greater the amount of random error that is considered, the less efficient NX’s movement is. But the topological model is still constructed successfully.

As we mentioned in Section 2.4 and Section 2.5, our approach uses a loose matching process for metrical information and the rehearsal procedure for a topological matching process. The procedural knowledge for distinctive places and travel edges, the metrical matching process with looseness, and the topological matching process make our approach robust in the face of metrical errors.
3. Simulator and Results

We have developed a simulation system NX-SIM. Our simulator is implemented on the Symbolics 3600 and is written in Common Lisp. Figure 3 is a copy of the simulation window. We explain our simulator NX-SIM and its system architecture, and give exploration results in this section.

3.1 NX-SIM

NX is represented on the simulation window as a triangle. The sharpest corner of the triangle denotes the direction of NX's forward movement. The range of measured distance of each sonar sensor is from 18 to 220 pixels, and a minimum safety distance is 30 pixels. At the right top corner of the simulation window, measured distances for the sonar range-finders are displayed. F, R, L, and B stand for Front, Right, Left, and Back, respectively.

Status indicators:

NBD: a neighborhood of a DP  REH: Rehearsal  Am : current place
EDG: on an edge  NAV: Navigation  Was : previous place
EXP: Exploration  To : destination place

(Values of several distinctive measures are shown on the left top, and measured distances with errors are shown on the right top.)

Figure 3. NX-SIM Window
The metrical lines in the "Measured Distances" box in the upper right corner show the 16 sensor readings at the current instant. The length of the line represents the sensor reading perceived by the robot. In this example, the sensor readings are subject to a ten percent random error, so the true distance is indicated by an "x" (perceived only by the researchers). In the figure, ten percent random error is incorporated. This error simulation is based on Hickling and Marin [1986], Walter [1987], Flynn [1985], and Drumheller [1985].

At the top left corner, the result of analysis of each distinctiveness measure considered in the current simulation is displayed (it scrolls horizontally). In Figure 3, NX was located near Place1 and it was oriented to the left. After moving forward for a while, NX recognized that it was in a neighborhood and turned around and performed the hill-climbing search for Place1. The first peak on the second row at the top left corner shows the symmetric and equal distance analysis while it tried to find Place1. The second, the third, and the fourth peaks correspond to Place2, Place3, and Place4, respectively.

In the top middle, an energy pair supplied to two tractor-type chains is shown while it is moving. If the energies for the left chain and right chain are equal and greater than zero, NX moves straight forward. If both of them are positive and one is larger than the other, the result is forward motion and turning slowly to the right or to the left. If one is positive, the other is negative, and their magnitude are equal, NX makes a turn action in place to the right or to the left by some amount. While NX moves, it decides the energy pair continuously by analyzing the sensory information.

While NX is moving from one place to another, it keeps a record of the number of rotations of each chain, called Travel-history. This information is used to give rough information of the shape of the path and the relative position of two places. For instance, Travel-history ((10 10)) means that NX has moved straight, whereas Travel-history ((10 8)) means that NX has moved forward and turned slowly to the right. Since there is a round-off error in order to make a list of integers for Travel-history, we can say that a random error for the movement has been considered. But notice here that this error does not make our method weak at all. Because, this error possibly makes a metrical map distorted a little bit, but the topological model does not change depending on this error. Therefore, although we are able to simulate serious errors of the movement actuator, it is not done at the current simulation.

The current status of NX is also shown in the top middle. NBD means that NX is in the neighborhood of a distinctive place. EDG means that NX travels an edge. EXP means that NX is in the exploration stage. REH means it activates the rehearsal procedure to recognize a distinctive place. NAV means that NX is in the navigation stage. There are two navigation cases. The first happens during the exploration stage. When NX has traveled all directions from the current place and there is at least one place which requires more exploration, then it navigates to the place and continues its exploration. The second happens when NX believes that it has finished the exploration. After finishing the exploration, it navigates randomly to some place and continuously accumulates the metrical information. This process will be necessary to handle a dynamic world later. As mentioned earlier, the exploration agenda gives the places and what
directions NX should further explore.

3.2 System Architecture

The overall system architecture of our simulation system NX-SIM is presented in Figure 4. However we need to emphasize again that our exploration strategy does not depend on the types of sensors and movement actuators at all. The modules shown in the bottom half of Figure 4 depend on the kinds of devices and those modules shown in the top half do not. The function of each module in the figure and the modules’ interactions are described below.

![Diagram](image)

**Figure 4.** Overall system architecture
(Upper half independent of sensors and movement actuators.)

- **Sensory Device:** Senses the world and passes sensory data to the Processing Sensory Data module. In NX-SIM, sonar range-finders are used.
• **Movement Actuator:** Makes a robot move in the world. It is controlled by the Local Movement Control module. In NX-SIM, two tractor-type chains are used.

• **Processing Sensory Data:** Analyzes and understands sensory data qualitatively and quantitatively. This information is passed to the Integration and Modeling module to build a map. It also passes the information to the Local Movement Control module so that the robot decides what local exploration strategy should be used and implements the strategy. It tests the rules in the Defining Distinctive Place module to find a distinctive place.

• **Defining Distinctive Place:** Rules to define distinctive places or when the robot starts looking for distinctive places. Production rules are used with a sequential control of rule firing. Rules in this module can be easily substituted by other rules depending on the devices and users' intentions. Changing this module does not affect our exploration strategy.

• **Local Movement Control:** Controls local movement for a path and a distinctive place. It continuously decides the Energy-pair to be delivered to the Movement Actuator. A part of this module is a set of production rules to decide a proper local exploration strategy which is used in the exploration stage. The control of firing rules is also sequential. When the system updates a map, this module passes the local control strategy to the Integration and Modeling module.

• **Integration and Modeling:** Integrates necessary information and builds a map. The map is built in this module and the map information is passed to the Exploration and Navigation Control module and the Route Planner module. The annotation of a path is made by the information from the Local Movement Control module, and the annotation of a distinctive place is made by the information from the Processing Sensory Data module.

• **Exploration and Navigation Control:** Controls the global exploration and navigation. It has rules for the global exploration and maintains the exploration agenda to complete the exploration in the given world. When it is necessary for the robot to go to a known distinctive place through known paths, it passes the destination place to the Route Planner module and gets the information of a route to the place as an answer. Instructions for each segmented local movement are passed to the Local Movement Control module.

• **Route Planner:** Plans a route from the map information when the destination is given. The breadth-first search in the topological model is used to make a route. When the metrical information is considered for planning, the A* algorithm will be used. This module is used only when the robot needs to navigate a previously explored region in the given world.
3.3 Simulation Results

We explain how NX explores and builds a map in detail with the environment, shown in Figure 3. We show the graphic exploration results of three different error rates: error-free case in Figure 5a, five percent error in Figure 5b, and ten percent error in Figure 5c. In each case, NX starts near P1 in order to compare the results. The starting place is marked S in each figure, Pi means Place-i and Ei means Edge-i. We will trace NX’s movement with Figure 5c. NX constructs the correct map successfully in all three cases, but careful examination of figures 5a-c will reveal subtle differences.

NX starts its exploration from S between P1 and P7 in Figure 5c. It chooses Pass-on-the-midline control strategy and moves downward. Because of sensory error, it does not initially recognize near P1 that it is in a neighborhood of a distinctive place. It performs the local control strategy continuously. But while continuing to perform Pass-on-the-midline control strategy, it recognizes a qualitative change and so it performs a hill-climbing search to minimize the difference of distances (i.e., equal distances) to near objects. This search turns it around, converges on a local maximum, and defines the place P1. If we take a look at Figure 5a and 5b, we do not see this kind of backtracking around P1. NX recognizes the neighborhood sooner than in Figure 5c.

Figure 5a. Exploration result with no random sensor error
Figure 5b. Exploration result with five percent random sensor error

Figure 5c. Exploration result with ten percent random sensor error

Once NX finds P1, it records P1's information in the map as follows.

PLACE : Name = P1
Procedural = Symm-Equal (i.e., Symmetry and Equal distance)
Topological = Nil
Metrical = 
Direction-requiring-more-exploration: 345 and 282 degrees
Angle and Distance to Objects: (70 deg. 46 units)
(317 38)
(160 51)
The procedural information indicates the distinctive features of P1. There is no topological information for P1 at this time. The metrical information describes both the directions NX should explore further and the near objects. There are two directions in which NX can go from P1. That information is saved in the exploration agenda. If there is no particular reason to choose an indicated direction, it chooses the direction which requires the least rotation. When NX finds P1, the rotation angle to the direction toward P2 happens to be less than that toward P7. Therefore it rotates to the direction toward P2 and deletes the information corresponding to the direction from the exploration agenda.

The first thing to do is to get out of the neighborhood while moving to an open space. While NX is moving, it can see two walls on both sides and chooses the Pass-on-the-midline control strategy to pass through the current environment. While moving along the edge, it gathers metrical information about the edge such as distance, shape, width of the edge, change of the width, and so on. Then NX finds the second distinctive place, P2, which is characterized by Temporal-discontinuity. Two tasks are being performed at this point. The first task concerns the edge, E1, and the second task concerns P2.

**EDGE : Name = E1**

- **Procedural : Pass-on-the-Midline**
- **Topological :** from P1 to P2
- **Metrical :**
  - **Travel-history :** 
    - \((\text{DIR}^+ \ (8 \ 10) \ (6 \ 6) \ (11 \ 9) \ (18 \ 18)))\)
  - **Distance :** (43)
  - **Lateral-width :** 
    - \((\text{DIR}^+ \ (81 \ \text{ALMOST-STD} \ 43)))\)
  - **Minimum-width :** 80
  - **D-Orientation :** 
    - \((\text{DIR}^+ \ (-8)))\)

Once P2 has been defined, the above is recorded in the map for E1. The procedural information indicates the control strategy used for the edge. The topological information says that E1 connects P1 and P2. Notice here that a topological connectivity direction, from where to where, is recorded clearly. Much metrical information is saved. The travel-history indicates the number of steps of each chain from the neighborhood of P1 to that of P2. DIR+ specifies the topological direction from P1 to P2. From the travel-history, we can imagine a rough shape of this edge. The distance between two places is the sum average of each element in the travel-history. The profile of the edge is described by Lateral-width and Minimum-width in the current simulation. The description \((\text{DIR}^+ \ (81 \ \text{ALMOST-STD} \ 43)))\) means that when NX moves from P1 to P2, the distance between the two walls is approximately 81 units and it is almost steady while it moves approximately 43 steps. A list of Width (how many units), Change-indication (ALMOST-STeady, INcreasing, DEcreasing), and Change-interval (how many steps) is an element of the description. The minimum distance between the two walls along the edge is described by Minimum-width. For E2, it is 80 units. D-Orientation gives the net change of orientation in degrees when NX begins traversing the edge, (i.e., leaves the neighborhood of P1), and finishes traversing the edge (i.e., arrives at the neighborhood of P2).

As well as saving the edge information and the definition of P2, it needs to update the topological information of P1. When it found P1, there was no need to consider the connectivity. But right now it knows that E1 leading to P2 is connected to P1. It updates the topological information of P1. Generally, NX
updates the topological information of places whenever it finds a new edge.

While NX leaves P2, NX thinks that maintaining the same distance to an object on the right side and an object on the left side is the appropriate local control strategy. You can see a line stretching to the direction between E2 and E6. But it soon realizes that Move-along-object-on-left or Move-along-object-on-right are more appropriate. Because it prefers smaller rotation angles, it chooses Move-along-object-on-left. We can see a significant difference between this and what happens around P2 in the no-error and the five percent error cases, as the result of the different amount of errors. In these two cases, NX recognizes the wide space in front and chooses Move-along-object-on-left immediately after it leaves P2. The difference here is the result of the different amount of errors. We can see a dynamic reaction to errors in the sensory information. But it does not hurt NX’s exploration. The exploration process recovers from temporary errors, and is successful in all three cases. When it decides to move along an object on the left side, it saves a list of P2 and the upward direction in the exploration agenda for further exploration.

Then NX finds P3, E3, P4, E4, P5, E5, and P6 in each figure. P6 is defined by Temporal-discontinuity as P2. It explores downward to P2 in the five percent case and the ten percent error case, whereas it explores left side (right side from NX’s view) to P3. This depends merely on the orientation when it finds P6. In the ten percent error case, it moves along E6 and finds a place which looks similar to P2. Besides that, the information saved in the exploration agenda when NX found P2 and followed E2, and the orientation information while approaching to the place are matching in the opposite direction (i.e., orientation from P2 upward = 180 degrees + orientation from P6 to the place). Therefore there is a great possibility that the current position is P2, which NX previously visited.

NX performs the rehearsal procedure with the following reasoning. If the current position is really P2, then NX knows from the topological information that it can reach P1 with the information of E1, and P3 with the information of E3. It can use the procedural knowledge of each edge and place. It can use the metrical information to confirm the assumption. It actually follows E1 and E2 and visits P1 and P3. Notice here that NX does not make the same trace stretching to the middle direction between E2 and E6 as before. Because it already knows from the past experience that if the current position is P2, Move-along-object-on-left-side is the proper local control strategy. By performing the rehearsal procedure, it concludes that the current position is P2. The information saved for more exploration from P2 to P6 is deleted from the exploration agenda.

NX explored all possible directions from P2 to construct the topological model and it has three elements in the exploration agenda, which denote upward from P1, left downward from P6, and right downward from P3. It goes to P6 by following E6 and continues its exploration and gets to P3 again. It also performs the rehearsal procedure to confirm P3. After the confirmation, it navigates to P1 since it has to explore upward from P1. Notice here that all places do not need to have exactly the same location in the environment. An example of accumulated metrical information is the travel-history of E2 as illustrated. Again, D1R+ denotes movement from P2 to P3, while D1R- denotes movement from P3 to P2. The first list is the most recent one. The first list denotes that when NX moves from P3 to P2, it moves straight for the first approximately 100 steps, then it moves forward and turns to the left, and then it moves straight.
Travel-History of E2:
((DIR- (100 100) (8 12) (5 5) (22 38) (51 54) (72 72))
(DIR+ (4 6) (140 138) (57 35) (82 78) (3 2))
(DIR- (70 72) (26 32) (12 18) (46 56) (80 80) (6 4) (21 21) (4 6))
(DIR- (65 65) (4 6) (35 35) (4 6) (5 5) (20 30) (3 7) (37 43) (72 72))
(DIR+ (4 6) (141 139) (55 35) (85 85) (3 2))
(DIR+ (4 8) (78 78) (8 4) (46 28) (5 5) (50 40) (90 88))

Then NX visits P7 through E8, P8 through E9, P9 through E10, and P10 through E11. In all three figures, NX shows the same exploration order. From P10, NX moves downward in the ten percent error and the no-error cases, whereas NX moves to P7 in the five percent error case. This depends merely on the orientation after the hill-climbing search for P10. Since we consider the random sensor error, the order of exploration is nondeterministic. But NX gets a unique topological model in spite of the random error. NX continues its exploration until there is nothing in the exploration and no more unexplored directions from the current place. Interestingly, NX has a hard time finding P13 in the five percent error case, but not in the ten percent error case.

Once NX finishes its exploration completely, it selects a place randomly and navigates to the place. While NX is navigating, it accumulate more metrical information for the metrical accuracy of edges and places. There is another purpose of the random navigation stage. While NX explores, we assume that the environment is static. While NX navigates after the exploration, we are going to change the environment so that it can recognize what change has been made and update the map. This is not yet implemented. We present more results of various environments below.
Figure 6. More exploration results
Figure 6. More exploration results (Cont.)
4 Summary and Future Work

We have demonstrated a successful, robust, qualitative robot exploration and mapping method. The results show that our methods can solve several of the problems of traditional approaches. The major achievement of our approach is the elimination of cumulative metrical error. Key development tasks in the near future and a problem with our methods are discussed briefly below.

- **Orientation capability after dropped unexpectedly**: Once NX has built a map, it should be able to orient itself when it is picked up suddenly and dropped down at a certain location. Having this capability is one of demonstrations showing the strength of our approach.

- **Incorporating Systematic error**: The systematic error in sensory devices can be only partially recognized and considered an error in our approach. NX can find places which are distinctive according to its own perception of the world, but not necessarily geometrically. The systematic error in movement actuators can not hurt the topological model at all. It results in only less accurate metrical information. But it can be recognized and corrected by further metrical information analysis.

- **Metrical information**: Currently, NX-SIM accumulates metrical information, but uses it only for rough matching purposes. Metrical information can be used not only for shape description but also for path planning and more accurate matching, which could result in less time consumption of exploration and navigation. A relational map [Brooks, 1985] may easily be built from this information in our method.

- **Rehearsal procedure**: The current implementation of the rehearsal procedure can yield a false positive match. Finding a reference place or edge which is globally unique is required to avoid an incorrect conclusion.

- **Compass removal**: Although the current simulation uses a compass for global orientation, this assumption will be relaxed and a local orientation will be used. Then, each local orientation frame will be incrementally linked to the others.

- **Handling a dynamic world**: In a real world, there are a lot of moving objects. When a person is moving, a should be able to avoid a collision and build a correct map. When objects have been moved to different locations after NX built a map, it should recognize the change and update the map.

- **Hierarchical representation of complex maps**: Once a map is built, it can be represented hierarchically so that NX reasons about planning at the proper level. A region containing several places at a lower level can be represented as one place at a higher level.
References

R.A.Brooks (1982)
Solving the Find-Edge problem by Good Representation of Free Space,
Proc. the National Conference on A.I., AAAI, pp 381-387

R.A.Brooks (1985)
Visual Map Making for a Mobile Robot,

R.A.Brooks (1986)
A Robust Layered Control system for a Mobile Robot.

An Obstacle Avoidance Algorithm for an Autonomous Land Vehicle,
Proc. SPIE, Mobile Robots. Cambridge MA.

R.Chatila and J.Laumond (1985)
Position Referencing and Consistent World Modeling for Mobile Robots,

J.L.Crowley (1985)
Navigation for an Intelligent Mobile Robot,

M.Drumheller (1985)
Mobile Robot Localization Using Sonar, A.I. Memo 826, MIT A.I. Lab

A.Elles (1986)
A Sonar-Based Mapping and Navigation System,

A.M.Flynn (1985)

G.Giralt (1983)
Mobile Robots, Robotics and Artificial Intelligence, Edited by M.Brady
L.A.Gerhardt and H.F.Dividson, NATO ASI series, pp 375-393

R.Hickling and S.P.Marin (1986)
The Use of Ultrasonics for Gauging and Proximity Sensing in Air,
Journal of the Acoustical Soc. of Am., 79 (4), pp 1151-1159

Learned Navigation Edges for a Robot in unexplored Terrain,
IEEE Proc. the 2nd Conference on AI Application, pp 148-155

M.B.Kadonoff, F.Benayard-Cheriff, and A. Franklin (1986)
Arbitration of Multiple Control Strategies for Mobile Robots,
Proc. SPIE, Mobile Robots. Cambridge MA.

24
E. Koch, C.Yeh, G.Hilel, A.Meystel and C.Isik (1985)
Simulation of Edge Planning for a system with Vision and Map Updating,

D.T.Kuan, J.C.Zamiska, and R.A.Brooks (1985)
Natural Decomposition of Free Space for Edge Planning,


B.Kuipers (1979) Commonsense knowledge of space: Learning from experience.

Environment and Behavior 14; pp 202-220.

B.Kuipers (1983) The cognitive map: Could it have been any other way?
Spatial orientation: Theory, Research, and Application.
H.L. Pick. Jr. and L.P. Acredolo (Eds.), New York: Plenum Press. (a)


B.Kuipers and Y.T.Byun (1987)
A Qualitative Approach to Robot Exploration and Map-Learning,

J.Laumond (1983)
Model Structuring and Concept Recognition: Two Aspects of Learning for a Mobile Robot
Proc. IJCAI-83, pp 839-841

Computational Geometry - A survey,

Qualitative Navigation, Proceedings DARPA Image Understanding Workshop
Los Altos: Morgan Kaufmann

The Autonomous land vehicle preliminary road-following demonstration,

T.Lozano-Perez (1981)
Automatic Planning of Manipulator Transfer Movements,
IEEE Trans. on Systems Man and Cybernetics, pp 781-798
D.Miller (1985)  
A Spatial Representation System for Mobile Robots,  

A.C.C.Meng (1987)  
Free Space Modeling and Geometric Motion Planning Under Location Uncertainty,  

H.Moravec and A.Elfes (1985)  
High Resolution Maps from Wide Angle Sonar,  
*IEEE Robotics and Automation*, pp 116-121


Low Level Learning for a Mobile Robot: Environment Model Acquisition,  
*Proc. The 2nd International Conference on AI Application*,  
Miami Beach, FL, pp 156-161

First Results in Robot Road-Following, *Proc. IJCAI-85*, pp 1089-1095

S.Walter (1987)  
The Sonar Ring: Obstacle Detection for a Mobile Robot,  

C.R.Weisbin (1987)  
Intelligent-Machine Research at CESAR, *AI Magazine* Vol.8 No.1