# CHAPTER II

## A REVIEW OF MOBILE ROBOT RESEARCH IN NAVIGATIONAL PATH PLANNING, MOTOR CONTROL AND VISION

Considerable work has been undertaken, principally in the last decade, to provide a robot with the ability to navigate autonomously. This chapter will first survey the different approaches researchers have used for equipping their vehicles with navigational planning ability, concentrating on the control and representational strategies. The final part of the chapter will review various attempts to use machine vision as a sensor to effectively execute a plan developed within a navigational planning system. Spatial reasoning systems as in [82], although important for the semantic treatment of path planning, will not be dealt with in this review.

Path planning is a well-studied problem having an extensive history. It has been analyzed in the context of operations research, engineering, computer science, as well as other fields. The mathematics has been studied intensively and many algorithms have been developed, typically using graph representations, to determine a path in a completely known environment.

Navigation for a mobile robot, however, is a non-traditional path planning problem. The underlying reason is that a mobile robot operates at best in only a partially known or understood environment. The causes are manifold: uncertainty and conflicting information arising from inaccurate sensor data, dynamically changing environments, and incomplete models are some examples.

The goals of a navigational planning system for a mobile robot are:

- to maintain a model of the world surrounding it.
- to update that model using sensor data.
- to determine a path to a specified goal based on some criteria for "best" path.

**Monolithic control**

sensor dependent

**Hierarchical control**

single level representation

sensor dependent

sensor independent

multi-level representation

sensor dependent

sensor independent

**Distributed control**

simple blackboard representation

multi-level blackboard representation

**Spatial reasoning Systems** (independent of mobile robots)

**Figure 1: Mobile Robot System Taxonomy**

- to modify that path as necessary, during execution, based on new or conflicting sensor data.

A plan cannot be generated by a planner and blindly executed by the robot without additional sensor feedback. This would be courting disaster. Consequently, many approaches have been taken to try and cope with the uncertainty and non-monotonicity inherent in the problem.

A survey of the work done in the field of path planning for mobile robots is presented in the section following. A taxonomy of systems based primarily on control is given in Figure 1. This taxonomy is somewhat artificial but hopefully can shed light on the differences between the systems developed. It is not intended to be interpreted as an absolute classification of any system, as indeed there may be considerable overlap of categories. Some of the systems discussed later in this chapter may possibly be considered misclassified by some readers, but that should not affect the overall discussion.

A monolithic control structure, although occasionally implemented, is generally inappropriate for mobile robot path planning and navigation. Instead, a hierarchical planning system incorporating model and data representations that provide for easy exchange of

information as well as adaptation to changing conditions is unquestionably more suitable. Many variants of this strategy have been implemented. In a hierarchical control system, it may be desirable to use multiple representations instead of one single global all encompassing representation. Finally, a concurrent, distributed, cooperative system may be considered the ultimate in control schemes, typically utilizing a blackboard architecture for the passage of information. This strategy has several distinct advantages, not least of which is its ease of adaptation to parallel processing, so crucial to meeting the real-time constraints imposed by the mobile robotics domain. For distributed systems, multi-level blackboards may be preferred.

Some of the mobile robot systems developed have allowed their representations to be affected by their choice of sensors rather than taking a truly sensor-independent approach. A sensor-independent approach to representation and control is theoretically preferable, but sensor specific models are quite common due to the relative ease of implementation and other expediency issues.

Artificial intelligence techniques are far more evident in mobile robot path planning than are the traditional operations research algorithms. Heuristic search methods are commonplace. Considerable work has been dedicated to the treatment of uncertainty. Spatial reasoning and non-monotonic logic are also important although currently underdeveloped for this application.

## §1. Survey of control strategies

The tasks facing a mobile robot are many: path planning, obstacle avoidance, object detection, landmark recognition, position evaluation and world model building, to name a few of the goals and required capabilities. The way in which control is applied and the exchange of information between the different task modules constitute principal differences in the mobile robot systems developed to date. Three basic approaches can be seen:

- monolithic
- hierarchical
- distributed

Each will be described using existing systems where applicable as examples.

### §1.1 *Monolithic Control Systems*

Monolithic control is a fairly simplistic approach to a complex problem. No feedback is provided after the initial path generation decision and all planning and navigation is conducted at essentially a single representational level. Virtually all navigational knowledge is procedural. Consequently, monolithic systems would be used for quick-and-dirty type systems, typically in a commercial environment, that are constrained by limited computing power. In addition, several research systems that have been built to demonstrate sensory capabilities are monolithic, but no claims are made for their general use. In these cases, the representations used are strongly sensor dependent and thus hamper the possibility of generalization. Although it is theoretically possible to build a system that uses both monolithic control and sensor independent representations, none has been encountered to date.

One of these systems [130] uses a teaching pendant style approach to navigation, so common in commercial robotic arms. The robot is rolled to a particular location on the desired path where sensory data is recorded (for recognition purposes) and then it is moved to the next point. This process is repeated until the path is complete. The global path is never modified, and changing conditions are handled by using local obstacle avoidance strategies. In other words, once the global plan is made it is not modified.

The principal advantage of this control regime lies in its ability to respond rapidly, certainly crucial for meeting the real-time needs of a mobile robot. Although monolithic control is a short-term limited solution, this approach would best be avoided even by commercial developers so that prospective users (and buyers) of mobile robotic systems will not be discouraged by its shortcomings to the point of abandoning the technology.

### §1.2 *Hierarchical Control Systems*

The preponderance of the control systems for mobile robots documented in the literature utilize hierarchical control. That is not to say they are similar, quite the contrary. What they have in common is a structured and clearly identifiable subdivision of functionality. This functionality is relegated to distinct program modules which communicate with each other in a predictable and predetermined manner. Numerous examples illustrate this technique [90,100,108]. Two are described below.

A system developed by Hughes Artificial Intelligence [63] exploits hierarchical planning control. At the highest level is the MISSION PLANNER which establishes objectives, choice of optimality, and is responsible for invoking spatial and commonsense reasoning. The LONG RANGE PLANNER is at the intermediate level and charged with developing a global path subject to the constraints of the mission planner. The LOCAL PATH PLANNER, the lowest level, is used for obstacle avoidance and navigation around obstacles not modeled at the higher levels. A clear segmentation of control is evident. Replanning can be invoked due to the unattainability of goals at lower levels (e.g. unanticipated barriers).

One highly developed architecture from the University of Florida, implemented only in a simulation, is described in [66]. The modules for navigation are the planner, the navigator, the pilot, and low-level controller. The first three correspond in function with the three levels described in the Hughes system above. The low-level controller is responsible for issuing vehicle commands and status monitoring. In addition, a cartographer module is present, whose responsibility is to maintain the world model. This map-builder is accessible from all levels of the planning hierarchy. This planning decomposition, typical of many mobile robot systems, is reflected in AuRA as well.

### §1.3 *Distributed Control Systems*

Distributed control operates in an asynchronous manner. Global data structures such as blackboards are the keys to coordination and cooperation between the functional modules. These systems lend themselves extremely well to multiprocessing.

HILARE [51,52] is one of the more advanced and highly respected mobile robot systems, developed at LAAS in France. Its control structure falls in line with distributed systems. Specialized decision modules (SDMs) utilize a global database ("bulletin-board")

for communication purposes. These SDMs are used for planning, navigation, scene analysis and the like. The blackboard is bi-level: an announcement database and an information database. Communication between the SDMs occurs through this structure.

Ambitious work at Carnegie-Mellon University [120,127] uses knowledge sources to provide and interpret information on a multi-level local map. The local map-builder functions as a scheduler for the knowledge sources. Cognition modules are present, requesting information from the local map-builder for purposes of path planning, landmark recognition, global map updating and vehicle status monitoring. A major advantage of this system lies in its ability to easily adopt new knowledge sources, cognition modules and sensors into its structure. Clearly defined interface specifications will aid in the development of code at sites other than CMU and speed up the overall development of this system. A brief discussion of the blackboard representation used for the local map appears in Section 2.2.

Brooks [24], in a trend-setting paper, proposes a horizontally decomposed planning system. This allows for the concurrent operation of multiple behaviors in a layered manner. Although a pseudo-hierarchy exists, with certain of the behaviors having a higher priority than others, the concurrent operation of the system classifies it as a distributed control system.

Payton [107] and Kadonoff et al [60] also propose multiple asynchronous control systems that can produce multiple behaviors. Both of these systems as well as Brooks' are compared in Chapter 5 to the distributed control used in AuRA's motor schema manager. It should be noted that AuRA encompasses both hierarchical and distributed control: hierarchical for the development of the plan through its mission planner, navigator and pilot; then distributed for the actual path execution under the control of the motor schema manager.

### §1.4 *Assessment of Control Systems*

For any but the simplest mobile robotic systems, a distributed or hierarchical system should be chosen. Only if a particular theory regarding sensor usage in navigation is being tested should a monolithic control structure be employed. Monolithic control is too inflexible to handle the serious issues in navigation. In the non-monolithic approaches, the scope of planning can vary. Real-time responses can be facilitated using small-scale

planning on a local, sensor-driven level, whereas global plans can be developed (based on *a priori* knowledge) beyond the range of environmental sensing. Indeed the problem of navigation itself can be viewed as being solved by humans in either a hierarchical or distributed fashion.

The main question is: hierarchical, distributed, or some combination thereof? The advantages of each lies in the intended use of the vehicle test bed. Hierarchical control systems, in our opinion, are easier to develop, debug, and implement. For a production or commercial system, where all code is being developed on-site, the hierarchical approach enables a system to be built more rapidly, while still maintaining high functionality. A distributed system is in some ways more difficult to develop. Following the execution trace during asynchronous operation can be quite problematic. Although techniques are being developed for debugging distributed systems [16,35,75,122], much work remains to be done.

One major advantage the distributed approach affords is in incremental development of software. If it is properly designed, individual components (knowledge sources, SDMs, expert modules, motor schemas, etc.) can be plugged into the system with a minimum of disturbance. This leads to greater longevity for properly developed distributed systems. Incorporating new sensors is more readily accomplished. Additionally, by clearly defining the interface specifications, code can be developed by different groups, speeding up the overall software development process. The sharing of information is an additional major asset. Real-time processing demands are more likely to be satisfied, in part due to the greater ease in which a distributed system can be transported to parallel processors (in theory). The principal difficulty, however, is the development of a solid framework in which the individual components can execute and communicate effectively.

Essentially, if a system is to be developed rapidly with reasonably full capability and is not expected to undergo significant growth, but rather will be replaced by a new release, a hierarchical control system is the appropriate choice. If on the other hand, full functionality, high potential for growth, better high level reasoning support, and the ability to test out new ideas without a major redesign effort are more important than initial developmental difficulties, a distributed system is to be preferred.

## §2. Survey of Representations

This section contains a description of the characteristics of a good representation for the mobile robotics domain, a survey of some of the techniques used to meet those objectives, and an assessment of their merits. This survey is not intended to be exhaustive but rather representative of some of the approaches used in world modeling for mobile robots.

### §2.1 *Characteristics of a Good Representation*

To be able to evaluate different representational techniques, it becomes necessary to specify what properties are either essential or desirable for such a representation. Some of the characteristics described below are pertinent to artificial intelligence representations in general, while some are peculiar to the task at hand.

The principal characteristics of a good representation for the mobile robot domain include:

- efficiency
- representation of uncertainty
- multiple frames of reference
- sensor independence
- robot vehicle independence
- support for semantic information
- facilitation of parallel processing
- support of localization

**Efficiency**

In a real-time domain such as navigation in robotics, it is necessary to ensure that the algorithms used to process the data contained in a representation are efficient. If certain information is not explicitly contained in the representation, but must rather be computed at run time, real-time deadlines may be exceeded. The time-space tradeoff should

generally be resolved in favor of time with subsequent cost of memory. Nonetheless, integrity must be maintained to be certain that the stored data is not self-contradicting. For example, if an environmental feature's position needs to be updated, any other stored data that was determined based on the prior location of the feature must be updated as well. Obviously, if this will result in a large amount of change to the underlying data structures, it would be best to compute those quantities dynamically only when needed instead of storing them. The role and extent of uncertainty must also be considered in making the decision as to what should be explicitly represented.

**Representation of uncertainty**

Sensors do not provide exact information about the world surrounding them. Not only is that information inexact, it can be contradictory or erroneous. Consequently, any representation chosen to express the robot's world must take into account that things are not always as they appear to be, and that some data are more reliable or precise than others.

**Multiple frames of reference**

The robot's world is many faceted; it consists of objects relative to the robot, objects relative to each other, and objects relative to some absolute frame of reference. Uncertainty may exist regarding the absolute position of an obstacle within the world, but its position relative to the robot may be well established. Clearly, the object's relation to itself, i.e. its height, width, depth, and other features, does not depend on the absolute location of the object. By utilizing multiple frames of reference, these relationships can be expressed more readily.

**Sensor independence**

To be able to expand sensor capabilities, the most general case should be considered. The representation should encode sensor data in a form that is independent of the sensors themselves. Not all sensors provide the same type of data and each varies significantly in accuracy. Even when only one sensor device is used (e.g. video camera) there exist many different ways of processing the data provided, some more accurate, some more robust, some more reliable than others. In order to incorporate as much information as possible

into the representation, interfaces must be defined that are as sensor independent as possible. It should not matter that depth data came from ultrasound or a laser rangefinder or optic flow, or even from more than one visual depth algorithm executing concurrently. Only what the data are and an estimate of their reliability and accuracy is important. Granted, certain sensor data could best be combined directly in the presence of sensor models and algorithms for fusing sensor data. Nonetheless, the ultimate representations used specifically for path planning and navigation are more general if they are removed from sensor-dependent representations.

**Robot vehicle independence**

The use of a "virtual" vehicle, as described in [127], is the preferred modality. If a new vehicle is brought onboard, there would be no need to redesign the representation.

**Support for semantic information**

Symbolic reasoning will be essential if the robot is to do more than simply get from point A to point B. If spatial reasoning [as in 82] is to be implemented at the mission planner level, some means for storing symbolic knowledge about the world must be provided.

**Facilitation of parallel processing**

Realistically, to obtain real-time response to sensory data in an intelligent fashion to at best a partially known environment, parallelism must be exploited. This will enable the robot to move continuously through the world instead of in "lurch" mode.

**Support of localization**

It is not sufficient for the robot to only know a path to get from the start to the goal. It must also maintain information to determine where it currently is located relative to that path (i.e. localize itself). This necessitates the representation of more information than would be necessary if there was no drift in the robot's position from the path specified by the path planner.

## §2.2 *Representation Techniques*

Several different approaches have been used in representing the information for path planning. These include:

- Pure free-space methods
- Vertex graphs
- Hybrid free-space vertex graph methods
- Potential Fields
- Regular grid
- Quadtree
- Automaton
- Multi-level

Each has an associated history and advantages. They will be discussed in turn.

### *Pure free-space methods*

Two principal techniques fall under free-space methods: Voronoi diagrams and generalized cylinders. What is represented in both these approaches is the space between the obstacles, rather than the obstacles themselves.

Voronoi diagrams play an important role in computer science and mathematics, in particular when dealing with closest point problems [116]. A free space representation resulting from a Voronoi diagram represents space as a series of connected straight-line segments that typically split the distance between the closest pair of line segments of surrounding obstacles (including bounding walls). In appearance, the result can resemble a medial axis transform. More formally, a Voronoi diagram is produced by the generation of a set of polygonal regions, each representing an enclosed area in which all contained points are closer to one particular point in a given point set than to any other point in the set. This set of polygons partitions the plane into a convex net. The resulting diagram or its straight-line dual can be used to compute the safest path for navigation of a mobile vehicle.

Several problems are evident with this approach, including the fact that information regarding the obstacles themselves is discarded and is no longer available for high-level

reasoning processes. In addition, Voronoi diagrams cannot be readily used in changing environments or for moving objects. Every time the diagram is updated to navigate around unexpected obstacles, information is lost regarding the original clutter-free path. Maintaining the assumption that the paths traversed by the robot will seldom be identical, the utility of the Voronoi approach wanes.

One variation of the Voronoi diagram that is well adapted for robot localization using sonar is described by Miller [84]. He subdivides free space into regions that are characterized by their ability to provide localization information. Generally, the series of regions which yields the path through which the robot can maintain its position with the highest degree of certainty is chosen. A Voronoi diagram is produced of the regions within this path to yield a safe and certain route for robot travel using sonar navigation. The approach is limited in its applicability.

Another method using Voronoi diagrams coupled with a spatial vertex graph is proposed for the robot HERMIES [59]. Dynamic map making capabilities are provided using this representation. The Voronoi diagram is used to partition space into equivalence classes, guiding selection of the appropriate spatial vertex from which to initiate path planning. Two major assumptions are made, however, which may prove invalid in realistic situations: first, that the obstacle locations are unchanging and the world is static, and second, that sensor information is precise. Voronoi diagrams are poorly adapted in general to handling a dynamic world, whether or not the world is modeled *a priori*.

Brooks has developed a different free space approach that has extensions to three dimensions for classical pick and place operations for robot arms [26]. In a form not alien to computer vision specialists, he subdivides space (2D in the case of mobile robotics) into "generalized cones" or "freeways". (See [27] for the algorithm). These freeways, represented as spines with information regarding their clearance to the left and right, are produced by sweeping a 2 dimensional cross section through space according to a sweeping rule. An unusual characteristic of this representation is the allowed overlap of the freeways that are produced, contrary to the disjoint convex regions found with other methods. This representation, however, throws away too much information too soon, suffering from the same problems as the Voronoi approach. Recognizing these limitations, Brooks and co-workers have developed a hybrid extension [71] using both

passages (freeways) and convex regions of free space.

*Vertex graphs*

This technique owes its origins to path planning for robotic manipulators. Most forms have their basis in a two-dimensional version of Lozano-Perez's configuration space approach [76]. Here, the vertices of an obstacle are modeled by a polygon. The vertices are then grown by a distance equal to the radius of a circle enclosing the robot plus some margin of safety. Although the configuration space approach does not require making the assumption that the robot is circular, most working mobile robot systems do, thereby simplifying the computational problem dramatically.

A conspicuous feature of vertex graphs is their lack of explicit representation of the space between the obstacles. The advantage of this technique lies in the fact that the robot can now be treated as a point and occupies no space for path planning purposes. Extended versions of this algorithm have been developed for robots that are not circular, for path planning in 3 dimensions, and for conventional robot arms. Computational complexity for path planning can be of major concern for higher dimensions. See [85] for a discussion of the complexity issues regarding path planning algorithms.

Moravec, in his thesis [93], used an unusual representation that is not unlike vertex graphs. Obstacles were modeled as circles rather than polygons, and paths were constructed as a series of tangents to the circular obstacles in so-called "circle space". Needless to say, planning for an optimal path is considerably more complex with this representation than with a pure vertex graph approach. This lead Moravec to develop a sub-optimal algorithm that approximates the shortest path in circle space and produces the optimal path 95% of the time. This faster algorithm was used for a practical implementation in the robot Rover.

*Hybrid free space and vertex graph*

Representation schemes combining concepts from both vertex graph and free space representations are common in mobile robotics. Several examples are cited below.

A hybrid approach used by Chatila and Giralt in HILARE [34,51], Crowley with Neptune [36], and others (including our own work in AuRA), represents both the free space and the obstacles by vertex graphs. The space between obstacles is subdivided into

convex regions typically termed "meadows" or "places". It is a characteristic of a convex region that any point can be reached from another point within that same region without a collision (for all modeled obstacles). This reduces path planning to simply determining a sequence of piecewise linear traversals of meadows. This can be readily deduced from a connectivity graph.

Giralt [51,52] and Chatila [33] in HILARE have a highly developed free space representation that is developed dynamically from sensor data. It does not grow obstacles *a lá* Lozano-Perez and consequently path planning is complicated as the robot cannot be treated as a point. Space is modeled at two levels; a topological level which maintains information regarding places and their connectivity, and a geometric level which assigns dimensions to the components of the topological graph. Multiple frames of reference are available. This representation allows the robot to develop a map in totally unknown areas using only sensory data (laser rangefinder and vision). Its use is currently limited to indoor environments and fixed obstacles. Another problem is that uncertainty is not directly represented.

Crowley [36], in one of the more lucid papers on planning for navigation, uses an approach similar to Giralt's, but the convex region is first grown (shrunken actually) in a configuration space style. The algorithm used maximizes the area of the largest convex region. A "network of places" is produced which are connected not by vertices but by "adits" (definition [138]: a nearly horizontal passage from the surface of a mine). Despite the unusual terminology, these adits allow navigation through free space in a manner similar to that of the generalized cones approach described above.

Monaghan [90] uses a hybrid approach dubbed complex configuration space. Its only distinguishing characteristic is the use of arbitrary polygons instead of solely convex ones. As a consequence, it bears a stronger resemblance to the vertex-graph model than the other representations of this section, and thus inherits the advantages and disadvantages of that technique.

Brooks [25] argues strongly against using any two-dimensional representation for expressing the robot's world. Additionally, the claim is made that uncertainty cannot be modeled accurately by using any representation based on an absolute coordinate system. Instead a "rubbery, stretchy" relational map is proposed, incorporating both freeways and convex regions, as in [71], and explicitly storing the uncertainty in the represen-

tation. An abstract graph which deliberately avoids the use of a global 2D coordinate system is the principal representation medium. Local coordinate systems and their associated transforms (with their error functions) are used to relate the modeled regions. His claims are a bit extreme for the case where *a priori* knowledge is available for the modeled world. When a dynamic world map is developed from multiple and disparate sensor readings, his arguments seem more plausible.

*Potential fields*

This representational scheme uses a gravitational analog, converting obstacles into peaks and clear paths into valleys. The robot's initial position is placed at a higher elevation than the goal and the algorithm treats the robot as a marble rolling towards a hole. Vectors representing the attraction of the goal, the avoidance of obstacles weighted by urgency, and the existing acceleration are combined to yield the movement of the robot. Although extension is possible to mobile robotics [64], most of the work to date deals with manipulator trajectories. Potential fields in AuRA have found a place in the lower levels of a multi-level representation for a mobile robot; providing information for the motor schema manager to cope with obstacles, seek goals, follow paths and other behaviors during short range navigation, while using alternate representations for higher level planning.

Krogh and Thorpe [69] also apply potential fields techniques to the problem of mobile robot obstacle avoidance in conjunction with Thorpe's path relaxation techniques [126] based on a regular grid.

A principal advantage of this technique lies in its ability to represent uncertainty by changing the slope of the obstacle peaks. High confidence levels produce very steep cliffs, while uncertain obstacles result in a slowly rising broad slope. Optimal solutions were not part of the formulation of this control scheme, although the results are said to be "in some sense efficient" [68 p. 6]. Other problems exist in the susceptibility of the system to local potential energy minima, requiring specialized extensions (typically involving subgoals) to avoid "box canyons" [68].

*Regular Grid*

Grid representations have the richest history in mobile robot research. SRI's SHAKEY, Berkeley's JASON, and JPL's ROVER all used versions of the grid representation for navigational purposes [47]. This technique represents space in a classical 2-dimensional cartesian grid. Current systems using this technique have been developed by Thorpe [126], Mitchell [88] and others. A major impetus for the use of this representation lies in the fact that the Defense Mapping Agency (DMA) uses a similar format for their maps. Defense contractors working on the Autonomous Land Vehicle (ALV) Project will need to use the DMA data as a primary source of topographical knowledge.

A navigation representation using a regular grid approach has been developed by researchers at Hughes Artificial Intelligence [88] (see also *multi-level representations* below). This grid represents a 2D model of the world, each "pixel" representing a space 12.5 m by 12.5 m. The design decision is based on the availability of DMA data in a similar, though not identical, format. Connectivity is maintained through 8 arcs to each of the pixel's nearest neighbors. This poses significant problems for path planning, as paths, if unchanged, can only occur at angles of 45 or 90 degrees from node to node. This could cause any path developed by a navigator to be excessively long and thus non-optimal (using distance as a metric). Digitization bias is the term applied to this specific problem. Considerable effort has been made into minimizing the bias [87]. Additionally, compensation must be built into the search algorithm for the fact that a step in the diagonal direction is 1.4 times as long as one in the vertical or horizontal direction.

Thorpe [89,126] also uses a regular grid approach with 4 or 8 neighbor connectivity. An approach to overcoming the digitization bias, called path relaxation, is discussed in Section 2.4.

Moravec and Elfes [92] utilize a "world occupancy map" for representation of sonar data. Cells in the map (typical resolution of 6 inches by 6 inches) are used to represent the likelihood that an obstacle is present at a particular location. This model is one of the few to explicitly incorporate uncertainty. A numeric value in the range (-1,1) is associated with the cell. Negative numbers represent unoccupied regions, positive numbers occupied regions. The greater the absolute value of the number, the more certainty is associated with the conclusion; zero denotes no information.

One of the more unusual approaches using the regular grid method is presented by

Parodi [104]. Although the representation used is similar to the Hughes model, its use in global path planning is markedly different. The global planner accepts many different cost criteria from the mission planner: energy consumption, potential hazard, travel time, etc. and their bounds. Instead of computing a conventional point to point path from start to goal using a graph search algorithm, dynamic programming techniques coupled with relaxation are used to provide a cost function map. This in turn is used to develop a path description list which is passed to the pilot for execution. The major drawback of this method for path planning, as would be expected, is prohibitively high computation cost (up to 20 minutes of VAX-780 CPU time) necessitating the use of specialized processors for this algorithm's practical implementation.

*Quadtree*

SHAKEY was one of the first systems to use a quadtree representation of space for mobile robot navigation [97]. Researchers at the University of Maryland [3] have extended its use as a representation for planning purposes. Basically, space is recursively decomposed into $2^i$ by $2^i$ areas, not unlike what is found in the regular grid approach. If larger blocks have similar occupancy (binary valued - 0's for free space, 1's for obstacles), the decomposition stops before reaching the lowest levels of resolution.

The representation is used to generate a path which consists of a sequence of blocks through free space. Only vertical and horizontal neighbors, not diagonals, are used as a safety factor to minimize the likelihood of clipping of obstacles during path traversal. This only amplifies the already existing problem of digitization bias that this technique shares with regular grids.

The advantage afforded by this representation lies in the reduced computational cost for path construction as compared to the regular grid (although the initial map building is more expensive). On the other hand, free space representations are more expensive to construct than quadtrees, but are cheaper for path planning purposes. The question of how environmental uncertainty is handled in quadtrees is not dealt with by the Maryland authors but it would probably fall in between these two representations (regular grid and free space) in complexity. The principal drawbacks are perceived to be the possibly coarse and certainly varying resolution for path construction and the loss of uncertainty information due to a simple binary encoding of occupancy. Finer encoding of the occu-

pancy values for uncertainty would cause this representation strategy to degenerate into the regular grid approach. Also, additional information must be maintained about the nature of the obstacles in another level of representation if it is to be used for semantic processing or other reasoning.

*Automaton Representation*

In the most unorthodox representation scheme encountered, Tachi and Komoriya [121], in a well-funded and seemingly successful venture with "guide-dog" robots (more in Section 3.2), develop an automaton representation map for navigation purposes. Landmarks constitute the states of the automaton, each state containing information regarding the type of the landmark (currently only intersections). Inputs to the states are directional commands (left, right, straight) and outputs include steering angles for the robot and distance to the next landmark (among other things). The MELDOG system for visual navigation has received considerable attention, and therefore, although the automaton map is somewhat unusual, it warrants consideration.

*Multi-level Representations*

Multi-level representations are most appropriate for systems that afford multiple levels of control, i.e. hierarchical or distributed. The main idea is to represent information in forms that are best suited for different types of processing: high-level structures that can be tagged semantically for mission planning, absolute coordinate systems for low-level navigation and obstacle avoidance, etc. The greatest danger lies in the potential for inconsistent information to be stored in the different representations, so deliberate steps must be taken to maintain the overall integrity of the system. The choice of which representation to use at what level is still a major issue.

A multi-level blackboard system is being developed at CMU for DARPA's ALV project [120,127]. Sensor-dependent data resides at the lowest level. Here, representations will be chosen that are strongly influenced by the sensors employed. A three-dimensional coordinate system will be used. An intermediate blackboard level will consist of hypotheses or "partially instantiated models", not unlike a schema strategy as in the UMASS VISIONS system [55,53,42], which have been refined from the sensor-dependent data or established as expectations from higher level models. The blackboard's top level contains objects

that have been declared to be identified ("fully-labelled"), expressing components of the ALV's world in 3D coordinates. Information in the local map (short-term memory) is written to the global map (long-term memory) by knowledge sources when appropriate.

In another system developed for potential ALV use by Hughes Artificial Intelligence Center [63] involving the hierarchical control scheme described in Section 1.2, multiple levels of representation are used. At the mission planner level, symbolic manipulation of the data represented is important. An object-oriented data structure termed the "symbolic pixel array" is used. The details of this representation were not disclosed in the literature encountered at the time of this writing. The navigator, charged with developing long range plans, utilizes a two-dimensional integer lattice (described in the *regular grid* discussion above). This has obviously been influenced by the availability of DMA map information in a similar form. At the pilot level, and specifically geared for use in obstacle avoidance, a polygonal representation (as in HILARE) is proposed. It is worth noting that the scale used for path planning lies in miles rather than feet and that open terrain instead of indoor environments constitute the robot's domain. Using different levels of representation at different levels of control makes sense only if provision is made for the exchange of information between representations. This is necessary to preserve the integrity of the world model. Although planning control changes predictably from one level to the next in this system, it is unclear how any one of the representation levels could benefit from new information incorporated into another.

### §2.3 *Assessment of Representation Techniques*

Ad hoc representational strategies (particularly those developed for limited task domains), although useful for sensor-dependent approaches, do not support the future of mobile robotics. Most are too narrow in scope to contribute significantly to the field. Hopefully, a cogent approach to representation that is broadly applicable and easily extendible will appear.

One of the biggest problems faced by all developers is a means for supporting semantic processing. A consensus appears to exist that a rule-based reasoning capability is needed to achieve the full potential for decision-making in navigation (e.g. see [86]). None of the representations discussed above are geared specifically for this form of reasoning. Although representations have been developed for abstract spatial reasoning systems

used for path planning purposes, few of the implemented robotic systems encountered does more than offer lip service to symbolic reasoning. Of those that do [117,137], the problem is considered in isolation, and no real effort has been made to drive a robot equipped with multi-modal sensors.

Practical systems must deal with multi-level representations in order to support different types of processing requirements. Hierarchical or distributed control systems are a prerequisite for multi-level representations (see Section 1). The problem confronting designers lies not in which representation to choose for which level as much as in ensuring that the information maintained is consistent throughout the system. This favors global data structures. The use of a map-builder or cartographer, whose sole function is to maintain the consistency of the multi-level representations in a changing world, is crucial to a properly functioning system. Cartographers (map-builders) have been found in both hierarchical and distributed planning control systems [66,117,127]. Standard database techniques should be used, such as locking at different levels of granularity to allow for efficient operation during the frequent updating anticipated.

Certainly both freespace and the obstacles themselves should be represented somehow. Many researchers have recognized this need and as a result various hybrid systems exist. Whether the hybridization is done using generalized cones, vertex graphs, Voronoi diagrams, regular grids or whatever, poses a relatively fine point at this stage in the history of robotic navigation. It is expected that representation and control architecture development will be evolutionary, changing in response to new and different types of sensors, vehicles, and tasks. Any system designer would do well to recognize this phenomenon and make an effort to allow for adaptation and accommodation of new ideas and technology through inherent design flexibility.

### §2.4 *Path Generation Strategies*

Generating the path from most of the representations above is a classic artificial intelligence problem, requiring the search of a possibly large state space. The representation used determines the nature and extent of the space to be searched. Some representations will require greater search effort than others.

In the vast majority of cases, the A* algorithm is the search technique used. Although Dijkstra's graph search is occasionally mentioned (e.g. [88]), it is generally agreed that A*

is the algorithm of choice. The basic question instead is which heuristic should be used. Generally the most straightforward is employed: the remaining straight-line distance from the current position to the goal. This is sufficient to guarantee an admissible solution based on a distance metric for optimality. In several cases however, (including AuRA). criteria other than just straight-line distance are considered. Factors such as traversability of terrain, safety, etc. are weighed in developing an appropriate cost function. It has been stated that the heuristics are developed largely on an ad hoc basis [66], and more theoretical research is needed to understand just what constitutes an optimal path for mobile robots. The question of how important an admissible solution is to this particular search problem remains unanswered as well. Chattergy at the University of Hawaii has explored some of the heuristics that can be applied to mobile robot navigation [34].

One interesting approach to path generation developed by Thorpe at CMU is based on a path relaxation process [89,126]. It develops a better path from an original first cut solution initially put forth by operations performed on the underlying grid representation. Subgoal nodes, (produced from an A* search of the grid), are displaced according to specific constraints and the cost function recomputed. This process is repeated until convergence is observed. The net result is the removal of jagged edges as the path settles into a relaxed state. This idea of an iterative relaxation process for path refinement is extendible to other representations as well, and is used in AuRA (in a limited way) for multi-terrain path planning (Chapter 3).

In at least two cases, dynamic programming techniques were exploited for path generation instead of pure heuristic search [104,121]. In the first case cited, dynamic programming is used in conjunction with standard AI search techniques. Based on the claims made by the authors, dynamic programming seems a viable alternative to pure search, but it still appears as the exception rather than the rule in exploring paths for the mobile robot's world.

## §3. Summary of vision research in mobile robotics

Vision research in mobile robotics has tried to accomplish the very difficult. Working in a partially known and uncertain environment, with a camera that is in motion and subject to bouncing, the vision system must provide information to the robot that is both

useful and accurate. Nonetheless, in order to provide general robot abilities, vision must be exploited to the fullest. Several different tasks for the mobile robot are well-suited for vision. These include:

- obstacle avoidance
- path (road) following
- goal (target) identification
- landmark recognition for vehicle localization
- dynamic map making

Many different research groups are exploring these areas. One way of greatly simplifying the task is by restricting the domain. The discussion that follows first describes the approaches that are domain-specific and then looks at those research efforts that are less restrictive. As this chapter is concerned primarily with representation and control, the vision systems discussed below should be viewed as a sampler, rather than a comprehensive survey of current vision research in mobile robotics.

### §3.1 *Man-Made Environments*

Restricting the domain of a mobile robot to a known environment significantly reduces the number of possible interpretations of visual data. Several research vision systems have been constructed to exploit these constraints. A few are described below.

Obstacle avoidance using vision is one of the major areas of work. Tsuji [129,130] has developed a working system for a mobile robot operating in a hallway environment. It is based on two assumptions: first, there are many visible vertical lines in the scene; and second, the floor is almost flat. Although the second assumption is reasonable, the first may be brought into question for any area other than a hall. If that hall has smooth walls, it may be impossible to navigate even there. What is of note is the rapid sampling of data (1 image/second) while the robot is in motion (velocity 0.3m/sec). Optic flow analysis is used to provide information regarding collision avoidance and obstacle detection (the depth-from-motion algorithm described in Chapter 6 is also optic flow based). A 3 by 3 Sobel operator is used to detect vertical lines. Features are then extracted using Moravec's

interest operator in the neighborhoods surrounding these lines. Errors of only 10% in the distance to objects near the robot are claimed. This algorithm is also capable of detecting objects moving in directions other than the direction of motion of the robot. The inapplicability of this system to other environments makes it useless for the general case.

Another approach used for indoor obstacle avoidance is described in [89,125]. This system, developed at CMU, uses 2-eyed stereo, a marked improvement in computation time over the initial 9-eyed system previously used [93]. A thorough study of the constraints affecting processing speed is provided. These constraints involve both the imaging geometry (camera and robot) and motion geometry (predicted position). Nonetheless, the system is still too slow, taking in excess of 30 seconds per step. One second per step is claimed as the required speed for real-time obstacle avoidance.

### §3.2 *Following Roadways*

Several research efforts have been directed to the task of following roadways. Four are reviewed below.

Researchers at the University of Maryland [74,135,136,137] have looked at the recognition and following of roadways specifically with the DARPA autonomous land vehicle in mind. Their process for road following is broken down into two distinct phases: Bootstrap-Image Processing and Feed-forward Processing. The purpose of the bootstrap system is to find the road's location without prior information about the vehicle's position. Once the road is identified and the robot is in motion, a feed-forward strategy is employed that relies heavily on the inertial guidance system for dead-reckoning. This restricts the possible location of features to smaller windows that can be processed more rapidly. The constraints based on error studies have been set at 0.25 meter and 1 degree of orientation - fine for inertial guidance but impractical for most lower cost systems. In order to maintain these tolerances, an expensive pan and tilt mechanism would be required as well. The control system proposed is hierarchical, consisting of a pilot, mission planner and navigator. The navigator is the sole interface to the three visual processing modules. Line extraction, by combining evidence from multiple image windows to yield the long parallel lines of the roadsides, is the principal example cited in the paper.

A similar strategy is used for the same task, but at higher speeds, by Dickmanns

and Zapp [39]. This system also operates using a window concept to meet the real-time processing constraints. Greater consideration is given to vehicle dynamics and the groundwork is presented for the eventual high-speed (up to 65 km/h) guidance of ground-based vehicles by computer vision. A dedicated microprocessor is assigned to each feature tracked in its own window. Anticipatory control is utilized via a "preview" window (based on vehicle dynamics). Stationary obstacle recognition is considered as well.

The ALV group at CMU has implemented a road-following system on two working robots, Neptune and Terregator [133]. Although the approach is simpler in concept than either of the above two systems, a working robot plant using these concepts exists, not merely simulations. Initially, the only environmental sensor used was a single black and white TV camera [133]. This has since been extended to color [132].

More recent work at CMU on the NAVLAB [134] has demonstrated the ability to follow roads that are streaked with shadows and poorly registered in the color spectrum. A pattern classification scheme based on pixel values on a color surface distinguish sunny and shaded road from non-road regions. The image pixels associated with the road are grouped into a single region which is then used to servo the vehicle.

MELDOG [121] tracks road edges with a CCD camera whose field of view can be changed to detect the road edge. The velocity requirements are much less stringent than for the ALV, but real-time processing is still required and "lurch-mode" unacceptable. This road-following algorithm is supported by other vision algorithms that determine landmarks and handle obstacle avoidance based on ultrasonic sensor detection.

Road following is essentially as domain-specific as the man-made environments described above. A case could easily be made stating that roadways *are* man-made environments. Consequently, by understanding the nature of the roads to be traversed and restricting the robot's motion to those roads, it is realistic to expect that reasonable real-time speeds could be attained. Off-road navigation (or on poorly defined roads or trails) would then be an entirely different problem.

### §3.3 *Goal (Target) Identification*

Goal identification, primarily of military importance at this stage in development, is a means to determine targets in a changing environment. There is no absolute reason why this must be the only explored domain. A child running to hug its mother in a crowd

is an example of navigation being used to accomplish more pacific goals. It can also be argued that goal identification is the complement of obstacle avoidance; seeking objects rather than avoiding them. A goal/target can be defined as an object or collection of features of an object or environment which is to be recognized and approached. The goal has a stored representation for either the general class of the object in question or a specific instance of it.

Many of the issues in goal recognition are similar to those found in landmark identification; the context in which the results are to be used marks their distinction. Landmark identification is used principally for orientation of the robot, helping it to establish and maintain its position en route to a predetermined goal. Goal identification, however, must be carried out before final planning can be accomplished. The planner cannot determine an optimal path if it does not know where the goal is. Certainly exploratory path strategies could be employed to survey the area until the goal is encountered. However, if the robot has no means to determine that its goal is in view, it could not attain it.

Thus far, infrared imaging seems to be the dominant sensor modality for target identification. This regresses to the desire of the military to target hot objects (such as a tank or aircraft) in a relatively cold environment. Work done at Honeywell [20] involves optic flow and a partially rule-based system for obtaining range data from passive infrared sensors. This project utilizes the speed of an aircraft (or missile) and the sensed imagery to yield a dense range map (85-95% accuracy is claimed). Although currently the system is geared towards missile guidance, the designers claim it can be extended to ALV research for ranging and obstacle avoidance.

Goal recognition is important if only approximate knowledge of the final destination of the robot is available. If *a priori* knowledge indicates that the goal is in a certain room, the planner can march the robot into that room and then have goal identification algorithms determine the exact location of its target. This mimics the way in which humans might search for something when told it is in an approximate location. For example, if instructed to attend class in room GRC 301, an *a priori* map of a campus can get us to room GRC 301. Finding a vacant seat to sit in is another story. Although goal identification for autonomous machines is currently dominated and sullied by destructive applications, its necessity for truly autonomous robots cannot be underestimated. Little work has been done towards implementing a practical mobile robot system functioning

in an unstructured environment using this concept.

### §3.4 *Landmark Recognition*

This task actually subsumes the role of goal/target identification, the fundamental difference being that motor behavior is not necessarily a consequence of landmark recognition. Landmark recognition for vehicle localization is a more ambitious and more difficult task than just identifying a road or lines in walls. What constitutes a landmark and how to identify them can involve techniques as simple as spectral analysis or template matching to as complex as the full problem of natural scene analysis. Currently, real-time constraints will tend to favor the simpler approaches.

Landmark recognition in a highly simplified situation has been implemented in MELDOG [121]. White painted lines of a specific length and width at known locations have been painted on roads to assist in localization of the vehicle. A specific landmark sensor system, using two sensors, (one each at the front and rear of the vehicle facing down) provide landmark detection. This is extremely limited. The designers discuss the possibility of utilizing real-time landmark recognition with ultrasonic data (principally walls), but do not extend vision as a practical means of handling the localization problem any further than using the simple "white-stripe" approach described above.

An ambitious system encountered for landmark recognition has been developed at the University of Maryland [3]. It relies on a Hough transform based template matching algorithm to identify landmarks whose representations have been stored in a landmark database. After matching has been accomplished, a maximally consistent set of landmarks is constructed which is then used to provide information, by techniques based essentially on triangulation, on the vehicle's position. It can also be used to request the visual system to find additional landmarks that would improve the position's certainty by repositioning the camera. The full navigational system is hierarchical, involving long range, intermediate range, and short range planning.

The system consists of three major modules: the Matcher which establishes expectations for likely landmark positions; the Finder which directs camera angle and focal length to attempt to find the landmarks in the positions put forward by the Matcher; and the Selector which chooses a set of landmarks that would be most appropriate for localization purposes. Although it remains to be incorporated into a working robot, this

project is the first system that takes visual landmark recognition in the more general case seriously (see Chapter 7 for AuRA's approach and a comparison to the Maryland system).

### §3.5 *Dynamic Map Making*

Having a robot explore a region where it has never been before requires special capabilities. The vehicle must be able to acquire a model of its surroundings in order to determine the proper trajectories for it to navigate through. Vision has not been fully exploited as a sensor modality for localization to date. Most practical work has been concerned instead with sonar or laser rangefinder data [36,43,131]. Of the systems that rely heavily on visual data for dynamic map making, the results are quite sketchy.

Brooks dogmatizes important issues in visual map making in [25]. His approach is restricted to an indoor environment with the intent of developing a "rubbery, stretchy" relational map. The algorithms for vision expressed in the paper have not yet been implemented nor tested on real data. In several instances, only the general issues are described without tackling a specific computational approach for vision. Brooks' control paradigm espoused in [24] provide the framework for the implementation of these concepts.

Moravec, as previously mentioned [93,125], used stereo to obtain a sparse depth map to represent the location of obstacles in order to plan a path around them. In the later work, CPU times of 30 seconds to one minute per step is required. Sensed obstacles are represented as circles and path planning is conducted within the confines of "tangent space" - the paths that connect the tangents of the enclosing circles. Although obstacle avoidance is feasible with this approach, much work remains to be done in order to develop any semblance of higher level knowledge acquisition through vision. Moravec in [91] philosophizes on the roles of vision and locomotion in mobile systems (biological or otherwise).

HILARE [52] complements vision with a laser rangefinder as a means for successfully developing a world model of polygonal obstacles and walls. A low-level vision system detects planar surfaces of objects by contour extraction and filtering, serving as a guide for the laser range finder. The obstacle's slope is obtained which then is used to produce a 2D ground projection fit for incorporation into the world model.

## §4. Summary and Conclusions for AuRA

This chapter has presented several yardsticks against which to compare representational strategies. These include:

- Representation of uncertainty
- Efficiency
- Support for multiple frames of reference
- Sensor independence
- Robot vehicle independence
- Support for semantic information
- Facilitation of parallel processing
- Support of localization

Chapter 3 will introduce the AuRA architecture. AuRA incorporates many of the best features of existing systems. The use of a regular grid in short-term memory to reflect uncertainty, while providing a hybrid vertex-graph free-space (meadow map) model to facilitate intermediate level navigation, plus embedded landmark and semantic information for mission-level decisions, affords across the board support for navigational and spatial reasoning issues. A hierarchical planner is used to develop a global path, while actual path execution, monitored by a distributed control system, uncouples reactive/reflexive piloting from map-navigation. A cartographer separately maintains the global data structures used to support the map-navigation. Robot vehicle independence is guaranteed by providing a standardized vehicle interface for communication with the robot. Sensor independence is maintained through a sensor-independent short-term memory level wherein all relevant sensor data is fused. An egocentric frame of reference is maintained within the perception and motor subsystems, while a global model based on *a priori* knowledge exists within the cartographer.

Much credit must be given to those researchers whose previous and ongoing efforts are reviewed in this chapter. Without their advances into new territory, (some advancing more blindly than others), many of the mistakes and difficulties that were made were

doomed to be repeated by others. The diversity of their representation and control efforts reminds this author of the early days of aviation. Perhaps a clearly superior approach to the problem of intelligent sensor-driven mobile robot navigation will eventually become apparent. Until that time, best wishes to those daring young men and women and their walking, crawling, creeping, rolling, and hopping machines.