Knowledge/Geometry-based Mobile Autonomous Robot Simulator KMARS

Dr. Linfu Cheng, John D. McKendrick and Jefferey Liu
Elcee Computek, Inc., Boca Raton, FL 33431

ABSTRACT

Ongoing applied research is focused on developing guidance systems for robot vehicles. Problems facing the basic research needed to support this development (e.g., scene understanding, real-time vision processing, etc.) are major impediments to progress. Due to the complexity and the unpredictable nature of a vehicle's area of operation, more advanced vehicle control systems must be able to learn about obstacles within the range of its sensor(s). A better understanding of the basic exploration process is needed to provide critical support to developers of both sensor systems and intelligent control systems which can be used in a wide spectrum of autonomous vehicles.

Elcee Computek, Inc. has been working under contract to the Flight Dynamics Laboratory, Wright Research and Development Center, Wright-Patterson AFB, Ohio, to develop a Knowledge/Geometry-based Mobile Autonomous Robot Simulator (KMARS). KMARS has two parts: a geometry base and a knowledge base. The knowledge base part of the system employs the expert-system shell CLIPS ('C' Language Integrated Production System) and necessary rules that control both the vehicle's use of an obstacle detecting sensor and the overall exploration process. This initial phase project has focused on the simulation of a point robot vehicle operating in a 2D environment. Obstacles were depicted as complex (non-convex) polygons and the vehicle movement was constrained to the x-y plane. Rules controlling the vehicle's motion in free-space activated, when necessary, a sensor that derived obstacle information put into CLIPS working memory. The vehicle must use its sensor to learn about obstacles blocking its path toward the goal and what obstacle vertices can be seen from a given vehicle location. Factory supplied sensor technical performance specifications (e.g., range and bearing) can be selected under the "Sensor" menu option. The user can also select a number of "Display" options that show various aspects of the vehicle's environment (e.g., vehicle track, vehicle location, portions of obstacles discovered, etc.). With the use of an "Obstacles" option, a user can create new obstacles, delete and/or move old ones to new positions. Control of the CLIPS knowledge base activities is accomplished through various "Explore" menu options. A plan view of the environment on the screen, allows the user to monitor the progress of exploration and information being accumulated in working memory.

It is anticipated that this research will progress to develop operational capabilities for 3D environments.

I. INTRODUCTION

A. Need For Autonomous Systems

Today, autonomous systems are required for tasks in hazardous environments (ie., toxic, radioactive, etc.) that are extremely injurious to human health. Additionally, capabilities for autonomous operations are needed in those environments that are characteristically unstructured and, as such, are unpredictable. These environments may be the result of a catastrophe or the characteristics of the environment may have been unpredictably altered since last being visited. An autonomous system must be able to use its sensor(s) to detect the presence of and the locations of objects in an unknown environment. The system must also be able to incorporate updated spatial information into its task-reasoning capability.

B. Background Research Efforts

An autonomous vehicle's efficient utilization of available information for the purpose of exploration and navigation is a key problem in robotic research. The simplest expression of the problem of motion amongst obstacles is that of a point automaton which can move in the 2D plane, avoiding obstacles [1,3]. Research into robot motion planning has been approached from two different vantage points, each based on different assumptions about information or knowledge that the automaton has about its surrounding environment. In the first approach [4,10] the automaton is assumed to possess complete, a priori, knowledge of all aspects of each obstacle. Under this assumption, the vehicle's movement problem is that of "path planning with complete information," and the planning of an optimized path can be a one-time computation. Because all spatial information about the environment is known at the onset of vehicle operations, there is no need to use a sensor to acquire new information about the location of obstacles.

In the other approach, the automaton is assumed to have no knowledge or only limited knowledge of its surroundings [2,6,7,9]. The vehicle must rely on some sort of sensing capability to gather information about the environment. There is no opportunity for optimized transits to all parts of the environment until all aspects of the environment have been fully learned. However, once complete spatial knowledge has been accumulated for a certain region of the environment (ie., a complete regional map is available), regionally optimized transits to goals within this region can be undertaken. In this situation, regional path planning can be a purely computational process and no further sensor operations are required in that region. There may still be other unknown regions of the environment in which sensor operations will be required when transits into or through those regions are required.

Prior research has concentrated on robots operating in known environments and on algorithms for finding globally optimized paths. Research into algorithms for exploring and navigating in unknown environments is less able to address the problem of path optimization to a goal.

There is a need for the capability to simulate exploration and navigation activities so that the efficiencies of various autonomous systems techniques both for vehicle movement control and for sensing operations can be more fully assessed.

II. Exploration and Navigation

Navigation conveys the sense of directing the course of a mobile system based on an a priori knowledge of where impassable areas are located which have to be avioded.

Exploration concerns the initial acquisition of knowledge of where an object is located. Usually the discovery as to the existence of an object is made through a "sighting" of the object and a recording of its location is made. The format of the record of object locations can be either textural or spatial (i.e., map) such that the information can be readily used for subsequent navigation.

The acquisition of spatial knowledge involves three activities: (1) the use of a sensor (e.g., vision, sound, touch, etc.), (2) recording of spatial detail for possible future use, and (3) movement to a new vantage point for the reapplication of (1).

A. Expert System for Unknown Exploration and Navigation

It has been found that the use of expert systems combined with modular procedures provides a convenient and powerful method for controlling a robot vehicle's behavior [5, 11, 12]. It is possible to use an expert-system shell to make high-level decisions concerning exploration and navigation via the shell's internally implemented inferencing procedure. Within the shell, learning can be emulated through the updating of information into working memory.

Knowing nothing about what lies between it and a desired goal position, the first need of an autonomous system equipped with a vision/ranging system, is for its sensor to be activated to "see" if the goal can be detected. If the goal is visible, the implication is that there are no obstacles in the path of the vehicle [infinite width vehicle] and it can move directly to the goal. By treating the vehicle as a point, there are no passages too small for the vehicle to pass through.

A state-space representation of the exploration and navigation process is shown in figure 1. In an unknown environment a vehicle would be operating in the states in the upper-right portion of the graph. As more information is acquired, the vehicle might be operating in the states in the lower-left portion of the graph.

B. Sensor Operations

Long range sensor operations (vision/ranging) are not essential for a system to find a goal in an unknown environment. It has been shown that a goal can be found with a sense of touch and continuous knowledge of the direction to the goal [5]. Although some research has addressed the exploration and navigation process utilizing unlimited range sensors, little research has focused on how sensor range limitations affect the process.

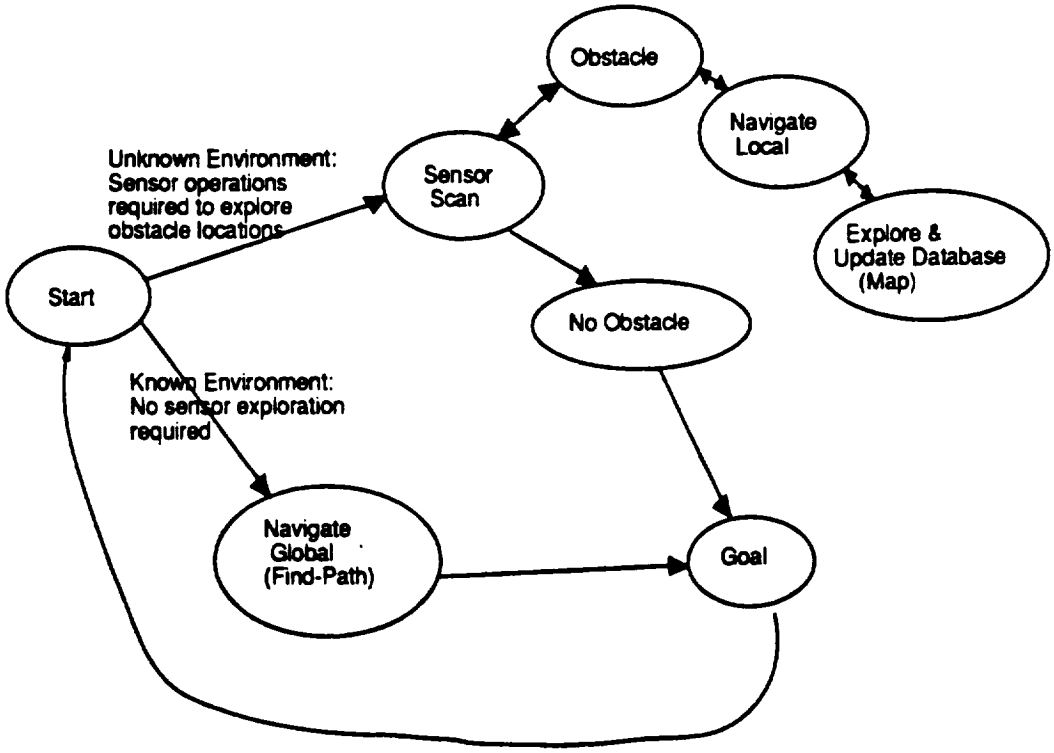

Figure 1. - State-Space Representation of Exploration Problem vs. Find-Path Problem

III. KMARS System

A KMARS user can specify the shape of and the placement of polygonal obstacles in a 2-dimensional environment, select characteristics for a sensor used by the robot vehicle, and compose rules that control the vehicle's activities in exploring the unknown environment. The firing of a rule might activate 'C' functions that perform necessary vehicle tasks. Figure 2 shows the relation between CLIPS and the activation of 'C' functions. The function may return updated information to CLIPS working memory.

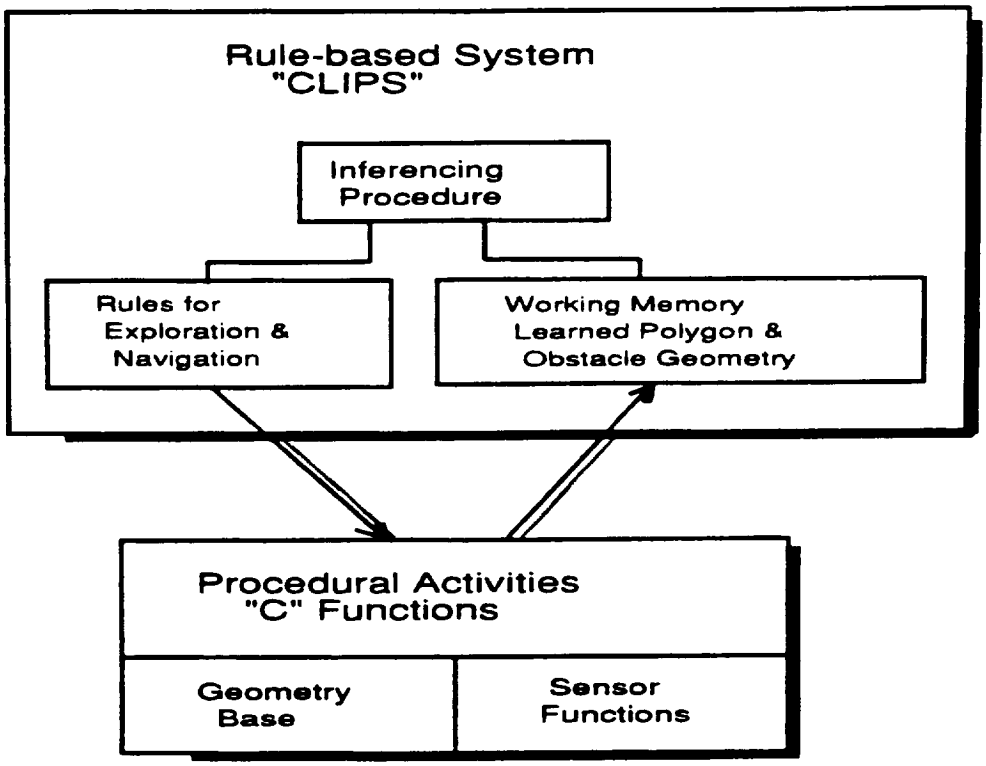

Figure 2 - The KMARS Control System Architecture

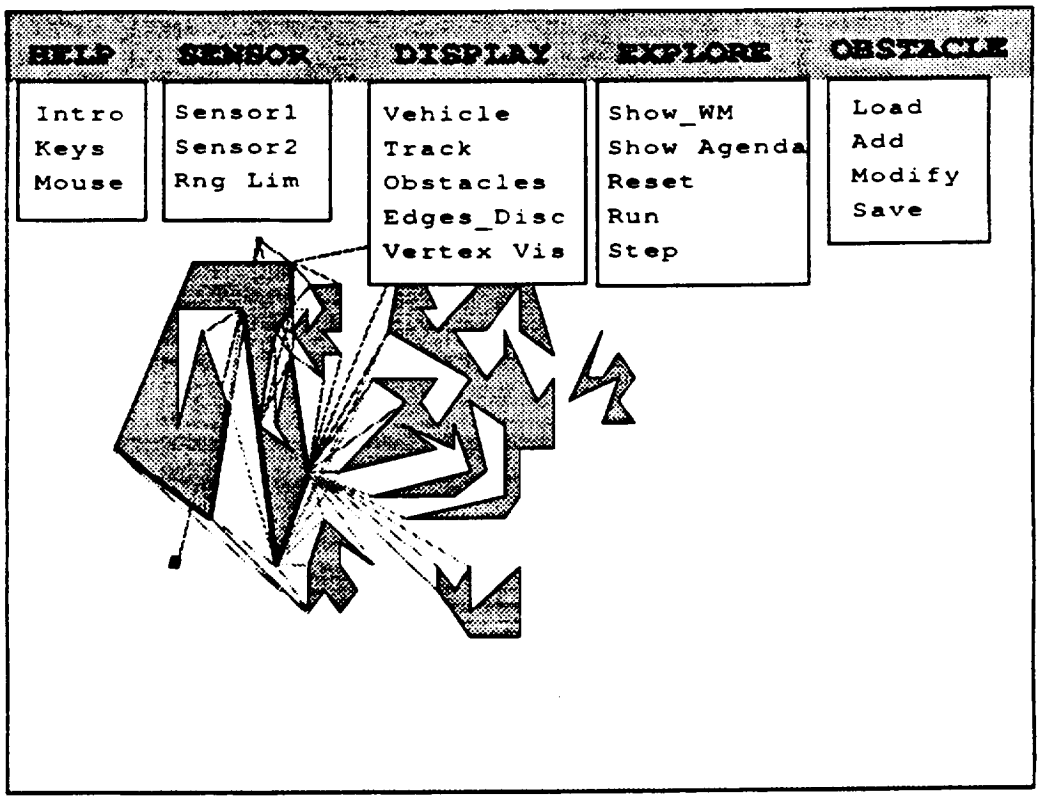

Figure 3 - KMARS Menu Bar and User Selection Options

The user's computer screen shows the menu selections and the vehicle's operating area. By using Menus, the user can select various KMARS operating options. The Menu Bar and individual Menu Selections are shown in Figure 3.

A. Geometric Model

The operating environment of KMARS can be constrained by the presence of 2-dimensional polygons. They are, generally, non-convex. The geometry-base verifies that a user-specified obstacle is a valid polygon and checks to insure that a newly defined polygon does not overlap one that has previously been defined. The geometry model also generates an edge-vertex matrix which relates visibilities amongst all polygon vertices. The matrix stores the polygon vertex visibility information.

A goal is obscured if the line-of-sight to the goal intersects a polygon edge. The question of visibility involves a computation that is handled by the [analytical] geometric model. The simulation and manipulation of 2D obstacles in KMARS is maintained by the geometry base that models the 2-dimensional polygon objects and the model provides information about the properties of the polygonal objects.

Polygon obstacles can be created, moved, deleted from the operating environment by the KMARS user. Figure 4 shows a polygon obstacle being created by point and click of the mouse at the position a polygon vertex is desired.

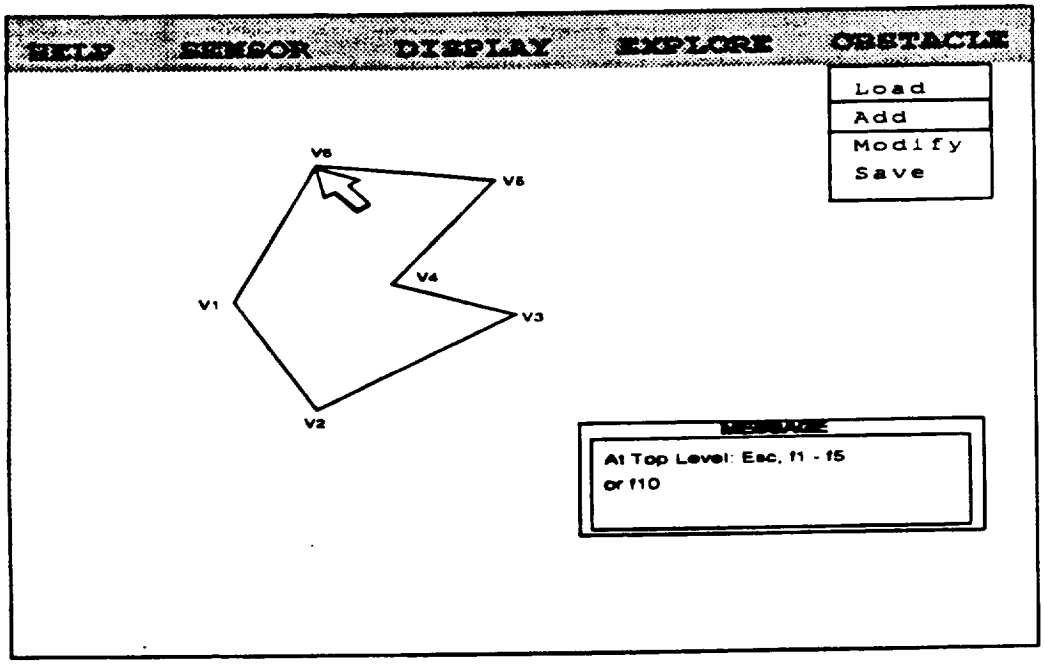

Figure 4 - Creation of Polygon Obstacle

B. Sensor information into working-memory

Figure 5 shows an environment with several 2D obstacles and a

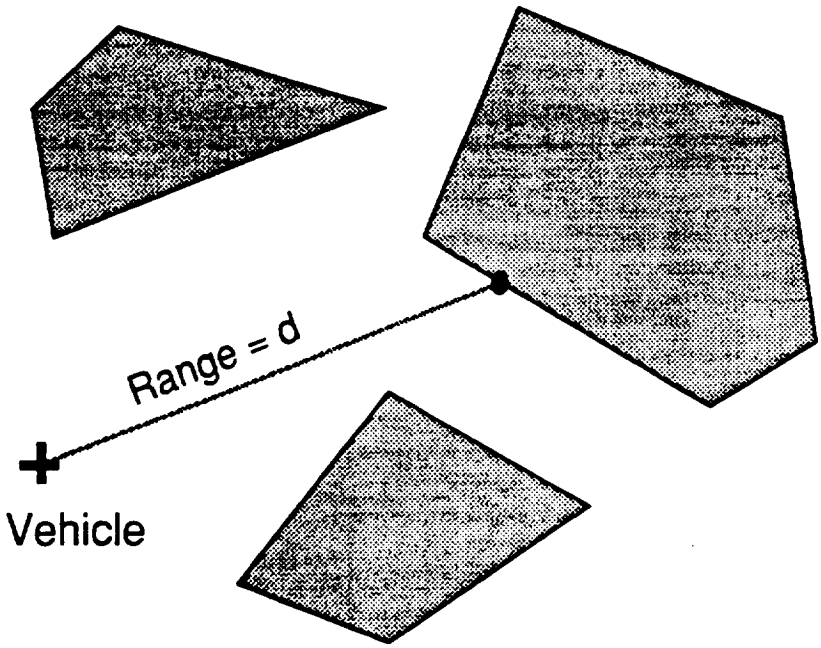

Figure 5 - Laser-Type Narrow-Beam Sensor Detection of an Obstacle

narrow-beam (e.g., laser) sensor pointing toward the goal. An object is detected at some range (d), less than the sensor's limiting range. The detection of an obstacle implies that the goal can not be seen from the vehicle's present position.

A vehicle operating in the KMARS environment can encounter three typical situations which a sensor attempting to check the visibility of a goal might experience:

- the goal is visible
- the goal is blocked by some distant edge
- the goal is blocked by the adjacent polygon

Once a rule has activated the sensor, a 'C' function is called which computes the visibility condition based on information (e.g., current vehicle position, goal position, etc.) passed to the function. The function also calculates spatial information about goal visibility and inputs it into working memory.

C. KMARS Rule-Base

The exploration and navigation activities of the vehicle in KMARS is controlled via a rule-based system. This rule-based system, using data received from its sensors, maps out the "visible" portion of the environment as the vehicle traverses toward a defined goal. The mapping is handled by additions and deletions of spatial facts to working memory.

KMARS has a basic exploration and goal finding strategy and some added rule refinements. The basic strategy is to move to the position beside the polygon that blocks the view toward the goal. From there, if the polygons left-most vertex has not been explored, the vehicle moves to that vertex. The vehicle then calls for a sensor activation and all of the other polygon vertices visible from that vertex are noted. Following vertex exploration, a sensor scan toward the goal is made. If the goal is visible, the vehicle moves to the goal. Otherwise, the vehicle moves to the vertex, right-most from the present vehicle location. If that vertex has not been explored, a sensor activation is made and all of the other vertices visible from that vertex location are noted. Next, another sensor activation determines if the goal is visible. If the goal is obscured by another polygon edge, the above process is repeated. Figure 6 shows the interaction within the KMARS rule-set.

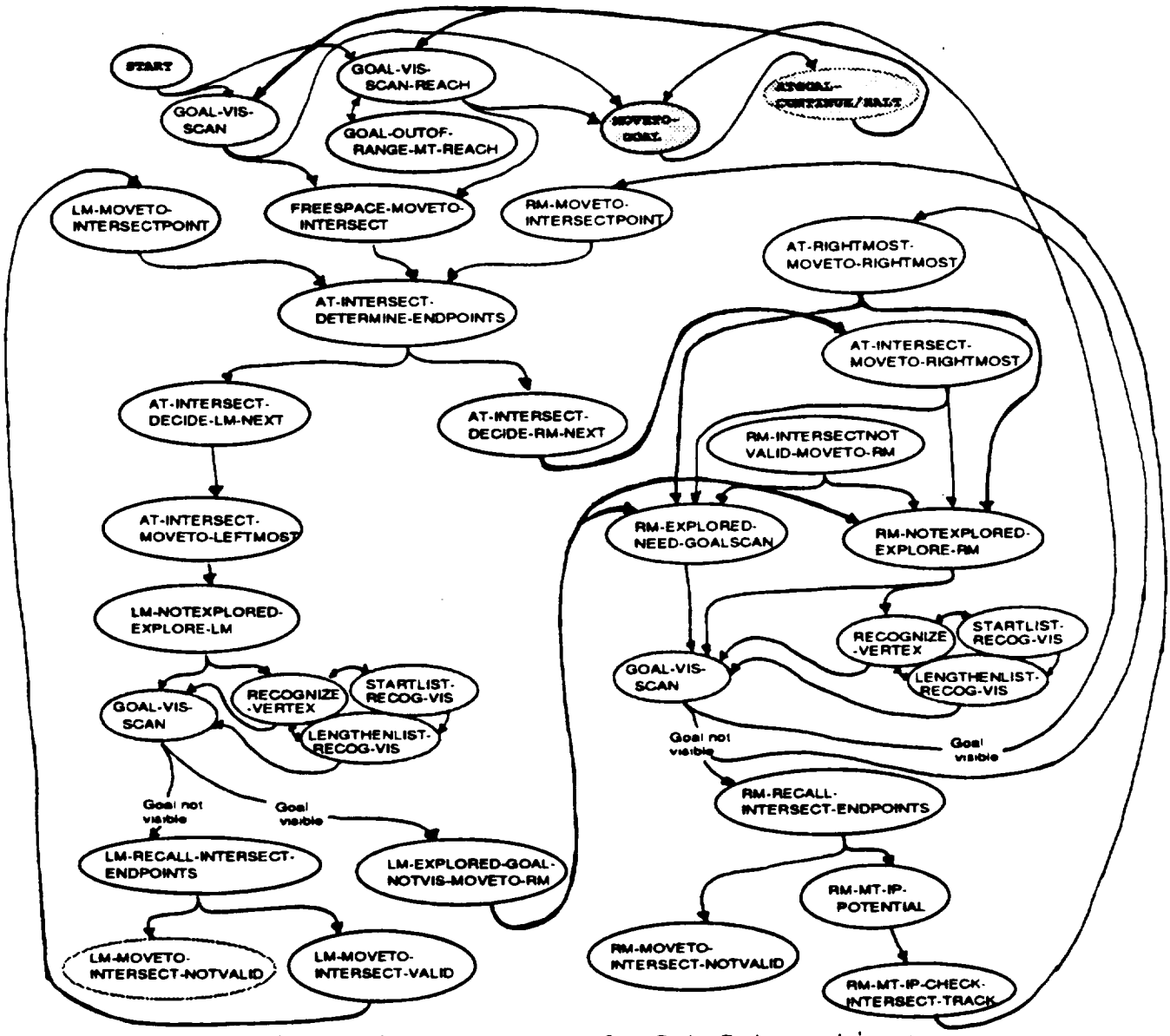

Figure 6 - KMARS Rule-Set Interactions

D. Example of KMARS Rule-Base 'C' Function Interaction

Once the preconditions for the sensor activation rule GOAL-VIS-SCAN have been met, a 'C' sensor function "clips_goal_vis" is called and the necessary arguments that specify both the present vehicle position and the goal position are passed to it.

Using those argument values, the function checks to find the polygon boundary [closest] that blocks the line-of-sight between the vehicle and the goal. If the line-of-sight is not blocked and the goal is within sensor range, the goal_vis function updates working memory with a fact noting that the goal is visible. When the goal is visible, a rule moving the vehicle to the position of the goal is fired. If the goal is blocked by a polygon edge, the sensor function updates working memory with a fact noting that the goal is not visible and a fact noting the coordinates of the point of blockage (i.e., the point-of-intersection of the line-to-goal and the closest blocking polygon boundary).

On other occasions, the vehicle may be at the vertex of a given polygon when its scanner is activated. If the goal is within the vertex-angle of this polygon, the sensor updates working memory with a fact giving the coordinates of the next right-most vertex.

```
(defrule START
   (initial-fact)
 =>
   (retract 0)
   (bind ?veh (clips_get_mouse_position Vehicle free-space))
   (bind ?goal_id (gensym))
   (bind ?veh (clips_get_mouse_position Goal ?goal_id ))
   (assert (Edges-Explored 0))
   (assert (Vertices-Explored 0))
   (assert (Goal-Count 0))
   (assert (Explore-Status Goals 0 Vertices 0 Edges 0))
   (assert (Agenda goal-scan)) )
```

Figure 7 - KMARS Rule START

Actions taken as a result of the START rule firing include:

The function "clips_get_mouse_position" is evaluated. The parameters passed from CLIPS to the function are the word 'Vehicle' and the word 'free-space'. The function in turn sends a prompt to the screen instructing the user to click the mouse at the position where the vehicle is to start from. The value returned by this function is bound to the dummy variable ?veh.

A symbol, needed to identify the next goal position, is generated and is bound to the rule-variable ?goal_id.

The function "clips_get_mouse_position" is again evaluated. The parameters passed from CLIPS to the function are the word 'Goal' and the word assigned to '?goal_id'. The function in turn sends a prompt to the screen instructing the user to click the mouse at the goal position which the vehicle is to find. The value returned by this function is bound to the dummy variable ?veh.

Input into WM are facts that will be used to keep count of Edges-Explored, Vertices-Explored and Goal-Count. Explore-Status will be used to keep track of and to update the exploration status each time a new goal is achieved. Finally, a fact is put into WM that keeps track of future actions to be undertaken.

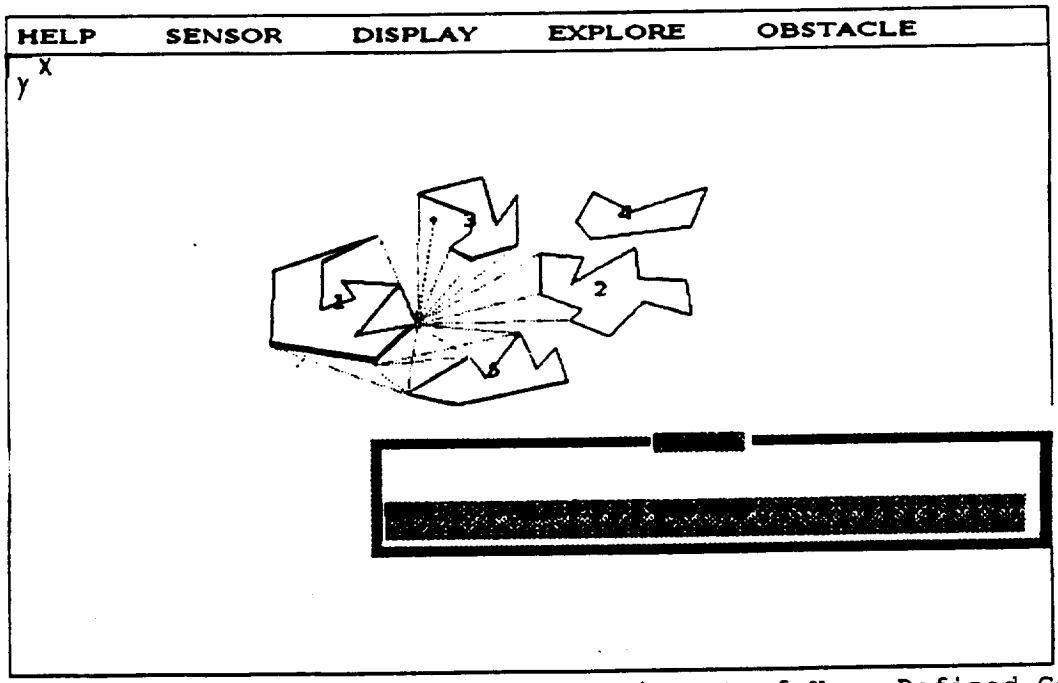

Figure 8 - KMARS Exploration and Attainment of User Defined Goal

E. Results of KMARS Exploration

As the number of traversals to new goals within the environment increases, the exploration of new vertices increases the number of known, obstacle-free paths. These are the vertex to vertex paths that are in working memory. The impact of exploring a vertex is that it does not have to be visited again solely to learn what other vertices can be seen from it. In addition to the saving of time from not having to travel to a known vertex, there is a saving in the time required for sensor exploration scanning. This economy can be monitored as the KMARS vehicle progresses. The next addition to the rule-base should, however, be that of performing a heuristic search of known free paths to find a regionally optimized path to a goal if it is within a totally known region. Although an algorithm has previously been developed to drive exploration and to determine when a complete knowledge of the environment has been acquired, KMARS rules to implement that capability have not yet been developed.

IV. CONCLUSIONS

There is much insight into spatial problem solving that can be derived from using KMARS. In particular this approach allows research into the exploration of unknown environments of an autonomous system. KMARS provides a capability for simulating exploration and navigation activities. The system allows the user to build complex 2D obstacle environments in which to test the efficiencies of various autonomous system vehicle control heuristics. It also allows the user to employ varying sensor characteristics that might be used by a real-world vehicle. Although the strategy implemented in the current rule-set is only one of many that can explore complex 2D environments and achieve goals hidden within the confines of obstacles, the methods implemented in KMARS can be extended to the operations of autonomous systems in real-world 3D environments.

BIBLIOGRAPHY

1. H. Abelson and A. diSessa, *Turtle Geometry*, MIT Press, 1980, pp. 179-199.
2. S.S. Iyengar et al., "Robot Navigation Algorithms Using Learned Spatial Graphs," *Robotica*, Vol. 4, 1986, pp. 93-100.
3. V.J. Lumelsky, A.A. Stepanov, "Dynamic Path Planning for a Mobile Automaton with Limited Information on the Environment," *IEEE Transactions on Automatic Control*, Vol. AC-31, No. 11, November, 1986, pp. 1058-1063.
4. T. Lozano-Perez and M. Wesley, "An Algorithm for Planning Collision-Free Paths Among Polyhedral Obstacles," *Comm. ACM*, Vol. 22, 1979, pp. 560-570.
5. J.D. McKendrick, *Simulation of Autonomous Knowledge-Based Navigation in Unknown Two-Dimensional Environment with Polygon Obstacles*, MS Thesis, Florida Atlantic Univ., Dept. of Computer Engineering, 1989.
6. J.B. Oommen et al., "Robot Navigation in Unknown Terrains Using Learned Visibility Graphs, Part I: the disjoint convex obstacle case," IEEE *Jour. Robotics and Automation*, Vol. RA-12, 1987, pp. 672-681.
7. N.S.V. Rao,et al, "On Terrain Acquisition by a Point Robot Amidst Polyhedral Obstacles," IEEE *Jour. Robotics and Automation*, Vol. 4, No. 4, 1988, pp. 450-455.
8. N.S.V. Rao, "Algorithmic Framework for Learned Robot Navigation in Unknown Terrains," IEEE *Computer*, Vol. 22, No. 6., June, 1989, pp. 37-43.
9. M. Sharir and A. Schorr, "On the Shortest Path in Polyhedral Space," *Proc. 16th Symposium on Theory of Computation*," pp. 144-153, 1984.
10. P.F. Spelt, G. de Saussure, E. Lyness, F.G. Pin, and C.R. Weisbin, "Learning by an Autonomous Robot at a Process Control Panel," IEEE *Expert*, Winter 1989, pp. 8-16.
11. C.R. Weisbin, G. de Saussure, and D.W. Krammer, "Self-Controlled: A Real-Time Expert System for Autonomous Mobile Robot," *Computers in Mechanical Engineering*, Vol. 5, No. 2, Sept, 1986, pp. 12-19.