NARROW AISLE MOBILE ROBOT NAVIGATION IN HAZARDOUS ENVIRONMENTS

Thomas R. Collins, Andrew M. Henshaw
Georgia Tech Research Institute
Georgia Institute of Technology
Atlanta, GA 30332-0840
(404) 894-2508

Ronald C. Arkin
College of Computing
Georgia Institute of Technology
Atlanta, GA 30332-0280
(404) 894-8209

William D. Wester
Simulation Systems Department
Lockheed Aeronautical Systems Company
Marietta, GA 30063-0670
(404) 494-0576

ABSTRACT

Routine monitoring of stored radioactive materials can be performed safely by mobile robots. In order to accomplish this in an environment consisting of aisles of drums, it is necessary to have a reliable means of aisle-following. This work describes the adaptation of successful road-following methods based on visual recognition of road boundaries to the waste storage problem. Since the effort is targeted for near-term usage in normal operating conditions, special emphasis has been given to the implementation of the visual processing on practical (i.e., small, low-cost, and low-power) hardware platforms. A modular, flexible architecture called ANIMA (Architecture for Natural Intelligence in Machine Applications) has been developed at Georgia Tech. The initial versions of this architecture have been based on the Inmos Transputer, a microprocessor designed for parallel applications. To address this application, an ANIMA-based real-time visual navigation module has been developed. The system described here has been implemented onboard a robot in our laboratory

I. INTRODUCTION

The environment for the mobile robot application can be generally described as rows of closely spaced pallets, each containing stacks of drums. It is required that the robot be able to travel between the rows without colliding with any drums or pallets. The drums vary in size, and they may extend past the edge of the pallets. The resulting visual boundary (at the floor) is a series of line segments and curves, sometimes interrupted by gaps. In order to provide steering cues, the visual algorithm must be able to identify the approximate left and right boundaries, in spite of their deviation from straight lines, and project an imaginary center line. The basis for this work is a mobile robot navigation system developed by Arkin,[1] using the Fast Line Finder (FLF) developed at the University of Massachusetts.[13] This algorithm has been used successfully on minicomputers and workstations by its authors and others, including some previous work at Georgia Tech, but for this application it was necessary to port it to a system more suitable for actual deployment on a robot.

To provide navigation in the semi-structured environment found in a hazardous waste storage facility, a sensor system should use the inherently available information of the storage structure. Since these structures tend to be organized for the convenience of visual observation, it is assured that visual navigation cues will be present. For this reason, vision currently provides the greatest information content among reasonably-priced sensor systems. In addition, vision can be applied to the task of drum inspection, making dual use of a single sensor.

A drawback to the use of vision in a sensor system is the large volume of data that must be processed to extract the necessary navigational cues. Associated with this drawback is the high computational processing needed to provide reasonable throughput rates for real-time control. Sophisticated image processing algorithms can alleviate the first problem, at the expense of adding to the second, simultaneously reducing the quantity and increasing the quality of the sensor data.

Our approach to this problem is to:

- utilize the simplest reliable navigational cues available from the environment,
- perform the necessary image processing to extract those cues and reduce the data volume to a manageable level,
- provide processing power sufficient to implement these algorithms within the time constraints provided by the robotic control system, and

- implement the system as a separate module so that it can operate without impacting the performance of established components

To the end of making this system completely portable, we have designed a modular parallel-processing architecture that is largely host-independent. The central processing element of this architecture is the Inmos Transputer, a high-performance microprocessor that has been designed for parallel processing. For this application, we used Transputer-based frame grabbers, processing elements, and interfaces for communication and storage. All of these components are available as modular, off-the-shelf products that communicate through high-speed communication links. Host interfaces for Transputer systems are available for most popular platforms. For many of these systems, the interface is integrated into a carrier board that provides for the physical mounting of Transputer-based modules (or TRAMs). To demonstrate the capabilities of a Transputer-based control system on a commercial (Denning) robot, a custom carrier was developed to match the robot's native STD-bus form factor.

Another assembly used for testing of the system was a single Eurocard carrier (similar to the GISC VME form factor) that contained all of the modules necessary for control of a three-axis motorized camera platform. Output of the vision system was used to direct the camera motor control system, which emulated the essential aspects of robot steering.

Our research concentrated on aisle boundaries as the essential visual cues for safe navigation. Two algorithms were studied for effectiveness of aisle-boundary extraction: the Fast line finder (FLF) and the Fast region segmenter (FRS). Only an FLF-based navigation system is discussed here. This system extracts left and right aisles by sampling a single video frame, performing edge detection, building connected regions of pixels with similar line orientations, and filtering out low-interest orientations. The output from the algorithm is used to derive an estimate of the aisle center. Using this estimate and knowledge of the current heading, the controller servos the robot to follow the expected center line.

II. TRANSPUTER-BASED CONTROL SYSTEM

Autonomous machines with sensory, manipulative, and locomotive capabilities are a significant class of intelligent systems holding great promise for performing hazardous or mundane tasks. Although much work has been performed with isolated aspects of intelligent machines, including vision, sonar, manipulator control, and knowledge-based reasoning, the algorithms are often not considered within the context of a complete machine. In many prior efforts, both software architectures and hardware architectures have been developed to meet the requirements of specific projects, with little regard to reusability in other applications. Often, experimental systems are not robust, failing due to relatively minor environmental variations or task redefinitions. This section describes the integration of two separate efforts to address these problems. One is a robot control software architecture that has been demonstrated to perform a variety of tasks well, and the other is a targeted, yet flexible, computer architecture that provides modularity and expandability.

A. AuRA – the Autonomous Robot Architecture

AuRA is a hybrid architecture encompassing aspects of both deliberative and reactive control. It consists of 5 major subsystems:

- Perception – charged with collecting and filtering all preliminary sensory data.
- Cartographic – concerned with maintaining long-term memory (*a priori* models of the environment), short-term memory (dynamically acquired world knowledge), and models of spatial uncertainty.
- Planning – consists of a hierarchical planner (the deliberative component) and the motor schema manager (the reactive component).
- Motor – the interface software to the specific robot to be controlled.
- Homeostatic – a component concerned with dynamic replanning in light of available internal resources.[4]

The overall architecture has been described in detail elsewhere. The reader is referred to Arkin for more information.[1, 3]

The hardware migration to ANIMA thus far has been concerned with the reactive and perceptual components of the system that run within the confines of the motor schema manager. Figure 1 presents the logical relationships between the varying schemas which constitute this portion of AuRA.

Action-oriented perception forms the underlying philosophy for channeling sensory information to the motor schemas (behaviors).[2] Only the information that is essential to a particular motor behavior is transmitted to it, essentially on a need-to-know basis. The message-passing

paradigm found in ANIMA is well suited for this type of information flow.

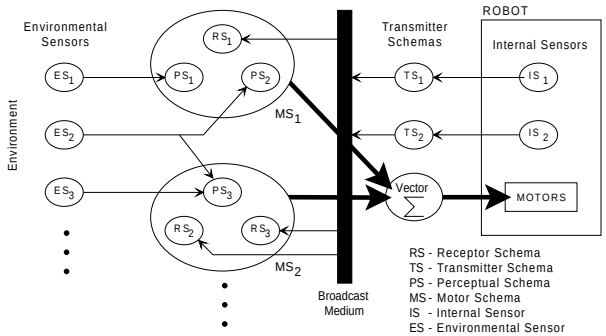

Figure 1. Inter-schema relationships.

Each of the active motor schemas generates a velocity vector in a manner analogous to the potential fields method[14, 15] with the individual results being summed, normalized and transmitted to the robot for execution. The speed of operation is highly dependent on the rate of processing of incoming sensory data. The parallelism found in the Transputer implementation described below is a natural match for this aspect of the AuRA architecture.

B. ANIMA hardware architecture

We have developed a flexible, real-time platform for the development of AuRA and other software architectures. The skeleton of our hardware architecture, ANIMA (Architecture for Natural Intelligence in Machine Applications), has been developed from basic principles.[8] The deliberative component controlling the input and output subsystems is called the *Reasoner*. A major aspect of this reasoning capability is the need to maintain some sort of world model based on sensory input, at least for anything more than basic reactive behavior.

The vast majority of intelligent machine research has assumed that the input/output devices, just as in a conventional computer, are largely independent in their low-level operation (at or below the level of the device driver). For input devices, the combination of these independent streams of data has often been referred to as *sensor integration*, and we include a process, called the *Integrator*, to perform this task. On the output side of the structure, the most appropriate term is *coordination*, although the specific definition varies considerably in the literature. Corresponding to the *Integrator*, we include a process called the *Coordinator*. Often, researchers wish to place *task planning* in this position. The most common form of task planning is *trajectory planning* (or *path planning*) in which one or more effectors are given commands that, taken together, result in a desired trajectory within the environment. Although this may be viewed as a sort of coordination, it is really a process much more closely aligned with the *Reasoner* in our nomenclature. A planned trajectory, given in a high-level description, could serve as an input to the *Coordinator* process, which would then take control of the effectors in order to implement the trajectory.

These additional parallel processes are illustrated in Figure 2. The independent sensor subsystems are called *logical sensors*, in much the same sense as those of Henderson[11] or Crowley.[10] At this point, a logical sensor is best thought of as a combination of a physical sensor, capable of estimating some property of the environment or the machine itself, and a generalized device driver. The extension of this concept to the *logical effector* is straightforward. Taken together, logical sensors and logical effectors are called *logical devices*. A single logical device can be composed of multiple physical devices, with appropriate drivers. This would be desirable in cases where the physical devices were virtually identical (except perhaps in physical location, scaling, or some other trivial factor), allowing the main driver to give the appearance of a single effective logical device.

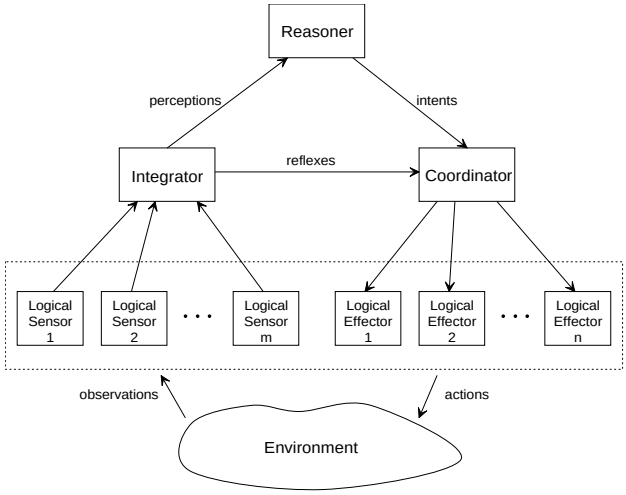

Figure 2. ANIMA structure.

C. Hardware implementation

Implementations of this architecture have been developed based on the Inmos T800, a member of the Transputer family of microprocessors developed for parallel processing. Each Transputer provides four high-speed serial links for the required processor interconnection.

Using commercial off-the-shelf components, the module consists of a frame-grabber, a dedicated FLF

processor, and an optional graphics display controller. The FLF processor is a 30 MHz T805 Transputer, which includes a floating-point processor capable of over 2.5 MFLOPs, but requiring few external components and minimal power. The processor and its four Megabytes of dynamic memory are packaged in the standard Transputer Module, or TRAM, format. Both the frame-grabber and the display controller also utilize T805 processors and are packaged as TRAMs as well. The ANIMA architecture requires no parallel data busses or backplanes of any kind. Instead, it consists of modular components connected only by high-speed serial links. This allows the processors to be distributed to any convenient locations within the intelligent machine.

D. Case studies – "Buzz" and "Stimpy"

AuRA and ANIMA were first brought together in the development of a machine called "Buzz," to compete in the first robot exhibition and competition sponsored by the American Association for Artificial Intelligence (AAAI). The competition stressed the ability of mobile robots to explore an arena, avoid static or moving obstacles, locate goals, and visit goals in specified order.[5,9] Buzz required an onboard PC to host its network of Transputers. To address more directly the needs of WSRC, a compact, low-power, onboard implementation was developed and used in a robot called "Stimpy." Like Buzz, this version was designed for one of the Denning commercial robots (an MRV-2), and it was used in a AAAI competition (held in July 1993 in Washington, D. C.).

The Denning MRV-2 is a three-wheeled cylindrical robot intended for general-purpose use, mainly in research. All three wheels turn simultaneously, providing (approximately) the ability to turn in place. The body itself does not rotate, except for gradual precession resulting from non-uniform slippage of the wheels against the floor. Twenty-four sonar sensors are equally spaced around the body.

E. STD Host Adapter

Integration of the Transputer Control System with the Denning MRV-2 was achieved by building two STD host adapters. These adapters meet the specifications for Transputer Module (TRAM) motherboards provided by Inmos[12] and allow for the mounting of up to four TRAMs on each adapter. The TRAM format has been defined by Inmos to cover various-sized modules that interconnect through Transputer-compatible links. The basic size of a TRAM is 3.66" by 1.05" (referred to as a Size 1 TRAM) with 16 pins for electrical services and mechanical retention. The largest TRAM, a Size 8, is 3.66" by 8.75" with 128 pins for retention, of which only 16 are used by the TRAM for electrical services. On TRAMs larger than Size 1, the extra unused pins are passed through the module so that other TRAMs may be piggybacked to avoid the loss of TRAM sites. In this fashion, the STD host adapter could support one Size 4 TRAM plus three Size 1 TRAMs.

III. VISION-BASED NAVIGATION

Using vision to guide a mobile robot has many difficulties. Among these is the problem of reducing a large amount of pixel-based information into a smaller representation that lends itself to computer-based reasoning. We have investigated two approaches developed by the University of Massachusetts while performing research for the DARPA Autonomous Land Vehicle Project.[13] Both of these algorithms, the Fast Line Finder (FLF) and the Fast Region Segmenter (FRS), rely upon selective processing to achieve reasonable performance levels in the real-time environment of a mobile robot. Selective processing is derived from *a priori* knowledge of the robot's environment and goals and can be used to limit the processing of visual information to regions of high interest. In the FLF-based research performed here, areas of interest are confined to regions of the visual field in which pathways are likely to be found. Additional selection is performed by processing only objects whose features, such as line orientation, match the constraints necessary for visual navigation.

The Fast Line Finder algorithm, based upon Burns's algorithm, works by extracting a set of image intensity gradients for each pixel in an image and groups pixels with related orientations and positions together.[7] From these groups, lines can be developed. In this research, these lines are used to define speculative road edges from which a path can be derived.

The following steps are taken by the Fast Line Finder algorithm to generate a set of lines that are of use in vehicle navigation:

1. Computation of the image intensity gradient (magnitude and direction) for each pixel
2. Quantization of pixel direction into a set of orientation ranges.
3. Grouping of adjacent pixels of like orientation into regions
4. Fitting of lines to match regions

The Fast Line Finder optimizes the processing of these steps by ignoring irrelevant pixels during the early stages and irrelevant regions during the later stages.

Figure 3. Example of an accurate FLF centerline determination.

Figure 4. Illustration of typical error sources in FLF algorithm.

For robot navigation, the line fragments provided by the Fast Line Finder must be accompanied by a control system that can interpret the results in a useful manner.[6] In this implementation, the navigation system[1] constrains the generation of line segments by the FLF algorithm and from these attempts to determine path edges and path centerline. The path edges are derived by applying the robot's knowledge about its current position and approximate position of the path from the last image. Using this information, the control system can make a prediction as to the location and orientation of the path edges in the current frame. This prediction is used to tune the FLF to produce sets of lines that can be used as fragments for candidate road edges. Fragments are considered by first filtering out those line segments whose position or orientation does not match the expectations for either of the path edges. Two sets of fragments, one for each path edge are produced, from which line equations are generated. From these left and right edges, a center line is computed which provides a basis for the generation of a new directional heading for the robot.

Figure 3 shows the operation of the navigation system under good conditions. Because the FLF has produced a majority of line fragments that closely match the aisle boundaries, the centerline is accurately computed. Due to orientation filtering, relatively few lines are generated from the drum images, even in the regions occupied by the rectangular labels. As will be discussed in more detail later, the most useful images are acquired when the camera is pointed steeply downward, as seen here. Ideally, none of the field of view is used for extraneous information above the horizon.

As shown in Figure 4, interference from extraneous fragments can cause the performance to deteriorate. In this image, unwanted fragments pull the left aisle boundary toward the edge of the frame. However, even in this case, the errors are not navigationally significant. Furthermore, the estimated boundaries are close enough to adequately predict their position in the next sequential image.

IV. TESTING

The environment for the mobile robot application can be generally described as rows of closely spaced pallets, each containing stacks of drums. The rows are separated by aisles that are large enough for the robot to pass through (about 1 m). It is required that the robot be able to travel down these aisles without colliding with any drums or pallets. The drums vary in size, and they may extend past the edge of the pallets. The resulting visual boundary (at the floor) is a series of line segments and curves, sometimes interrupted by gaps. In order to provide steering cues, the visual algorithm must be able to identify the approximate left and right boundaries, in spite of their deviation from straight lines, and project an imaginary center line.

Two methods were established for testing of the vision system and related algorithms. The first setup used videotape images from the Fernald Environmental Management Project that simulated motion of a robot through the waste storage facility. These images were digitized and stored for processing by both a Sun workstation and the Transputer-based vision system. The second test setup consisted of simulated environment

constructed from 50-gallon drums and pallets assembled into aisles. These aisles were navigated by a camera-equipped Denning robot driven by a remote Sun workstation running an image processing algorithm. The images shown here are all from the Fernald videotape.

Figure 5.

For demonstration and testing purposes, a Transputer-based three-axis motor controller was developed to drive a pan-tilt-verge camera platform. The implementation of this controller demonstrated the division of labor achievable with parallel-processing systems. The motor controller runs on a separate Transputer and interfaces with the 3-axis platform via an RS-232 serial channel. Communication with the rest of the Transputer network is through the high speed Transputer links. This real-time interface was added without impacting the performance of the processor-intensive vision computations. Demonstration of the entire control system was achieved by servoing the camera to look in the direction of the computed aisle centerline. In these tests, the vision system did not use images acquired from the camera, but was instead driven by previously acquired images from the Fernald facility.

Figures 5 and 6 illustrate the performance of the FLF-based navigation system. For these examples, all of the derived lines resulted from only knowing that the aisle was approximately in the center of the image.

V. CONCLUSIONS

Both the Sun and Transputer implementations ran in under six seconds per frame. Since only a single Transputer was used for the image processing, significant speedup could be achieved by using additional processors, without adding significantly to the size or power consumption (the frame grabber is already the largest single component). In order to perform thorough radiation detection, the robot's speed would be relatively low, which constrains image processing speed more than the on-board hardware. Since Transputers were used, the architecture can easily be expanded to achieve higher frame rates if required. An example of an obvious parallelization would be to use a different processor for the left and right boundary extractions.

Figure 6.

Testing of the FLF-based system with images of stored hazardous waste showed that this implementation is robust and well suited for autonomous navigation. Use of aisle-boundary extraction techniques appears to provide suitable navigational cues and is tolerant of variations in the storage environment.

The navigation system always found a reasonable approximation to the path, as long as it had a reasonable expectation of where to look. Normally, this expectation arises from previous knowledge combined with dead-reckoning to account for position changes, but it is also possible to use the Fast Region Segmenter (FRS) as a means of providing bootstrap information to the FLF. The FLF appears to be a more reliable primary strategy, supplemented by the FRS for bootstrapping or confirmation.

This research has shown that a low-cost, high-performance vision system can be built using a modular parallel-processing system. Because of the modularity and communication techniques, portions of this system could be rapidly integrated into a completely different hazardous

waste inspection system. Additionally, this architecture supports both automated and manual visual inspection since the vision system (and its associated computing power) need not be dedicated solely to the visual navigation task. Standard image compression techniques could be easily adapted to this system with the parallel-processing ability providing the necessary computing power for real-time image archiving.

ACKNOWLEDGMENTS

This work was sponsored by the Westinghouse Savannah River Corporation. Our thanks to the Fernald Environmental Management Project for providing videotape of their storage facilities.

REFERENCES

1. Arkin, R. C. "Towards Cosmopolitan Robots: Intelligent Navigation of a Mobile Robot in Extended Man-made Environments," Ph.D. Diss., COINS Tech. Rep. 87-80, U. Mass., 1987.

2. Arkin, R. C. "The Impact of Cybernetics on the Design of a Mobile Robot System: A Case Study," *IEEE Trans. Sys. Man Cyber.*, vol. 20, no. 6 (Nov/Dec 1990), pp. 1245-1257.

3. Arkin, R. C., Riseman, E. and Hanson, A. "AuRA: An Architecture for Vision-based Robot Navigation," *Proc. of the 1987 DARPA Image Understanding Workshop*, (Los Angeles, CA), pp. 417-431.

4. Arkin, R. C. "Homeostatic Control for a Mobile Robot: Dynamic Replanning in Hazardous Environments," *Journal of Robotic Systems*, vol. 9, no. 2.

5. Arkin, R. C. *et al.* "Buzz: An Instantiation of a Schema-Based Reactive Robotic System," *Proc. International Conference on Intelligent Autonomous Systems: IAS-3* (Pittsburgh, PA), Feb. 1993.

6. Arkin, R. C., Riseman, E., and Hanson, A. "Visual Strategies for Mobile Robot Navigation," *Proceedings of the 1987 IEEE Workshop on Computer Vision*, December 1987.

7. Burns, J., Hanson, A., and Riseman, E. "Extracting Straight Lines," *Proceedings of the IEEE 7th International Conference on Pattern Recognition*, (Montreal, Canada), 1984.

8. Collins, T. R. "A Computer Architecture for Implementation within Autonomous Machines," Ph.D. Diss., Ga. Inst. of Technology, 1994.

9. Collins, T. R., Arkin, R. C., and Henshaw, A. M. "Integration of Reactive Navigation with a Flexible Parallel Hardware Architecture," *Proc. 1993 IEEE Intl. Conf. Robotics and Automation*, vol. 1, pp. 271-276.

10. Crowley, J. L. "Dynamic World Modeling for an Intelligent Mobile Robot" *Proc. 1984 Intl. Conf. Patt. Rec.*, pp. 207-210.

11. Henderson, T. and Shilcrat, E. "Logical Sensor Systems" *J. Robotic Systems*, vol. 1, no. 2 (1984), pp. 169-193.

12. Inmos, Ltd., *The Transputer Development and iq Systems Databook*, Redwood Press, Ltd., 1991.

13. Kahn, P., Kitchen, L., and Riseman, E. M. "Real-Time Feature Extraction: a Fast Line Finder for Vision-Guided Robot Navigation," *University of Massachussetts Technical Report 87-57*, 1987.

14. Khatib, O. "Real-time Obstacle Avoidance for Manipulators and Mobile Robots", *Proc. 1985 IEEE Int. Conf. Rob. Aut.* (St. Louis, MO), pp. 500-505.

15. Krogh, B. "A Generalized Potential Field Approach to Obstacle Avoidance Control", *SME - RI Technical Paper* MS84-484, 1984.