Journal of Aviation/Aerospace Education & Research

Volume 11
Number 2 *JAAER Winter 2002*

Article 1

Winter 2000

An Investigation of Different Modeling Techniques for Autonomous Robot Navigation

Jedidiah Crandall

FORUM

AN INVESTIGATION OF DIFFERENT MODELING TECHNIQUES FOR AUTONOMOUS ROBOT NAVIGATION

Jedidiah Crandall

ABSTRACT

This research aims to give recommendations towards modeling the navigation control architectures for an autonomous rover designed for an unstructured, outdoors environment. These recommendations are equally applicable to other autonomous vehicles, such as aircraft or underwater vehicles. Many successful architectures for this application have been developed, but there is no common terminology for the discussion of robotics architectures and their properties in general. This paper suggests the use of terms borrowed from administrative theory to facilitate interdisciplinary dialog about the tradeoffs of various kinds of models for robotics and similar systems.

Past approaches to modeling autonomous robot navigation architectures have broken the architecture up into layers or levels. The upper levels or layers make high-level decisions about how the robot is going to accomplish a task, and the lower levels or layers make low-level decisions. This is analogous to a CEO of a corporation telling the managers how he wants the corporation to work towards its goal. The managers each oversee a part of the corporation. The workers are told what to do, but still make low-level decisions such as how hard to twist a screw, what tool to use to remove a rivet, or to do something other than what they were told in the interest of safety.

Traditionally, there have been two or three layers for robot architectures, and every module developed fits into one of these layers. Every branch of the hierarchy has one module in each of the layers. The reasons given for breaking the architecture up into two or three layers vary from implementation to implementation. This paper aims to take a more generalized view.

The benefits of the two or three layered approach are well published, including reliability, reusability, and scalability among others. This paper asserts that these layers are unnecessary, and that vertical specialization can be implemented to a different degree on different branches of the hierarchy. For example, the velocity controller on a rover might have two layers, whereas the steering controller on the same rover might have four. They share the highest layer, which is the navigational planner that coordinates them. But the two branches of hierarchy between the navigational planner and the two actuators look very different from one another. This facilitates a decentralization of the decision making duties and greater freedom in the process of breaking the navigation system up into modules.

INTRODUCTION

The NASA Space Grant Rover Project at Embry-Riddle Aeronautical University in Prescott, Arizona is building a rover that will navigate autonomously in an unstructured outdoors, environment. A "tour-guide" rover is also being built that will serve as a prototype for this rover. The "tour-guide" rover will navigate in a structured, indoors environment.

A heterogeneous model based on the model presented in (Pirjanian, 1995) is being implemented for the "tour-guide" rover. This heterogeneous model has three levels for navigation: Mission, Skill, and Reactive. The levels are separated by the kind of model representation that each uses. A semantic model is usually a high-level representation such as a graph or a procedural representation. A geometric model gives positions in terms of distances and angles, like a CAD model. An iconic model is usually a bitmap of the robot's world where

different numbers represent obstacles, paths, and other objects.

The Mission level of the "tour-guide" implementation uses a semantic model and produces high-level instructions when given the current position and goal position of the rover. These instructions look like "Go ten meters down the hallway," "Turn right ninety-degrees," and "Stop and play the audio file that talks about the Controls Lab."

The Skill level executes these commands using geometric models. For example, one Skill level function uses edge detection and a Hough transform to detect the edges of the hallway and follow them. The reactive level uses a semantic model of the edges detected in front of the rover to avoid collisions by stopping.

Other implementations have used two or three levels, which are also commonly called layers, but there is no consensus about why the architectures are broken into levels. In (Alami, 1993) the need for adaptability is juxtaposed against the need for reliability. Adaptability means that the robot can plan its navigation autonomously. Reliability means that the robot should have low-level behaviors so that it can be asserted that it will work properly before it is launched. These seemingly opposing criteria are met with two different levels: the Decision Level for adaptability and the Functional Level for reliability.

(Alami, 1998) adds another level, the Executive Level, which executes the high-level commands from the Decision Level. A similar three-leveled approach is presented in (Simmons, 1998), which calls them the Planning Layer, the Executive Layer, and the Behavior Layer. (Gat 1998) uses the names Deliberator, Sequencer, and Controller and the need for three layers is attributed to the problem of internal state, where information about the world is stored in the robots memory. The Controller provides a tight coupling between detection and reaction by using no internal state in the making of decisions. The Sequencer uses some internal state data about the past to execute the commands of the Deliberator. The Deliberator makes plans through time-consuming search algorithms and relies on internal state data to make predictions about the future.

(Coste-Manière, 2000) points out that the use of a layered approach creates different levels of abstraction, which promotes verification and validation. (Nesnas, 2001) uses a two layered approach, with a Decision Layer and a Functional Layer, to facilitate the reusability of components in different robots. The architecture presented in (Singh, 2000) has a Global Planning layer and a Local Traversability layer, a major distinction being that "all processing and decision making is done on a single processor." This means that the levels are not always broken up physically.

What are the underlying reasons for these layers? Reusability? Verifiability? Internal state? Reflexivity? Scalability? Should there be two layers or three? The next section of this paper will present a terminology for discussing these architectures in general. The section after that suggests some advantages and disadvantages of what will be called *vertical specialization.* The conclusion will discuss plans for future work on the Embry-Riddle outdoors rover and make recommendations for modeling its architecture.

TERMINOLOGY

Before engineers from a variety of disciplines can have a productive discussion about a proposed robot architecture they should have a common terminology for talking about the tradeoffs involved. The terms presented here are borrowed from [8]. Administrative theory and its hierarchic organizations parallel robotics architectures in many ways.

Horizontal specialization will be defined as the division of work by the scope and nature of duties. For example, one controller might be specialized for steering while another might be specialized for controlling the throttle.

Vertical specialization is the division of decision-making duties. An administrator in administrative theory makes high-level decisions about the future and goals of the institution, and a worker on the assembly line makes low-level decisions about how to get their job done and deal with real-time constraints. The decision-making duties are similarly divided in the hierarchic models for autonomous robot navigation.

Taken directly from (Simon, 1997): "A subordinate [node] may be said to accept *authority* whenever [it] permits [its] behavior to be guided by a decision reached by another [node]." Authority is an important concept in modeling a robot's behavior because it can provide trace-ability from high-level decisions to real observed behaviors of the robot.

Bounded rationality will be defined here in a way that includes the engineers building the robot. The ability

of a node to reach a decision from given decisional premises in real-time is bounded by software complexity and processing speed. The bound on software complexity is due more to the ability of the engineer to understand and verify the software than anything else.

Communication will be defined as "any process whereby decisional premises are transmitted from one [node] of an organization to another (Simon, 1997)." These decisional premises might be sensory data, high-level commands to be executed by a lower-level node, or progress reports transmitted to higher-level nodes by lower-level nodes.

THE TRADEOFFS OF VERTICAL SPECIALIZATION

The reasons for horizontal specialization are fairly obvious. In an autonomous airplane it is easy to see why it might be useful to have a different controller for the ailerons than for the elevators or the rudder. This promotes reusability, verifiability, scalability and all kinds of other desirable properties. But what do we gain from vertical specialization?

First of all, if there is any horizontal specialization then vertical specialization is necessary for coordination. The rudder on an autonomous airplane can be programmed to control yaw and avoid slip. But there must be some coordination between the ailerons and elevator if the airplane is going to get where it intends to go. This necessitates a higher-level node with the authority to turn and climb and descend to execute a navigational plan.

The division of authority that vertical specialization allows can help with the debugging process, as well. A decision can be traced back to the node that made it. It is not entirely unfeasible that a system could be implemented that had only one node to make all of the decisions. One example would be a learning robot. But then if the robot does not behave as specified it will be difficult to discern why.

Placing authority at different levels can be used to satisfy problems associated with reflexivity and global planning. Global planning is done best by a central, high-level node because it has the authority to make the plan be implemented. Low-level nodes should still have as much authority as possible within their domains, though. This allows for reflexivity to avoid hazards. For example, the steering mechanism in a rover might sense that the rover is about to roll and turn the wheels into the roll to try to prevent it. This should be done regardless of whether or not it is part of the global plan. The goal of a rover is not just to navigate to its destination, but also to do so without incurring physical damage or creating a hazard to its environment.

Finally, vertical specialization can provide different levels of abstraction to promote modularity. The global planner does not necessarily need to know whether it is being used on a rover or a hovercraft. This modularity facilitates things like reusability and incremental development.

Less obvious are the disadvantages of vertical specialization. In the extreme case vertical specialization can create a "tall, skinny" hierarchy that inhibits the flow of decisions from the high-level planner down to the low-level actuators.

Vertical specialization requires more bandwidth for communications unless authority can be decentralized. If a node leaves no authority to its subordinates then there is no need for vertical specialization. Beyond a certain point, vertical specialization complicates interactions between nodes more than it simplifies the decisional processes within the nodes.

CONCLUSIONS AND FUTURE WORK

The recommendation of this paper is that a navigational model should be broken up into nodes that have well defined decisional premises, decision-making processes, and authority. The authority can be divided in such a way that it is decentralized enough for reflexivity and bandwidth reduction, but centralized enough to keep the robot always coordinated and working towards its goal.

Future work at Embry-Riddle Aeronautical University in Prescott, Arizona will be to implement a model for the outdoors rover, which will probably be an automated All-Terrain Vehicle. The process of designing, building, and debugging a robot's architecture should be a good test of the utility of the ideas presented in this paper. It is anticipated that several disadvantages to breaking away from the layered approach might arise.❑

Jedidiah Crandall is a senior majoring in Computer Science at Embry-Riddle Aeronautical University in Prescott, Arizona. He is team coordinator of the NASA Space Grant Rover project and is currently working to develop an interactive learning tool to help students understand buffer overflows. He spent the last two summers developing software to process streaming video input and navigate a robot through the halls of the King Engineering Center at the ERAU Arizona campus. After receiving his bachelor's degree, he will attend graduate school and do research in computer security.

ACKNOWLEDGEMENTS

This research was funded by the Ronald E. McNair Scholars Program at Embry-Riddle Aeronautical University in Prescott, Arizona. Dr. Raymond Bellem, my McNair mentor, as well as Dr. David Viger and David Brandstein, have all helped me a great deal.

The NASA Space Grant Rover Project is funded by the Arizona Space Grant Consortium and was started thanks to Dr. Ron Madler.

The people I am most indebted to for this research are Dr. Gary Gear and Kjersti Kleven, with whom I've had many discussions about modeling the architecture of an autonomous rover.

REFERENCES

Alami, R. Chatila, R. Espiau, B. (1993) Designing an Intelligent Control Architecture for Autonomous Robots.

Alami, R. Chatila, R. Fleury, S. Ghallab, M. Ingrand, F. (1998) An Architecture for Autonomy. *International Journal of Robotics Research (Special Issue on "Integrated Architectures for Robot Control and Programming").*

Coste-Manière, E. Simmons, R. (2000, April) Architecture, the Backbone of Robotic Systems. *Proceedings of the 2000 IEEE International Conference on Robotics and Automation, San Francisco, CA.*

Gat, E. (1998) On Three-Layer Architectures. *This paper appears in D. Kortenkamp et al. AI and Mobile Robots. AAAI Press.*

Nesnas, I. Volpe, R. Estlin, T. Das, H. Petras, R. Mutz, D. (2001) Toward Developing Reusable Software Components for Robotic Applications. *Jet Propulsion Laboratory, California Institute of Technology, Pasadena, CA 91106.*

Pirjanian, P. Christensen, H. (1995, Nov. 1) Hierarchical Control for Navigation Using Heterogeneous Models. *Submitted to Dagstuhl Seminar, "Environment Modeling and Motion Planning for Autonomous Robots," Proceedings.*

Simmons, R. Apfelbaum, D. (1998) A Task Description Language for Robot Control. *School of Computer Science, Carnegie-Mellon University, Pittsburgh, PA 15213.*

Simon, H (1997). Administrative Behavior Fourth Edition. *The Free Press.*

Singh, S. Simmons, R. Smith, T. Stentz A., Verma, V. Yahja, A., Schwehr, K. (April 2000) Recent Progress in Local and Global Traversability for Planetary Rovers. *Proceedings, IEEE Conference on Robotics and Automation, San Francisco.*