Obstacle Avoidance using Predictive Vision based on a Dynamic 3D World Model

D. Paul Benjamin[a], Damian Lyons[b], Tom Achtemichuk[a]

[a]Pace University Computer Science Department, 1 Pace Plaza, New York, New York 10038;
[b]Fordham University Department of Computer & Information Science 340 JMH, Fordham University, 441 E. Fordham Rd., Bronx, New York 10458

ABSTRACT

We have designed and implemented a fast predictive vision system for a mobile robot based on the principles of active vision. This vision system is part of a larger project to design a comprehensive cognitive architecture for mobile robotics. The vision system represents the robot's environment with a dynamic 3D world model based on a 3D gaming platform (Ogre3D). This world model contains a virtual copy of the robot and its environment, and outputs graphics showing what the virtual robot "sees" in the virtual world; this is what the real robot expects to see in the real world. The vision system compares this output in real time with the visual data. Any large discrepancies are flagged and sent to the robot's cognitive system, which constructs a plan for focusing on the discrepancies and resolving them, e.g. by updating the position of an object or by recognizing a new object. An object is recognized only once; thereafter its observed data are monitored for consistency with the predictions, greatly reducing the cost of scene understanding. We describe the implementation of this vision system and how the robot uses it to locate and avoid obstacles.

Keywords: Mobile robotics, active vision, cognitive architecture, predictive vision

1. INTRODUCTION: ADAPT

The ADAPT project (**A**daptive **D**ynamics and **A**ctive **P**erception for **T**hought) is a collaboration of three university research groups at Pace University, Brigham Young University, and Fordham University to produce a robot cognitive architecture that integrates the structures designed by cognitive scientists with those developed by robotics researchers for real-time perception and control. Our goal is to create a new kind of robot architecture capable of robust behavior in unstructured environments, exhibiting problem solving and planning skills, learning from experience, novel methods of perception, comprehension of natural language and speech generation.

We are focusing on the inability of robots to understand their environment and act appropriately. Currently, all of the perceptual processing and decision-making is done by the programmers before the robot begins its task. The semantics for the symbols and structures the robot uses are determined and fixed by these programmers. This leads to fragmented abilities and brittle performance. The robots cannot adapt their knowledge to the task, cannot solve tasks that are even slightly different from those they have been programmed to solve, cannot communicate effectively with humans about their goals and performance, and just don't seem to understand where they are.

The overall research goal of this project is to implement and test a robot cognitive architecture. Much work has been done in the cognitive science community to develop general unified theories of human cognition. These theories have been tested on a number of tasks, and exhibit a wide range of human-like behaviors, including perception, effective problem solving, emotion, and learning. It is time to extend these models to encompass robotic applications. This will bring a whole new range of capabilities to robots, ranging from improved perceptual and problem solving skills to improved communication and cooperation with humans. The specific abilities concentrated on in this project are the learning of perception and the learning of navigation.

This paper focuses on the vision system of ADAPT, including the details of the vision hardware and software, and the interaction with the world model. A separate paper in this conference [3] explains ADAPT as a whole. In the next

section, we provide a brief sketch of ADAPT as background material. The subsequent sections explain the structure of the vision system.

1.1. RS and Soar

ADAPT organizes its knowledge using *sensory-motor schemas*. Schemas are organized networks of actions (abstract and/or concrete), percepts, words, facts and beliefs about some aspect of the world. Arbib (1992) has developed a schema theory that essentially describes our use of the term. His schema theory is especially useful because it has been precisely specified with a formal semantics (Steenstrup, Arbib & Manes, 1983) and implemented in a language for real-time concurrent perception and action (Lyons & Arbib, 1989). We use this language, called RS (for Robot Schemas), to give our cognitive model the concurrent, real-time reasoning capability that it needs.

For example, the schema for a table would contain various images of tables and words that describe tables, and would connect them to facts about tables, such as their typical size, and to actions that typically are performed with tables, such as sitting and eating at one (which is another schema). The table schema would be connected to the schemas for chairs, dining rooms, conference rooms and many other relevant concepts.

ADAPT uses schemas by instantiating them to create schema instances that are connected to each other to form a network. Over time, ADAPT maintains this network, monitoring its behavior and adding schema instances and removing old ones as its goals and the environment change.

RS has been used successfully in industrial settings for applications such as kit assembly (Lyons & Hendriks, 1995). RS thus provides a mature language for robust, real-time distributed control. A video is available on the Pace Lab's website at http://csis.pace.edu/robotlab/clips/puma.avi that shows RS controlling a robot arm and camera to track and grasp an erratically moving object.

RS provides a powerful representational language for the system's dynamics, language and percepts; however, RS does not provide a mechanism for synthesizing the dynamics. Furthermore, RS lacks cognitive plausibility, and in particular lacks a learning method.

We have implemented RS in Soar (Laird, Newell & Rosenbloom, 1987) to take advantage of Soar's cognitively plausible problem-solving and learning mechanisms, as well as the existing work on other aspects of cognition in Soar, including natural language (Lonsdale, 1997)(Rytting & Lonsdale, 2001), concept learning (Miller, 1993)(Wray & Chong, 2003), and emotion (Marsella, Gratch & Rickel, 2003).

Soar uses *universal subgoaling* to organize its problem solving process into a hierarchy of subgoals, and uses *chunking* to speed and generalize that process. ADAPT uses Soar's subgoaling ability to create hierarchies of schemas, and uses Soar's chunking ability to speed the creation of these hierarchies. One of the principal goals of the ADAPT project is to demonstrate how this form of learning can speed scene comprehension.

1.2. Comprehension through Generation

ADAPT's approach to comprehension and planning is based on the work of Green & Lehman (2002), who used Soar to model discourse planning in natural language. Their goal was a computational model of discourse, in which Soar would interact with a human using English. In their task, Soar had to do two things: construct an explanation of an incoming utterance and generate an appropriate response. Their approach was to use *explanation based on generation*, in which Soar would construct an explanation for the incoming utterance by attempting to generate a similar utterance on its own. Soar would search among possible combinations of goals and expectations to try to generate this utterance, and when it succeeded in matching the utterance it assumed the human's goals and expectations were those it had found. This gave Soar an understanding of the goals of the person it was conversing with, so that Soar could choose appropriate goals for generating a response. Soar also formed a chunk for this search, so that a similar utterance in the future would be understood in one step.

For example, if the discourse were about cooking and the incoming utterance were, "Turn the flame down." then Soar would try to generate that same sentence by hypothesizing various combinations of goals and knowledge. One combination that works is the knowledge that a high flame cooks food faster and the goal is to cook it slowly. If Soar succeeded in finding this combination, then it would assume the speaker has this same combination of knowledge and goal. This approach requires Soar to have knowledge about cooking, so that it can search that knowledge.

ADAPT uses this approach as the basis for its planner. To understand its environment, ADAPT attempts to recreate the state of the environment in its world simulator. This includes comprehending its own actions, as well as the actions of other agents and the dynamics of the world. This approach to comprehension and planning is slow, as it involves searching for explanations for actions that occur. Soar's learning mechanism is the key to the success of this approach, as it compiles generalized explanations from experience to speed up future response. The measurement and evaluation of the effectiveness of this type of learning is one of the main goals of this research project.

This approach requires the robot to be able to simulate not only the objects in its environment but also its dynamics, including the goals of other agents in the environment. For this reason we use a full-featured gaming platform as the world model.

2. USING A MODEL FOR VISUAL PREDICTION

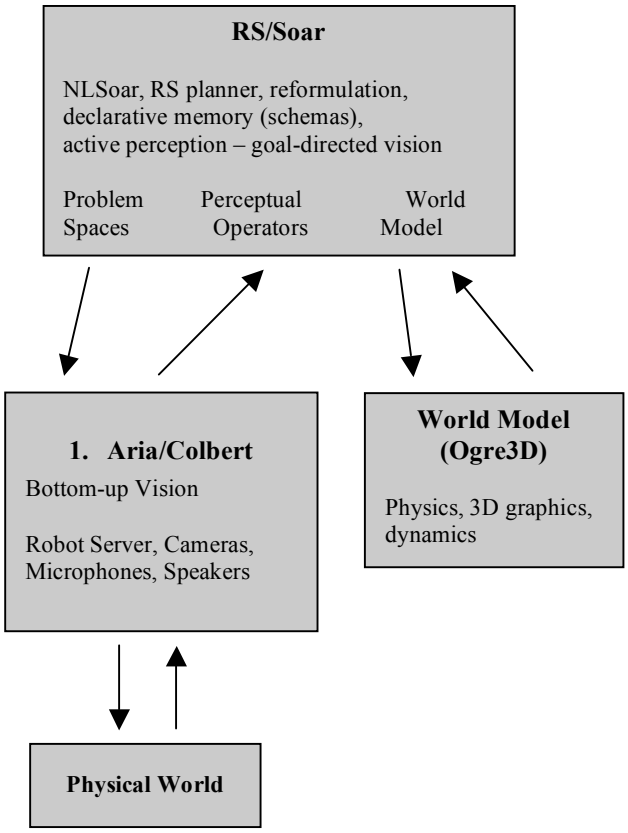
The distinctive feature of ADAPT's vision system is how it uses a model to predict what it will see. ADAPT's World Model (WM) consists of two components, a linguistic one and a multimedia one. The linguistic component resides in Soar and is composed of Soar working memory elements. The multimedia component consists of a powerful multimedia 3D gaming platform, which is currently Ogre3D (http://www.ogre3d.org). This open source platform possesses sophisticated graphics, and uses a physics plugin based on ODE (Open Dynamics Engine) to model the dynamics of the world. Ogre is capable of modeling a wide variety of dynamic environments, and also of modeling other agents moving and acting in real-time in those environments. This world model can be used to predict the near-term future of the environment for planning and comprehension.

The robot uses this world model in a fundamentally different way than is typical in other work. The basic idea is that ADAPT uses the 3D component to build a virtual replica of its environment (and itself), and uses Soar to annotate and explain the 3D model. ADAPT's world model is connected only to RS/Soar, not directly to the external world, so that sensory data must be processed by RS/Soar and modeled in Ogre. This reflects our belief that perception is an active process; perception is *not* done by peripheral processes that interpret the sensory data and create entities and relationships in the world model. Instead, ADAPT makes explicit decisions about how to process the sensory data, e.g. what filter to apply to vision data, and explicitly models the results in Ogre. In this way, *perception is a problem-solving process*, capable of drawing on all the knowledge of the system and permitting Soar's chunking capability to be applied to perception.

For example, if the robot sees a table in front of it, with a chair pushed under the table, ADAPT will construct a virtual table and chair in Ogre3D in approximately the same relation to the virtual robot (its copy of itself in the virtual world). In addition, ADAPT will create Soar working memory elements for the chair and table (with all their perceived attributes) and connect them to the Ogre3D objects for the table and chair. Also, ADAPT will create Soar working memory elements that describe the scene, e.g. stating that the chair is pushed under the table. Then, if Soar proposes and selects an operator to move towards the table, the virtual robot is moved in approximately the same direction and at about the same speed as the real robot.

Ogre embodies the graphical and dynamic aspects of ADAPT's world model. RS/Soar contains the symbolic part of the world model, which consists of "annotations" describing the entities, relationships and activities in Ogre. For example, when RS/Soar recognizes a person sitting in a chair, it will construct virtual copies of the chair and the person in Ogre and create symbolic structures in Soar's working memory that point to them, as well as a symbol structure for the relationship of sitting. Ogre serves as the model that interprets the symbols in RS/Soar's working memory. This permits Soar to reason about and talk about its mental model. The relationships between the RS/Soar and Ogre parts of the world model are updated automatically each cycle of the system.

The overall structure of ADAPT is shown in the next figure.



So the gaming platform provides ADAPT with a means of "understanding" the world by creating and manipulating a virtual copy of it. The gaming platform's objects are connected with Soar working memory elements, which permits Soar to reason about and talk about its model of the world.

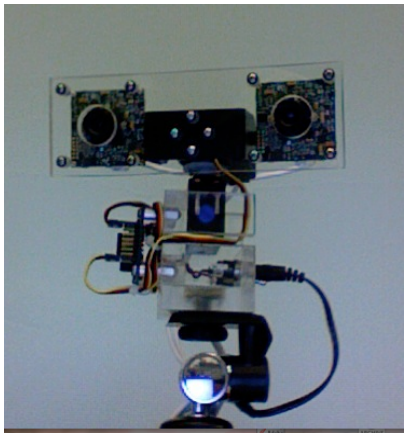
http://csis.pace.edu/robotlab/movies.html contains video clips showing an ADAPT robot performing a simple maneuver and creating a virtual model of its environment.

This world model also permits ADAPT to visualize and explore possibilities by creating structures in the world model that do not correspond to sensory data, i.e. it can "imagine" possibilities. Using the physics plugin, the robot can examine how the world might evolve in the near future. ADAPT's planner, currently under construction, will use this visualization capability in the same way that NL-Soar understands the world using "comprehension through generation" (Green & Lehman, 2002). In this approach to comprehension, NL-Soar understands an utterance by searching for ways to generate it. ADAPT will comprehend its environment by searching among ways to recreate it. Comprehension through generation is a slow process, and depends upon the speedup of chunking to be practical. Evaluating the success of this approach to modeling the world is another of the central goals of the project.

3. THE STRUCTURE AND OPERATION OF THE VISION SYSTEM

The Vision System is currently operational on two Pioneer robots, a P1 and a P2. The P2 is equipped with two firewire cameras (Canon VC-C4) and two framegrabbers. The cameras are mounted on a Directed Perceptions pan-tilt unit, as shown in the images below at right. The P1 is equipped with two Firefly firewire cameras on a custom acrylic

pan-tilt mount, as shown below at left and center. Both robots are controlled by onboard linux computers connected by wireless ethernet to standalone PCs.



ADAPT's vision system software consists of two main components, a bottom-up component that is always on, and a top-down goal-directed component controlled by RS/Soar.

The bottom-up component component is designed to be simple and fast. It does this by not producing much detail. The idea is for it to produce a basic stereo disparity map, a coarse-grained image flow, and color segmentation in real time. It runs on the robot's onboard computer using Intel's open vision library, and segments the visual data from the robot's two framegrabbers. These "blobs" are transmitted together with stereo disparity data and optical flow to the offboard PC that is running RS/Soar, where it is placed into working memory. This component is always on, and its output is task-independent.

The top-down component consists of more expensive image processing functions. These functions are executed only when their RS schemas are selected for application. For example, the ERVision software performs object recognition. This is controlled by a schema, so that RS/Soar needs to propose this operator, i.e. it must want to recognize a portion of the image. When activated, this schema runs the ERVision software on a portion of the image to recognize what's in it. The goal is to reduce the cost by recognizing an object just once, then modeling it in the WM so that it doesn't have to be recognized again. After its initial recognition, an object is "checked" each cycle by the bottom-up VS component, which just monitors whether its blobs are still where they are expected to be.

Object recognition is not the only image processing function that can be controlled by schemas. Other examples include more detailed image flow analysis and application of specific visual filters to the image data.

So the bottom-up component is always on, and produces a steady stream of basic, coarse-grained vision data. The top-down component is implemented in Soar, using RS schemas to control a variety of image processing algorithms. This permits Soar's learning method to be applied to learning perception.

Calling these functions in a task-dependent and goal-dependent manner by RS/Soar operators greatly reduces their frequency of application and speeds the operation of the vision system significantly.

These two components are not connected to each other; instead, the output of the bottom-up component is used by RS/Soar to determine when to call the top-down operations. RS/Soar does this by comparing the bottom-up output to the visual data predicted by the world model.

The world model can display the view that the virtual copy of the robot "sees" in the virtual environment. The output of this graphics "camera" in Ogre is segmented and placed into RS/Soar's working memory, together with distance information and motion information from the mental model. Soar operators test for significant differences between the expected view and the actual view, e.g. the appearance of a large new blob or a large change in optical flow. Any significant difference causes an operator to be proposed to attend to this difference.

The overall functioning of the visual system is:

The bottom-up component places segments, stereo info and motion info into Soar,

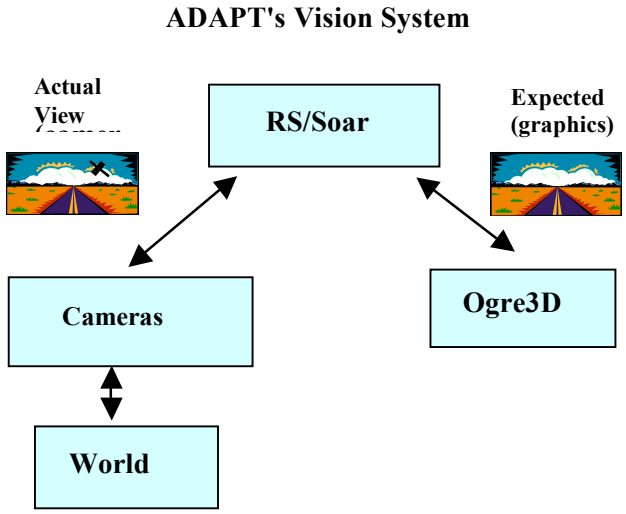
Soar fires rules that notice a new blob it has never detected before or notices new depth or motion info,

Soar fires rules that tell the robot to turn its cameras to bring the noticed info into the center of the visual field,

Soar fires rules that process the new info in the appropriate way, e.g. for a new blob it grabs a small rectangle containing the new info and sends it to the object recognition software, which places its result into Soar.

For example, if a new blob appears, an operator will be proposed to look at this blob and try to recognize it. If this operator is selected (if there is nothing more important to do at the moment) then RS/Soar will instruct the robot to turn its cameras towards this blob, and then call its recognition software to process the rectangular region of the visual field that contains the blob. The recognition software currently used is ERVision 3.0.

Once the object is recognized, a virtual copy is created in Ogre. The object does not need to be recognized again; as long as the blobs from the object approximately match the expected blobs from Ogre, ADAPT assumes it is the same object. Recognition becomes an explicitly goal-directed process that is much cheaper than continually recognizing everything in the environment. The frequency with which these expensive operations are called is reduced, and they are called on small regions in the visual field rather than on the whole visual field.



The vision system compares actual data from the cameras with predicted data from the world model. Unexpected differences become perceptual goals.

Thus, ADAPT's vision system spends most of its time verifying hypotheses about its environment, instead of creating them. The percentage of its time that it must spend attending to environmental changes depends on the dynamic nature of the environment; in a relatively static environment (or one that the robot knows well from experience) there are very few unexpected visual events to be processed, so visual processing operators occupy very little of the robot's time.

Another advantage of this approach to visual comprehension is that it opens the possibility of learning recognition strategies.

SUMMARY AND CURRENT STATUS

We have sketched the overall design of a new approach to visual comprehension that is part of a robotic cognitive architecture. The goal of ADAPT is to integrate visual comprehension, use of natural language, problem solving and learning in an architecture that is embedded in the physical world. The design of the visual system reflects this goal by using a sophisticated world model to predict what the robot will see, thus reducing the cost of visual comprehension.

The implementation of the basic components of ADAPT is complete, including the vision system. Integration of these components is currently under way. The vision system has been integrated with RS/Soar and the mental model and successfully models the robot's environment in real time.

Further information on this work, including video clips showing the robot moving under the control of schemas and the use of the world model, can be downloaded from the website for the Pace University Robotics Lab: http://csis.pace.edu/robotlab

REFERENCES

Michael A. Arbib, 2003. "Schema theory", in Michael A. Arbib, editor, The Handbook of Brain Theory and Neural Networks, 2nd Edition, pages 993–998. MIT Press.

Arbib, M. A., 1992. Schema Theory, in S. C. Shapiro (Ed.), Encyclopedia of Artificial Intelligence (2nd edition), Wiley-Interscience.

Benjamin, D. Paul, Damian Lyons and Deryle Lonsdale, 2006, "Embodying a Cognitive Architecture in a Mobile Robot", Proceedings of the SPIE Conference on Intelligent Robots and Computer Vision, Boston, October, 2006.

Benjamin, D. Paul, Damian Lyons and Deryle Lonsdale, 2004a. "Designing a Robot Cognitive Architecture with Concurrency and Active Perception", Proceedings of the AAAI Fall Symposium on the Intersection of Cognitive Science and Robotics, Washington, D.C., October.

Benjamin, D. Paul, Damian Lyons and Deryle Lonsdale, 2004b. "ADAPT: A Cognitive Architecture for Robotics", Proceedings of the International Conference on Cognitive Modeling, (ICCM-2004), Pittsburgh, Pa., July.

Benjamin, D. Paul, 2000. "On the Emergence of Intelligent Global Behaviors from Simple Local Actions", Journal of Systems Science, special issue: Emergent Properties of Complex Systems, Vol. 31, No. 7, 861-872.

Benjamin, D. Paul, 1997. "A Decomposition Approach to Solving Distributed Constraint Satisfaction Problems", Proceedings of the IEEE Seventh Annual Dual-use Technologies & Applications Conference, IEEE Computer Society Press.

Benjamin, D. Paul, 1994. "Formulating Patterns in Problem Solving", Annals of Mathematics and AI, 10, pp.1-23.

Benjamin, D. Paul, 1992. "Reformulating Path Planning Problems by Task-preserving Abstraction", Journal of Robotics and Autonomous Systems, 9, pp.1-9.

Green, N. and Lehman, J.F., 2002. "An Integrated Discourse Recipe-Based Model for Task-Oriented Dialogue." Discourse Processes, 33(2).

Laird, J.E., Newell, A. and Rosenbloom, P.S., 1987. "Soar: An Architecture for General Intelligence", Artificial Intelligence 33, pp.1-64.

Lonsdale, Deryle, "Modeling cognition in SI: Methodological issues. International Journal of Research and Practice in Interpreting", **2(1/2)**: 91–117, 1997.

Lyons, D. and Hendriks, T., 1995. "Planning as Incremental Adaptation of a Reactive System", Robotics and Autonomous Systems, Vol. 14, No. 4.

Lyons, D.M., 1993. "Representing and Analysing Action Plans as Networks of Concurrent Processes", IEEE Transactions on Robotics and Automation, June.

Lyons, D.M. and Arbib, M.A., 1989. "A Formal Model of Computation for Sensory-based Robotics", IEEE Transactions on Robotics and Automation 5(3), June.

Marsella, Stacy, Jonathan Gratch and Jeff Rickel, "Expressive Behaviors for Virtual Worlds," Life-like Characters Tools, Affective Functions and Applications, Helmut Prendinger and Mitsuru Ishizuka (Eds.), Springer Cognitive Tech. Series, 2003.

Miller, C. S. (1993), “Modeling Concept Acquisition in the Context of a Unified Theory of Cognition”, EECS, Ann Arbor, University of Michigan.

Roy, Deb and Alex Pentland, 2002. "Learning words from sights and sounds: A computational model", Cognitive Science, 26(1):113–146.

Rytting, C. Anton, and Deryle Lonsdale. IntegratingWordNet with NL-Soar. In WordNet and other lexical resources: Applications, extensions, and customizations, pages 162–164. North American Association for Computational Linguistics, 2001.

Steenstrup, Martha, Michael A. Arbib, Ernest G. Manes, 1983. Port Automata and the Algebra of Concurrent Processes. JCSS 27(1): 29-50.

Wray, R. E., and Chong, R. S., Explorations of quantitative category learning with Symbolic Concept Acquisition. 5th International Conference on Cognitive Modeling (ICCM), Bamberg, Germany, 2003.