Shakey: From Conception to History

Benjamin Kuipers, Edward A. Feigenbaum, Peter E. Hart, Nils J. Nilsson

■ *Shakey the Robot, conceived 50 years ago, was a seminal contribution to AI. Shakey perceived its world, planned how to achieve a goal, and acted to carry out that plan. This was revolutionary. At the 29th AAAI Conference on Artificial Intelligence, attendees gathered to celebrate Shakey and to gain insights into how the AI revolution moves ahead. The celebration included a panel that was chaired by Benjamin Kuipers and featured AI pioneers Ed Feigenbaum, Peter Hart, and Nils Nilsson. This article includes written versions of the contributions of those panelists. —ed.*

An Introduction

Benjamin Kuipers

At AAAI-15 (25–29 January 2015) in Austin, Texas, we met to celebrate the impact of the Shakey project, which took place from 1966 to 1972 at the Stanford Research Institute (now SRI International) in Menlo Park.

We researchers in artificial intelligence during this time in history have the privilege of working on some of the most fundamental and exciting scientific and engineering problems of all time: What is a mind? How can a physical object have a mind?

Some of the work going on today will appear in future textbooks, even centuries from now. We gain insights into our own struggles in the field today by learning about the historical struggles of great scientists of the past about whom we read in today's textbooks. The textbooks tempt us to think that they moved surely and confidently from questions to answers. In reality, they were frequently as confused then as we are now, by the mysterious phenomena they were trying to understand. When we read their history, we know the

  ISSN 0738-4602

answers they were seeking, and we can learn from the blind alleys they spent time in, and the insights that led them to the right paths.

Artificial intelligence marks its birth at the 1956 Dartmouth Conference. There have been many important milestones along the way. The important milestone we will celebrate today is the Shakey project, which created a physical robot that could perceive its environment and the objects within it. Shakey could make a plan to achieve a goal state. And it could carry out that plan with physical actions in the continuous world. The Shakey project laid a foundation for decades of subsequent research. We are here to celebrate and understand that project.

The centerpiece of the Shakey celebration was a panel presentation at AAAI-15, designed to give the audience an understanding and appreciation of the process of the research in the Shakey project, and of the long-term impact of that work on the larger field of AI. The goal was to have three speakers address (1) the state of the art in AI before the Shakey project (Ed Feigenbaum); (2) the progress of the Shakey project itself (Peter Hart); and (3) the impact of the Shakey project on the future of AI (Nils Nilsson).

Celebrating Shakey and Its Builders

Edward A. Feigenbaum

The history of science is a source of knowledge of the complex search for solutions to difficult problems. Not only is this history endlessly intriguing and awe-inspiring; but also it should be of particular interest to AI scientists because this kind of complex problem solving and discovery is at the heart of many of our theories of mental activity.

Life is lived in the moment. Everything else is memory and stories. The word *history* itself contains the word *story*. This talk is constructed as several stories of the Shakey project situated in its time, and among other landmark AI projects.

I have been lucky enough to have lived and worked through the entire 60 years of AI, from early 1956, months before the famous "founding" Dartmouth Conference, to today's AAAI-2015. My stories are drawn from those 60 years of memories, helped, but only a little, by the best memory assistant ever, the web.

My role today is to set the historical context in which the Shakey project was born, lived a remarkable but short life, and was terminated. Shakey research set the stage for decades of important experimental work in AI and robotics, and in other AI applications that will be mentioned later by Nils Nilsson.

I phoned several well-known robotics scientists to ask about the grandchildren of Shakey. All of them said the robots they developed were grandchildren of Shakey.

As shown in an original Shakey video, we remember Shakey as slowly and laboriously computing models of its environment; planning; moving and navigating its way around obstacles toward a goal on the far side of one large room.

Fast forward to some recent news about grandchildren of Shakey, from Manuela Veloso at Carnege Mellon University (CMU):

> I am very pleased to tell you that today, on November 18, 2014, the CoBot robots (3 of them) have jointly autonomously navigated for 1,000 km in our multi-floor SCS buildings at Carnegie Mellon University!

A great-grandchild of Shakey, Stanford's self-driving car Stanley, the car that drove itself across the Mojave Desert, is in the Smithsonian National Air and Space Museum in Washington, DC. Other cars like Stanley, built at Google, have driven more than 700,000 miles, navigating the San Francisco Bay Area and other roads, according to the *San Jose Mercury News* of November 12, 2014. (Consider this: Shakey's traversal, integrated over the whole life of the experiment, probably never made it to one kilometer).

Shakey's grandchildren on Mars are still having a productive long life — 11 years into a planned 90-day visit, semiautonomously assisting planetary scientists.

I would now like to tell you personal stories that together made the importance of Shakey research vivid to me.

First Story

In 1993, a major Japanese corporation asked me to do an evaluation of the quality of a robotics project that its research lab been working on for several years. After signing a nondisclosure agreement, I was shown a robot that was "humanoid," but very big (scary, actually). Tethered to a power source, its motion was fluid, a marvel of modern electromechanical engineering.

Though heavy, it could walk reliably without falling, and it could even climb a flight of stairs. But this creature had no Mind. It did no symbolic processing, no problem solving. It did not have goal-directed behavior.

There was more than enough space inside for a PC-sized computer and there was plenty of power. What this project lacked were scientists and engineers trained in AI, or even trained in software systems. There were no young Nils Nilssons, no young Peter Harts, no young Bert Raphaels or Richard Fikes — and of course no visionary like Charles Rosen to integrate AI with electromechanical engineering.

And this was 1993, twenty years after the end of the Shakey project! It can be perilous to ignore scientific history.

Second Story

The Computer History Museum in Mountain View, California, is the world's premier museum for the history of computers and information technology and is

recognized for its interpretation of that history. In January 2011, the museum opened its permanent exhibition, called Revolution. Here is a quote from the museum's press release:

> Ten years in the making, Revolution is the product of the Museum's professional staff collaborating with designers, content producers and more than 200 experts, pioneers and historians around the world.

Revolution showcases 20 different areas of computers, computer science, semiconductors, and communications, from early history to futuristic visions. Among those 20 areas is one called AI and Robotics.

For each area, the museum staff has chosen one historical artifact to be the icon exhibit for the area. For AI and Robotics, the icon is Shakey the Robot, beautifully exhibited.

The museum could have chosen any one of a dozen or more landmark AI artifacts. It could have chosen AI's first heuristic problem-solving program (the Logic Theorist of Newell, Shaw, and Simon); or a speech-understanding program from Reddy; or one of the early expert systems from our Stanford group; or Deep Blue, the AI system that beat the world's chess champion. It could have … but in the end the museum chose Shakey.

So let's put the first story and the second story together to make a:

Third Story

SRI's Shakey work was a decade or two ahead of its time in demonstrating the power of integrating AI with robotics. Remarkably, even today, when robotics is being taught to high school students, and computing and sensors cost almost nothing, most robots in labs and companies do not have the AI capabilities that Shakey had in the 1970s.

Historians of the field have given Shakey deserved recognition, but the field of AI had not. It took a while for an AAAI national program committee to recognize this and make room for this celebration. I want to thank the AAAI-15 program committee, and hope that this will be a model for bringing forth other important parts of AI's history.

Fourth Story

The Shakey Project was done from 1966 to 1972. What was AI and computer technology like before and during that period?

There is a generation of younger researchers that have no idea how few were the powerful ideas of the first decade of AI (1956 to 1966) to build upon for new AI systems. Nor can that younger generation envision the lack of power of the computers that we had upon which to build these systems.

But there was no lack of enthusiasm, and excitement; no lack of interaction, because almost everyone in the field knew almost everyone; and we all read each other's papers, tech reports, and books. That's what it's like, when a field is small and emerging.

The AI science had a workable set of ideas about how to use heuristic search to solve problems. But proving things about heuristic search had to wait until later (the Shakey group's A*). Some powerful successful experiments had been done: the Logic Theorist; Gelernter's Geometry Theorem Proving program; Slagle's calculus problem solving programs are examples. These were all on the "cognitive" side of AI work. On this side, much discussion and energy was focused on generality in problem solving: Newell and Simon with means-ends analysis; McCarthy and other "logicists" with theorem proving.

On the "perceptual" side of AI work, a similar story can be told about research on vision. There were several basic workable techniques involving line finding, curve finding, and putting elements together into logical descriptions of objects. Generality of the techniques was also an issue, as it still is today.

What did we have with which to do this work? Our programming languages were great! List processing was invented at CMU and then made more powerful and beautiful in LISP at the Massachusetts Institute of Technology (MIT). But there was almost no interaction between people and computers. Time-shared interaction did not become available to most researchers in this first decade.

Try to imagine this about computer processing power and memory: I did my thesis work on an IBM 650 computer in the late 1950s: maximum 2500 operations per second; memory was 20,000 digits (what we would now call bytes). Not only your program, but your language interpreter had to fit into this memory. There was no virtual memory.

In 1959, the IBM's large multimillion-dollar transistorized computer was introduced. It ran at 100K FLOPS, and had about 150K bytes of main memory. The largest DEC computer that would have been available in 1966 for the Shakey group to buy was the PDP-6, which operated at 250,000 additions per second with a memory of about 150K bytes.

Compare these numbers with, say, today's Apple MacPro at four gigaops/sec with memory of 16 gigabytes; or even today's smartphones at about 1 gigaop/sec but with memories going up to 128 gigabytes.

Fifth Story

All projects end, even the great ones. The DARPA funding pendulum for support of AI swung away from robotics and toward both knowledge-based systems and the national speech understanding project. As funding shifted, SRI continued to do world-class work in both of these other themes of the 1970s.

Final Story

The Shakey project, as cutting edge work in computer science, inspired young people to do great things. In an email to Eric Horvitz, former president of AAAI,

let me quote from one of these people, a junior in high school at the time. He and a high-school friend traveled to visit the Shakey project in 1971, unannounced, but were welcomed by the Shakey team.

> I was inspired by the Shakey video from SRI. I actually went down and visited when I was a junior in high school and they showed me the lab.
>
> Shakey was pretty cool — vision, modeling, planning. It decided to move things around so it could go up a ramp.
>
> Paul — do you remember how we got this video?

The "Paul" is … Paul Allen; and the author of the quote is Bill Gates.

Making Shakey

Peter E. Hart

The proposal that launched the Shakey project was submitted by the Artificial Intelligence Center of Stanford Research Institute (now SRI International) in January, 1965. SRI proposed to develop "intelligence automata" for "reconnaissance applications." But the research motivation — and this was the inspiration of Charles A. Rosen, the driving force behind the proposal — was to develop an experimental test bed for integrating all the subfields of artificial intelligence as then understood. SRI wanted to integrate in one system representation and reasoning, planning, machine learning, computer vision, natural language understanding, even speech understanding, for the first time.

Readers interested in technical details of Shakey's development will find an excellent summary[1] in an SRI report. A 25-minute video, made by the Shakey team at the time, is available.[2]

The design of the "automaton," as it was initially called (perhaps out of a justifiable concern that "robot" sounded like science fiction, which it was before Shakey), was governed by two ground rules: First, in order to keep it mechanically as simple as possible, no arm was installed. And second, to avoid issues of miniaturization, the design evolved as an electronics rack on wheels with a sensor assembly mounted on top.

The project team was well aware of Shakey's limited mechanical and sensory capabilities, and designed a correspondingly simple experimental environment consisting of half a dozen rooms populated with large, geometric blocks. The blocks were painted so that edges were visible to the low-resolution TV camera, while still being sufficiently reflective for our homemade laser rangefinder to work. We also used dark baseboards, again for visibility, and exploited them to update the position error that accumulated in the dead reckoning process that relied on Shakey's stepping motors.

Our first computer was an SDS 940, an early commercial time-shared mainframe (whose main memory was smaller than the L2 cache of most laptops). In 1970 we upgraded to a more powerful DEC PDP-10. Shakey talked to the PDP-10 through a communications processor, and the system was one of the handful of nodes that constituted the birth of the ARPANET. Around this time we embarked on a complete rewrite of much of Shakey's software, while making only minor upgrades to the robot hardware. In the next section we describe this version 2 of Shakey.

Figure 1. Charles A. Rosen and the "Automaton."

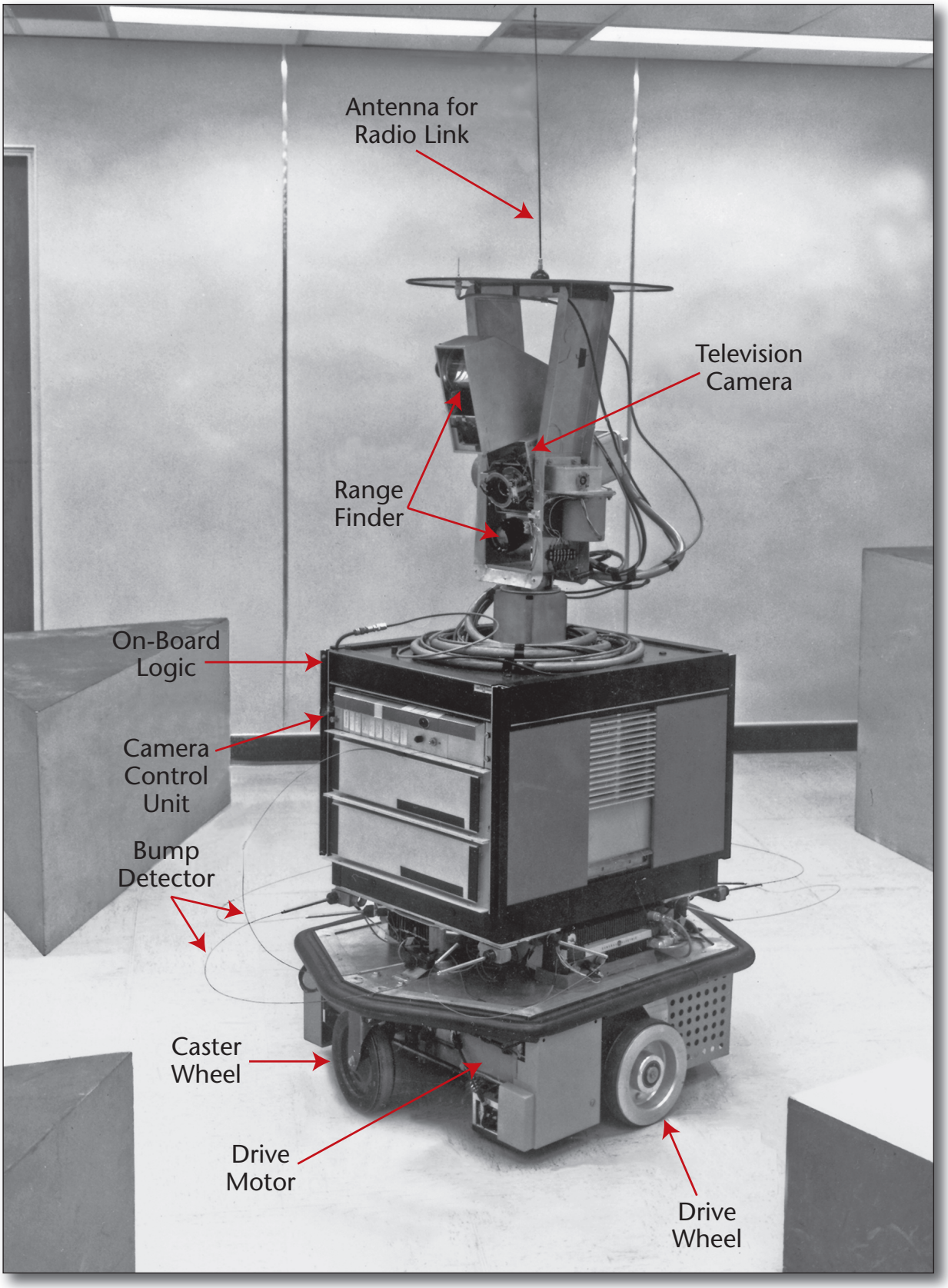

Figure 2. Shakey with Components Labeled.

Shakey's Control Software

There were two big ideas behind version 2. The first idea was to represent Shakey's world by statements in the first-order predicate calculus, augmenting a form of grid model that was a key component of the first version (figure 4).

The second idea was to structure Shakey's control software as a series of layers, the first time this design was used to control a robot (figure 5). In the following we briefly describe each layer, beginning with the low-level actions.

Low-Level Actions

Low-level actions like ROLL and PAN talked directly to Shakey's hardware (figure 6). Also in this layer are actions like PANTO, which rotates the "head" to a specified orientation.

Intermediate-Level Actions: Markov tables

Above this level are intermediate-level actions like GOTHRUDOOR. These actions are put in their own layer because all of them are represented as Markov tables (figure 7).

One interprets a Markov table by scanning down the left column until the first true condition is reached, executing the corresponding action, and then looping back to the top. Accordingly, Markov tables have an inherent perseverance: they keep trying to do something useful. (This account is slightly simplified, but the looping behavior is fundamental and, as we'll see, an important feature of these tables.)

If these intermediate-level actions were the end of the software story, Shakey would be very limited in what it could achieve. It would only be able to achieve goals that require just a single preprogrammed action. To do more, Shakey has to be able to compose a sequence of actions into a plan. That's the job of STRIPS, the Stanford Research Institute Problem Solver, which constitutes the next higher software level.

STRIPS, the Stanford Research Institute Problem Solver

STRIPS came about by combining two big ideas of the day. The first was the planning strategy called means-ends analysis, as exemplified by the General Problem Solver program of Newell and Simon.

The second big idea was theorem proving in the predicate calculus and its application to question answering systems, as exemplified by the work of Cordell Green. Richard Fikes and Nils Nilsson combined these ideas to create STRIPS (Fikes and Nilsson 1971), which applied means-ends analysis to predicate calculus representations (figure 8).

PLANEX, the Plan Execution Executive

Shortly after designing STRIPS, the SRI team found a way to generalize a STRIPS plan by replacing constants in the plan with variables. They also invented a data structure called a triangle table that represents the internal dependencies of a generalized plan. These constructs formed the basis of PLANEX, the Plan Execution Executive that is the top layer of Shakey's control software (Fikes, Hart, and Nilsson 1972).

Figure 3. Triple Exposure of Shakey Moving Among Boxes.

Using this software machinery, PLANEX could monitor the real-world execution of a plan. It could detect if something had gone wrong, and could replan from that point, reusing portions of the existing plan wherever possible. It could even be "opportunistic": If by chance Shakey was closer to achieving its goal than anticipated, it could capitalize on its good fortune.

This error detection and recovery ability was a critically important part of Shakey's control software. A chasm separates planning for a physical robot, that has to execute plans in the real world where things often go wrong, and an "abstract" planner that merely needs to print out a symbolic plan once it is computed. The Plan Execution Executive, together with those persevering Markov tables, was the solution to the problem of achieving robust, real-world plan execution.

Computer Vision

The initial project plan did not call for intensive research in computer vision. Rather, the plan was to integrate existing computer vision techniques into the experimental test bed. But, as it turned out, very little technology was available, so a focused effort in computer vision was started.

One important result of this work was the invention of what could be called the modern form of the Hough transform for finding lines in images (Duda and Hart 1972). This result came about by combining two concepts that on the surface appear unrelated.

The first idea is contained in a patent by Paul Hough, in which he described a transform from points in an image plane to straight lines in a trans-

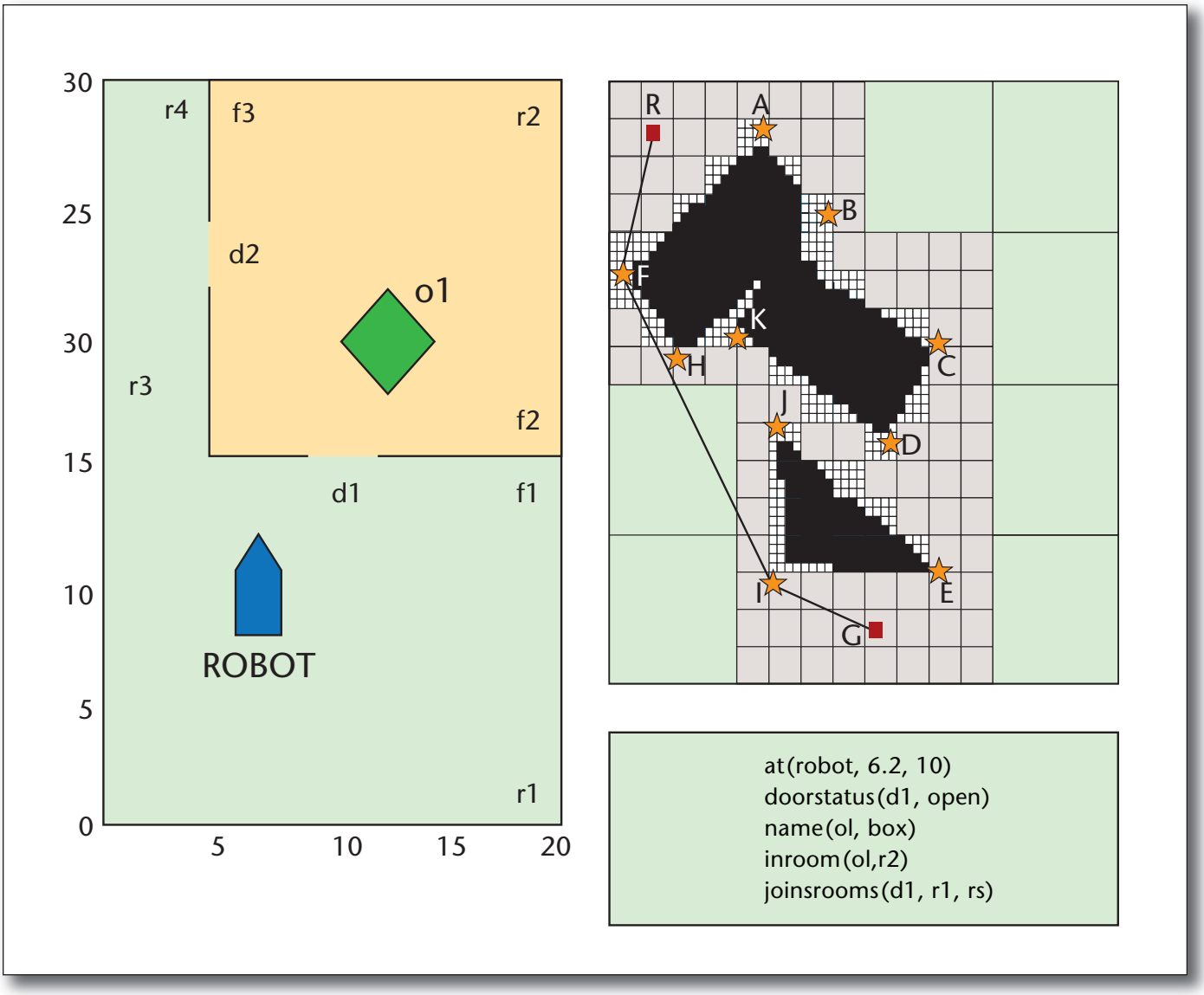

Figure 4. Predicate Calculus Model Fragment with Plan View of World and Grid Model

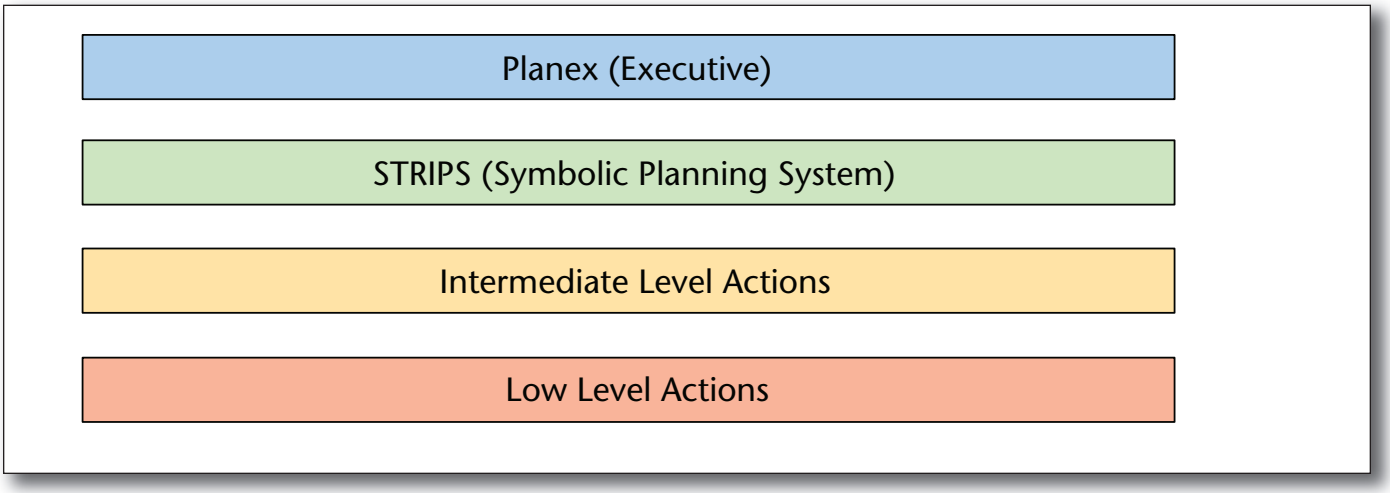

Figure 5. Layered Software Architecture.

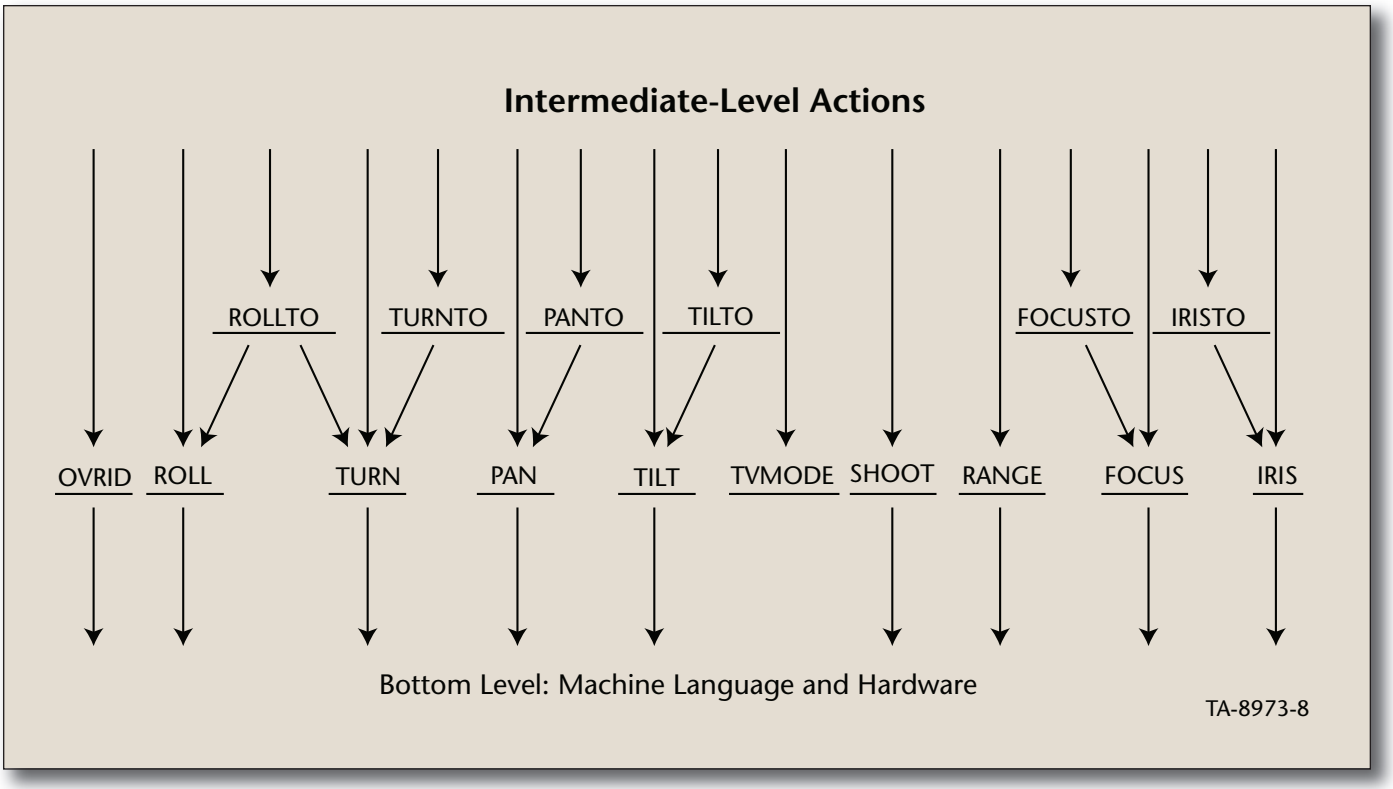

Figure 6. Low-Level Actions.

Condition	Action
infrontof(door) /\eq(s,OPEN)	bumblethru(room1,door,room2)
near(door) /\eq(s,OPEN)	align(room1,door,room2)
near(door) /\eq(s,UNKNOWN)	doorpic(door)
eq(s,CLOSED)	return [fail]
T	navto(nearpt(room1,door))

Figure 7. Markov Table for GOTHRUDOOR.

form space. Intersecting lines in the latter correspond to collinear points in the form. But a problem of infinite slopes arises that makes this transform computationally unwieldy (figure 9).

The second idea comes from an obscure branch of 19th century mathematics called integral geometry. Mathematicians had theoretical reasons for using an angle-radius parameterization of a line, rather than the more familiar slope intercept used by Hough. Peter Hart noticed that by replacing Hough's linear transform with a sinusoidal one, not only is the problem of infinite slopes avoided, but the new transform is invariant to choice of coordinate accesses. Hart and Richard Duda also extended this method to detect analytic curves in images, and this transform has been used ever since.

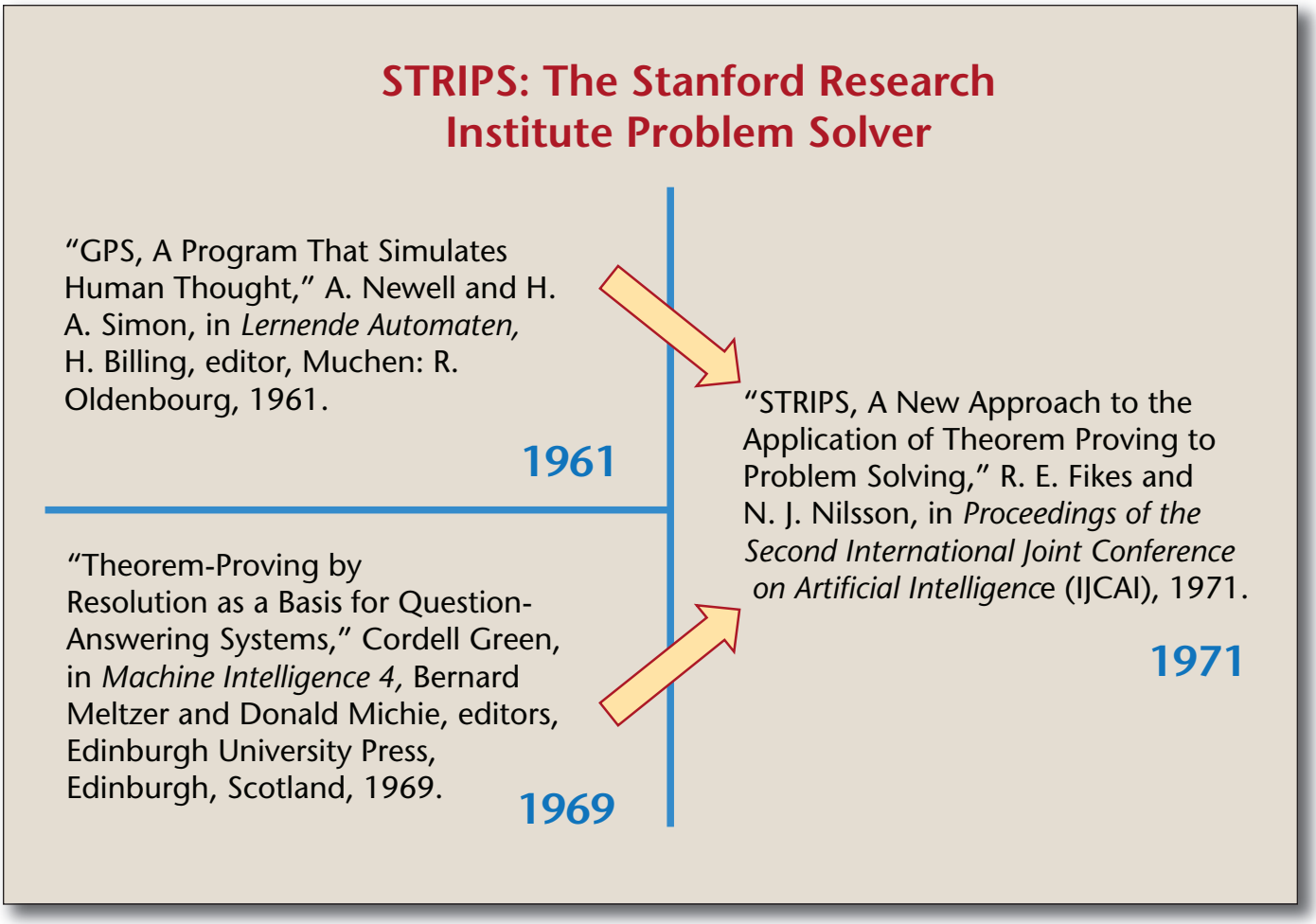

Figure 8. STRIPS.

Navigation and the A* Algorithm

Shakey had to find its way around, so several shortest-path algorithms were developed. One, called A* by its creators, Peter Hart, Nils Nilsson, and Bertram Raphael, had two very desirable properties. It can be rigorously proved that (a) it always finds the shortest path, and (b) that it does so while considering the smallest possible number of alternatives. In non-mathematical shorthand, we can say that it always works and it's computationally efficient.

One would think that such a strong result would be eagerly accepted by any reputable publication, but it's perhaps a sign of those times that just the opposite was the case. The A* manuscript was rejected by the most prestigious journals of the day. Looking at those old reviews, we can speculate that review editors sent the manuscript to mathematicians, because of all those intimidating-looking theorems. But mathematicians were unimpressed because the proofs were limited to graphs with only a finite number of nodes. It seemed to the authors at the time that mathematicians saw no difference between a graph with ten nodes and one with ten trillion nodes, but to computer scientists that difference matters.

The paper (Hart, Nilsson, and Raphael 1968) was finally accepted by the *IEEE Transactions on Systems Science and Cybernetics,* where it eventually got noticed and continues to be referenced more than 45 years after it was published.

The World Back Then

The foregoing gives a glimpse of some (though by no means all!) of the work done by the Shakey project team. To place this work in a broader societal context we can take a brief look at the intellectual and cultural climate of the time.

In 1970, *Life,*[3] a popular magazine of the day, ran a big story about Shakey (figure 10). The author, journalist Brad Darrach, seemed to hyperventilate a bit, with a subtitle, "the fascinating and fearsome reality of a machine with a mind of its own." But while some like Darrach worried that machines might take over the world, there were deep skeptics. Hubert Dreyfus was one, who argued on deep philosophical grounds that AI is in principle impossible. And some-

where between Brad Darrach and Hubert Dreyfus were labor union leaders who worried that robots might some day take manufacturing jobs.

Charlie Rosen was undaunted by the critics, noting that there will always be "naysayers," as he called them, whenever something new is done. The best response is to push onward.

Shakey's Visitors

The Shakey team was generous with its time and welcomed virtually any visitor who was interested in the work. We can get another perspective on the world back then by viewing it through their eyes. Here are some examples:

A school group visited, and a teacher asked what our "real jobs" were. "This robot is your hobby, isn't it?"

A general visited and asked "Can you mount a 36-inch blade on that?"

Arthur C. Clark visited just after the movie *2001* appeared, but was more interested in talking about the *New York Times* review of the movie than about the future of robots.

A young high school student drove all the way from Seattle to Menlo Park, California, to see Shakey. Decades later Bill Gates recalled being impressed.

A US government auditor visited and asked whether SRI had indeed taken delivery of billions of "packets of bits." This question was followed by others regarding the state of those packets, including whether there was any tarnish or corrosion on any of those bits.

The End of the Shakey Project

The Shakey project ended in 1972, not for lack of exciting ideas to pursue, but because the funding climate had changed and the research program became unsupportable. What had been achieved, as viewed from the perspective of 1972?

While there are likely as many views as there were project team members, it seems safe to make a few broad generalizations:

There was an appreciation that many of the individual results — STRIPS, PLANEX, A*, and the new form of the Hough transform are good examples — were solid technical contributions.

Overall, Shakey was a significant achievement, being both the first mobile, intelligent robot, and also being the first system that integrated AI software with physical hardware.

But Shakey's overall capabilities, both mechanical and software, didn't reach the level of the initial aspirations. This would hardly be surprising, given those lofty early goals. Indeed, it would take decades before some were reached, while others remain as research challenges.

Today's perspective is very different from the view in 1972. Shakey has had impacts on both current research and on the everyday lives of all of us that could not have been recognized or anticipated at the time. Those impacts are the subject of the remaining sections of this article.

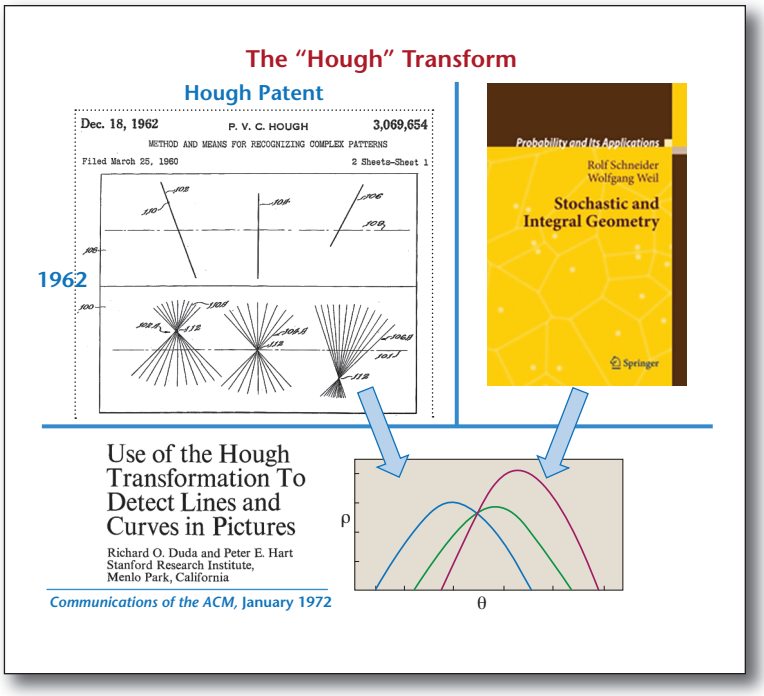

Figure 9. The Hough Transform.

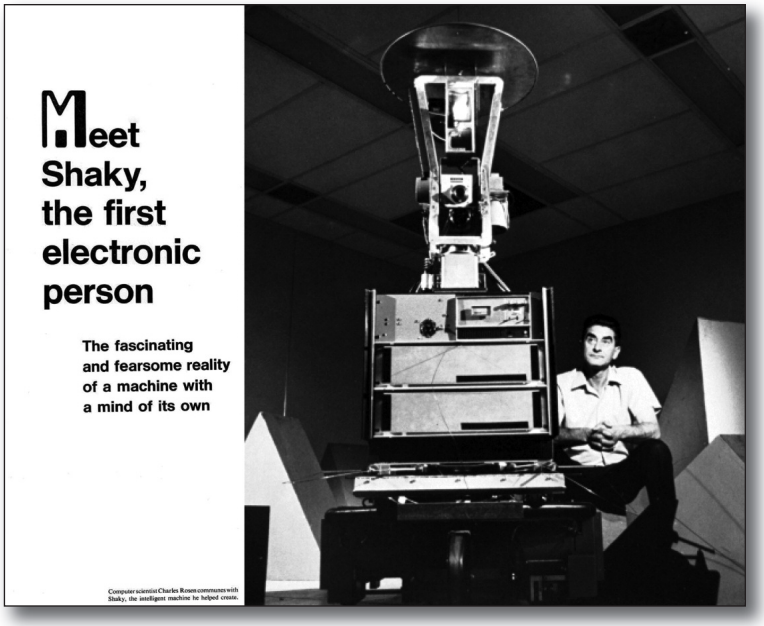

Figure 10. Shakey Story in Life Magazine.
Photo © by Ralph Crane, *The LIFE Picture Collection*.

Shakey's Legacy

Nils J. Nilsson

"Shakey the Robot" was the first system that integrated artificial intelligence programs (most of which were newly developed during the project) with physical hardware. In this part of the panel discussion,

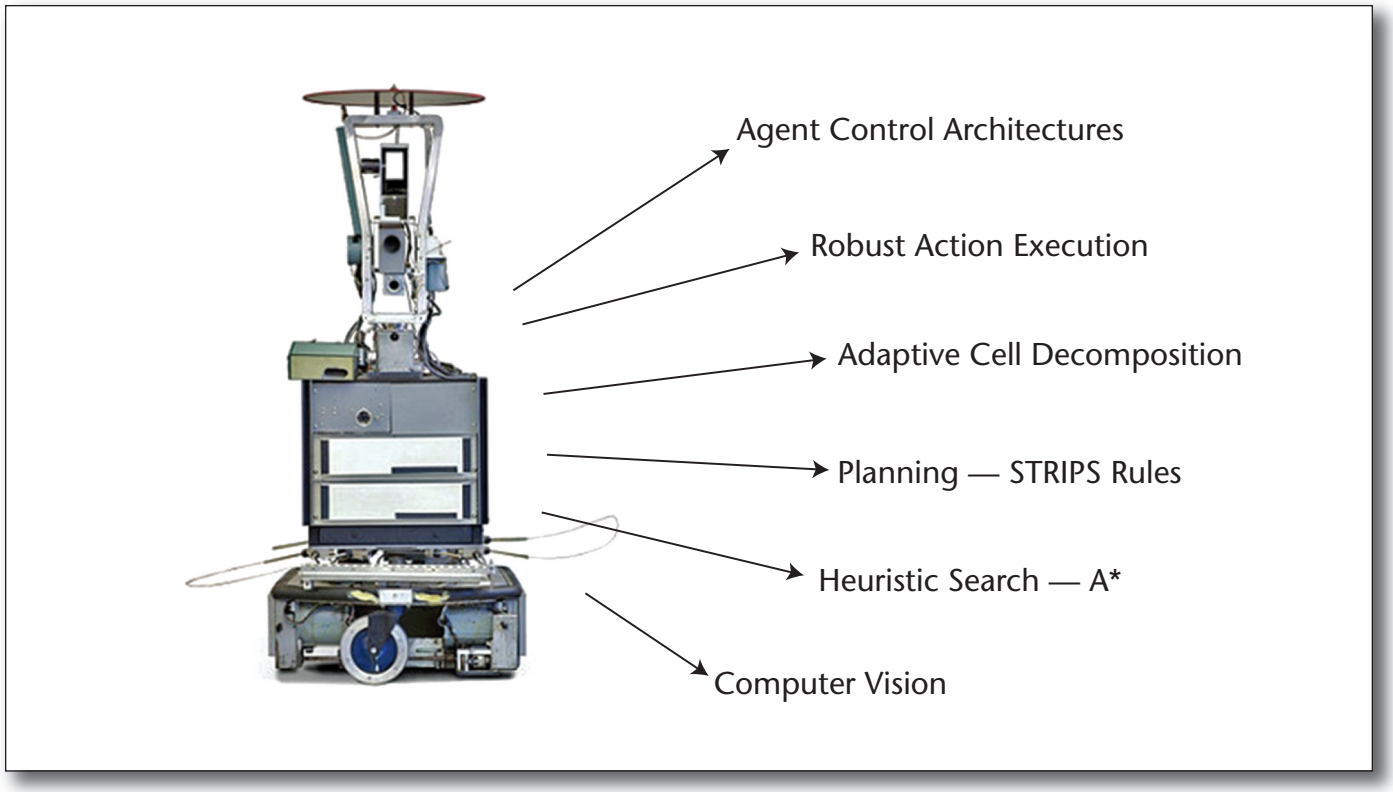

Figure 11. Shakey's Achievements.

Nils Nilsson used the chart (figure 11) to list some major achievements of the project. In greatly elaborated and extended form, descendants of some of this software are still in use today.

Agent Control Architectures

Although hierarchies and layers had previously been used in software systems, Shakey was the first robot to be controlled by a layered architecture. As illustrated earlier, there were four main layers, namely, the PLANEX (executive), STRIPS (symbolic planning system), the intermediate-level actions, and the low-level actions. Layered control architectures have been used in several subsequent robot systems, among them the DS1's "Remote Agent" (RAX), which controlled a space craft (Bernard et al. 1999), and the Monterey Bay Research Institute's (MBARI) autonomous underwater vehicle (McGann et al. 2008). Of course the control architectures of these modern systems, although layered as Shakey's were, are much more complex.

Robust Action Execution

As described earlier, Shakey used two main techniques to guarantee robust action execution. One was Markov tables, which scanned a list of conditions to find the first one that was satisfied by the current situation and then invoked the corresponding intermediate level action. The second was a structure we called a "triangle table," which stored preconditions and actions assembled by the STRIPS planning system. These techniques evoked actions that were both reactive to the current situation and opportunistic in unforeseen situations.

One follow-on to those techniques used by Shakey is the concept of teleo-reactive (T-R) programs developed by Nilsson and his students during the 1990s (Nilsson 1994) (figure 12).

That action associated with the first currently satisfied condition in the list (or tree) is the one that is executed. But execution continues only so long as that condition remains the first one currently satisfied. As soon as it is no longer satisfied, the list is scanned again to find the one now first satisfied, and so on. In the T-R formalism, the actions could themselves be T-R programs. Dozens of papers have been written about T-R programs, one book (Clark and Robinson 2016) is soon to appear, and many robots have been controlled by them.[4]

Another control technique uses structures called "hierarchical state machines." Hierarchical state machines are similar to T-R programs except that actions are represented by nodes and conditions by links. Several robots use them, including the PR2 robots developed by Willow Garage and the SaviOne robot developed by Savioke. There is a Python library, called SMACH,[5] that can be used to build hierarchical state machines.

Adaptive Cell Decomposition

Shakey used a grid model, such as the one shown in figure 13,[6] to map the obstacles in its environment. If a cell is not completely empty or completely full, it is divided into smaller cells and so on until one of these conditions is met for all cells. We believe Shakey's was the first use of an adaptive grid model. Adaptive cell decomposition is still used in robot navigation and in computer-aided design and manufacturing.

STRIPS Rules

STRIPS was the system Shakey used for generating plans to accomplish goals. Figure 14 shows a STRIPS rule for modeling the action of moving a toy block from C to B. The *preconditions* must be satisfied before the action can be applied, and the terms on the *delete list* can no longer be guaranteed to be satisfied after the action is applied, so they are deleted from Shakey's post-action model of the world. The terms on the *add list* are added to the post-action model.

STRIPS rules (or their derivatives) are used in most modern planners. (The STRIPS paper gets over 5000 citations on Google Scholar, and "STRIPS-style planning" gets over 3410 results on Google.) It's the rules that are used, not the STRIPS program itself. STRIPS rules were a practical solution to the "frame problem" — inherent in the use of the "situation calculus," proposed by McCarthy and Hayes (1969) for generating plans.

Hierarchical task networks (HTNs) are much used and powerful planning systems that use STRIPS rules.[7] These systems assemble plan steps into networks of actions, some of which can be executed in parallel and others that must be executed serially. We show an example in figure 15.

SIPE-2,[8] O-Plan (Currie and Tate 1991), and SHOP2[9] are examples of implemented HTNs. Among other important applications of HTNs, SIPE-2 has been used for production planning at an Australian Brewery.

Some video games make use of STRIPS rules and HTNs for planning the actions of nonolayer characters (NPCs).[10]

Heuristic Search and A*

A* is a heuristic search algorithm developed during the Shakey project for efficiently searching a graph of navigation waypoints. It uses an evaluation function to rank the nodes reached during search and continues the search below the best-ranked node. The evaluation function for a node, *n,* is the sum of the cost of the links traversed already on the way to *n* plus an estimate of the cost from *n* to a goal node.

There are lots and lots of descendants and variants of A*. Here is a list of just some of them: D*, Field D*, Theta*, Real-Time A*, Iterative Deepening A*, Life-Long Planning A*, Simplified Memory Bounded A*, and Generalized Adaptive A*. Richard Korf at UCLA

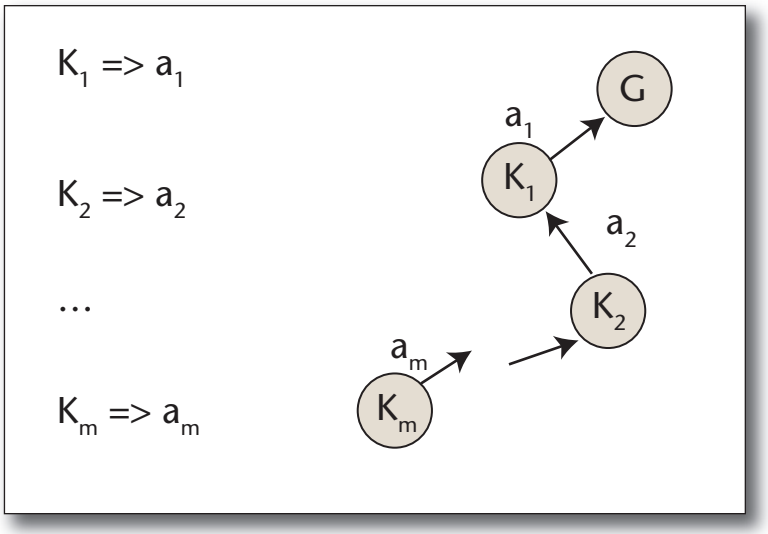

Figure 12. Teleo-Reactive Programs.

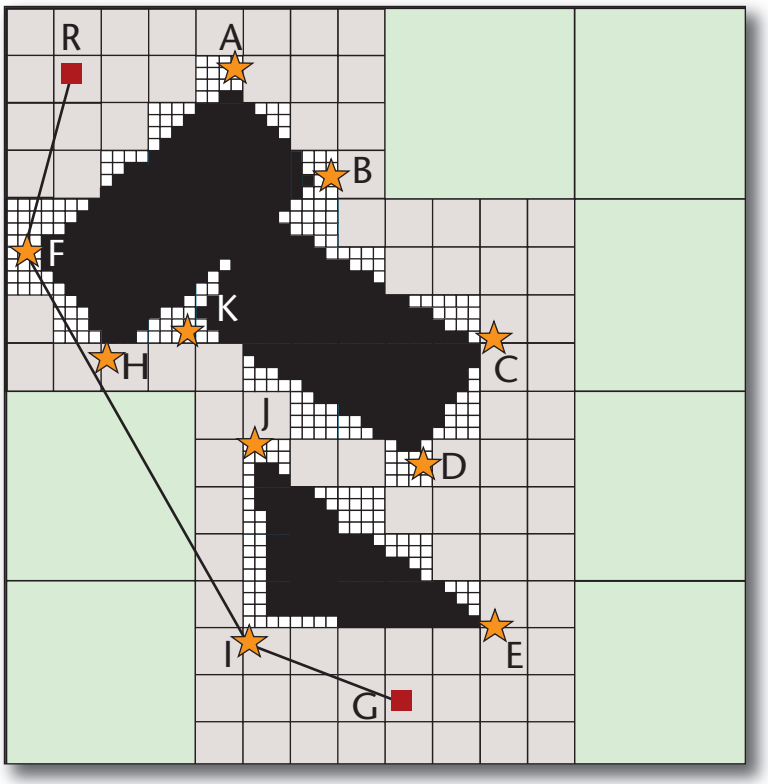

Figure 13. Shakey's Adaptive Grid Model.

and researchers at Carnegie-Mellon University have played major roles in the development of many of these.

The Mars rover, *Curiosity*, uses Field D*, a derivative of A* written by CMU's Tony Stentz and his student, Dave Ferguson (now with Google). It is capable of planning paths around obstacles in unknown, par-

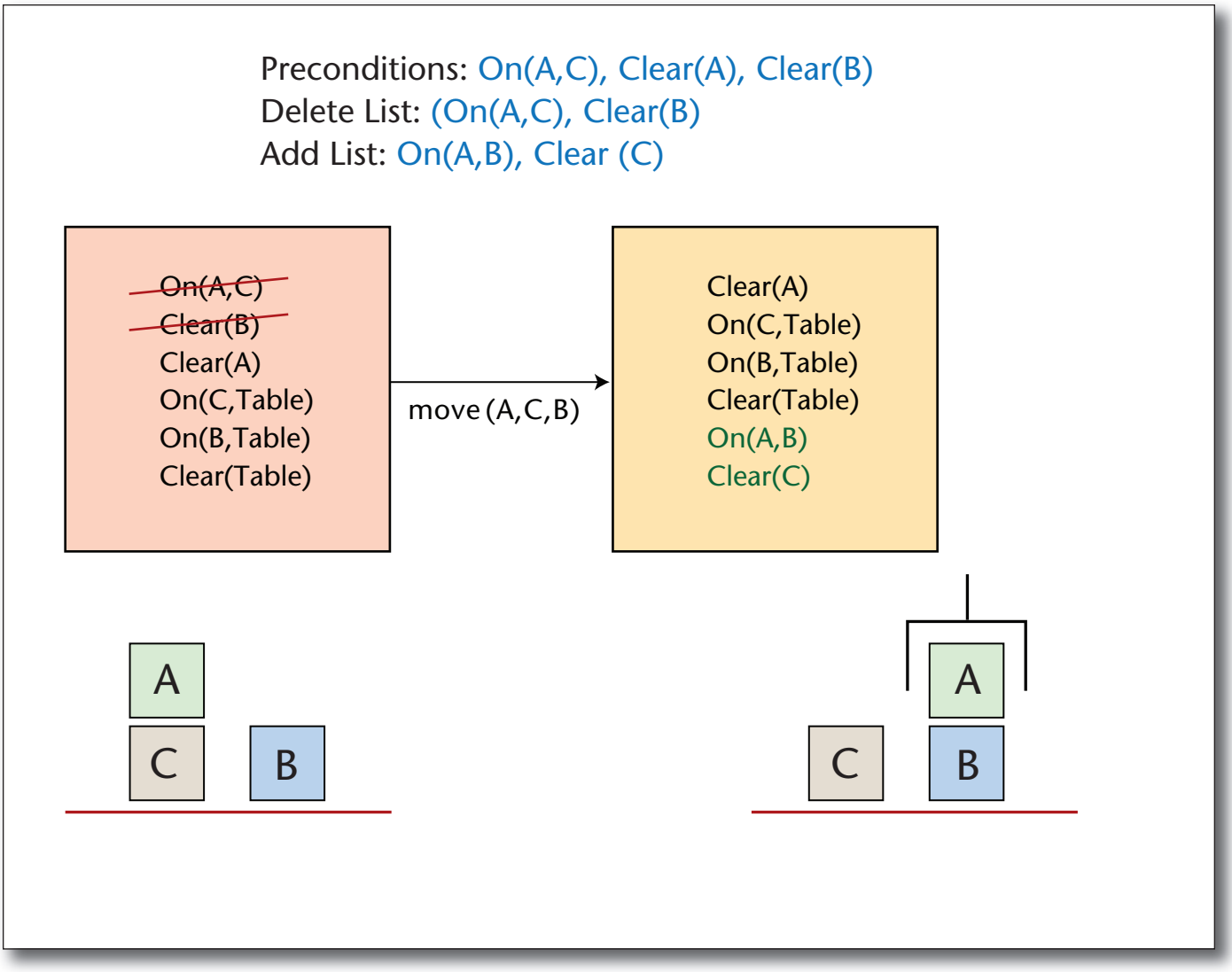

Figure 14. Application of a STRIPS Rule.

tially known, and changing environments in an efficient, optimal, and complete manner.

Most route-finding algorithms in maps use variants and elaborations of A*. Elaborations include the use of hierarchies, saved routes (which don't need to be recomputed), and much more.

In other uses of A*, linguists Dan Klein and Chris Manning write, "The use of A* search can dramatically reduce the time required to find a best parse ..." (Klein and Manning 2003). And Steven Woodcock, a computer games consultant, wrote that "A* is far and away the most used ... and most useful ... algorithm for [nonplayer-character] path finding in games today .... developers have noted that they make more use of A* than any other tool for pathfinding."[11] (Actually, we are gratified that the major applications of A* are on problems a bit more serious than video games!)

Computer Vision

As mentioned earlier, we had hoped that we could use then-existing computer vision routines to process images from Shakey's camera. But the state of computer vision was quite primitive at that time, so we did have to develop some routines of our own. One, which influenced subsequent vision systems, was a system for segmenting images into "like-appearing regions" (Brice and Fennema 1970). An example of the regions found for some of the objects in Shakey's environment is illustrated in figure 16. Segmentation is still a major technique used in computer vision today.[12]

Another result of work on computer vision during the Shakey project was the development of the "modern form" of the Hough transform for finding lines and curves in images. As mentioned earlier, Richard Duda and Peter Hart modified the original

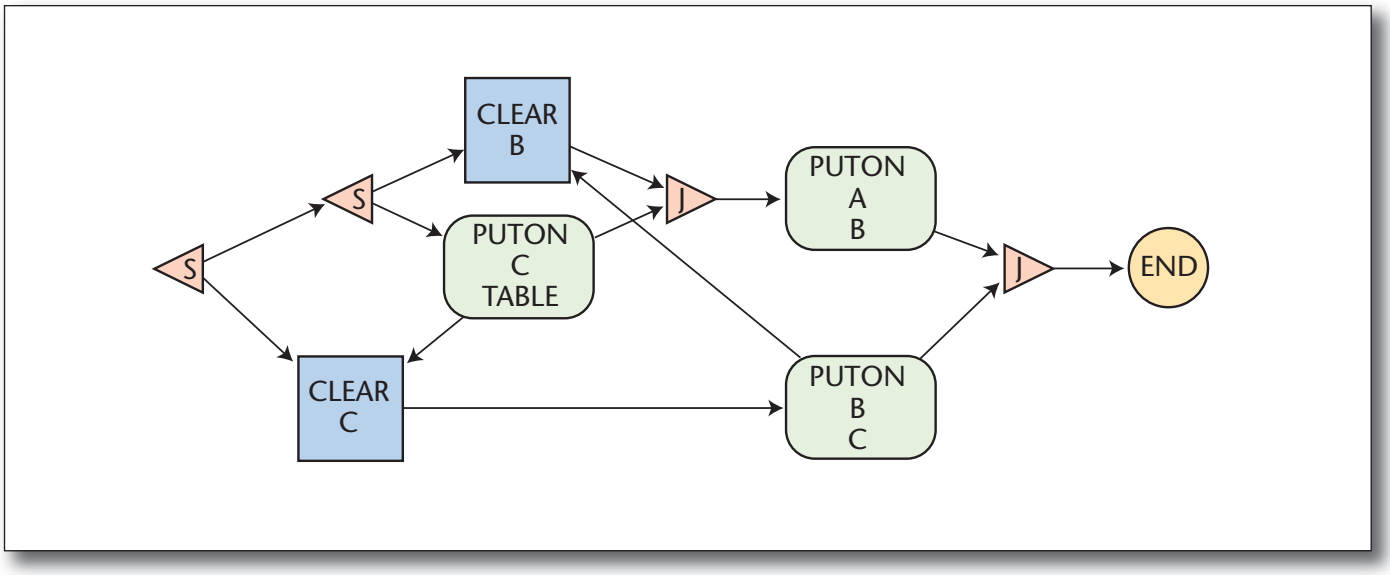

Figure 15. A Hierarchical Task Network.

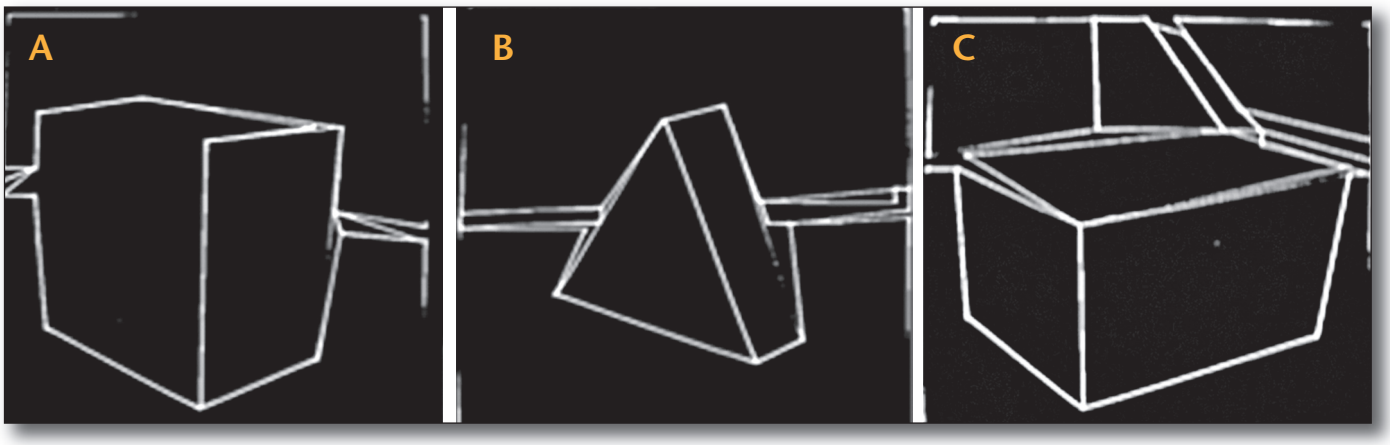

Figure 16. Region Finding as Used by Shakey's Vision System.

Reprinted with permission from Claude Brice and Claude Fennema, Scene Analysis Using Regions. *Artificial Intelligence* 1970 (3-4).

version of the Hough transform to include circles and analytic curves and to use a rho-theta parameterization (Duda and Hart 1972). Their paper gets 4500 hits on Google Scholar.)

The modern form of the Hough transform is used in automobiles to detect lane markings to warn the driver about drifting out of his or her lane.[13]

Conclusions

One reason for the success of the Shakey project and for its extensive legacy is that we were the first group to think that developing a robot that could perceive its environment and make and execute plans was a feasible idea. At the time, there was little existing software for us to use, so we had to invent what we needed. It turned out that the new inventions were ones that had broad applicability once people heard about them.

Another reason for our success is that we had a very talented team of AI researchers and software developers, along with people who could make the connections between software and hardware (figure 17). Some team members had a reunion at SRI International in November 2014.

There are still many problems in AI where talented researchers could be first. An idea mentioned by Nilsson during the panel is to develop an "action hierarchy" analogous to the deep learning hierarchies that are being used for vision and speech recognition.[14] Some of these are said to be rough models of the perceptual part of the neocortex. But the cortex also coordinates and plans actions, as illustrated in the diagram

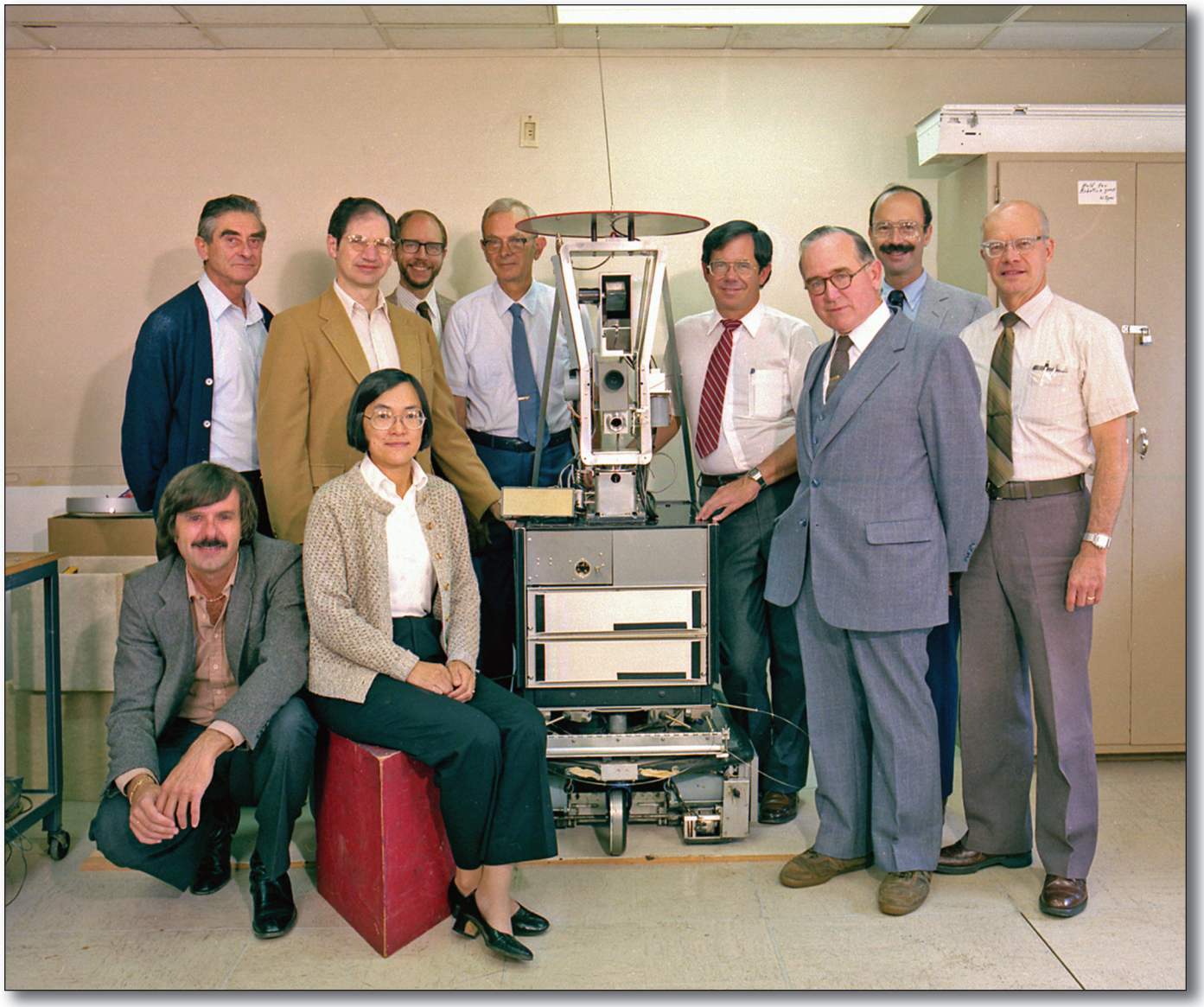

Figure 17. Some of the People Who Worked on Shakey

Front Row, left to right: Richard Fikes, Helen Chan Wolf. *Rear row, left to right:* Charles A. Rosen, Bertram Raphael, Richard O. Duda, Milt Adams, Jerry Gleason, Alfred E. (Ted) Brain, Peter E. Hart, and Jim Baer.

(figure 18).[15] How about developing a "deep action" system, with cross connections to the perceptual hierarchy and using (perhaps) hierarchical reinforcement learning to learn the actions?[16] One could then try to use both hierarchies to control a robot.

Notes

1. www.ai.sri.com/pubs/files/629.pdf.

2. ai.stanford.edu/~nilsson/Shakey.mp4.

3. LIFE, November 20, 1970.

4. For more information, see the T-R website teleoreactiveprograms.net.

5. See wiki.ros.org/smach.

6. The figure is from Nils J. Nilsson, "A Mobile Automaton: An Application of Artificial Intelligence Techniques," *Proceedings of the International Joint Conference on Artificial Intelligence*, 7–9 May, 1969. Washington, DC. Los Altos, CA: William Kaufmann Inc.

7. See en.wikipedia.org/wiki/hierarchical_task_network for more information.

8. See www.ai.sri.com/~sipe.

9. See www.cs.umd.edu/projects/shop.

10. See aigamedev.com/open/review/planning-in-games.

11. Email from Steven Woodcock sent to Nilsson on 6/14/2003.

12. See, for example, en.wikipedia.org/wiki/Image_segmentation.

13. For a video of the Hough Transform in action, see www.youtube.com/watch?v=DPApsnpPjuU/.

14. See, for example, www.cs.toronto.edu/~hinton/.

15. Diagram from willcov.com/bio-consciousness/sidebars/Perception—Action%20Cycle.htm.

16. The following paper seems relevant to this problem: Nicholas K. Jong and Peter

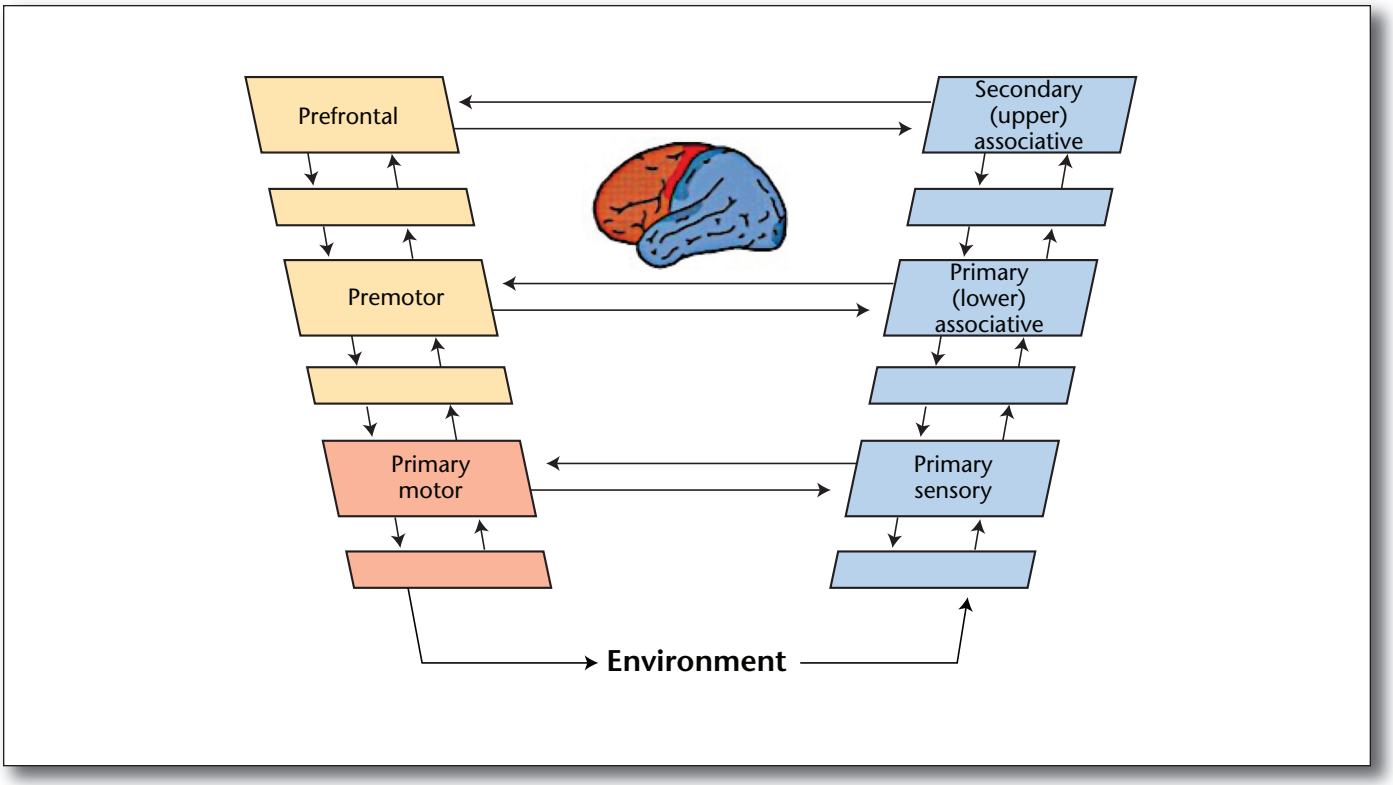

Figure 18. Cortex Motor and Perceptual Hierarchies.
Adapted from Joaquín Fuster, *The Prefrontal Cortex*, 360 (New York: Raven Press.)

Stone, Hierarchical Model-Based Reinforcement Learning: Rmax + MAXQ, in *Proceedings of the Twenty-Fifth International Conference on Machine Learning*, July 2008.

References

Bernard, D.; Dorais, G.; Gamble, E.; Kanefsky, B.; Kurien, J.; Man, G. K.; Millar, W.; Muscettola, N.; Nayak, P.; Rajan, K.; Rouquette, N.; Smith, B.; Taylor, W.; Tung, Y. W. 1999. Spacecraft Autonomy Flight Experience: The DS1 Remote Agent Experiment. In *Proceedings of the American Institute of Aeronautics and Astronautics*, 28–30. Reston, VA: American Institute of Aeronautics and Astronautics.

Brice, C. R., and Fennema, C. L. 1970. Scene Analysis Using Regions. *Artificial Intelligence* 1(3): 205–226.

Clark, K., and Robinson, P. 2016. *Programming Robotic Agents: A Multi-Tasking Teleo-Reactive Approach.* Berlin: Springer.

Currie, K., and Tate, A. 1991. O-Plan: The Open Planning Architecture. *Artificial Intelligence* 52(1): 49–86.

Duda, R. O., and Hart, P. E. 1972. Use of the Hough Transform to Detect Lines and Curves in Pictures. *Communications of the ACM* 12(1): 11–15.

Fikes, R. E., and Nilsson, N. J. 1971. STRIPS: A New Approach to the Application of Theorem Proving to Problem Solving. *Artificial Intelligence* 2(3–4): 189–208.

Fikes, R. E.; Hart, P. E.; and Nilsson, N. J. 1972. Learning and Executing Generalized Robot Plans. *Artificial Intelligence* 3(4): 251–288.

Hart, P. E.; Nilsson, N. J.; and Raphael, B. 1968. A Formal Basis for the Heuristic Determination of Minimum Cost Paths in Graphs, *IEEE Transactions on Systems Science and Cybernetics* SSC-4(2): 100–107.

Klein, D., and Manning, C. D. 2003. "A* Parsing: Fast Exact Viterbi Parse Selection. In *Proceedings of the Human Language Technology Conference and the North American Association for Computational Linguistics (HLT-NAACL),* 119–126. Stroudsberg, PA: Association for Computational Linguistics.

McCarthy, J., and Hayes, P. 1969. Some Philosophical Problems from the Standpoint of Artificial Intelligence. In *Machine Intelligence 4,* ed. B. Meltzer and D. Michie. Edinburgh, Scotland: Edinburgh University Press.

McGann, C.; Py, F.; Rajan, K.; Thomas, H.; Henthorn, R.; McEwen, R. 2008. A Deliberative Architecture for AUV Control. In *Proceedings of the 2008 IEEE International Conference on Robotics and Automation.* Piscataway, N.J.: Institute for Electrical and Electronics Engineers.

Nilsson, N. J. 1994. Teleo-Reactive Programs for Agent Control. *Journal of Artificial Intelligence Research* 1: 139–158.

Benjamin Kuipers is a professor of computer science and engineering at the University of Michigan. He is a Fellow of AAAI.

Edward A. Feigenbaum is a professor of computer science emeritus at Stanford University, a joint winner of the 1994 ACM Turing Award, a Fellow of AAAI, and an AAAI former president.

Peter E. Hart was chair and president of Ricoh Innovations and an alumnus of SRI International. He is a Fellow of AAAI..

Nils Nilsson is the Kumagai Professor of Engineering (Emeritus) in the Department of Computer Science at Stanford University, a Fellow of AAAI, and an AAAI former president.