AI Magazine Volume 6 Number 3 (1985) (© AAAI)

TECHNOLOGY TRANSFER

Robot Eye, and ROI: Technology Transformation Versus Technology Transfer

Earl Sacerdoti

Teknowledge Inc., 525 University Avenue, Palo Alto, California 94301

I want to discuss two aspects of technology transfer. First I've been asked to present a brief perspective on how AI is fitting into a particular application area: Industrial automation. Then I want to give my two cents worth on AI as a business activity.

Commercial AI

My particular focus is on commercial AI, that is, products that incorporate AI that are being sold for profit, as opposed to "practical" AI, in which AI is incorporated into in-house systems to be used internally within an organization. Commercial AI products take the form of equipment, systems, or software.

My perspective is also a function of what I'll call the current socio-techno-economic climate by which I mean the current power of computer hardware, the current state of demonstrated successes in artificial intelligence systems—maybe one or two—and the current number of big, publicly visible failures in artificial intelligence systems—which at this point is zero. (I am concerned that it won't stay that way.)

What Is a Robot?

Everybody has a different definition of a robot. The Japanese define a robot to be a machine that replaces or nearly replaces human movement as an instrument to reduce labor. In other words, if it does a job that a person would normally do, then it's a robot. This is quite a broad definition, which is one reason that the Japanese are said to be so far ahead in the robot business—they count them up differently.

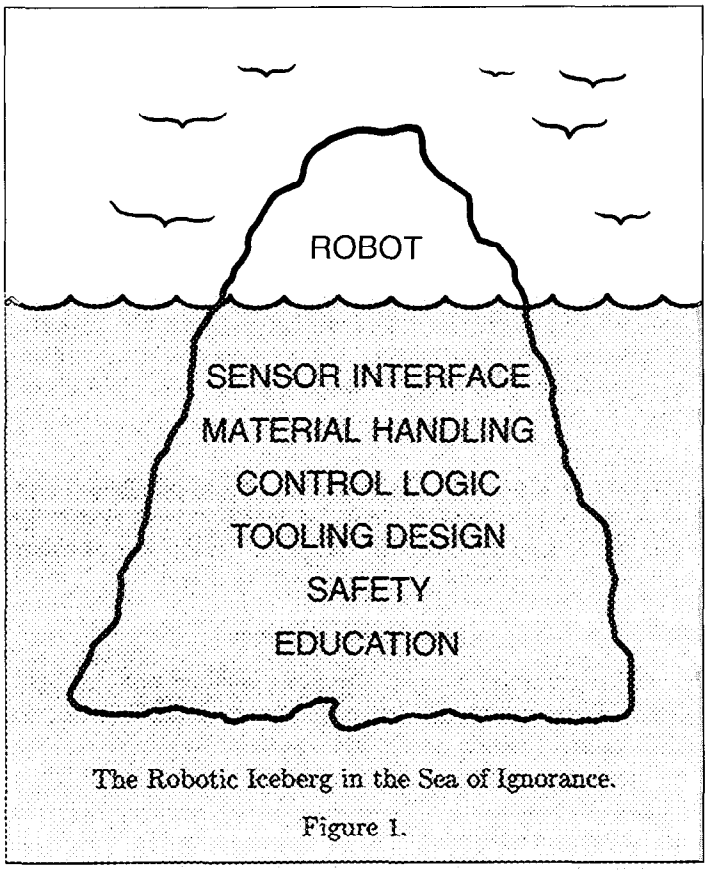
The definition of the Robot Institute of America says a lot about where our technology is: A robot is a reprogrammable multi-function manipulator designed to move material, parts, tools, or specialized devices through variable programmed motions for the performance of a variety of tasks. It took a committee several years to come up with this. It has a lot of words and you can't understand it, but basically it emphasizes the fact that a robot is reprogrammable and that it can do a variety of tasks. Even by this definition the Japanese have a big lead over us in the application of robotics. Interestingly enough, so does Europe. The United States is dead last in utilizing this technology—most of which came out of U.S. industry and AI labs.

The definition we used at Machine Intelligence views a robot as a computer system with a peripheral attached that waves around in the breeze: A robot is a computer-based system whose computations have side effects involving physical manipulation and, optionally, acquisition of sensory data. This forces a different view of a robot, a view that is useful in actually thinking about applications. To illustrate this, I like to use an idea I got from the marketing manager of another AI robotics company. He called it "the robotic iceberg floating in a sea of ignorance." As Figure 1 suggests, all you actually see is the manipulator waving around, and you say, "Well, I see, so that's the robot." But when you consider putting a robot into your factory, you have to pay attention to a lot of other issues such as user interface and system interface, which computer science deals with regularly.

What Is in a Robot?

By 1990 all the optimistic projections say that the U.S. robot market will follow a "hockeystick" curve upward and reach the two billion dollar per year level. This is dramatic and impressive growth, and means equalling a fairly substantial fraction of the U.S. potato chip market. It also means that we are selling futures in the robot business just as we are in any other AI-oriented activity. People are buying robots in small numbers now largely because they are swept up with the potential of the technology, or because they are thinking defensively and want to get some experience with the technology along with their

This is an edited transcript of a talk given two years ago (technology transfer is as slow as technology transfer!) at AAAI-83 Note that the "socio-techno-economic climate" has changed notably in the intervening period The focus on robotics reflects the author's affiliation at that time with Machine Intelligence Corporation (MI)



The Robotic Iceberg in the Sea of Ignorance.

Figure 1.

competitors. But the vast majority of purchases will come later, when robotics becomes a lot more cost-effective. The steepness of the hockeystick curve implies that almost every single robot that gets installed in the next couple of decades will be used by someone who has never used one before. Robots will have to come with a lot of on-board user support, which is one of the things AI is supposed to facilitate.

To make robots broadly usable, then, we have to put a little intelligence into them. I generally refer to three generations of robots. The first generation was quite simple. A robot was programmed by moving it from one specific point to another in the workspace. It could remember the two points and repeat from one point to the other. However, there was no guarantee of how it would behave in between those points. A robot is a complex, articulated structure, and it is a complicated computational job to get it between those two points in any kind of straight line.

The second generation had more computer power and made it possible to interpolate a linear path between the two taught points. These robots could do things like arc welding and other operations that require moving in a controlled way. The third generation of robots can not only move to pre-taught points, but also compute new points in space to go to. This requires even more computer power.

Yet, what we sell as an "AI-based third-generation robot" does not in fact have a lot of AI in it. The key to making this work is a set of equations to transform a point in space into a sequence of relative joint orientations. This technology was developed in the AI labs, but is now becoming standardized algorithmic technique. While the techniques we've embodied in the third-generation robot controllers came out of AI research, the bulk of the incremental value over earlier robot generations comes from things ancillary to the fundamental AI component. The technology was "transferred" by being transformed into something more conventional. That's a major point I want to make.

Third generation robots can work adaptively as opposed to strictly by rote. They can be programmed, not just by physically moving the manipulator, but also with a symbolic programming language. Symbolic programming permits adaptive behavior based on sensory input, from a vision system, for example. It also permits a host-computer interface. However, our symbolic programming language isn't LISP or any other specialized AI language. It's BASIC, because that's what the robot users are most likely to have learned from a technical trade school. In summary, then, while there's a smidgen of ex-AI-lab technique required to provide the third-generation capability, most of the value added comes from good solid 1975-level computer engineering.

Third generation robots can also employ sensors in more sophisticated ways. I like to distinguish between three different levels of sensor interface. The first level is simple sensory switching; the sensor essentially tells the robot to run one fixed program or another. For example, a spray painting robot might work on three different parts as they come down the line. A simple vision system recognizes part B and tells the robot to run fixed program B. This doesn't require a third-generation controller. The second level I call sensory ballistics. In this case, a sensor is used to obtain a static snapshot of the environment it's working in. A path is computed for the robot based on the snapshot. From this point, the robot flies blind and does its work. This is practical in some situations today. "Seeing robots" certainly sounds deeply AIish; in fact, now that it's practical, there's very little AI left in the process. Again, technology transfer has been achieved through technology transformation. At the third level of sensory interface there is true sensory servoing, in which sensory information is continually fed back to the robot. With a few exceptions, this is beyond the state of the art, but it is coming fast.

Perhaps the most important of all for the actual industrial application of third generation robots is a side-effect of the basic technology: The program you put into the robot controller will continue to work even if you have to change the robot manipulator. If you repair it or replace it, you don't have to reprogram it. Previous generations of robots simply remember the number of turns of the mo-

tor on each joint for each taught point in space. Because each physical manipulator is different, a program that is taught to one robot will not move a second robot to the same place, and this can lead to disaster. If a robot breaks down and must be replaced with a spare, the replacement robot must be reprogrammed on the line, and this means additional costly downtime. A third-generation replacement can run the old program immediately, saving time and therefore money.

Vision Systems

I'll now briefly describe the vision system technology used at Machine Intelligence. This will give you a feeling for the current commercial state of the art, and point out again how little of what we would call AI from an academic perspective remains by the time the technology is transformed into a product.

What are vision systems used for? Inspection, material handling, and robotic assembly applications. The state of the art is such that we are dealing only with very restricted scenes, situations in which there are a small number of stable states for the parts. We take the gray level image from a TV camera and turn it into a black and white image by transforming everything darker than a certain level of gray to black and everything lighter to white. Touching and overlapping objects used to be a severe problem. This can still cause trouble, and we prefer to separate objects if we can. Vision systems can be used to verify that your production is good, to identify which of a number of parts you've got, or to find the position and orientation of things.

The key piece of AI technology underlying these systems generally isn't considered AI anymore, because it works so reliably and we understand it thoroughly. It is a simple but powerful technique called connectivity analysis, which was developed at SRI in the early 1970s. Essentially, it chunks together all of the picture elements that are the same color and are contiguous. The connected region is known as a "blob." You can compute several key features about each of the blobs and use that information to analyze the image. For example, you can compute the extents in X and Y of each blob, determine the minimum and maximum radii from the center of the object to the edge, or find the ellipse that has the same first and second moments of inertia about the origin. Or, for example, you can determine the smallest box that fits around the object and that's oriented with this ellipse, which is helpful for picking the thing up with a robot.

Our original MI system could do these things but was not very smart. It would look at every scene with amazement; it had no expectation about what it was going to see. (For most industrial applications, of course, these systems must view the same objects over and over again.) Humans typically perform image analysis much better if we know ahead of time what we're looking at. Our vision system, on the other hand, performed like a typical AI program—the more it knew about something, the worse it did, because it had more data to process. So we, thinking like AI people, asked whether we could process information such that, if the vision system knew more about the situation *a priori,* it could perform better. We asked the AI question, but then we answered it in a thoroughly non-AI way, and got a very effective product, by developing a hardware add-on.

The system can now inspect fixed objects by overlaying electronically an image of what the object is supposed to look like with the real-time image from the camera. In this way, all the system has to process is the difference between the intended scene and the real-time scene. This enables us to reduce processing times for analyzing a complex casting, for example, from many seconds to a fraction of a second. The fact that we implemented this AI approach in a non-AI way, by building a little custom hardware, may be a useful paradigm for looking at other aspects of this so-called AI business as well.

Intelligent Robots

The situation is much the same with robots. In 1980, MI and Unimation (which is the robot market leader in this country) jointly developed what had been in years past the ultimate desideratum of an AI research laboratory, a "hand-eye" system. The technologies we used were the bases for a number of solid, no-questions-asked AI theses back in the late 1960s and early 1970s. By the early 1980s they had come to be regarded as systems engineering, mechanical engineering, and well-understood algorithms. This has been a continuing problem with the field: If you understand how it works, it's not AI any more. All of our successes end up being called something else.

Today's state-of-the-art "AIish" robot doesn't have a whole lot of computing power. The most advanced might incorporate a DEC LSI-11/23. It can't do exciting things from the perspective of the AI developer. But it can do exciting things for the customer. You can put a vision system inside the controller. You can use a dectape to store your robot programs digitally, rather than a more typical audio casette tape, for higher reliability and more data storage. This is solid computer engineering, not a lot of AI. One big advance is to be able to program the robot, in a number of different coordinate systems, in BASIC. Again, back in the late 1960s this kind of thing got you an AI thesis. Today it is straightforward mathematics.

Our robots have other features that are viewed as avant-garde and very attractive to our customers. The programming language can run asychronously from the robot. And the robot can communicate with a host computer, which is going to become an increasingly important feature. Again, big advances in robot intelligence from our users' perspective are simply application of good 1970s-style computer engineering.

What are some of the applications these robots can

be applied to? The possible applications of our robots include:

- In an inspection system, as unoriented workpieces come down the production line, the vision system helps the robot orient them and place them for another vision system to inspect.
- In another inspection setting, a vision system can find an oil drip pan, guide the robot to put down liquid gasketing, and then check for a complete seal.
- In an assembly operation, the robot torques door bolts on an automobile door; the eye is located in the hand, a very neat concept out of the AI labs.
- In a material handling application, the robot picks things that aren't organized and places them on an assembly line.

Directions for Evolution

Where can AI make a difference in robotics? I believe it will make the most sense for AI-oriented businesses to focus on some of the ancillary things. Leave the robots themselves to GE, Westinghouse, IBM, Bendix—not to mention the Japanese. The directions for application of AI in robotics might include:

- Languages for programming.
- User interfaces and networking.
- VLSI implementations of control and sensing algorithms.
- A whole range of sensors—visual, range, acoustic, motion.
- Semi-sentient workpieces—that is, putting some of the smarts on the workpieces themselves.
- Safety issues: People who make robots are very concerned about product liability (for good reasons).
- Geometric modeling techniques.
- Grippers and tooling.
- Mobility.
- Telerobotics: Mixed-mode, partly automatic, partly man-controlled. For instance, you could use remote control to drive the robot to a work area, orient to a few places on the work piece, and then invoke automatic processing once the robot is oriented. There are potential applications in mining, welding repair onboard ship, or even in space.
- Microbotics: Robots small enough, for example, to dredge around in your inner ear and weld bones together.
- Education: Small robots that help people learn about robots and AI. The turtle is a genuine AI concept—a little robot that trains kids how to think like programmers.
- Automated pets: Things that wave around in the breeze or move with some autonomy are interesting and fun to play with. It might be the biggest AI success you could point to.

Development Environments

How have we been programming these robots? For the first and second generations, you drove each robot around to the points you wanted it to get to. You said remember, remember, remember, and it remembered, remembered, remembered. If you had 12 robots, you had 12 trivial development environments. So when people first developed third-generation programmable robots, they put a little development environment and a terminal on each robot. Then for 12 robots, you have 12 terminals and 12 inadequate program development environments. This means you have to do development online; because the development environment was hooked to the robot, you could not have the robot running one job while you programmed a second job. You also had very poor file management capability and terrible user interfaces.

This situation persists because the robot people have been hiding the processor under the skirts of the robot. Their buyers don't want to hear about computers. This is a serious problem that has to be addressed. Robots are going into operating environments where the repair people may try to diagnose a microprocessor-based robot controller with a multimeter.

We have made huge strides in improving computational environments for robot programmers by giving them 1975-style computer engineering tools, which all of us in the computer business simply expect should be there. Now you can have an offline development system with serious file storage capability a well as software development tools and a real operating system that supports multiple users. You can use hard disks instead of floppies or cassette tape. People can use such systems without a lot of training; we put particular emphasis on developing good menu selection capability. That way, chemical engineers or manufacuring engineers don't need to understand the operating system to be able to write useful robot programs. It's also good to standardize all the different robot manipulators around the same programming language. Then you can write one program to control any of them. They all have the same host computer interface and can interface to other equipment. These development systems use a multi-tasking operating system, have interactive online programming, and richer programming environment support offline. Once again, the things of real value in a development environment are conventional computer engineering technologies, some of which are spinoffs from AI.

By the same token, some of the areas where there is potential for using AI commercially in the next ten years were explored in the labs in the 1970s. As robots become more adaptive, problem-solving cababilities will be

increasingly important. As complex assembly tasks become feasible for robots, they will need automated planning capabilites that can use a computer-aided design data base. Interactive modification of robot programs or semi-automatic programming of the jobs will also be very useful. At present maybe 10-12% of robot programming time actually goes to describing the job. The rest of the effort is in handling all the possible error conditions, a task within reach of present AI research systems.

Interactive tools for creating such programs are already being developed. Programming by voice is another technology that is coming down the line.

Intelligent Business Strategies

Generalizing from my experience in trying to bring many kinds of AI technology into practical use (including language understanding, machine vision, robotics, and expert systems), I'd like to leave you with some advice about commercializing AI. I'll focus on three aspects of business strategy: Product strategies, marketing strategies and company strategies.

Product Development Strategies: Technology Transfer Versus Technology Transformation

We need to distinguish between technology transfer and technology transformation. In the area of AI software, technology transfer involves taking systems that run and do something impressive from the laboratory, and modifying and adapting them to make them really robust. In a software-based business, you have to put a lot of work into handling error conditions gracefully because users out in the field are going to do all kinds of nasty things to your program. You need to offer extensive documentation, training, and user support. You can often get away with less software development work by throwing more hardware at the problem.

Technology transfer as I've characterized it here is only practical when the target application closely resembles the original development project's goals. The application should be something that is truly valuable to the customer so that his commitment to it will survive the pain and agony of maintaining the system. And if the application isn't stable, the AI development people may have to be brought back continually as things change. In sum, if you have a stable, high-value application that very closely resembles a successful AI-lab prototype, you can consider technology transfer. Anything else requires what I characterize as technology transformation.

Technology transformation begins with thinking through the application once again from the ground up—even after you have done your prototype. The system should be reimplimemented to guarantee supportabilitity, evolvability, and time- and space-efficiency. LISP is a great environment for prototyping, for figuring out your algorithms, data types, structures, and so forth. But once you know those things, a lot of the value of LISP goes away. In fact, LISP's biggest asset for the experimental computer scientist is that it lets him get away without thinking about storage management. But this is a major detriment for the system deliverer, because LISP makes it quite difficult to manage storage carefully. To get speed of operation or to run the system on a cost-effective microcomputer, use PASCAL or FORTRAN or ADA or the like rather than LISP. Small companies going out into the world with AI products can grow quickly. That means the people who are going to have to support a product won't be the people who developed it. Using a standard language such as PASCAL will make their job easier, and will make recruiting staff easier since many more people are already familiar with the language. While reimplementation in a standard language may cost more initially than simply beefing up an AI prototype, you'll save time and money, and be more responsive to your customers, in the long run.

Keep these caveats in mind: Where you need AI, use it. Where you don't need AI, don't use it. Stick narrowly to the objectives you're trying to get your software to achieve. Eschew creeping elegance. Approach your problems as computer engineers rather than computer scientists. A lot of people who've been trained in AI labs have a very hard time doing something in a straightforward way if it is ugly. I have had the hardest time over the years training people to do things elegantly only when the elegance buys them something and to do things in a straightforward way where it is not going to make any difference. Finally, because AI wizards are relatively scarce, you had better plan from the start to be later than you'd wish and over budget. Because trained AI people are scarce, you'll have even less success than usual throwing more people and more dollars at the system in mid-project, because you won't be able to obtain additional trained AI specialists. So the bottom line here is: Use AI techniques where needed, but try to use conventional methods for the bulk of your system.

Marketing Strategies: In Search of the Real World

A former colleague of mine at SRI told me once in a fit of depression that he was going to entitle his autobiography "In Search of the Real World." Every time he thought he had taken his software away from a research focus and applied it, he discovered there was another, more real world behind it for which his "application" looked like research. The main point I want to make here is that you need to be problem-oriented rather than solution-oriented. Identify a real problem and then go sell a solution to it. Just because a computer system behaves intelligently doesn't mean it's good for something.

Many AI types who I talk to presume that customers are going to be interested in purchasing software because

it has AI in it. I believe, on the contrary, that customers are wary of AI systems that cost more and have more perceived risk than systems built with a standard programming approach. The majority of the market for AI-based software will be people who will buy it not because it embeds AI, but in in spite of the fact that it embeds AI. So what should you do until the market and the technology mature? You need to find risk-taking companies or risk-taking groups within companies. Initial sales are likely to be to R&D divisions of large organizations where the buying decision can be made at the local level—or else directly to top management based on the long-term potential. It is a tough sell. Try to pick tasks with immense value to the customers. Then you have some margin if you run into problems. Be prepared to support the customer twenty times more than you thought you would have to.

Finally, listen to the customer. Those of us who stuck with the field through the "dark ages" of the mid-seventies are now AI leaders precisely because we didn't listen when our funder "customers" urged us to abandon the field as fruitless. Those who listened to the customer are now working in other areas. The rest of us who wish to make AI practical must now change our habits. Don't go off and do what you think the customer ought to want. He knows what he wants, even though people in AI sometimes have a hard time believing that.

Company Strategies: The Hippo from the Hat Phenomenon.

People who aren't familiar with AI see some of the surprising things that can be done in a prototype system and walk away feeling as if they've just seen somebody pull a rabbit out of a hat. Because they don't really understand the underlying technical problems, they often extrapolate from the demo in inappropriate ways. They figure that if flakey research people can get rabbits out of a hat, then if they could just get some real good people, throw five times the money at the problem, and manage the darn thing properly, they could get hippos out of that hat. But we're dealing with problems where the cost of solutions grows exponentially with the complexity of the job. The last 10% of solving anything is the toughest; the last 10% of solving problems with AI techniques has rarely been attempted and may in fact be often impossible.

This leads to the unfortunate prediction that most impressive AI prototypes will never be fielded. Companies with impressive demos will fail to get products to market. Organizations that believe they are nearly done with applications will fail to complete them. This suggests several strategies. Try to tackle small problems. If possible, don't make the system completely automatic. Let your system be highly interactive so that you can let people do the hard parts of the problem. Solve the easy 90% ; let people continue to do the hardest part.

Stick to the reliable technology that has proven itself in multiple applications. It's a lot less exciting to use the same boring approach that worked in lots of other applications. On the other hand, you're more likely to find yourself with a boring success rather than an exciting, unfinishable failure.

Remember that wizards wave two-edged wands. Until the kind of AI you're using becomes conventional, you'll need them to put AI in your system. You have to pamper them and pay them well. You have to give them a good computing environment. But in terms of company strategy, do everything you can to make them superfluous. Keep the piece of the system that has AI narrow, focused, and well bounded.

It's okay to trumpet the glamor and pizzaz of AI if it's part of your product. Just don't believe the glamor story yourself. Most of any succesful application or product is going to be standard computer engineering. Make sure that you know what your reality is at the bottom line.

Don't expect cookie-cutter production. In the last year and a half I have seen half a dozen business plans that all have the following scenario: We are going to do a prototype AI system for an initial customer, then make it into a really rugged product. We can sell it to six other customers for almost the initial price, although our incremental costs to adapt the system will be maybe 5% of that of the original system. However, when the plans get implemented, I fear that each of those followon systems is really going to be at least 10% different. My earlier argument about the last 10% suggests there will be a lot of work and a lot of cost left to adapt the system for each new customer. So there go the profits. Sticking to reliable, boring techniques may be the key to profitability here.

Stick to a business. You have to decide whether you are a software producer, a manufacter of equipment, a consulting group, or a contract R&D house. You probably can't do a good job if you are trying to do more than one of these. At Machine Intelligence, for example, we faced the problem of whether we were a software producer or an equipment manufacturer. We had to be very hardnosed about staying out of interesting AI software areas, even though there were people willing to throw money at us to try them.

Finally, don't define yourself as an AI business. AI may be a set of techniques, or it may be a technology, but in any case it's not intrinsically the basis for a business for anybody. AI is only of economic value to the extent that it enhances solving problems of value. So in defining your value as a business, don't let the AI tail wag the problem-solving dog. I expect that the business successes from the AI field will be those that solve customer problems, or help customers solve their own problems, and not those that simply facilitate using AI technology.