Real Robots Don't Drive Straight

Fred G. Martin
University of Massachusetts Lowell
Computer Science
1 University Avenue
Lowell, MA 01854 USA

Abstract

Over last fifteen years, robot technology has become popular in classrooms across our whole educational system. Both engineering and AI educators have developed ways to integrate robots into their teaching. Engineering educators are primarily concerned with engineering science (e.g., feedback control) and process (e.g., design skills). AI educators have different goals—namely, AI educators want students to learn AI concepts. Both agree that students are enthusiastic about working with robots, and in both cases, the pedagogical challenge is to develop robotics technology and provide classroom assignments that highlight key ideas in the respective field. Mobile robots are particularly intriguing because of their dual nature as both deterministic machines and unpredictable entities. This paper explores challenges for both engineering and AI educators as robot toolkits evolve.

Introduction

Feedback is a central process in our lives, but its operation is often invisible. Because of our own abilities to learn, we are generally not aware of how pervasive and ubiquitous feedback is. Autonomic body processes, like temperature regulation and breathing, happen without our conscious attention, and learned activities, like balancing and walking, are performed without deliberate attention.

Feedback is also a central process in engineered systems. Historic and modern technologies, from clocks to automobiles and ovens to jets, make extensive use of feedback in their controls.

Over the last fifteen years, many educators have introduced mobile robotics to students. For engineering educators, robotics is popular for introducing students to feedback. As robots become more mainstream, though, "more advanced" toolkits now offer closed-loop movement commands as a primitive. By abstracting away feedback, we run the risk of short-circuiting students' learning.

A similar pedagogical challenge exists for AI educators who are integrating robotics into their courses. Certain aspects of AI map well onto practical robots, while other parts of the conventional AI curriculum do not.

In this paper I examine parallel pedagogical challenges for both engineering and AI educators. There is a deep interaction among the technological capabilities of the materials we provide to our students, the practical challenges and structured problems we give to them, and the ideas we want them to learn. As robotics technology continuously changes, we must keep our learning goals in the front of our minds.

Feedback Is Invisible, Yet Pervasive

Every day we use feedback. Walking down the street, driving a car, and riding a bicycle are all examples. Our ability to constantly and continuously correct our motion is so competent, so automatic, that we do not realize that it is happening.

While it is normally invisible, our personal feedback processes are sometimes easily observable. Often this is the case when we are learning something new. Here are two examples, both dealing with locomotion:

- I have recently starting swimming laps. Swimming straight to stay in the lane was a challenge. If I didn't pay attention, I would veer off and side-swipe the lane barrier. Then I discovered the lane stripe painted on the floor of the pool. Without exactly realizing how, I discovered that I was taking regular glances at the lane stripe. By using this visual feedback—performed without my explicit focused attention—I was able to stay in a straight path.
- I was helping my 3-year-old ride his tricycle down the sidewalk. He was pretty well able to pedal and move along, but steering required extra attention. I gave the same advice repeatedly: "Keep steering toward the middle of the sidewalk." Presently, he is successfully veering back and forth across some imaginary middle of the sidewalk.

In one sense, my son is not yet "driving straight." In his riding, his corrections to stay on the sidewalk are large, obvious, and deliberate. But in another sense, he *is* driving straight: an experienced rider is also makes constant course corrections. The only difference is that for the experienced rider, the corrections are small and unconscious.

Students Don't Believe In Feedback

As we become competent with our physical selves, the feedback actions we perform in our everyday lives become in-

ternalized and invisible to us. We literally are not aware that we are doing it.

In earlier work, I found that students are unlikely to develop feedback-based approaches in their designs of mobile robots in contest events. They prefer imperative programming, in which they expect robots to reliably execute command actions. This approach leads to brittle designs, but even after this is demonstrated to the students, they resist adopting approaches that embrace feedback (Martin 1996).

I was reminded of this in a recent lecture in my undergraduate mobile robotics course. The topic was wall-following using a distance sensor. I explained a "1-threshold" algorithm where the robot takes a distance reading and then decides if it is too close or too far from the wall. If it's too close, it steers away; if it's too far, it steers in.

Upon hearing this, a student burst out, exclaiming, "You mean it will be always weaving back and forth?!" Yes, exactly, I replied. The student's reaction had reminded me that most people do not consider weaving back and forth to be an effective or recommended algorithm.

Later, in reviewing the class's work, it was hard to tell if their robots were actually wall-following properly. Yes, they told me, they tested them on the bench and they were sure that the left and right motors were powering up and down depending on whether the robot was too close or too far. But several of them used only a small power differential between the two sides—that is, to turn toward the wall, the outside wheel was at full power while the inside wheel was at more than 50% power.

In other words, they were so reticent to see feedback in action that they made sure that any turns would be so gentle as to be nearly imperceptible! Of course, that solution doesn't allow the robot to navigate a corner very well.

Robot as Machine vs. Robot as Creature

In the earlier work cited, I also observed that students like it when robots drive in straight lines. This was part of what I called the "omnicient robot fallacy"—the idea that students design robot performance structures by imagining what they would do at the helm of their robot, looking down at its performance arena with an all-knowing eye (Martin 1996).

Of course, robots don't work this way. But students still hold on to the idea that a robot with a left- and right-side motor should drive in a straight line if you turn both motors on in the same direction and at the same power. Never mind that there are many internal factors (motor and geartrain performance) and external ones (irregularities on the ground) which would prevent the robot from indeed going straight.

Related to this is the notion that timing is an effective way to get a robot to translate a known distance. In other words, if I would like my robot to move 10 centimeters, I can accomplish this by turning on its motors for a particular period of time (determined by experimentation).

Experienced roboticists know that this is a crude and only marginally effective approach. Depending on battery level, surface friction, internal gear friction, and other variables, the actual distance traveled may vary widely. Nevertheless it is a simple and sometimes effective method.

The extent to which students are inclined to use timed motions may depend significantly upon performance arena (e.g., contest) that is provided to them. For example, students might be much more likely to think of straight line movements in a contest where the game elements are placed in highly structured surrounds versus one that distributes those elements more arbitrarily.

For example, consider the following two robot contests. The first is shown in Figure 1. The upper diagram depicts the 2004 First LEGO League (FLL) contest. In it, robot designers may conduct various missions, which include retrieving the CD-ROM, collecting balls and delivering them to the basket, and pushing in the chairs at the little dining table (First LEGO League 2004).

In the close-up of the ball / CD-ROM / dining table areas, it is apparent that there are *no features in the environment that can help robots locate the contest components*. Remember that these robots do not have vision systems. Thus, in order for robots to interact with these game objects, they typically must dead-reckon toward them, maybe using the edge of the playing field for reference. The concentric circles around the dining table seem to be there for human reference; the lines are too faint to be reasonably detected by robot sensors. The basket area, on the other hand, does have black lines that might be readily sensed by a robot for positioning.

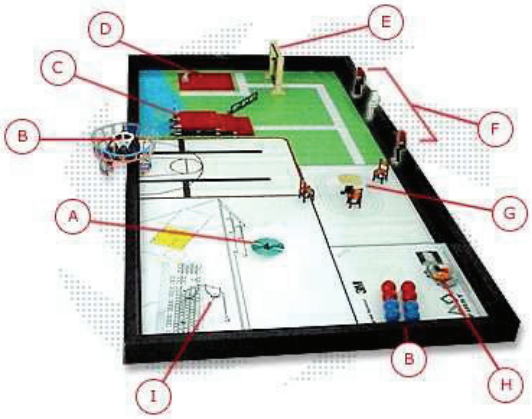
Now consider the Case-Western Reserve "Egg Hunt" contest, shown in Figure 2 (Beer, Chiel, & Drushel 1999). Robots traverse a large, unstructured playing field, foraging for plastic eggs. When they find one, they decide whether it's a good egg (and then bring to their own goal) or a bad egg (maybe bring to the opponent's goal). Each contest round lasts 10 minutes—quite long by the standards of most robot contests.

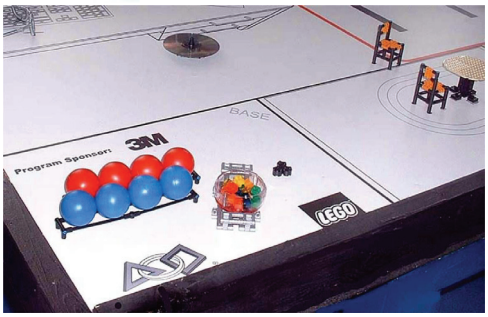
It is no accident that the principals in the Case-Western contest are first biologists, and secondarily roboticists. The design of the Egg-Hunt contest clearly reveals thinking about the *robot as a creature*, while the design of the FLL contests (and the many like it) conceive of the *robot as a machine*.

There are real implications to these two different worldviews. In the Egg Hunt contest, the notion of driving in a straight line literally does not arise. There are no features that are aligned in a straight path; indeed, the game objects are randomly scattered about. As such, students are explicitly encouraged to think of their robots as creatures, with the ability to "survive" for a protacted period of time. Ten minutes is long for a contest robot, and the Egg Hunt robots are expected to not get stuck in a corner and take themselves out of commission for significant chunks of time.

In the design of the LEGO NXT system, introduced in 2006, the developers of the LEGO materials have gone the opposite route. Since it is so difficult to get a toy robot to drive straight reliably, and yet it seems to be so necessary (per the design of the FLL contests), LEGO has decided to *making driving straight a primitive*.

The technical solution involved creating a motor with an integrated encoder (Figure 3) and software that transparently builds a closed-loop control system, including keying two





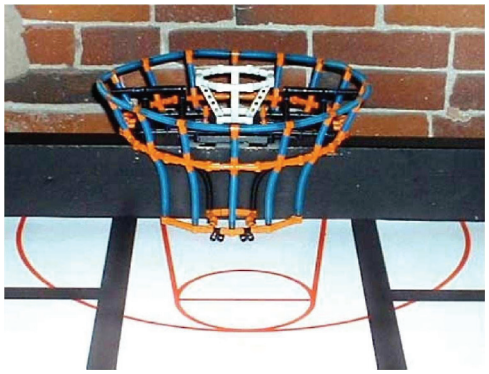


Figure 1: First LEGO League 2004 “No Limits” arena (top); ball, table, and CD-ROM closeup (middle); basket close-up (bottom).

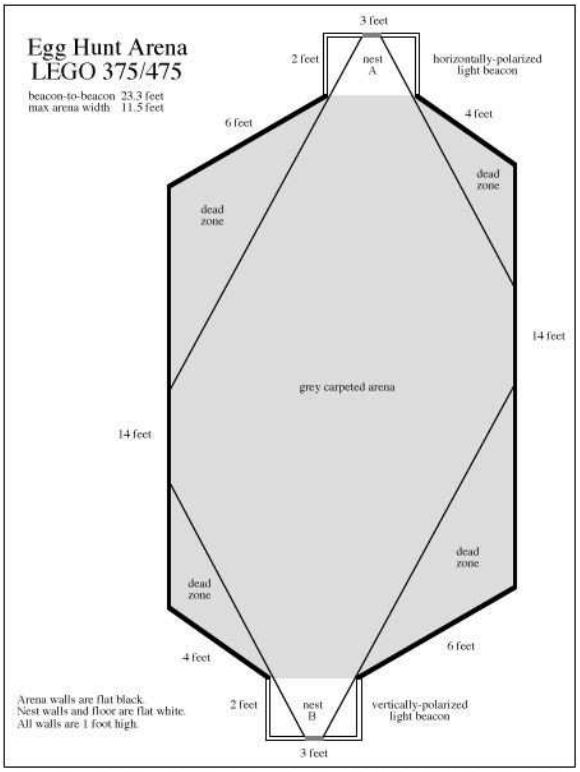


Figure 2: Case-Western Reserve University “Egg Hunt” competition arena (top); game play (bottom)

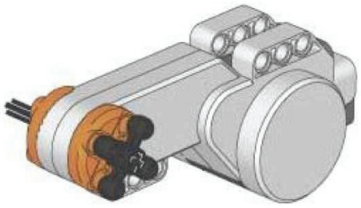


Figure 3: LEGO NXT motor

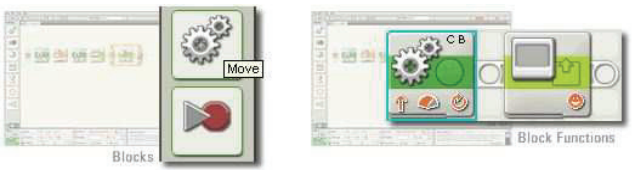


Figure 4: LEGO NXT motor software control block

motors together (Figure 4).

The motor control function accepts as parameters both the desired velocity and the total number of axle rotations to be performed. Thus, with the LEGO NXT materials, it is easy to build a robot that moves in a straight line, and for a predictable distance. Also, robots can be made to rotate in place for predictable angles.

Why a Straight Line Primitive is a Bad Idea

By providing these built-in closed loop primitives, LEGO is depriving its users of one of the critical engineering lessons of classroom robotics: the principle that robots need to sense their local environment and react to it in order to perform useful work. The central premise that robots are feedback systems has been buried.

Also, for middle school-age students, the First LEGO League contests are a crucial exemplar of a robot environment. In other words, they matter. Historically, these contests have not included a plethora of local features for robots to sense in their quest for aligning with game objects. Thus, the contests themselves also encourage students to think about planned, fixed movements. This is why the FLL community has been so frustrated with difficulties in using the earlier LEGO technology to carry out pre-planned motion.

But the cure is worse than the disease. There is ample evidence that people learn through failure. Also, the community learns over time. For example, instructional videos for FLL teams are freely available from the Minnesota-based "High Tech Kids" parent/teacher/student group. These videos, developed by a FLL student-turned-educator, cogently and intelligently explain how to use light sensors to detect contest features (High Tech Kids 2006).

As the FLL community migrates to the NXT controller, contest designers will be more likely to make use of the ability of robots to carry out pre-programmed motions. Contest designs may include even fewer features that allow the development of genuine feedback solutions. FLL robots will migrate toward open-loop solutions, in which the robots simply carry out a series of pre-planned actions.

Of course, in order to drive straight, these robots are doing feedback on using their encoder sensors and modulating their motor power output. But by burying this feedback into a primitive behavior, it will be even harder for students to appreciate its central role in robotic systems (and engineering in general). Without the feedback of failure, students will be deprived of the opportunity of developing this authentic and deep knowledge of what robotics is about.

AI and Robotics

Many educators are enthusiastic about using robotics in their undergraduate artificial intelligence classrooms, but there are challenges in aligning AI content with what robots can do. Also, AI educators do not typically wish to focus students' attention on engineering issues, and this has been difficult to avoid when using low-cost robotics materials.

The content taught by AI educators using robots encompasses a range from traditional AI to material that is more specific to robotics. For example, Klassner describes a course he developed that builds upon agent paradigm popularized by Russell and Norvig (Klassner 2002). The content includes stimulous-response robots, sensor accuracy and functional simulation, robot odometry using encoders, and robot algorithms that used hill-climbing, knowledge representation, and probabilistic modeling. Also, in the final unit, students confronted the difficulty of translating AI algorithms for execution on limited hardware. As such, this final project blends AI and engineering themes.

In another example, Greenwald and Artz describe particular algorithmic content that can effectively be taught with low-cost robots. If robots are equipped with wheel rotation sensors, then the trigonometry required for forward and inverse kinematics can be developed. Map-building and vector-field histograms can then be taught and demonstrated. But as they note, "While these approaches to localization are educational, they are not considered to be a part of a modern artificial intelligence curriculum." (Greenwald & Artz 2004)

Other work focuses on knowledge-based AI algorithms (Kumar 2004; Schafer 2004). Typical projects include AI topics such search, expert systems (forward- and backward-chaining), simulated annealing, planning, and game-playing.

In these two papers, the authors are concerned about excessive student time being spent on engineering or play. Even when students report having enjoyed these activities, it is problematic if they are spending inordinate amounts of time on the course, or if their work is insufficiently focused on appropriate content.

Both authors also report problems that students have in getting robots to perform properly. For example, in the Schafer course, students were supposed to be completing robot development and separate software-only AI programming simultaneously. When the robot work apparently required too much time, Schafer cut back on required software-only assignments. But then students might be missing this necessary content.

Schafer then postulates that programming time can be reduced by providing students with a "calibrated robot control package" which then might allow students to complete more sophisticated coding. Then students could tackle more complex labs that incorporated the previously software-only content. This package would then replace a series of calls such as:

```
motorA.setVoltage(9);
motorB.setVoltage(8);
motorA.forward();
motorB.forward();
```

with a single call to a helper method, such as:

```
RobotControlPackage.robotForward();
```

Schafer realizes that "development of this package is expected to be non-trivial." Let me go further—the quest for such a package is actually a wild goose chase!

Kumar also has students developing traditional AI-style control programs for their robots. He has several creative solutions for the problem of robots that don't work right. Students are encouraged to have their robots announce their internal state before each gesture. This is to help the student and instructor "evaluate the correctness of the underlying knowledge-based algorithm." Kumar also uses pliable performance environments for the robots: "A flexible environment can significantly alleviate the problems arising from the unpredictability of robot behavior." He explains further:

> In a fixed-wall maze, if a robot turns by 75 instead of 90 degrees, the robot may end up in a room other than the one it intended to visit. In a moveable-wall maze, if a robot is found to turn incorrectly, the wall that it should have encountered can be moved into its path, essentially making the room it visits the one it intended to visit!

Kumar summarizes: "As long as the focus of the robot project is a knowledge-based algorithm (and not robotics), and the robot announces its state and intention before each movement, moving walls to address errors in the navigation of a robot is inconsequential to the correctness of the project."

Kumar's paper also includes an exemplary student-learning evaluation. In short, students like the class, and are learning reasonably well. In figuring out why their robots don't work—caused by a composition of issues from both the hardware domains and their own programming of the AI algorithms—students may confront the AI algorithms and understand them more deeply.

But, the AI that these students are learning can not be applied to solving the actual robotics problems they are facing. In other words, while they are learning AI algorithms, they are not learning them in the context of problem domains in which the AI is suited. It may even be argued they they are learning ways in which AI fails!

In contrast, consider the Greenwald/Artz work mentioned earlier. The main theme of this paper is the development of neural and Bayesian networks to perform processing of IR reflectance sensors on low-cost robots. The result is the transformation of seeming unreliable—and definitely noisy—sensor data into useful information.

The paper not only presents the work itself, but also demonstrates an application of modern AI theory that is deeply connected with the capabilities of the pedagogical hardware (classroom robots). Indeed, the work itself draws out latent capabilities of the inexpensive hardware, which are revealed by powerful techniques of modern AI.

To summarize, *mobile robotics changes AI*. Because of profound uncertainties in sensing the world and modeling it, knowledge-based AI can be quite difficult to apply to building effective mobile systems. This was Brooks' key observation in his seminal work in behavior-based robotics (Brooks 1986). One of the great strengths of classroom robotics as pedagogical tool is that students' robots become real systems, not just classroom exercises. We must pay attention to which ideas from our field do help our students build effective machines.

Conclusion

For many teenagers, LEGO Mindstorms is much more than a toy. It is part of a crucial formative experience in engineering and robotics that carries them toward a career path into technology. For many of our college students, robotics can also serve a crucial role. We have all noticed the extravagant amounts of time that many devote to their robot projects. The values and ideals embedded in the materials and practical challenges that we give to our students—whatever their age—do matter.

Returning to the central theme of feedback, no biological or engineered system ever moves in a straight line. As biological systems, we personally get better and better at correcting our navigation movements, until we perceive ourselves as "walking straight" or "driving straight." Over time, our ability to engineer feedback systems has also steadily improved, to the point where vehicles and processes that would normally be highly unstable (e.g., a fighter jet aircraft) can be made to "fly straight."

But in practice, both biological and engineered systems do make constant corrections, be they minimal to the edge of perception. As educators, we must introduce students to this central principle, not hide it from them. Real robots don't drive straight.

References

Beer, R. D.; Chiel, H. J.; and Drushel, R. F. 1999. Using autonomous robotics to teach science and engineering. *Commun. ACM* 42(6):85–92.

Brooks, R. A. 1986. A robust layered control system for a mobile robot. *IEEE Journal of Robotics and Automation* RA-2(1):14–23.

First LEGO League. 2004. `http://www.firstlegoleague.org/default.aspx?pid=14190`.

Greenwald, L., and Artz, D. 2004. Teaching artificial intelligence with low-cost robots. In *In Accessible Hands-on Artificial Intelligence and Robotics Education*, number SS-04-01, 35–40.

High Tech Kids. 2006. `http://www.hightechkids.org/?2-1-1054`.

Klassner, F. 2002. A case study of LEGO Mindstorms' suitability for artificial intelligence and robotics courses at the college level. In *SIGCSE 2002*, 8–12. New York, NY, USA: ACM Press.

Kumar, A. N. 2004. Three years of using robots in an artificial intelligence course: lessons learned. *J. Educ. Resour. Comput.* 4(3):2.

Martin, F. G. 1996. Ideal and real systems: A study of notions of control in undergraduates who design robots. In Kafai, Y., and Resnick, M., eds., *Constructionism in Practice: Designing, Thinking, and Learning in a Digital World*. Mahwah, NJ: Lawrence Erlbaum Associates.

Schafer, J. B. 2004. Hands-on artificial intelligence education using LEGO Mindstorms: Lessons learned. In *Proceedings of the 2004 Midwest Instruction and Computing Symposium*.