Erdos: Cost-effective Peripheral Robotics for AI Education

Zachary Dodds and Ben Tribelhorn

Harvey Mudd College Computer Science Department
301 Platt Boulevard
Claremont, CA 91711
dodds, btribelh@cs.hmc.edu

Abstract

This work combines hardware, software, and curricula in order to create robots capable enough to advance the field of AI yet inexpensive enough to be widely accessible. Costs are kept low by pairing iRobot's roombas with existing laptop or palmtop computers and their accessories. The result is a sub-$200 untethered physical platform capable of running and testing state-of-the-art AI algorithms.

Motivation for Peripheral Robots

The promise of peripheral robots is emerging atop two foundations: the powerful (and costly) selection of built-for-research robots and the inexpensive (but far less capable) world of robotic kits. Figure 1 highlights the price/performance tradeoffs among several well-established and in-development robotic systems. Because iRobot's roomba [1] vacuum offers a compelling balance among these factors, we at Harvey Mudd College are developing software, curricula, and a hardware interface atop this commercially successful base. We have named this freely available library of resources *erdos* for its potential to convert coffee into proofs: in particular, proofs-of-concept in which an embodied system might validate an AI algorithm or approach.

Figure 2 contains a summary of the built-in capabilities [2] of the $150 RoombaRed platform. Figure 1's $175 cost includes the USB-to-TTL converter required to interface with the robot's serial port. For $60 more, iRobot also sells a docking station which the robot can find and self-recharge, provided a line-of-sight path to its IR emitter.

Erdos: Software for the Roomba

Figure 2 illustrates the components of the erdos library. The base operates as a serial device, either cabled or wireless, according to iRobot's Serial Command Interface (SCI) specification [2]. SCI provides a byte-level interface; our `SRSerial` class simulates this interface *byte-for-byte* in case a physical platform is unavailable or unwanted.

Erdos	**\$175**	**PC; Vis,Mic,Bmp,Cli,Enc,WL**
Lego RCX	**\$200**	**μP; Bmp,Lt**
Lego NXT	**\$250^{nr}**	**μP; Bmp,Lt,Son,Enc,WL**
CMU TeRK	**\$250^{nr}**	**PC; Vis,Mic,Bmp,Lt,IR,a2d,WL**
IntelliBrain	**\$300**	**μP; Bmp,Lt,IR,a2d,IR**
CMU PPRK	**\$325**	**μP; IR,a2d**
HandyBoard	**\$350**	**μP; Bmp,Lt,IR,a2d**
KIPR XBC	**\$500**	**PC; Vis,Bmp,Lt,IR,Enc**
UMN eROSI	**\$500^{nr}**	**μP; Lt,Enc,Pyr,Acc,WL**
HandyBoard 2	**\$700^{nr}**	**PC; Vis,Bmp,Lt,IR,Enc,a2d,WL**
Hemisson	**\$780**	**PC; Lt,IR,WL**
Garcia	**\$1725**	**PC; Vis,IR,Enc,WL**
Sony AIBO	**\$2000**	**PC; Vis,Mic,Bmp,Enc,WL**

Figure 1 Comparing untethered robots, their prices, and their capabilities. **Legend**: **nr** indicates a platform not yet released as of 5/'06: prices are estimated. **Computation**: **PC** = personal-computer-based, **μP** = microprocessor-based. **Sensing**: **Vis** = vision, **Mic** = microphone, **Bmp** = bump sensing, **Cli** = cliff sensing, **Enc** = encoders/odometry, **Lt** = light sensing, **Son** = sonar ranging, **IR** = infrared ranging, **a2d** = ttl-level inputs.

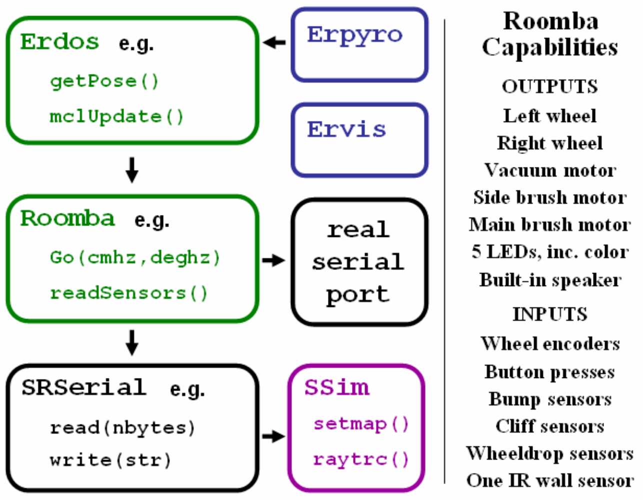

Figure 2 At left is a block diagram of the erdos software library; at right, some of the roomba's sensing and actuation abilities.

The **`Roomba`** class can thus provide an identical *low-level API* of actuation and sensing commands for both simulated and real-robot development. Arbitrary error models for simulation may be used; reasonable ones are built-in. The **`Erdos`** class then adds an *interpretive* computational layer, e.g., integrating odometry to estimate location with particle filters available to refine those estimates. Accompanying classes (**`SensorSimulator`**) provide support for geometric computation, visualization (**`Ervis`**), and hooks for external tools, such as pyrobot [3] (**`Erpyro`**).

Though external interfaces are completely in keeping with the spirit of the erdos project, it *needs no prerequisites* beyond standard python with serial support. As a result, erdos works out-of-the-box under Windows, MacOSX, Linux, and many other systems. Erdos may be obtained from [4]. That site also details soldering-free do-it-yourself robot setup. For those who might prefer to outsource the setup, we also link to vendors for the interface hardware. They typically add about $25 to the total cost.

Erdos in Education: Spring '06 Experiences

We used the erdos library during the Spring 2006 semester of CS 1 (CMP 202) and a robotics-themed seminar (CMP 304) at Chatham, an all-women's liberal arts college in Pittsburgh, PA. As is, the erdos system comfortably supports a curriculum – or research – including reactive robotics, state-based architectures, the interaction of multiple agents, monte carlo localization and probabilistic reasoning, as well as both traditional and sensor-limited path planning. Because the roomba is a serial device, there are no language, system, or computational limitations except those of the controlling onboard palmtop, laptop, or desktop computer – the last presumably offboard!

The CS 1 class used the erdos simulator halfway through the semester to introduce the notion of functions and computational abstraction. The motivating problem was escaping from arbitrary mazes: students implemented the pledge algorithm [5] as an assignment. Later in the term the students programmed the physical erdos-run roombas as a review project touching on while loops, exceptions, and conditionals. The advanced class delved more deeply into AI robotics per se: they implemented state-machine architectures, the bug algorithms for sensor-limited navigation [6], Monte Carlo Localization [7], and graph-based path planning. Figure 3 depicts some of the student interactions with the platform along with screenshots students created with the erdos library.

During the summer of 2006 we look forward to including additional sensing and algorithmic capabilities to the library, starting with webcam-based vision and FastSLAM [8]. We are seeking partners who may wish to try erdos and the roomba in their curriculum; classroom trials at Harvey Mudd will continue through this academic year.

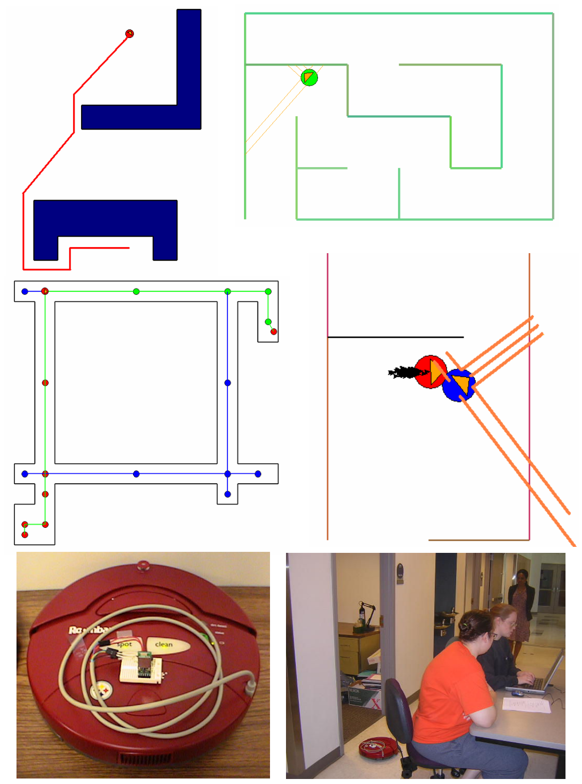

Figure 3 *Clockwise from top left* A simulated roomba running a *bug* algorithm; maze escape via the Pledge algorithm; Monte Carlo Localization with the robot's odometric estimate in blue, simulated range sensors in orange, and a population of 100 particles (black) successfully localizing at the actual robot pose (shown in red); students programming with erdos; a soldering-free, Bluetooth-enabled Roomba; path-planning in a blue graph of waypoints, with planned path in green and visited points in red.

Acknowledgments

The authors gratefully acknowledge support provided by NSF DUE projects #0411176 and #0536173, the Mellon Foundation, Chatham College, and Harvey Mudd College.

References

[1] www.irobot.com
[2] www.irobot.com/sp.cfm?pageid=198
[3] pyrorobotics.org
[4] www.cs.hmc.edu/~dodds/erdos
[5] Abelson, H. and diSessa, A. *Turtle Geometry* MIT Press Cambridge, MA 1980.
[6] Lumelsky, V. J. and Stepanov, A. A. Path planning strategies for a point mobile automaton moving amidst unknown obstacles of arbitrary shape. *Algorithmica*, 2:403-430, 1987.
[7] Fox, D., Burgard, W., Dellaert, F. and Thrun, S. Monte carlo localization. Proceedings AAAI, July 1999.
[8] Montemerlo, M., Thrun, S., Koller, D., and Wegbreit, B. FastSLAM. Proceedings AAAI, July 2002.