Roomba Pac-Man: Teaching Autonomous Robotics through Embodied Gaming

Brendan Charles Dickinson **Odest Chadwicke Jenkins** **Mark Moseley**
David Bloom **Daniel Hartmann**
Department of Computer Science, Brown University, Providence, RI, 02912-1910
{bcd | cjenkins | mmoseley | dbloom | dhartman }@cs.brown.edu

Abstract

We present an approach to teaching autonomous robotics to upper-level undergraduates through the medium of embodied games. As part of a developing course at Brown University, we have created the Roomba Pac-Man task to introduce students to different approaches to autonomous robot control in the context of a specific task. Roomba Pac-Man has been developed using commodity hardware from which students explore standard methods in robotics, namely subsumption, localization, and path planning. Our development of Roomba Pac-Man is founded upon grounding robotics in a compelling and accessible application in a non-contrived real-world environment in a manner than can be reproduced, giving students a sense of ownership.

Introduction

As the field of robotics advances, robotics education must adapt to incorporate both technical developments that become core topics and compelling new challenges of societal-level interest. Undergraduate autonomous robotics curricula have been adept at exposing students to relatively modern topics, such as behavior-based control (Arkin 1998) and Monte Carlo localization (Thrun, Burgard, & Fox 2005). However, the impact of such coursework can be difficult to conceptually translate beyond the academic setting. While Lego Mindstorms and Pyro (Blank *et al.* 2004) are excellent simplified platforms for teaching, the artifacts created with them are not directly applicable to real world applications. Such a setup makes the teaching of robotics easier, but it does not ground robotics in the real-world for the students. On the other end of the spectrum are platforms such as Pioneer robots, which can be prohibitively expensive and are distant from deployment in society.

We believe that Roombas, along with related efforts (Nourbakhsh *et al.* 2006), may fit into a "sweet-spot" between educational platforms (e.g., Lego Mindstorms, Pyro) and highly sophisticated and expensive platforms (e.g., Pioneers). Our method is to explore different approaches to autonomous control with focus on a specific task that is: 1) applicable to real world environments, 2) practically reproducible by students, 3) a compelling task that motivates extra-academic interest reinforces concepts covered in class.

To this end, we have developed the *Roomba Pac-Man* task (RPM) as a central theme in the Brown course CS148

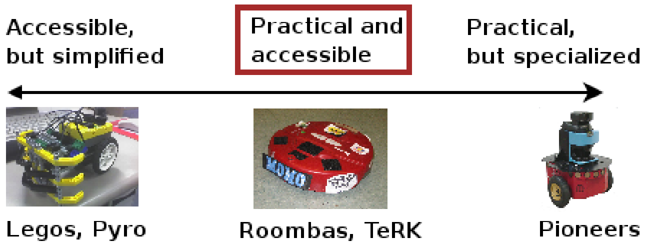


Figure 1: Spectrum of Educational Robots.

("Building Intelligent Robots") for exploring topics in reactive and deliberative robot control. Working from the appeal of video games, RPM is an embodied version of the classic 1980s arcade game *Pac-Man*. In RPM, a virtual Pac-Man is replaced with a physically embodied iRobot Roomba vacuum equipped with a webcam and on-board computing. For two of the four projects in the course, the fifth floor of Brown's Computer Science building becomes a Pac-Man world complete with fiducialized versions of: "food pellets," "ghosts," and "power ups." Demos of these projects consist of student's robotic clients competing for high scores by vacuuming pellets and visiting power-ups within a fixed amount of time. The other two projects focus on developing understanding of methods to improve RPM performance (localization and path-planning). The first three projects cover subsumption (Arkin 1998), localization (Thrun, Burgard, & Fox 2005), and path planning (Choset *et al.* 2005) in the context of RPM, allowing for normalized comparison and appreciation of the relative strengths of the approaches. The final project is a culmination of what has been learned in class, and offers the student opportunity for innovative design.

RPM leverages Roombas as a cost-effective and deployable robot platform. Following the desire for reproducibility, students develop robot controllers for the Player robot server and Gazebo simulation platform (collectively referred to as PSG). The Roombas provide a commercial off-the-shelf platform that requires no proprietary software/hardware and allows emphasis on algorithms rather than hardware engineering.

Course Structure

Brown's CS 148 is geared toward undergraduates drawn mainly from CS, but also Engineering and Cognitive Sciences. All students are expected to have a minimum of two semesters of programming experience. While there are no required texts, recommended readings are drawn from (Thrun, Burgard, & Fox 2005), (Martin 2001), (Choset *et al.* 2005), and (Arkin 1998). The structure of Brown CS148 consists of three introductory labs, three control projects, and a final project, all of which reinforce material presented in lecture. Labs are simple projects designed to give students an introduction to PSG and the Roomba hardware. The labs progress through basic reactive obstacle avoidance, blobfinding, object/fiducial seeking, and color calibration with a physically simulated robot. These labs are designed to be straightforward exercises (implementable within a given lab period) that give the students a chance to familiarize themselves with robotics.

The course projects are designed to explore reactive and deliberative approaches to robot control in the context of RPM. The first project, implementing a reactive subsumption controller to compete in Roomba Pac-Man, integrates all the topics covered previously in the labs. In the second project, students implement Monte-Carlo Localization (MCL) for a simulated Pioneer 2AT in PSG. The third project involves writing a path planner to play RPM, again in simulation, using the estimates from their MCL system from the second project. For final projects, students develop robot clients using their MCL and planning systems developed in previous projects to compete in a final real-world competition.

The projects involve two main deliverables, in-class competition/demonstration of the project and an electronic submission of the work (project write-up, source code, and other materials). The in-class competition involve a demonstration of the students' robot controllers and should show understanding of the specific aspects of the project (statelessness in the subsumption project, localization estimates for MCL for example). In the project write-ups students are asked to scientifically present the results of their implementations. The goal is to not only implement the ideas, but also to explain the concepts back to the course staff to show understanding. Specifically, project write-ups should address the design choices relevant to individual projects and various strengths and weaknesses.

RPM Platform

We aimed for RPM to be cheap, reproducible, and usable in normal environments. For this purpose, the iRobot Roomba was a logical choice, being a cost effective solution with brand familiarity. Because of its popularity in society, students immediately see it not as a toy, but as a robot with real-world applicability. Each Roomba costs $150. Standard Dell Dimension laptops, which cost $500 each, control an individual Roomba. The Roombas are connected to the laptops via a Robo Dynamics RooStick, which can be purchased for $25 each. PSG has basic support for the Roomba Serial Command Interface, which was extended to incorporate more of the Roomba's features (e.g., IR, vacuum, etc.). Finally, Logitech Communicate STX webcams were mounted on the Roombas, costing about $30 each. Thus, for under $700 one can procure, a functional, real world robot with the ability to manipulate objects (e.g., vacuum). As a result of the low cost of the set-up, there were nine Roombas

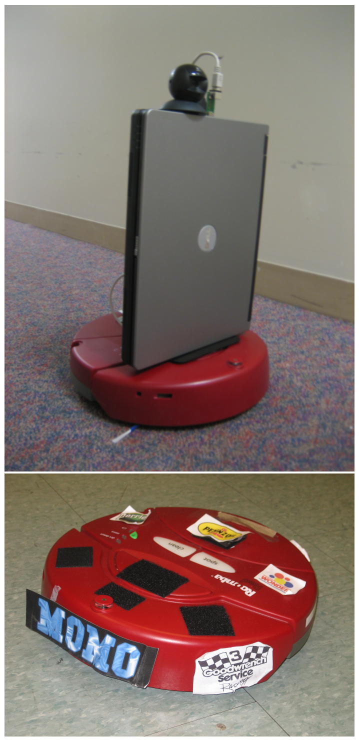


Figure 2: Our Roomba Pac-Man prototype and final student-outfitted RPM.

available to students. An early version of RPM is shown in Figure 2.

While cheap, our infrastructure is far from optimal due to the size and weight of the laptop and the inconsistent nature of webcams. The setup presented in Figure 2 proved infeasible because of the dramatic change in the center of gravity of the Roomba due to the weight of the laptops. The laptops could not lay flat, as they would have extended beyond the radius of the Roomba. The final, non-optimal, solution was to tether the Roombas to laptops that students carried. Other efforts have explored better options such as Gumstix embedded boards[1], but were prohibitively expensive for our purposes. The Roombas were tethered to the laptops via RooSticks rather than via a Bluetooth or wireless connection, because at the time of procurement the RooSticks appeared to be cheaper and easier for the students to implement. In reality, the tethering was awkward, and RooSticks proved to be unreliable (often melting after prolonged use). Bluetooth and wireless options will be explored for the next iteration of RPM. These problems aside, the platform is portable and flexible to the specifics of the robot hardware.

PSG and its Modifications

Our robot software infrastructure is based on PSG (Gerkey *et al.* 2001). Player is a client/server middleware system for transparently interfacing software controllers with robot hardware. Player enables focus on the algorithmic aspects

[1]http://www.ai.sri.com/∼gerkey/roomba/index.html

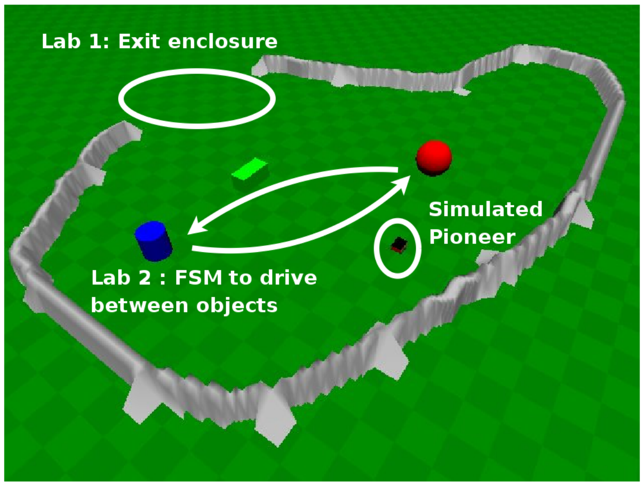


Figure 3: Simulated world in Gazebo for Labs 1 and 2

of robotics by abstracting control and sensory information exchanged between a controller and a robot into *proxies*. Player allows for relatively easy portability between simulation and real-world robot platforms and independence from a particular development language for writing controllers. Player is complemented by Gazebo, a 3D physics-based robot simulator that uses the Open Dynamics Engine (ODE) suitable for smaller numbers of robots simulated at high fidelity.

Several changes had to be made to Player in the development of RPM.The original Player Roomba support allowed only for position control and reading the bump sensor. The ability to read the other sensors on the Roomba was added, this included: the six infrared sensors including the IR-wall detector and the various buttons on top of the Roomba. This ability was added by utilizing the Proxy structure supplied by Player.The ability to control more of the Roomba outputs besides just the wheel motors was also added. This included the ability to control the color and brightness of the LEDs on the Roomba and to turn on and off the vacuum.

Several other modifications to Player were made in relation to the camera. The ability to auto-detect camera parameters was implemented to enable the webcams to function in Player. Modifications were also made to playercam, a PSG utility that streams the camera frames to the screen and overlays the blobfinder results in this image. This program was modified to report a range of YUV values when the user clicked and selected a rectangular region of interest in the image. This change allowed for faster camera calibration by making it easier to prepare the Roombas to play in various lighting conditions.All of these changes have been submitted to the PSG developers, and all of the Roomba driver changes have been incorporated into the latest build of Player.

Labs and Projects

The coursework consists of three labs completed over the course of the first month of the semester. The four projects are larger undertakings, for which the students are given 3-4 weeks to work.

Lab 1: Obstacle Avoidance

The goal of Lab 1 is for students to write a basic feedback controller to exit an enclosure shown in Figure 3. This lab also serves to acquaint the students with the subtleties involved in working with PSG. After a brief tutorial of the Player client library, they are given the task of writing a reactive client for a simulated Pioneer 2AT equipped with a SICK 2000 laser rangefinder.

Lab 2: Object Seeking

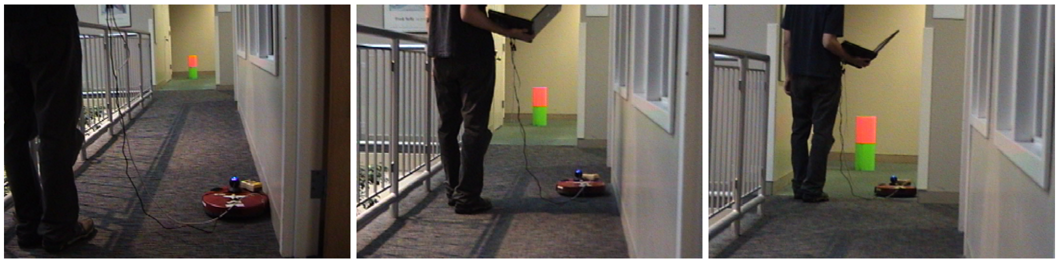
In Lab 2, students become familiar with finite state machines and proportional control through an object seeking task. They extend their control client to build a finite state machine that continually drives between two different fiducialized objects (Figure 3). In the seeking task, their robot must recognize the object of current interest from blobfinding, provided by CMVision (Bruce, Balch, & Veloso 2000) with a simulated Sony VID30 camera. Once recognized, students must use P-servoing to keep the object centered in view while driving toward the object. After reaching the object, the robot changes state and seeks another object. Students also experiment with positioning of the light sources to get a controlled sense of how lighting affects robot perception.

Lab 3 and Project 1: Color Calibration and Reactive Roomba Pac-Man

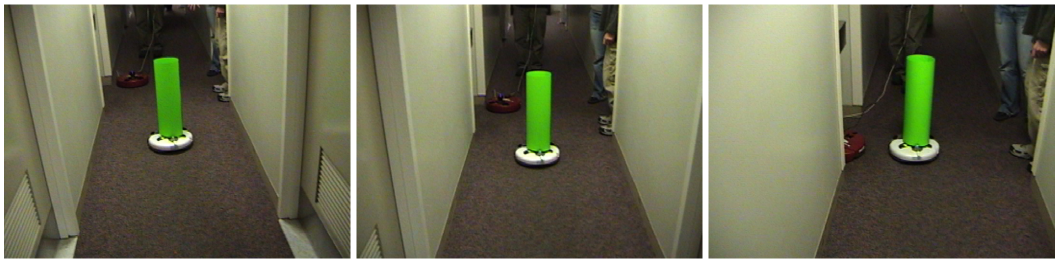
Lab 3 serves as a gateway into the first project, writing a subsumption client for the Roomba Pac-Man task. Chapter 4 from *Behavior Based Robotics* (Arkin 1998) is used as the reading for this project. Lab 3 extends Lab 2's object seeking client to work with a physically embodied Roomba. Using the same Player proxies, the client controls a Roomba endowed with touch/bump, IR, and camera sensing. The camera sensing is accomplished by attaching a web-cam to the Roomba.While the Lab 2 client could theoretically perform on the Roomba without modification, there are issues caused by the uncontrolled nature of the real world that must be addressed. Specifically, the blobfinder must be calibrated to recognize fiducial colors that vary under different lighting conditions, camera sensors, camera viewpoints, etc.

In Project 1, students compete for high scores in a game of Roomba Pac-Man on the fifth floor of Brown's CS department using a reactive subsumptive control policy. Lab 3 prepares students for this project by having them implement the following basic unprioritized functions:

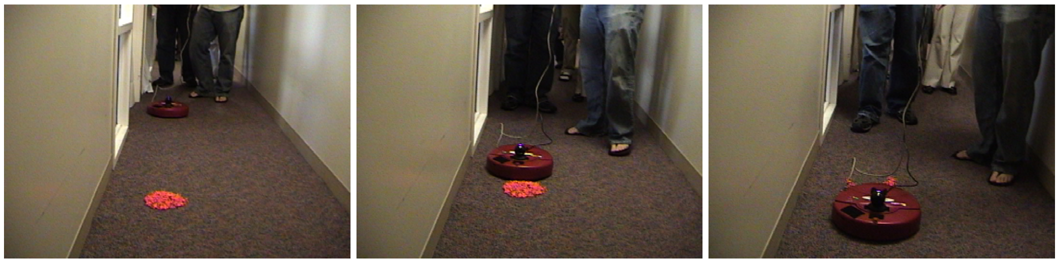
- Fiducial attraction: same as in Lab 2, except the sought cylindrical "Power up" fiducial will be composed of two colors, orange over green.(Figure 4(a))
- Fiducial avoidance: detect and avoid a green cylindrical "Ghost" fiducial by turning away from it. (Figure 4(b))
- Pellet consumption: detect and drive over a pile of orange colored "food pellets" on the floor. (Figure 4(c))
- Wander: wander around an environment to achieve "coverage."



(a)



(b)



(c)

Figure 4: Examples of the robot driving to a fiducial (a), avoiding a ghost (b), and driving to food (c)

- Wall avoidance: detect collisions with physical or virtual walls and move to avoid these contacts.

Project 2: Monte-Carlo Localization

Project 2 involves implementing Monte-Carlo Localization (MCL) with the goal of improving the performance of student's Project 1 by having a localization estimate. Chapter 8 from *Probablistic Robotics* (Thrun, Burgard, & Fox 2005) is used as the reading for this project. Students are given the fifth floor of Brown's Computer Science building as a Gazebo world file (Figure 5). This world file is a recreation of the world in which the students will compete in Deliberative Roomba Pac-Man in the final project. Fiducials of the same color are distributed throughout the world at known locations. Fiducials are used in the world so as to allow the students to write their MCL using a blobfinder. The fiducials have the same color in order to make it impossible to dead reckon off of a single fiducial forcing the students to maintain a probability distribution of hypothesises.

The goal for the student is to implement MCL on a simulated Pioneer 2AT using a blobfinder, bump/touch sensor, ir sensor, and odometry. The fact that the project is implemented in PSG makes it easier to deal with bugs and noise from the real world. For one, the lighting in PSG is constant and controllable. It is reasonable to ensure that the color of the fiducials only occur on fiducials. Furthermore, testing does not involve the set up of a lot of equipment, which means that bugs can be found and fixed expeditiously. Finally, the successful completion of Project 2 allows student to concentrate their full attention to planning in Project 3.

Project 3: Path Planning for the Roomba Pac-Man

Project 3 uses the code developed in Projects 1 and 2 to create an effective deliberative robot control policy for the RPM task. Chapter 7 from *Principles of Robot Motion* (Choset *et al.* 2005) is used as the reading for this project The goal for this project is to create a control policy that uses a model of the world from a known map and state estimation to plan a path and execute it intelligently. While we initially planned to have the students implement this project in the real-world, students had a greater than anticipated amount of difficultly implementing vision-based MCL in Project 2. Thus we elected to allow the students to do Project 3 in simulation.

For this project, the students' clients deliberatively plan and navigate paths in order for the robot to maximize the

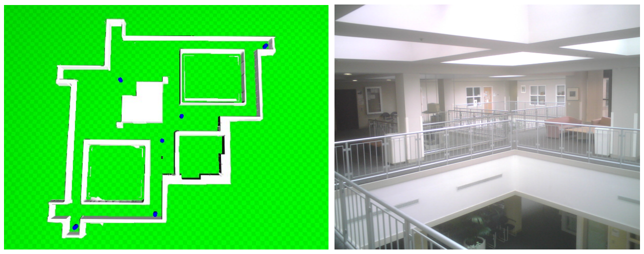


Figure 5: Map and snapshot the Brown CS Department 5th Floor.

number of pellets eaten and power-ups visited while avoiding ghosts in a simulated environment. The clients use the pose estimates given by localization from Project 2 towards path planning. Students then must develop a planning algorithm to deliberatively control their Roomba. Students may choose to implement any of a number of planning algorithms including: Dijkstra's/A* (Cormen *et al.* 1990), potential fields, wavefront planning, and roadmaps. The algorithm, however, must take in to account the presence of ghosts and their random movements throughout the world.

Final Projects: Making Robotics Relevant

CS148 concludes with a final paper and project. The final project is an independently designed competitive RPM controller. Given what the students had learned over the course of the semester, students are tasked with developing the best controller possible for the Roomba Pac-Man task. Collectively, student's clients are evaluated in a tournament-style RPM contest. The final paper is analysis of a fictional robot that discusses its technological feasibility (in terms of perception, decision making, motor control, and platform engineering) and possible means to develop innovations for realizing the robot. This subject is a means by which the student can demostrate their understanding of important course concepts beyound their experiences with implementations. Additionally, the paper should try to answer the following question: "What is the point of robotics?", specifically constructing an argument about most pertinent applications for robotics in society.

Conclusion and Future Work

On the whole, RPM was a success. Students were able to grasp the RPM task and were motivated by the challenge it presented. Particularly successful were Projects 1 and 3. A number of students wrote very effective reactive control policies for the RPM task and embraced the competitive nature of the game. The open-ended nature of the planning project, prompted sincere exploration of different algorithms as well as interesting and well informed class discussion. However, there are a number of issues to address in future instantiations of the course.

For equipment, the tethered connections between the Roomba and laptop was awkward and made the robots feel less "robot-like." We are currently exploring alternatives for the USB to MiniDin serial connection. A larger hardware issue pertained to the STX webcams and their lack of a control to adjust their white-balance. The resulting image washout in the presence of direct natural light greatly limited the environments where RPM could functionally perform.

Robotics poses difficulty for students trained to expect determinism in computer science when faced with real world uncertainty. We found that students either embraced such uncertainty or became frustrated by the imperfect nature of the sensors. We are currently debating whether RPM, as currently designed, may be too difficult. Given the success the course staff (last year's CS148 students) had implementing the projects within a much more limited time frame, RPM should be within student's capabilities. The true problem remains making RPM engaging enough to properly motivate student interest.

We want to establish a stronger connection between human and robot decision making through embodied gaming. We are implementing an off-board, wireless tele-operation client to replace the very basic *playerv* that will allow a person to play RPM. Ideally, the tele-operator would observe only the perceptual features used by the robot (color blobs, IR, bump, and odometry). Such tighter human-robot interaction would help motivate the difficultly of developing robot control policies, provide students a baseline for their work and enable a greater competitive spirit.

References

Arkin, R. C. 1998. *Behavior-Based Robotics*. Cambridge, Massachusetts, USA: MIT Press.

Blank, D.; Kumar, D.; Meeden, L.; and Yanco, H. 2004. Pyro: A python-based versatile programming environment for teaching robotics. *Journal of Educational Resources in Computing*.

Bruce, J.; Balch, T.; and Veloso, M. 2000. Fast and inexpensive color image segmentation for interactive robots. In *In Proceedings of the 2000 IEEE/RSJ International Conference on Intelligent Robots and Systems*, volume 3.

Choset, H.; Lynch, K. M.; Hutchinson, S.; Kantor, G.; Burgard, W.; Kavraki, L. E.; and Thrun, S. 2005. *Principles of Robot Motion: Theory, Algorithms, and Implementations*. MIT Press.

Cormen, T.; Leiserson, C.; Rivest, R.; and Stein, C. 1990. *Introduction to Algorithms*. MIT Press.

Gerkey, B.; Vaughan, R.; Stoy, K.; Howard, A.; Sukhatme, G.; and Mataric, M. 2001. Most valuable player: A robot device server for distributed control. In *Proceedings of 2001 IEEE/RSJ International Conference on Intelligent Robots and Systems*, 1226–1231.

Martin, F. 2001. *Robotic Explorations: A Hands-On Introduction to Engineering*. Prentice-Hall.

Nourbakhsh, I.; Hamner, E.; Lauwers, T.; Bernstein, D.; and Disalvo, C. 2006. A roadmap for technology literacy and a vehicle for getting there: Educational robotics and the terk project. In *IEEE RO-MAN*.

Thrun, S.; Burgard, W.; and Fox, D. 2005. *Probabilistic Robotics*. MIT Press.