AI Magazine Volume 27 Number 1 (2006) (© AAAI)

CMRoboBits: Creating an Intelligent AIBO Robot

Manuela M. Veloso, Paul E. Rybski, Scott Lenser, Sonia Chernova, and Douglas Vail

■ CMRoboBits is a course offered at Carnegie Mellon University that introduces students to all the concepts needed to create a complete intelligent robot. In particular, the course focuses on the areas of perception, cognition, and action by using the Sony AIBO robot as the focus for the programming assignments. This course shows how an AIBO and its software resources make it possible for students to investigate and work with an unusually broad variety of AI topics within a single semester. While material presented in this article describes using AIBOs as the primary platform, the concepts presented in the course are not unique to the AIBO and can be applied on different kinds of robotic hardware.

Since 1997, we have researched teams of soccer robots using the Sony AIBO ERS-210 robots as the robot platform (Veloso and Uther 1999; Veloso et al. 2000; Lenser, Bruce, and Veloso 2001a; Lenser, Bruce, and Veloso 2001b; Uther et al. 2002). Our experience runs across several generations of these four-legged robots, and we have met with increasing success every year. In the fall of 2003, we created a new course building upon our research experience with the AIBO robots. We have since refined the course and taught it again in 2004. The course, which we entitled CMRoboBits: Creating an Intelligent AIBO Robot, introduces students to all the concepts needed to create a complete intelligent robot. We focus on the areas of perception, cognition, and action (illustrated in figure 1) and use the Sony AIBO robots to help the students understand in depth the issues involved in developing such capabilities in a robot.

The course has a two-hour weekly lecture and a one-hour weekly lab session. The coursework consists of weekly homework assignments and a larger final project. The homework assignments include written questions about the underlying concepts and algorithms as well as programming tasks for the students to implement on the AIBO robots. Evaluation is based on the students' written answers, as well as their level of accomplishment on the programming tasks. All course materials, including student solutions to assignments, are made available on the web.

Our goal is for our course materials to be used by other universities in their robotics and AI courses. In this article, we present the list of topics that were covered in the lectures and include examples of homework assignments as well as the rationale behind them. This curriculum shows how the AIBO and its software resources make it possible for students to investigate and work with an unusually broad variety of AI topics within a one-semester course.

The Sony AIBO is a commercially available robot that is relatively inexpensive as research robots go (on the order of US$2,000). While purchasing large quantities of these robots may

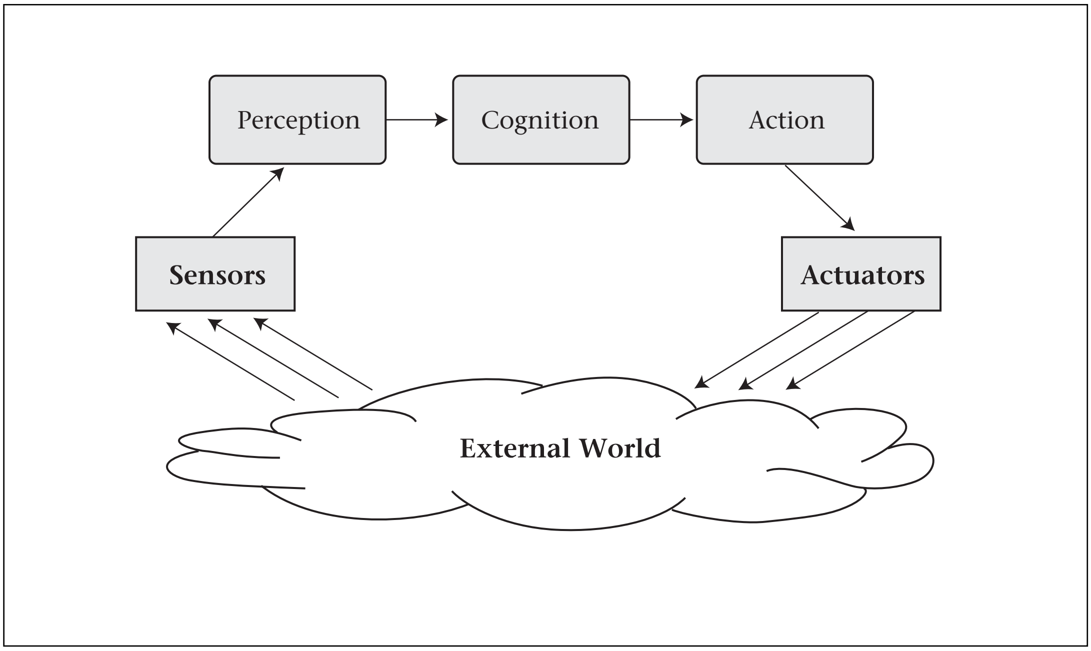

Figure 1. The Modules Used in the Complete Robot.

be beyond the available budget for smaller institutions, the robots could be purchased in small quantities for use in special topics courses and independent research projects. Additionally, commercially available pedagogical electronics and robotics kits from other companies are rapidly approaching the CPU performance capabilities of the AIBOs. Coupled with inexpensive cameras and actuators, such kits could very easily be used in a complementary fashion to the courseware described in this article.

The Goals of the Course and the Schedule

The main goal of the course is to learn how to create an intelligent robot, using the AIBO platform as a concrete example. We want the students to understand how to program the robots to perform tasks. We view a robot as a complete intelligent agent, in the sense that it includes perception, cognition, and action. Our aim through the course is to demystify robot programming so that it becomes clear and accessible to all of our students.

A parallel goal of particular interest to us is to bridge from our research code in robot soccer to modular code that can be used for any general robot task. We aim to provide course materials that are modular and well structured so that instructors at other universities can use the materials in their own courses. We have found that reorganizing and cleaning up our robot soccer code has had several additional positive effects, namely facilitating our research work and easing students' transition into that research.

The AIBO, shown in figure 2, is a remarkable piece of commercially available robotic hardware. An AIBO has 15 degrees of freedom (DOF) in its legs and head, a color CCD camera that can process images at 25–30 frames/second, a three-axis accelerometer for body pose estimation, buttons on its back, head, and footpads, LEDs for visual debugging, and a wireless Ethernet (802.11b) card for interrobot and host-computer communication. Students program AIBOs using a free SDK called OPEN-R,[1] which lets them compile control code on a workstation with a MIPS cross-compiler (available for GNU Linux, Microsoft Windows, and Mac OS X). The AIBO's low cost allows an instructor to purchase several of them for the price of a more traditional research robotic platform.

This 15-week course contains several main components, as we present below. In the first version of the course in the fall of 2003, we followed these main components. In the second version in the fall of 2004, we put significantly more emphasis on behaviors, multirobot cooperation, and learning. Although all the components are relevant to the goals of an AI/robotics course like ours, it is easy to get lost in the low-level details of perception and action, for which there may be other courses specifically addressing these topics. Instead the focus of our course is the complete intelligent agent, hence the em-

Figure 2. Annotated AIBO ERS-210.

phasis on cognition and learning. We always teach perception and action, but we balance the topics toward cognition. The following descriptions outline how we strike this balance.

Behaviors

This course familiarizes the students with the concept of behaviors for robot control. Every component, from sensors to localization, is cast in the framework of how a mobile robot can use those techniques in its behaviors. We reintroduce behaviors at several times in the course, since behaviors are the basic components of virtually any robot task. Initially, we introduce finite-state machines and incrementally address more complex behavioral structures, such as hierarchical behaviors and planning. Figure 3 depicts a decomposition of several parallel and sequential behaviors used in class.

Sensors and Actuators

Robots perceive the world using their sensors, and they affect their environment with their actuators. Sensors and actuators mediate all interactions between the robot and its environment and are equivalent to input and output operators in computer programming. This component of the course introduces students to the idea of acting in the face of uncertainty. Unlike traditional programming where input values are completely known, robots must perform with only limited, noisy knowledge of their environment. Additionally, robots must cope with noise and uncertainty in the effects of their actions; motors do not always perform the requested movements, and factors such as friction and slip are difficult to take into account when predicting the outcome of actions. Students are introduced to the idea of uncertainty, which is central to robot programming. Figure 4 shows an example plot of the three-axis accelerometer data that students can use to determine what “state” the robot is in.

Motion

The AIBO robots offer an interesting and challenging platform for exploring robot motion (figure 5). AIBOs differ from most educational

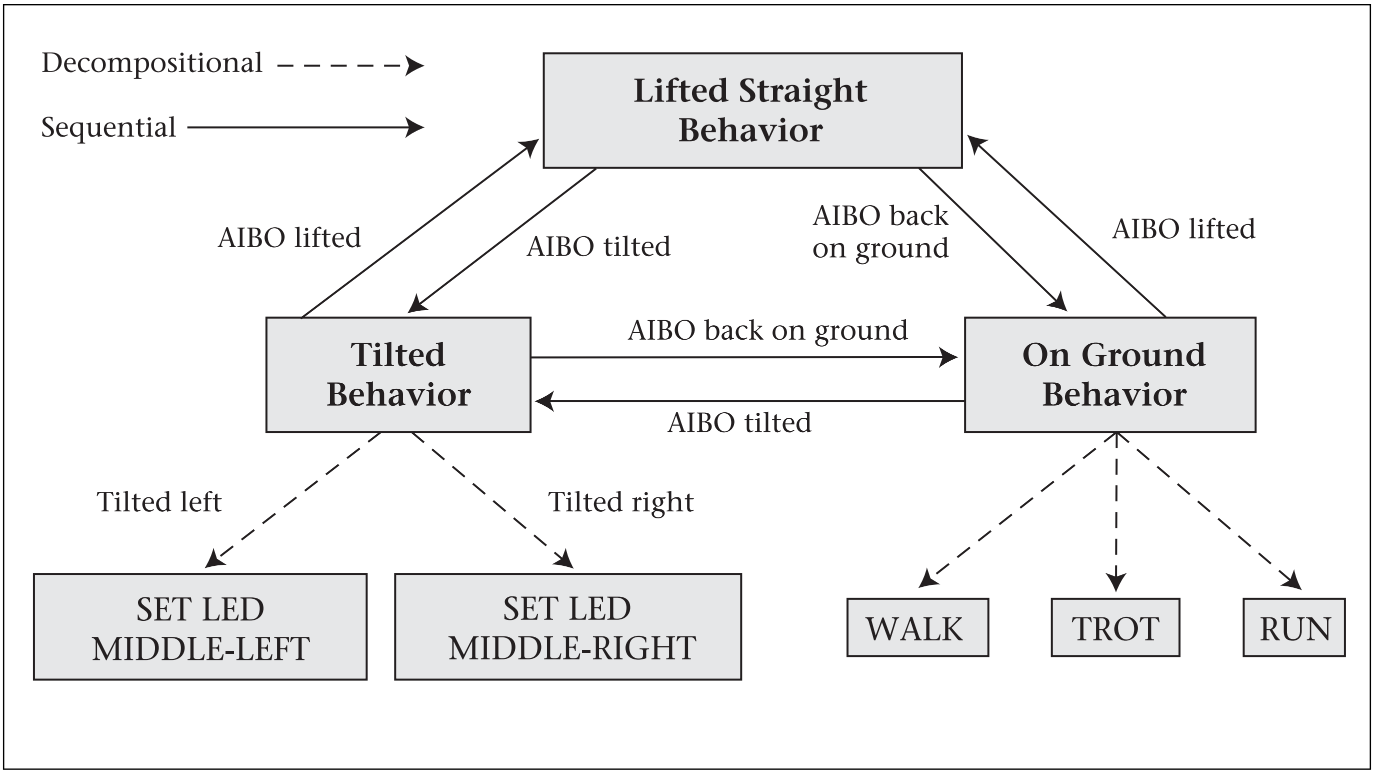

Figure 3. Description of a Behavior Hierarchy

robots because they are a legged platform with 15 degrees of freedom (DOF) in their head and legs. Each of the four legs has three DOF, and the head has pan, tilt, and roll joints. In addition to these major joints, students can actuate the tail, mouth, ears, and eye LEDs to create more expressive behaviors. In this unit, we introduce students to the ideas of forward and inverse kinematics. We also include a practical introduction to our motion system on the AIBO. We describe our parameterized walk engine, which uses approximately 50 numeric parameters to specify an entire gait for the robot. These parameters include factors such as robot body height, body angle, lift heights for each leg, and timings. We also introduce students to the idea of frame-based motion where all joint angles are specified for a few key frames and the robot interpolates between them. This type of motion is useful for scripting kicking motions for soccer, dance motions, climbing, and other predefined motions.

Vision

The AIBO robots use vision as their primary sensor. Color images in the YUV color space arrive at a frame rate of 25Hz. The vision unit of the course acquaints students with the basics of robot visual processing. Students briefly learn about the YUV color space commonly used by image capture hardware, as well as real time color segmentation, and camera calibration. The final topics include higher-level concepts such as object recognition from the color-segmented images, including weeding out false positives (figure 6). Students also learn how kinematics ties back to vision for calculating the real world position of objects in the vision frames. Figure 6 shows an example frame of postprocessed AIBO video. This image illustrates the output of the AIBO's real-time color-segmentation algorithm.

Localization

In order to act effectively, a robot often needs to know its location in the environment. Localization becomes an essential component that interacts with perception, decision making, and motion. This unit introduces the ideas of probabilistic localization beginning with the basic ideas of probabilistic localization and including different methods of representing locale belief, such as Kalman filters and particle filters. We also cover more advanced localization challenges, such as recovering from errors in motion models (for example, the kidnapped robot problem) through sensor-based resampling algorithms, and study various trade-offs that may be made between computational cost and resource consumption.

Multirobot Cooperation

Once the students understand how to program a single AIBO to do interesting behaviors, we teach them how to use the AIBO's onboard 802.11b wireless Ethernet system (figure 7). This feature allows the robots to communicate among themselves. We teach the students how to solve problems with multirobot behaviors, discuss the challenges, and present several approaches for multirobot communication and coordination, including market-based approaches, and cognitive architectures, such as the skills-tactics-plays architecture.

Learning

Throughout the course we explain how robot programming involves a considerable amount of parameter tuning in a variety of algorithms. We introduce learning approaches for motion optimization, control learning, and team play adaptation to an opponent team. Learning is a theme that we have taught in the lectures, but students do not currently investigate it deeply in their programming assignments. In future iterations of the course, we plan to make learning a more significant component of students' robot programming assignments.

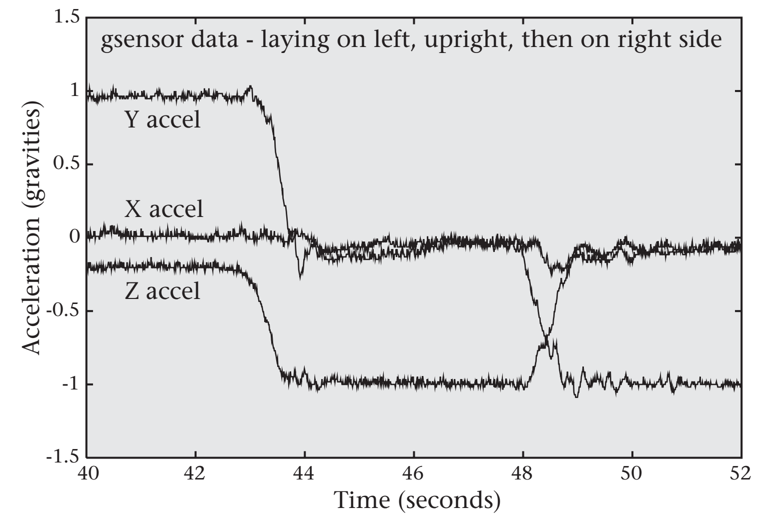

Figure 4. Three-Axis Accelerometer Signature for a Robot That Starts on Its Left Side, Is Rotated to an Upright Position, and Then Is Rotated to Its Right Side.

CMRoboBits

The CMRoboBits code distribution provides all the software necessary for complete control of an AIBO. The default code distribution contains all of the functionality necessary to run simple behaviors (such as chasing a ball). Students are able to use the existing codebase for their homework assignments and typically write behaviors that make use of the preexisting systems. For homework assignments that expand on a particular aspect of the robot's capabilities (such as vision homework assignments), the students are expected to augment the existing CMRoboBits code with the new functionality.

The software includes a number of specific capabilities, such as a behavior engine for specifying complex robot actions. Behaviors can run in parallel in a priority-based scheme, and each behavior can invoke a tree of subbehaviors. It also includes a real-time low-level color-segmentation algorithm for identifying objects by specific color types. This also includes the tools necessary for learning the segmentation based on labeled training images. Other capabilities include (1) a high-level vision object-recognition system for identifying and returning the three-dimensional positions of specific objects such as colored markers, balls, and even other AIBOs; (2) a frame-based inverse kinematics engine for performing complex motions with the AIBO's limbs; (3) a parameterized walk engine for specifying different trot gaits that the AIBO can use to translate and rotate in any direction; (4) a Monte-Carlo localization system (particle filter) using landmarks at known positions to track the robot's pose and orientation; (5) a world modeling system for tracking the positions of detected objects over time; (6) an interrobot communication system for sharing world state with other robots; and (7) stream-based debugging tools for viewing debug text strings and frames of color-thresholded images in real time or from log files.

Homework Assignments

In this section, we briefly describe the rationale, requirements, and grading of the course homework assignments. Students are given one week to complete each assignment. We suggest that the students work in groups of two or three, taking into account the number of students relative to the number of robots and the weekly homework timing. Assignments are due at the beginning of the lab period each week. This routine allows either a demonstration session at the beginning of the lab or a review session where the teaching assistant reviews students' code and watches the robots to diagnose problems. Helping students debug problems requires the availability of both the robots and source code.

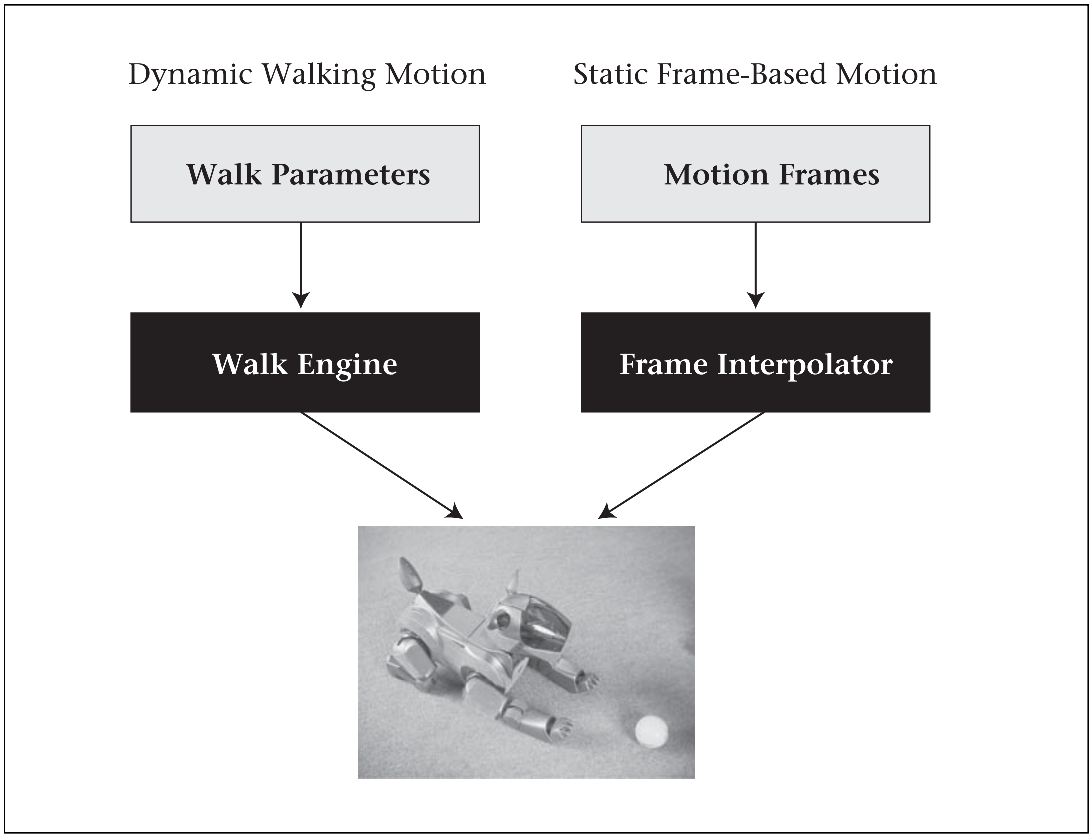

Figure 5. The Two Motion Control Systems in the CMRoboBits Code Base.

All the homework assignments in the course include several written questions that cover the lecture material as well as the robot programming assignment. This homework structure allows us to address both theory and practice. We discuss the programming part of the homework assignments, while the complete questions can be viewed online at our course web site.

Introductory Homework Assignments

We designed a set of initial homework assignments that successfully familiarizes the students with the robot's software development environment, core hardware, and simple motion behaviors.

Development. Our first homework serves as an introduction to the development environment and brings students up to speed on how to access the source code from our repository, compile the code using the OPENR SDK (freely available from Sony), and copy the final programs to memory sticks for use with an AIBO. This first homework assignment teaches the students how to operate the robot's software infrastructure. The homework also shows students how to select which behavior runs using our framework and allows us to test code submissions using a dropbox system. This simple first assignment allows us to assess how quickly the students familiarize themselves with the software and robots. We then tune the other homework assignments accordingly.

Basic Sensors. We follow with a homework to familiarize the students with the sensors on the robot. The background section covers how to subscribe to sensor messages, specifically, data from the robot's accelerometer and the touch sensors on its feet. The homework sets up a series of specific sensor-actuator behaviors that the robot needs to perform exactly. In particular, LEDs need to be set on the robot's face in different situations, including when the robot has a foot in contact with the ground and when the robot is lifted off the floor. LEDs also need to be set to display whether the robot is leveled, tilted toward its left side, or tilted to its right.

This assignment gives the students practical experience with the sense-think-act loop. They must read (noisy) sensor data from the robot, determine which actions to take based on this sensor data, and finally send commands to the robot to perform these actions. This sequence is repeated with a frequency of 25 Hz on the robot (the frame rate of the camera). The homework focuses exclusively on the basic sensors and does not require use of the vision-processing system.

Robot Motion Homework

Robot motion involves a substantial amount of trial and error. In this motion homework, students learn how to build up to a complete motion. We provide our walk engine, which implements the needed forward and inverse kinematics and sets up a parameter-based interface for motion variations.

The assignment is broken down into two parts. In the first part, students create a set of walk parameters to describe a gait. The walk engine uses a set of 51 parameters to generate the actual trajectory that the end of each foot follows over the course of a single step. The parameters include limits on how high each foot can rise above the ground, the desired angle of the robot's body, and other similar factors. Finding an effective walk is an optimization in this 51-dimensional parameter space. The walk engine parameters are not completely independent, which makes finding an optimal set of parameters extremely difficult. Typically we optimize for speed and stability, although other factors such as a walk with a high body height are possible. We challenge the students to develop a learning approach, as extra credit, as an alternative to the incremental, trial and error experimentation for parameter setting. While this extra credit offers a natural connection with machine learning, we were surprised to find an interesting secondary benefit to hand-tuning gait parameters. Doing so greatly increased students' understanding of and familiarity with the robots.

The homework also tests frame-based motion. The students are to create a new motion using our key-frame animation–based approach. We ask for specific motions, such as a motion that makes the robot perform a complete rollover and then climb back onto its feet. The students learn how to convert between the positions of the robot's limbs in space and the corresponding angles of the robot's joints in their own coordinate frame. Since rolling over is a dynamic activity that depends on building up momentum and moving different legs in concert, the students also learn how to coordinate different joints simultaneously. Success with the frame-based portion of the assignment requires an incremental, experimentation-based approach.

Vision Homework Assignments

The purpose of our course is to teach the students the knowledge that they need to program a complete intelligent robot. We focus on the high-level vision perception and behaviors, but working with real robotic hardware also includes calibrating sensors for use in specified environments. Our homework assignments address these low-level and high-level vision-processing challenges.

Calibrating Vision. The CMRoboBits vision

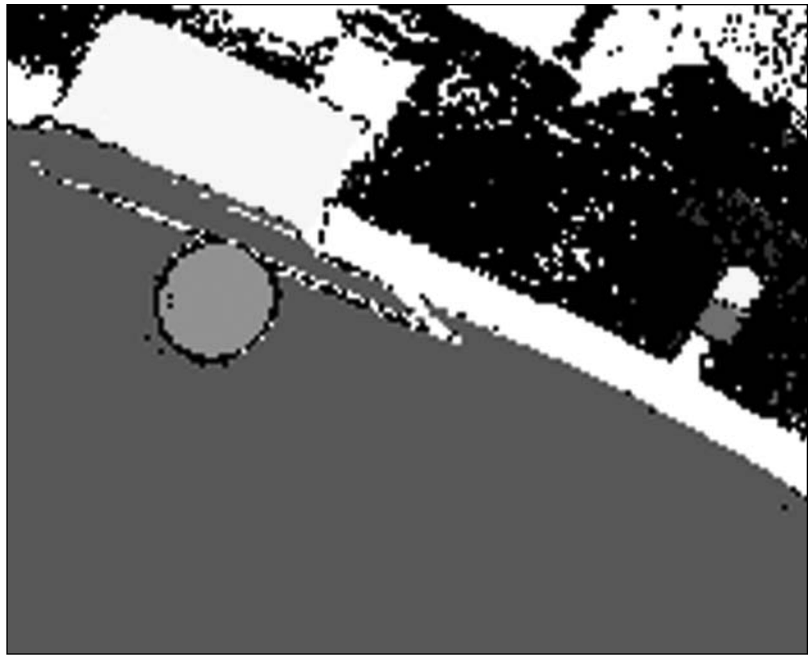

Figure 6. An Example Frame of Video from the AIBO's Camera after Color-Segmentation Postprocessing.

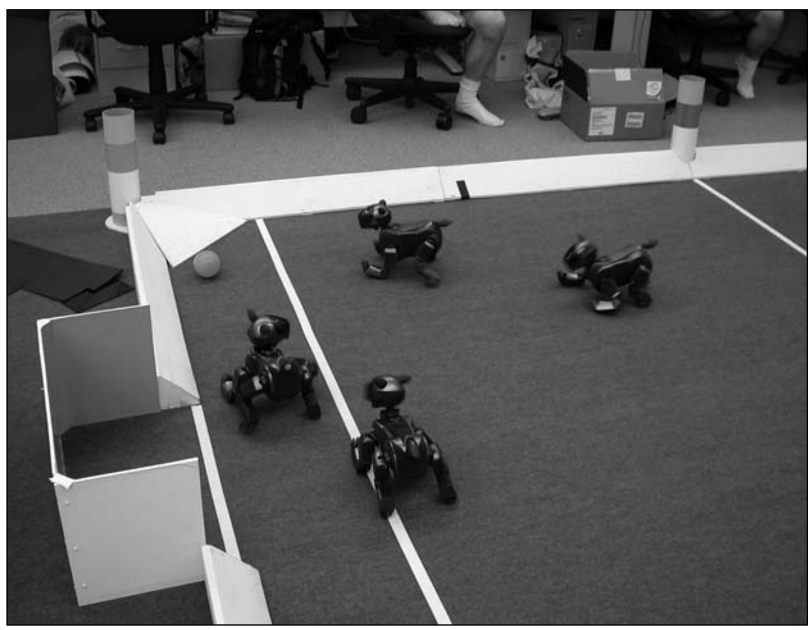

Figure 7. Students Learn Some of the Challenges with Programming Behaviors for Cooperative Multirobot Tasks.

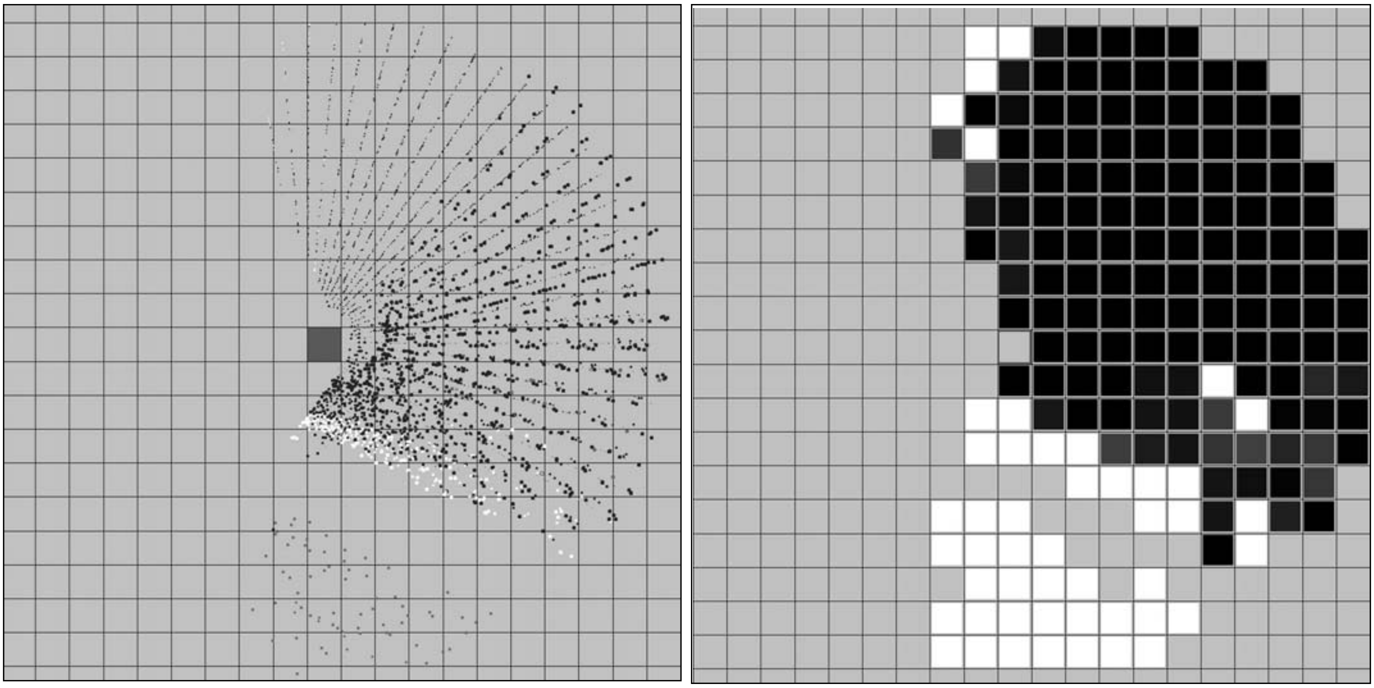

Figure 8. The Egocentric Local Obstacle Model Where the Robot Is in the Center of the Grid.

Left. Obstacles and freespace are represented as samples from the samples; black = freespace, white = obstacles.

Right. Occupancy grid generated from samples; black = freespace, white = obstacles.

Scanlines from the visual perceptual system are parsed for colors that are freespace and obstacles (according to the color-segmentation algorithm). The scanlines are sampled and a collection of points is added to the local model's database, as shown in the figure on the left. These points have a finite lifetime (approximately two seconds) before being forgotten. These points can be turned into a more traditional grid-based occupancy grid by summing the contribution of each of the freespace and obstacle points in that grid.

system uses a color-segmentation algorithm to semantically label pixel colors with specific meanings (for example, orange is a ball and green is a playing field). These labels are stored in a lookup table that assigns labels to every pixel in the full YUV color space. The students must train the lookup table from captured image data.

In this assignment, students use the robot's camera to capture still images from the environment and then save those images to a log file that is transferred to a workstation. The students use standard image-editing software to assign semantic labels to each of the colors in the images. As an example, they assign the ball label to the parts of the image that include a ball. These labeled images serve as training data for a supervised learning algorithm that learns a mapping between YUV color values and symbolic color values. One important aspect of this assignment is to give students an appreciation of how much work is required to obtain a usable vision calibration.

Object Recognition. Once students understand low-level vision concepts such as color segmentation, they must program the AIBO to perform higher-level tasks such as object recognition. Students program their AIBOs to detect a bright orange ball, a colored bull's-eye, a small scooter, a two-color sign, and a tower built from colored cylinders using color-segmented images. We chose the objects so that the students would have to face many of the challenges caused by ambiguous vision information. For instance, the wheels of the scooter are the same shade of orange as the ball, so the students must consider shape and relative size. The students have to recognize the orientation of the two-colored sign (vertical, horizontal, and diagonal), so they have to consider what the image looks like from various camera angles.

Single-Robot Behavior Homework Assignments

In the third set of homework assignments we focus on single-robot behaviors. The main goal is to teach the students how to connect perception, cognition, and action in more complex tasks. The homework assignments require the students to process perceptual data, effectively select actions, and move in different environments. The students build upon their skills from previous homework assignments to solve these problems.

Maze Traversal. In this maze-traversal assignment, students use a (provided) egocentric spatial world model to track regions of free space around the robot. Figure 8 illustrates data captured from the egocentric spatial world model. Students create a behavior to traverse a convoluted mazelike path. The students have to consider how to aim the camera on the robot's head to ensure that the local spatial model contained accurate and up-to-date information at all times. The path is not a true maze, as it has no dead ends, but the robots have to navigate through several turns without touching the walls.

Localization. Students create a behavior that allows the robots to avoid a large square in the center of a carpet. The robot starts in a home position and calculates (and memorizes) its position. The robot is manually picked up and moved to a new position (typically on the other side of the carpet) and replaced on the carpet. The robot has to identify that it has been moved so it can return to its original position. While returning, the robot has to avoid the square in the center of the carpet by using only its localization. The robot uses six colored markers placed around the carpet for localization. Students have to reason about how to control the head to seek out and fixate on these markers. Without accurate visual measurements of the landmarks, the robot cannot localize properly. Localization requires more formal mathematical background than the rest of the material in the course. For the written component of this homework, students have to manually calculate a posterior probability of the robot's position given a uniform prior distribution of robot poses in a grid world.

Mounting a Charging Station. Students use the object-detection code written in previous homework assignments to find a colored bull's-eye and tower beacon. These two landmarks allow the robot to compute the distance and orientation of a charging station. The robot needs to search for and then climb onto the charging station. Once on the charging station, the robot has to shut itself off so it can recharge.

This assignment requires the students to use much of the material that they learned in previous assignments and tie it together into a unified whole. Besides having to use vision to find the landmarks, the students have to create a custom walk that lifts the body high enough to step onto the station. Students also need to create custom motions to move the robots into position over the chargers and settle themselves onto the contacts.

Initialization

Moderator randomly determines a sequence of length n drawn from a set of m possible colors (with repeated colors allowed)

Play

Player guesses a sequence

Moderator tells player how many colors are:
- Correct and in the right position
- Correct but in the wrong position

Continue until the player has deduced and guessed the hidden sequence from the m^n initial possibilities

The original game rules and an on-line playable version is available here:
http://www.kongtechnology.com/index.asp?im=mastermind

Figure 9. Rules for the Game of Mastermind

Multirobot Behavior Homework Assignments

One of the major challenges of creating intelligent robots is to have them coordinate with other robots. We focus a major part of the course on multirobot behaviors and therefore also develop a corresponding set of homework assignments.

Mastermind. This homework introduces students to the concept of human/robot interaction and learning on a real robot. The students program their AIBOs to play a guessing game by which one player (either the human or the AIBO) guesses a sequence of colored markers that the other player (AIBO or human, respectively) makes up ahead of time. The AIBO communicates to the human by a predefined set of motions. When guessing the colored sequence, the AIBO has to reason about the patterns of the colors as well as about the clues given to it by the human. This homework connects more traditional AI techniques (game playing and state space search) with the challenges of human/robot interactions. This homework is described in greater detail in the next section.

Multirobot Ball Collection. Students write multirobot behaviors that make use of interrobot communication. The students program a pair of AIBOs to cooperatively push colored balls into a goal. There are two components to this homework, whereby the students first solve the task by not having the AIBOs communicate, but rather treat each one independently, and then solve the task where they can use communication (802.11b wireless Ethernet) to coordinate the actions of the robots. After completing the behaviors, the students need to analyze the performance of both the noncommunicating and communicating cases and to report on any differences in performance.

Example Assignment: Mastermind

This section provides the text for an actual homework from the fall 2004 CMRoboBits class. This homework gives the students an opportunity to work with vision and motions and requires them to reason about complex behaviors that involve interacting and communicating with a human in a deterministic fashion.

In this homework, the students program the AIBOs to play the game of Mastermind with a human and do so with either the robot as player and the human as moderator or vice versa. Mastermind is a guessing game in which the player must guess an ordered sequence of colors defined in secret by a moderator. The moderator provides the player with simple but not necessarily decisive clues that the player must use to infer the correct colored sequence. Figure 9 describes the rules. The following homework description mentions a program by the name of chokechain, which is a Linux-based console debugging tool that connects to an AIBO over wireless TCP/IP. Students use this program to view debug statements (referred to in the homework as pprintf), as well as color-thresholded frames of AIBO video.

Introduction

This homework is geared towards switching between different behaviors (state machines in state machines), contains an opportunity to experiment with learning, and requires some augmentation to the robot's vision. In this homework, students play the game Mastermind with the AIBOs.

Game Setup

The students play a simplified version of the game with colored squares with only two positions and three colors (repeats allowed) to choose from. The moderator picks a sequence while the player uses the responses from the moderator to guess the sequence. Students interact with their AIBOs through simple motions, buttons, and LEDs.

Part One: AIBO as Moderator

In the first part, the AIBO generates a random sequence of two colors that the student must guess. The required steps for play are as follows:

1. The student presses the back button to unpause the robot.

2. The AIBO starts in the play position, in this case a resting position on its belly.

3. The student presses the button under its chin to start the game. The AIBO selects a random sequence and blinks both middle LEDs twice to indicate the start of game play. The AIBO will also transmit the color sequence over the wireless network for debugging purposes.

4. Students choose two colored squares and place them in front of the AIBO at a specified distance.

5. The students indicate that they are ready for the AIBO to evaluate their choice by pressing the back head button.

6. If all goes well, the AIBO will see the two balls and respond by (a) lighting *n* green LEDs to illustrate the number of squares that are of the correct color and in the right place (corresponding to the black pegs in Mastermind), (b) showing *m* red LEDs to illustrate the number of balls that are of the correct color but in the wrong place (corresponding to the white pegs in Mastermind), and (c) nodding the head to signify that its evaluation has completed.

7. The game continues with the students placing squares and requesting a response with the AIBO's back head button until the right sequence is observed.

8. The AIBO rolls over when it observes the correct sequence.

Part Two: AIBO as Player

In this section, the AIBO takes the role of player. Students must implement the logic for Mastermind formally in the AIBOs. This requires keeping some history of what the AIBO has seen and the responses it received. The required steps for play are as follows:

1. Students press the back button to unpause the robot.

2. Students pick a random sequence of colors and press the button under its chin to start the play sequence.

3. Students retrieve the AIBO's guess by showing it different colored squares (one at a time) and eliciting a position response for each square. Specifically, for each square of color $C = C1, C2, C3$, if the AIBO wants to use square C as part of its guess, it will nod "yes" (up and down) and raise its right or left paw to indicate whether it wants the square in the left or right

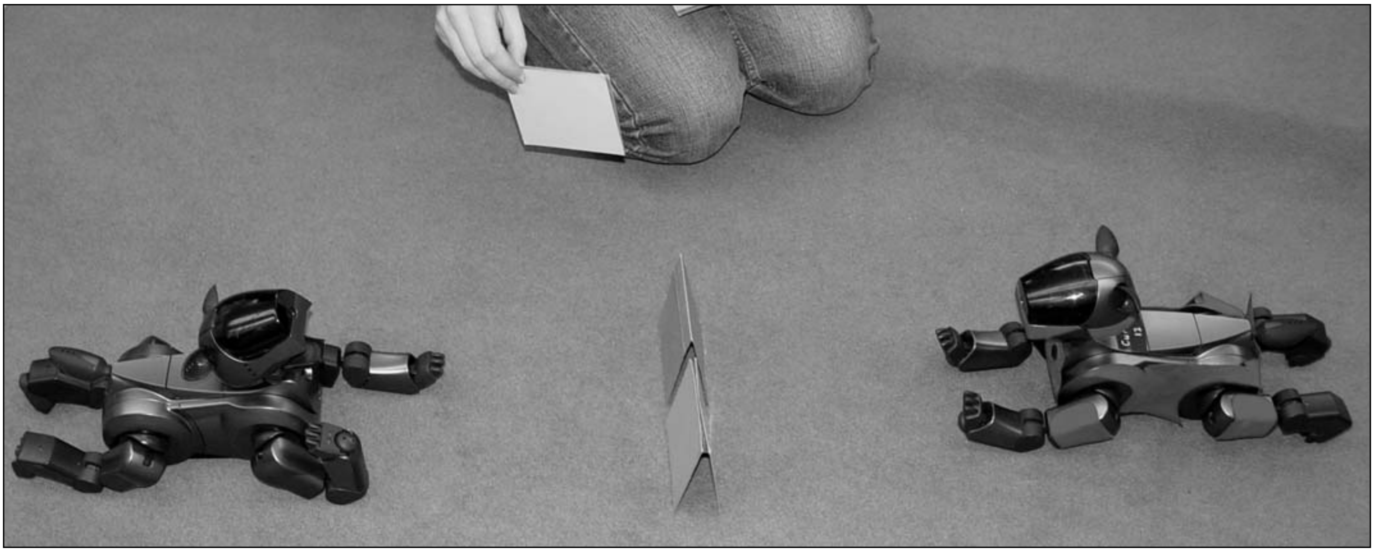

Figure 10. Mastermind: The AIBO on the Left Is the Moderator and the AIBO on the Right Is the Player.

position. If it wants the color in both slots, it will raise both paws. If it does not want the color, it will shake its head "no" (left and right).

4. When both colored squares have been selected by the AIBO, the students give feedback to the AIBO by (a) pressing the rear button on its head to tell it to "start input;" (b) pressing its left front footpad *n* times where *n* is the number of squares of the correct color in the correct position; (c) pressing its right front footpad *m* times where *m* is the number of squares of the correct color in the wrong position; and (d) pressing the rear button on its head to tell it to "stop input."

5. Students repeat this sequence of square guesses and input until the AIBO correctly guesses the sequence. Then, they press the button on the AIBO's back to tell it that it has guessed correctly. As a final motion, the AIBO should roll over or do some cute thing.

Questions

In addition to the programming component, the students are expected to answer two written questions regarding how they might generalize the game rules to an arbitrary number of colors and slots.

Question 1: Using candidate elimination by enumerating all the values can become ridiculous for large problem spaces. For arbitrary *m* and *n,* how can one perform candidate elimination concisely?

Question 2: With large problems, it is desirable to guess those hypotheses that produce the most information and thereby reduce the total number of guesses to win. Explain how to choose a hypothesis from a remaining set.

The human manipulates the colored squares for the player. In this picture, the player notifies the human that it wants to place the colored square on the left.

Class Final Project

Students in the 2003 and 2004 CMRoboBits courses completed a final project worth 30 percent of their overall grade. In 2003, the students proposed a project that demonstrated some interesting behavior of the AIBOs, whereas in 2004, the students all completed the same project. We feel that both approaches have specific advantages. The open-ended style of the 2003 project encourages creativity and allows students to explore what they learn and enjoy most in the course, while the fixed style of the 2004 project tests the students' complete knowledge of the course materials.

Final Project, 2003

In this project style, the students turn in a written proposal of their project and then are coached by the professor to ensure that they can complete their project by the end of the course. Because this style of project is so open-ended, we feel that this encourages the students to express their creativity and do a project that is interesting to them. Several interesting project ideas are described in the following paragraphs and illustrated in figures 11–15.

Obstacle Course. An AIBO is programmed to navigate an obstacle course that consists of an archway, a straight corridor, a tunnel, and a step. The robot is programmed to identify each obstacle by viewing a colored strip on the

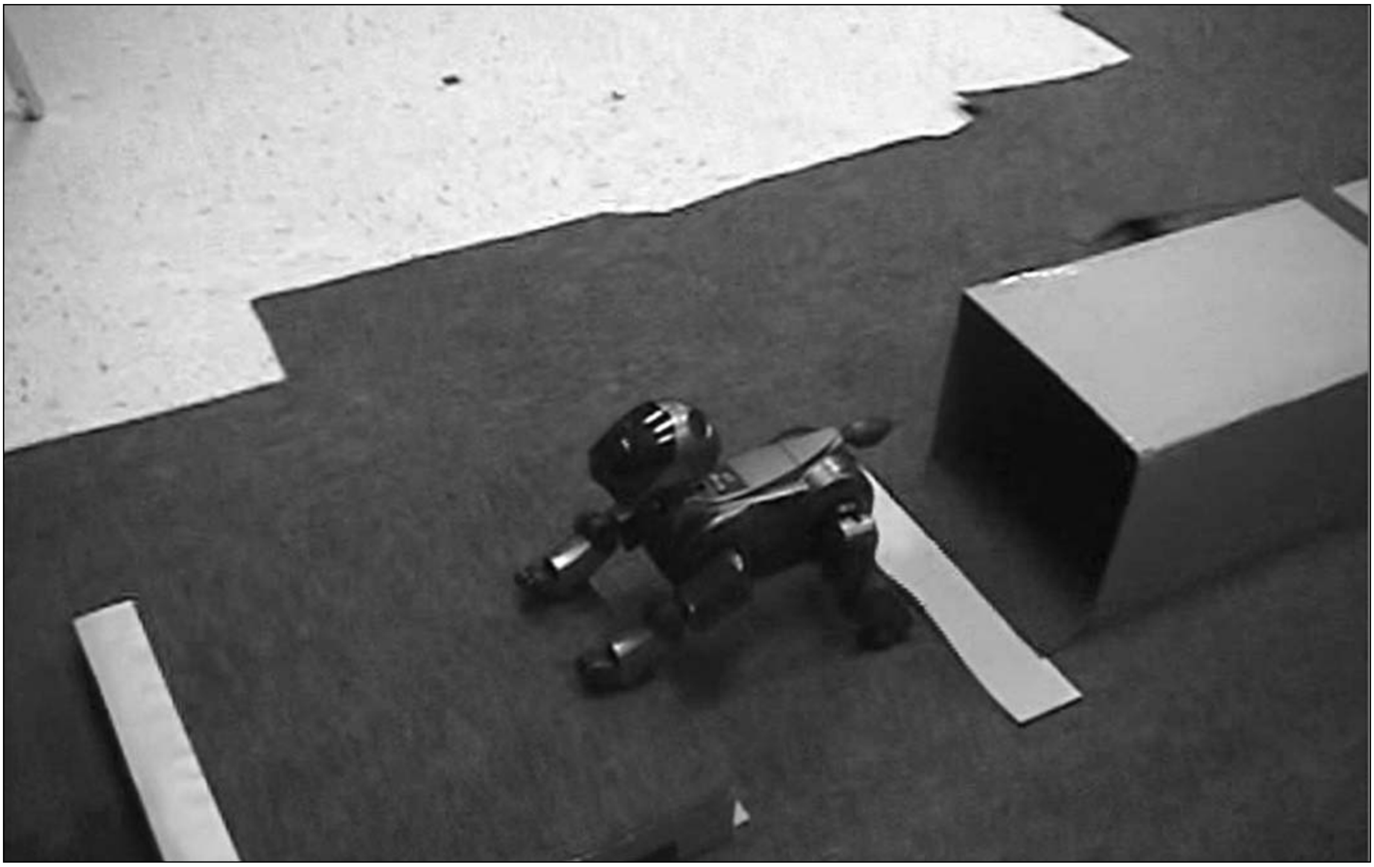

Figure 11. Student Final Project: An AIBO Navigating through an Obstacle Course.

(Thanks to Katie Chang and Ling Xu.)

ground. When encountering an obstacle, the robot performs a custom motion to move past the obstacle. Students learn to use vision to identify obstacles through markers and to write custom motions to surmount them.

Tag. This project uses localization and inter-robot communication to allow two robots to play a game of tag. Both robots use the six landmarks placed around the carpet to localize themselves. They use this information to transmit their *x, y* locations to each other. With this shared information, the robot designated as "it" tries to come within a minimum distance of the other robot while that other robot tries to evade. The AIBOs switch roles when one comes within a minimum distance of the other. Students learn how to use localization, communicate effective state information, and synchronize behaviors.

Maze Learning. An AIBO explores a maze of t-junctions and dead ends using the robot's visual sonar sensor module to identify distances to the maze walls. The robot starts at one part of the maze and explores until it finds an orange ball at the end. Whenever the robot encounters a t-junction, it remembers the last direction that it took. After reaching the goal, the AIBO restarts the maze at the beginning and demonstrates that it always takes the same correct path through the maze to the goal. Students learn to use the vision system for navigation and obstacle avoidance as well as to program their AIBOs to reason about topological maps.

AIBO Dance. Two AIBOs dance in time to a custom song. This project demonstrated the wide range of possible AIBO motions. Such motions include flipping over to walk on their backs, sitting and waving both front paws in the air, and rolling completely over. Students focus on developing custom motions for a pair of AIBOs and synchronize their motions.

Final Project, 2004

In this project style, every student is required to program the robots to complete the same task. Controlling the subject material allows the instructor to more thoroughly test the students' cumulative knowledge of the AIBOs and the CMRoboBits source code. This project required students to program two AIBOs to cooperatively find their way through two identical mazes. Figure 15 shows example mazes.

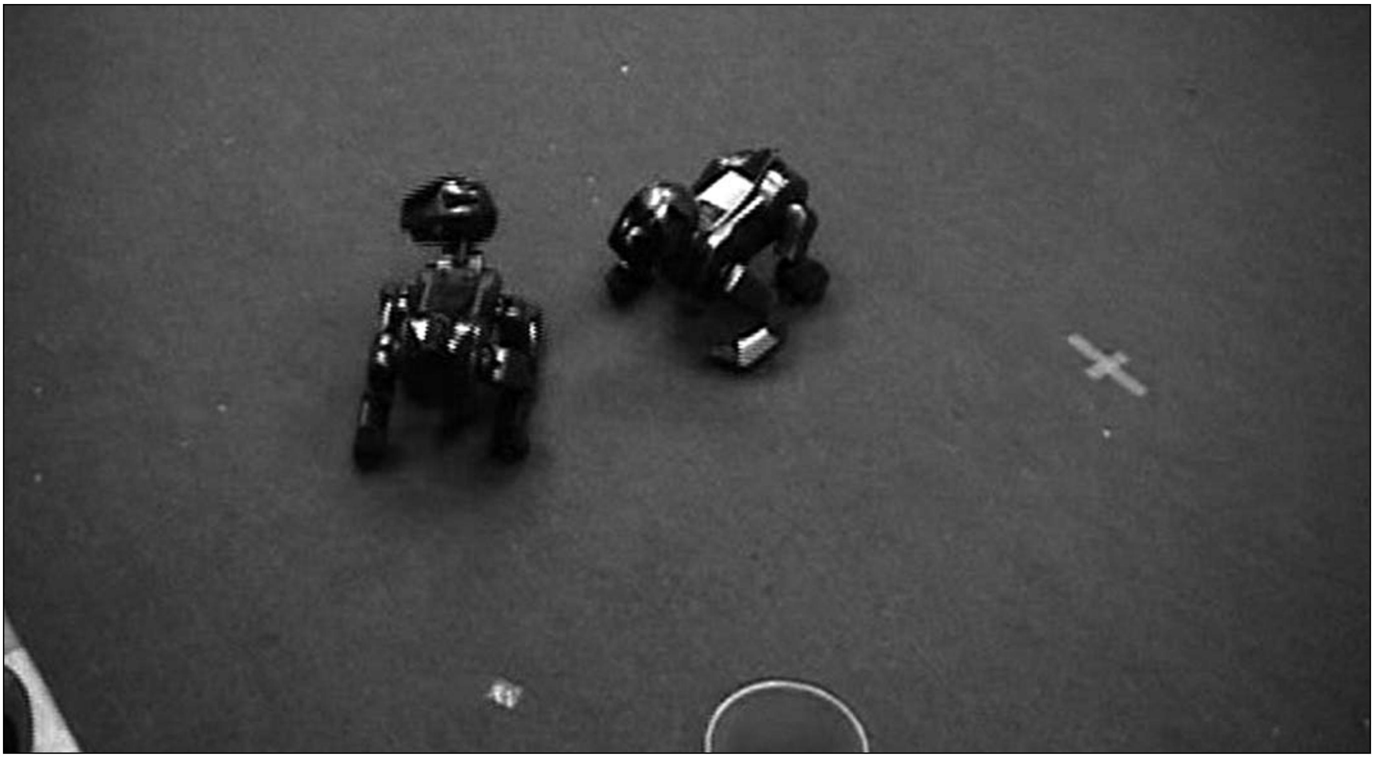

Figure 12. Student Final Project: Two AIBOs Playing Tag.

(Thanks to Noah Falk and Joe Delfino.)

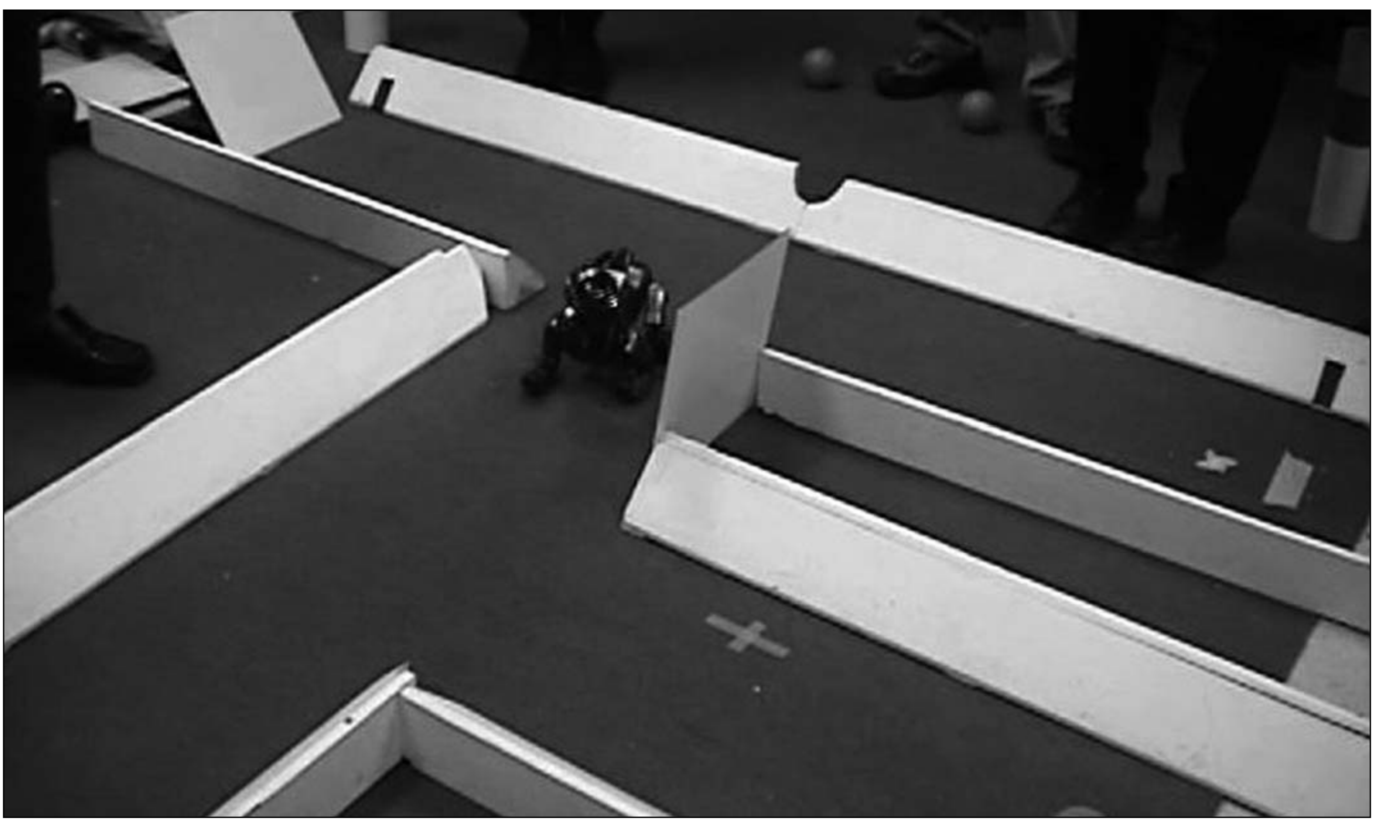

Figure 13. Student Final Project: An AIBO Learning a Maze through Exploration.

(Thanks to Sylvain Paillard and Abe Wong.)

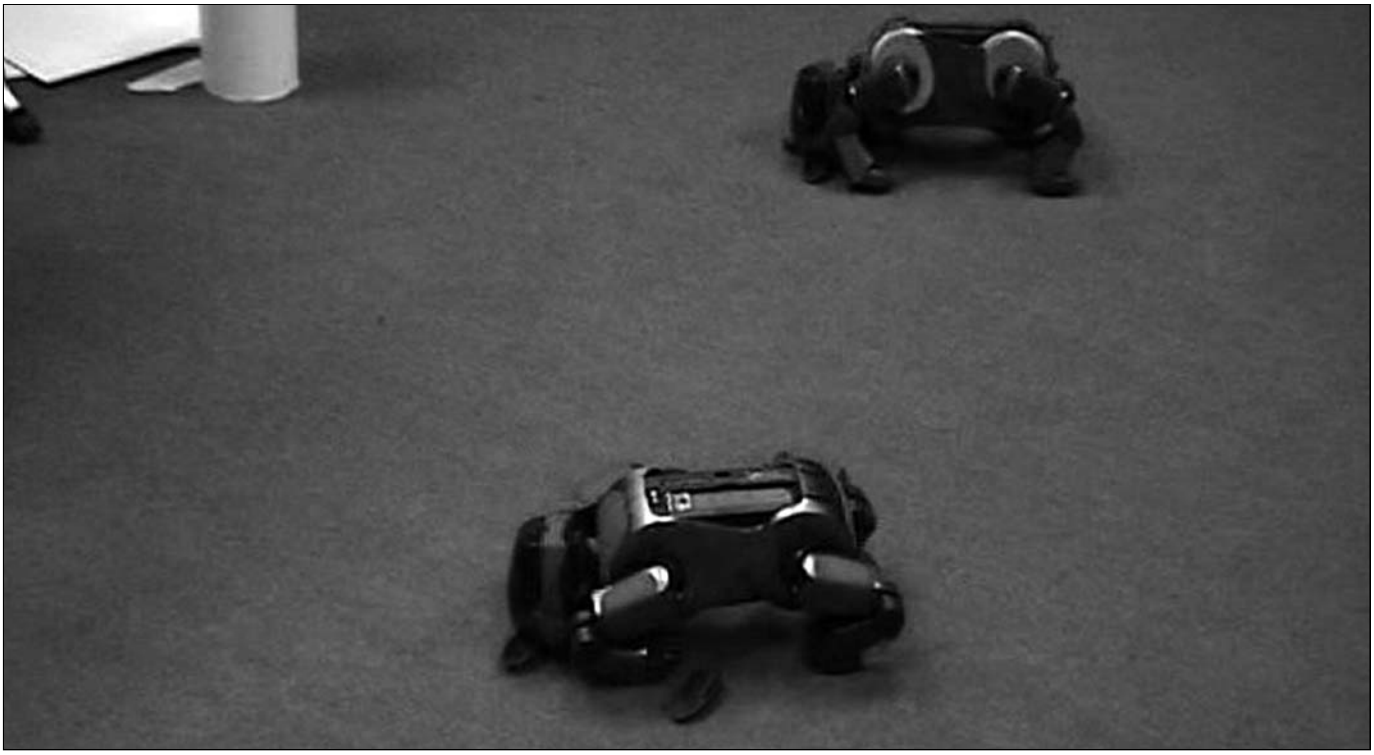

Figure 14. Student Final Project: Two AIBOs Dancing in Time to Music.

(Thanks to Titika Sanghi and Yash Patodia.)

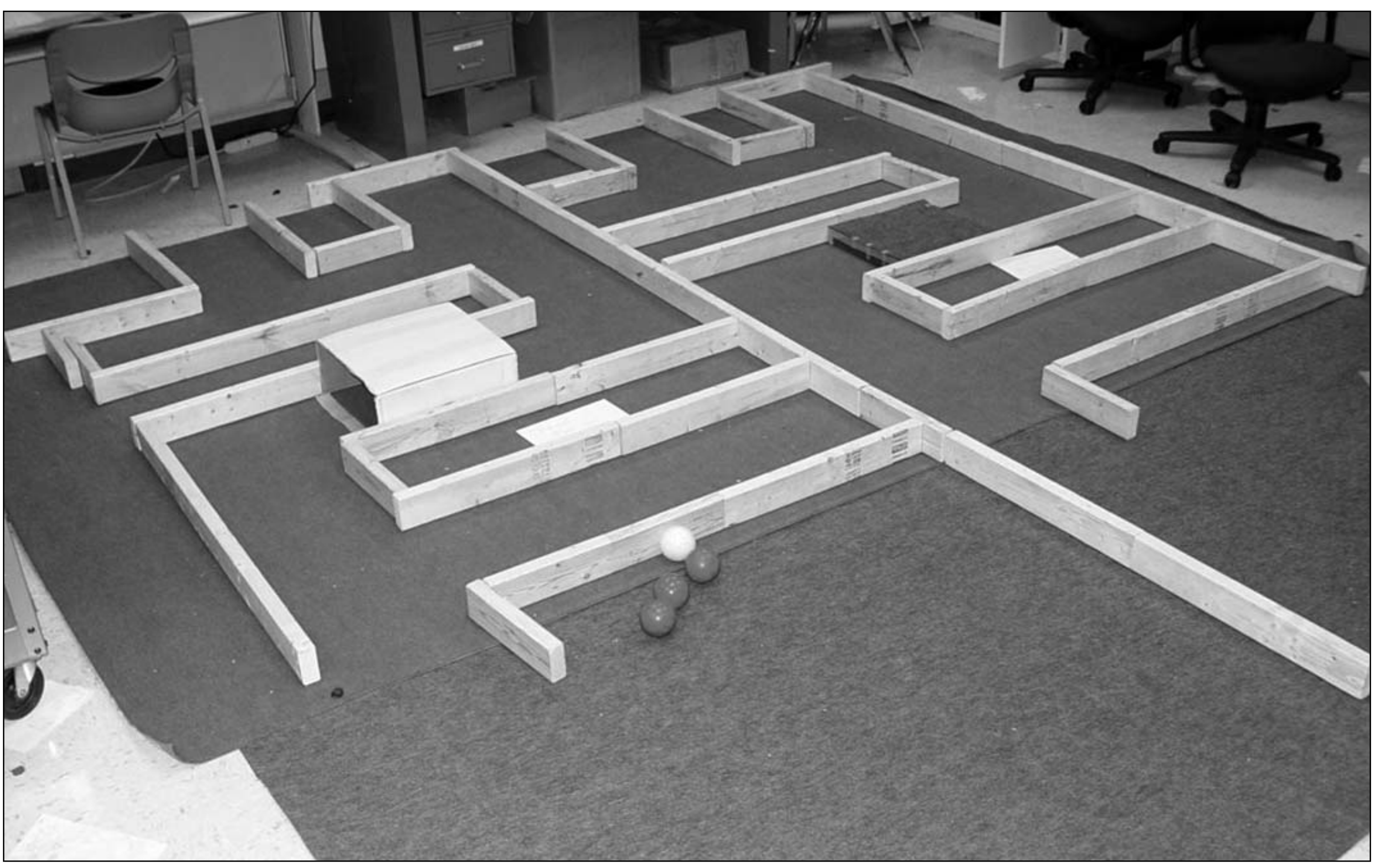

Figure 15. Final Project Maze in 2004.

Each AIBO starts in its own maze and has to find its way through to the end. Each maze contains an obstacle that the AIBO must either climb over, crawl under, or crawl through. The end of the maze contains an open area filled with colored balls that the robots must cooperatively empty. This project emphasizes navigating through the maze, overcoming obstacles with motions, and cooperating through wireless communication.

Cumulative Skill Testing. For this final project, students must make use of their knowledge of both single and multirobot behaviors. Single robot behaviors include (1) navigation and obstacle avoidance; (2) visual recognition and interpretation of landmarks; (3) motions to climb over obstacles; (4) visual navigation to colored targets. The multirobot behaviors include (1) synchronization of world model for maze navigation; (2) negotiation and agreement on shared action for ball task.

Cooperative Navigation in the Maze. In order to navigate the maze, the AIBOs receive clues at each of the t-junctions as to which direction they should turn to reach the end of the maze. These clues consist of blue and pink signs that the robots must interpret. If the sign tilts to the left, the robot takes the left path. If the sign tilts to the right, the robot takes the right path. If the sign does not tilt, the robot turns around and takes the path behind it. In order to read and interpret the signs correctly, the robots must face the sign head on.

In order to encourage cooperation between the robots, not every t-junction in a specific robot's maze has a sign. However, between the two mazes, each t-junction has a sign. Therefore, students program their AIBOs to communicate with each other to share the information as to which direction they should turn when they arrive at the same t-junction.

Cooperative Cleaning. When both robots find their way through the maze, they work together to remove all of the colored balls from two independent open areas (each robot had its own separate area). In order to remove a ball, both robots need to navigate a ball of the same color and stop with their head over it. If both balls are the same color when the robots stop, the balls are manually removed by an observing human. If the robots stop over two balls of different colors, the balls are reset to a new area. Because the robots do not see each other's activities, they must communicate with each other.

Future Final Projects

In 2005, we plan to develop a final project that includes a significant learning component whereby the robots have the opportunity to improve their performance through experience. We seek to combine the best features of each of the last two final project approaches: the creativity that arose from carefully and realistically chosen student-designed projects and the breadth of skills demonstrated in the cooperative maze-clearing project. We intend to offer the same final project to all students but will allow exceptional students to propose projects. Such project proposals must exhibit the same breadth of coverage over the course materials before they are approved.

Conclusion

We are very interested in teaching AI concepts within the context of creating a complete intelligent robot. We believe that programming robots to be embedded in real tasks illustrates some of the most important concepts in artificial intelligence and robotics, namely, sensing uncertainty, reactive and deliberative behaviors, and real-time communication and motion.

We seek to find a good balance between the theoretical aspects of the in-class lectures and the hands-on labwork. We feel very strongly that both are required to achieve a well-rounded learning experience. As of the 2004 class, the students did not have any in-class midterm or final exam in lieu of a more in-depth final project; we have decided to include them the next time the course is taught. There are too many concepts to fit into a cumulative final project. A more traditional exam schedule will fill this gap. In 2003 and 2004, the class used the ERS-210 model of the AIBO. In 2005, the students began using the newer AIBO ERS-7s, which have higher-resolution cameras, slightly different kinematics, and more processing power.

It is a testament to the AIBO's capabilities that the course explores too many AI topics to capture in a single final project. Indeed, we are seeking to better integrate topics such as machine learning (and perhaps there are others) in future offerings. We have found that, with the AIBO's hardware and software resources, undergraduate students can efficiently and effectively investigate the broad and growing connections among AI robotics topics.

The current course materials, including some final project videos, are available on the course web page listed in table 1.

Acknowledgements

We would like to thank Sony for its remarkable support of our research, specifically by making the AIBO robots accessible to us since their first

Web Site	URL
Course web page for the current year	http://www.andrew.cmu.edu/course/15-491
Archived course materials from previous years	http://www.cs.cmu.edu/~robosoccer/cmrobobits
SONY OPEN-R SDK home page	http://openr.aibo.com
AIBOs at RoboCup	http://www.robocup.org
CORAL research group web page	http://www.cs.cmu.edu/~coral

Table 1. Web Resources for Using AIBOs and the CMRoboBits Course Materials in Education.

conception in 1997. Sony has continued its support through these years and is very interested in following the impact of an AIBO-based course. We would also like to thank the Carnegie Mellon Computer Science Department for approving this new course in the fall of 2003 and providing the lab facilities.

Thanks to Nick Aiwazian, James Bruce, and Nidhi Kalra for help with the course and robot development. Thanks also to Sony for having provided to us the remarkable AIBO robots since early 1997.

Note

1. Found at http://openr.aibo.com/.

References

Lenser, S.; Bruce, J.; and Veloso, M. 2001a. CMPack'00. In *RoboCup 2000: Robot Soccer World Cup IV.* Lecture Notes in Computer Science 2019, ed. P. Stone, T. Balch, and G. Kraetzschmar, 623–626. Berlin: Springer Verlag.

Lenser, S.; Bruce, J.; and Veloso, M. 2001b. CMPack: A Complete Software System for Autonomous Legged Soccer Robots. In *Proceedings of the Fifth International Conference on Autonomous Agents,* 204-211. New York: ACM Special Interest Group on Artificial Intelligence, Association for Computing Machinery.

Uther, W.; Lenser, S.; Bruce, J.; Hock, M.; and Veloso, M. 2002. CM-Pack'01: Fast Legged Robot Walking, Robust Localization, and Team Behaviors. In *RoboCup 2001: Robot Soccer World Cup V.* Lecture Notes in Computer Science 2377, ed. A. Birk, S. Coradeschi, and S. Tadokoro. Berlin: Springer Verlag.

Veloso, M., and Uther, W. 1999. The CMTrio-98 Sony Legged Robot Team. In *RoboCup-98: Robot Soccer World Cup II*. Lecture Notes in Computer Science 1604, ed. M. Asada and H. Kitano, eds., 491–497. Berlin: Springer Verlag.

Veloso, M.; Lenser, S.; Winner, E.; and Bruce, J. 2000. CM-Trio-99. In *RobotCup-99: Robot Soccer World Cup III.* Lecture Notes in Computer Science 1856, ed. M. Veloso, E. Pagello, and H. Kitano, 766–769. Berlin: Springer Verlag.

Manuela M. Veloso is professor of computer science at Carnegie Mellon University. She received her Ph.D. in computer science from Carnegie Mellon in 1992. She received a B.S. degree in electrical engineering in 1980 and an M.Sc. in electrical and computer engineering in 1984 from the Instituto Superior Tecnico in Lisbon. Veloso's research focuses on teams of intelligent robots where cognition, perception, and action are seamlessly integrated to address dynamic environments. In fall 2003, building upon her extensive experience with artificial intelligence and robotics, she created a new undergraduate course, CMRoboBits: Creating an Intelligent AIBO Robot, at Carnegie Mellon to concretely teach all levels of robot algorithms. Veloso is a Fellow of the American Association of Artificial Intelligence. She is vice president of the RoboCup International Federation. She was awarded an NSF Career Award in 1995 and the Allen Newell Medal for Excellence in Research in 1997. Veloso is the program chair of the 2007 International Joint Conference on Artificial Intelligence.

Paul E. Rybski is a systems scientist in the Robotics Institute at Carnegie Mellon University. He received his Ph.D and M.S. in computer science from the University of Minnesota in 2003 and 2001, respectively. He received an interdisciplinary B.A. in mathematics and computer science from Lawrence University. His research interests include distributed sensing and state estimation algorithms for teams of mobile robots, robust high-level environment modeling for sensor-impoverished robotic systems, and recognition of agent (human/robot) activities through observation. He served as cochair for the 2005 and 2006 AAAI Mobile Robot Competition and Exhibition.

Scott Lenser graduated with a B.S. in computer engineering from the University of Illinois in Urbana-Champaign in December 1996. After working for Intel Corporation in Portland, Oregon, for a year and a half, he enrolled in the Ph.D. program at Carnegie Mellon University. While at Carnegie Mellon, he participated in Carnegie Mellon's entry into the legged league of the RoboCup robot soccer competition from 1999 to 2005, leading the 2002 world champion team. He graduated with his Ph.D. in August 2005. He is currently employed at iRobot Corp. as a robotics researcher.

Sonia Chernova is a Ph.D. student in the Computer Science Department at Carnegie Mellon University. Her research interests focus on learning and adaptation in autonomous robotic systems.

Douglas Vail is a Ph.D. student in the Computer Science Department at Carnegie Mellon University. Advised by Manuela Veloso, his research interests include building probabilistic models to detect abstract robot states from sensor data, multirobot teams, and robot soccer.