AiboConnect: A simple programming environment for robotics

Eric Chown, Greydon Foil, Henry Work, Yi Zhuang

Bowdoin College
8650 College Station
Brunswick, ME 04011
echown@bowdoin.edu

Abstract

AiboConnect is a program designed to simplify the transition to advanced research with robots. It is simple enough to be used in introductory level courses, yet with enough features that it has been used in advanced research projects. This article describes AiboConnect in some detail and provides concrete examples of how it has been used in the classroom in a variety of different courses.

Robots in the curriculum

The National Science Foundation (NSF) Shaping the Future report suggests that science education should "focus on the processes of inquiry and discovery; and rekindle the unique curiosity, the sense of wonder, with which every child is born" (NSF, 1996). Pedagogical studies and reports (Bonwell & Eison, 1991; Tewksbury, 1995; NRC, 1996) recommend that undergraduate courses create opportunities for students to "do science" by simulating or engaging in research activities that provide students with life-long learning skills in problem-solving, quantitative reasoning, and communication.

Computer scientists are increasingly addressing these issues by teaching computer science through the use of robots in undergraduate curricula (Meeden, 1996; Turner, et al., 1996; Beer, et al., 1999; Harlan, et al., 2001; Blank, et al., 2003; Klassner, 2002). Many schools have incorporated research-level robots into upper-level courses, but the cost is high, both in terms of dollars as well as the amount of work involved. A typical solution to some of these issues has been to use LEGO robots which are attractive because they are relatively cheap and simple to program.

We have, however, encountered problems using LEGO robots in our courses. LEGO robots have very primitive sensors and actuators, so there are a limited number of projects for which they can be used. They are not as reliable as research-level robots and students often spend as much time building the robot as experimenting with it. Further, the increased popularity of LEGO robots in K-12 curricula means that many students have already done projects with these robots. The use of advanced robots with greater capabilities is likely to strengthen the positive effects of using robots to reinforce conceptual learning, as well as mitigating the effect that the students are using "toys." Finally, as Blank, et al. (2004) have pointed out, using LEGO robots, while valuable in many respects, can lead to a robotics dead-end and make it difficult for students to make the transition to higher-level robotics. Indeed, we have found that the transition from LEGO robots to research-level robots is substantial.

Increasingly the Sony Aibo has been seen as a potentially useful alternative to LEGO robots. While Aibos are substantially more expensive than LEGOs (currently costing approximately $2,000 apiece) the difference is not prohibitive especially in light of the difference in capabilities between the platforms. Aibos work right out of the box and their advanced sensors and actuators have helped make them a leading platform for robotic research.

There are several challenges for using Aibos in the classroom. Because Aibos run on Sony's proprietary Aperios operating system there is a significant startup cost in terms of time involved for students and faculty. Further compounding this problem are the simple mechanics of working with Aibos and the lack of reasonable debugging tools in the Open-R programming environment used to program the Aibos in Aperios. Finally it represents a significant challenge to program the Aibos to function on a basic level (e.g. getting them to walk).

On the other hand, the rewards for using Aibos are high. Our students that have had the opportunity to work with them have reported exactly the sense of wonder hoped for by the NSF. Working with robots quickly makes otherwise abstract problems in computer science and artificial intelligence concrete. Further, there is very little sense that the Aibos are "toys;" rather, the students feel a sense of pride to be working on the same robots that are used in advanced research. An additional advantage is that students transitioning to advanced research come with a level of familiarity and confidence that is unusual for novice researchers.

The major goal of AiboConnect is that it be useful at every level of computer science from CS1 to advanced research in the same way that a language such as Java might be. To accomplish this it is necessary that AiboConnect enable students to work at these different levels in a seamless way. For example, for a 101 student just trying to master loops it is important to have access to commands at a very high-level. However, as a student's mastery of computer science grows and the projects they do become more complex they will need more and more fine-grained control over what the robots do.

In this paper we describe AiboConnect and how it can be used in a computer science curriculum. We will first describe the basic Aibo platform and how it has been used at other schools. Much of this has taken place in the context of the international Robocup competition (Robocup, 2005). Robocup has been directly responsible for a number of advances in Aibo programming, and because it is open source educational programs such as AiboConnect can take advantage of this work. After looking at what other schools are doing we will discuss AiboConnect itself and its capabilities. Finally we will review how AiboConnect has been used throughout the curriculum at Bowdoin College.

Current Practice

Ostensibly Aibos are "entertainment robots" that look and act like dogs. Indeed this is part of the attraction of using them in the classroom. They are engaging and interesting to watch. In recent years, however, they have been increasingly used as research robots in academic institutions. One of the primary platforms that has pushed this interest is Robocup (Robocup, 2005). Robocup consists of a number of robot competitions, notably including a 4-legged soccer league in which teams of Aibos compete against each other in soccer.

Aibos are equipped with CMOS image sensors, stereo microphones in their ears, touch sensors in a variety of locations and other, less useful, sensors. Their head has three degrees of freedom, each of the legs has three degrees of freedom and they are also capable of moving their ears, their tails and their mouth. In addition they have more than 40 LEDs to output colored lights at various points on their body and a speaker on their chest to output sound. Finally Aibos also have wireless built in.

Since Aibos are produced by Sony, they must be programmed in Sony's proprietary Open-R environment which uses a form of C++. Programs are transferred to a Sony memory stick and put onto a robot. Since Open-R is relatively inelegant a recent trend in Robocup teams is to put high-level wrappers around a basic core. For example, a common practice is that low-level code such as vision and movement are done in Open-R, but that behaviors are programmed in higher-level languages like Python. This is very close to what AiboConnect is, but we move the high-level programming off of the robots completely and on to workstations. This is not legal in Robocup, but is extremely useful for general development. Programmers are not forced to constantly load their programs onto memory sticks, debuggers can be used, graphic windows can display what the robots see, etc. In short, programming the robot is no different from any other sort of programming. There is no special language or operating system involved.

Several other educational packages are in development for the Aibo platform. Tekkotsu (Touretzky & Tira-Thompson, 2005), developed at Carnegie Melon, contains a number of built in modules that programmers can use in creating their own projects. The basic philosophy of Tekkotsu is not far from that of AiboConnect in that the idea is to create a basic structure that programmers can build on.

We believe, however, that Tekkotsu has a number of limitations that make it difficult to use in courses not explicitly devoted to Aibo programming and especially at the undergraduate level. Tekkotsu itself is a rather complex environment and requires the completion of several tutorials just to be able to write even simple programs. More importantly, Tekkotsu is a robotics environment more than a programming environment and is designed as such. Rather than "programs" developers write "behaviors" that must fit within Tekkotsu's behavior-based architecture. While this is useful for research on robotics, simpler designs are necessary for more general classroom use. Further, Tekkotsu is built explicitly around C++. As already noted most Robocup teams have explicitly rejected C++ as a platform for high-level programming in favor of languages like Python.

The Pyro group (of which Bowdoin is a part) provides another alternative (Blank, et al., 2003). The Pyro platform is designed to provide a standard development environment usable on a range of robots including Aibos. Pyro is still in the development stage for use on Aibos. While we consider it an extremely attractive choice in many respects, it too has potential drawbacks in courses outside of robotics. For one, Pyro on Aibos is implemented on top of Tekkotsu. This places one complex system on top of another and adds a level of complexity and frustration to issues such as support. For another, Pyro, like Tekkotsu, is built for doing advanced work on robots. Our goal is to provide a platform which is not only adequate for doing advanced work on robots, but which is simple enough to use in even introductory computer science courses and can even be used to teach programming.

We have tried to build on the best of these ideas in AiboConnect. AiboConnect consists of a client-server package. The server runs on an Aibo and connects to the client over wireless. As with Tekkotsu and Pyro, the server program contains a package of useful robot routines that are callable by the client. The difference is in the way the client interacts with the server. Client programs merely send and receive messages from the server program. This means that client programs can be written in virtually any programming language (we currently run versions implemented in Java and in C++ and should soon have a Python version). AiboConnect has also been written such that clients can interact at very low levels (e.g. sending commands that specify joint angles for each of the robot's legs) or at much higher levels (e.g. simply telling the robot to walk at different rates of speed).

In the rest of this article we will describe the AiboConnect platform in more detail as well as how it has been used in courses at Bowdoin.

Description

AiboConnect is built out of a number of components. The raw material for AiboConnect was the EchoServer client provided by Sony that provides the capability for wireless communication between Aibos. Starting with that software as a base we have added some of our own work in addition to adding software or ideas developed by the University of Pennsylvania robocup team (Penn, 2005), Tekkotsu, and especially (in terms of ideas) from the Aibo programming course at Carnegie Melon (CMU, 2004) and the CMU robocup team (CMUDash, 2005). As of now the software in AiboConnect consists of the main server, a system for controlling joints, and a vision system. We will discuss these in turn.

Echo Server

The EchoServer program provided by Sony gives Aibos the ability to communicate with each other or a workstation. We have modified this to function on the one hand as a message passing system, and on the other as a main controller for the robot. AiboConnect's EchoServer module is constantly waiting for instructions from client programs. These instructions come in the form of strings such as "walk 0 10 0." The strings are then parsed and the robot changes its behavior accordingly. In the example the robot would walk forward as fast as it could. These strings specify the type of command and the parameter settings for the command. For example, if the client is working on implementing a walk, it might send a command that specifies all 12 joint angles for the robot's legs. In terms of movement there commands that control the exact positions of the robots joints and other high-level command that tell the robot which direction to walk and how fast. There is a similar low/high-level dichotomy in vision. Clients can request raw vision or alternatively they can request that the robot do its own color segmentation and send back just the segmented vision (at significantly lower bandwidth). This provides instructors with a great deal of flexibility in terms of the kind of assignments they can provide.

Movement

As noted in the previous subsection there are two ways to control the Aibo's movement. The high-level motion control system consists primarily of a walk engine developed at the University of Pennsylvania (Penn, 2005). The low-level motion control system gives the client program direct control over the joint positions of the robot.

Commands to the walk engine are sent as strings of four items – the walk command itself and three integer parameters. The first two parameters specify relative speed in the horizontal and vertical directions (with magnitude from 0 to 10) with respect to the dog's current facing. The third parameter overrides the first two when set to a value other than 0 and specifies a fast turning in place motion in either direction depending on whether the value is positive or negative. For example, the command "walk 10 10 0" would specify that the Aibo should walk at a 45 degree angle from its current facing as fast as it can. "walk 0 -5 0" on the other hand would specify that the Aibo should walk backwards as fast as it can.

As previously noted the low-level motion control system takes joint positions as inputs. Rather than directly calling for the robot to move its joints to the correct positions, it first ensures that the difference between the current settings and the requested settings are not too large. If they are large, the motion system will essentially chop the motion into a fast series of smaller motions.

The low-level system affords a great deal of flexibility in terms of teaching. The two main methods of getting a robot to walk are "frame-based" motions and inverse kinematics. In a frame-based system motion is viewed as a series of snapshots much like a computer animation. These can be used as the basis for determining joint positions to send to the robot. Aibo walk engines, such as the one used in AiboConnect, use an inverse kinematics system. Walks are specified as elliptical movements of the foot through space. Inverse kinematics is then used to determine the necessary positions of the joints to trace the ellipses. Either of these methods can be implemented on the client side of AiboConnect though the results will not be as smooth and fast as, for example, the included implementation of inverse kinematics. A simple example assignment is to have students develop their own walk using frame-based methods. For example, we have given students a series of snapshots of robots walking and had them use the pictures as a basis for coming up with scripts to replicate the walks.

Vision

Aibos posses a color vision camera that produces images in the YCbCr color space (this is similar to YUV and the two are often confused). In AiboConnect clients have the option of taking the raw camera feed in either YCbCr or RGB form (the conversion is simple and can be done onboard). This is obviously useful for low-level vision assignments, but it is also slow, as it requires substantial bandwidth.

Aside from the bandwidth involved the other drawback of using a raw camera signal is that each pixel can take one of more than 16 million values. In practice most robot vision systems start by doing some form of color segmentation to transform each pixel into one of a few basic colors (currently we use eight in AiboConnect, but the number can be expanded if the segmentation is done in software). This segmentation can be done on either the client or server side. The advantages of doing it on the server side are reduced bandwidth and speed (the segmentation can be done directly in hardware). AiboConnect contains the ability to do color segmentation on the server side so clients can deal with a vastly simplified vision problem. In addition we are currently working on adding some very basic object recognition to the vision system.

An additional problem with robot vision is the sensitivity of color segmentation to factors such as lighting. In Robocup, for example, lighting is carefully controlled to be as uniform as is possible. Even so, color segmentation algorithms need to be calibrated to work reasonably in any given environment. This means that any programming package such as AiboConnect must also include some additional tools to do the calibration. This is an ongoing project in AiboConnect – we have tools that work, but are always working on improving them. To do a calibration we first use the Aibos to take snapshots of the environment in which we are going to work. These snapshots are then put into an image-viewing program where a human looks at the YCbCr values for individual pixels and uses them to build a rough color map. For example, the person might be trying to figure out what YCbCr values map to the color orange. They would look at orange objects in the captured picture and retrieve the corresponding values. Using such values they can create ranges for each color. While this is fine for many tasks, especially where the colors are sharply defined, it has limited accuracy in complex environments. Our most recent version of this tool is slightly more sophisticated in that it can automatically extract regions of colors from the saved snapshots and collect data on the colors with minimal human intervention (in this case simply identifying the area and what the color is). These can be used to generate individual tables for each color. One side-effect of this is that particular values can be mapped to multiple colors. This is actually quite useful as it reflects uncertainty related to context about those values.

The algorithms for color segmentation and the calibration tools are very much an ongoing project. This is one clear area where programs like AiboConnect benefit from Robocup research. Since this vision is so crucial to Robocup a number of good tools and algorithms have been developed to address these problems. Within the last week, for example, we have implemented a new Robocup vision algorithm that is based more on relative changes in YCbCr values between adjacent pixels than on their actual values. The new algorithm appears to be far more robust with regard to factors such as changes in lighting.

Usage

The goal of AiboConnect is to provide an environment simple enough to be used in introductory classes, but robust enough for advanced projects as well. One of the difficult things about working with Aibos are the problems that arise with programs that run on the robot and within the context of Sony's Open-R environment. Updating and debugging, for example, are extremely difficult. The process is much simpler with AiboConnect. The program running on the dog need not change at all so that all debugging can be done on a workstation running any operating system in virtually any language. At Bowdoin we run AiboConnect using Java and C++ client programs. In each version it is a simple matter to pop up a graphics window to display what the dog is seeing in real-time. We used this technique to build a Robocup team from scratch that competed in the 2005 American Open. The code was primarily developed and debugged in the C++ client version of AiboConnect and then ported into Open-R which uses a variant of C++.

AiboConnect in the Classroom

At Bowdoin our goal is to integrate Robots into the entire curriculum, not just artificial intelligence classes. To that end we have experience with using AiboConnect in three of our primary courses and plans for more of our courses. What follows is a brief summary of the kinds of assignments that we have given in our courses.

Robotics

In the fall of 2004 we offered our first Robotics course. The Aibos were used for approximately half of the course and that portion of the course was modeled heavily on Carnegie Melon's Aibo Programming course (CMU, 2004). The major difference being that our students did their programming in AiboConnect while at CMU students worked within Open-R. The Robotics course contained three major AiboConnect assignments – a vision assignment, a motion assignment, and an integrative assignment.

The motion assignment was straightforward – use the frame-based idea to create a walk for the Aibos. The assignment is a nice introduction to robotics and gives the students an excellent feel for basic issues in motion and kinematics. Alternatively it is simple to create an optimization assignment based on motion. Walk engines for Aibos have a number of settable parameters. These can be used as the basis for an optimization problem where the problem is to discover good settings.

In the vision assignment the goal is object recognition and extraction. We populate the lab with brightly colored, easily recognizable objects (we use items from Robocup such as beacons and colored balls) and the assignment involves having the robots search for and point to objects in a specific order (e.g. first find the orange ball, then the blue goal . . .).

At the next stage of the course the students have experience with the two most basic aspects of robotics – vision and motion – so it is possible to give them more integrative assignments. Since our students were interested in Robocup we built a third assignment around Robocup. In this assignment we created a facsimile of a Robocup field (green carpet, colored goals at each end) and the assignment was that the robots would start at a random point on the field and were to find the ball and try to move it towards the goal. As an assignment it proved to be slightly on the ambitious side, but students enjoyed working on it nonetheless and made significant progress. We broke the class into two groups of six students and each group came up with its own solution. One group actually implemented Monte Carlo localization on their own and came very close to completely solving the problem (the main problem was moving the ball effectively). The second group, by contrast, spent more time using AiboConnect to explore different motions for kicking and walking (the number of degrees of freedom of the joints makes finding competitive walks a very difficult problem) and the competitive nature of the groupings kept interest high. The progress made by both groups provided a substantial code base for use by the Bowdoin Robocup team.

The use of Aibos and AiboConnect in the Robotics course was a great success. In the other part of the course we had used Pioneer robots. While students enjoyed that part of the course too they reported far more satisfaction in working with the Aibos.

Programming Languages

The following spring (2005) we used AiboConnect in our Programming Languages course. None of the students in this course had taken Robotics. The nature of some of the assignments in our Programming Languages course is essentially to get some experience with a new language. Since our curriculum is grounded in Java one of the languages we teach in Programming Languages is C/C++. One of the goals of this assignment was simply to force students to use pointers and memory management techniques.

The assignment was a variation of the vision assignment that had been used in the Robotics course. In this assignment the students were again given the thresholded vision, but this time they were required to find all of the "blobs" in every frame (a blob is a connected group of pixels of uniform color) and draw rectangles around the largest blob of each color. Since the assignment required a recursive algorithm to do blob detection students were required to create copies of the data and use other basic memory management techniques.

Data Structures

In the fall of 2005 we used AiboConnect in our Data Structures course. Since it is a Java course we used the Java version of the client. For this assignment we again used a variation of the vision assignment, this time extracting blobs using a *non-recursive* algorithm. The challenge for the students was creating data structures capable of describing and merging blobs.

The following is the skeleton code that was provided for the students doing the lab.

```
public class AiboConnect {
    public static final String MODEL = "220";
    private Aibo220 aibo;

    public AiboConnect(String dog) {
      aibo = new Aibo220(dog);
      aibo.getConnection();
    }

    public void run() {
      while (true) {
        aibo.getThresholdedImage();
        // this is where student code goes
        aibo.nowDrawThresholdedImage();
      }
    }

    public static void main(String[] args) {
      String dog = "aibo3";
      AiboConnect myAibo = new
                     AiboConnect(dog);
      myAibo.run();
    }
}
```

The work for the assignment consisted of processing a two-dimensional array of color values extracted in the loop of the **run** method. From the point of view of the programmer the Aibo is just another Java object from which the program can send and receive messages (this is useful when teaching object oriented programming). In this case the messages just happen to involve vision. The code shown runs on the ERS-220 version of the Aibos. We also have versions for the newer ERS-7s.

AiboConnect at Bowdoin

Our plan is to integrate AiboConnect into virtually all of our courses, with an initial robotics assignment being tested in the CS1 course in the spring of 2006. We have found that students get very excited about working with real robots and will work unusually hard on robot-related assignments. Since introducing the Aibos our enrollments have risen, bucking the current national trend. For example, enrollments in our CS2 course were at 12 for the year before we introduced the Aibos and AiboConnect, went up to 24 the first year, and are at 31 this year. Our Robocup team consists of 16 students this year. To put this into context the college only has 1666 students total and our department has averaged 8 majors a year over the last 10 years. The team includes several sophomores, and even a first-year, who have been able to get involved because of their experience with the Aibos in Data Structures, and because the team uses AiboConnect as a development tool. For these students using AiboConnect is really no different than writing any other program in their chosen programming language except for the added excitement of working with a robot.

Conclusions

John Laird has championed games as a perfect environment for AI teaching and research (Laird and van Lent, 2000). They make a great platform for research because of all of the challenging problems they entail. They are also enticing for teaching because students enjoy them so much. The same arguments can be applied to robotics. Robots do not have to be the carrot at the end of the curriculum anymore but can actually be used to lure new students into computer science courses from the beginning. As an added bonus students can get very early experience with a range of problems central in AI including pattern recognition, vision, planning, reasoning with uncertainty, etc. For an undergraduate institution like Bowdoin this is vital since it gets students into the research pipeline at very early stages. The success we have had with Robocup is indicative that this is possible with robots. Our students will be able to work in a research environment for several years as undergraduates. Nor does the cost need to be high. Aibos currently cost approximately $2,000, about the price of a decent workstation. We have found that four Aibos is more than sufficient for classes with fewer than 20 students.

AiboConnect is very much a work in progress on the server side. The more functionality we can port to the robot the more possibilities are opened up for the clients. We continue to work on new vision algorithms for example. This effort is made considerably easier because Robocup is an open source competition (our code is open source as well) and the tools useful in Robocup are generally useful for applications like AiboConnect.

The use of Aibos at Bowdoin has brought a sense of wonder to computer science that extends beyond department walls. Other faculty at Bowdoin have expressed "tech envy" after seeing Aibo demos. Talks given to faculty, parents, and alumni are always full. Our students have become caught up in that spirit and that can only help them become better scientists.

Acknowledgements

This material is based upon work supported by the National Science Foundation under Grant No. 0092605. Any opinion, findings and conclusions or recommendations expressed in this material are those of the author and do not necessarily reflect the views of the National Science Foundation (NSF). We owe a special debt of thanks to the Carnegie Melon Robocup team – CMDash – and especially Manuela Velosa and Doug Vail. The Robotics course was co-taught with Steve Majercik and his help and comments have been crucial in the development of AiboConnect.

References

Beer, R.D., Chiel, H.J., & Drushel, R.F. (1999). Using autonomous robotics to teach science and engineering. *Communications of the ACM.*

Blank, D., Meeden, L, & Kumar, D. (2003). Python robotics: An environment for exploring robotics beyond LEGOs. *Proceedings of the Thirty-fourth SIGCSE Technical Symposium on Computer Science Education* 35.

Blank, D., Yanco, H., Kuamr, D., & Meeden, L. (2004). The Karel-the-Robot paradox: A framework for making sophisticated robots accessible. *AAAI 2004 Spring Symposium.* (citeseer.ist.psu.edu/662305.html).

Bonwell, C. C., and J. A. Eison, (1991). Active Learning: Creating Excitement in the Classroom: ASHE-ERIC Higher Education Report 1, The George Washington University, Washington, DC.

CMU (2004). http://www.andrew.cmu.edu/course/15-491/

CMUDash (2005). www.cs.cmu.edu/~robosoccer/legged

Harlan, R.M., Levine, D.B., & McClarigan, S. (2001). The Khepera robot and the kRobot class: A platform for introducing robotics in the undergraduate curriculum. *Proceedings of the Thirty-second SIGCSE Technical Symposium on Computer Science Education* 33, 1, 105-109.

Klassner, F. (2002). A case study of LEGO Mindstorms suitability for artificial intelligence and robotics courses at the college level. *Proceedings of the Thirty-third SIGCSE Technical Symposium on Computer Science Education* 34, 1, 8-12

Laird, J.E., & van Lent, M. Human level AI's killer application: Interactive computer games. *Proceedings of the Seventeenth National Conference on Artificial Intelligence, Twelfth Conference on Innovative Applications of Artificial Intelligence*

Meeden, L. (1996). Using robots as introduction to computer science. In *Proceedings of the Ninth Florida Artificial Intelligence Research Symposium (FLAIRS),* J.H. Stewman (ed.), Florida AI Research Society, 473-477.

NRC (1996). From Analysis to Action: Undergraduate Education in Science, Mathematics, Engineering, and Technology, National Academy Press, Washington, DC.

NSF (1996). Shaping the future: New expectations for undergraduate education in science, mathematics, engineering and technology. (NSF 96-139), Arlington, VA.

Penn (2005). www.cis.upenn.edu/robocup/index.php

Robocup (2005). www.robocup.org

Touretzky, D.S., & Tira-Thompson, E.J. (2005). Tekkotsu: A framework for AIBO cognitive robotics. Proceedings of the Twentieth National Conference on Aritificial Intelligence. (AAAI-05). AAAI Press.

Tewksbury, B. J., (1995). Strategies for Moving Away from Lectures. Workshop on Effective and Innovative Teaching Techniques at the National Meeting of the Geological Society of America. P. 23-96.

Turner, C., Ford, K., Dobbs, S., & Suri, N. (1996). Robots in the classroom. In *Proceedings of the Ninth Florida Artificial Intelligence Research Symposium (FLAIRS),* J.H. Stewman (ed.), Florida AI Research Society, 497-500.