Artbotics: Challenges and Opportunities for Multi-Disciplinary, Community-Based Learning in Computer Science, Robotics, and Art

Fred Martin, Hyun Ju Kim, Linda Silka, and Holly Yanco
University of Massachusetts Lowell
Lowell, MA 01854

Diana Coluntino
The Revolving Museum
Lowell, MA 01852

I. INTRODUCTION

Artbotics is a collaboration between faculty in computer science, the arts, and the social sciences. It is a complex project that involves the following elements:

- High school and university students with diverse backgrounds coming together to create interactive, microprocessor-enabled installation art.
- A collaboration among faculty from different departments and intersecting interests who are facilitating a learning experience for their students.
- A collaboration between a university, a metropolitan art gallery, and a public high school.
- Students who are participating in different roles—learners, mentors, collaborators, and research assistants.

This paper explores issues in interdisciplinary teaching and learning, as well as community-based collaborations and service learning, which are major themes in the Artbotics project.

In past publications, we have described results from the summer pilot and first after-school session with high school students, and the planned design for our university course [4]. We described initial results from the university course, including a discussion and photographs of several student artworks [2]. In [3], we focused on the challenge of forming partnerships across academic departments and external institutions.

In this paper, we extend this earlier work in several ways. The full challenge of operating a multi-faculty, multi-disciplinary, project-oriented, community-based university course, over a 15-week period, is now apparent to us and there are some areas for improvement. These observations are likely to be independent of subject matter and therefore may be of value to educators with courses with similar structure or themes regardless of content.

This paper also includes samples software code written by students as part of their exhibits, and an analysis of the ideas that are represented in this code. One of the persistent challenges of project-based work is achieving a good balance between theoretically interesting student work and time-consuming production work. Student-created software code is one place where we can find evidence of learning of ideas that are important in computer science.

We also describe benefits of this work from the standpoint of the faculty who are involved. Operating the undergraduate course entailed a weekly planning meeting in addition to contact hours with students in lecture/discussion and laboratory sessions. We co-taught the course, with all faculty members present whenever possible. The workload was significant, but the benefits were real, including meaningful, contextualized exposure to each other's fields.

II. COURSE AND PROGRAM STRUCTURE

The Artbotics university course was advertised as a 100-level (freshman) course. At the beginning of the semester, we had already received approval for general education credits for our university's Technology standard, making the course valuable for arts and humanities students. By the end of the semester, we received approval for Studio Art as well, meaning that science and technology students could count it towards their art and humanities distribution requirements.

Because of the need for a significant art-theory component to satisfy the arts and humanities aspects, the course included an art history research component (of new media art, not classical art) as well as several writing assignments, along with the project focus. In earlier writing, we also had expected that students would participate in a structured mentoring process, helping develop instructional material that they would then use in working with high school students.

The course was organized with two 75-minute lectures per week and two 75-minute facilitated labs per week. Students were expected to attend one of the two lab slots per week. Faculty generally attended all of the lectures and shared leadership responsibility. Faculty did not divide lab responsibility, which led to high demand on their time and disappointment from some students when all faculty were not present at every lab. In future iterations of the course, we plan to continue joint participation in lectures, but will take turns facilitating the labs, to have clearer lines of responsibility and to manage time demands.

The syllabus included the following deliverables that were expected of the students over the course of the semester:

- Four focused lab assignments (1 lab session plus writeup)

- Two substantial projects, with associated proposals and post write-ups
- Two theory papers
- Two in-class presentations

Most crucially, the projects were at the heart of the class. And the first project's due date was set for us by our community partner, The Revolving Museum. We were offered the opportunity to have our students develop pieces that would be exhibited as part of a museum show developed for Spring 2007.

The opening was called "Electrifying!: The Art of Light and Illumination." The theme was ideal for our students and the Artbotics course, but the date—barely 5 weeks into the spring semester—was not.

Essentially the first third of the course was a quick introduction to our technology (the Handy and Super Cricket microcontroller), followed by a pressured few weeks to conceptualize and then execute original, interactive artworks with embedded computation.

Notably, most of the students in the class rose to the occasion, and the resultant pieces were at least good, if not excellent. However, the recovery period from the scramble was difficult. We back-filled students' knowledge with a combination of technical lectures, assigned lab exercises, lectures and discussions of new media art, and student presentations of the same.

Then, in the last third of the semester, it came time to focus on a second significant (and final) project. This project was not connected with a large Revolving Museum show opening; it was for the Artbotics project particularly.

In this period, the spring high school after-school program was also started (with a new group of high school students). Undergraduate and high school students had shared use of the dedicated Artbotics lab next to the Revolving Museum (about a 20 minute walk from the university).

Both groups of students were working toward the same dedicated opening at the end of the university's semester.

In general, the undergraduate students were not highly motivated to repeat the same process at term's end (conceptualizing and then carrying out an original project). There were factors that contributed to this. The students were frustrated by a change in venue of the course lab. We had moved it from an on-campus location to near the Revolving Museum, in order to facilitate our students' mentoring of the Artbotics high school students. Students' other courses also had significant end-of-term projects, and students' non-school commitments (work, family) did not go away.

All of this put additional scheduling pressure on their already tight schedules. There were successful projects, but broadly the level of effort was not what students had invested in the first project. Ultimately, having two substantial, original, risk-taking projects in one semester may be too much.

Our learning from this can be summarized as the following:

- It is challenging to introduce theoretical material in lecture and labs when students are focused on producing an original project on a short deadline.

Fig. 1. *whadyalookinat*, an interactive table lamp created for the Electrifying! show

- The rewards of partnering with a community institution like The Revolving Museum are huge—the opportunity to work with talented art educators and leaders, and the opportunity for students to see their own work exhibited alongside that of professional artists. However, student exhibitions must be scheduled with the cycle of the semester in mind—e.g., at the two-thirds point of the term.

The next time we offer the university course, we plan the following changes:

- We will start students with a series of weekly skill building labs, inspired by the "knowledge and skill-builder" approach advocated by Burghardt [1]. In parallel with the technology labs, lectures and discussion will introduce themes in new media art.
- In the second third of the semester, students will embark on one substantial design project / art piece.
- In the final part of the semester, students will have the opportunity to re-engineering their work to be robust and lasting, an important consideration for exhibition in a gallery. We will also use this time for reflection on the design process.

III. COMPUTER SCIENCE CONTENT

In submitting the course for evaluation by our general education committee, we described our computer science core content as the following:

- Introduction to imperative programming: functions, arguments and return values.

- Introduction to real-time systems including sensors, actuators and control loops.
- Agent-based models of computing (sense-act loops).
- Elements of robotics systems and how to physically create them (e.g. wiring and construction techniques).
- Uses of computing in a variety of fields.

We can examine the extent to which students' work achieved these goals by looking at their code. Representative work ranges from extremely simple code which is literally one command followed by the next:

```
to main
note 79 6
note 79 2
note 119 2
note 79 2
note 79 10
a, thisway setpower 8 onfor 55
wait 2
```

etc. ... more of the same for a total of 52 lines

```
wait 1
note 54 2
note 54 10
a, thisway setpower 8 onfor 55
end
```

to more sophisticated code that incorporates sensor readings into feedback systems. For example, in earlier work we described *whadyalookinat*, the interactive desk lamp shown in Figure 1 as "A robot with a lightbulb, coffee can, metal wires, servo motors and 6 Sharp IR range sensors along with a Super Cricket. ... Depending on the viewer's relative proximity to the robot, *whadyalookinat* blinks its light as if it is greeting the viewer. As the viewer moves closer, ... *whadyalookinat* turns its neck to follow the viewer" [2].

The core of *whadyalookinat*'s control program is as follows:

```
loop
[
setmaxi 10
getaves
if aave > (maxi + pad)
  [setmaxi aave  setposloc posa ]
if bave > (maxi + pad)
  [setmaxi bave  setposloc posb ]
if cave > (maxi + pad)
  [setmaxi cave  setposloc posc  ]
if dave > (maxi + pad)
  [setmaxi dave  setposloc posd ]
if eave > (maxi + pad)
  [setmaxi eave  setposloc pose ]
if fave > (maxi + pad)
  [setmaxi fave  setposloc posf  ]

servo 7 posloc

ifelse (aave < 15) and
    (bave < 15) and
   (cave < 15) and
    (dave < 15) and
   (eave < 15) and
   (fave < 15) [ a, off ] [a, setpower 8 on ]
]
```

whadyalookinat has 6 distance sensors that are radially positioned to detect the viewer. The `getaves` subroutine (not shown) polls each of the sensors twice and stores an average reading for each sensor in global variables `aave`, `bave`, etc.

Then the series of `if` statements determines which sensor is registering a reading with the closest distance to the viewer. (The sensor readings are inverted; a larger reading indicates a closer object.) The `maxi` variable is initialized at the top of loop with a "far away" reading; each sensor value is sequentially tested to see if it has a closer reading. If so, that sensor's value is then stored as the new `maxi`; also, the position to which to turn is stored in a variable named `posloc`. Precomputed values, already stored in variables `posa`, `posb`, etc., correspond to turning to the direction each of the sensors are facing.

After going through the `if` statements, the servo motor is made to turn to the value of the `posloc` position variable that corresponds to the sensor that registered the nearest reading.

Finally, the `ifelse` statement with the compound logical predicate determines whether the viewer is *close enough that the lamp should be turned on*! This is done by either turning on or off a motor output. That motor output energizes a relay coil; the relay switch applies 120 VAC power to light the desk lamp.

Sophisticated thinking is clearly evident in this algorithm. It is important to note that the students who wrote this code did so with little help (other than syntactical assistance) from the course instructors.

The team of students who worked on the *whadyalookinat* project continued to work together as part of a larger group, and created a multi-instrument musical band for their final project. The system was based on a modified DDR pad for user interaction. Instruments included a xylophone, cymbals, maracas, and a "chicken shake" percussive egg. The automated band made music on its own but then responded in time when you played the DDR pad. The project was called *MUSIC: Maker of User-Synced Instrumental Compositions* (Figure 2).

In the xylophone portion of the piece, a circular plate holds a set of xylophone bars, which hang from the edge of the plate. The plate rotates back and forth, moving a selected bar into a location where it is struck by either of two motorized mallets (see Figure 3).

In the final version, a single Handy Cricket controls the rotating plate and striking mallets. The xylophone plays its own song, with the plate rotating through a sequence of the xylophone bars, where they are struck by either the left or right mallet.

The control code the students developed for the piece implements a state machine. A global variable, `nextnote`, is used to keep track of the position within the song. Here is the main routine from the code:

```
to main
vars ; initialize servo variables
setnextnote 1 ;start with first note of song
loop [
if sensora < 24 [song]
]
end
```
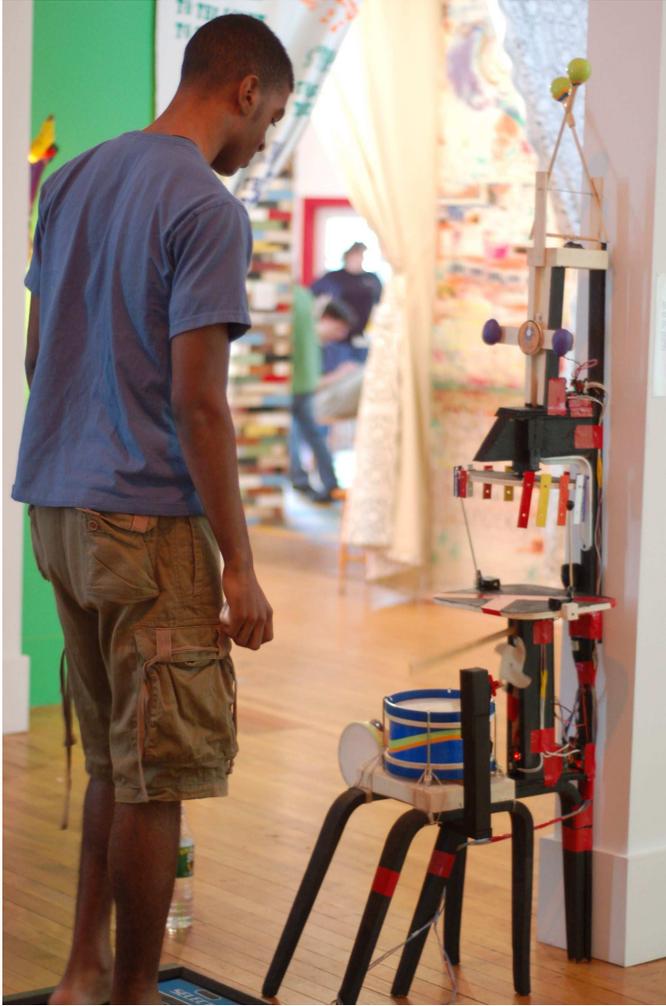
The main loop polls sensor A, and if it returns a "bright enough" value, calls the `song` procedure. Each call to `song` plays exactly one note of the song, and advances the `nextnote` state global. The result is that the song can be stopped after any note by a change in the sensor value, and then resume where it left off.

Here is an excerpt of the `song` procedure, showing how the state machine is built:

```
to song
; song plays the current note, adds 1 to the note var
; when looped plays the entire song and restarts
if nextnote = 1  [lo_a setnextnote nextnote + 1 stop]
if nextnote = 2  [lo_e setnextnote nextnote + 1 stop]
if nextnote = 3  [lo_c setnextnote nextnote + 1 stop]
if nextnote = 4  [lo_b setnextnote nextnote + 1 stop]
...
if nextnote = 29 [hi_c setnextnote nextnote + 1 stop]
if nextnote = 30 [lo_a setnextnote 1 stop]
setnextnote 1    ;shouldn't get here
end
```

Each of the notes is played with an individually constructed, named subprocedure. For example, here is the code for the `hi_a` note. The global `xylo` has been initialized with the number of the rotating plate's servo motor:

```
to hi_a
servo xylo 255
wait 2
leftmallet
end
```

Subprocedures are used to control the mallets as well. Here is code for `leftmallet`. As with the `xylo` global, `left` has been initialized with the number of the left mallet's servo:

```
to leftmallet
wait 4
servo left 137
wait 2
servo left 110
wait 2
end
```

The construction of the state machine to step through the song is the most advanced aspect of the design. It is elegantly done and allows the main loop to interrupt the song at any point as the sensor changes.

In sum, this code exhibits significant, advanced programming practice. There is good use of procedural modularity as well as using global variables to name servo assignments. Cricket Logo does not have the equivalent of #define's or constants, so assigning globals to hold fixed values is a legitimate method.

IV. CONCLUSION

Courses like Artbotics put into stark relief the different talents and commitment levels that students bring to us. For the students who were excited about the material, willing to try something new, and who made ample time available to explore, play, and create, the course was gratifying. For some others, it was a struggle from which they could not easily hide.

V. FOR MORE INFORMATION

The Artbotics project web site is `artbotics.org`.

Fig. 2. *MUSIC: Maker of User-Synced Instrumental Compositions*

Fig. 3. Microprocessor-controlled xylophone from *MUSIC*

REFERENCES

[1] L. Akins and D. Burghardt. Improving K-12 mathematics understanding with engineering design projects. In *Proceedings of the 36th ASEE/IEEE Frontiers in Education Conference*, San Diego, CA, 2006.

[2] H. J. Kim, D. Coluntino, F. G. Martin, L. Silka, and H. A. Yanco. Artbotics: Community-based collaborative art and technology education. To appear in *Proceedings of SIGGRAPH 2007, The 34th International Conference and Exhibition on Computer Graphics and Interactive Techniques*, San Diego, CA, August 2007.

[3] L. Silka, H. J. Kim, F. Martin, H. Yanco, J. Beck, D. Testa, and D. Coluntino. Artbotics: The challenge of new partnerships. Presented at the 8th *Annual Conference of the Committee on Industrial Theory and Assessment (CITA)*, Lowell, MA, 2007.

[4] H. A. Yanco, H. J. Kim, F. G. Martin, and L. Silka. Artbotics: Combining art and robotics to broaden participation in computing. In *Proceedings of the AAAI Spring Symposium on Robots and Robot Venues: Resources for AI Education*, Stanford, CA, March 2007.