# DEPLOYING A LOW-COST VISUAL POSITIONING SYSTEM IN FOUNDATIONAL ENGINEERING DESIGN COURSES

Michael A. Vernier, Craig E. Morin, Patrick M. Wensing, Ryan M. Hartlage,
Barbara E. Carruthers and Richard J. Freuler
The Ohio State University

## Abstract

Although the concept of autonomous robot design projects has existed in engineering education for years as a tool for giving engineering students hands-on experience, in practice, the autonomy of these projects has been limited due to cost. Student programmers participating in these projects often have limited ways to interact with their environment autonomously, relying on low-cost sensors such as touch sensors instead of interacting with a high-cost camera-based positioning system. This not only limits the autonomy of the robot, but robs the student of valuable design and programming experience, since professional engineers often have access to tools such as the Global Positioning System (GPS).

Thus, it was desirable to build a low-cost positioning system that could be used to track the movements of student-built robots and to transmit position and orientation information. This system would not only aid in the autonomy of the student-built robots, but give the students hands-on experience interacting with such a positioning system, experience that would transfer to working on autonomous vehicles in industry.

Such a system was built, utilizing Nintendo Wii remotes as infrared cameras to track the movement of high-intensity infrared beacons mounted on student robots. Each of these Wii remotes tracked up to four LED locations in their field of view and conveyed the information to a C application running on a Linux machine. This application packaged the information and transmitted it to a Microsoft Visual Studio C# library, which grouped LED locations into robots' positions. A National Instruments LabVIEW application transmitted the resultant orientation and position information to student's autonomous robots via radio frequency communications. This system performed well and significantly improved the design experience for the students involved.

## Introduction

For years, the education of engineering students has been supplemented with lost-cost robot design projects [1,2,3]. However, due to cost restrictions, robots designed by students are typically limited by the type of information they can detect about their environment; although researchers in the field of autonomous vehicle design often have the budget to use elaborate camera-based positioning systems to track their robots, such systems are usually too expensive for student use. Instead, student robots often employ touch and on-board optical sensors for autonomous navigation along with stop-gap measures such as dead reckoning. Although teaching these techniques provides a valuable skill set to students, the application to the real world is limited, as professional engineers often have access to tools such as the Global Positioning System (GPS) to aid in autonomous outdoor navigation [4,5]. A low-cost camera-positioning system capable of tracking several robots, calculating their positions and orientations, and reporting this information over radio frequency would drastically improve the autonomous navigation of student-built robots. In addition, this would give students some experience working with an outdoor navigation system, skills easily transferable to a professional autonomous vehicle interacting with the GPS satellite network.

A low-cost camera positioning system was designed, using commercially available Nintendo Wii remotes as infrared cameras.

These remotes were mounted eight feet above a 112 square foot robot course and were used to visually recognize the position and orientation of infrared beacons mounted on four student-built autonomous robots. A C# library was written and used by a National Instruments LabVIEW [6] application to aid in the identification of robots, grouping the LED locations into clusters on the course's coordinate system. This information was then transmitted over a radio frequency network to the individual robots and was used by students to augment the autonomous navigation of the robot.

In this paper, we discuss the components used in constructing this positioning system, including the Wii remotes, the radio frequency transmission of information, and the multi-language software set used to determine the position and orientation of student-built robots in real-time. In addition, the effect of the low-cost camera-based positioning system on the educational experience of the students is evaluated.

**System Layout**

Figure 1 shows the overall design of the positioning system. Wii remotes collect information about the position of high-intensity infrared LED beacons affixed nine inches above the course surface on student-built robots. This information is transmitted over Bluetooth to a C application running on a Linux machine, where it is gathered and sent to a Microsoft Visual C# library. The C# library interprets LED locations as robots and converts pixels from the Wii remote image to inches in a global coordinate system. A National Instruments LabVIEW application interacts with the C# library and transmits location and orientation information to the student-built robots over a radio frequency network.

In total, seven computers were used to monitor and control the various aspects of the system. All sixteen of the Wii remotes were connected to a single computer. Two instances of the LabVIEW application were used on two separate computers that were used to control

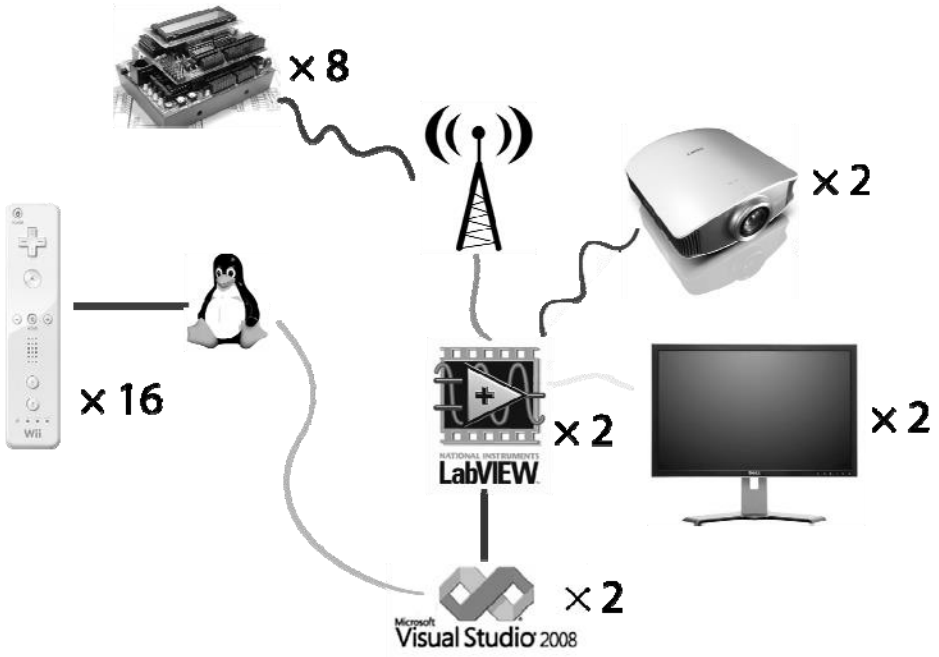


Figure 1: Positioning System Component Diagram.

each of the two courses used in competition. Two additional computers were used to project a birds-eye view of each course overlaid with rectangles representing the positions of the robots competing, and the last two computers were used to monitor the status of the system and handle any problems that might have occurred.

### Wii Remotes

Eight Nintendo Wii remotes were suspended eight feet above the surface of the 112 square foot "H"-shaped robot course in order to provide the resolution necessary for the tracking of the student-built robots. Figure 2 shows the rectangular-shaped coverage areas and positions of the Wii remotes suspended above one of the robot courses. These Wii remotes were purposely positioned such that a portion of the field of view of two adjacent remotes would overlap. This aided in the calibration of the positioning system. The height of the Wii remote locations was determined by the resolution (1024 x 768 pixels) and the viewing angle (35º by 45º) of the Wii remote camera. Intended for interacting with the Nintendo Wii gaming console, Wii remotes also had the ability to track the four brightest spots in their field of view and communicate via Bluetooth to transmit the point data to another system. This proved helpful in using the Wii remotes to track robot movement, since no additional image processing was needed.

The software system that controlled the Wii remote communication consisted of a Linux machine running a C program, which, in turn, called the specific C-based library, "CWiid" (pronounced: "seaweed") [7]. Linux was chosen as an operating system because its Bluetooth driver provided the ability to manually connect a specific Wii remote to a specific Bluetooth adaptor. Since two complete robot courses were used at the final competition, sixteen Wii remotes needed to be connected to a single computer. It was found that only four remotes could be reliably connected to a single USB Bluetooth adaptor, so multiple adaptors were required. Initial tests were conducted in

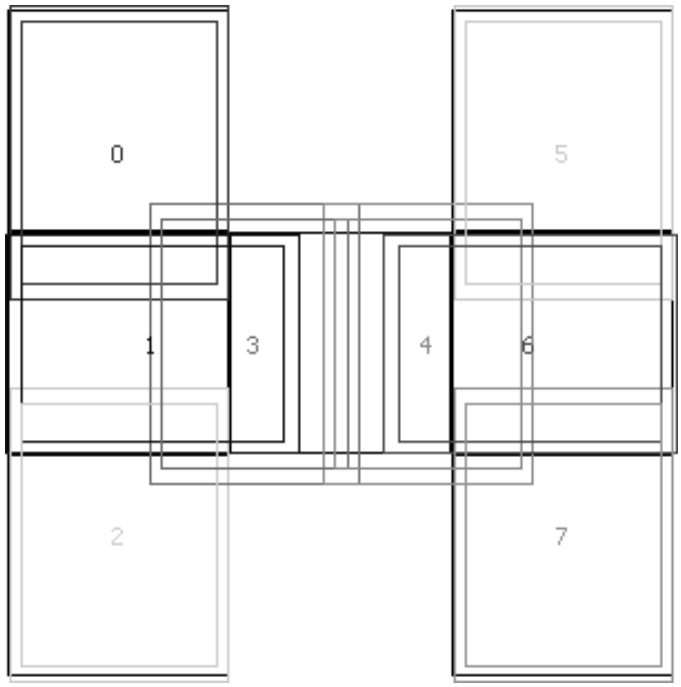


Figure 2: Wii Remote Locations.

Microsoft Windows XP using the BlueSoleil Bluetooth driver. After the successful connection of four Wii remotes, a fifth would not reliably connect to the single USB Bluetooth adaptor. This implementation was soon abandoned in favor of a Linux implementation since the BlueSoleil driver did not provide the ability to connect multiple Bluetooth adaptors to a single computer.

An interface was created in the C application to give other applications access to the remote data over a network socket connection. This interface allowed the other applications to query the system for the visible infrared points of an individual remote and to control the connections of the remotes.

Specifically, the "CWiid" C-based library was used to handle the low-level data processing and communication between the Wii remotes and the C application on the Linux machine. The library was modified to allow the manual connection between the remotes and the adaptors, but was otherwise left unchanged.

## Calibration

Before the calibration was performed for each of the Wii remotes, steps were taken to account for variances in the height of the high-intensity infrared beacons mounted on the student-built robots. Since the calibration was performed on the surface of the course, differences between the datum beacon height (nine inches) and the actual height (between eight and ten inches) caused a beacon passing through an area of overlapping images to appear as several closely spaced beacons. To correct for this, a scaling factor was used to adjust the image coordinates at each point acquired by the Wii remote before the following calibration was performed.

A multi-step calibration method was used to transform the image coordinate frames of each Wii remote into the robot coordinate frame. Ten infrared LEDs were placed in the course surface in the areas where the Wii remote images overlapped in order to provide the point correspondences required for calibration. Figure 3 shows the positions of the calibration LEDs on the course surface as well as possible misalignment of the Wii remotes. During the calibration, these LEDs were turned on one at a time to build a set of coordinates that corresponded to each calibration LED, as seen by each Wii remote. The two centermost LEDs were placed on the course equidistant from the physical center of the course so that an origin for the robot coordinate frame could be defined. The fields of view of individual Wii remotes were aligned starting from the center of the robot course and expanded radially outward. Calibration was simplified into three adjustments, two translational and one rotational, which defined the transformation of a point acquired by the Wii remote into a point representing the physical location of the LED on the robot course.

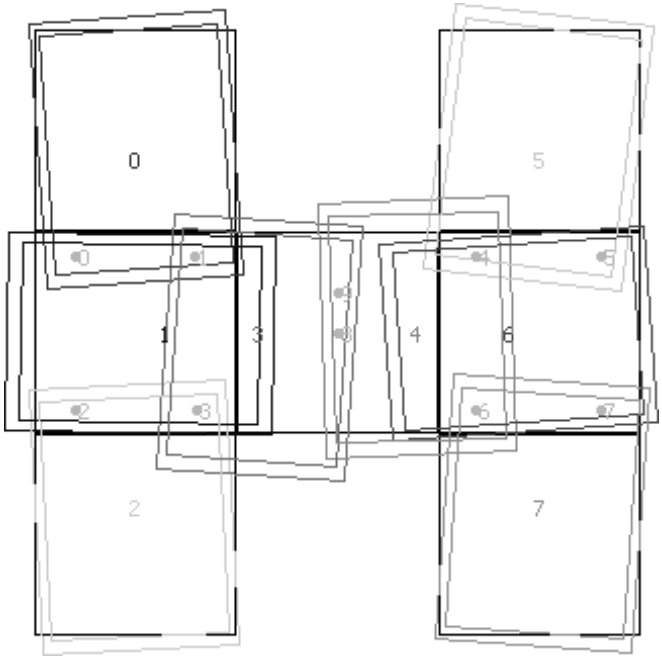


Figure 3: Calibration LED Locations.

To calibrate a Wii remote to an adjacent Wii remote that had been previously calibrated, the first of two point correspondences were used to translate one image so that the points were coincident. Figure 4 shows the field of view for

two adjacent remotes. In Figure 5, the first calibration step has been performed for the Wii remote on the right. A dot product was used to determine the angle needed to rotate the images so that both sets of points are coincident. The result of this rotation can be seen in Figure 6. Finally, the image is translated into the robot coordinate frame. The orientation specification is shown in Figure 7, and the final robot coordinate system is shown in Figure 8 with positions measured in 1/4 inch increments.

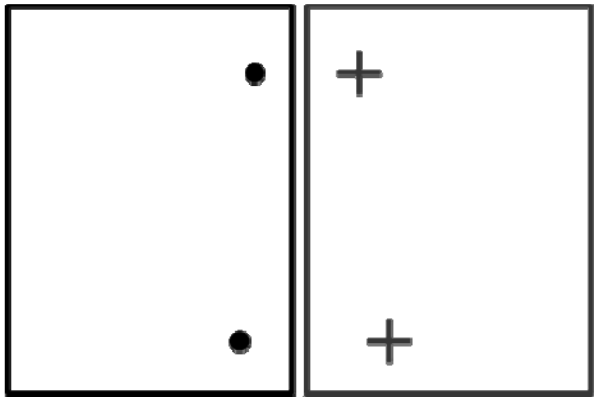

Figure 4: Two Wii Remotes before Calibration.

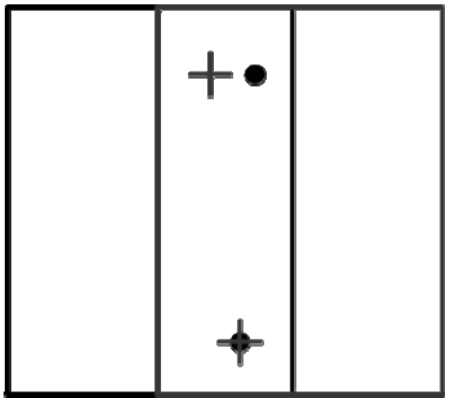

Figure 5: Two Wii Remotes after First Translation Step.

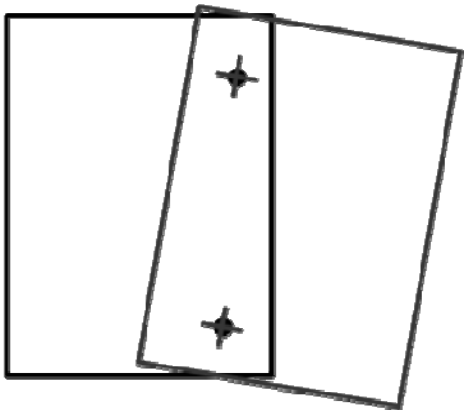

Figure 6: Two Wii Remotes after Calibration.

The first of the two center remotes was calibrated in a slightly different manner. Its image was rotated so that the line segment connecting the two center LEDs was vertical. The origin for the robot coordinate frame was defined by the midpoint of this line segment.

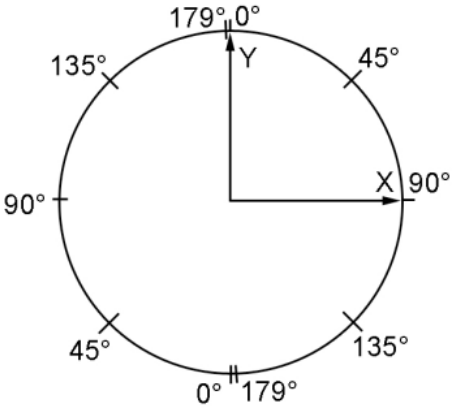


Figure 7: Robot Orientation Specification.

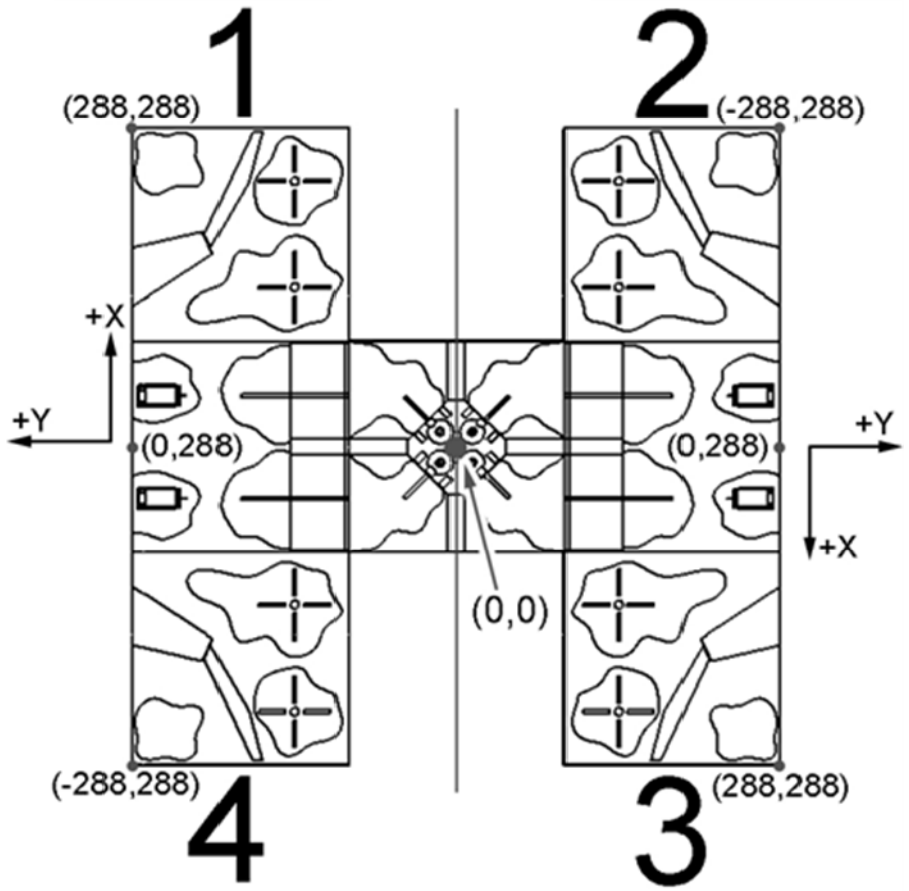


Figure 8: Robot Coordinate System.

### Software Robot Identification from LED Locations

At each update, Wii remotes were queried for the positions of any points visible in its field of view. These points were then transformed into the robot coordinate frame and processed to determine the position and orientation of the robot. First, groups of points were identified that were within one inch of each other. These groups represented single LEDs that were found on a beacon in one of the overlapping regions of the Wii remotes. These groups were replaced by the average of the points. At this stage, any point represented a single LED on a beacon.

Another pass through the point list returned pairs of points that were about six inches apart - the separation distance between the two high-intensity LEDs on the beacon. These pairs were averaged to get candidate robot positions. Candidate positions that were very close to one another were evaluated to determine whether robots in overlapping regions were generating errors that had passed the first grouping (i.e. multiple sets of the same data that were more than one inch apart). If this was the case, the data was grouped and averaged, resulting in an approximate robot location.

The resulting two points of the beacon were used to estimate the robot's position and orientation on the course relative to the course's origin.

### Data Transmission to Robots

A LabVIEW application was used to transmit the positions and orientations of the student-built robots via radio frequency. This application received the robot data from the C# library and transmitted it over an RS-232 serial

connection to a radio frequency transmitter. A level shifter was used to convert the ±12V RS-232 signal to 0-5V levels and then amplified to 0-12V to boost signal range.

The transmitter was capable of sending data at a rate of 2400 bits per second with a 434 MHz carrier frequency. Because only a single communication frequency was available, the channel was time multiplexed between each of the student-built robots. A custom network protocol was written to allow data packets to be unicasted to a single robot. The five-byte data packet contained an address number along with the other relevant data which identified the robot that was to receive the packet.

## Software Interface

A low-level driver was written in Motorola 68HC11 Assembly language so that the MIT Handy Board used by the students could receive and decode data packets sent by the transmitter. When enabled, the driver updated the robot data via the custom network protocol and kept only the most recent data in memory. Invalid packets and packets that were addressed to another competing robot were ignored by the driver. This driver also provided a mechanism to disable the students' robots upon completion of a competition run.

Because of the limited programming experience of the students, the robot's interface to the positioning system was designed to be simple. The students were given the ability to choose the address number of the data packets assigned to their robot, to enable or disable the radio frequency receiver, and receive the latest data from the positioning system. The data packet address could only be changed before the receiver was enabled for the first time. This ensured that the students would not attempt to receive another robot's data in an effort to intentionally interact with it. A set of global variables was given to provide the students with an easy way to access their robot's position and orientation as well as any other data for their competition run.

## System Results

The resulting low-cost camera-based positioning system performed well, with a location resolution of 1/4 inches and an orientation resolution of 1°, with 180° ambiguity. The system transmitted position and orientation information to the robots at a rate of four times per second, including proximity warnings, indicating possible collisions with other robots. These warnings indicated collisions from two and three feet away and also at the time of impact. In addition, the overall cost for the positioning system for two courses totaled roughly $700, well within the desired "low-cost" design specification. This was roughly half the cost that would have been spent on a single high-resolution camera capable of visually tracking robot positions and orientations to the desired accuracy.

Figure 9 shows the final interface for the navigation system as seen by the students and audience, displaying four robot locations overlaid with the layout for the robot course. This interface was designed to show, large-scale, the behavior of the student robots during competition to spectators. However, it proved helpful during the debugging process for students.

The navigation system also added significantly to the educational experience of the students. Students were able to improve the autonomous navigation of their robots by receiving communications from a realistically modeled system, learning different software design techniques. In addition, in interacting with the low-cost camera-based positioning system, students were better able to understand the current abilities and limitations of the field of autonomous vehicle control, where the use of GPS is widespread.

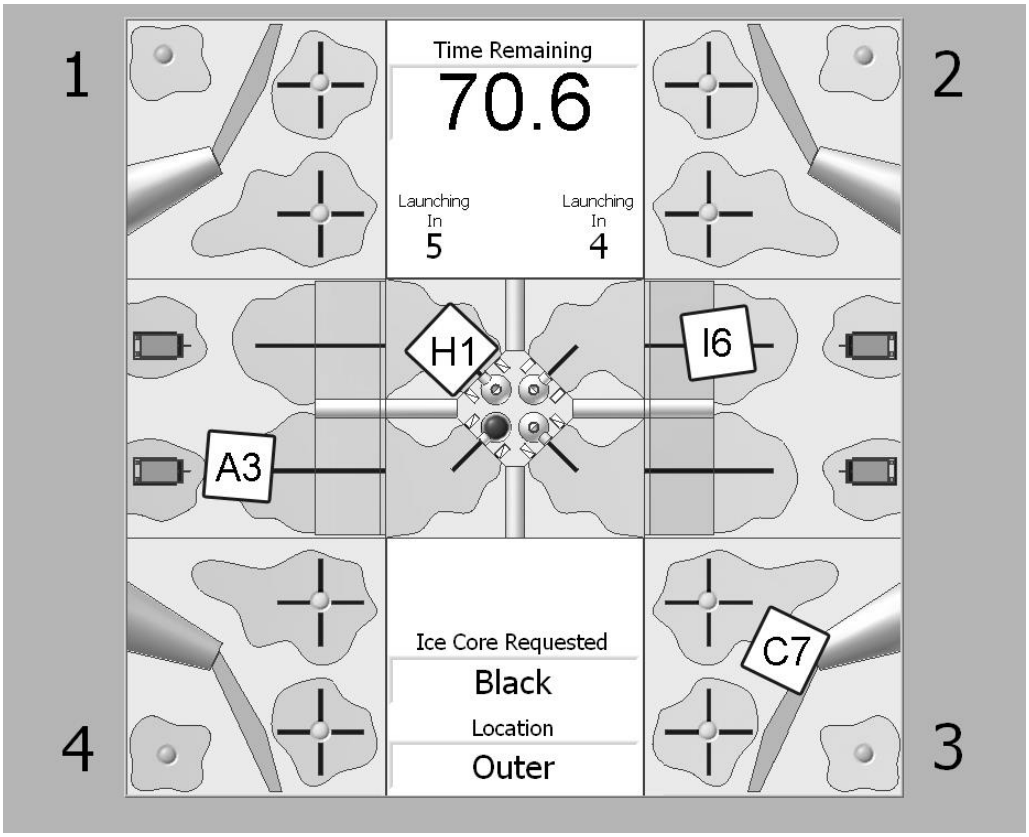


Figure 9: Visual Interface for Positioning System Showing Position of Four Robots.

## Concluding Remarks

Overall, the use of a low-cost camera based navigation system supplemented the design experience for first-year engineering students programming autonomous robots. This system worked by collecting the images of high-intensity infrared beacons, as seen by Nintendo Wii remotes mounted above the robot course. This visual information was converted to numerical information conveying the position and orientation of student-built robots and transmitted to the robots via radio frequency communications. From interacting with this low-cost camera-based positioning system, students learned different techniques of programming autonomy because they had different tools with which to interact. In doing so, they learned how to interact with various positioning systems, such as the Global Positioning Systems commonly used to aid autonomy in professional projects.

## References

1. Freuler, R.J., M.J. Hoffmann, T.P. Pavlic, J.M. Beams, J.P. Radigan, P.K. Dutta, J.T. Demel, and E.D. Justen, "Experiences with a Comprehensive Freshman Hands-On Course – Designing, Building, and Testing Small Autonomous Robots", *Proceedings of the 2003 ASEE Annual Conference & Exposition*, June 2003.

2. Beer, R.D., "Using autonomous robotics to teach science and engineering", *Communications of the ACM*, vol. 42, issue 6, 1999, p. 85.

3. Verner, I.M., "Robot contest as a laboratory for experiential engineering education," *Journal on Educational Resources in Computing (JERIC)*, v. 4 issue 2, 2004.

4. Borenstein, "Mobile Robot Positioning – Sensors and Techniques", *Journal of Robotic Systems*, vol. 14, issue 4, 1997, p. 231.

5. Sukkarie, "A high integrity IMU/GPS navigation loop for autonomous land vehicle applications," *IEEE Transactions on Robotics and Automation*, vol. 15, issue 3, 1999, p. 572.

6. National Instruments Corporation, "NI LabVIEW", Internet: http://www.ni.com/labview/, accessed 19 Mar 2009.

7. Smith, D., "CWiid", Internet: http://abstrakraft.org/cwiid/, accessed 6 Feb 2009.

## Biographical Information

Michael A. Vernier is a Graduate Teaching Assistant for the OSU Fundamentals of Engineering for Honors (FEH) Program where he teaches the laboratory portion of the three-quarter FEH engineering course sequence and develops course materials. Mr. Vernier earned his BS in Electrical and Computer Engineering (2007) from The Ohio State University and is currently a Master's Candidate in Electrical and Computer Engineering at The Ohio State University, researching control system design for autonomous vehicles.

Craig E. Morin is now a Design Engineer with MindWare Technologies in Columbus, Ohio where he develops medical research equipment. Previously, he was a Graduate Teaching Associate with the OSU Fundamentals of Engineering for Honors (FEH) Program where he taught labs and developed course materials. Mr. Morin earned his BS in Electrical and Computer Engineering (2004) and his MS in Biomedical Engineering (2008), both from The Ohio State University.

Patrick M. Wensing is a senior honors student in the Department of Electrical and Computer Engineering (ECE) and has served as an Undergraduate Teaching Assistant for the OSU Fundamentals of Engineering for Honors (FEH) Program. He is also an undergraduate research assistant, working in the area of robotic locomotion. Mr. Wensing will graduate with his B.S.E.C.E. from The Ohio State University in June 2009.

Ryan M. Hartlage is a dual-major senior honors student in the Departments of Electrical and Computer Engineering (ECE) and Mathematics at The Ohio State University. He is also an Undergraduate Teaching Assistant for the OSU Fundamentals of Engineering for Honors (FEH) Program for the three-quarter FEH engineering course sequence. Mr. Hartlage received the FEH Most Outstanding Undergraduate Teaching Assistant Award in June 2008 and 2009. He will graduate with his B.S.E.C.E. in June 2010.

Barbara E. Carruthers is an Aeronautical and Astronautical Engineering student at The Ohio State University and an Undergraduate Teaching Assistant for the OSU Fundamentals of Engineering for Honors (FEH) Program. She also works as a student research assistant at the Aeronautical and Astronautical Research Laboratory at Ohio State, studying engine test cell design. Ms. Carruthers will graduate with her B.S.A.A.E. from The Ohio State University in June 2010.

Richard J. Freuler is the Faculty Coordinator for the Fundamentals of Engineering for Honors (FEH) Program in the OSU Engineering Education Innovation Center, and he teaches the three-quarter FEH engineering course sequence. He is also a Professor of Practice in the Aerospace Engineering Department and Associate Director of the Aeronautical and Astronautical Research Laboratory at Ohio State. Dr. Freuler earned his Bachelor of Aeronautical and Astronautical Engineering (1974), his BS in Computer and Information Science (1974), his MS in Aeronautical Engineering (1974), and his Ph.D. in Aeronautical and Astronautical Engineering (1991) all from The Ohio State University.