A ROBOTIC AIR HOCKEY OPPONENT

M. Lakin
Department of Computer Science and Industrial Technology
Southeastern Louisiana University
Hammond, LA 70402
(985) 549-2189
MatthewHLakin@gmail.com

ABSTRACT
This paper discusses the process undertaken to create a system to play air hockey autonomously against a human. Such an application presents challenges in both the robot's sensing of the environment, the design and set up of the robot, and the programming of how to appropriately react to the sensor data. How these issues were encountered and dealt with are discussed along with how the project might be adapted to start learning to perform better using existing artificial intelligence methods.

INTRODUCTION
The challenge is to make a robotic system to play air hockey against a human on a physical air hockey table. To accomplish this, it is necessary to be able to sense the environmental conditions, to process the appropriate response to these conditions, and to have some way of affecting the environment in order to achieve the system's goal. For these requirements the system was given a camera to sense the environment, programming to plan appropriate responses, and a robotic arm to carry out these responses.

BACKGROUND
Researchers have been trying to perfect autonomous robots for the past decades. [3] As technology has improved, autonomous robots have become better at performing tasks on their own, yet still suffer from such problems presented by an open world. [3] The problem associated with this being that it is difficult to design a robot that is capable of reacting appropriately to all the events possible in an open environment. This application however can be considered a closed world due to the ability to safely assume that all possible occurrences on the table can be accounted for. In addition to this, operating on a bound two-dimensional plane reduces the room for error associated with tracking objects and the movement of robotic limbs. Also it is safe to assume that consistent lighting is present, which makes observations easier for sensors. These qualities make such an application a popular project for those attempting to build autonomous robots.

METHODOLOGY

The Camera
For the robot to be able to react to incoming shots appropriately the system needs to know where the puck is located at any given time. For this task, a color blob tracking camera is appropriate. Color blob tracking systems are designed in such a way that they look for regions of an image for a specific color, and can return the location of these regions. [2] Used in this system was a

surveillance camera that can track sixty frames per second connected to a Cognachrome Vision System Board from Newton Labs, which contains all the appropriate hardware and software to track objects. [5]

The vision board can be programmed to search for a given color and return its X and Y coordinates of the largest blob's center when requested. Up to three objects of the same color can be tracked, and it can store up to three different colors in its memory. The vision board includes functions that allow one to set the color for each channel, which is typically done by holding the color in question in front of the camera before sending the command to save that color in memory to the vision board. Once set there is the option of broadening or narrowing the range of what the system considers that target color.

For example, if the vision board has difficulty detecting a red ball as the desired object to track, the range of what it considers the red of that ball can be broadened to include both the light red section of the ball close to a light source and the dark red of the shaded area. Broadening this range too much, however, runs the risk of having the system pick up undesired objects as matching the color in question. In the aforementioned example this may result in red-orange or red-purple balls also being interpreted as the object it's looking for. To help with debugging, vision board can output to a TV screen white where it sees regions of the chosen color, and black everywhere else. This display helps with making sure the camera is tracking the correct color, and how clearly it sees it.

Although useful for this application, Color blob tracking systems have their own set of limitations. [2] Even though to the human eye colored objects are easy to keep track of even as the lighting changes, the blob tracker has a set window of what is or isn't the color in question. Shiny or glossy objects are poor subjects to track due to how the perceived color of them might change as the angle of view changes. Slight changes in lighting, such as a shadow passing over the object, will often change the color of the object enough for the vision board to lose sight of it.

Fortunately this application is indoors with consistent lighting. After some testing it was found that neon colors made for the easiest tracking. This is due to them being so dissimilar to the other colors already present on the air hockey table. The original red color of the puck presented problems due to the same color being found elsewhere on the board. A neon pink circle was taped to the puck to resolve this issue. At present, however, it can still lose track of the puck if it is against the left wall where the shadow of the rim spoils its color slightly. In the future portable lights could be used to solve the shadow issue, and also give it consistent lighting if the hockey table needed to be moved to another location.

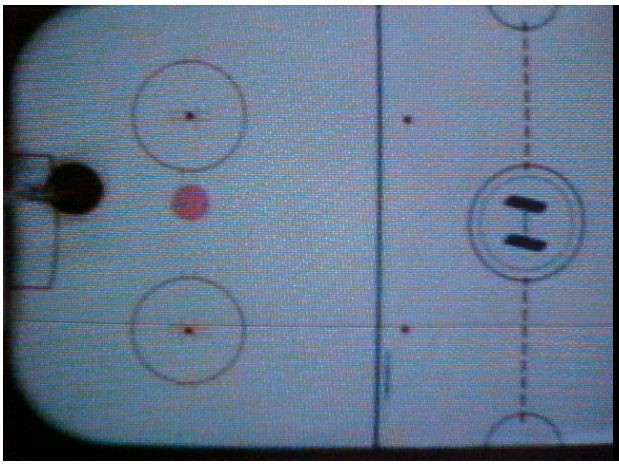

Figure 1. The view as seen by the camera.

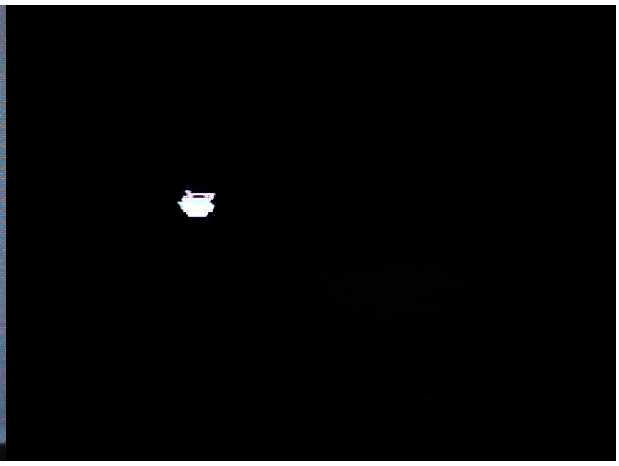

Figure 2. The debugging window of the vision board

Figure 2 gives an example of this debugging output next to what security camera outputs when connected to a TV. The object being tracked in this case is the puck located in the mid left portion of the screen. Notice that in the debug window the blob the vision board perceives as the object in question is not the complete circle it is in actuality. This may be due to slight differences in color, which could lead to what the vision board interprets as being the center of the object being a little off from its true location.

The Robot

The mechanical arm was built out of mostly Vex parts. [7] Similar to an erector set, the Vex parts made it easy to construct a swinging arm that moved an air hockey bumper left and right in front of the goal. Before a more complex set up with an elbow or wrist joint can be implemented, it was important to first master a configuration with just one joint. With one joint, the robot arm can swing the bumper in an arch in front of the goal and guard it from incoming shots, but lacks the ability to effectively return the puck in an attempt to score points.

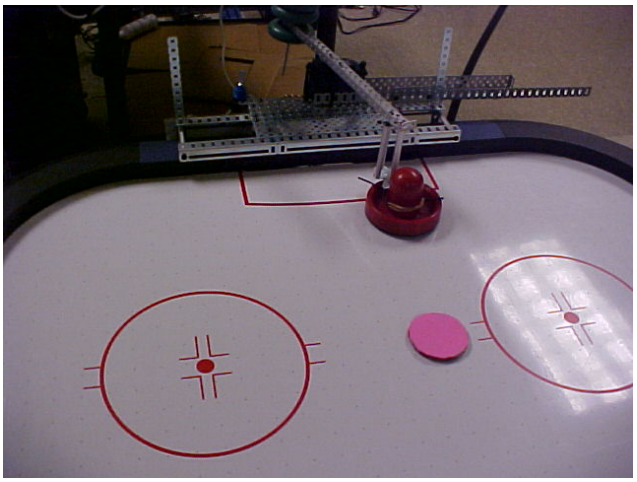

Figure 3

If the vex could be thought of as the skeletal structure of the robotic arm, then the muscle is the servo. A HS-805BB Hitec servo was used in this application both for its high torque and potential speed. Due to both of these qualities, it was important to implement kinematic constraints on the servo movement inside the program itself, or else the servo might break the table if instructed to move to far. [2] The servo by itself is not designed to receive instructions from the computer. To interface the two a SSC-32 board was used. Similar to the vision board, the SSC-32 board communicates to the computer through a serial port. Commands consist of a string of characters usually instructing what desired numerical position the SSC-32 board should power the servo to move to.

The Programming

A simple method of having the robot intersect incoming shots was to set it up in such a way that it would move the bumper as close as it could to the position at which the puck was last recorded. This way if the puck tries to enter the arch of reach the robotic arm covers in front of the goal, the blocker should already be there to oppose it, provided the system can react in time. This could be accomplished by finding a function that takes the X and Y values of the puck's position and plugging it into some function that returns the position where the servo should be instructed to move. The way the camera views the board, the X axis runs the length of the table, while the Y axis runs from left to right as viewed from a person standing behind a goal. The function was made simpler yet still effective by taking into account only the Y value of the puck position. Discovering this function was a matter of setting the puck at different intervals along the Y axis, finding the position number that would move the blocker in between the puck and the goal, and then recording both of these values. After enough of these values have been collected, the function was found using the least squares method to fit a line to the data points. The best fit line ended up being cubic, which makes sense because small change in Y values see a bigger change in servo position near the ends of the arc the arm makes than it does in the middle, so the line the function would graph should change concavity twice.

Once a function was found and implemented in the robotic system could now play against a human, if only defensively. After tweaking the structure of the robotic arm to maximize response time and reduce room for unwanted movements, the robot became challenging for most players to score points against. With this behavior it was observed that it handled shots whose Y component of velocity is small very well. The only few shots that managed to make it past the robot were fast shots that bounced off a side wall a moment before it went in the goal. Perhaps this could be due to the servo still being in the process of moving to its last instructed position when new instructions are given. This is not a problem with shots whose rate of change in Y values is relatively small because in this case the distance between where the servo was last instructed to move and where it receives new instructions to go can be covered quickly.

Another issue arose from the implementation of this method. If the table or camera were ever moved, the function for finding the desired servo position became invalid. Such movements are unavoidable should the table ever be relocated for demonstration purposes. To overcome this, a calibration routine was implemented. When the system is activated, it now looks for the position

of two color markers at fixed locations on the two corners of the board nearest the robot. The function determines what X and Y offsets would be needed to add to one corner to make it the origin, and what angle would rotate the other corner to line both of them up with the Y axis. After determining this, all incoming X and Y coordinates coming in from the camera are offset by the offsets, then rotated by this angle. With this in place, what the system views as being in one place on the board will still be seen as being in the same place no matter how the board is shifted, provided that the camera can still see the important parts of the board and the system has a chance to recalibrate itself, making the system much more reliable and consistent.

FUTURE WORK

The next phase of this project is to use it as a means of researching ways of machine learning. It has been proven that an automated system can compete fairly well against a human player in air hockey. Better yet would be to set it up in such a way that it learns from experience. This would help in contributing to the advancement of artificial intelligence systems.

Others have achieved a level of success implementing such AI methods as neural networks or fuzzy logic to power the reasoning behind air hockey robots.[5, 7] Perhaps a better suited method to apply to this application would be a genetic algorithm such as discussed in [4]. Designed to closely resemble evolution found in organisms, genetic algorithms take a population of possible solutions and then scores each one according to how close it gets to the desired answer. The next phase would be to “breed” or combined high scoring solutions with other high scoring solutions to produce children who are made up of parts of the parents. There would also be a chance for these children to be mutated, or slightly altered in order to ensure that something new is tried other that what already exist in the population. These children are then put into the population and the whole process is repeated again and again. Eventually the population converges to the optimal solution to the given problem.

Such an algorithm can be applied to the technique described earlier where the Y component of the puck’s position is plugged into a function to get the desired position of the servo. After generating random functions to be use, the system could rate each one by how effectively it blocks incoming pucks. Based on these scores, the system could select functions to be bred. The components to be combined when breeding could be the coefficients of each cubic function. Combining would be as simple as swapping a few coefficients to make a new function. Mutating the function would involve offsetting one or more coefficients. After enough iteration the algorithm should converge on the most optimal solution.

CONCLUSION

To appreciate the challenges associated with this project, it is helpful to consider a counterexample. If the task was to implement the artificial intelligence of a computer opponent in a simulation air hockey video game, there would be far fewer issues. There would be no error in the sensor input, because all the variables such as puck position are known and managed by the system. There would also be no way for incoming shots to move faster than the computer opponent could react unless the programmer intentionally places limits on this. The system could also accurately predict the path of the puck after it is struck, because the path is dictated by the system itself. Real physical robotic applications do not benefit from these advantages, however.

What the sensor sees and what is actually going on are not always the same. There are limits on how quickly servos and motors can react, and how a system predicts its actions would affect the real world, and what those effects actually are don't always match up.

The implementation of this project proved to be a wonderful experience in contrasting the different issues associated with autonomous robotic design and implementation as opposed to conventional programming to be implemented exclusively on a computer. As an introduction to robotics one could attempt an open world project such as a walker designed to navigate unknown terrain, but a closed world application such as an autonomous air hockey robot would be much more appropriate for a beginner to challenged yet not overwhelmed with the amount of issues.

REFERENCES

[1] Bishop, B., Spong, M., Vision-based control of an air hockey playing robot, Control Systems Magazine, IEEE, (3) 23-32, 1999

[2] Jones, J., Robot Programming: A Practical Guide to Behavior-Based Robotics, NY, McGraw-Hill, 2004

[3] Murphy, R., Introduction to AI Robotics, Cambridge, MA: The MIT Press, 2000.

[4] Negnevitsky, M., Artificial Intelligence: A Guide to Intelligent Systems, Harlow, England, Addison Wesley, 2005

[5] Newton Research Labs, http://www.newtonlabs.com/, 2008

[6] Park, J., Partridge, C., Spong, M., Neural network based state prediction for strategy planning of an air hockey robot, http://citeseerx.ist.psu.edu/viewdoc/summary?doi=10.1.1.32.5032, 2008

[7] Vex Robotics, http://www.vexrobotics.com/, 2008