AC 2010-879: ROBO-BILLIARDS: A GAME TO UNDERSTAND ADAPTIVE BEHAVIOR OF MIDDLE SCHOOL STUDENTS

Kurt Brown, University of South Alabama

Kurt Brown was a graduate student in the School of Computer and Information Sciences at the University of South Alabama. He recently completed a thesis in the area of adaptive autonomous robotic systems.

Michael Doran, University of South Alabama

Dr Michael Doran is a Professor in the School of Computer and Information Sciences at the University of South Alabama. He is also the Coordinator of the CS program and the Assistant Director of the Honors Program. His research interest includes robotics, real-time systems and engineering education.

David Langan, University of South Alabama

Dr. David Langan is a Professor in the School of Computer and Information Sciences at the University of South Alabama. His research interests include expert systems and engineering education.

Tom Thomas, University of South Alabama

Dr Tom Thomas is an Associate Professor of Electrical and Computer Engineering in the College of Engineering at the University of South Alabama. He is also the Director of Graduate Studies in the College of Engineering. His research interest includes robotics and engineering education.

Robo-Billiards: A Game to Understand Adaptive Behavior of Middle School Students

Abstract

Robo-Billiards is a timed competition in which middle school students remotely control robots to place billiards balls in pockets. Points are awarded in the game based on how the balls are combined in the pockets. If a ball is accidently knocked off the game surface it is out of play and the opportunity for the overall maximum score is lost. It was observed that once a ball was knocked off the surface, creating a fault, that students tended to panic and seldom adjusted their strategy to achieve the maximum score with the remaining billiard balls. The capability to adapt in real-time is what makes humans successful in chaotic environments. In order for computer systems to be successful, they too must adapt in real-time to unplanned events. Two problems for adaptive systems are how to represent knowledge and how to use that knowledge to make adaptive decisions. With this research we examined the use of scripts (that specified robot ball selection and placement) to represent knowledge, and heuristics and dynamic scripting to make adaptive decisions. In this research, an Autonomous Robo-Billiards Opponent (ARBO) was developed to play Robo-Billiards. As a baseline, ARBO was tested to determine if it could execute a script in less than five minutes and still earn the maximum score. Then faults were simulated while ARBO was executing a script to test if the system could adapt and earn the optimal score with the remaining balls. A potential use of this system is to compete against middle school students. Two immediate benefits would be to: (i) stimulate interest in STEM disciplines and (ii) demonstrate adaptive behavior when a fault occurs.

Introduction

Humans must adapt in real-time to new problems and situations. The ability to adapt well to problems makes humans successful. For intelligent software systems to succeed in human environments, they too must adapt in real-time to unexpected problems. The focus of this research is to explore how software systems can adapt in real-time in dynamic and chaotic environments to lead to successful results.

Expert systems have been developed that exhibit the same sophistication as humans to adapt in real-time [1]. These systems are successful in environments that have constrained complexity. Because of real world complexity, it becomes necessary to do research with software systems in constrained environments. By constraining the environments, researchers reduce the number of variables in their experiments and increase confidence in their simpler results. Games provide environments that are rich enough to answer research questions that relate to the real world, yet are constrained enough to support experimentation.

Our interest in developing an adaptive real-time system began with our observation of a game called Robo-Billiards. In Robo-Billiards, participants use radio controllers to maneuver a robot that they have constructed. The objective of the game is to manipulate the robot's movements to place the standard sixteen billiard balls into pockets located at the corner of a playing field.

The playing field in Robo-Billiards is a 4 x 4 foot sheet of carpet-covered plywood. Four coffee cans are placed at the corners of the playing field to serve as the pockets where balls are placed. Before each individual competes in the event, his or her robot must completely fit within a 15-inch square at one end of the playing field. All the numbered billiard balls are racked in increasing numerical order at the opposite end of the playing field. The cue ball is placed on the playing field, between the robot and the racked balls.

Once the event begins, a participant is given five minutes to place all sixteen balls in the corner pockets. Points are awarded to a participant based on the following rules:

- **Rule 1:** One point for each ball placed in a pocket (4 points max per pocket)
- **Rule 2:** One point for each matching pair in a pocket (2 points max per pocket). A matching pair is a striped and solid color ball with the same color.
- **Rule 3:** One point for two or more striped balls in a pocket
- **Rule 4:** One point for two or more solid colored balls in a pocket

The cue ball serves as a "wild card." Therefore, after placing the cue ball in a can, a participant can let it become any ball that he or she desires to achieve a maximal score. Using these rules, a maximum of 31 points can be achieved. If two participants have the same score, the participant with the fastest time is the winner.

While observing participants compete in Robo-Billiards, we noticed that their actions followed a script. Hence, each task participants performed corresponded to a statement in their script like "capture cue ball" and "place cue ball in closest pocket." We also observed that participants usually used the same script during all their runs. Periodically, participants would introduce faults by misguiding their robots and knocking a ball from the playing field. After a fault has occurred, most participants would abandon their script for a more simplistic one. An example of a script simplification might be to replace a statement like "capture cue ball" with "capture the closest ball." By simplifying their original script, participants typically earned a sub-optimal score. A better solution is to reorganize the script so that the maximum number of points, based on remaining balls, can be earned.

Strategy games such as Robo-Billiards can help to engage students in activities that are fun and support STEM concepts. As observed in the student behaviors, the most successful results occur when a clear and defined plan (algorithm) is used to form the necessary script. Even in the face of a fault, it is the ability to adapt to the new circumstances that allowed further success. The robot's design likewise impacts the potential STEM learning. Participants in the Robo-Billiards contest created designs of the physical robot that were quite varied and subsequently impacted their script. For example, the designs varied on the number of balls selected at any time in the run. Most participants selected one or two balls, but others tried more, as many as five or six balls at a time. It was observed that the two ball selection approach yielded the best results. Hence, for our research, we designed and built a robot to select two balls at a time. It is this reflective learning in both design and implementation that is often discussed as central to STEM education [2], [3].

Based on these observations, the following research questions arose. First, can an Autonomous Robo-Billiards Opponent (ARBO) be constructed to compete in the game of Robo-Billiards?

Second, can the system adapt to faults by using heuristics and scripts? In other words, can the ARBO system succeed where humans were frequently observed to fail?

Motivation

The proposed research topic is important for several reasons. Mobile robotic competitions have dramatically improved research and education in the field of robotics [4]. Many competitions are similar to Robo-Billiards, requiring balls to be captured and placed in a goal. However, in the past, Robo-Billiards has been a competition involving only human competitors. With our research, we hope to inspire a new competition where autonomous real-time systems compete in the game. This research will provide a framework for future researchers who may follow in our footsteps.

Another contribution of this research would be increased human performance. Previous research has demonstrated that real-time robotic systems can be used as a tool to improve human performance [5]. From observation, humans are not typically capable of modifying their scripts on the fly in order to achieve an optimal score once a fault has occurred. By demonstrating the dynamic modification of scripts to humans, the system may aid in improving their adaptive capabilities.

The motivation for this research was to determine the capacity of real-time systems to compete with humans. Over the years, research projects have demonstrated that some real-time systems can compete successfully with humans (albeit with limitations). Some projects have demonstrated that real-time systems do not yet successfully compete with humans. How would the ARBO system compare with other projects?

Scripts

In 1977 Psychologist Robert Abelson and Artificial Intelligence researcher Roger Schank published their book Scripts, Plans, Goals and Understanding [6]. The book examined the convergence of their two fields of study, which came to be known as Cognitive Science. In their writing, Schank and Abelson defined a script as a "a structure that describes appropriate sequences of events in a particular context." Schank and Abelson explained how people make use of scripts in their daily lives in order to properly respond to certain events and situations. As an example, they provided their famous restaurant script. In the restaurant script, a customer might enter the restaurant and wait for a waitress. The script might end with the customer exiting the restaurant [6]. Similar to a person making an order in a restaurant, an individual who plays the game of Robo-Billiards makes use of a script as well.

In a study, 28 professional actors and 28 undergraduates with no theatrical experience were given scripts to memorize. The study concluded that the 28 professional actors were more successful at learning the script because they adopted the character's perspective and summarized scenes more [7]. Similar to the actors in the study, competitors in Robo-Billiards also have to remember scripts in order to successfully compete in the game. In addition, the competitors in Robo-Billiards that we observed were more like the novice actors in the study. The competitors appeared to remember their scripts in a rather robotic fashion instead of

attempting to summarize the script. It is hypothesized that this robotic memorization of scripts is the reason why many competitors were not able to adapt whenever a fault occurred. If the novice actors in Noice's study experienced an unexpected event while performing a script, perhaps a prop falling down, they were less likely than the professional actors to continue the performance without acknowledging the event.

Researchers performed a study of online learning in commercial computer games. In the study, the term dynamic scripting was defined. Dynamic scripting involves using an adaptive rulebase for generating the artificial intelligence of a game as it was played. In the study, the researchers concluded that dynamic scripting was superior to manually designed tactics. The study also concluded that dynamic scripting can be successfully when applied to commercial computer games [8].

Simulation

During the course of this research serious consideration was given to simulating a Robo-Billiards environment and performing all experiments in simulation. Even though a Robo-Billiards environment is much less complex than other real-world environments, there are still numerous hardware variables that would make accurately simulating the environment very difficult. For example, it was observed that wheels of the robot sometimes pulled in one direction whenever the system was attempting to drive straight. Even after adjusting the wheels, predicting which direction and how aggressively the wheels would pull in one direction proved to be a daunting task. Because of these and other hardware-related variables, it was decided that simulation would not be a plausible approach to conduct the entire experiment. However, it was decided that simulation might be a good technique to test navigational strategies to capture balls. This portion of the simulation allowed us to test a given dynamic script in a "perfect" world free from hardware variance. This simulator did help to determine a successful strategy for the needed script.

Implementation and Results

This section will discuss the various scripts developed as part of this research. A set of baseline scripts to achieve the maximum score, without the consideration of faults, were defined. These scripts were first tested using the simulator described above. Those tests showed that each of the scripts would yield the maximum score. The implementation on the physical robot would now be used to confirm that the time deadline would likewise be met.

Based on the initial tests it was observed that faults did not occur naturally while using the conservative movements/selection for the robot. In order to test the dynamic scripts, a fault had to be artificially introduced into the system. This was accomplished by interrupting the robot execution of the script, removing ball(s) from the environment, and then informing the script of the fault. This need to create a fault was due to the conservative behavior of the robot. This behavior was necessary due to hardware limitations such as a limited accuracy of the vision system, and communication delays.

Baseline Scripts

Three baseline scripts were created and carried out. The reason for this was to determine if the task could be successfully completed in less than five minutes without the introduction of a fault. It was also necessary to consider the impact of the distance traveled by the robot on the time to complete the task. This led to what was defined as the "Good Script" which used what was calculated (by physical measurements of the environment) to be a minimal travel-distance that achieved the maximum score. The Good Script, intuitively would lead to the best time, however, it did not take into account recovery from a fault that might occur. A variation on this Good Script might include saving the cue ball for later recovery. The first step of the Good Script was to place the cue ball into the target pocket since the cue ball is the first ball reached. By saving the cue ball until later it can be placed to maximize the score after a fault has occurred. This modified script is called the "Retain Cue Script". Based on observation of human participants, it was also determined to define scripts where the distance traveled was greater than the minimum. The same combination of ball placement was used as in the Good Script but different target pockets or selection order of balls was used. This led to a greater distance traveled but still solved the task in less than five minutes while achieving the maximum score. This script was called the "More Distance Script".

Baseline Results

Each script was executed three times successfully. The average run time was recorded for the Good Script, Retain Cue Script and the More Distance Script. As the table below shows, each run time is less than the five minute limit, and as expected the Good Script has the best times for all runs, with an average of three minutes sixteen seconds. Also, recorded was the distance traveled by the robot in pixels. The use of the pixels to measure distance was acquired from the vision system. The actual measurements for distance in pixels confirmed the calculations based on physical measurements used in the Good Script's design.

	Distance in pixels	Average Time (mm:ss)
Good Script	3169	3:16
Retain Cue Script	3546	4:06
More Distance Script	3226	3:29

Dynamic Scripts and Results

A dynamic script will be one that is modified after the execution has begun. In this research, the occurrence of a fault will motivate the need for a dynamic script. To achieve the most flexibility in the presence of faults the Retain Cue Script was selected. With this script in use, five tests with a variety of faults were carried out. For the first three tests only one ball was removed from the environment to introduce the fault. For the last two tests, two balls were removed. The instance that the balls were removed to create the fault was selected to provide the most interesting possibilities for adaption of the script. As the table shows, each new script was still able to complete the task in less than five minutes, and in each case the maximum score possible, with the remaining balls, was reached.

Fault	Distance in pixels	Time (mm:ss)
Fault 1 (1 ball)	3558	3:44
Fault 2 (1 ball)	3599	3:48
Fault 3 (1 ball)	3603	3:50
Fault 4 (2 balls)	3727	3:58
Fault 5 (2 balls)	3623	4:08

Summary

In this work, we used the observed behavior of middle school participants playing the game to motivate the research. Subsequently, this research created the ARBO system which (i) follows a script without creating faults to solve the Robo-Billiards game in less than five minutes and (ii) allows for dynamic scripts to respond to faults when encountered. The dynamic scripting achieved the desired effect of maximizing the score in the presence of a fault. The actual heuristics for the dynamic scripting are described in great detail in Brown [9].

In this project, the ARBO system did not compete directly with humans. However, based on the data produced by the ARBO system and previous observations made of human competitors in Robo-Billiards, it was realized that the ARBO system would be relatively competitive. Due to hardware limitations a conservative approach was taken on the movements of the ARBO system. As a result of these conservative movements, the normal operation of the ARBO system always resulted in a perfect score, but the robot moved and reacted slowly and thus would lose to a well trained human participant.

A simulator was also created in the course of this research. This tool provided valuable insights into the behavior of the scripts needed to play the game. It is assumed that this tool would likewise serve as a valuable resource to future human participants to test their scripts before using an actual robot. Currently, when humans cause a fault, their ability to adapt their predetermined script is often flawed.

A final hope for this work is to stimulate interest in the STEM disciplines by furthering interest in this game. This broader impact of the work to STEM education, although not tested explicitly, is hoped to foster the same kind of interest as other games and robotic projects have done.

References

[1] Tyran, C. K., & George, J. F. (1993). The implementation of expert systems: A survey of successful implementations, *Database* 24(1): 5-15.

[2] Aiken, R. M. (1991) “The New Hurrah: Creating a Fundamental Role for Artificial Intelligence in the Computing Science Curriculum”. *Education & Computing*. 7(7): 119-134.

[3] Soloway, E. (1986), "Learning to Program = Learning to Construct Mechanisms and Explanations", *Communications of the ACM*, 29(9):850-858, September 1986.

[4] Almeida, L. A., J.; Cardeira, C.; Costa, P.;Fonseca, P.; Lima, P.; Ribeiro, F.; Santos,V. (2000). *Mobile robot competitions: Fostering advances in research, development and education in* Larsen *robotics.* Paper presented at the CONTROLO 2000, Guimaraes, Portugal.

[5] Li, Y. F., Ho, J., & Li, N. (2000). Development of a physically behaved robot work cell in virtual reality for task teaching. *Robotics and Computer-Integrated Manufacturing, 16*(2), 91.

[6] Schank, R., & Abelson, R. (1977). *Scripts plans goals and understanding: An inquiry into human knowledge structures*. Hillsdale, New Jersey: Lawrence Erlbaum Associates.

[7] Noice, H. (1991). The role of explanations and plan recognition in the learning of theatrical scripts. *Cognitive Science: A Multidisciplinary Journal, 15*(3), 425.

[8] Spronck, P, Ponsen, M, Sprinkhuizen-Kuyper, I, and Postma, E. (2006). Adaptive game ai with dynamic scripting. *Mach. Learn., 63*(3), 217-248.

[9] Brown, A. K., (2009). Real-Time Intelligent Behavior: Adaption to Possible Faults in the Robo-Billiards Environment, Thesis School of CIS, University of South Alabama, December 2009.