Leveraging Mixed Reality Infrastructure for Robotics and Applied AI Instruction

Jacky Baltes and John Anderson
Department of Computer Science
University of Manitoba
Winnipeg, Canada R3T2N2
jacky,andersj@cs.umanitoba.ca

Abstract

Mixed reality is an important classroom tool for managing complexity from both the students' and instructor's standpoints. It can be used to provide important scaffolds when introducing robotics, by allowing elements of perception and control to be abstracted, and these abstractions removed as a course progresses (or left in place to introduce robotics to younger groups of students). In prior work, we have illustrated the potential of this approach both in providing scaffolding, building an inexpensive robotics laboratory, and also providing control of evaluation of robotics environments for student evaluation and scientific experimentation. In this paper, we explore integrating extensions and improvements to the mixed reality components themselves as part of a course in applied artificial intelligence and robotics. We present a set of assignments that in addition to exploring robotics concepts, actively integrate creating or improving mixed reality components. We find that this approach better leverages the advantages brought about by mixed reality in terms of student motivation, and also provides some very useful software engineering experience to the students.

Introduction

In recent years, robotics has become a common teaching vehicle in artificial intelligence (AI). While control problems and other forms of real time decision making have obvious and direct connections to robotics applications, robotics today is used to teach everything from basic search to probabilistic approaches. The use of robotics in AI teaching brings about a number of important benefits: it allows the grounding of problems that might be much harder to learn from an abstract perspective, and forces concrete, verifiable solutions, for example. It is hands-on, both requiring students to consider aspects of the problem that might be easily overlooked in an abstracted environment, and has proven to be highly motivating because of its emphasis on learning by doing. Moreover, because robotics involves a host of supporting technologies outside of AI, it reinforces the relationships between AI and other areas of computer science.

As teachers have gained more experience in deploying robotics in the AI curricula, the need for better ways to manage some of the complexities associated with using robotics in the classroom has become apparent. Our own focus on managing these complexities is through the use of a mixed (or augmented) reality environment. In a mixed reality environment, the perception and physical laws of the world we inhabit are augmented with others that are artificially created but are equally real to an agent inhabiting that world. The agent's perception of, and effect on, its world is divided into those of two different planes, and these may not be distinct to the agent. For example, a number of current augmented reality games (e.g., (Huynh et al. 2009)) involve creating virtual characters or other game elements, and superimposing these over a physical environment through the display from a camera. Humans then view and interact with this multi-layered environment through handheld camera-coupled devices such as mobile phones. Mixed-reality applications are currently being explored in a number of areas with humans beyond games, such as providing enhanced data visualization and better human-machine interfaces (e.g,. (Schall et al. 2009)).

In our own prior (Anderson and Baltes 2009; 2007a) and ongoing work, we employ mixed reality to manage the the complexity of classroom material working with robotic applications, as well to provide better evaluation in the classroom and laboratory. When students are expected to work with rich robotic applications, there is a steep learning curve to integrating both perception (especially vision) and the many systems-level problems associated with robotics that are not the main focus of any AI course. This makes it difficult to provide adequate scaffolds to support interesting assignments while students are still learning new material. Vision in particular requires sophisticated algorithms to deal with even relatively simple issues such as recognizing basic shapes, let alone dealing with perspective and noise, or tracking objects over time. Because of this, domains for the classroom are often not as rich as they could be, commonly adding contrivances to avoid vision and other rich sensing (e.g (Sklar, Parsons, and Stone 2004)). Our prior work (Anderson and Baltes 2009; 2007a) presents an architecture and approach for working with mixed reality robotics in the classroom, which allows vision and other problems to be abstracted to the degree desired by the instructor, by greatly simplifying (or completely removing) the task of recognizing and tracking physical and virtual objects in the environment. This allows students to deal with domains that involve the richness of vision, and the flexibility of mixed reality, without necessarily requiring a deep understanding of computer vision techniques.

At the same time, our approach allows the virtual layer that augments each robot's physical reality to be tightly controlled. Repeatability is one of the most significant issues in working with real robots: replicating or randomizing robot placement, obstacle movement, etc., is very difficult when working with only physical objects. When moving some of these to the virtual plane, however, randomization models can guarantee the same degree of difficulty between trials, while still working with physical robots. Thus, student projects can be compared accurately against one another or against an absolute milestone, much more easily than could be done working purely in the physical world.

Previous work (Tanner and Jones 2000) has noted that dynamic or reflective scaffolding - the ability to define scaffolds and remove them over time, among other things - is useful in teaching mathematically-oriented thinking skills. This approach directly supports such scaffolding, by allowing a degree of abstraction set by the instructor, and the gradual removal of abstractions over time, depending on the level of the class being taught. It is also portable, easily maintainable, and inexpensive. We have employed this approach to teach both AI and Robotics to undergraduates at the University of Manitoba over the past four years.

Our current work involves taking better advantage of mixed reality in a fourth year applied AI/robotics course, by making improvements to the mixed reality framework part of the assignments themselves. This set of assignments, along with some reorganization of course material, allows us to cover elements of computer vision, along with standard mobile robotics problems, under the control and scaffolding provided by mixed reality. This paper describes our efforts at integrating mixed reality more tightly with robotics and AI material in the assignments we now employ, to better leverage the advantages of mixed reality, and also emphasizes the educational benefits that result from the standpoint of software engineering experience. Before describing this class organization and term work, we briefly present the organization and basic requirements of this system, so that the work done by students can be understood in context.

Overview of Approach

A high-level overview of our approach to mixed reality robotics for education is shown in Fig. 1. The environment is inhabited by a collection of robots, and consists of both a virtual layer (provided by displaying an image on a horizontally-mounted LCD panel) and a physical layer (anything physically placed above the LCD image on the screen itself). At an abstract level, the same approach is used in the RoboCup Mixed Reality Competition (which itself was adapted from the prior E-League (Anderson et al. 2003)). Our use of this approach in education pre-dates the first Mixed Reality competition, and in our work, all of the components of are our own open-source software. The design of the platform has emerged from the necessity of supporting both these perceptual elements as well as the specific requirements for deploying this in the classroom.

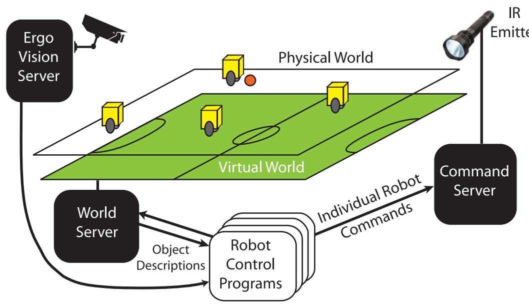

Figure 1: A mixed reality platform using global vision.

Mixed reality (we prefer the term *mixed* over *augmented*, because in our work robots do not need to perceptually differentiate the physical environment from additional augmentations) is provided through visual perception of both the physical and virtual layers through an obliquely-mounted camera. As such, the elements provided virtually and physically are not perceived distinctly from one another. The use of global vision also assists in educational scaffolding, in that a student can work with completely abstracted perception (the reported X and Y location and orientation in a global coordinate system) rather than dealing with the complexities of low-level vision if this is desired. This also means that all robots have the same visual feed. A vision system must still be maintained in terms of setup and calibration, no matter how simplified the environment, and this can serve as a very gentle means of introducing the rudiments of vision. Ergo (Furgale, Anderson, and Baltes 2005; Anderson and Baltes 2007b), the global vision system employed, is simple enough that it is possible for students to calibrate it themselves after a very brief introduction. The system also requires no predefined colors, further enhancing robustness under lighting variation compared to other vision systems, and requiring little set-up time.

The finished size of the overall system used depends on the robots employed, the portability desired, and the budget at hand. The latter two are obvious considerations, in terms of storage and the affordability of LCD panels. The former, however, is an important constraint, in that the larger the robot used, the larger the panel to support this is required. Having previously participated in the RoboCup Mixed Reality competitions, we have previously acquired the standard Citizen robots (1 1/2" or smaller), depending on the version (Fig. 2). These units will allow environments such as soccer or video game approximations (e.g., Pong, Pac-Man) to be deployed on environments as small as a laptop screen (Fig. 2). While these robots are certainly not accessible to any classroom, there are a wide array of cheap (˜$20US) infrared toys in the range of 2"-4", which are more suitable for use with a larger size LCD. These can be seen on a 40" LCD playing student-coded versions of Pac Man (left) and Hockey (right) in Fig. 3. The size of the playing area and robots also directly affects the quality and price of the one other significant piece of equipment, the camera. Small robots on a small screen typically require a higher resolution

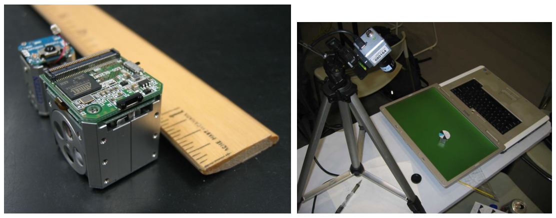
Figure 2: Micro-Robots for Mixed reality: left, Citizen Eco-Be, v1 and v2; right: using a 17" laptop as a playing field;.

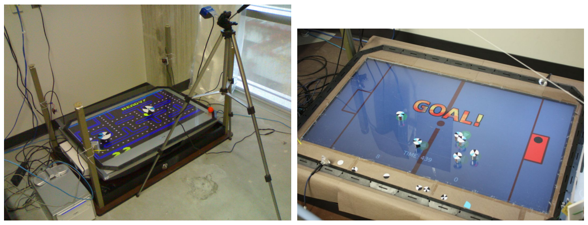

Figure 3: Applications: left, Pac Man using toy tank robots; right: Hockey.

camera, while we have used the larger cheaper robots with cameras as cheap as basic webcams (˜$40US).

Our model makes use of separate computational facilities (e.g., laptops, for portability), to control the robots, since most very small robots do not possess enough computation power locally to run sophisticated programs. These control programs receive perceptual information from Ergo over Ethernet, and asynchronously make decisions based on those perceptions to control individual robots. There is no limit on the number of robot control programs supported, beyond the ability of a network to transmit perception and action decisions. The commands issued by robot control programs are intercepted by a command server, which batches these for infrared transmission to the robots themselves using IrDA. We chose infrared-based communication because this is commonly supported in cheap toys, and also because it is robust and easily isolated over multiple playing fields by simple curtains.

Just as a robot can perceive the physical and virtual realities in this approach, it can affect objects in either domain. Affecting physical objects is a natural consequence of having physical robots moving on a playing field. The interactions between robots and virtual objects, however, must be supported in software. Our approach includes a world server, which defines the visual appearance of virtual objects on the playing field, as well as the behaviour of these objects and the effects of interaction with robots. The world server also has the ability to provide ongoing information about virtual objects to robot control programs. This allows a large range of decision for perception, since virtual objects could be tracked visually in real time, for example, or have their locations supplied directly without the possibility of perceptual error.

Setting up this platform for a given environment involves providing Ergo with descriptions of the physical objects to track, implementing the physics necessary in the world server for altering the virtual world and its display, and developing agent control programs. In our prior work, only the latter was performed by students, and our concentration was on using this model for educational scaffolding and motivation. This was done through a series of assignments involving increasingly sophisticated robot motion control (from simple reaction through dynamic path planning and complete agent architectures). During the course of our experience working with this framework, however, it became obvious that there were equally interesting and challenging elements in the mixed reality framework itself, and that these would be just as effective as a vehicles for teaching AI, robotics, and computer vision as traditional robotic control assignments. In 2008 and 2009, we reorganized a fourth-year course on applied artificial intelligence and robotics to more tightly employ the mixed reality framework by having the students' assignments revolve around extending elements of the framework. The remainder of this paper describes the coursework performed by these students, its relationship to the material presented in class, and our experiences with this approach.

Teaching Robotics and Computer Vision Using Mixed Reality Infrastructure

Assignment 1: Ergo Vision Server

Name	Ergo Vision Server Extensions
Primary Objectives	Object recognition and tracking, Haar features
Secondary Objectives	Code maintenance

All work in this course is performed by students in groups (3-4/group is ideal, depending on the number of students in the class). Initial classroom material introduces very basic computer vision, and the first assignment involves implementing additional features for the vision server that supports students' mixed reality applications, to improve robustness and flexibility. By focusing on adding features to an existing system, students have the benefit of working software to begin with as well as the sense of accomplishment that comes from producing code that will be used by others. The isolated nature of additional features also lets students gain experience with elements of computer vision without all the complexity of 3d tracking. The extensions chosen are based on new research ideas and require students to implement a novel algorithm described in a recent research paper. In 2009, the major portion of the assignment asked students to re-implement the Viola and Jones Haar based feature tracker (Viola and Jones 2001) to replace the current capture engine. A second extension was a fully automated camera calibration system, which would use a world server to project calibration points with known world coordinates on the screen and then record their position. This reduced the system's interface from several panels and check boxes to a single "Calibrate" button.

Apart from the specific computer vision techniques that the students learn during this assignment, there are also several software engineering skills that they use. The Ergo code base is about 20,000 lines of C++ and C code at the moment. Students will have to implement their new techniques within this framework. As any professional programmer knows, software maintenance on other peoples' code is an interesting and challenging problem.

Another interesting point is that the current set of students often complain about inconsistencies in the library routines, lack of documentation, or choices of data structures. They have to put themselves in the position of prior developers and ask why they made the choices they did. Many of the students either become graduate students, or at least follow along with the next year's course for additional experience. For many of them it is an eye opening experience when the new batch of students complain about some aspects of the software and the old students realize that they are now the prior developers whose decisions are being questioned.

Ergo is currently used by several other robotics research teams in Germany, Uruguay, Finland, England, and New Zealand. Of course, not all extensions by the students are successful, but students whose extensions make it into the official code base are rightfully proud of their contribution. They also often have to provide technical assistance and documentation to those teams that are trying to use their extensions.

For 2010 we plan to ask students to replace the aging configuration scripting language with a more modern one such as LUA or Python. Another reasonable extension is to reduce the number of patterns on the hats needed to track the robots.

Assignment 2: Aucklandianapolis and Treasure Hunt

Name	Aucklandianapolis
Primary Objectives	Steering, velocity, and path tracking control
Secondary Objectives	Hardware dependencies, visualization, project management

The *Aucklandianapolis* is a path tracking problem for mobile robots. Robots must drive five laps around an "A" shaped race track (Fig. 4) as quickly as possible. It is similar to time trials in Formula 1 racing. The lectures support this assignment by introducing the basics of differential drive and robots with Ackerman steering control and present a number of alternative path tracking algorithms (e.g., PID, sliding mode, look-ahead, fuzzy control, CMAC) that have been successfully implemented in previous years.

This is the first assignment that introduces students to the full sensor - reasoning - action cycle in the real world. We believe that it is essential to introduce control of real world robots as soon as possible. Many students are unable to predict the harsh realities of the real world at first. Classic examples are having to deal with noise or total loss of the robot in the vision server, delays in the networking, and to confirm coverage of the infrared transmitters.

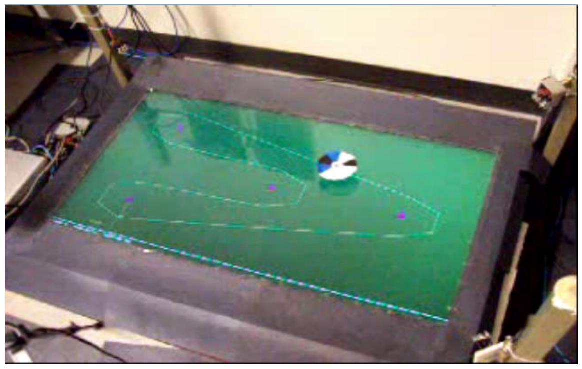
Figure 4: A toy tank robot driving the Aucklandianapolis.

Since many of the control algorithms in the research literature either ignore the dynamics or make unrealistic assumptions about the detailed knowledge available of the robots (e.g., the tire torsion coefficient), students realize that practical concerns must be taken into consideration. They quickly realize, for example, that most papers only discuss the steering control of the robots, and that the velocity also must be controlled since robots will otherwise topple over when taking a sharp turn or will be lost by the vision system when turning too quickly.

Many of our students are used to using print statements as their main debugging technique, since this is sufficient for many of the small assignments that they have to do during the course of their studies. While developing their controllers, students quickly realize the gross inadequacy of this technique for robotics problems. After trying to decipher thousands of floating point numbers that fly by on the screen, students realize that code must be debuggable and testable in this environment. They then quickly implement visualizations of the system which graphically show the current position of the robot, the destination, the closest point on the path, the angle error, and other useful features.

Students are also often perplexed that their control performs well in the first couple of minutes, but then deteriorates when the charge in the battery drops and the reaction of the robot to the control changes. Students realize the importance of using the vision feedback to control their motions.

Real robots also introduce project management problems such as battery charging schedules, coordination with the other teams for access to the playing field, etc.

Another aspect of this assignment that students find appealing is that it has remained basically unchanged since 1997. Videos of previous teams are readily available and students feel as part of a larger research community than an isolated point in time.

Name	Treasure Hunt
Primary Objectives	Point stabilization, greedy and global search, multi-agent coordination, virtual world
Secondary Objectives	Systems engineering, system level coordination, system integration

The second part of the second assignment is a multi-agent treasure hunt. Two robots start at opposite ends of the playing field. Five "treasures" (circles with a 5cm diameter on the virtual plane) appear at random locations on the playing field. The task for the robots is to "dig up" all five treasures as quickly as possible, by sitting on top of the treasure for an uninterrupted period of one minute. The position and number of treasures is noisy since it is provided by the video server.

This part of the assignment introduces the concept of the virtual world that interacts with the physical world. Students must implement a world server that monitors the position and time spent on top of a treasure to keep score and to grey out treasures that have been dug up. This world server needs to be synchronized to the physical world through sensor feedback (the vision server). The students must also deal with the fact that the virtual world is perceived through vision: as soon as robot drives on top of a treasure, for example, the video server will not be able to see the treasure anymore. Their control software must remember the previous position of the treasure to dig it up

In this application, the control problem implemented for each robot is a point stabilization problem. Most students find these simpler than path tracking. However, since it takes the robots a non-negligible time to stop, the robots need to break before they overshoot the treasure. When using robots with Ackerman steering, approaching a point is made more difficult by the minimum turn radius of the robot. A simple controller will continuously circle a spot when the treasure is inside the minimum turn circle of the robot. In this case, the robot needs to back up and turn towards the spot.

The main part of the assignment is to compute a plan (i.e., the sequence in which a robot wants to dig up the treasures). A simple greedy scheme provides a reasonable plan with very limited computational cost and is quite robust to noise. To improve on this scheme, a global search of all or many possible sequences is required. Obviously, this increases the computational cost. This is made difficult by the fact that the actions of the other robot introduce uncertainties and need to be taken into consideration. There are also many interactions between the point control and the plan. For example, a planner that tries to find the shortest path may not return the fastest plan since it ignores the time for the robot to turn to a new direction. Furthermore, some teams have fast and efficient turn control, but perform poorly on long straights, whereas the situation is reversed for some other teams. Students realize that the optimality criterion used in their planner is a complex function with many parameters.

Another important aspect of the assignment is the use of multiple robots, which requires multi-agent coordination and communication strategies. Students know from the beginning of the course that the end goal they are working toward for the final assignment requires a large team of robots, and that central control does not scale to large teams. So, students implement distributed control architectures that communicate via the network. Real-world multi-agent communication and coordination is surprisingly difficult and subtle, even in an application such as this. For example, a simple approach would be to have each agent communicate which treasure it is currently trying to reach, so the other robot can ignore this treasure. However, in this case, teams need to deal with conflicts (e.g., two robots want to dig up the same treasure). More importantly, the students need to devise strategies to deal with cases when a robot communicates that it intends to dig up a treasure and then fails to achieve its goal (e.g., drives outside of the playing field while trying to reach the treasure, and is lost from the vision server). These strategies involve monitoring the other robots via visual feedback and estimating their progress.

This assignment also introduces students to system level coordination issues. The command server is able to keep up with one agent sending messages at full speed, but gets bogged down with two agents sending messages at full speed. Therefore, teams need to develop strategies for how often and when to send messages to the command server. .

Secondary objectives of the assignment are systems integration and system engineering problems. More elaborate planners may result in a better plan, but require more time to implement. The utility of a better plan depends on the speed and accuracy of the point control of the robot, in that a robot with poor control will not be able to take advantage of a better plan. So, students need to weigh whether spending time on control or path planning is more useful. It should also be apparent to students that the visualization and path tracking control for the Aucklandianapolis share a lot with the treasure hunt. They experience the importance of basic software engineering techniques such as encapsulation, reuse, and modularity.

Assignment 3: Obstacle Run and Obstacle Pong

Name	Obstacle Run and Obstacle Pong
Primary Objectives	Local and Global Path planning
Secondary Objectives	Code Refactoring

The third assignment focuses on global and local path planning. In lectures many different path planning algorithms including visibility graphs, Voronoi diagrams, quadtree decomposition, and flexible binary space partitioning and their strength and weaknesses are discussed. Potential fields are discussed as an example of a local path planning method. Students implement a path planning method of their choice for two dynamic environments.

In the obstacle run, two robots start at opposite ends of the playing field. The world server projects obstacles that move in straight lines and constant speed across the playing field. A team is awarded a point each time one of the robots traverses the whole of the playing field without touching any of the obstacles. The goal is to collect as many points as possible within a 5 minute period. The challenges in this assignment are similar to those described above for the Treasure Hunt competition, while adding a dynamic environment.

The second part of the assignment is the first competitive environment where teams of a single robot play against each other in a game of obstacle pong - a modified version of Pong, the very first computer game. Mixed reality is integrated not just through the robots moving on a virtual field, but by defining a virtual paddle is attached to each robot, used to hit a virtual ball. This lets students see that mixed reality can augment the robots themselves, rather than just the world they inhabit. A team scores a point if the ball crosses the goal line of the other team. The virtual ball increases speed slowly based on the number of times it has been hit.

The environment is made more complex by having several virtual obstacles projected in the playing field. A robot is not allowed to touch any obstacle, and if it does, then its paddle will be deactivated for a certain period of time. This requires real-time path planning in static domains.

This assignment is deliberately not quite as challenging as the other assignments. Students use this time to refactor their code and make various improvements to their code to make it more robust and flexible for the final assignment.

Capstone Assignment 4: Soccer, Hockey, Pac-Man

Name	Pac-Man, Hockey, or Soccer
Primary Objectives	Behaviour Trees
Secondary Objectives	Living with design choices

The final assignment is always a competitive event which pits the student teams against each other. Any dynamic multi-agent domain can be used, but in the past we have used soccer, ice hockey and Pac-Man as capstone projects. This capstone project is announced at the very beginning of the course and students expect to spend most of their time working on this assignment.

However, more time is spent on implementing a solid foundation through the previous assignments than on the actual capstone. At this point, students have implemented at least preliminary versions of path and point control as well as path planning and obstacle avoidance. Based on such a solid foundation, it is easy to implement a large variety of game-like environments in a relative short period.

The mixed reality approach really shines here. It would be extremely difficult to implement a small robot that picks up pellets and power pellets for the Pac-Man game, but it is easily done by extending the world server that students are by now already familiar with. In the ice hockey game we used a virtual puck, which allowed us to implement a game where robots could use two different types of shots (a slap shot which is faster, but less accurate than a wrist shot), which would again be very difficult with physical robots. Again, allowing students to modify elements of the mixed reality architecture gives them a richer experience than would be possible programming robotic applications alone.

The new concepts that students learn in class are various high level agent architectures such as finite state machines, behaviour trees, subsumption architectures, and BDI. Most implement a version of behaviour trees since they are flexible, robust, and easily implemented. Students face the problem of an agent that oscillates between behaviours. The most common solution is a timer which prevents switching to a different behaviour or hysteresis functions.

From this capstone students also learn that design choices made early on in the course can become liabilities later on, but are really hard to change. For example, one team in the ice hockey final realized that the other team had a flexible control which allowed it to drive forward and backwards. The other team was unable to change their code base quickly enough to implement this feature as well.

Discussion

Mixed reality brings a great many advantages to teaching robotics. We have found that leveraging elements of the mixed reality framework for assignments, through the students' support of virtual environments and vision server improvements, gives them a greater understanding of the elements the approach abstracts (e.g., vision). It also more strongly motivates the students at this level, and provides a richer experience in terms of experience in software engineering.

References

Anderson, J., and Baltes, J. 2007a. A mixed reality approach to undergraduate robotics education. In Holte, R., and Howe, A., eds., *Proceedings of AAAI-07 (Robot Exhibition Papers)*. Vancouver, Canada: AAAI Press.

Anderson, J., and Baltes, J. 2007b. A pragmatic global vision system for educational robotics. In *Proceedings of the AAAI Spring Symposium on Robots and Robot Venues: Resources for AI Education*.

Anderson, J., and Baltes, J. 2009. Using mixed reality to facilitate education in robotics and AI. In *Proceedings of the 22nd International FLAIRS Conference*.

Anderson, J.; Baltes, J.; Livingston, D.; Sklar, E.; and Tower, J. 2003. Toward an undergraduate league for RoboCup. In *Proceedings of RoboCup-2003*.

Furgale, P.; Anderson, J.; and Baltes, J. 2005. Real-time vision-based pattern tracking without predefined colors. In *Proceedings of the Third International Conference on Computational Intelligence, Robotics, and Autonomous Systems (CIRAS)*.

Huynh, D.-N. T.; Raveendran, K.; Xu, Y.; Spreen, K.; and MacIntyre, B. 2009. Art of defense: a collaborative handheld augmented reality board game. In *Proceedings of the 2009 SIGGRAPH Symposium on Video Games*, 135–142.

Schall, G.; Mendez, E.; Kruijff, E.; Veas, E.; Junghanns, S.; Reitinger, B.; and Schmalstieg, D. 2009. Handheld augmented reality for underground infrastructure visualization. *Personal and Ubiquitous Computing* 13(4):281–291.

Sklar, E.; Parsons, S.; and Stone, P. 2004. Using robocup in university-level computer science education. *Journal on Educational Resources in Computing* 4(2).

Tanner, H., and Jones, S. 2000. Scaffolding for success: Reflective discourse and the effective teaching of mathematical thinking skills. *Research in Mathematics Education* 2(1):19–32.

Viola, P., and Jones, M. 2001. Rapid object detection using a boosted cascade of simple features. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 511–518.