# Design and Implementation of MARS2020: Shaping future engineers through PC Gaming

Eve Powell
Department of Computer Science
University of North Carolina at Charlotte
9201 University City Boulevard, Charlotte NC
empowell@uncc.edu

Advisors: Dr. Ayanna Howard, Dr. Tiffany Barnes, Sekou Remy
School of Electrical and Computer Engineering.
Georgia Institute of Technology. Atlanta, GA
ayanna.howard@ece.gatech.edu

## Abstract

MARS2020 is a futuristic game/simulation that's purpose is to introduce middle and high school students to the fundamentals of robotics. The game employs high quality, realistic graphics, a believable storyline and high paced missions that keeps the game engaging, despite the complexities of the game. MARS2020 is to be developed using Microsoft Robotics Studio. MARS2020 draws from the successes and failures of various academic projects that intended to teach advanced problem-solving skills and/or programming concepts through a computer simulation or a serious game. By exposing our target audience to basic engineering concepts in game, we hope to inspire learning as well as a enthusiasm of engineering and robotics. Though still in its design and beginning implementation phases, MARS2020 looks to be an attractive method of encouraging children and young adults to pursuing an education in engineering.

## Motivation and Background

MARS2020 is an educational virtual simulation of working with robots in various environments that are currently unexplored. Though designed as a video game, its primary concern is to *educate* through realistic problems associated with robot use and operation.

Our primary audience is general high-school and middle school students. The goal is that our target audience can be drawn into the game for periods of at least 20 minutes or more, and afterwards demonstrate a greater understanding of using robots for problem solving, programming for robots, and data analysis. By coaxing these students to think in this way, we hope to promote an interest in engineering and computer science.

To appeal to our target audience, MARS2020 is set to take place 13 years (year 2020) in the future, a time when most of our core audience will be finishing up with their college careers and applying their knowledge in industry. MARS2020 takes our audience into the semi-academic career of an astronaut that is involved directly in Project Mars: an initiative to send a human with the assistance of robots to Mars.

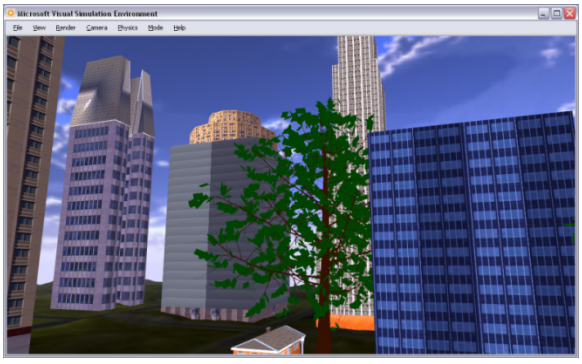

**Figure 1: Urban Search and Rescue environment that I created this summer. The potential realism in this environment was later motivation to create MARS2020.**

Students will be immersed in a realistic 3d simulation of multiple environments, all leading up to conducting critical studies on Mars. They will be given multiple objectives and will complete with other astronauts and forms of in-game artificial intelligence in order to complete tasks.

MARS2020's design also stems from the growing desire in academia to create more realistic environments on which to test prototypes or code to be used on physical robots. If done correctly, this suggest the possibility of later using the simulation technology used by MARS2020 to experiment with a variety of scenarios and processes, without actually using and/or endangering the physical robot itself [3].

When designing MARS2020, we drew from the successes and failures of various serious games projects that were meant to teach through use of robots and virtual environments. Very few of such attempts can be seen as huge successes as far as achieving learning while in game, however, most of them could be seen as successful at fostering deeper interests in their focus areas. Such projects include:

**United States Military Academy at West Point**. The Computer Science Department at West Point requires their students take an Information Technology course in which they use computers and robots for problem solving. Students spend a semester working with Lego Mindstorm robots and the Jago package in which they program behaviors of these robots using Java. This class makes use of both the physical robots and computer simulations. They have reported high success in this program[4].


Figure 2: Screenshot of ALICE. Ongoing project at Carnegie Mellon University.

**ALICE at Carnegie Mellon**- Alice is a 3d, open source programming environment designed to target middle school girls and motivate students to continue their education in programming. Girls were introduced to this environment as a way to create and animate 3d characters and environments. Students were encouraged to work individually and in groups to either express themselves creatively or to achieve some predetermined outcome. So far, the social a creative nature of this environment has been highly successful in appealing to this target audience. Storytelling ALICE later was built into the project as a way for the children to further express their creativity through telling a story using this platform[5].

**Game2Learn at UNC-Charlotte**. Game2Learn is an initiative to teach fundamental programming concepts to beginning computer science students at UNC-Charlotte. Students play through situations in a role-playing environment, using concepts they learn in class and through prior levels to complete the games. Multiple games have come through this project, some games having more success than others[1]. Currently, selected introductory classes use the games as study tools and provide feedback on the usability of the game[1].

Among the projects reviewed for the purpose of this research I noticed that ALICE has shown the most success at inspiring young audiences at pursuing an education in their intended field. In a recent paper, Kelleher suggest that most

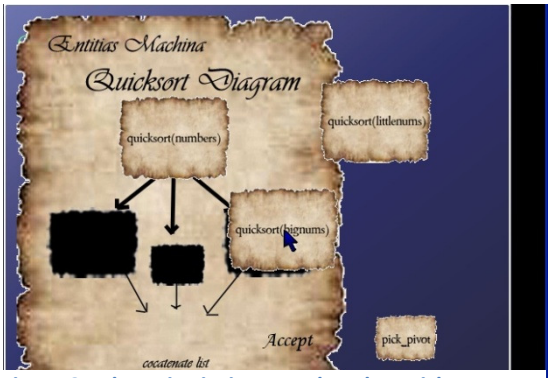

Figure 3: Player is piecing together the quicksort algorithm in a Game2Learn game.

students did attempt to use either methods or do-togethers in Storytelling Alice. Users of Storytelling Alice were also extremely motivated to take home Storytelling Alice as opposed to earlier Alice software[5]. The other projects, while showing moderate success, all indicated a need for redesign in order to either increase learning or increase interest. While designing MARS2020, care has been taken to ensure that past mistakes will not be repeated.

## Design of MARS2020

From studies such as the ones listed above, there were many observations and assumptions I was able to derive from past successes and failures of similar initiatives:

**Provide Immediate and accurate frustrations.** Immediate and accurate feedback is critical for any such project. Students need to immediately see that they have done something right or wrong. Not only that, they need to see the appropriate outcomes for both right and wrong answers. Such interaction serves many purposes. Students will be forced to consider the right questions when playing the game. Many initiatives wrongly force students to try and outsmart the system rather than try to solve the problem at hand. Giving them feedback that is more in depth than 'right' and 'wrong' forces them to consider what was right and wrong about what they did. In this way, simulations have proven more effective than games.

**Encourage exploration.** When a simulation offers more than one outcome, students are more inspired to play with the system, to test more outcomes and ask questions about certain reactions. Normally, it is a good thing when participants purposely test wrong answers to confirm suspicion of a negative reaction. Such interaction displays a growing understanding of the system and a desire to learn more about the system and or the subject matter.

**Learnable Interface.** Another important aspect of the game to consider was the interface with which participants would be interacting with the game/simulation. The interface has been in many instances, what can make or literally break a game and or simulation. The interface determines the ease with which the participant can interact and ultimately learn from the system. A complicated interface can cause unnecessary confusion [10] while a simple interface can deter from learning the intended subject matter.

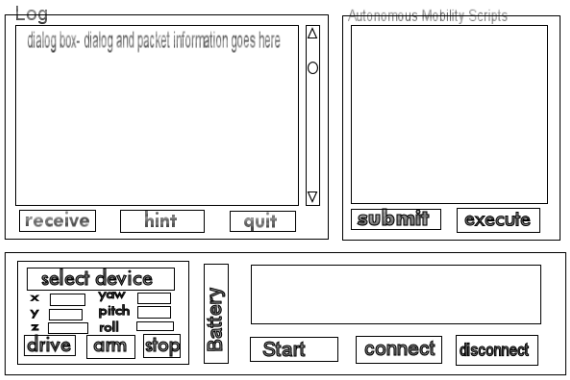


**Figure 4: Mockup of the GUI players will use to interact with in-game robots and camera.**

The important things to consider when designing a suitable interface for this type of system is to keep the interaction engaging as well as thought inspiring. For MARS2020, it was decided to try a different type of interface. One that had different elements that all required varying amounts of understanding. While using the MARS2020 interface, participants should notice that there are certain parts of the interface that they are not using at first, but should be able to interact with the game without much trouble. However, later, as the game progresses, it should prove necessary to learn more about the interface and use its more complicated elements to adapt to problems encountered in the game/simulation. While playing the game, they should realize gradually why the more complicated elements of the interface are necessary to understand in order to be competent in this field of study. The interface should inspire them to want to know more about controlling and manipulating robots. The more advanced features of the interface should also serve to engage participants that are more capable to still maintain interest in the MARS2020 world, even if the rest of the class is behind.

**Autonomous Mobility Scripts.** The highlight of the game MARS2020 takes place on the final levels of the game, where you no longer have direct control over your robot all of the time. With a text-editor like interface, participants will be required to script the intended movements of their robots and then send the robot out to fend for itself on Mars terrain, collect data, dodge obstacles, recover from unforeseen circumstances and return back to base. Concepts behind autonomous movement in robots are ultimately what students are encouraged to learn and understand through playing MARS2020. While playing MARS2020, students should be allowed to see scripts being executed by other AI robots in the game.

**In game story.** This is one element that is not considered very much in this area of research. Not too many projects have noted this aspect of game play necessary when creating serious games for the purpose of learning engineering or cognitive science concepts. However, I've noticed that serious games used for the purpose of learning other concepts have added this element of game play and have had positive results. Story in game is something that is rarely overlooked when creating consumer gamers and so it was with this observation that I decided that it was necessary to both add and underlying game story as well as to monitor its significance to this project. In MARS2020, the minimalistic story in this game ensures that levels and missions get progressively harder and that learning occurs.

**Artificial Intelligence.** MARS2020 will include a small amount of intelligence in game. The artificial intelligence included in this game is to serve both purposes of providing competition for the participants playing the games as well as to serve as tutors in game for learning more efficient ways of accomplishing given tasks. Artificial agents in the game aren't interacting directly with the player. They are mostly working to get the same task or a similar task done. So the artificial intelligence in this game is minimal, they most serve as tutors,

distractions, and sometimes, obstacles. Later versions of MARS2020 should consider more advanced intelligent tutoring systems, achieving learning directly through game play and limiting interaction between participants and game moderators.

## Implementing MARS2020

MARS2020 was designed to use features provided in Microsoft Robotics Studio that would set it apart from other robot simulations. MRS, though a new and virtually unexplored environment, offered a platform best suited for the amount of realism and immersion sought after in the design of MARS2020.

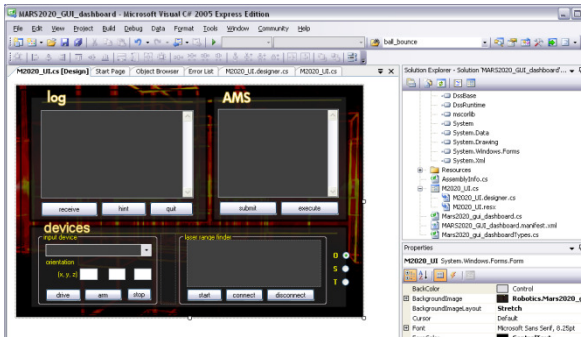


Figure 5: Implementing the GUI for MARS2020 in Visual C# Express.

**Microsoft Robotics Studio.** MRS is a new development environment created for use by academic and hobbyist robotic application developers. MRS boasts an easy-to-use visual programming language to simplify the process of programming robots through a drag and drop environment. Other features include its 3-D simulation tool that simulates physics-based virtual environments, using the PhysX engine from AGIEA Technologies Inc, as well as providing an environment for using realistic 3 dimensional models to simulate robotics applications[7]. Many third party companies have shown support for MRS by offering their services and robots in the MRS package. Because of its initial support and its realistic simulation environment, it was decided to develop MARS2020 within Microsoft Robotics Studio.

While MRS did prove to be ultimately a great platform for development and the way of the future for robotics, there were also many drawbacks to the environment that slowed development significantly. Outside of the MSDN forums, Microsoft provided very little documentation on their environment. Also, Microsoft seemed to push for quantity instead of quality when offering support for programming environments. Finding help in with any particular language proved difficult, but there is wide and varied support for each language. Also, documentation is different in each location and related subject matter is often posted in several different locations.

Despite drawbacks in documentation, there were few qualms with the environment itself. The simulation editor already takes into account many things that virtual environment developers struggle with everyday, like shadows and physics simulation, as well as generating terrain through height maps or .ISA files, converting 3d model files to binary files, or reducing the pressures of drawing complex meshes on the screen with simplified run-time generated meshes. There were usually simple solutions to the problems encountered in MRS. For example, after a few hours of searching I discovered that physics could be toggled on an off per entity by setting the mass to Zero, only later to discover that this fact had in fact been documented but overlooked because it was in a location unexpected.

**Simulation Engine Service.** One problem unforeseen when introduced to this project, however, was that the hardware that MARS2020

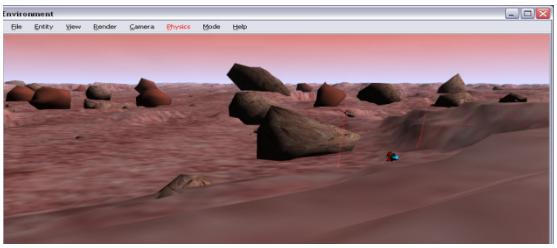

Figure 6: Screen shot of Mars Environment in MARS2020

was to be operated was too outdated to run the simulation engine service adequately. This service, required for rendering entities to the screen and visualizing effects, uses the programmable pipeline of graphics accelerator cards, as per DirectX9 shader standards[9]. This proved problematic for the older machines in the Humans Labs, especially those not equipped with a discreet graphics card. Because of this, Microsoft recommends that users have a graphics card suitable for gaming. Unfortunately, neither the lab, nor the schools intended to participate in play testing can be expected to have computers equipped with such graphics cards in the near future.

**MobileRobots Pioneer3dx.** In MARS2020, the player makes use of the MobileRobots Pioneer3dx. Though not the ideal robot for traversing terrestrial surfaces, in the MRS simulation editor this robot is best of those provided for navigating uneven or course terrain found on surfaces like the Moon or Mars.

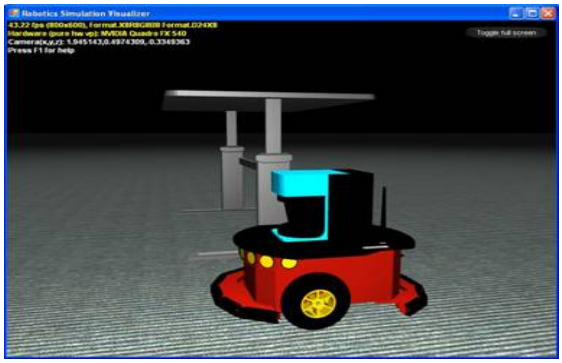

Figure 7: Pioneer3dx rendered in MRS [9].

Also, the P3DX is a machine most likely found and used in an academic setting when presenting robotics to adolescents. These reasons made the P3DX the likely candidate for operating a robot in a virtual environment. Later, students could use the skills learned in game to operate a physically present robot. When used in MARS2020, students will interact with the Pioneer3dx using a navigation/programming interface that was designed for the game.

**Xbox360 Controller.** MARS2020 was designed to make use of the Xbox360 controller. Through this design, we hope to use the familiar nature of the controls setup to appeal to the players. The Xbox360 controller design allows for easy control and navigation of the robot, which is sought after during their first experience with the game. The controller however, not only provides visual feedback on the simulation, but textual feedback in the provided GUI. We also plan to provide force feedback. While force feedback is not a realistic element in the simulation, adding to the level of immersion that the player experiences in the game is what I hope to accomplish with this addition. Michael Zyra generalizes, when discussing immersion in serious games, that it is important that researchers learn to use sensory simulation such as haptics for serious games [8]. The Xbox360 controller was designed to work somewhat with MRS. Built in to the simulation editor the Xbox360 controller by default manipulates the camera. In the Microsoft provided Simple Dashboard the Xbox360 controller manipulates the differential drive and the laser finder of the Pioneer3DX entity. The Xbox360 controls of MARS2020 will control the differential drive, a KUKA arm (with grippers) attached to the Pioneer3DX, the laser ranger finder, and the camera.

**Generating Heightmaps from satellite images of Mars.** This area was what I spent most of my implementation time working on. In order to accurately represent Mars in this game/simulation, I had to figure out how to take an overhead image, no different from an image taken through satellite and translate that image into a grey scale height map of Mars. What started as a search for such an algorithm quickly changed into an attempt to come up with my

own algorithm, as no one has successfully translated one image to the other through code.

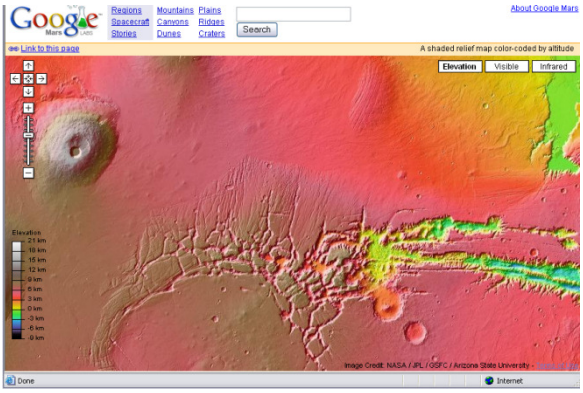


**Figure 8: Overhead satellite image of Mars provided by Google Mars.**

It seems most attempts at such work have only been successfully performed by artists. The original terrain images contain information, that hint towards the geography, but none of that information has yet been linked to a direct translation from 2 dimensional peculiarities to 3 dimensional geographical information. Two weeks were spent trying to pull this information out of several images. The result was a semi-believable terrain on which to place the Mars levels of MARS2020.

## Future Work

The design of MARS2020 is complete and much of pioneering has been done in learning how to create realistic environments and game like interfaces within MRS. With this foundation, MARS2020 is a project that should be play testable after two to three months of development time. Most of the work done in the MRS environment has been testing solutions that could be applied to the final game. Very little of the code structure should be used in generating a final game, however, many of the manifests and code examples generated during the summer should be considered when creating the final game. The last of my work over the summer will be finalizing documentation of code examples I worked on during the summer, answering those hard to find questions that I had to find through trial and error or through extensive research. Through my final documentation of the game design, technical design, concept, and time line, MARS2020 should be able to be realized by any student that takes on this research project.

## Acknowledgements

This project was supported by the CRA Distributed Mentor Project(CRA-DMP) and the Summer Undergraduate Research Experience (SURE) at Georgia Institute of Technology. I'd like to thank all of those that made this project possible.

## References

[1] Barnes, T., H. Richter. Game2Learn: Building CS1 Learning Gamers for Retention. Submitted to ACM SIGCSE Conference on 2007.

[2] Becker, Byron. Teaching CS1 with Karel the Robot in Java. Proceedings of the thirty-second SIGCSE technical symposium on Computer Science Education SIGCSE. 2001.

[3] Bernhardt, R., Schreck G., Willnow, C. Realistic Robot Simulation. Computing & Control Engineering Journal. August 1995. Pp 174.

[4] Flowers, T. Gossett, K., Teaching Problem Solving, Computing, and Information Technology with Robots. Consortium for Computing Sciences in Colleges 2002. Pp 45-55.

[5] Kelleher, C., Pausch R., Kiesler S., Storytelling Alice Motivates Middle School Girls to Learn Computer Programming. CHI 2007 Proceedings.

[6] Kay, Jennifer. Teaching Robotics from a Computer Science Perspective. Consortium for Computing Sciences in Colleges 2003.

[10] Rabin, Steve. Introduction to Game Development. Charles River Media. 2005. Pp 120.

[7] Tick, Jozsef. Convergences of Programming Development Tools for Autonomous Mobile Research Robots. SISY 4th Serbian-Hungarian Joint Symposium on Intelligent Systems 2006.

[8] Zyda, Michael. From Visual Simulation to Virtual Reality to Games. Computer 2005. Published by IEEE Computer Society. Pp 25-32.

[9] Microsoft Robotics Studio Website. http://msdn2.microsoft.com/en-us/robotics/default.aspx. Microsoft Corporation. 2007.