NIST Simmers
Team Description Paper
Infrastructure Competition
RoboCup 2010

Stephen Balakirsky, Taylor Brent, Jon Dukto
National Institute of Standards and Technology, 100 Bureau Drive, Gaithersburg, MD 20899, USA

Abstract. The Unified System for Automation and Robot Simulation (USARSim) is the simulation framework at the heart of the RoboCup Rescue Virtual Robot Competition. The construction of this framework has been a community effort with many different institutions contributing code and models for various sensors and robot platforms. These contributions have proven to be an invaluable lasting resource for both research efforts and the RoboCup competition. One often overlooked component of the simulation framework are the simulated environments that the robots will operate in. Contributions to this area are not long-lasting, and new environments must be constructed for each round of every competition. Research interests also require the constant creation of "fresh" experimental environments. Up to now, the creation of these worlds has been a time-intensive problem that steals resources from other important areas. In this paper, NIST presents a new tool that automatically creates various classes of environments that may be used in both the RoboCup competitions and in general robotic research.

Introduction

The Unified System for Automation and Robot Simulation (USARSim) is based upon Epic Game's Unreal Tournament 2004 Game engine[1] with a new release now available for the Unreal Tournament 3 Game engine. Open source modifications to the engine are available on the USARSim Sourceforge website[2] that transforms the game into a full framework for robotic simulation. The RoboCup Rescue Virtual Robot Competition takes advantage of this open source framework for its competition. In addition to the robot and sensor models, a successful competition requires specially designed challenging environments (worlds) that the robots will operate in. These environments must have corresponding ground truth in order to facilitate scoring. In order to assure that the competition closely matches reality, it is desired that teams do not have prior knowledge of the layout of the environment that they will be faced with. Last year's competition required the creation of 7 unique environments at great

[1] Certain commercial software and tools are identified in this paper in order to explain our research. Such identification does not imply recommendation or endorsement by the authors, nor does it imply that the software tools identified are necessarily the best available for the purpose.
[2] www.sourceforge.net/projects/usarsim

expense in terms of time and resources. The two worlds that were used in the finals involved several hundred man hours of labor to create. This year's event will take place in the newly released UT3 which will render all of these worlds useless (even if we had wanted to reuse them).

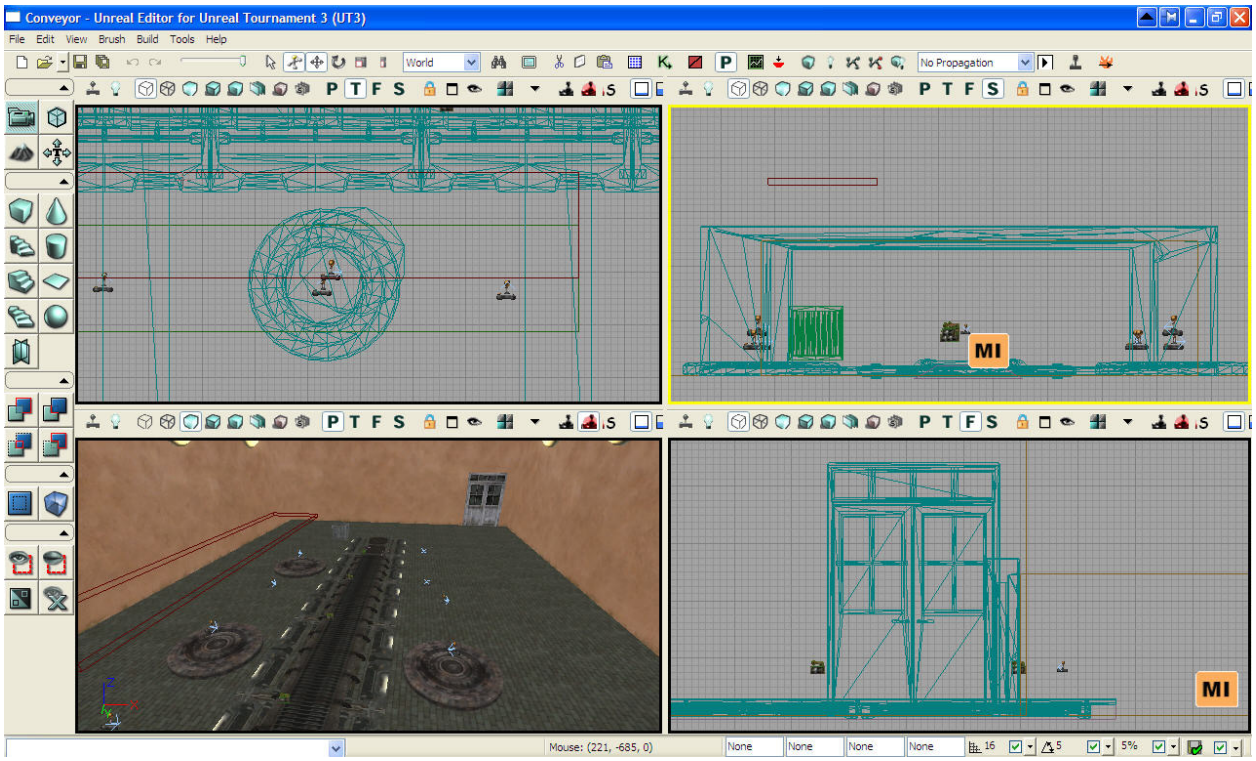

Figure 1: Screen shot of Unreal Tournament 3 editor.

Currently, several classes of worlds are built for the competition. All of these worlds are built in the Unreal Editor that is part of the game's distribution. This editor is a powerful 3D computer aided design (CAD) system that requires conquering a large learning curve to use effectively. Figure 1 shows a snapshot of the editor. Perhaps the simplest form of world to build is a maze world. These worlds are designed to closely match the class of environment that is used in the RoboCup Rescue Physical Robot League. For the maze world, walls are carefully constructed so that there are constant width hallways between the walls of the maze. All of the walls must then be textured, and lighting must be added to the environment. Finally, ramps are placed on the floor of the maze. For the "yellow" maze worlds, the ramps are continuous, and care must be taken as to which direction each ramp is placed so that no discontinuities result. The "orange" maze world allows for discontinuities in the ramps, so the robots may have to traverse over a step.

Indoor buildings are more complex to create. First, the floor plan is carefully laid out by constructing individual rooms. Then doorways between rooms must be added, textures must be applied to all of the surfaces, lighting must be placed in the rooms, and finally the rooms must be populated with static meshes (objects like desks, chairs, etc.).

Ground truth must now be created for these worlds. The current technique for creating ground truth involves creating a screen dump of the overhead view from the editor and then geo-registering the image. This is a time consuming process and involves the use of Geospacial Information Services (GIS) software. This software has its own expense and learning curves associated with its use. Two separate ground-truth products are created for each world. The first is a geo-registered image that is created from the screen dump. The second is a vector file that contains the location of all of the important features of the world (walls, obstacles, etc.). This file is created by hand by having a person draw polygons and polylines over the associated image. This process is time consuming and error prone. For research applications where it is desired to have a new world for each run, researchers may spend more time in designing their custom worlds then in performing actual research.

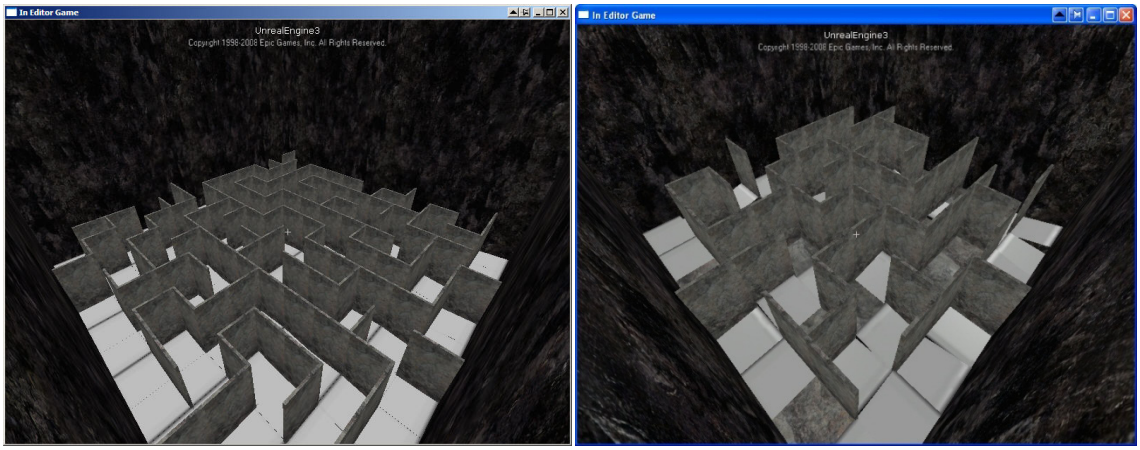

Figure 2: Randomly generated mazes. The left figure illustrates a maze with continuous ramps (yellow) and the right figure illustrates a maze with discontinuous ramps (orange) that occur in the maze with a placement probability of .8.

NIST World Generator

In order to alleviate this problem, NIST has created an automated world generation program. This program is capable of creating worlds for both the UT2004 and UT3 version of the Unreal Engine and is written in Java to support the widest variety of runtime platforms. By using the applet, even a novice is able to create unique, rich worlds and ground truth for use in either competitions or their research. The world generator currently contains two separate applets. The first creates random mazes of a size specified by the user. The user is able to specify items such as hallway width through a configuration file and items such as ramp slope, the creation of a “yellow” vs. “orange” maze, maze size, and compliance with UT2004 or UT3 via an easy to use dialog system at run time. Figure 2 shows both a “yellow” and an “orange” automatically generated maze. The yellow maze is created in a way that assures there are no discontinuities in the ground surface, although the robot will experience front, back, and side slopes as it traverses through the environment. The orange maze allows for discontinuities or misalignments in the ramps. This requires the robots to climb up or down steps while traversing the maze. These mazes closely resemble the real mazes that are in place with the real robot league and are expected to foster closer cooperation between the two competitions.

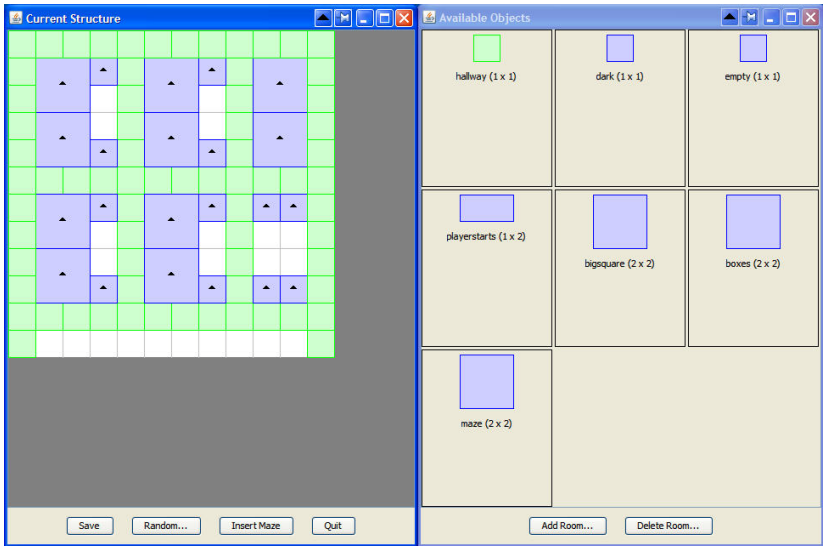

Figure 3: Snapshot of the building creation GUI.

In addition to the maze generator, a building interior generator has been created. Figure 3 shows a snapshot of the graphical user interface (GUI) for this applet. This applet allows for the automatic generation of interior building environments through the use of a "room library". The room library consists of pre-created complete room layouts. These rooms are pre-lit, textured, and fully furnished. The applet then allows for a user to create their own building layout and populate the grid with room modules, or to have a completely random building automatically generated. The layout shown in Figure 3 was created automatically with the "random" option. It is also possible to have the applet insert random mazes as room modules.

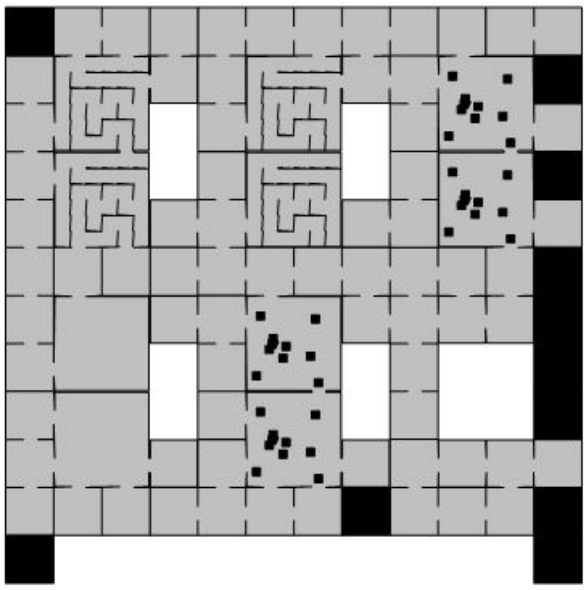

Figure 4: Ground truth of automatically generated building from Figure 3.

Vector ground truth for the building is also automatically generated. A screen shot of the ground truth may be seen in Figure 4. This ground truth contains the location and orientation of all of the static meshes (features such as desks, chairs, etc.) in the world as well as polylines and polygons for the building outline, room outlines, and mazes.

The building generator contains detailed documentation on how users can create their own library modules for use with the applet. It is anticipated that 10's of rooms will be available for use during the RoboCup 2010 event. Users with a working knowledge of the UnReal Editor are able to add their own room modules and expand this library.

Summary

The maze generator is already showing great impact on the RoboCup 2010 competition. Environments that previously took hundreds of hours to create are now being made in minutes. All of the worlds for the semi-finals and finals as well as the several of the preliminary round worlds will be created using this package. Once the package is officially released and comes to the attention of the user community, it is anticipated that it will also find great use in many research efforts that require a large variety of environments for algorithm testing.