A realistic RoboCup Rescue Simulation based on Gazebo

Shimizu, M.; Koenig, N.; Visser, A.; Takahashi, T.

Published in:
RoboCup 2015

DOI:
10.1007/978-3-319-29339-4_27

Citation for published version (APA):
Abstract. Since the first demonstration of the Virtual Robot Competition, USARSim has been used as the simulation interface and environment. The underlying simulation platform, Unreal Engine, has seen three major upgrades (UT2004, UT3 and UDK). These upgrades required a whole new USARSim simulator to be built from scratch. Yet, between those versions the USARSim interface has not been modified, which made USARSim a stable platform for more than 10 years. This stability allowed developers to concentrate on their control and perception algorithms. This paper describes a new prototype of the USARSim interface; implemented as plugin to Gazebo, the simulation environment native to ROS. This plugin would facilitate a shift of the maintenance of the simulation environment to the Open Source Robotics foundation and attract new teams to the Virtual Robot Competition.

1 Introduction

The challenge provided by the RoboCup Rescue Simulation League is to have a team of robots cooperate inside a devastated area. Most research institutes have access to only a few robots with a limited sensor suite and do not have access to all the robotic hardware necessary to build a complete rescue team. Simulators allow teams to experiment with algorithms for cooperation between robots in a safe, low-cost environment. However, to be useful, the simulator should provide realistic noise models for sensors and actuators; noise models which should be validated [4, 1, 14, 13, 5, 8, 2, 17].

The Robot Operating System (ROS) has been steadily gaining popularity among robotics researchers as an open source framework for robot control [15]. Gazebo is the simulation environment used by ROS, although it was originally developed for the Player-Stage environment [11]. Gazebo is based on the Open Dynamics Engine (ODE), although it has the flexibility to switch between physics engines. The Pioneer robot is validated based on the default ODE physics engine [7].

This is the author’s final version. The original publication is available at www.springerlink.com.
2 Related Research

The Unified System for Automation and Robot Simulation (USARSim) environment has been used for many years by robotics researchers and developers as a validated framework for simulation [6, 3]. The original version was developed in 2003, based on the concept of GameBots [10].

The validation approach applied to USARSim is to perform the same experiment in simulation and with a real world system, and to quantitatively compare the results. This effort may sometimes be costly, because it entails developing accurate models of the robotic systems at hand, but it has proved to be a formidable advantage which makes it possible to extrapolated from simulation to reality quickly and to identify early which algorithms are not generally applicable. Part of the USARSim success [3] draws from this extensive validation efforts. This validation has been performed for the Pioneer robot [4], the Kurt3D robot [1], the Kenaf robot [14], the Nao robot [13], the AR.Drone robot [17], the camera sensor [5], the laser sensor [8] and the GPS sensor [2]. It would be nice if not only the interface, but also part of this validation effort could be ported from USARSim to Gazebo.

Note that there exists an interface between ROS and USARSim [12], but this interface works precisely the other way around, making it possible for ROS-nodes to connect to simulation of USARSim. USARSim has many benefits (for instance, the realism of the lighting from the Unreal Engine), but is difficult to maintain with the current small developers community. Gazebo, the simulation environment native to ROS, is a much better choice for the future. To demonstrate the benefits of the change from USARSim based on Unreal to a USARSim based on Gazebo, this paper describes a prototype of such simulation environment. Table 1 shows functional comparison between USARSim and Gazebo. Table 2 shows merits of using Gazebo compared to USARSim. Those tables clearly show the possibilities of USARSim based on Gazebo for the RoboCup Virtual Robot League.

3 Benefits

There are several additional benefits of making this choice. Firstly the progress made by the Open Source Robotics foundation in improving Gazebo would be directly available to the RoboCup Rescue Simulation League community. The Open Source Robotics foundation recently extended the open source Gazebo robot simulator extensively on request of the Defense Advanced Research Projects Agency (DARPA). The new interface described in this paper would allow the teams active in the Rescue Simulation and the research institutes active in the DARPA Robotics Challenge to use the same simulation environment, allowing for cross development. Thirdly the maintenance of the simulation environment of the Virtual Robot competition would come in professional hands, now that USARSim is no longer actively supported by the National Institute of Standards and Technology (NIST). Last, but not least, this would allow to attract new teams to the RoboCup Rescue Simulation League.
Table 1. Functional comparison between UDK and Gazebo

<table>
<thead>
<tr>
<th></th>
<th>USARSim with UDK</th>
<th>USARSim with Gazebo</th>
</tr>
</thead>
<tbody>
<tr>
<td>Simulator</td>
<td>UDK</td>
<td>Gazebo</td>
</tr>
<tr>
<td>Changeability</td>
<td>Modification packages on UDK</td>
<td>All</td>
</tr>
<tr>
<td>Physics Engine</td>
<td>Unreal Engine</td>
<td>ODE(Default), Bullet, DART, Simbody</td>
</tr>
<tr>
<td>3D Simulation</td>
<td>Possible</td>
<td>Possible</td>
</tr>
<tr>
<td>Performance for</td>
<td>Real Time</td>
<td>Accuracy</td>
</tr>
<tr>
<td>Kinds of robot included by simulator</td>
<td>AirRobot, ATRVJr, Cooper, ERS, HMMWV, Kenaf, Kurt, Lisa, P2AT, P2DX, Pssarola, Qrio, Rugbot, Sedan, SnowStorm, So- ryu, Submarine, Talon, Tarantula, TeleMax, Zerg</td>
<td>Atlas, Kuka, Pioneer 2DX, Pioneer 3AT, PR2, Robo Naut, Quad Rotor, Kuka, youbot</td>
</tr>
<tr>
<td>Flying robots</td>
<td>Possible</td>
<td>Possible</td>
</tr>
<tr>
<td>Multi robots</td>
<td>Possible</td>
<td>Possible</td>
</tr>
<tr>
<td>Capability of adding objects and fields by users</td>
<td>Possible (Not so easy in scaling)</td>
<td>Possible</td>
</tr>
<tr>
<td>Capability of adding robot by users</td>
<td>Possible (Not so easy in scaling)</td>
<td>Possible</td>
</tr>
<tr>
<td>Realistic rendering</td>
<td>Impressive</td>
<td>Possible</td>
</tr>
<tr>
<td>Lighting and shadowing</td>
<td>Impressive</td>
<td>Possible</td>
</tr>
<tr>
<td>Simulation of water, mud, sand</td>
<td>Limited</td>
<td>Possible by change physics engine</td>
</tr>
<tr>
<td>Capability of connection with ROS</td>
<td>Possible</td>
<td>Possible</td>
</tr>
<tr>
<td>Arbitrary viewpoint</td>
<td>1</td>
<td>Any numbers of cameras which you want</td>
</tr>
<tr>
<td>Capability of connection with each camera video stream</td>
<td>Impossible (solved by tiled approach)</td>
<td>Possible</td>
</tr>
<tr>
<td>Getting Ground Truth data from map</td>
<td>Possible (done for competition visualization)</td>
<td>Possible</td>
</tr>
<tr>
<td>Active environment</td>
<td>Possible</td>
<td>Possible</td>
</tr>
<tr>
<td>Disaster environment</td>
<td>Possible</td>
<td>Possible</td>
</tr>
<tr>
<td>Movable obstacles</td>
<td>Possible</td>
<td>Possible</td>
</tr>
<tr>
<td>Fog effect</td>
<td>Possible</td>
<td>Possible</td>
</tr>
</tbody>
</table>
Table 2. Merits of using Gazebo instead of UDK

<table>
<thead>
<tr>
<th></th>
<th>USARSim with UDK</th>
<th>USARSim with Gazebo</th>
</tr>
</thead>
<tbody>
<tr>
<td>Commands and sensor data transferring protocol</td>
<td>GameBot Protocol</td>
<td>Topic or Any protocols which you need</td>
</tr>
<tr>
<td>Transferring data protocol with ROS</td>
<td>GameBot Protocol (Need protocol converter at ROS side)</td>
<td>Topic or Any protocols which you need</td>
</tr>
<tr>
<td>Changeability inside of simulator</td>
<td>Not yet (Unreal Engine 4 is Open Source)</td>
<td>Possible (Open Source)</td>
</tr>
<tr>
<td>Programming Language</td>
<td>Unreal Script</td>
<td>C, C++, Python</td>
</tr>
</tbody>
</table>

4 Design

The interface between Gazebo and USARSim is designed as a WorldPlugin; the preferred method to modify the simulation environment. The plugin starts a server-routine, which listens to port 3000. Multiple clients (currently limited to teams of 16 robot controllers, as in the original UT2004 version) can connect to this port and spawn a robot into the Gazebo world.

At the moment, the robot is spawned with a specific sensor-suite. In principal, an user can modify this configuration in the Gazebo GUI. The USARSim interface has commands to query the current configuration (GETCONF and GETGEO commands), but the format of the STATUS message should be updated to notify the robot controller that the configuration has been changed (and should be queried again).

The location of the spawn-position is important, because with a team of robots a designer wants each robot to have an unique start position. In an Unreal world, start-positions are specified by inserting PlayerStart positions with their corresponding coordinate system to the world. This is a native feature of Unreal, because Unreal Tournament was a multi-player game where each player also needed an unique start position. Gazebo has a comparable way to specify start positions, once a PlayerStart model is created.

Once a robot is spawned and configured, the regular sense-plan-act cycle starts. Sensor messages are received via SEN messages, the actuators are controlled by DRIVE, SET and MISPKG messages.

The implementation of the interface is based on the efficient Boost-library [16], the same library which is used inside Gazebo.

5 Requirements

The goal of this study is to create a fully functional prototype, which will allow to control robots inside Gazebo via the USARSim interface. This would mean that
– New robots could be dynamically spawned in the world by an INIT command.
– The robots could be configured with SET commands.
– The robot’s sensor suite could be queried with GETCONF and GETGEO commands.
– The robots could be steered by sending DRIVE commands.
– Sensor updates would be sent via SEN messages.
– Camera images would be published via a separate high-speed socket (typically port 5003)
– A private socket (typically port 50000) should be available for the Wireless Server Simulation, which needs Ground Truth information to calculate distances between robots and the number of walls in the line of sight between the robots.

6 Architecture

This USARSim prototype is built by a diagram which indicates how the USARSim interface should be incorporated in the Gazebo architecture. This diagram (Figure 1) shows connections between an USARSim user client and Gazebo simulator via the new plugin.

![Diagram of our prototype simulator with Gazebo. At right middle of the diagram, white blocks are our added plugin software in Gazebo. The plugin translate bidirectionally USARSim commands and Camera images and Sensor data between USARSim protocol (GameBot) and Gazebo Native protocol (Topic).](image)

Fig. 1. Diagram of our prototype simulator with Gazebo. At right middle of the diagram, white blocks are our added plugin software in Gazebo. The plugin translate bidirectionally USARSim commands and Camera images and Sensor data between USARSim protocol (GameBot) and Gazebo Native protocol (Topic).
7 Results

In the current implementation\textsuperscript{5} it is possible to query for start poses, to spawn a Pioneer 3AT robot and to publish images over a high speed binary channel. The later accomplishment was the most critical.

![Fig. 2. A screenshot of a world in Gazebo, converted from a USARSim world created in Unreal Editor: RoboCup 2013 Virtual Robot competition Preliminary 1.](image)

We were also able to import a world generated in Unreal Editor into Gazebo. This is quite attractive scenario, because the Unreal Editor is really very professional. The latest version of Unreal, the Open Source version 4.7, is able to create very large maps, which is essential in rescue situations. In addition, a lot of effort is spent to create the realistic worlds for the RoboCup Virtual Robot competition. With this method the existing maps can be ported to Gazebo. An example of the level of detail needed to provide Ground Truth data for the Wireless Simulation Server over socket 5000 can be seen in Figure 3.

8 Future Work

One of the assets of USARSim is the focus on validation. It would be an advantage if all validation effort \cite{4, 1, 14, 13, 5, 8, 2, 17} could be repeated in the Gazebo environment. This is not only beneficial for the RoboCup community, but for all users of Gazebo. The first validation performed which could be performed is to perform driving experiments with the Pioneer 3AT; accelerating along a straight trajectory and turning circles. A comparison could be made between the real system, USARSim based on the Unreal Engine and USARSim based on Gazebo.

\textsuperscript{5} Can be downloaded from https://github.com/m-shimizu/RoboCupRescuePackage
This could be continued with experiments on slopes and trajectories over obstacles. The NIST institute has made a very useful test-document for Rescue Robots, with a wide variety of experiments which could be performed [9]. Because the effort of many RoboCup Rescue teams concentrate on mapping and object recognition, a comparable set of experiments could be performed to estimate the level of realism for each sensor.

9 Conclusion

The new prototype will be presented at the RoboCup 2015 in China. In addition, the teams will be allowed to stress-test the solution in their laboratory. When tested by the experienced teams, the new USARSim interface to Gazebo will be presented on the Leagues website and in relevant newsgroups and social media. It is the intention to accompany this announcement with an invitation for a workshop, at an international robotics conference, to introduce the new design to a larger audience. It will be used as showcase for both the RoboCup and the robotics rescue community as a whole.

Acknowledgement

This project was supported by RoboCup Foundation 2015 (Application Title:”Building a USARSim interface inside Gazebo”).
References


