Player/Gazebo Simulation Environment

John I. Martin, Keith Redmill
{martinj,redmill}@ece.osu.edu
15 December 2006

Introduction

The Player/Gazebo simulation environment provides a virtual world in which high level robot control code may be tested. The two primary parts of Player/Gazebo consist of "Player", which is a robot-independent interface layer, and "Gazebo", which provides a virtual environment which acts upon the robot. Figure 1 tries to show the relationship between these software components.

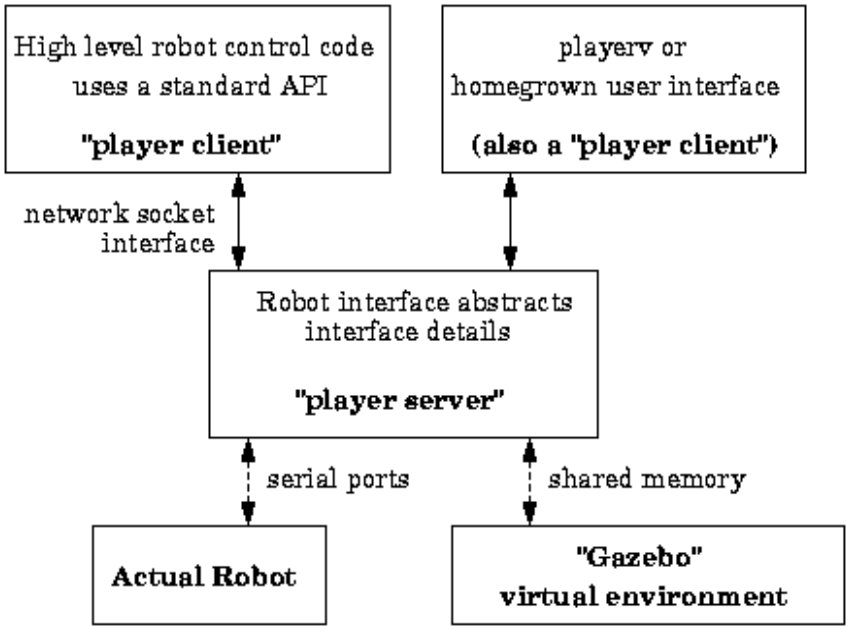

Figure 1

Note that Figure 1 shows that the Player server may either be communicating with an actual robot or it may be communicating with a virtual environment.

The installed version of Player is 1.6.5 and of Gazebo is 0.5.3.

Logging in

The Player/Gazebo software is currently running on "cccstb-pc-1.ece.ohio-state.edu". You may log in from the console or by using an ssh client from another Linux box, HP workstation, or other computer with X-Windows server software. Usernames and passwords will be supplied by email. Under Linux, the command which starts an ssh client could look like:

[martinj@cccstb-pc-2]$ ssh -X cccstb-pc-1.ece.ohio-state.edu

Please note that graphics will be much slower when connecting remotely via ssh.

The Gazebo Configuration File

The "wxgazebo" and "gazebo" program is run from the command line. The configuration information which Gazebo requires may be found in an text-based xml file specified on the command line. An example configuration file may be found at "/home/redmill/Examples/gazebo_examples/example1.world". To run Gazebo, you may copy "example1.world" to your home directory on "dhcp5220.ece.ohio-state.edu". "example1.world" references two other files, "sample_walls.gzb" and "grid.ppm". You should copy those from "/home/redmill/Examples/gazebo_examples/" as well.

The configuration file consists of the following sections:

- Gravity Configuration
- Observer Window Configuration
- Light Source Configuration
- Ground Texture (in this case it is grid.ppm)
- Terrain Map (in this case it sample_walls.gzb)
- Simple Solid (an easy way to drop simple things into the world)
- Robot model with the available sensors and configurations

The robot model used in the Gazebo simulation is a small robot called a Pioneer 2. It is similar enough to the Pioneer 3 to enable us to perform initial testing without dealing with the difficulties of actual hardware.

Note: sample_walls.gzb was built from sample_walls.png (a bitmapped image) using:
"gzbuilder -i sample_walls.png -o sample_walls.gzb -h 0.1 -v 0.1 -e5".

Running Gazebo

After all the necessary configuration files exist in your home directory, you may run "Gazebo" by issuing the following command:

[martinj@cccstb-pc-2]$ wxgazebo example1.world

The "[martinj@cccstb-pc-2]$" is part of the command line prompt-- you shouldn't type this. Also, the command line prompt will have your username within it (instead of mine). Gazebo can also be run without the graphical interface using "gazebo example1.world".

A window which looks like Figure 2 will appear, as well as a smaller window that contains simulation control and status functions. Click on the [] boxes to view/hide the various control or status windows for different parts of the simulation. For example, if you check [x] Pioneer2AT:position:robot1, a window will open that will allow you to drive the robot manually- using "(lin vel x)" to move forward and backwards and "(cmd ang vel Z)" to rotate.

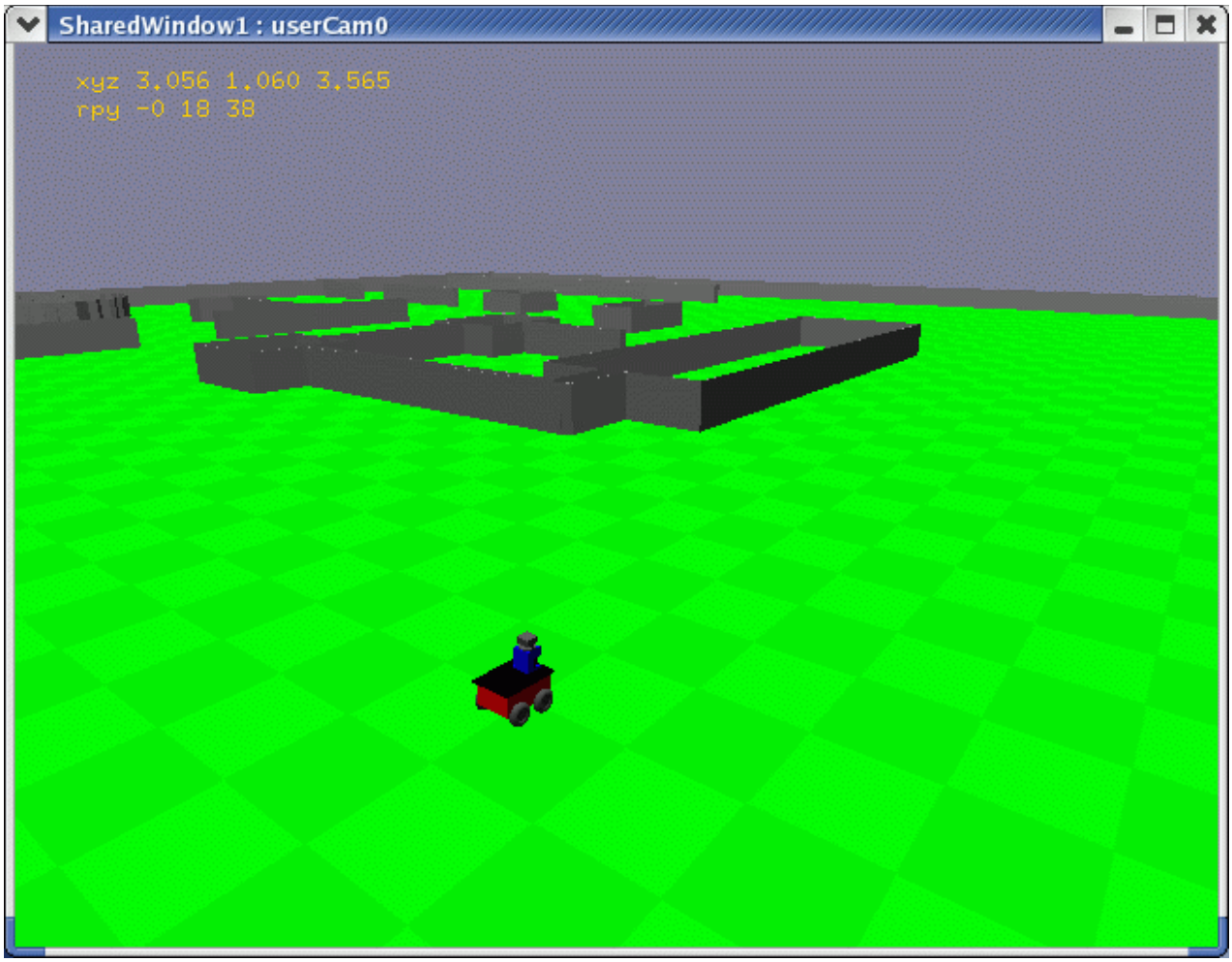

Figure 2

The Player Configuration File

The "Player" server requires a configuration file as well. The Player configuration file may be found at "/home/redmill/Examples/gazebo_examples/player_gz_example.cfg". You may copy that configuration file to your home directory. The Player configuration file tells Player what interfaces it should make available to it's clients. The configuration file also tells the Player server how to manipulate devices on the low level side. The Player configuration file is text-based so you may edit it with any editor. The player configuration file consists of the following sections which define the available interfaces:

- A request for a simulation interface which gives general "world" s tatus
- A position interface which allows the client to move the robot in an abstract fashion
- The Sick LMS 200 laser range finder interface
- The onboard camera image
- The sonar interface
- The GPS interface

Running Player

You may run Player for operation with "gazebo" by issuing the following command:

[martinj@cccstb-pc-2 martinj]$ player -g default player_gz_example.cfg

Player should output the following:

```
** Player v1.6.5 **
* Part of the Player/Stage Project [http://playerstage.sourceforge.net].
* Copyright 2000-2005 Brian Gerkey, Richard Vaughan, Andrew Howard,
* Nate Koenig and contributors.
* Released under the GNU General Public License.
Startup options: [TCP]
gz_client.c:122 opening /tmp/gazebo-redmill-0
gz_iface.c:214 opening /tmp/gazebo-redmill-0/sim.default 050 180

Parsing configuration file "player_gz_example.cfg"
Using device table:
----------------------------------------------------------
driver gz_sim provides 6665:simulation:0
driver gz_position provides 6665:position:0
driver gz_laser provides 6665:laser:0
driver gz_camera provides 6665:camera:0
driver gz_sonar provides 6665:sonar:0
driver gz_gps provides 6665:gps:0
----------------------------------------------------------
listening on ports: 6665
```

Player is now waiting for connections from client programs which may connect to any of the above devices. For example, the "simulation:0" device provides Gazebo specific information while "position:0" provides an interface for moving and estimating the position of the robot.

Running a Player Client

Player provides an example client application called "playerv". You may run this program by typing "playerv" at the command prompt. Figure 3 shows Playerv's GUI. Playerv allows the user to send commands to the robot running in the Gazebo simulation. You may drive the robot by using the "devices->position->subscribe" followed by "devices->position->command" selections and then dragging the cross-hairs with the mouse. You may view sensor outputs such as the sonar and lasers by selecting "devices->laser->subscribe" and "devices->sonar->subscribe".

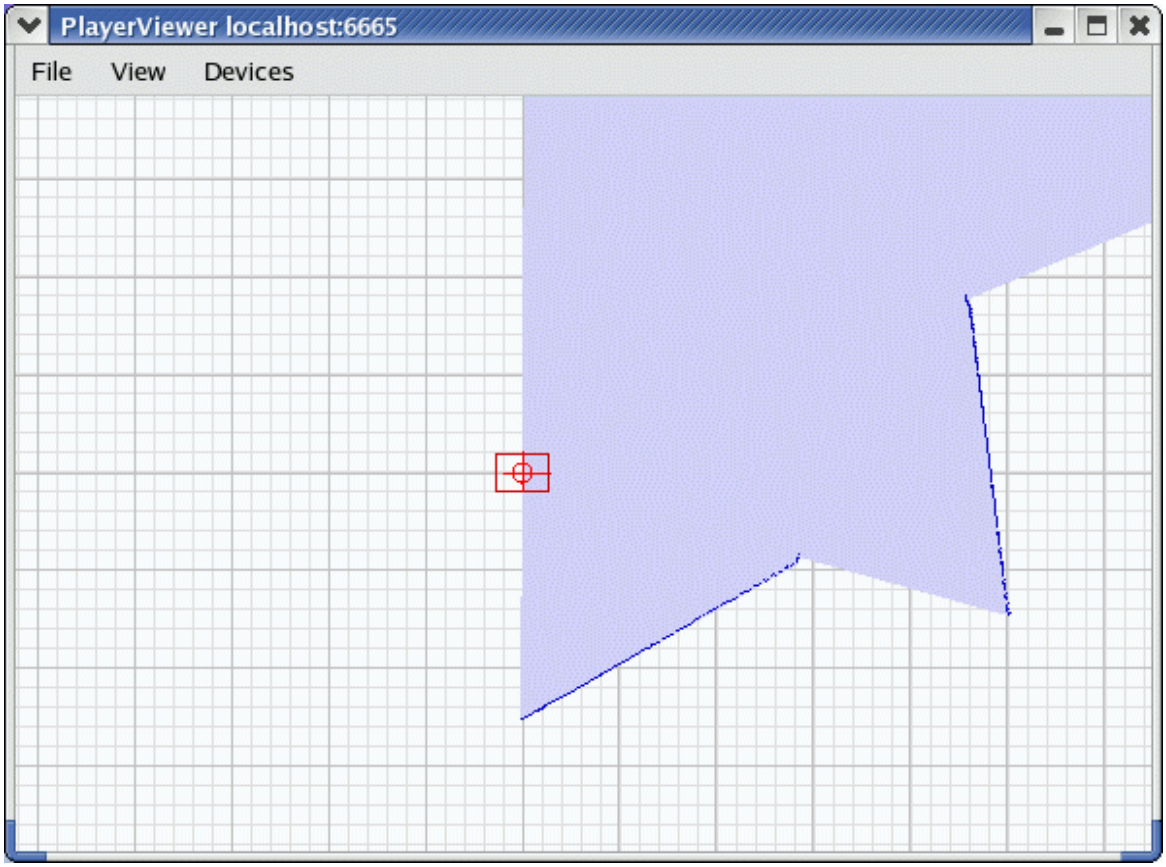

Figure 3

Running a Homegrown Player Client

"playerv" functions as a quick example interface for commanding a robot in Gazebo. However, the goal should be to write code which allows a robot to function autonomously. There are two examples of source code for moving the Gazebo robot. These are in the "/home/redmill/Examples/client_examples/example1-turn" and the "/home/redmill/Examples/client_examples/example2-dodge" directories.

If you run the binary found in the "example1-turn" directory, the Gazebo robot will spin about it's center axis. If you run the binary found in the "example2-dodge" directory, the Gazebo robot will move around while using a simple obstacle avoidance algorithm. For further details, check out the code. Both of these programs may be used in an actual robot (with a different player configuration file).

The Shared Memory Status Viewer

The example client code found in the "example2-dodge" directory keeps the current robot status in a chunk of shared memory. This status may be viewed graphically by running the

"/home/redmill/Examples/client_examples/shm_display/robot_display"

program which will read the shared memory and present the laser and gps information in a GUI-type application as shown in Figure 4.

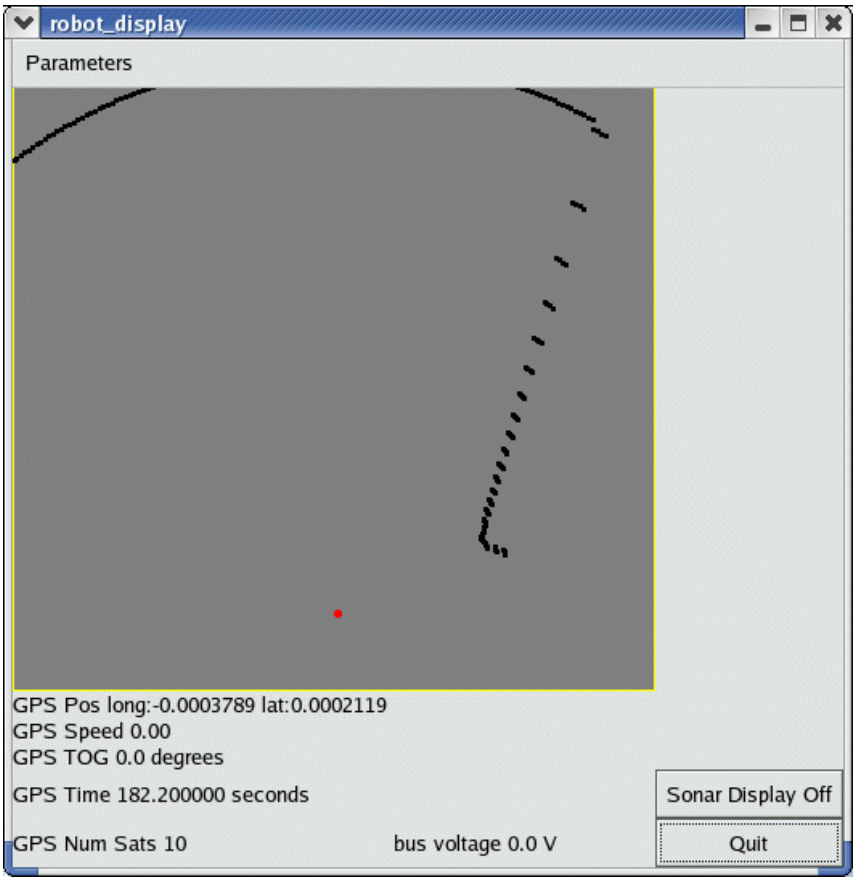

Figure 4

Further Information

For further information you may find the Player/Gazebo documentation at:

http://playerstage.sourceforge.net/doc/Gazebo-manual-0.5-html/

http://playerstage.sourceforge.net/doc/Player-1.6.5/player-html/