

PolyBot Simulation Environment

Ying Zhang
May 3, 2000

The hardware for the **PolyBot** robotic system is still under design. A simulation environment for the distributed control of the locomotion and reconfiguration for **PolyBot** robots is designed and implemented to prove the feasibility of control systems while the hardware is being built. The simulation can be found on the web site at <http://www.parc.xerox.com/modrobots/PolyBot/simulation>. This document explains how to use this simulation environment.

1. Overview

The simulation environment is implemented in **Java** Platform 2 with **Java 3D**. There are three modes of operation: stand-alone, applet, and remote server. Figure 1 shows the main view of the environment when it runs stand-alone or remote server.

The applet can be directly viewed from web browsers, in addition to the *appletviewer*. The ability to view through web browsers allows people who are curious about these systems to try it without actually downloading and building the software.

There are two levels of usage for this environment, curious and serious.

1. Curious users try the simulation in the above web page in its applet mode to get an understanding about the **PolyBot** modules.
2. Serious users will download the jar file, write locomotion or reconfiguration code and use the java RMI interface to control **PolyBot**.

2. Main Features

This simulation has the following features:

1. Simulates static stability, i.e., the robot is always in a stable position, that is, the gravity center is always within the ground supporting points. Friction is not simulated, but implied by the “balancing” algorithm.
2. Simulates faulty modules.
3. Generates four types of initial configurations from GUI, and any initial configuration from a text input file.
4. Generates various floor types, including flat, slope, step, wall and ditch.
5. RMI interface for setting and getting joint angles, obtaining 6D offset for any two faces, and attaching and detaching two faces.

Note that this simulation only works for **PolyBot** structures with no loops. Structures with loops are over-constrained. The simulation does not solve the constraint satisfaction problem. The simulation does not detect self-collision either.

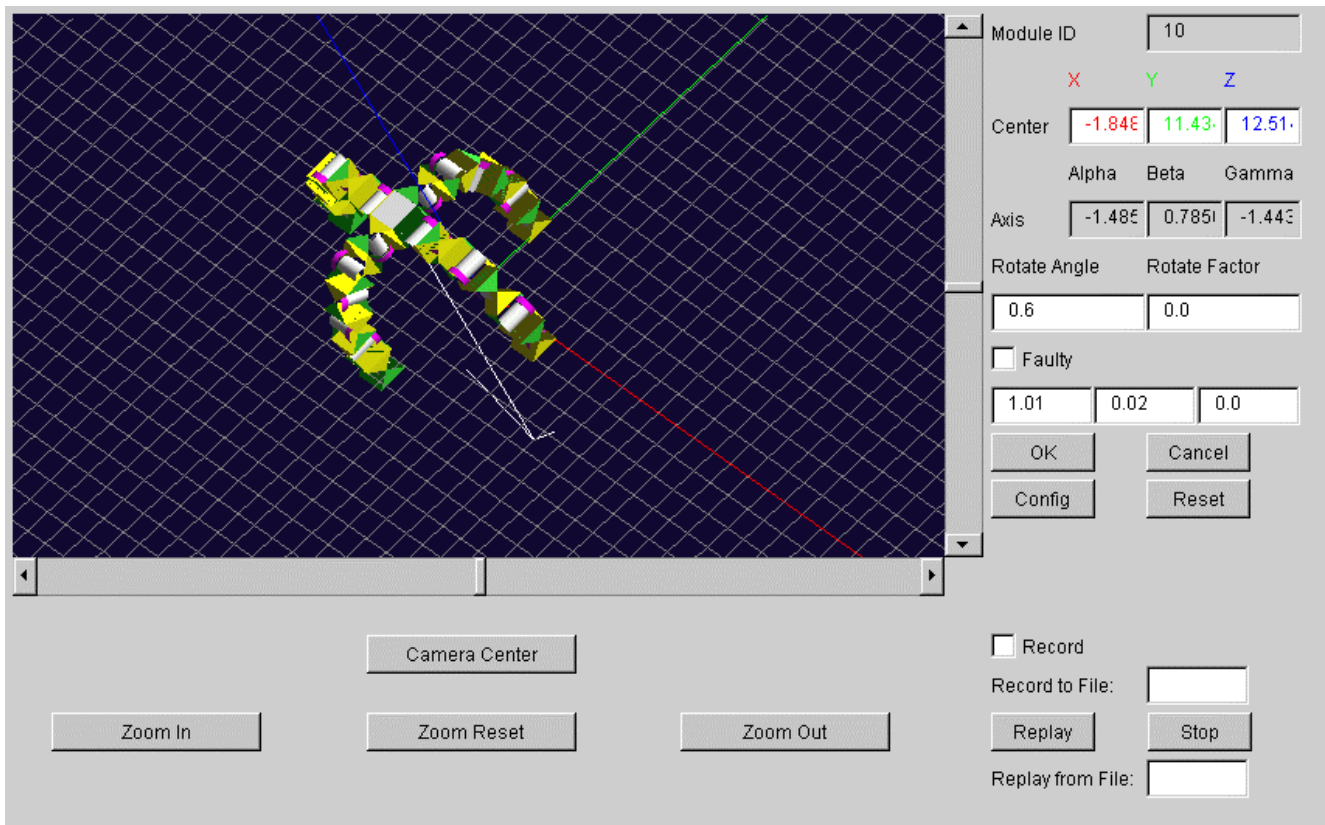


Figure 1. PolyBot Simulation Environment

3. GUI

3.1 3D Canvas

The 3D canvas displays the 3D modules. The modules can be viewed from any angle by simultaneously clicking and dragging the mouse. The horizontal and vertical sliding bars are used for moving the point of view. In addition, there are zooming and centering buttons for better viewing.

3.2 Initial Configuration

The **Config** button pops up a configuration-setting window for selecting four different initial types of configurations, *snake0*, *snake1*, *snake2*, and *spider*, with various sizes, and setting the 6D position for the first module. In addition, a file input can be defined for any initial configuration. This can only be used for the stand-alone or remote server mode. The format of the input files is in “PBSim/infile”. There are four types of floors: flat, step, wall or ditch. A norm can be defined for the flat floor; slop, width and height can be defined for the step, wall or ditch. (Note that ditch is a wall with a negative height).

3.3 Status and Information

On the right side of the 3D canvas, there are fields for displaying related information. Use the mouse to pick a module, which will be highlighted in red in the 3D canvas. The module ID, and its 6D position will be displayed on the right side panel. If the module is a segment with joint actuation, the joint angle and its motion factor will be shown. The motion factor is a number from 0 to 1, characterizing which direction the joint is moving: 0 means the green face is fixed and 1 means the yellow face is fixed, 0.5 means both are moving. The 3D position, the joint angle and its motion factor can be set when **OK** is hit. The **Cancel** button ignores any change for the module. When **Faulty** is checked, the faulty module is simulated by equation: $a \times \alpha + b \times r + c$ where α is the desired angle and r is a random variable ranging from -1 to 1 , and a , b and c are constants, which are displayed below the checkbox and can be changed. The **Reset** button sets all the joint angles to 0.

3.4 Record and Replay

The major difference in the GUI between the applet and stand-alone modes is the middle right side of the 3D canvas. In the applet mode, control status is displayed and in the stand-alone mode, **Record** checkbox and **Replay** buttons are shown. When **Record** is checked, the changes in joint angles will be recorded to a file, specified in the text field below. And **Replay** will replay the recorded file, specified in the text field below. The recorded file is always put in the directory “PBSim/record” with .rec surfix. For example, “test” in **Record to File** will generate a file “PBSim/record/test.rec” when **Record** is checked. And “test” in **Replay from File** will replay the previous recorded process in “PBSim/record/test.rec” after hitting **Replay**.

3.5 Remote Server

In addition to the applet and stand-alone modes, remote server mode can also be used. This mode is most useful for testing control algorithms, namely controlling the robot via an RMI interface. The details of the interface functions can be found in “PBSim/server”. To run the server, just like any RMI server, run the `rmiregistry` first, and then

```
java -Djava.security.policy=PBSim/policy PBSim.server.SimImpl
```

To test it, also run a test client:

```
appletviewer SimTest.html
```