

# Player-Stage Tutorial

CS594 – Machine Learning

Rasko Pjesivac

pjesivac AT cs.utk.edu

## Player

### ==== What is Player? =====

Player is a network server for robot control. Player can be running on your robot, or if used with Stage or Gazebo simulator on your machine. Player provides a clean and simple interface to the robot's sensors and actuators. Your client program talks to Player over a TCP socket, reading data from sensors, writing commands to actuators, and configuring devices on the fly.

Client program can be written in any of the following languages: C++, Tcl, JAVA, and Python. In addition, the client program can be run from any machine that has a network connection to your robot or to the machine on which your simulator is running.

Link to the official Player Robot Device Tutorial:

<http://playerstage.sourceforge.net/doc/Player-cvs/player/index.html>

### ==== Using Player =====

In order for Player to know which drivers it needs to instantiate we need to create a configuration file. This configuration file is a plain text file with *\*.cfg* extension, and it is composed of one or more **driver** sections.

For each driver we can specify following **options**:

----

**name**: The name of a driver to instantiate (required)

**plugin** : The name of a shared library that contains the driver (optional)

**provides**: The device address(es) through which the driver can be accessed. Can be combination of strings. (required)

**requires:** The device address(es) to which the driver will subscribe. Can be combination of strings. (optional)

**always\_on:** If 1, then the driver will be setup when the player server starts, without waiting for any client connection. (optional)

----

Bellow is an example how to write simple drivers in configuration file

```
# example.cfg
# Loads a driver for SICK laser
driver
(
    name "sicklms200"           # The driver's name
    provides ["laser:0"]       # The device address
)
# Loads a vfh (vector field histogram) driver used for
# local navigation and obstacle avoidance
driver
(
    name "vfh"
    provides ["position2d:1"]
    requires ["position2d:0" "laser:0"]
)
```

Once configuration file is created we just need to run following command:

```
linux:~>/working_directory/player example.cfg
```

## ==== Stage Simulator =====

### ==== What is Stage? =====

Stage simulates a population of mobile robots, sensors and objects in a 2D environment. Stage provides several different models for robot's sensors and actuators. Following are currently supported models: **sonar** and **infrared** rangefinders, scanning **laser** rangefinder, **color-blob** tracking, **fiducial** tracking, **bumpers**, **grippers** and mobile robot bases with **odometric** and **global localization**.

Link to official Stage tutorial:

[http://playerstage.sourceforge.net/doc/stage-cvs/group\\_\\_stage.html](http://playerstage.sourceforge.net/doc/stage-cvs/group__stage.html)

### ==== Using Stage with Player=====

Stage simulates an environment composed of different models defined in a "**world**" file.

In **.world** file we define all necessary information about the "world" in which our robots are running.

For example, we need to define **resolution** property which is the size of a pixel in meters, then **interval\_sim** and **interval\_real** which represent milliseconds per update step for both simulation and real time, **gui\_interval**, etc)

Besides defining parameters about the world we also need to define which models we want to load and the map that we want to use for the simulation.

Bellow is an example of world file.

```
# simple.world
# the size of a pixel in Stage's (in meters)
resolution      0.02

# milliseconds per update step
interval_sim 100
# real-time milliseconds per update step
interval_real 100

# defines Pioneer-like robots
include "pioneer.inc"

# defines 'map' object used for floorplans
include "map.inc"

# defines the laser model `sick_laser' (Sick LMS-200)
include "sick.inc"
```

```

# Size of the world in meters
size [40 20 ]

# GUI options
gui_disable 0
gui_interval 100
gui_menu_interval 20

# Configure GUI window
window(
  size [ 755.000 684.000 ]
  center [-7.707 2.553]
  scale 0.009
)

# Load an environment bitmap
map(
  bitmap "sample.png"
  map_resolution 0.02
  size [40 18]
  name "sample"
)

```

Stage is not a program that you run standalone, and there is no binary called "Stage". Instead, Stage provides a Player extension, or "plugin", which adds simulated robots to Player. To use the Player/Stage system, you run the "player" program with appropriate configuration files.

This is how we define stage driver in the configuration file.

```

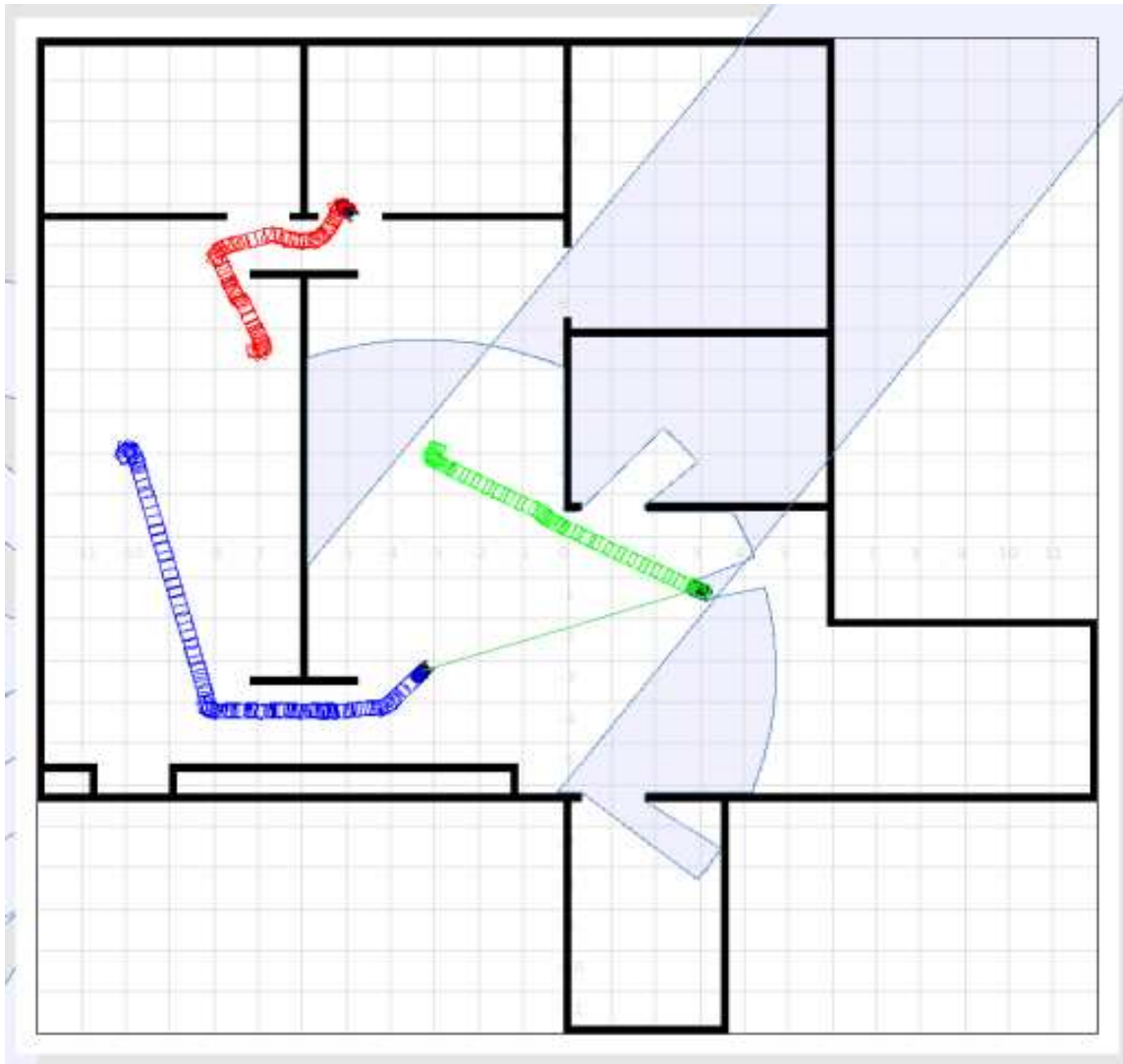
#simple.cfg
# load the Stage plugin simulation driver
driver
(
  name "stage"
  provides ["simulation:0" ]
  plugin "libstageplugin"
  # load the named file into the simulator
  worldfile "simple.world"
)
# Create a Stage driver and attach position2d and laser
# interfaces to the model "robot1"
driver
(
  name "stage"
  provides ["position2d:0" "laser:0" ]
  model "robot1"
)

```

Once we have created both **simple.world** and **simple.cfg** we just need to execute the following command in order to start the simulation:

```
linux:~>/working_directory/player simple.cfg
```

## Screenshot of Player/Stage simulation.



## ===== Code Examples =====

Example configuration and world files for Player/Stage can be found in  
/research/playerstage/examples/ directory .

Name of the files are: **everything.cfg**, **everything.world**, **simple.cfg**, and **simple.world**.

Example of a client code and Makefile that can be used to run a robot in these two  
environments are also provided in the same directory.

Name of the files are: **laserobstacleavoid.cc** and **Makefile**.

## ===== Useful Player/Stage links =====

\* Supported devices (robots, hardware, software, simulators, and algorithms)

[http://playerstage.sourceforge.net/doc/Player-cvs/player/supported\\_hardware.html](http://playerstage.sourceforge.net/doc/Player-cvs/player/supported_hardware.html)

\* Detailed description of all drivers for Player

[http://playerstage.sourceforge.net/doc/Player-cvs/player/group\\_drivers.html](http://playerstage.sourceforge.net/doc/Player-cvs/player/group_drivers.html)

## ===== Environments Paths =====

In order to be able to run the client code with the simulator the following environment  
paths need to be set for your shell. Example for **zsh** (*.zshrc*)

```
export PATH="$PATH:/pkgs/player-stage-2.0.3/bin"  
export PLAYERPATH="/usr/local/lib"  
export LD_LIBRARY_PATH="/usr/local/lib:/usr/local/pkgs/Python-  
2.4.1/lib"
```