HIRO Robot Setup

# Overview

There are three computers:

1. Standing terminal – runs Windows.
2. Linux Desktop Station
3. HIRO’s QNX computer.

## Running the forceSensor Plugin

* Turn on the HIRO Robot by turning the switch on the back.

### **qnx Terminal**

* To run the controller on HIRO robot from QNX, a script called **hrpsys.sh** has to be called from the Windows terminal.
  + Normally, the script is running, but in case of having to do an emergency reset, it has to be restarted.
* In case of reset perform the following steps:
  + Open the Windows **Putty** program.
  + Click on the pre-saved “Robot connection configuration”.
  + A shell window will open.
  + Type in **hrpuser** both as the *username* and the *password*.
    - Login as a super user by first typing “su” and then typing “hrpuser” without the quotes.
    - Kill the running but defunct program by typing “slay hrpsys”
    - Run the new script by typing: “sh script/hrpsys.sh”

### **Linux Desktop Terminal**

* For the first time user, one must personally configure a version of the forceSensorPlugin in the QNX machine via the linux machine to run the user’s own code.
* To do so, we will use SSH:
  + Type: “ssh [hrpuser@192.168.128.2](mailto:hrpuser@192.168.128.2)”
  + Type the following password upon request: “vmrguser”
  + Change directories to the forceSensorPlugin: “cd forceSensorPlugin”
* At this stage create a folder with your own name appended to forceSensorPlugin such as “forceSensorPlugin\_YourName”.
  + Copy the existing files under forceSensorPlugin into your new directory
  + Ensure that all necessary header files, library files, path names, and makefile configuration options have been updated.
  + “Make” the files. Note that if another user uses the terminal before you, they could have tested their own files by Making their own version of the code. Make sure you re-make yours each time. Also look at the section named “*Compiling Additional Classes*” in this document in case you need to rename the DynamicsSimulator.orig folder.

### Windows Standing Terminal

* After the forceSensorPlugin has been made, the actual program will be called from the windows terminal.
* Here are the basic steps to run the plugin:
  + Double click on the GrxUI.exe icon
  + Load the HRPSTEP2 .xml file
  + When the simulation loads, under the IOBClient tab, click on the RobotSetup icon
  + A new window will pop-up, there click on:
    - Servo On
    - Joint Calibration
    - Start Control
* Personal Configurations:
  + If you would like to personalize the position of the robot’s homing position, there are two ways to do so:
    - From within your forceSensorPlugin, or
    - Using a Python Script.
* Yamanobe’s Python script:
  + On the Window’s Terminal Station you can find a python generated GUI “gui\_yamanobe5.py” that has a set of preconfigured modes and routines. This can be a sample program to personalize your own.
  + The source code can be viewed by going to the source directory of the shortcut.
  + This program would have to be copied to a USB device and modified in another computer, and then copied back into this station to be executed.

# Turning off the robot

1. From the linux desktop type [halt@192.168.128.2](mailto:halt@192.168.128.2) and type “halt” as the password.

# Tools

1. Using **Nautilus** to move/copy/transfer files through ssh/ftp/sftp is much easier.
   * To go to a desired location, go to nautilus and click “Ctrl+L” to open the location bar.
   * You can type” ssh://username@ssh.server.com” or “ssh://username@ip”
   * Same for ftp or sftp

# compiling additional Classes

1. If you have any header files that refer to OpenHRP files like: link.h, body.h, linkpath.h, etc;
   * You will have to update their path address. You will want to redirect them to */home/hrpuser/src/OpenHRP3/DynamicsSimulator/server*
   * If you use of the function **attitude()** anywhere in your code, note that the definition for this function is found in the same folder as above but where *DynamicsSimulator* is *DynamicsSimulator.orig*.   
       
     The easiest way to deal with this discrepancy in the QNX terminal is to change the name of the *DynamicsSimulator.orig* folder to *DynamicsSimulator* and to change the name of *DynamicsSimulator* to *DynamicsSimulatorTemp* each time you work on the machine.
   * Look for any other differences between QNX and Linux terminals.
2. SSH into the QNX Machine.

### Build external Clasess.

1. Go to the folder repository of your external classes. Typically found at: \home\hrpuser\YourName\
   * You will MAKE the files. But before you MAKE the files, check for the following:
   * Inside the Make file:
     + From personal experience, the makefiles were most easily built when the include/library paths for tvment/boost libraries where included manually.
     + Search their corresponding paths carefully.
     + QNX and Boost:  
       Boost has special libraries specially designed for QNX. They are boost\_filesystem, boost\_regex, and boost\_thread and need to be called with the BOOST\_LIB\_SFX flag.
   * MAKE each external class.
   * You can set your make file to save the header and shared library files to the following two directories respectively:
     + header’s: \home\hrpuser\src\OpenHRP3\Controller\IOServer\include
     + shared library:\home\hrpuser\src\OpenHRP3\Controller\IOServer\robot\HRP3STEP2\bin

### Force Sensor plugin

1. Go the your personal forceSensorPlugin folder typically found at:z
   * \home\hrpuser\forceSensorPlugin\_YourName
   * Verify that your forceSensor\_Plugin complies with the QNX OS.
2. Verify that the makefile reads from the correct directories for libraries and headers.
3. Make the file.
   * A shared lib file will be stored at:

\home\hrpuser\src\OpenHRP3\Controller\IOServer\robot\HRP3STEP2\bin\

# Code repository

|  |  |  |
| --- | --- | --- |
| Linux | Directory | \home\vmrguser\Documents\YourName\OpenHRP3.0-HIRO |
| QNX | OpenHRP Source | \home\hrpuser\src\OpenHRP3\Controller |
| ForceSensorPlugin\_Yours | \home\hrpuser\forceSensorPlugin\_YourName |
| External Code Repository | \home\hrpuser\YourName |

Two sources at different terminals

* **Linux**:  
    
  Include Files:   
  \home\vmrguser\Documents\YourName\OpenHRP3.0-HIRO\Controller\IOServer\include  
    
  Shared libraries:  
  \home\vmrguser\Documents\YourName\OpenHRP3.0-HIRO\Controller\IOServer\robot\HRP3STEP2\bin
* **QNX**:   
    
  ForceSensorPlugin:  
  \home\hrpuser\forceSensorPlugin\_YourName  
    
  Source Code:  
  \home\hrpuser\src\OpenHRP3   
    
  Header Files:  
  \home\hrpuser\src\OpenHR3\Controller\IOServer\Include\  
    
  Library Files:  
  \home\hrpuser\src\OpenHRP3\Controller\IOServer\robot\HRP3STEP2\bin  
    
  Data Files (Trajectory Files and Results):  
  \home\hrpuser\forceSensorPlugin\_YourName\data  
    
  Additional Local Copy of code  
  External Classes (include files and library files):  
  \home\hrpuser\YourName\  
  \* Usually build these first, and then copy to local folders within QNX