Robai Cyton Gamma 300 Robotic Arm System Outline

**1. Abstract**

The goal of this project is to evaluate different controller’s effectiveness at performing multiple types of servicing operations as well as evaluating the capabilities and effectiveness of ROS. These tasks will be done through connecting Xbox 360 Controller, Wii U Gamepad, NVIDIA Shield, and Space Navigator to the Robai Cyton Gamma 300 by way of developing ROS plugins and manual controller interfaces.

**2.  Systems Engineering**

**2.1.   Deliverables**

The project's goals stated above outline the deliverables for the end of the project:

    1. Code to sync Wii U Gamepad with both and computer and Cyton Gamma 300            arm.

    2. Code to sync NVIDIA Shield with both and computer and Cyton Gamma 300            arm.

    3. Report on the effectiveness and capabilities of ROS.

    4. Report on effectiveness of each controller to perform servicing tasks.

**2.2.   Timeline**

See Appendix A

**3. Procedure**

**3.1. Initial Setup**

3.1.1.Required Software

In order to begin using the Cyton Gamma 300 certain software is required:

* Ubuntu 12.04 (ROS Hydro is not compatible with 14.04)
* Cyton Viewer v3.3.3.A12DE6
* Actin SE Kinematics software
* ROS Hydro
* ROS\_Cyton\_Gamma package

3.1.2. Procedure

The required software should be installed in the following order:

* Install the Cyton viewer (Provided by Robai)
* Includes Actin SE kinematics software (Provided by Robai)
* Install ROS Hydro
  + Full instructions can be found at <http://wiki.ros.org/hydro/Installation/Ubuntu>
  + run the following commands in the terminal
  + Setup sources.list
    - $sudo sh -c 'echo "deb http://packages.ros.org/ros/ubuntu precise main" > /etc/apt/sources.list.d/ros-latest.list'
  + Setup Keys
    - $ wget https://raw.githubusercontent.com/ros/rosdistro/master/ros.key -O - | sudo apt-key add -
  + Installation
    - $ sudo apt-get install ros-hydro-desktop-full
  + Initialize rosdep
    - $ sudo rosdep init
    - $ rosdep update
  + Environment Setup
    - $ echo "source /opt/ros/hydro/setup.bash" >> ~/.bashrc
    - $ source ~/.bashrc
  + Getting rosinstall
    - $ sudo apt-get install python-rosinstall
* Install the ROS\_Cyton\_Gamma package
  + Provided by Robai upon request

**3.2. Controller Interface**

3.2.1.Xbox 360 Controller

3.2.1.1. Software Development

3.2.1.2. Hardware Implementation

3.2.2.Wii U Gamepad

3.2.2.1. Software Development

In order to develop applications for the Wii U, it first must be synched to a computer. Detailed instructions to sync the Wii U Gamepad were provided by the Libdrc.org, a website dedicated to open source development of Wii U programming and applications. Certain software packages need to be installed before beginning the process of syncing the Wii U gamepad:

* libglew-dev
* yasm
* libnl-dev
* build-essential
* SDL
* linux-firmware-nonfree
* software-properties-common
* screen

Additionally, the code that allows the sync to occur requires specific hardware as well:

* Computer with Linux Ubuntu 13.10+ and Linux Kernel 3.13.1 installed
* USB WiFi Adapter with rt2800usb chipset

Once all the necessary software and hardware has been acquired, the next step is to follow the installation instructions for the software patch found at:

<http://libdrc.org/docs/installation.html>

After running into issues building the code provided by Libdrc, additional instructions compiling the hack code and building the Linux kernel patch was found at:

<http://rememberdontsearch.wordpress.com/2014/01/05/libdrc-wiiu-linux-setup-with-rt2800usb/>

Upon following the instructions of the help page, it was realized that the provided code from Libdrc.org contained seveal erros in the code that need to be fixed before precedding. These changes are outlined as follows:

All of the changes to the Libdrc.org code must be saved. All of the instructions in Libdrc.org and the supporting websites was combined into one code that has been constructed to simply run these instructions at once. This application allows one to pair the Wii U controller by simply launching one code and turning the Wii U Gamepad on.

3.2.2.2.            Hardware Implementation

3.2.3.NVIDIA Shield

Android Studio (Preview) Version 0.6.1

3.2.3.1.  Software Development

Use ROS Java to develop Android SDK

 Gives the ability to make ROS nodes on Android devices

Create an app that reads the joystick and touch events from the Nvidia Shield and publishes them on an ROS topic

Inside the ROS\_Cyton\_Gamma package, create a ROS node that converts the joystick commands into robot dynamics

3.2.3.2.            Hardware Implementation

3.2.4. Google Glass

3.2.4.1.            Software Development

3.2.4.2.            Hardware Implementation

3.2.5. Space Navigator

**3.3.   Experimentation**

**4. Risk Assessment**

**4.1.   Software Risk**

**4.2.   Hardware Risk**

**4.3.   Safety Review**

**5. Cost Budget**

1. **6. Conclusion**