HD^2 API Quick Start Guide

1. Introduction

This document is prepared to serve as a quick start guide to get the Quanser HD^2 Haptic Device up and running. It is assumed that you have already read the HD^2 Device User Manual that provides detailed explanation on hardware components, wiring, and specifications of the device.

The following is a listing and a brief description of the supplied controllers, API and test software that can be used to get the device up and running. Detailed discussion on each of these components as well as possible applications of the supplied API's is given in the next section.

- **Driver (*File Name: HD2_Driver*):** This file is saved on the following location in the customized PC that was shipped with the device:

  C:\Quanser\Imperial\Release\Driver

  This is an executable file generated using Quanser's QuaRC software and contains the interface between the PC and the external hardware via the Q8 data acquisition board that is installed in the PC. This allows real-time communication with the device's sensing and actuating elements. Using the Quanser Stream API this data can be monitored and manipulated in user developed applications (C, C++, C#, etc) providing complete control over the device. It seamlessly connects to the device via the DAQ and starts real-time communication between the PC and the external hardware (HD^2 Device). At this point user developed applications (such as the supplied API's which will be discussed below) can connect to this driver, and use its real-time data to perform any desired task such as position control, force control or torque control on the HD^2 Device.

- **End User Test (*File Name: force_position_test_application.exe)*:** This file is saved on the following location in the customized PC that was shipped with the device:

  C:\Quanser\Imperial\Release\End User Test

  This application can be used as a quick test utility to make sure the HD^2 Device is operating normally. It also serves as an example of how applications written in C, C++ or C# can communicate with the “Driver” that was mentioned above to access the device's sensing and actuating elements. This application provides a graphical user interface in which you can monitor the HD^2 Device's end-effector position and the status of the supplied foot pedal. You can also enable/disable the amplifiers, calibrate the device, and apply forces and torques in different freedom degrees using this interface.

- **API Examples (*File Name: API_Examples.sln)*:** This Microsoft Visual Studio solution file is saved on the following location in the customized PC that was shipped with the device:

  C:\Quanser\Imperial\Release\API Examples

  This solution contains examples of programs written in C++ that again use the above-mentioned driver as well as the Quanser Stream API to access the sensing and actuating elements of the HD^2 Device as well as all of its Digital I/O/. These examples can be thought of as a starting point for developing your own customized applications depending on your specific needs. Examples currently included in this solution include a calibration API that can be used to calibrate the HD^2 Device as well as a force/position control API.

2. File/Project Explanations

2.1. Driver

As mentioned earlier, this file seamlessly starts a real-time communication channel between the PC and the HD^2 Device. You can run the file by simply double-clicking on it. This allows custom developed applications to access the data available from this driver that includes encoder measurements, digital channels status, scissor angles, etc. The driver also accepts commands from custom applications which in turn are applied to the device. The data currently available to be received from the driver is a 22 element vector of doubles with the following structure:

- [0:2] :end-effector x, y and z task space position in SI units.
- [3:4] :end-effector roll, pitch task space orientations in SI units.
- [5:6] :left and right pedal status of the supplied foot-pedal (1: pressed)
- [7:11] :end-effector x, y, z, roll, and pitch velocities in SI units.
- [12] :time elapsed since the driver was started in seconds.
- [13] :safety flag showing one of freedom degrees is vibrating with frequency higher than the allowed threshold.
- [14] :safety flag showing that one of the freedom degrees has acquired a velocity higher than the allowed threshold.
- [15:20] :current measured at each motor in Amps.
- [21] :end-effector rotation about its own axis in radians (yaw)

The data currently accepted by the driver is a 17 element vector of doubles with the following structure:

- [0:2] :force to be applied to the end-effector in x, y and z DOF's in SI units.
- [3:4] :torque to be applied to the end-effector in roll, pitch DOF's in SI units.
- [5] :control mode (0: force control, 1: position control)
- [6] :emergency stop flag. Raising this signal from 0 to 1 stops the driver.
- [7:11] :position control stiffness for the 5 freedom degrees in N/m for x, y and z and N.m/rad for roll and pitch.
- [12] :safety enable flag with default value of 1. When this value is 0 if a safety flag is raised, the driver will continue running. When the value is set to 1 if a safety flag raises the driver will stop automatically.
- [13] :vibration safety threshold. This sets the number of vibrations the device is allowed to have in each of its freedom degrees in 1.9 seconds before the vibration safety flag is raised. The default is 40.
- [14:15] :amp enable signal. [1 0] enables the amps while any other configuration keeps them disabled.
- [16] :calibration flag. Raising this flag will reset the encoder counts on the Q8 DAQ.

The vibration detection safety feature monitors the number of sign changes in the velocity of each freedom degree in periods of 1.9 seconds and then resets. There is a threshold that the user can set that corresponds to the number of vibrations allowed before the safety flag is raised (element 13 in driver input list). By default this value is 40 for the HD^2 device meaning if in 1.9 seconds the device vibrates back and forth in any direction more than 40 times, the vibration flag is raised and the driver will be stopped if safety is enabled by element 12 in the driver input list.

The high velocity detection feature monitors the velocity of each freedom degree. The high velocity vibration flag is raised in any of the following conditions:

- The device end-effector acquires a velocity of 1 m/s or more for 0.08 seconds or more in x, y or z directions.
- The device end-effector acquires a velocity of 1.5 rad/s or more for 0.2 seconds or more in the roll or pitch directions.

If the safety enable flag (element 12 in the driver input list) is set to 1, the driver will stop automatically if the high velocity flag is raised.

2.2. End User Test (*force_test_application.exe)*

This project implements a C# application that uses the Quanser Stream API to connect to the driver. The application can issue forces to be applied to each DOF of the HD^2 Device. You can also command the device to hold its current position in space using this application.

The program also receives and displays the current position and orientation of the end-effector. In addition the status of each of the pedals of the supplied foot-pedal is shown. You can enable and disable the amplifier using the respective buttons. You can also calibrate the HD^2 Device using this application. All these properties make this application a quick test tool to make sure that the device is operating normally. It is assumed that the HD^2 Device is wired to the PC as instructed in the user manual when running this program. The vibration threshold value can also be changed from this utility in addition to being able to completely disable the safety feature by un-checking the corresponding box.

In order to calibrate the device, place the end-effector in the calibration jig and click on the Calibrate button. Remember to always calibrate the device at the beginning of use to ensure that encoder readings you start with are reset. You can click on the Emergency Stop button to stop the driver and halt the communication with the external hardware if the device is not acting normally. The supplied external emergency stop can also be used in these situations to disable the amplifiers.

It is highly recommended to always hold the end-effector with one hand when applying forces and torques to the device to avoid possible damage as a result of sudden movements.

Remember that the driver must be running prior to starting this application.

2.3. Example API's (*API Examples.sln)*

This solution contains C++ based applications that can be used as examples of how the Stream API can be used to communicate with the driver and hence with the HD^2 Device. Currently there is a Calibration project included in this solution that can be used to reset the encoders of the HD^2 Device. Note that the driver must be running prior to running the examples included in this solution. Simply run this program from Visual Studio. A console will open that asks you to place the end-effector in the calibration jig and press Enter. Once you do this, a message will be shown saying that the device has been calibrated successfully.

This application communicates the same set of data that was mentioned in section 2.1 with the driver. For this specific application which is only used to calibrate the device, the calibration flag is simply set to 1 in the code and sent to the driver. In more advance

applications such as position control API's other elements of vectors described in section 2.1 are also used. This can be seen in the other supplied project (position_control) where the same set of data are communicated and parsed with the driver. By setting control mode to 1, the system goes into position control mode and holds its position in space upon application start.