Notes on Installing and configuring MDX platform software

**Overview**

The platform control software is a directory named runtime and typically runs on a Raspberry Pi with the motion simulation application (coaster etc) running in a Windows PC. Additional PCs are required for two or more synchronized Vr experiences.

The software requires python 3 (tested with Python 3.8 and 3.9)

Platform controller requires:

PyQt5, numpy, matplotlib

Space\_coaster requires:

pywin32 (for import win32gui)

**Install Raspberry Pi files**

Hardware requirements: Raspberry pi model 3B+ or model 4 2GB. (TK info on hardware controls)

Software requirements: Mdx runtime software, Python 3, PyQt5 python GUI library

Change password to Concorde

copy start\_runtime.sh bash script to Desktop:

The startup bash script is as follows:

#!/bin/bash

cd /home/pi/runtime

/usr/bin/python /home/pi/runtime/platform\_controller.py

Make this script executable as using chmod +x or using the GUI file properties

add following line to the end of: pi.config/lxsession/LXDE-pi/autostart:

@/home/pi/Desktop/start\_runtime.sh

Copy the runtime files from here (TK) to /home/pi/runtime

Update apt-get

$ sudo apt-get update

$ sudo apt-get upgrade

Install the following python packages:

sudo apt-get install python3-pyqt5

if using ufw firewall do sudo ufw allow 22,

sudo ufw allow 1000, ???? proto udp

if needed: sudo apt-get install ufw

sudo ufw enable

Add static IP address by adding the following to the end of /etc/dhcpcd.conf

(note the static address is not used if there is a DHCP server on 192.168.0.1)

# define static profile

profile static\_eth0

static ip\_address=192.168.0.2

static routers=192.168.0.1

static domain\_name\_servers=192.168.0.1

# fallback to static profile on eth0

interface eth0

fallback static\_eth0

TODO enable wifi to motion platform ssid ?

Disable wifi and bluetooth by adding these lines to the endof: /boot/config.txt

# turn wifi and bluetooth off

dtoverlay=pi3-disable-wifi

dtoverlay=pi3-disable-bt

for debug and maintenance

install spacefm

install

**Install PC files**

Hardware requirements: The target PC must be suitable for the Oculus Rift runtime software (see Oculus website for details)

Software requirements:

Sim application (NoLImts2, Space coaster etc)

Python 2 or 3 with PyQt5 GUI library

python 2.7: pip install python-qt5

python 3: pip3 install pyqt5

pip3 install pyserial

pip3 install numpy

pip3 install scipy

pip3 install matplotlib

use settings troubleshooter to enable firewall for python

MDX support software: Encoder server

The batch file used requires Windows 10 Enterprise although other variants of Windows 10 can be used with some manual configuration (see below)

If you want to disable auto update of windows, disable the Windows Update service in services.msc

The preferred configuration provides a headless startup – the system automatically logs in and starts all required software and services. Each PC uses a unique login user name to determine the correct license files. Names are in the form: MdxPlatform% where % is an integer starting from 1.

Use netplwiz to untick “users must enter a user name and password…”

**Startup process**

Currently the system can support:

Single NL2 coaster on PC with controller on Raspberry Pi

Single space coaster and controller on PC when started with dongle

Proposed options that can be supported with the new software:

The new software will have two PCs running the Vr rides and a Raspberry P running the control software. I need to decide the mechanism for starting up and synching the software on these machines.

The simplest thing to implement is to have copies of a text file on the three machines, however this may not be easy to modify at an event and problematic if the files are not consistent.

I am considering to have a simple gui config screen at Pi startup offering selection of the number of ‘riders’ (one or two) and the ride to use (i.e. NoLimits or space coaster).

If nothing selected after 5 seconds, preselected defaults in a text file on the Pi will be used.

The PC(s) will be told via network message to start the appropriate ride.

Once selected, these choices can only be changed by rebooting the pi and the PCs If NoLimits was chosen, the park can be selected any time system is disabled (as now).

Initially the only available options will be one or two space coasters.

Is that workable. Is there a simpler option that would be suitable?

Platform controller on pi at startup

Select either one or two simultaneous users

Select one of the following rides:

NoLimits coaster (if selected, parks can be chosen as now)

Space coaster

If nothing selected after 5 seconds, preselected defaults in a text file on the Pi will be selected

Once selected, these can only be changed by rebooting the pi and the PCs

Initiially the only available options will be one or two space coasters.

The proposed implementation will have the PCs at bootup witing for a network message from the pi controller informing the sim to run (nl2 or space coaster) .

Pi