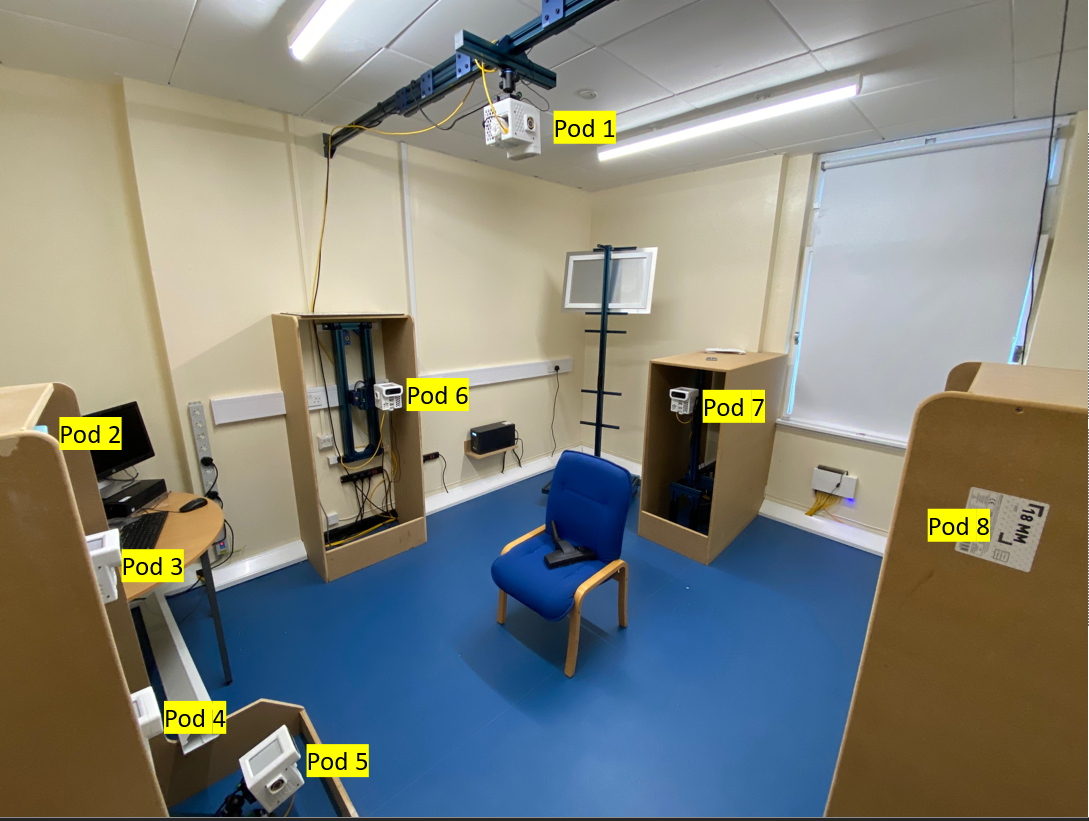
3D Telemedicine Rig First-Run Instructions

Updated: 1/26/2022

These instructions are for setting up a “production” rig, not a development rig. No development tools or libraries are installed as part of these steps.

# Physical Setups

## Nanos (Kinect PODs)

1. Pods should be connected to rigid structures and protected so they cannot move or be easily jostled or bumped. Any movement of a pod requires a re-calibration of the system.
2. Position all of the pods in their desired locations. Standard configuration is to have POD-1 on the ceiling facing down, POD-2,3,4, and 5 at the patient’s front, and then 6, 7, and 8 around the room. sh
3. Connect a power cord to the back of the pod enclosure and connect a Cat-6 ethernet cable to the RJ-45 plug on the bottom of the pod. This cable should connect to the 9-port switch. It may be helpful to label which port is used for which pod, or follow a convention, such that Pod-1 connects to port 1, etc.
4. ~~Connect the sync cables in a daisy-chain fashion starting at POD 1. The pod ordering is not required to be sequential, but the connection of sync cables must be in a daisy chain that starts at POD 1 and goes from one pod to the next. For example, valid sync cable connections can be 1-2-3-4-5-6-7-8, 1-8-7-6-5-4-3-2, 1-7-6-5-4-3-2-8, 1-6-7-8-5-4-3-2, etc. Again, POD-1 should have a sync cable leaving the sync-out (left plug when looking at the rear of the pod) port, but nothing connected to the sync-in (right plug when looking at the rear of the pod) port.~~
5. Connect the sync cables in a tree fashion originating at POD-1. In other words, POD-1 should have a 3.5mm sync cable coming out of the port labeled “sync out” that feeds into the 8-port RCA amplifier. All other pods should have a 3.5mm syncssh cable connected **only** to the “sync in” port, that is also connected to the RCA amplifier.

# Software Setup

## Nanos (Kinect PODs)

Nanos should be imaged with the latest available Jetson Nano Ubuntu linux image. We should have a copy of the latest that should get you 90% of the way there. If not, follow the steps under “bare ISO image”.

### Bare ISO Image

1. Install the Nvidia Jetson Nano Ubuntu image, this should include OpenCV and CUDA
   1. The primary user should be peabody and have a home directory of /home/peabody
2. Copy the root folder in the git repo [https://dev.azure.com/msrp/\_git/PeabodyMain?path=/Scripts/NewPodSetup to /home/peabody/](https://dev.azure.com/msrp/_git/PeabodyMain?path=/Scripts/NewPodSetup%20to%20/home/peabody/) on the nano.
3. Copy <https://dev.azure.com/msrp/_git/PeabodyMain?path=/Scripts/NewPodSetup/configureNewPodImage.sh> to /home/peabody and execute it.

#### NTP Client

On Fusion: <https://computingforgeeks.com/how-to-configure-ntp-server-in-windows-server/>

Edit /etc/ntp.conf and remove all the pool entries, and add:

server 192.168.101.250

sudo ntpdate 192.168.101.250

You should see a message similar to:

22 Jan 10:39:21 ntpdate[8659]: adjust time server 192.168.101.250 offset 0.002539 sec

#### Set up 3DTM Binaries

1. Copy the latest AzureKinectNanoToFusion, AKLauncherDaemon, and K4ARecorder binaries to /usr/local/bin and make sure they are executable

chmod a+x /usr/local/bin/AKLauncherDaemon  
chmod a+x /usr/local/bin/AzureKinectNanoToFusion

chmod a+x /usr/local/bin/K4ARecorder

#### Set up SSH config to handle backup Fusion servers

To allow the PODs to communicate with a backup Fusion workstation, we need to ensure that SSH is allowed to access machines associated with hostname “peabodycontrolpanel” that may have a different key. We do this by

1. Creating file .ssh/config if it does not exist
2. Adding the lines below to the file

Host peabodycontrolpanel

StrictHostKeyChecking no

### Imaging or Re-Imaging an SD Card with a backup image

### Installing a Spare Pod/Replacing an Existing

If you’re installing a spare pod (adding an additional or replacing a non-responsive), it should show up as POD-11 if it was imaged after 9/18/2021. You should be able to connect it into the network, and SSH to it at 192.168.101.11. Then perform the following:

1. If ~/RenamePod.sh exists, just run ./RenamePod [PODNUMBER] to rename the pod. Note, this is just the number, not the name. Put 10 here, not POD-10. Skip the rest of these steps.
2. Otherwise, Modify /etc/network/interfaces.d/eth0 and set the IP address to the proper IP for the pod number this should be (e.g. if this is replacing pod 4, set it to 192.168.101.4, if this is a new pod #9, set it to 192.168.101.9)
3. Modify /etc/hosts and make sure it contains the following:

|  |
| --- |
| 127.0.0.1 localhost  127.0.0.1 POD-10  192.168.101.250 ControlPanel  192.168.101.1 POD-1  192.168.101.2 POD-2  192.168.101.3 POD-3  192.168.101.4 POD-4  192.168.101.5 POD-5  192.168.101.6 POD-6  192.168.101.7 POD-7  192.168.101.8 POD-8  192.168.101.9 POD-9  192.168.101.10 POD-10 |

And make sure that the second line has the name of what this pod will be (POD-4, POD-9, etc…)

1. Modify /etc/hostname and change the hostname to the correct name
2. Reboot the pod. It should come up at the new IP address you specified in step 1. Make sure you update the software on this new pod, as it’s probably out of date. If it was imaged after 9/18/2021, you should be able to just update from the control panel. If it was before, see below:

#### Updating from a build prior to 9/18/2021

If you do have to update from a build prior you may need to make more drastic changes. Copy the following files from another pod in the system:

sudo scp peabody@pod-1:/usr/local/lib/libz\* /usr/local/lib/

scp peabody@pod-1:/usr/local/bin/AzureKinectNanoToFusion /usr/local/bin/

scp peabody@pod-1:/usr/local/bin/AKLauncherDaemon /home/peabody/staging/

scp [peabody@pod-1:/home/peabody/RunK4A.sh](mailto:peabody@pod-1:/home/peabody/RunK4A.sh) /home/peabody/

such as copying over libzmq, opencv 4.5.4 libraries, RunK4A.sh scripts, and binaries from another pod.

#### Other Setup

~~Create /mnt/RamDisk so calibration has a location where to mount the ramdisk for mkv capture~~

Ssh into fusion from the pod once to make sure it has accepted the server’s ID

ssh 3dtm@peabodycontrolpanel

(you might have to wait until the fusion machine is set up if you haven’t done that already)

### Test Kinects and Download Factory Calibration Data

1. Reset the Kinect sensor

AzureKinectFirmwareTool -r

1. Verify you have the latest Kinect firmware.

AzureKinectFirmwareTool -q

You should be running at least   
 RGB Firmware: 1.6.110  
 Depth firmware: 1.6.79

If you are not, download the latest from the Azure Kinect DK site and install using

AzureKinectFirmwareTool -u nameoffirmware.bin

1. ~~Run AzureKinectNanoToFusion in factory calibration write mode to get the factory intrinsics for the connected sensor~~

~~AzureKinectNanoToFusion /home/peabody/K4AToFusion/3DTelemedicine.cfg 5~~

~~This will not say much (output is logged to /var/log/K4AToFusion/AzureKinectNanoToFusion.log) but you will know it was successful if you now have colorConfig00.json and depthConfig00.json in your /home/peabody/K4AToFusion folder  
Throwing a segmentation fault and exiting is actually~~ *~~expected~~* ~~operation~~

1. Set up logrotate to rotate the logfiles in /var/log/K4AToFusion
   1. Create the file /etc/logrotate.d/K4AToFusion with the following contents:

/var/log/K4AToFuion/\*.log

{

rotate 7

daily

missingok

notifempty

delaycompress

compress

create 660 peabody users

}

1. Make the reboot command accessible by non-root:  
   sudo chmod u+s /sbin/shutdown
2. Reboot

### Setup USB drives for K4ARecorder

1. Create directory:

sudo mkdir /mnt/USB

1. Check USB drives identifiers

ls /dev/disk/by-id/

1. Edit /etc/fstab file so we mount the drives correctly, avoiding confusion with other devices (LED displays, ring lights) and preventing failures (option ‘nofail’) when the drive is not inserted before booting the POD. Example:

/dev/disk/by-id/usb-Lexar\_USB\_Flash\_Drive\_04ALK7BFP8JDGJS4-0:0-part1 /mnt/USB ext4 user,noauto,nofail,rw,exec 0 0

1. Mount drive and change permissions to write MKVs:

mount /mnt/USB

sudo chmod 777 /mnt/USB

**IMPORTANT: mount the drive \*first\* and then change permissions** to the folder. Otherwise it does not work and K4ARecorder will not be able to save files into the folder.

1. Reboot

## Fusion PC

### System Setup

1. Connect OneDrive to the [telemedicine3d@outlook.com](mailto:telemedicine3d@outlook.com) account (find user/password at 1Passord account).
2. Download PowerShell script **NewComputer\_3DTM\_Environment\_Setup.ps1** from *OneDrive\Tools\scripts* folder
3. Open PowerShell as administrator and run commands:  
   > set-executionpolicy -executionpolicy Bypass   
   > enable-psremoting -force
4. Import “./NewComputer\_3DTM\_Environment\_Setup.ps1”
5. Run:

> SetupFusion

This script will download the binaries from WSUS, install the required dependencies, add the required directories and environment variables, run the services, set the firewall rules, and transfer the latest binaries to the nanos.

1. SSH into the pods and restart AKLD service:

> sudo service aklauncherdaemon stop

> sudo service aklauncherdaemon start

1. Follow the instructions for implementing log file rotation.
2. Run:

> SetupCalibrationSoftware

This script will download the binaries from WSUS, install the required dependencies, add the required directories and environment variables, run the services, and set the firewall rules.

1. Install OpenSSH server on Fusion: <https://winscp.net/eng/docs/guide_windows_openssh_server#win10>
2. Copy the contents of the ~/.ssh/id\_rsa.pub key from all of the pods (if these are images, they may all be the same, and you'll only need one) to C:\ProgramData\ssh\administrators\_authorized\_keys
3. Change the permissions on administrators\_authorized\_keys:

$acl = Get-Acl C:\ProgramData\ssh\administrators\_authorized\_keys  
$acl.SetAccessRuleProtection($true, $false)  
$administratorsRule = New-Object system.security.accesscontrol.filesystemaccessrule("Administrators","FullControl","Allow")  
$systemRule = New-Object system.security.accesscontrol.filesystemaccessrule("SYSTEM","FullControl","Allow")  
$acl.SetAccessRule($administratorsRule)  
$acl.SetAccessRule($systemRule)  
$acl | Set-Acl

From <[*https://www.concurrency.com/blog/may-2019/key-based-authentication-for-openssh-on-windows?msclkid=4c88ea72ac7d11ec9c16358222a43e3f*](https://www.concurrency.com/blog/may-2019/key-based-authentication-for-openssh-on-windows?msclkid=4c88ea72ac7d11ec9c16358222a43e3f)>

1. (Re)start the OpenSSH Server service (and set it to automatic start)
2. Test the connection by ssh'ing into fusion from any pod:  
   $> ssh 3dtm@peabodycontrolpanel

If you're **not** prompted for a password, it's working correctly. You may have to accept the host key the first time. If you do, make sure you do this on **all pods** so they don't prompt and freeze during a calibration run

### Viewer PC

Currently, Viewer runs on the Fusion PC.

1. After running the System Setup steps, run:

> SetupViewer

### SSH Setup

If you will be needing to push new bits to the nanos often, SSH setup may be useful

1. Generate a private/public keypair by opening powershell and running

Ssh-keygen

You can leave it without a passphrase, and it can make the id\_rsa files in C:\users\[username]\.ssh which is the default

1. Copy the contents of the id\_rsa.pub file into /home/peabody/.ssh/authorized\_keys on each pod
2. On the nanos, give normal users the ability to reboot by issuing the command

sudo chmod u+s /sbin/shutdown

1. Edit C:\Windows\System32\drivers\etc\hosts file to include the hostnames:

192.168.101.1 POD-1

192.168.101.2 POD-2

192.168.101.3 POD-3

192.168.101.4 POD-4

192.168.101.5 POD-5

192.168.101.6 POD-6

192.168.101.7 POD-7

192.168.101.8 POD-8

192.168.101.9 POD-9

192.168.101.10 POD-10

1. On the pods, change folder permission so you can copy the binaries to /usr/local/bin:

sudo chown peabody /usr/local/bin

sudo chmod 744 /usr/local/bin

1. Now you should be able to use the UploadBinariesToNano to push AKLauncherDaemon and AzureKinectNanoToFusion (and any other files) to all 10 pods with one command.

## Render PC

### System Setup

1. Repeat steps at **Fusion PC - System Setup** replacing *SetupFusion* command by:

> SetupRender

## Control Panel PC/Tablet

### System Setup

1. Repeat steps at **Fusion PC - System Setup** replacing *SetupFusion* command by:

> SetupControlPanel

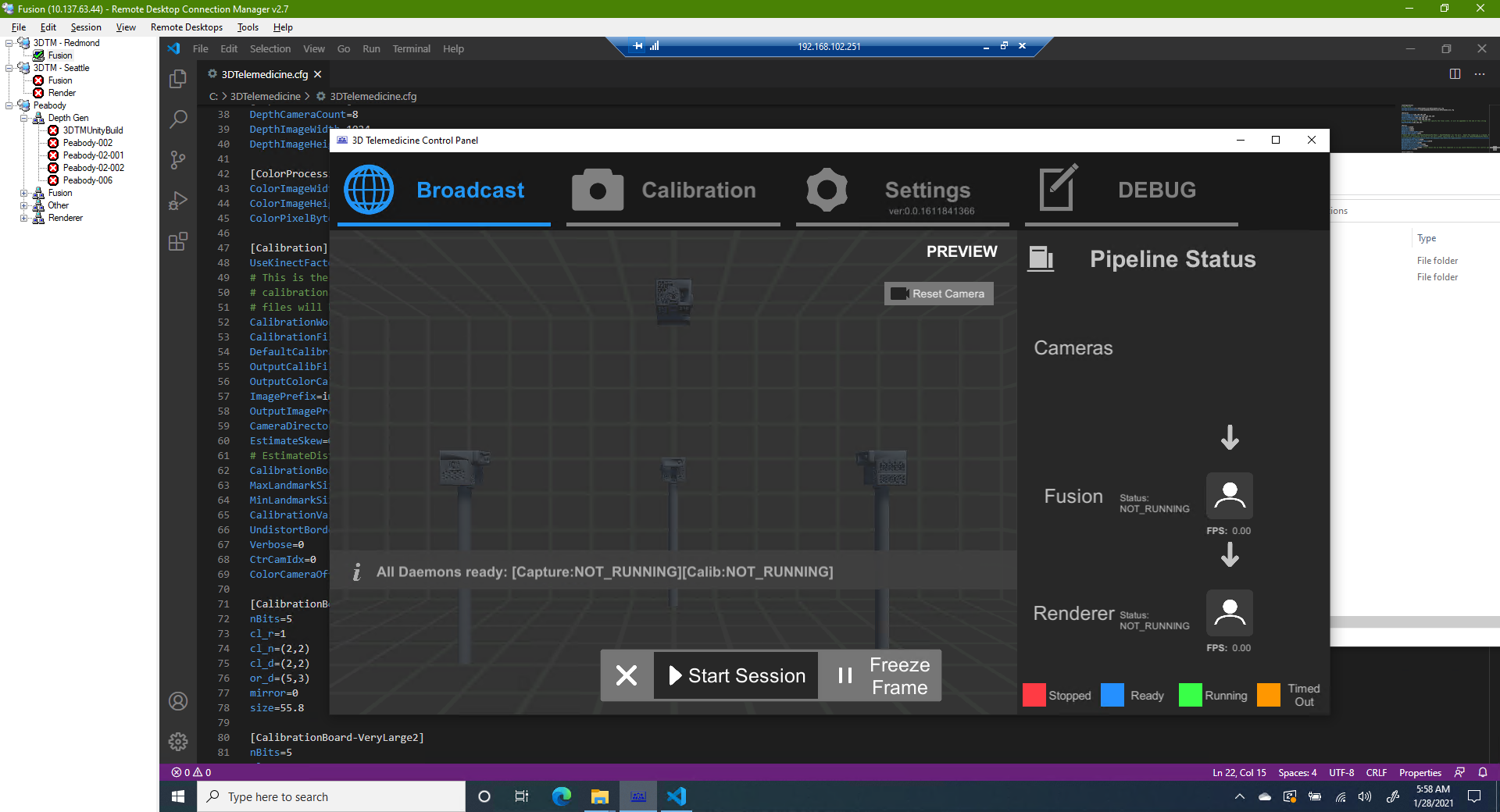
### Basic Configuration

Control panel requires a basic configuration file to operate. This file should define the network addresses of the various components. It defaults to C:\3DTelemedicine\3dTelemedicine.cfg

# System Setup

Start the control panel on the control panel PC

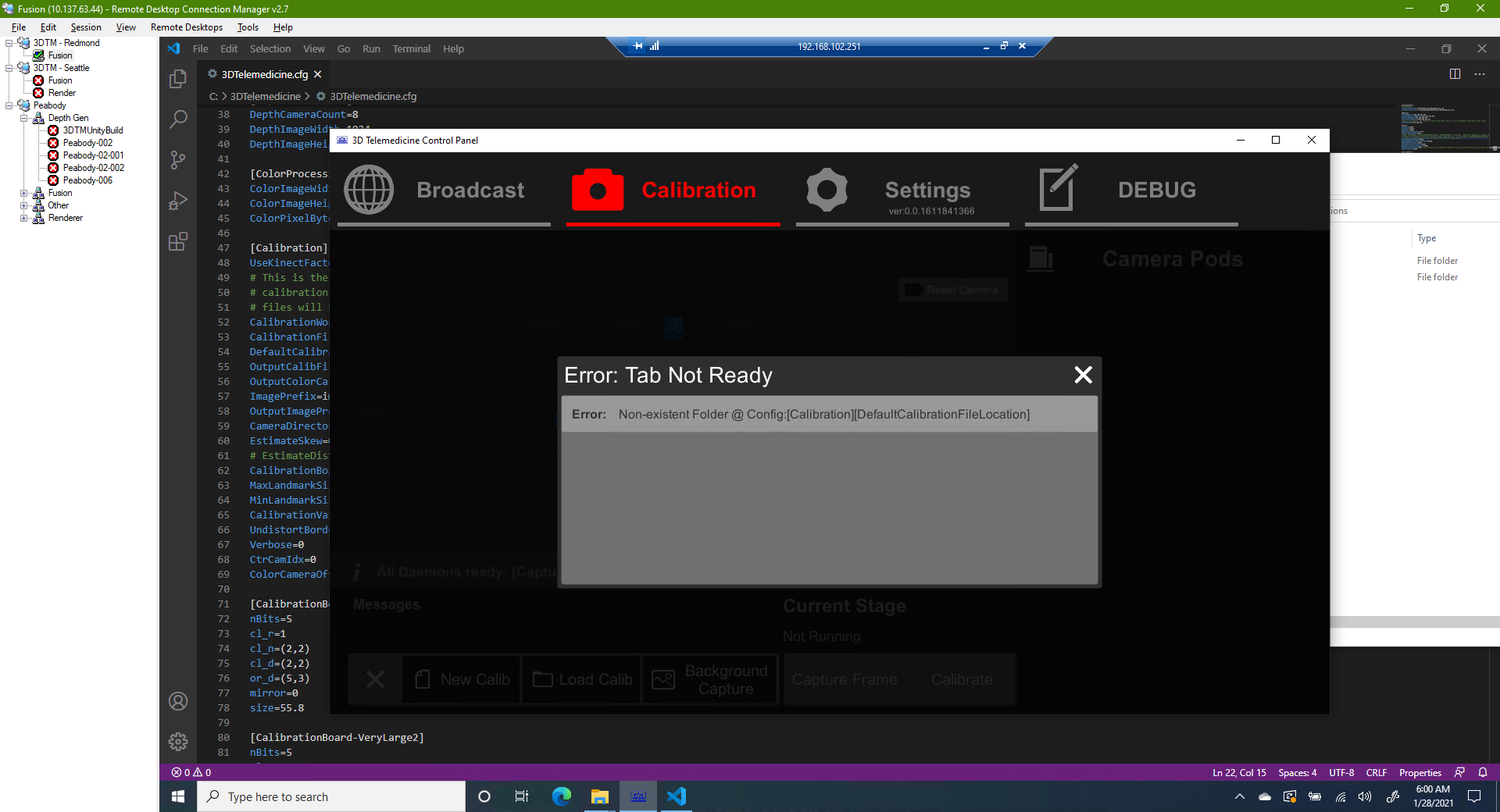
1. Check for all daemons working

You should see this screen:  


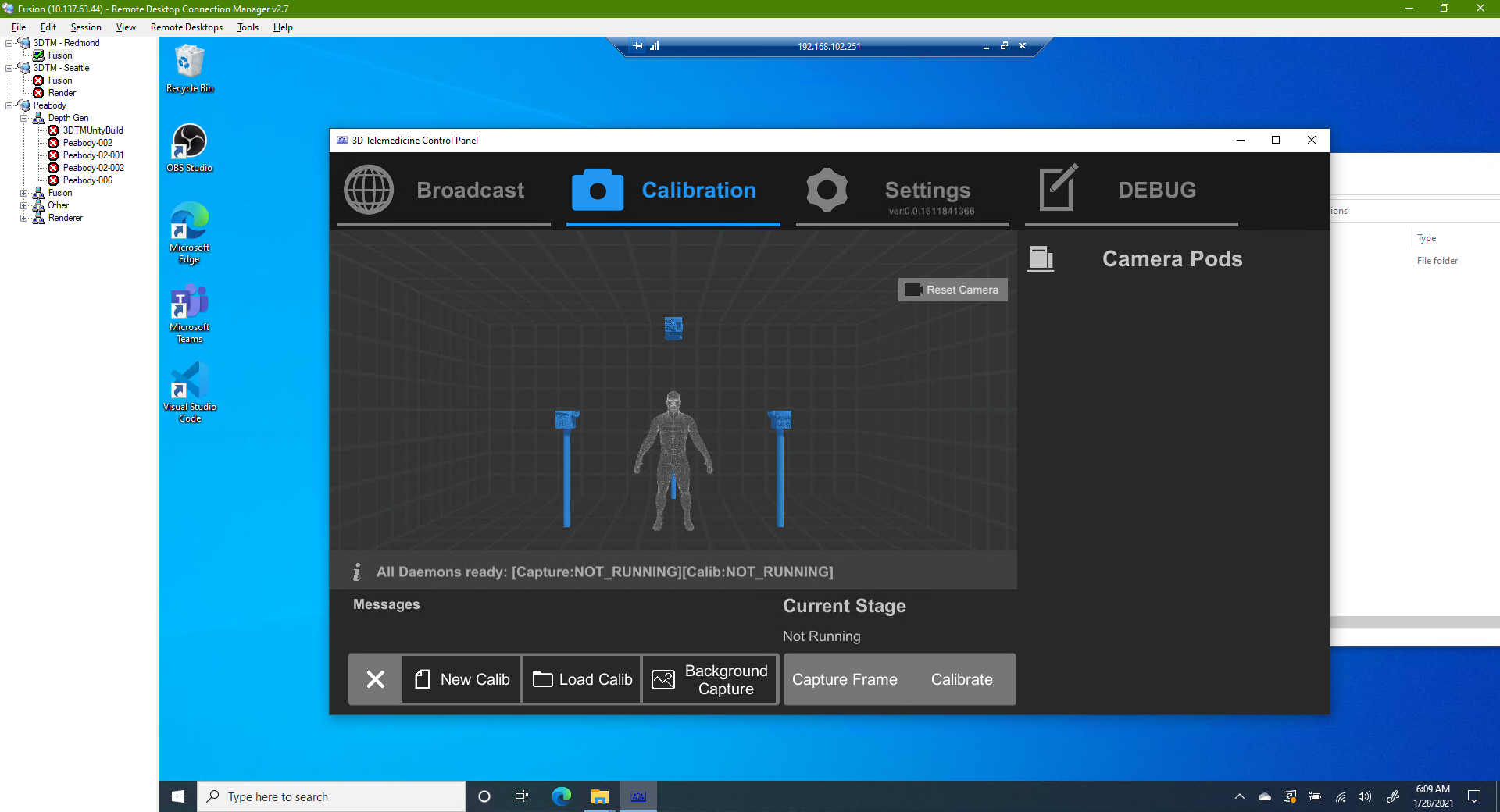
Note the text across the middle of the visual area that says “All Daemons ready”. If you don’t see that, there is a problem with Kinect Nano Communicator Service, your firewall, your network, or one of the pods. It will indicate which pod is not transmitting.

The Status: text next to Fusion and Renderer should also say “NOT\_RUNNING”.

1. Click the Calibration tab. A window should pop up letting you know if the system is not ready for calibration.

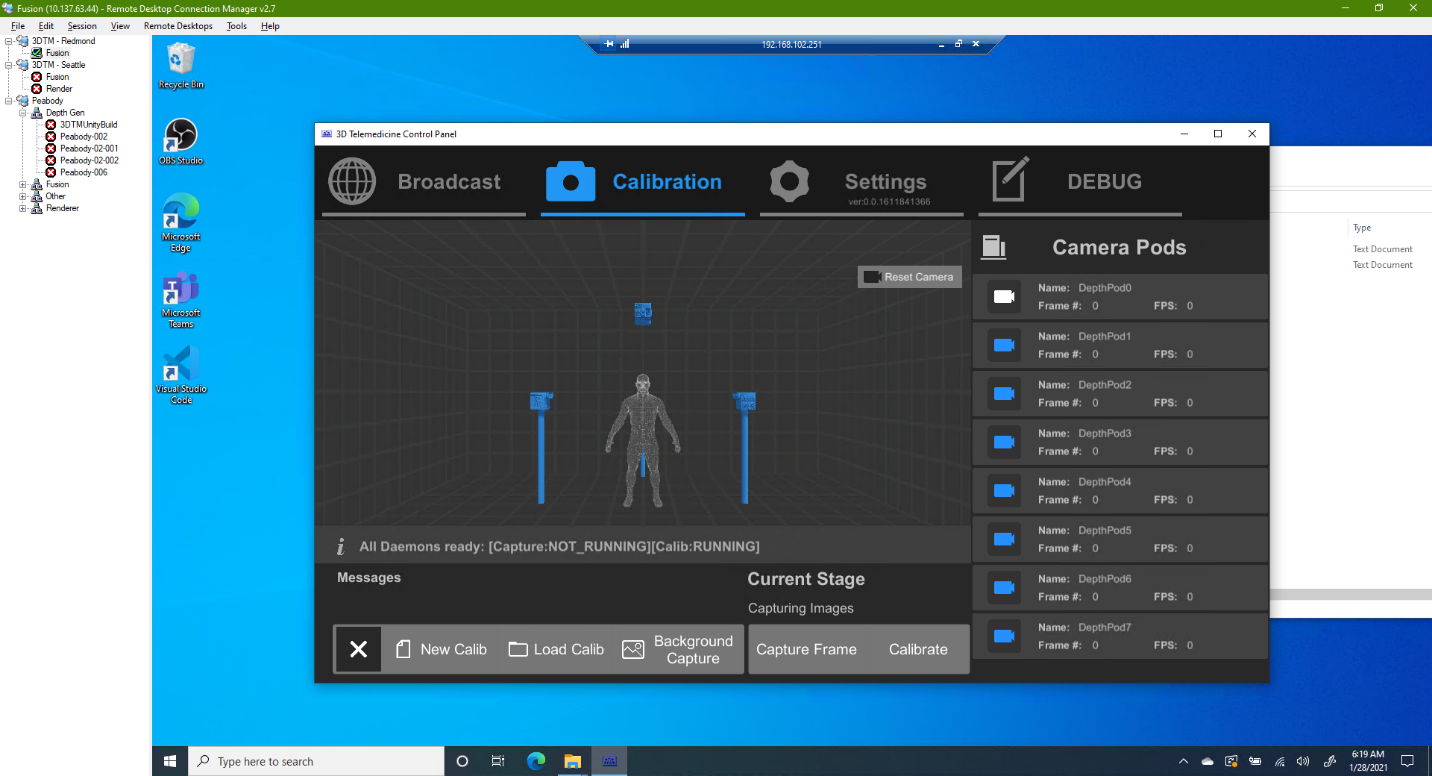
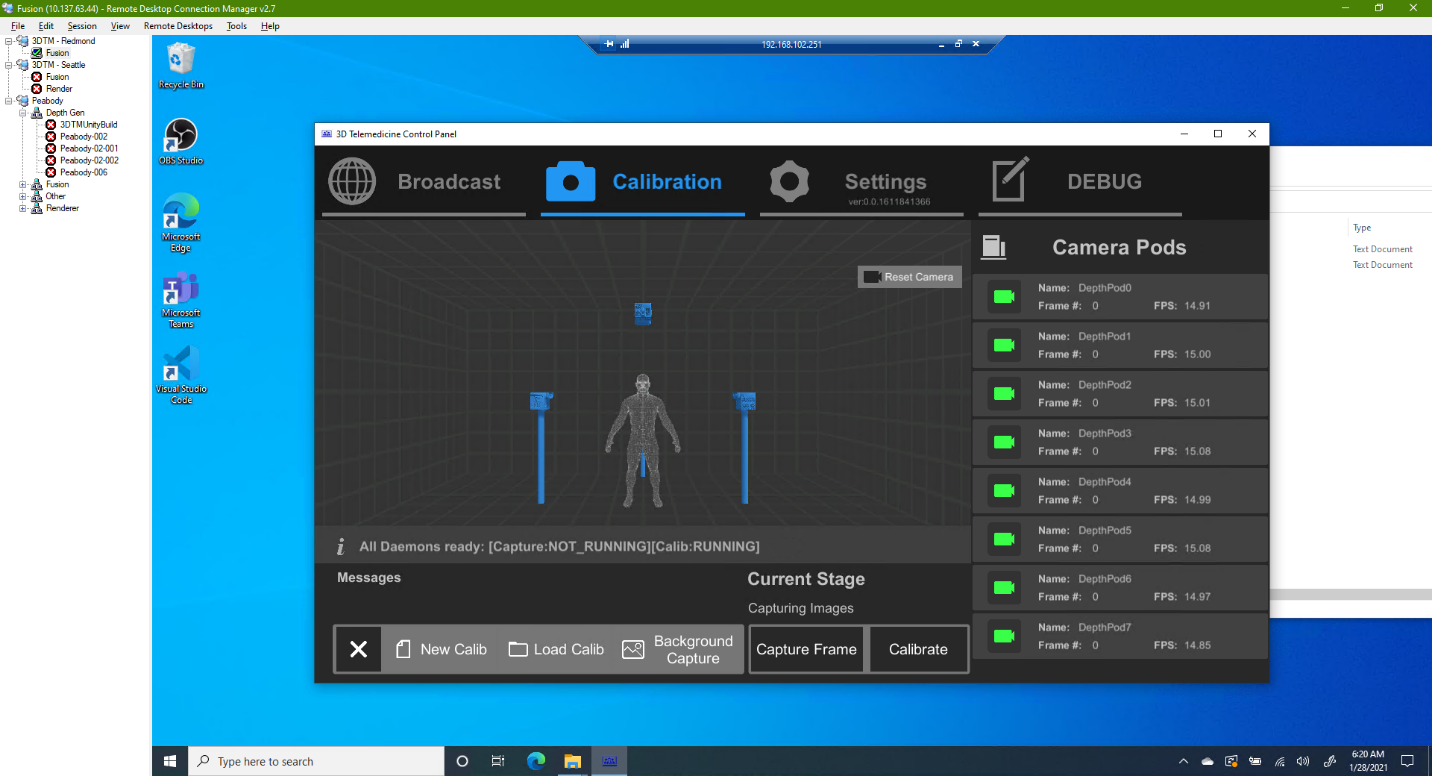


Correct operation will look like:



1. Go to Debug and click “Request AK Factory Calibrations”
   1. Check C:\3DTelemedicine\DATA\Calibration and verify that there is now a K4AFactoryCalibs folder, and it contains one folder per pod, cam0-x, and each folder contains two files, colorConfig00.txt and depthConfig00.txt
   2. If the files do not exist, repeat step 2 under Software Setup > Nanos > Test Kinects and Download Factory Calibration Data
2. Go to Debug and click “Transmit System Configuration” to put the full 3DTelemedicine.cfg file to all systems.
   1. Note that this will change the status of Fusion and Renderer to Undefined or TIMEDOUT while the 3DTMLauncherService reloads. Wait until it changes back to NOT\_RUNNING

# System Calibration

1. Click on the Calibration tab on the control panel, and click the “New Calib” button
   1. You should see a list of all of the pods show up on the right pane, with white icons
   2. Pods 1-n will turn blue, while pod 1 stays white
   3. Pod 1 will then turn blue, and then all pods will turn green and the FPS numbers will start changing, showing that the PODS are online 
   4. Complete Calibration

A quality calibration can be achieved typically with 12 images. Position the checkerboard in four different locations in the room (typically corners) where it can be seen by more that one camera at a time. Also position the board at three distinct heights. Typically this is 1 – close to the ground, facing upwards at a 30 degree angle, chest-height facing perpendicular to the floor, and head-height facing downward at a 30 degree angle. After capturing the 12 images, press the Calibrate button and the system will download images, and then attempt to calibrate. Success or failure will be displayed in the messages section of the control panel.

* 1. Once calibration is successful, copy the newest folder in the C:\3DTelemedicine\DATA\AllCalibrations\ folder and rename the copy C:\3DTelemedicine\DATA\AllCalibrations\Default
  2. Remove any non-fixed objects from the rig, and press the “Background capture” button to capture the background data. This process will complete and shut down automatically when finished.

# Stream Test

You can now switch tabs to Broadcast.

1. Verify that the daemon status bar along the bottom of the 3D view still says “All Daemons ready” and that the status messages next to Fusion and Renderer in the right pane say “NOT\_RUNNING”. If they say “UNDEFINED” or “TIMEDOUT” fusion and renderer are not ready. This can be for a few reasons:
   1. The 3DTM Launcher Service is restarting after receiving a new config file. Wait a few minutes to see if it changes to NOT\_RUNNING
   2. The 3DTM service has crashed. Log into the machine and verify the service is running
   3. There is a network issue (firewall or bad 3DTelemedicine.cfg file) on one of the systems, and it cannot talk to the control panel

Many times, simply restarting the fusion or render PC will clear up this issue. But do not proceed until both systems report back NOT\_RUNNING.

1. Click “Start Session” to begin a session.

You should see 8 cameras appear on the right with white icons. Again, cameras 2-8 will turn blue, then camera 1, then they will all turn green. Next the fusion icon should turn green, and then the renderer icon. At this point you will see a 2D camera view in the bottom left corner, and the 3D model should appear in the center of the 3D screen. Pan by holding down the middle mouse button, rotate by holding down the right mouse button. Zoom by scrolling the mouse wheel. Touch controls also work here.

# Color Correction

To perform color calibration, you need to capture an image of the standard color calibration chart (we typically use the X-Rite ColorChecker classic: <https://www.amazon.com/X-Rite-ColorChecker-MSCCC-Photography-Filmmaking/dp/B000JLO31C/ref=sr_1_6?dchild=1&keywords=color+calibration+board&qid=1622585574&sr=8-6>) directly facing each camera. Currently, the easiest way to do this is to start the system in [Calibration mode](#_System_Calibration). Once the system is ready to capture images for calibration, hold the calibration board directly facing camera 1 and capture an image. Proceed to camera 2, and so on, until you have an image for each camera. Then press “calibrate” to cause the system to download the images. Calibration will fail because it won’t detect a checkerboard, but the images will now be stored in C:\3DTelemedicine\Data\Calibration\cam1 … camN.

Until we have developed an in-house solution, the current method to calculate calibration look-up-tables (LUTs) is to use DaVanci Resolve (<https://www.blackmagicdesign.com/products/davinciresolve/>).

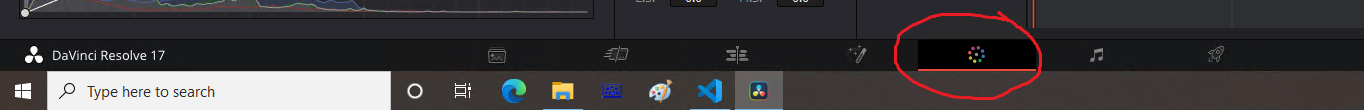
1. For each camera, find the image that has the clearest, most direct view of the board, and save a copy of the image as camX.png. (Davinci cannot read bmps, so it needs to be converted to png).
2. Start Davinci Resolve and open a new project
3. Drag the cam0…camN.png images into the Media Pool (top left section of the Davinci screen)

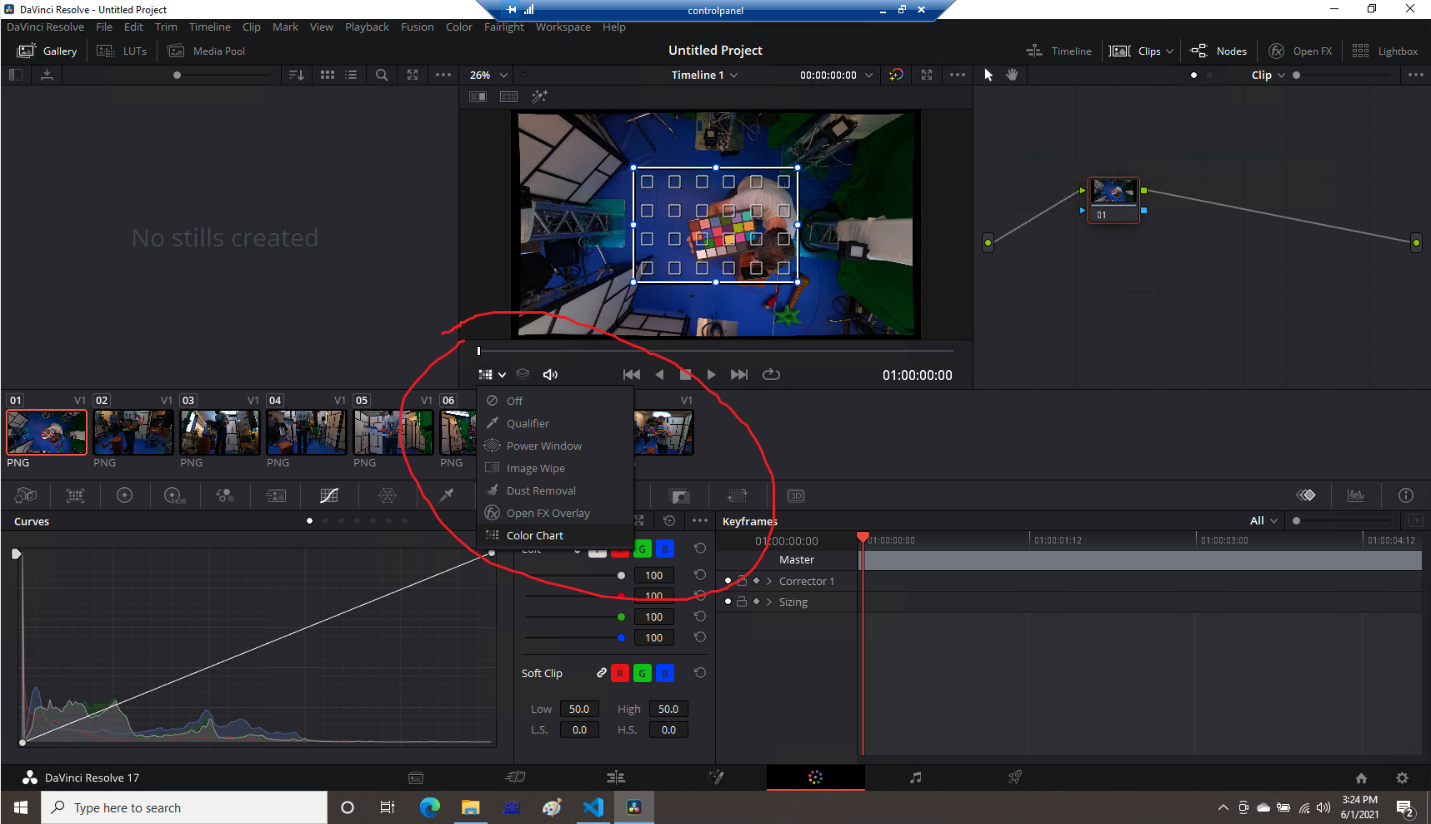
A screenshot of a computer

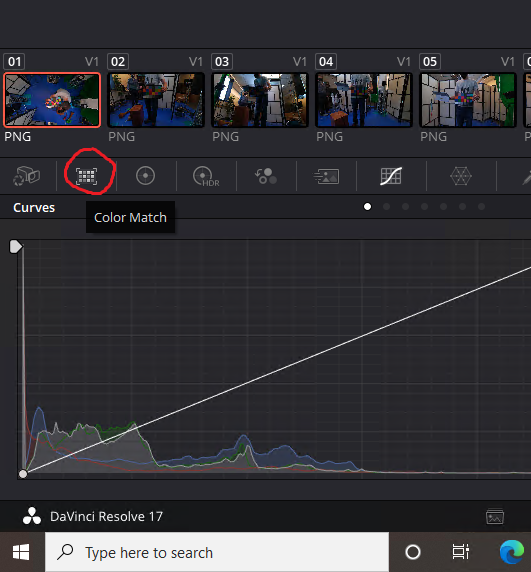
Description automatically generated with medium confidence

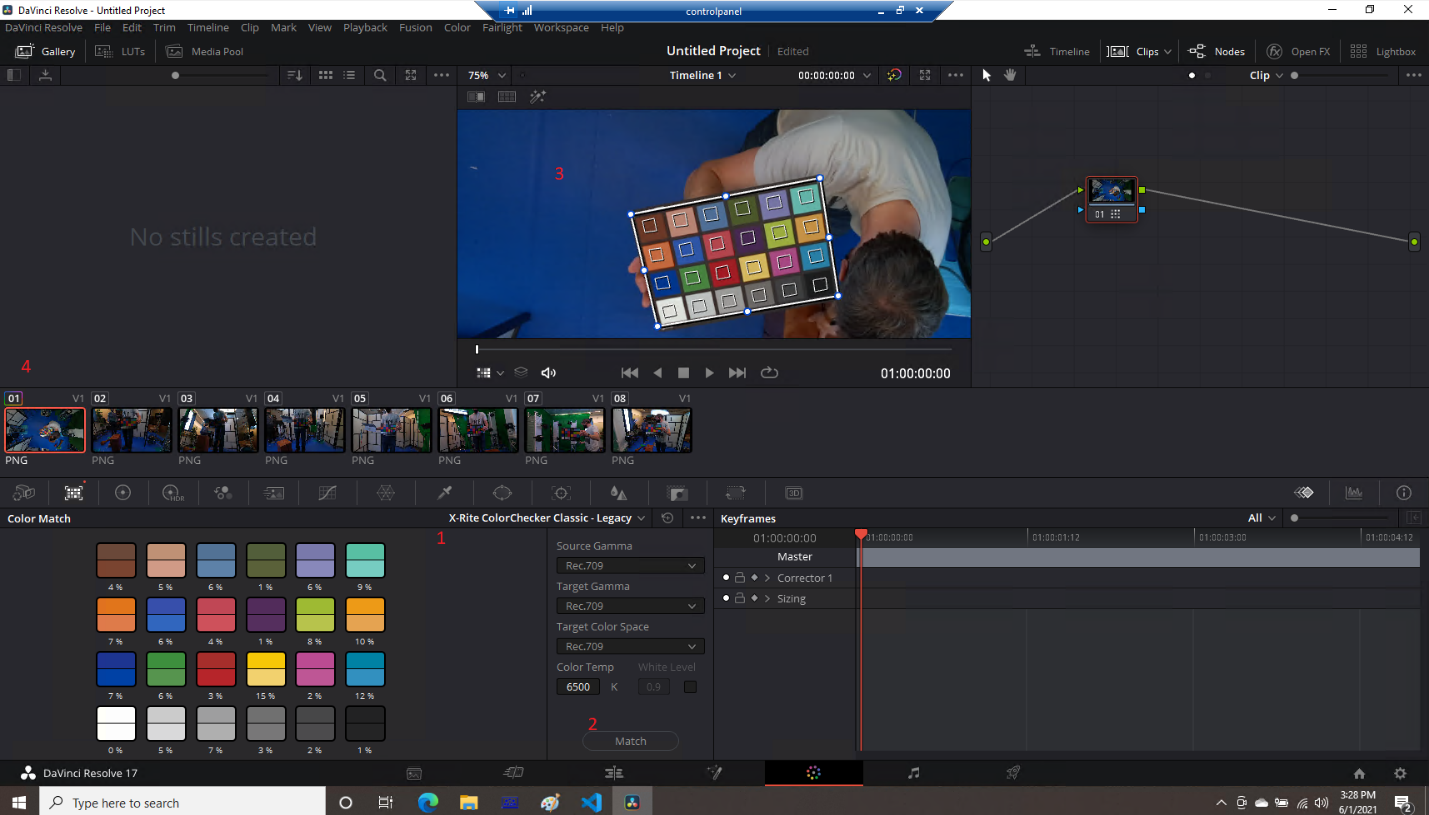
1. Once you have all of the images loaded into the media pool, highlight them all and drag them into the timeline (bottom half of the screen)  
   A picture containing text, electronics, indoor, display

   Description automatically generated
2. Now click on the “Color” button at the bottom of the screen:

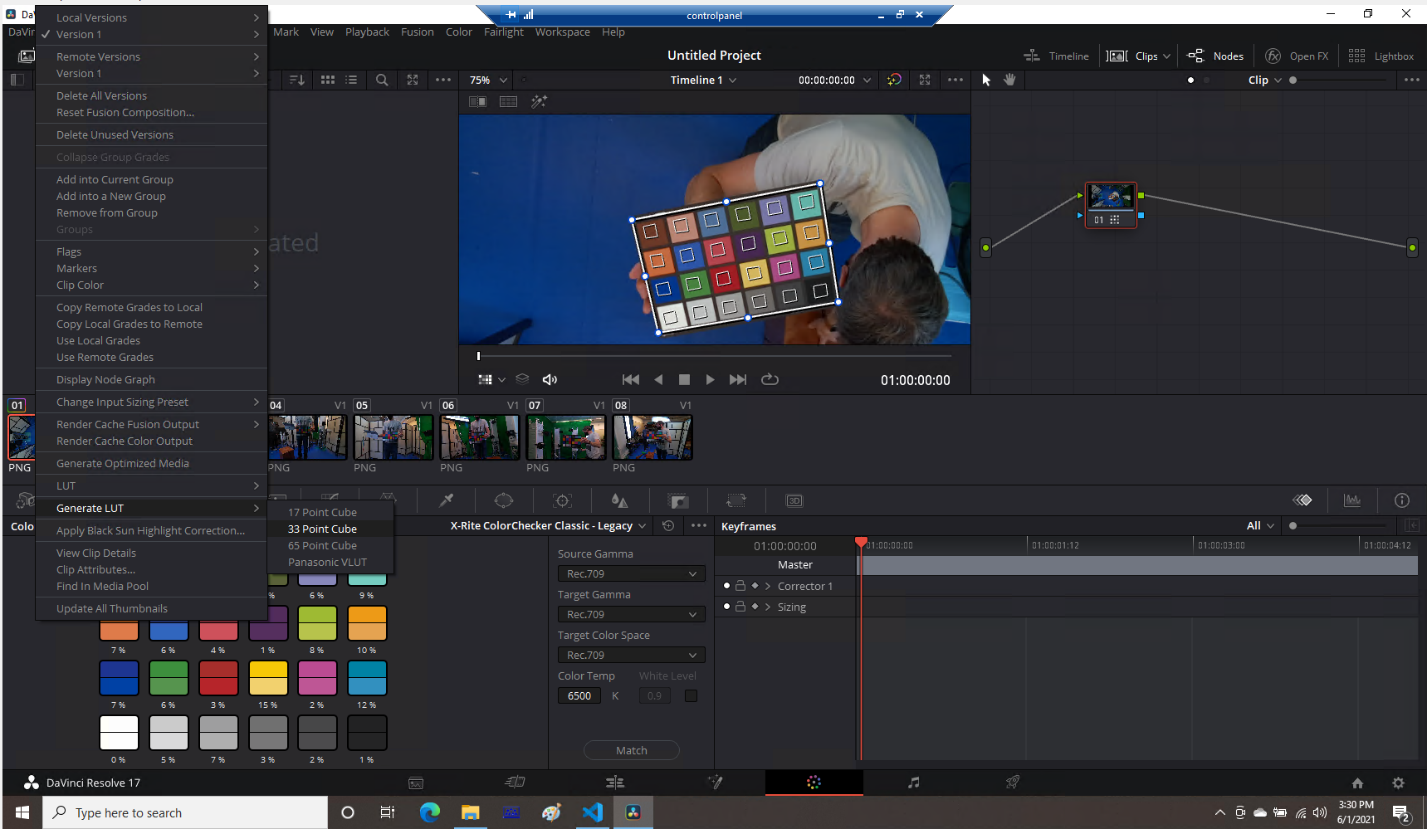
****

1. And now change the “Qualifier” (eyedropper) tool to “Color Chart” on the timeline preview window:  
   
2. Adjust the color chart outline so it lines up with the actual color chart in the image by pulling on the corners:  
   Graphical user interface, application

   Description automatically generated
3. Now, pick the color chart icon below the timeline:  
   
4. Make sure the color match board listed on the title matches your color match board (in this example (1), ours is the “X-Rite ColorChecker Classic – Legacy”). Change it if necessary, and then click “Match” (2). You’ll see the colors of the image preview change (3), and the numbered icon above the image in the timeline will become colored (4).



1. Right click on the image in the timeline and choose “Generate LUT” > “33 Point Cube”



1. Save this file in a safe place. You can name it whatever you’d like. Davinci will append the original filename (camX) to the filename anyway.
2. Repeat for the rest of the images.
3. Once you have all of the images processed, you can close Davinci. Rename all of the .cube files you saved to camX.cube, and copy them to the C:\3DTelemedicine\DATA\RendererCalibration\ folder.
4. Make sure that the 3dtelemedicine.cfg file contains  
   UseColorCorrectionLUT=true  
   in the [Renderer] section. And make sure you transmit this config over to renderer if you had to add that line.
5. You can verify the color correction is being used by looking at the log file from renderer. You should see lines like this:

Reading LUT:0

(Filename: C:\buildslave\unity\build\Runtime/Export/Debug/Debug.bindings.h Line: 35)

Reading LUT:1

(Filename: C:\buildslave\unity\build\Runtime/Export/Debug/Debug.bindings.h Line: 35)

Reading LUT:2

(Filename: C:\buildslave\unity\build\Runtime/Export/Debug/Debug.bindings.h Line: 35)

Reading LUT:3

(Filename: C:\buildslave\unity\build\Runtime/Export/Debug/Debug.bindings.h Line: 35)

Reading LUT:4

(Filename: C:\buildslave\unity\build\Runtime/Export/Debug/Debug.bindings.h Line: 35)

Reading LUT:5

(Filename: C:\buildslave\unity\build\Runtime/Export/Debug/Debug.bindings.h Line: 35)

Reading LUT:6

(Filename: C:\buildslave\unity\build\Runtime/Export/Debug/Debug.bindings.h Line: 35)

Reading LUT:7

(Filename: C:\buildslave\unity\build\Runtime/Export/Debug/Debug.bindings.h Line: 35)