Setting up Jetson TX1 Headless with usb cameras

Step 1:

**Flash the SD card with the appropriate image for the Jetson TX1 module** - Flashing an SD card for the NVIDIA Jetson TX1 involves installing the operating system and other necessary files to prepare it for use with the Jetson module. Here are the general steps to flash the SD card for Jetson TX1:

Prerequisites:

SD card with sufficient capacity (at least 16 GB).

Download the Jetson TX1 SD Card Image: Visit the NVIDIA Jetson Download Center and download the appropriate JetPack version for the Jetson TX1. [URL: <https://developer.nvidia.com/embedded/downloads#?tx=$product,jetson_tx1>]

The JetPack package includes the SD card image. After the package is downloaded, the image should be in .iso format. Flash the SD card using Etcher and it should take about 10 minutes. Once the flashing is done, insert the SD card in the Jetson TX1.

Step 2:

**Setup the host computer to access Jetson TX1 headless** – Develop an SSH connection over the same network through Ethernet. Connect the ethernet to the Jetson TX1 and connect your personal computer with the same network through WIFI.

Steps after Jetson is connected to a stable internet to set up an effective seamless SSH connection:

1. Identify the IP address for Jetson TX1 and the host computer and make it static so it is not needed to modify the SSH command every time a connection needs to be established.
2. Generate an SSH key pair on the host computer to share it with Jetson. Commands used:
   1. SSH key generation - ssh-keygen -t rsa
   2. Copy ID from host computer to Jetson - ssh-copy-id username@JetsonTX1IP
   3. Test the SSH connection – ssh username@TX1IP
   4. If a password is asked disable password authentication and enforce key-based authentication in the SSH configuration file –
      1. OpenSSH (most Linux distributions): sudo nano /etc/ssh/sshd\_config (or any other text editor like vim)
      2. Locate the line that starts with PasswordAuthentication. If the line is commented out (prefixed with #), remove the # to uncomment it. It should look like this:

PasswordAuthetication yes

* + 1. Change the value of PasswordAuthentication to no [If the line is not present, it can be added]
    2. Save the changes based on the editor used; in nano it is ctrl+o
    3. Restart the ssh - sudo systemctl restart ssh
  1. Test by sending a file from TX1 to host to see if any password is asked – scp or rsync either can be used; using rsync :- /path/to/rsync -r “/path/to/the/file” hostusername@hostIP:”path/to/the/destination/folder” [Note: Entire path is necessary for any command line tool used]

Step 3:

**Start OpenCV code remotely on the Jetson TX1 from host computer:** The above is only test whether SSH connection is working seamlessly. The same has been replicated in a separate piece of code where the SSH connection is generated using the “paramiko” library. Code is in repository.   
[Note: Video streaming cannot be started through SSH unless X11 forwarding is supported or while generating SSH connection, ssh -X was used but in my case it still did not work so I commented out the “cv2.imshow()” and it executed properly and the files were saved according to the OpenCV code which was based on timestamps.]

Step 4:

**Setup a cron job to schedule file transfer after a specified interval:** Cron is a time-based job scheduler in Unix-like operating systems, and it allows you to schedule and automate tasks at specific intervals. Steps to set it up:

1. In the Ubuntu terminal of TX1 open crontab using “crontab -e”
2. Define the cronjob for eg in my case it looked something like this:

1 \* \* \* \* /usr/bin/rsync - r “/path/of/the/videos/”hostname@hostIP:”/path/to/destination”

Here 1 represents at an interval of 1 hour.

1. Save and exit.
2. Verify the cronjob – In the terminal execute the command “crontab -l” and see if the changes were made.

**Key points to remember:**

1. Update pip and install all dependencies. Update python and make sure to mention the entire path to python of the latest version.
2. Generating key multiple times when once the SSH connection is not established creates overwriting of keys which can cause problem in the systems to identify the right key. It is important to remove the previous keys first and then create a new one to pair. It can be done using these command lines:
   1. cd ~/.ssh/
   2. ls
   3. rm id\_rsa id\_rsa.pub [replace id\_rsa with the name of your keys]
3. If there are lot of issues that are being encountered because of package dependencies or missing packages or packages not being located, use command “sudo apt autoremove” to remove all package and their dependencies and download again. Best to use everything through a virtual environment.
4. If there is an error that goes along the line “\*\*\*\*\*\* is in use” or some package not being able to be accessed because it is used by another tool, disable the auto update on the Ubuntu system, kill all the running processes and try again.

*[P.S. If there is any other issue encountered, please add to the document for future reference purposes]*