Journal

Wednesday, September 25, 2019 9:38 PM

9/25/2019

Goals:

- [x] Boot device
- [x] Connect to the device
- [x] Does the camera work on the device?
- [x] Can I manipulate the remote switch with the device?
- [x] Found best position for the camera

Upon taking the contents out of the bag I noticed the SATA connection was bent up. I took pictures of the connector in this state. In setting up the Jetson I bumped the SATA connecto and it fell off. =<

Looking for a IP to log into. It's easier to work on a Raspberry PI through SSH so I'm assumin will be for this as well.

```
sudo nmap -sP 192.168.1.0.24
```

SSH Login commands:
NOTE: -Y is to allow X11 forwarding

```
ssh -Y learner@192.168.1.124
deep
```

This camera script worked.
NOTE: VideoCapture is 1. 0 must be the default camera for the device but cv2 throws an err when trying to connect to the device.

```
import cv2
camera = cv2.VideoCapture(1)
For i in range(10):
    return_value, image = camera.read()
    cv2.imwrite('opencv'+str(i)+'.png', image)
```

r

g it

or

```
camera.release()
```

I just ran this in the python debugger cli tool. I didn't build this into a script.

Viewing pictures works using eog but the images take a little while to transfer over. It might faster to just send them using scp

Had to get pyHS100 installed to talk to the smart plug. The pip3 command wouldn't work unless I used the sudo -H command like so:

```
sudo -H pip3 install pyHS100
```

The pyHS100 package can discover the plug on the network like so:

```
from pyHS100 import Discover
for dev in Discover.discover().values():
    print(dev)
```

And then to turn it on and off it like so:

```
from pyHS100 import SmartStrip
plug = SmartStrip("192.168.1.131")
plug.turn_off()
plug.turn_on()
plug.turn_off(index=1)
plug.turn_on(index=1)
```

Adding the 1 as the parameter will turn off or on just one side of the plug.

Script to get images from the camera, saved at ~ as pic.py:

```
import cv2

cam = cv2.VideoCapture(1)
return_value, image = cam.read()
cam.imwrite('mid-bar3.png', image)
cam.release()
```

The best spot might be on the bar looking down about 6" in from the right.

be

Oct 1, 2019

- [x] Rebound from having to switch devices.
- [x] Finalize camera location - done is better than perfect

Looked into other job scheduling systems.
Celery looks cool
Python schedule should work for this project.

Oct 4, 2019

- [x] Take 100 pictures

Took 2000+

Oct 5, 2019

Taking more pictures.
Changed the way I named the image files.
This script helped rename the files

```
for old in 21*; do new=$(echo $old | sed -r -e 's/(^[0-9]{2})([0-9]{2})([0-9]{2})/04_\1-\2-\3/g'); mv "$old" "$new"; done
```

Which was adapted from this stack overflow response
https://stackoverflow.com/questions/1961255/rename-files-using-a-regex-with-bash

Oct 6, 2019

Moved sorted pictures into clean and dirty photos while weeding out unusable photos

Oct 7, 2019

- [x] Retrain model
- [x] Write script
- [x] Test

- [x] Improve (remove foreground?, better dataset?, different algorithm?)

All the files should have been JPGs instead of PNGs. Converted using:

```
for i in *.png ; do convert "$i" "${i%.*}.jpg" ; done
```

Basic script outline
Take picture
Analyze with ML
Switch plug depending on output.

~~Switch plug depending on output~~

Oct 8 2019

The current model for some reason calls everything dirty. Going to retest to recreate the err to see if I just didn't enter the data correctly or if the data I have isn't sufficient to retrain th model on.

Or maybe I can work with the values I'm getting if the messy score is near 100 we know it's messy if it's lower than some threshold we could say it is then clean enough.
Values are Messy / Clean

Test	Test1	Test2
Test with lights on full and clean floor	88% / 11%	78% / 21%
Test with lights on dim and clean floor	99% / 0.006%	99% / 0.003%
Test with lights on full and battery on floor(close)	99% / 0.000%	99% / 0.002%
Test with lights on full and battery on floor(far)	58% / 41%	58% /41%

Looking at the model details here:
https://tfhub.dev/google/imagenet/inception_v3/feature_vector/1
It looks like the model was made with grayscale training data and my color images might no do so well seeing as how they don't match the training data very well.
The version of the model is outdated. Performance might be improved by retraining with th updated model.

Second model tested is:
https://tfhub.dev/google/imagenet/resnet_v2_152/feature_vector/3
Ran out of memory on this model. Retrying with
https://tfhub.dev/google/imagenet/resnet_v2_101/feature_vector/3
This one failed to produce the labels file after making the model.
Tried these and others and none have finished to completion.

Trying to convert the images to grayscale and rerun with the model I know is working.
Not much changed

Tried removing the really dark images from the clean set.
Not much changed
Values are Messy / Clean

Test	Test1	Test2

ror
e

t

e

Test	Test1	Test2
Test with lights on full and clean floor	87% / 12%	90% / 9%
Test with lights on dim and clean floor	92% / 0.07%	83% / 16%
Test with lights on full and battery on floor(close)	99% / 0.009%	99% / 0.001%
Test with lights on full and battery on floor(far)	98% / 0.011%	96% / 0.032%

Attempting to further refine the dataset to find better representations of a clean and messy room did not provide any benefits to the results.'

The best results are achieved by putting a lamp on the floor that is pointed towards the area covered by the webcam. When the picture is taken the light comes on and the shadows cast by the light on the floor get the most consistent readings of clean or messy. The model alwa reports that it's messy but the clean values are either very very small or not so small.

Conclusion

I could not find any set of values that definitively told me the room was clean or messy. The values seemed to be different based on the levels of light in the room. Sometimes with a cle room the model still predicted the room to be messy.

I think the model didn't end up functioning as intended because I lacked enough data to thoroughly represent clean and messy. Given weeks of data the model would probably function closer to what I was expecting it to do.

Also possibly, a model trained in generic image classification struggles at confidently answering just a general question of whether the floor has objects on it or not. This may just be a case where transfer learning wouldn’t achieve the same results as a model trained specifically for the task.