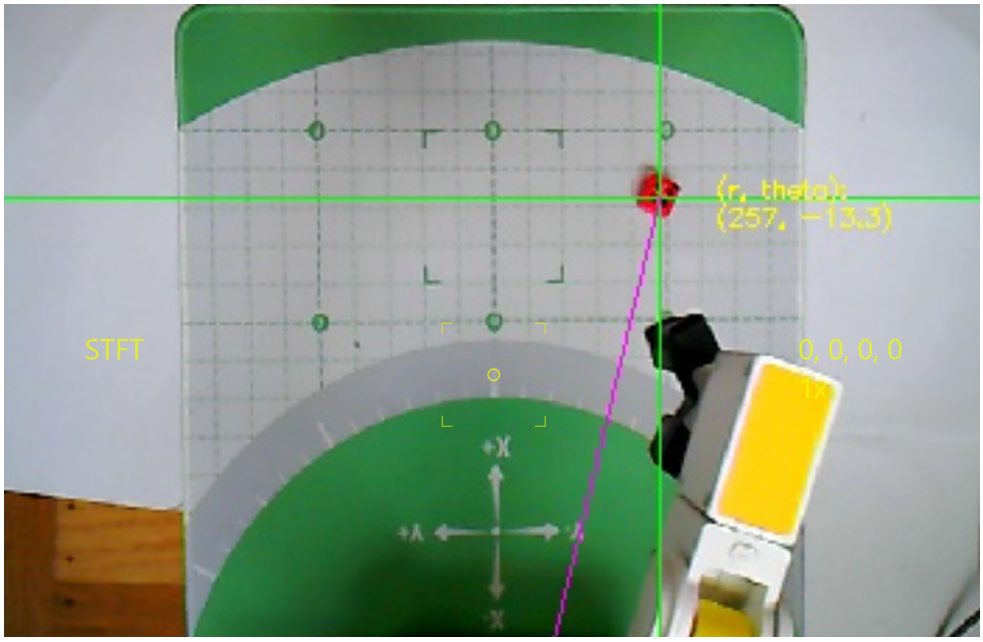
STFT Exhibit:
Color Tracking Dobot with Overhead Camera



[Video link]

Description: RETA-like app which streams video feed of a color sorting Dobot arm to the IP address of the network to which the server device (Pi) is connected. While streaming, a series of Opencv2 functions are used to detect colored objects from video feed. The Pydobot module is used to center the Dobot's claw on colored objects detected by Opencv2 functions. The app runs in an infinite loop until manually stopped by a user on the Pi. When running, the app will instruct the Dobot to pick up colored objects and drop them to a predetermined position to its left until all objects of a desired color are cleared from the original camera view.

Estimated Time: 2 - 3 hours (more is better)

Things to show off:

- Demonstrating the dobot-color-tracker-overhead app
    - Explain that this app just uses computer vision. No AI or machine learning is used but this app, but this app is our step in that direction
- Demonstrating how to track objects of another color

Space Required:

- 1 to 2 square meters

Materials:

- Device
    - Laptop or phone which is connected to hotspot is necessary to ensure success

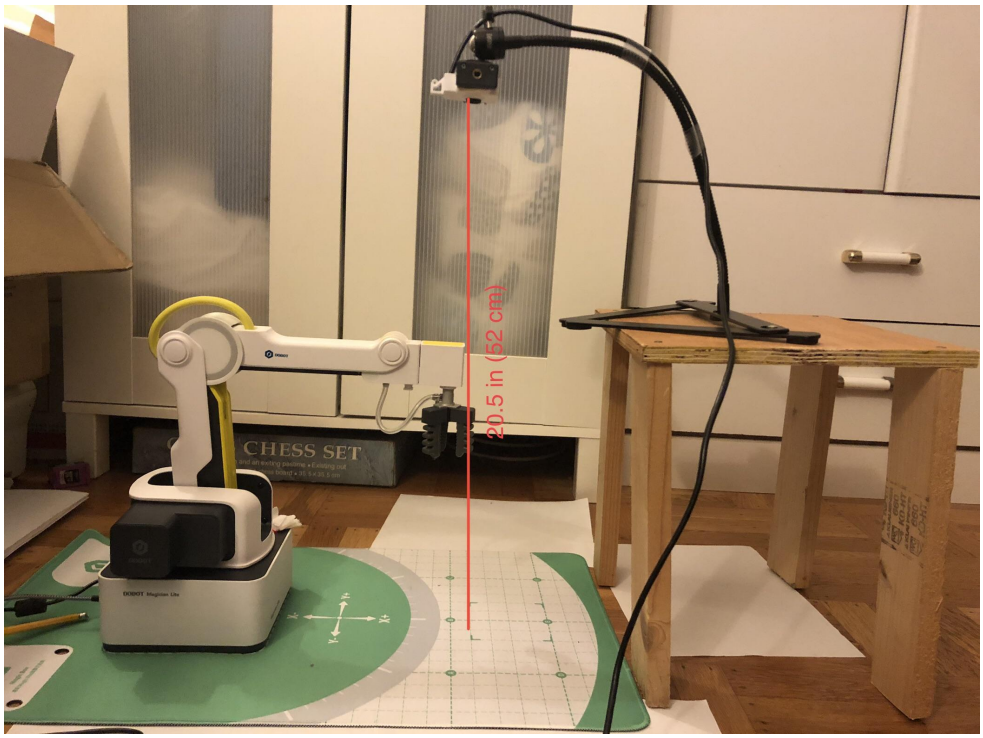
    - I HIGHLY recommend bringing a laptop which can be connected to whatever network the Pi is connected to
        - Laptop will make editing code on the fly much easier
    - A phone will work ONLY if you have the app Terminus or another app that gives you the ability to SSH into devices
    - **Without a device that can SSH into the Pi this exhibit will not work**
- Dobot Kit
    - Dobot Magician Lite
    - Power supply
    - Claw gripper
    - USB Type C (for connecting to Raspberry Pi)
    - Colored Blocks
    - Dobot mat
- Raspberry Pi
    - Pi power supply
- Camera setup
    - Dobot USB camera
    - Camera stand
    - Flat elevated surface (if needed)
- Paper
    - VERY necessary for ensuring that color detection works
- Tape
    - Could be useful in case something comes loose
    - Can be any tape, just don't leave a mess

Setup: Place the Dobot on the dashed square of the Dobot mat. Just like a regular RETA setup, you need to power the Pi, Dobot, and camera. Then you need to make the USB Type C connection between the Pi and Dobot. The camera should be plugged into one of the Pi's USB ports. Connect the Pi to an internet source so that it can at least broadcast the app to devices connected to that internet source. To execute commands on the Pi's terminal, you will need to SSH into the Pi with a device which is connected to the same network which the Pi is connected to. If using an iPhone, I suggest using the app Terminus to access the Pi through SSH. If you have a laptop (not chromebook) that is even better! Just use the laptop's built-in command line or terminal SSH command.

Running App: The app is stored in a directory called dobot-color-tracker-overhead. In this directory is a file called app.py. To start up the app, the Pi must be connected to an internet source. Hotspots work. If you are using a hotspot, you just need to know the IP address of the hotspot. Once the app is running, enter the IP address followed by ":5000" to view the app. ":5000" refers to port 5000 which the app is streamed on. Once the app is running, the Dobot should start moving once you open the app in your browser!

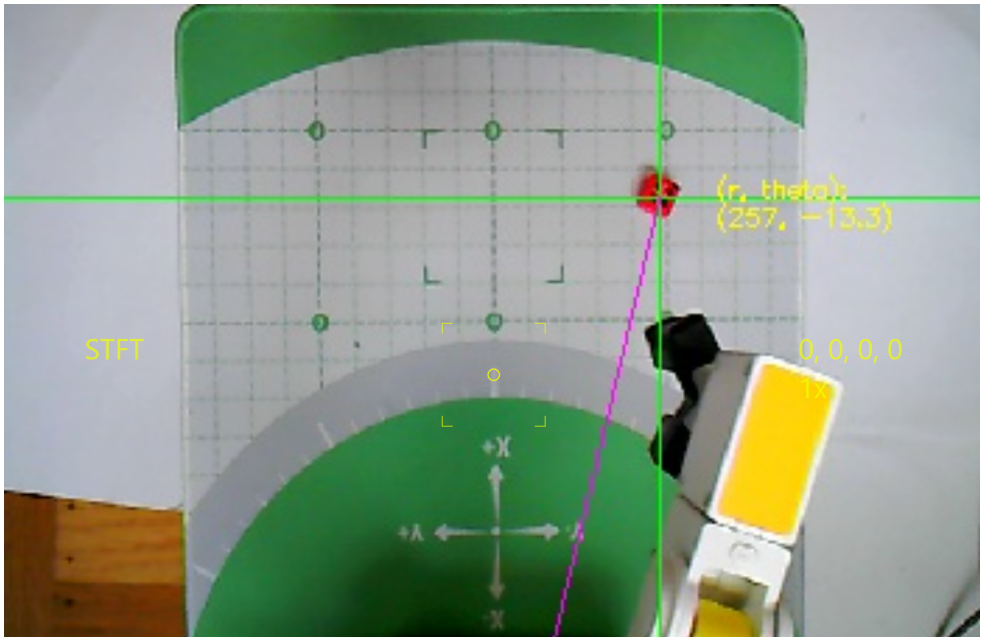
Camera Setup: Remove any blocks on the mat. While the app is running, position the Dobot USB camera so that it is pointing directly down and its lens is 52 centimeters above the mat. The side of the camera with the hinge should be pointing towards the Dobot.

Example Setup:



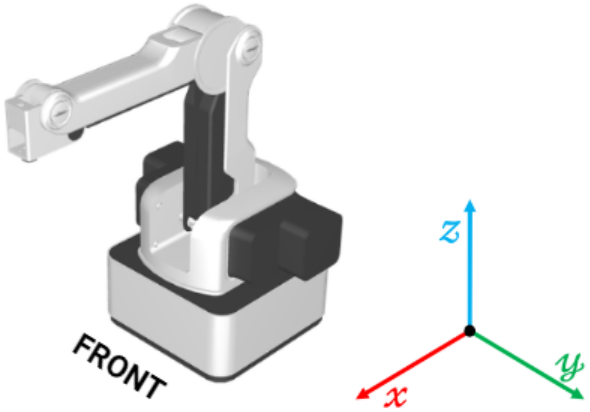
The Dobot USB camera is pointing down and located 52 centimeters above the mat. Note how the hinge protruding to the left of the camera is pointing towards the Dobot. (Raspberry Pi not shown)

The camera feed should resemble something like this, with the Dobot arm extending from the bottom of the image.

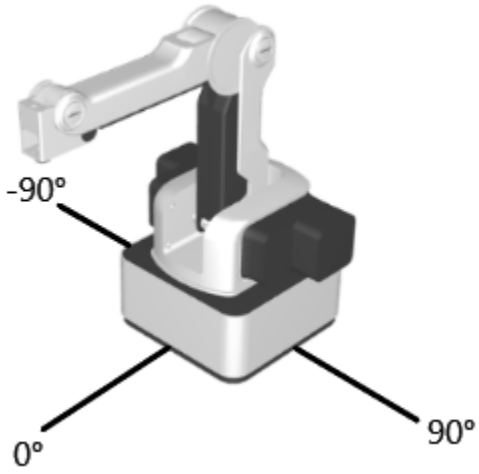


Place a block on the mat and immediately remove it. A pair of green perpendicular lines that intersect where the block was should appear. Adjust the camera to ensure that the horizontal green line is parallel to the mat's y-axis and that the vertical green line is parallel to the mat's x-axis. The center of the image, indicated by the small yellow circular reticle, should be the rounded end of the thick white 0° increment mark at the center of the mat. (See below for a better idea of spatial stipulations.)

Cartesian coordinate system:



Polar angles:



Notes on editing code on Pi: You can either use the text editors nano or vi which are already on the Pi's terminal or VS Code's remote SSH extension if you have a laptop. The thing you need at the very least is a phone with Terminus. With this, you can use nano or vi through Terminus on your phone (this is a really not fun way to code though).

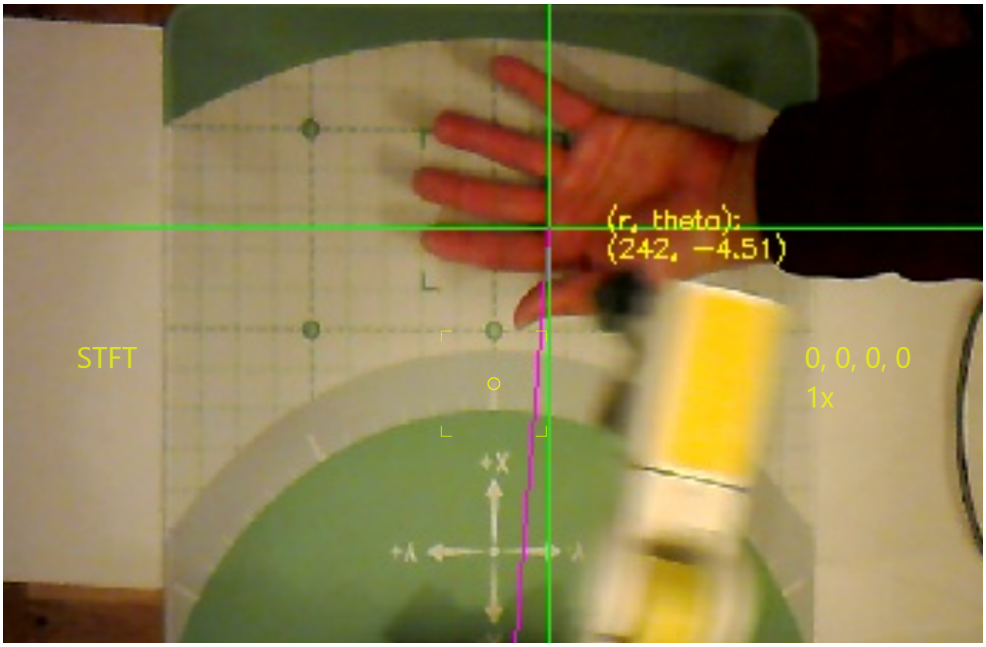
Detecting different colors: Unfortunately no button to simply switch colors is available, however I've made "options" for switching colors to detect in app.py. To switch colors, simply uncomment lines 101-102 for red, 103-104 for green, 105-106 for blue, or leave 107-108 uncommented for yellow. ONLY ONE COLOR SHOULD HAVE ITS LINES UNCOMMENTED. Since app is written in Python, comments use hashtags "#". This should give you some experience with working on some code for the app lol. Sorry about this inconvenience.

```
99
100     # Refer to this for post for how to input colors in HSV format: https://stackoverflow.com/quest
101     low_color = np.array([161, 155, 84]) # Red in HSV format
102     high_color = np.array([179, 255, 255])
103     # low_color = np.array([40, 155, 84]) # Green
104     # high_color = np.array([80, 255, 255])
105     # low_color = np.array([95, 155, 84]) # Blue
106     # high_color = np.array([130, 255, 255])
107     # low_color = np.array([20, 150, 140]) # Yellow
108     # high_color = np.array([50, 255, 255]) # WARNING: YELLOW REQUIRES GOOD/NEUTRAL LIGHTING
109
```

Screenshot of the lines of code responsible for which color the Dobot detects. Red and blue work well, but yellow is a little problematic in poor lighting and green requires paper covering the mat.

Known issues:

- The algorithm detects the Dobot's LED as a block to be picked up. To avoid this, cover the LED up with many layers of paper.
- The algorithm detects hands as a block to be picked up. To avoid this, tell audience members not to place their hands in front of the camera.



- Frames are not outputted during grabbing.