

11/01

my mind is currently blank on what SPECIFIC topics our research project will contain but here are some general links for now

- no matter what i really want include something to do with videos

a

Brainstorming (basic ideas no plans for them just whatever i can think of rn):

-Doubles: the ai analyzes and finds patterns in player rotation related to shot selection that lead win/lose points -> helps better rotation. the articles below mainly about shot selection and racket form and not where to move to better control court.

05/20 (Angela):

- at first, I thought our choices were limited due to the features of the Clutch AI app
- however the app works only with pre-installed cameras so you have to be at a specific location
- so there are many many possibilities:
 - video analysis (how many shots are out, the amount of shots in each corner of the court)
 - it is true that we will need the camera to be at a specific angle (no random camera placement; i mean it is possible but extremely complicated I THINK)
 - what is different about this is that any camera use is possible and not just a “special ai camera” at specific locations around the US

AI + video + badminton examples

<https://www.mdpi.com/2313-433X/9/5/99>

- presents a method for predicting the future trajectory of the badminton shuttlecock during a match. One of the simple methods utilizes sequential models, such as recurrent neural networks (RNNs) [9], to model the motion of the shuttlecock by inputting previous shuttlecock trajectories and outputting future shuttlecock trajectories. To better reflect a player’s decision-making process during a match, the proposed method also considers the player’s position and posture information in addition to the shuttlecock’s position information.

<https://pmc.ncbi.nlm.nih.gov/articles/PMC11244353/>

- First, TrackNet, a deep neural network designed for tracking small objects, is used to extract the flight trajectory of the shuttlecock. Second, the YOLOv7 model is used to identify whether the player is swinging. As both TrackNet and YOLOv7 may have detection misses and false detections, this study proposes a shot refinement algorithm to obtain the correct hitting moment.

<https://arxiv.org/html/2403.12385v2>

- In this paper, we introduce the VideoBadminton dataset, which is derived from high-quality badminton footage.

<https://apps.apple.com/is/app/clutch-ai-for-racket-sports/id1607986804>

- app for sports analysis

▶ Clutch: The AI-powered Badminton Analysis and Coaching App - OpenCV Weekly 90

▶ VIRL: Immersive Match Video Analysis for High-Performance Badminton Coaching | VIS ...

[Harvard Paper](#)

Signature shot. Of players you can gain insights on things to try during your game

<https://www.airitilibrary.com/Article/Detail/10247297-202006-202007060015-202007060015-201-213>

Badminton new beginners. Analyze

<https://www.mdpi.com/1424-8220/24/13/4372>

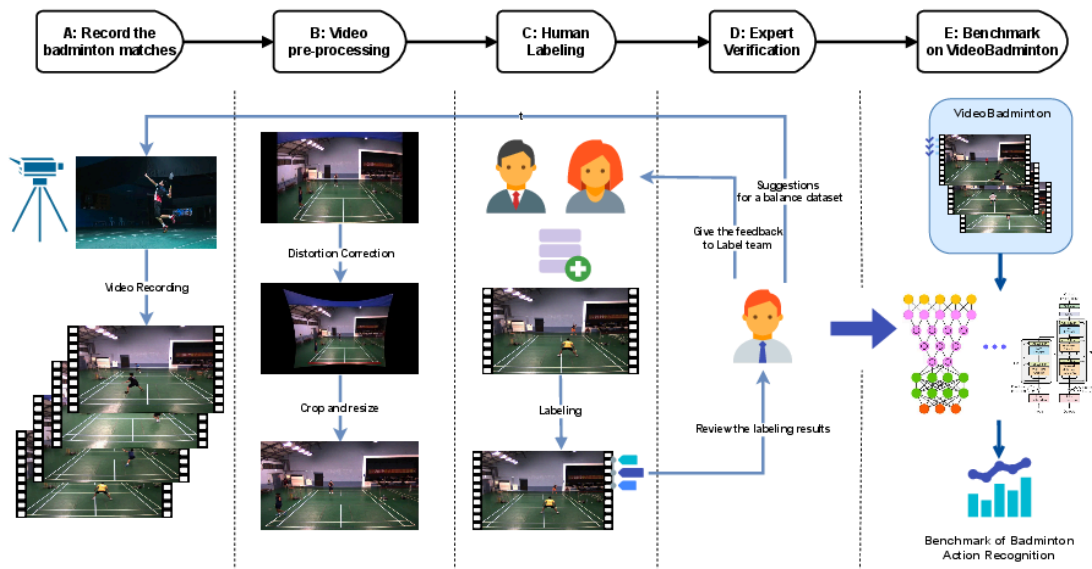
Hi, I'm testing the tracknet thing mentioned in the paper on collab. Lemme see if i can share

Rough Idea: App that allows you to find **signature shot** & maybe form stuff for new beginners. Clutch is only working on specific courts and ours is on like every video so we are unique to them

I think we should figure out a rough project idea at cif maybe instead of just learning this stuff.

General timeline:

1. Figure out how to load videos and opencv basic stuff
 - a. Use opencv to process the video, grayscale and cropping, etc. It was super easy i followed a tutorial teaching basics online.
2. Figure out yolo or mediapipe
 - a. Put a box around players(**s**) maybe birdie?
2. THE HARD PART (probably hard part?): classify the shots
 - a. <https://github.com/qilimk/VideoBadminton/tree/main>
 - b. Not sure



c. <https://github.com/qilimk/VideoBadminton/tree/main>

What are the features that Clutch AI has?

- utilizes AI to track player movement and shots
- creates highlights
- OOOHHH SNAP this only works if there is a camera installed
 - i downloaded the app and it told me to scan a QR code in order to get AI analysis

Form Correction

1. Get Data
 - a. Elbow Position
 - b. Elbow angle
 - c. which way the racket is facing
 - d. how do you tell if the form is not open? → elbow position is closer
 2. Elaborating on get data:
 - a. 100 good ones
 - b. 150 bad ones
 3. Labeling Data
 - a. I go to training thursday or friday EBC
 - b. What days are we getting data?
-
1. Head orientation (looking up at birds?)
 2. Left elbow, left wrist
 3. Right Elbow
 4. Right Wrist (or maybe racket orientation?)

Get the photos from one side and set distance. Vary angles photo is taken from <- not sure if needed or not

tasks

- ☐ connect CNN with mediapipe - angela (**July 9th**)
- ☒ ~~Mediapipe and camera to react native - max (July 7)~~
- ☒ ~~npv expo ios???? - angela (July 3rd)~~
- ☐ make UI look better (**before summer ends HARD DEADLINE**)
 - canva first - angela, actual code - both
- ☐ Label new images, Add project to github (might be a different repo) 7/14 - Max
- ☐ Fix small camera issues (stuck in back camera, etc.) - Max (Sometime)
- ☐

<https://www.mdpi.com/1424-8220/24/13/4372>

 sensors-24-04372.pdf - my annotated document

https://colab.research.google.com/drive/1Ha6Wm-_30jWMrqJpDaQ3oalgW0iGwr_s#scrollTo=5JsfD87z344v

Idk how collab sharing works but i was testing tracknet
hoow are you already done reading the paper 🤔 oh i see
Lemme finish reading it

Notes:

- “After extracting the ankle coordinates of the players in each frame, these coordinates are projected onto an aerial view by **utilizing perspective transformation** (see Figure 6b)” How did they do this?
- “These videos feature multiple players performing strokes against various backgrounds, taken from official competition videos.” The VideoBadminton dataset I found might not be good
- Change confidence scores of YOLO to get more accuracy, Peak Identification and Direction change with TrackNet
- “The objective of the proposed SRA is to combine the results of both modules, thereby providing the precise identification of the hit moment”
- Cases”
 - One bird trajectory in one hitting segment
 - More than one hit trajectory in one segment (cannot determine which player hit)

- Case 3: Multiple $HD-T$ in $HS^{(k)}$, $k = 1, 2$

This scenario represents a concurrent occurrence of correct detection and false positives (see Figure 11), given that only a single hit moment can exist within a swing action. Therefore, the frame with the highest confidence score in $HS^{(k)}$ is initially identified. Subsequently, set the $HD-T_i$ closest to this frame as $HM^{(k)}$, while the others are discarded.

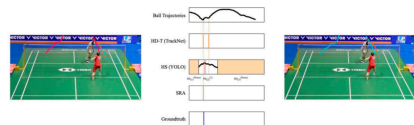


Figure 11. An example of case 3. The colored lines represent the extracted trajectories. Multiple $HD-T$ s occur in a hit sequence which belongs to a player, so only one $HD-T$ is retained.

Case 3 is weird im≠ not sure i get it

- **Hit Trajectory breaks but YOLO has high confidence it is a hit shot**
- **No HDT no one is currently trying to hit the bird**

- "The primary objective of the algorithm is to eliminate false positives generated by HD-T with the assistance of HS"
- "Regarding the shuttlecock-tracking module, we follow the algorithm proposed by TrackNet, which includes the "Peak Identification Method" and "Direction Identification Method", as described in **Section 2.5**, to **detect the sudden direction change of the shuttlecock.**"
- "Because the proposed method includes an additional YOLO model, the overall processing time is increased. Even though the processing time of SRA is longer than using TrackNet alone, we believe this trade-off is worthwhile.
- **Shot classification:**
 - Both players' ankle positions (x_1, y_1, x_2, y_2) at the hit moment, which is obtained by the method proposed in Section 3.1.
 - Flight duration of the shuttlecock, denoted as T in regard to the number of frames
 - The displacement of the shuttlecock ($\Delta x, \Delta y$), quantified as the difference in its position
 - PERSPECTIVE TRANSFORMATION

Okay I found the tracknet github it actually works i just tried it on google collab.

<https://github.com/yastrebksv/TrackNet?tab=readme-ov-file>

11/02 Mediapipe is okay for multi-person tracking

<https://shawntng.medium.com/multi-person-pose-estimation-with-mediapipe-52e6a60839dd>

I implemented it this morning. The code is long but the logic is easy to understand.

WAYYYY better I'll show today at sujus

- I was thinking of making yolo only track moving targets so that the approach in the article performs even better.

Annotation

https://drive.google.com/file/d/1hro9UVzncW3bgra3EVgM5msuZFFQgRqW/view?usp=drive_link

11/03

Ugh tried an idea of only tracking moving people but players don't always move in each frame and setting a good threshold is hard to also rule out random ppl who move in background. Will try some other stuff later and talk to parents about ebc pictures

11/10

testing real-time web recording on mediapipe

first error:

```

Ⓢ (base) angela@Eric-MBP ~ % pip install opencv-python mediapipe
Collecting opencv-python
  Downloading opencv-python-4.11.0.86.tar.gz (95.2 MB)
    95.2/95.2 MB 6.4 MB/s eta 0:00:00
Installing build dependencies ... done
Getting requirements to build wheel ... done
Preparing metadata (pyproject.toml) ... done
ERROR: Could not find a version that satisfies the requirement mediapipe (from versions: none)
ERROR: No matching distribution found for mediapipe

```

mediapipe can only support python 3.7-3.10?

it's taking forever to download python 3.10 (i needed to create a virtual environment)

i have coaching until 4PM i will finish once i get back home ← yeah this was really weird I had to downgrade too

commands i ran in the terminal:

python --version

brew update

brew install pyenv

brew install pyenv-virtualenv

nano ~/.zshrc

```

export PYENV_ROOT="$HOME/.pyenv"
export PATH="$PYENV_ROOT/bin:$PATH"
eval "$(pyenv init --path)"
eval "$(pyenv init -)"
eval "$(pyenv virtualenv-init -)"

```

(PUT AT THE BOTTOM)

THEN CONTROL+O & ENTER TO SAVE

CONTROL+X TO EXIT

source ~/.zshrc

pyenv virtualenv 3.10.13 mediapipe-env

pyenv activate mediapipe-env

**THERE SHOULD BE (mediapipe-env) IN FRONT
FOR EXAMPLE (next command):**

(mediapipe-env) (base) angela@Eric's-MBP ~ % pip install mediapipe opencv-python

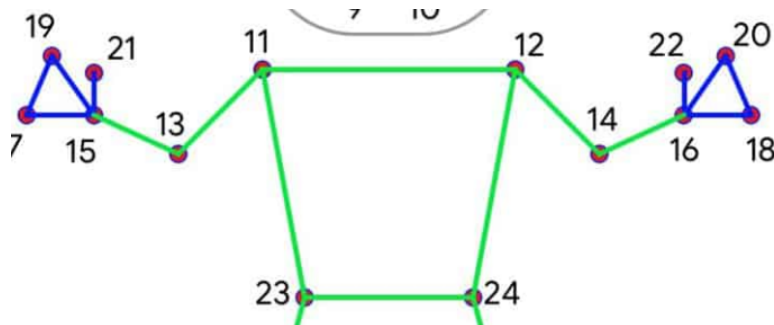
MAKE SURE CAMERA IS ENABLED FOR VSCode

python mediaPipeTest.py

```
lmList = []
if results.pose_landmarks:
    for i in range(11,25):
        lm = results.pose_landmarks.landmark[i]
        if lm.visibility > 0.5:
            h,w,c = img.shape
            cx,cy = int(lm.x*w),int(lm.y*h)

lmList.append([mp.solutions.pose.PoseLandmark(i).name,cx,cy])
mpDraw.draw_landmarks(img, results.pose_landmarks,
mpPose.POSE_CONNECTIONS)
for idx, (name, x, y) in enumerate(lmList):
    print(f"Name = {name}, x = {x}, y = {y}")
```

Get useful screen coordinates and save to array of useful joints



```
import cv2 #only exists after it is downloaded in virtual environment
import mediapipe as mp #same situation as above

mp_drawing = mp.solutions.drawing_utils #draw landmarks and skeleton
mp_pose = mp.solutions.pose #access mediapipe's post estimation model
```

```

#open webcam (0 = default webcam)
cap = cv2.VideoCapture(0)

with mp_pose.Pose( #initialiize mediapipe Pose model
    min_detection_confidence=0.5, #minimum confidence to detect a person
    min_tracking_confidence=0.5) as pose: #minimum confidence to track keypoints across
frames
    #pose = Python object that represents the MediaPipe Pose estimation model
    # ^ contains everything needed to proccess images and return body landmark
positions

    while cap.isOpened(): #cap = VideoCapture object
        ret, frame = cap.read()
        # ret: returns True if a frame was succesfully read (i think you know this part
already)
        # frame: the actual image frame (NumPy array)
        if not ret:
            break

        #converts the BGR to RGB
        image_rgb = cv2.cvtColor(frame, cv2.COLOR_BGR2RGB)

        results = pose.process(image_rgb)
        # "results" object contains list of detected joint positions & (x, y, z,
visibility) for 33 body landmarks

        # draw pose annotations on image
        mp_drawing.draw_landmarks(
            frame,
            results.pose_landmarks, #landmarks (dots)
            mp_pose.POSE_CONNECTIONS, # lines connecting the joints
            # DrawingSpec = COLOR (visual styling)
            mp_drawing.DrawingSpec(color=(0,255,0), thickness=2, circle_radius=2),
#green for joints
            mp_drawing.DrawingSpec(color=(0,0,255), thickness=2) #red for lines
        )

        # opens a new window to display frame
        cv2.imshow('MediaPipe Pose - Webcam Feed', frame)

        # press 'q' to exit

```

```

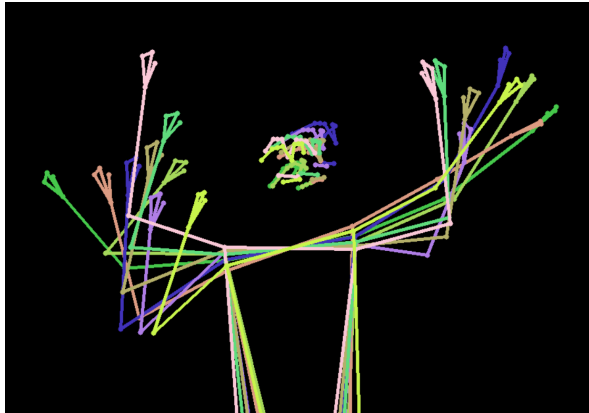
        if cv2.waitKey(1) & 0xFF == ord('q'):
            # the waitKey(1) makes OpenCV wait 1 ms to check if any key has been
            pressed (to stop the program).
            # if not, it returns -1 and moves onto the next frame
            # 0xFF ensures compatability of different keyboards
            break

cap.release() #stop access to webcam
cv2.destroyAllWindows()

```

Omg

I love chatgpt



The following code scales every person in the images so that their shoulder lengths are a set size.

```

import cv2
import numpy as np
import mediapipe as mp
import os
import random

mp_pose = mp.solutions.pose
pose = mp_pose.Pose(static_image_mode=True)
connections = mp_pose.POSE_CONNECTIONS

IMAGE_FOLDER = "images"
CANVAS_WIDTH = 1920
CANVAS_HEIGHT = 1080

#mediapipe is faster with smaller images and it only takes rgb
def preprocess_image(image_path, target_size=(640, 480)):
    img = cv2.imread(image_path)
    img_rgb = cv2.cvtColor(img, cv2.COLOR_BGR2RGB)

```

```

img_resized = cv2.resize(img_rgb, target_size)
return img_resized

def extract_keypoints(image):
    result = pose.process(image)
    if not result.pose_landmarks:
        return None
    return np.array([(lm.x, lm.y, lm.z) for lm in result.pose_landmarks.landmark])
    #returns an array of 33 landmarks. Each landmark the normalized x,y,z,visibility scaled
    0-1

def draw_skeleton(canvas, keypoints_3d, color):
    origin = np.array([CANVAS_WIDTH // 2 - 100, CANVAS_HEIGHT // 2 - 100])
    points = keypoints_3d[:, :2] #means only get first two values x,y

    left_shoulder = points[mp_pose.PoseLandmark.LEFT_SHOULDER.value]
    right_shoulder = points[mp_pose.PoseLandmark.RIGHT_SHOULDER.value]

    #basically distance formula but big words I think
    current_width = np.linalg.norm(left_shoulder - right_shoulder) + 1e-6

    # Scale so that shoulder width becomes exactly 200 pixels
    #change this later on if needed
    scale_factor = 200 / current_width
    scaled_points = points * scale_factor

    # Center skeleton around midpoint of shoulders
    shoulder_center = (left_shoulder + right_shoulder) / 2.0
    scaled_points -= shoulder_center * scale_factor
    scaled_points += origin

    #can't use mpDraw because scaled and everything not directly drawing.
    for x, y in scaled_points[:25]: # only joints 0-24
        cv2.circle(canvas, (int(x), int(y)), 4, color, -1)

    for start_idx, end_idx in connections:
        if start_idx <= 24 and end_idx <= 24:
            p1 = scaled_points[start_idx]
            p2 = scaled_points[end_idx]
            cv2.line(canvas, tuple(p1.astype(int)), tuple(p2.astype(int)), color, 3)

canvas = np.zeros((CANVAS_HEIGHT, CANVAS_WIDTH, 3), dtype=np.uint8)
for file in os.listdir(IMAGE_FOLDER):
    path = os.path.join(IMAGE_FOLDER, file)
    img = preprocess_image(path)
    keypoints = extract_keypoints(img)
    if keypoints is not None:

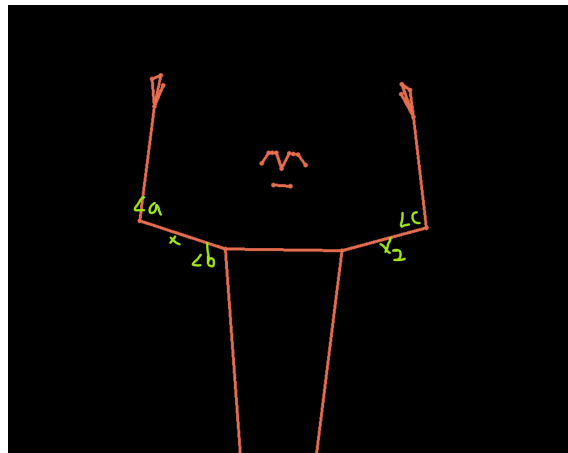
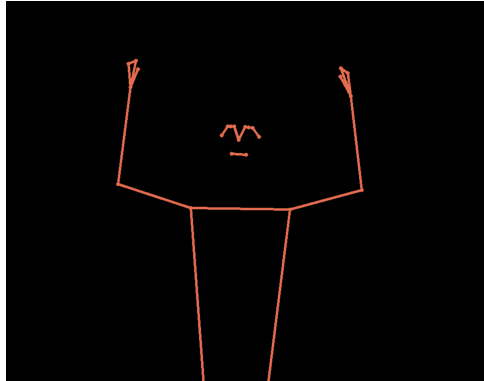
```

```

color = tuple(np.random.randint(50, 256, size=3).tolist()) #gives random color
draw_skeleton(canvas, keypoints, color)

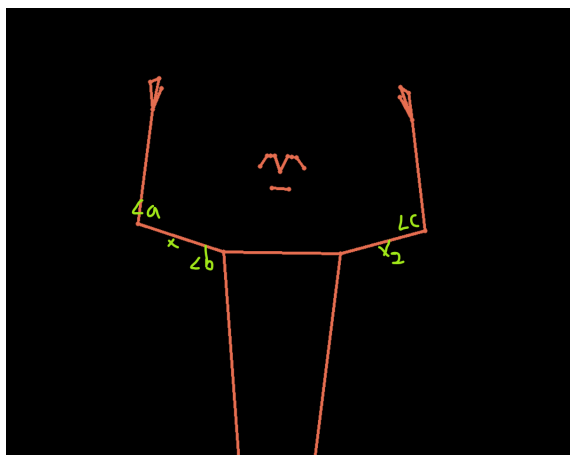
cv2.imshow("Spooky Skeletons", canvas)
cv2.waitKey(0)
cv2.destroyAllWindows()

```



Tomorrow other than getting more photos, goal: find angles/lengths (above is what I thought of so far)

11/12



joint info I thought of so far.

My idea of how training it will work?:

- Label the joints/angles on each image as right or wrong, Label form overall as right or wrong, for each joint/angle train a model to classify it as good or bad (sci-kit learn logistic regression or smth else), another model to classify form overall.
-
- Problemos: camera angle could be different so need to normalize each photo so person is exactly front view

- Need more images.

Do you think this will work to address the camera angle Problem. The function will check if the person is facing the camera and not rotated to the side to an extent. I'll send a video of it working later.

```
def is_front_facing(keypoints, angle_threshold_deg=60):
    ls = np.array(keypoints[mpPose.PoseLandmark.LEFT_SHOULDER.value])
    rs = np.array(keypoints[mpPose.PoseLandmark.RIGHT_SHOULDER.value])

    #get vector of your shoulders X,Z values and .norm makes it a unit
vector
    shoulder_vec = rs - ls
    shoulder_vec_xz = shoulder_vec[[0, 2]] #
    shoulder_vec_xz /= np.linalg.norm(shoulder_vec_xz)

    x_axis = np.array([1.0, 0.0])

    #Check angle between x_axis and your shoulder vector to determine what
angle you face away.
    angle_rad = np.arccos(np.clip(np.dot(shoulder_vec_xz, x_axis), -1.0,
1.0))
    angle_deg = np.degrees(angle_rad)
    return abs(angle_deg-90) >= angle_threshold_deg

while True:
    success, img = capture.read()
    if not success:
        break

    cropRGB = cv2.cvtColor(img, cv2.COLOR_BGR2RGB)
    results = pose.process(cropRGB)

    if results.pose_landmarks:
        keypoints = [(lm.x, lm.y, lm.z) for lm in
results.pose_landmarks.landmark]
        keypoints = list(keypoints)
        if is_front_facing(keypoints):
            pass
        else:
```



```

        cv2.putText(img, "FACE The Front Please", (100,100),
cv2.FONT_HERSHEY_SIMPLEX, 1, (0,0,255),3,3)

    #get screen pixel coordinates of keypoints
    h, w, c = img.shape
    lmList = []
    for i, (x, y, z) in enumerate(keypoints):
        if results.pose_landmarks.landmark[i].visibility > 0.5:
            cx, cy = int(x * w), int(y * h)
            name = mpPose.PoseLandmark(i).name
            lmList.append([name, cx, cy])
    # for idx, (name, x, y) in enumerate(lmList):
    #     print(f"Name = {name}, x = {x}, y = {y}")

    mpDraw.draw_landmarks(img, results.pose_landmarks,
mpPose.POSE_CONNECTIONS)

    cv2.imshow("img",img)
    if cv2.waitKey(1) & 0xFF == ord('q'):
        break

```

11/15 if we want to implement on a mobile application we also have to account for how the models and algorithms will run on a mobile device (which is not as powerful)

- some (especially the shot classification one with TrackNet) might need to run on the cloud or a server

platform-specific integration???

React Native & Flutter - cross platform framework

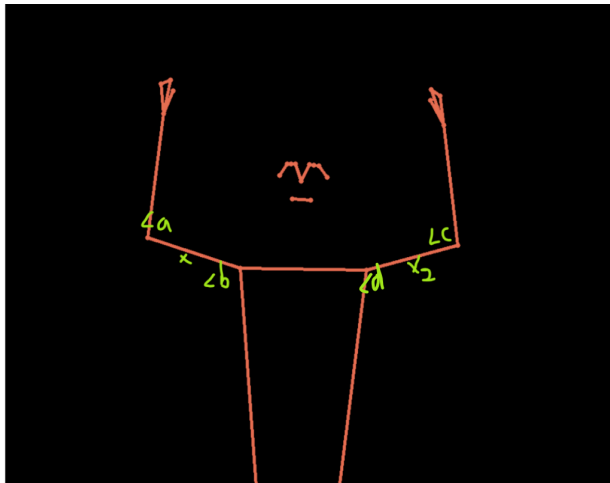
seems like we have to do a lot of backend processing

11/18

LABELS

- good
- Right elbow too low **Angle d**
 - Angle above or below shoulder line ML
- Right arm not opened
 - Z-index/Distance
 - ML
- Left arm not opened

- Z-index/Distance
 - ML
- Right elbow angle **Angle C**
 - ML
- Left Elbow too low **Angle b**
 - below the shoulder line hardcoded
- Left elbow angle **Angle A**
 - Hardcode greater than 90
-



LABELS

- good
- Right elbow too low **Angle d**
 - Angle above or below shoulder line ML
- Right arm not opened
 - Z-index/Distance
 - ML
- Left arm not opened

- Z-index/Distance
- ML
- Right elbow angle **Angle C**
 - ML
- Left Elbow too low **Angle b**
 - below the shoulder line hardcoded
- Left elbow angle **Angle A**
 - Hardcode greater than 90

good true

good true

good true

good true

good true

Right arm not opened false

Right elbow angle true

Right elbow angle, Right arm too low true

good true

Right elbow angle true

Right arm too low, Right arm not opened false

Right elbow angle true

Right arm not opened false

no label true

Right arm too low true

good true

Right arm too high true

Right arm not opened false

Right arm not opened false

Right arm not opened false

Right arm not opened false

Right arm not opened false

Right arm not opened false

Right arm not opened false

good true

good true

good true

good true

Right elbow angle true

Right elbow angle true
Right elbow angle true
good true
good true
good true
Right arm too low true
Right arm too low true
Right arm too low true
no label true
no label true
no label true
Right arm too low true
Right arm too low true
good true
good true
good true
Right arm too low true
Right arm too low true
Right arm too low true
Right arm too low true
Good true
Good true
Good true
Good true
Good true
Good true
Good true
Good true
Good true
Good true
Good true
Good true
Good true
Good true
Good true
Good true
Good true
Right arm too low
Right arm too low
Good true

Good true
Good true
Good true
Good true
Good true
Good true
Good true
Good true
Good true
Good true
Right arm not opened false
Right arm not opened false
Right arm not opened false
no label true
no label true
Right arm too low
Right arm too low
Good true
Good true
no label true
no label true
Right arm too low true
Right arm too low true
Right arm too low true
Right arm too low true
Good true
Good true
Good true
Right arm too low, Right arm not opened false
Right arm not opened false
Right elbow angle true
Right arm too low true
Right arm too low true
Good true
Good true
Right arm too high, Right elbow angle true
Right arm too high, Right elbow angle true
Right elbow angle true
Right arm too low, Right arm not opened false true
no label true

Right arm too low, Right arm not opened false
Right arm too low, Right elbow angle true
Right arm too low, Right arm not opened false
Good true
Good true
Good true
Good true
Good true
Good true

dataset making process

list of labels

- each individual index is also a list (especially when more than one label describes why the form is wrong)

dictionary:

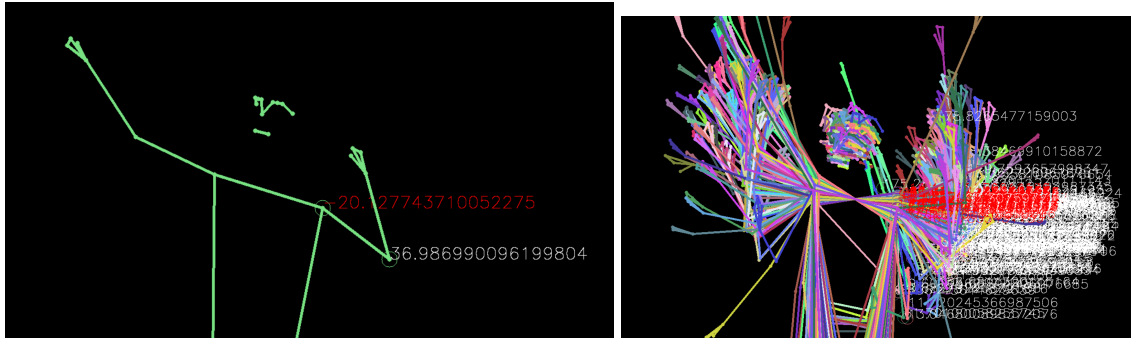
- {000.jpg : label, 001.jpg : label, etc.}

multi-label one hot encoding:

- 0 = not present, 1 = present
- ex: three labels [good, arm too low, elbow angle]
 - if form is good: [1, 0, 0]
 - arm too low: [0, 1, 0]
 - arm too low AND elbow angle: [0, 1, 1]

csv structure:

img_name, encoded label, (angle C?), (angle D?)



I still need to do the arm open/close thing tomorrow but I got Angle C and D.

Let me know how I should save the values

Currently just in an array

```
('bbb\\IMG_2429.jpg', 4.643728388337567)
('bbb\\IMG_2430.jpg', 7.743867519528239)
('bbb\\IMG_2431.jpg', 9.08527164141838)
('bbb\\IMG_2432.jpg', -0.29738724263919103)
('bbb\\IMG_2433.jpg', -5.465512317209335)
('bbb\\IMG_2434.jpg', -30.29124649229814)
```

Main Function. Hard to read I'll make it clearer later or just explain in person gtg sleep.

```
def draw_skeleton(canvas, keypoints_3d, color, path):
    origin = np.array([CANVAS_WIDTH // 2 - 100, CANVAS_HEIGHT // 2 - 100])
    points = keypoints_3d[:, :2]

    left_shoulder = points[mp_pose.PoseLandmark.LEFT_SHOULDER.value]
    right_shoulder = points[mp_pose.PoseLandmark.RIGHT_SHOULDER.value]
    current_width = np.linalg.norm(left_shoulder - right_shoulder) + 1e-6
    scale_factor = 200 / current_width
    scaled_points = points * scale_factor

    shoulder_center = (left_shoulder + right_shoulder) / 2.0
    scaled_points -= shoulder_center * scale_factor
    scaled_points += origin

    sleftElbow =
np.array(scaled_points[mp_pose.PoseLandmark.LEFT_ELBOW.value])
    sleftShoulder =
np.array(scaled_points[mp_pose.PoseLandmark.LEFT_SHOULDER.value])
```

```

    srightShoulder =
np.array(scaled_points[mp_pose.PoseLandmark.RIGHT_SHOULDER.value])
    sleftWrist =
np.array(scaled_points[mp_pose.PoseLandmark.LEFT_WRIST.value])

    #angle c
    ba = sleftShoulder - sleftElbow
    bc = sleftWrist - sleftElbow
    cosine_angle = np.dot(ba, bc) / (np.linalg.norm(ba) *
np.linalg.norm(bc))
    angle = np.degrees(np.arccos(cosine_angle))
    relbowANGC.append((path, angle))

    #angle d
    ba = sleftShoulder-srightShoulder
    bc = sleftElbow-sleftShoulder

    x1, y1 = srightShoulder[0], srightShoulder[1]
    x2, y2 = sleftShoulder[0], sleftShoulder[1]
    x3, y3 = sleftElbow[0], sleftElbow[1]

    slope1 = -1*(y2-y1)/(x2-x1)#*-1 because like y increases as it goes
down
    slope2 = -1*(y3-y2)/(x3-x2)
    print(f"slope1 {slope1} Slope2 {slope2}")

    cosine_angle = np.dot(ba, bc) / (np.linalg.norm(ba) *
np.linalg.norm(bc))
    angle2 = np.degrees(np.arccos(cosine_angle))
    if (slope2 < slope1):

        angle2 *= -1
    relbowlowANGD.append((path, angle2))

    i = 0
    for x, y in scaled_points[:25]:
        if (i==mp_pose.PoseLandmark.LEFT_ELBOW.value):
            cv2.putText(canvas, str(angle), (int(x),int(y)),
cv2.FONT_HERSHEY_SIMPLEX, 1, (255,255,255),1,1)
            cv2.circle(canvas, (int(x), int(y)), 14, color, 1)

```



```

        if (i==mp_pose.PoseLandmark.LEFT_SHOULDER.value):
            cv2.putText(canvas, str(angle2), (int(x),int(y)),
cv2.FONT_HERSHEY_SIMPLEX, 1, (0,0,255),1,1)
            cv2.circle(canvas, (int(x), int(y)), 14, color, 1)
            i += 1

        cv2.circle(canvas, (int(x), int(y)), 4, color, -1)

    for start_idx, end_idx in connections:
        if start_idx <= 24 and end_idx <= 24:
            p1 = scaled_points[start_idx]
            p2 = scaled_points[end_idx]
            cv2.line(canvas, tuple(p1.astype(int)), tuple(p2.astype(int)),
color, 3)

    return scaled_points[:25]

```

11/23

beepbooopdataset.ipynb

- created dataframe

Something is definitely wrong with my z value code or mediapipe's z value depth values. I am leaning towards mediapipe's z value not being accurate because

MediaPipe's **Z value** comes from its **machine learning model's regression output**, not from actual depth sensors or stereo vision. <- basically good to see if something is ahead of something else but not super accurate values across images



So here like the last image z value seems reasonable. The second one seems rilly off; it should be like closer to 0. The first one is off too it should be more than the third one. Not quite sure if I'm wrong right now or mediapipe is. Going to go exercise and then watch mission impossible with friends, I'll get on tonight and figure it out or see if there are other ways to check.

Not full code but basically all I have for z value

```
scaled_z = z_points * scale_factor
scaled_points = np.hstack([scaled_points, scaled_z[:, np.newaxis]])

sleftElbow = scaled_points[mp_pose.PoseLandmark.LEFT_ELBOW.value]
sleftShoulder = scaled_points[mp_pose.PoseLandmark.LEFT_SHOULDER.value]
srightShoulder = scaled_points[mp_pose.PoseLandmark.RIGHT_SHOULDER.value]
sleftWrist = scaled_points[mp_pose.PoseLandmark.LEFT_WRIST.value]


z_shoulder = sleftShoulder[2]
z_elbow = sleftElbow[2]
z_wrist = sleftWrist[2]

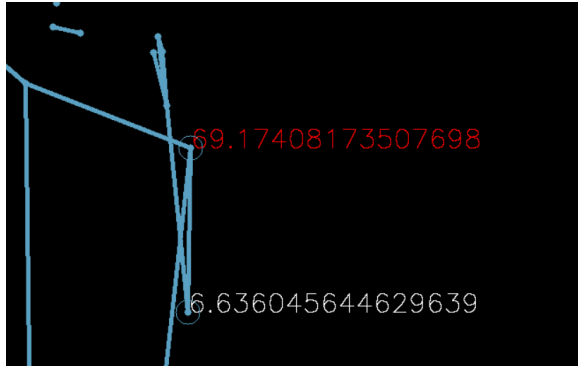

z_diff_elbow = z_elbow - z_shoulder
z_diff_wrist = z_wrist - z_shoulder
z_diff = min(z_diff_elbow, z_diff_wrist)
rarmZValues.append((path, z_diff))
```

i see

What is the most simple way for you to copy and paste it over?

Like if I just print the angle with a space in between 1 2 3 4 5?

Let me know if it matches up. Angle C is influenced by if its open or not I think
oh ye



probably not 6 degrees but it will give that. I'll try

something like the front facing thing rn

this is a probable way to deal with wrong Angle C

- imma just replace the incorrect values with a blank (no value)
- if we use XGBoost or a "tree-based" model then it ignores missing values

i can also verify if the z axis thing is accurate by seeing what values it leaves blank
okay

Right Elbow Angle C

```
('bbb\\IMG_2429.jpg', 58.57656221879531)
('bbb\\IMG_2430.jpg', 56.464979895970195)
('bbb\\IMG_2431.jpg', 57.1587105672319)
('bbb\\IMG_2432.jpg', 63.96606019944868)
('bbb\\IMG_2433.jpg', 60.26731364419235)
('bbb\\IMG_2434.jpg', 28.72595600750258)
('bbb\\IMG_2435.jpg', 114.15939481634749)
('bbb\\IMG_2436.jpg', 71.9966581389252)
('bbb\\IMG_2437.jpg', 59.088122402166924)
('bbb\\IMG_2438.jpg', 126.85802434701108)
('bbb\\IMG_2439.jpg', 18.271654795402632)
('bbb\\IMG_2440.jpg', 108.41184192378122)
('bbb\\IMG_2441.jpg', 6.636045644629639)
('bbb\\IMG_2442.jpg', 76.85546812475958)
('bbb\\IMG_2443.jpg', 114.20245366987506)
('bbb\\IMG_2444.jpg', 110.67248492987345)
('bbb\\IMG_2445.jpg', 128.38469910158872)
('bbb\\IMG_2446.jpg', 10.333352544055442)
('bbb\\IMG_2447.jpg', 3.484349327536794)
('bbb\\IMG_2448.jpg', 11.524311093403805)
('bbb\\IMG_2449.jpg', 40.68279220332538)
('bbb\\IMG_2450.jpg', 9.600156212214834)
('bbb\\IMG_2451.jpg', 59.76285029994273)
('bbb\\IMG_2452.jpg', 175.28230161992852)
('bbb\\IMG_9473.jpg', 38.33371238169897)
```

('bbb\\IMG_9474.jpg', 41.82473251776196)
('bbb\\IMG_9475.jpg', 42.43922241511418)
('bbb\\IMG_9476.jpg', 38.87103826949509)
('bbb\\IMG_9477.jpg', 31.986893091299272)
('bbb\\IMG_9478.jpg', 34.249596184221026)
('bbb\\IMG_9479.jpg', 36.47043877807168)
('bbb\\IMG_9480.jpg', 58.671382559721565)
('bbb\\IMG_9481.jpg', 55.04564412226081)
('bbb\\IMG_9482.jpg', 60.246410303386455)
('bbb\\IMG_9483.jpg', 38.61170967796264)
('bbb\\IMG_9484.jpg', 32.993779360156296)
('bbb\\IMG_9485.jpg', 34.962076211390894)
('bbb\\IMG_9486.jpg', 62.163698544488106)
('bbb\\IMG_9487.jpg', 60.83063634439706)
('bbb\\IMG_9488.jpg', 58.99765642008919)
('bbb\\IMG_9489.jpg', 76.47199997544638)
('bbb\\IMG_9490.jpg', 75.11354633705895)
('bbb\\IMG_9491.jpg', 57.38424662455953)
('bbb\\IMG_9492.jpg', 64.78241647349901)
('bbb\\IMG_9493.jpg', 61.14738086156535)
('bbb\\IMG_9494.jpg', 45.9735673983704)
('bbb\\IMG_9495.jpg', 44.522378849170934)
('bbb\\IMG_9496.jpg', 45.647404738956666)
('bbb\\IMG_9497.jpg', 48.272883212640714)
('bbb\\IMG_9498.jpg', 68.50021193597054)
('bbb\\IMG_9499.jpg', 67.137024173932)
('bbb\\IMG_9500.jpg', 69.7399347183792)
('bbb\\IMG_9501.jpg', 67.53789466717173)
('bbb\\IMG_9502.jpg', 62.11858712595392)
('bbb\\IMG_9503.jpg', 62.343053280151665)
('bbb\\IMG_9504.jpg', 62.359397466646286)
('bbb\\IMG_9505.jpg', 66.20494960225264)
('bbb\\IMG_9506.jpg', 66.4435514269494)
('bbb\\IMG_9507.jpg', 67.10344750591469)
('bbb\\IMG_9508.jpg', 66.22220487316426)
('bbb\\IMG_9509.jpg', 54.094087022335714)
('bbb\\IMG_9510.jpg', 52.6097372902878)
('bbb\\IMG_9511.jpg', 54.749846695208106)
('bbb\\IMG_9512.jpg', 54.041946579416)
('bbb\\IMG_9513.jpg', 45.90346652213378)
('bbb\\IMG_9514.jpg', 44.774746092012755)
('bbb\\IMG_9515.jpg', 41.96409649548871)
('bbb\\IMG_9516.jpg', 43.950622422575854)
('bbb\\IMG_9517.jpg', 55.945757365374625)

('bbb\\IMG_9518.jpg', 53.760609666125184)
('bbb\\IMG_9519.jpg', 54.26979517081429)
('bbb\\IMG_9520.jpg', 53.05013804046248)
('bbb\\IMG_9521.jpg', 70.6341688849127)
('bbb\\IMG_9522.jpg', 69.1527169909756)
('bbb\\IMG_9523.jpg', 72.7813743780907)
('bbb\\IMG_9524.jpg', 70.0844208452802)
('bbb\\IMG_9525.jpg', 103.3627260616751)
('bbb\\IMG_9526.jpg', 101.95581085976654)
('bbb\\IMG_9527.jpg', 100.92011520111143)
('bbb\\IMG_9528.jpg', 33.82599686074196)
('bbb\\IMG_9529.jpg', 36.986990096199804)
('bbb\\IMG_9530.jpg', 39.859122735057525)
('bbb\\IMG_9547.jpg', 83.63922154838623)
('bbb\\IMG_9548 (1).jpg', 84.97980028658306)
('bbb\\IMG_9549.jpg', 70.88240513226725)
('bbb\\IMG_9550.jpg', 63.0611753028886)
('bbb\\IMG_9551 (1).jpg', 69.56469535099504)
('bbb\\IMG_9552.jpg', 69.04847879571331)
('bbb\\IMG_9553.jpg', 82.74721238184445)
('bbb\\IMG_9554.jpg', 76.5474593612195)
('bbb\\IMG_9555.jpg', 58.808878037778165)
('bbb\\IMG_9556.jpg', 57.316000363443464)
('bbb\\IMG_9557.jpg', 65.99052245148633)
('bbb\\IMG_9558 (1).jpg', 65.7029673127874)
('bbb\\IMG_9567.jpg', 62.95887324873158)
('bbb\\IMG_9568.jpg', 61.14742330598467)
('bbb\\IMG_9569.jpg', 59.393457354822594)
('bbb\\IMG_9570.jpg', 24.68894806076685)
('bbb\\IMG_9571.jpg', 44.94483968172694)
('bbb\\IMG_9572.jpg', 101.22920737228193)
('bbb\\IMG_9573.jpg', 166.6811320115164)
('bbb\\IMG_9574.jpg', 144.2244827846264)
('bbb\\IMG_9576.jpg', 70.86589761561005)
('bbb\\IMG_9577.jpg', 73.03579841174158)
('bbb\\IMG_9578.jpg', 178.8265477159003)
('bbb\\IMG_9579.jpg', 179.7593657998347)
('bbb\\IMG_9580.jpg', 178.41158835095476)
('bbb\\IMG_9581.jpg', 61.70100058255745)
('bbb\\IMG_9582.jpg', 75.81658439570339)
('bbb\\IMG_9583.jpg', 95.72281418791816)
('bbb\\IMG_9584.jpg', 90.2470438479359)
('bbb\\IMG_9585.jpg', 13.046800898372076)
('bbb\\IMG_9586.jpg', 52.47048137211558)

('bbb\\IMG_9587.jpg', 50.50703090615055)
('bbb\\IMG_9588.jpg', 73.82206224875866)
('bbb\\IMG_9589.jpg', 92.30759722963425)
('bbb\\IMG_9590.jpg', 50.86086489395192)
('bbb\\IMG_9591.jpg', 46.189521962234565)

Right Elbow Angle D

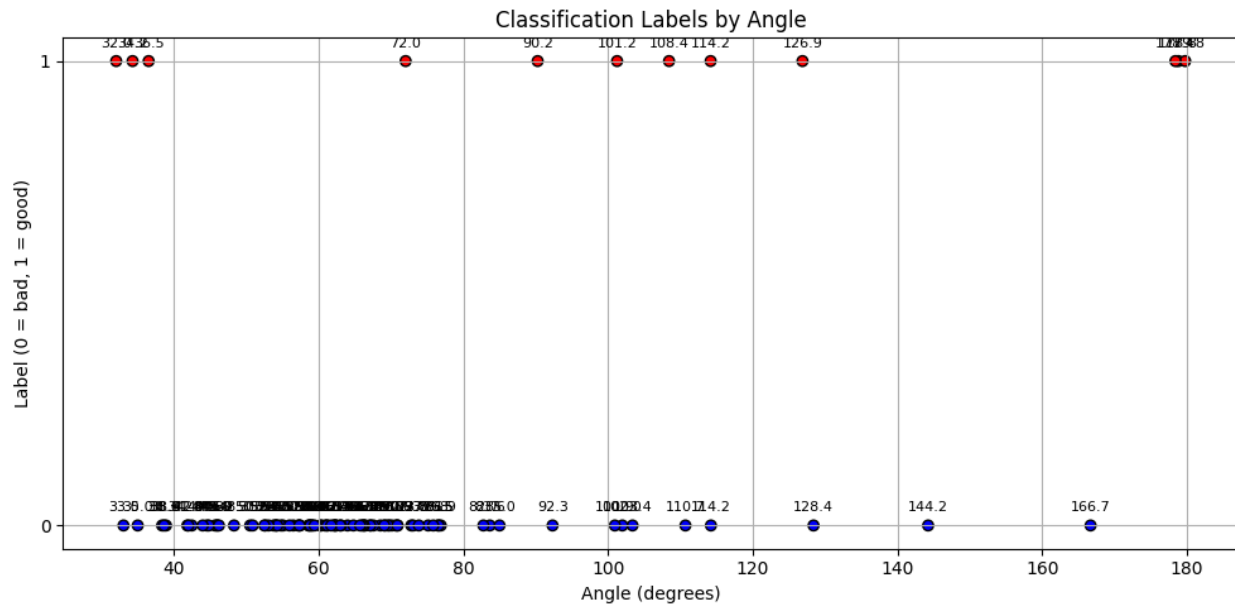
('bbb\\IMG_2429.jpg', 4.643728388337567)
('bbb\\IMG_2430.jpg', 7.743867519528239)
('bbb\\IMG_2431.jpg', 9.08527164141838)
('bbb\\IMG_2432.jpg', -0.29738724263919103)
('bbb\\IMG_2433.jpg', -5.465512317209335)
('bbb\\IMG_2434.jpg', -30.29124649229814)
('bbb\\IMG_2435.jpg', -3.936242334931494)
('bbb\\IMG_2436.jpg', -66.85146018505792)
('bbb\\IMG_2437.jpg', 10.565531467330906)
('bbb\\IMG_2438.jpg', 7.305090088947545)
('bbb\\IMG_2439.jpg', -36.86315487550854)
('bbb\\IMG_2440.jpg', 14.746402619564211)
('bbb\\IMG_2441.jpg', 69.17408173507698)
('bbb\\IMG_2442.jpg', -6.574414037799502)
('bbb\\IMG_2443.jpg', -65.5102855389984)
('bbb\\IMG_2444.jpg', 28.79764308082964)
('bbb\\IMG_2445.jpg', 53.573965277693)
('bbb\\IMG_2446.jpg', -32.05067846853882)
('bbb\\IMG_2447.jpg', -40.40315318153763)
('bbb\\IMG_2448.jpg', -31.322020530220833)
('bbb\\IMG_2449.jpg', -72.33626407040344)
('bbb\\IMG_2450.jpg', -52.77035285293556)
('bbb\\IMG_2451.jpg', 16.121753941321483)
('bbb\\IMG_2452.jpg', -136.48646383818743)
('bbb\\IMG_9473.jpg', -12.08510556336189)
('bbb\\IMG_9474.jpg', -8.484122819441327)
('bbb\\IMG_9475.jpg', -6.673246850860677)
('bbb\\IMG_9476.jpg', -7.332837671782191)
('bbb\\IMG_9477.jpg', -36.15137789377519)
('bbb\\IMG_9478.jpg', -33.697704149512276)
('bbb\\IMG_9479.jpg', -28.933005083939857)
('bbb\\IMG_9480.jpg', -22.007342455520046)
('bbb\\IMG_9481.jpg', -24.88635358986734)
('bbb\\IMG_9482.jpg', -20.59421560840738)
('bbb\\IMG_9483.jpg', -35.70892944585118)

('bbb\\IMG_9484.jpg', -32.71584137507628)
('bbb\\IMG_9485.jpg', -32.11771552348507)
('bbb\\IMG_9486.jpg', -34.00636603631033)
('bbb\\IMG_9487.jpg', -36.236136548247075)
('bbb\\IMG_9488.jpg', -37.0915015064897)
('bbb\\IMG_9489.jpg', -33.26474678958514)
('bbb\\IMG_9490.jpg', -33.13659252214208)
('bbb\\IMG_9491.jpg', -34.089571385034475)
('bbb\\IMG_9492.jpg', -30.433871089993794)
('bbb\\IMG_9493.jpg', -33.72749240591516)
('bbb\\IMG_9494.jpg', -54.85300358496177)
('bbb\\IMG_9495.jpg', -55.30547043537291)
('bbb\\IMG_9496.jpg', -56.198925132382094)
('bbb\\IMG_9497.jpg', -52.61066899537182)
('bbb\\IMG_9498.jpg', -14.751233705651524)
('bbb\\IMG_9499.jpg', -16.433963026002736)
('bbb\\IMG_9500.jpg', -12.977399740250252)
('bbb\\IMG_9501.jpg', -19.44871341628047)
('bbb\\IMG_9502.jpg', -23.042151360240243)
('bbb\\IMG_9503.jpg', -21.636624730359042)
('bbb\\IMG_9504.jpg', -22.420620228671684)
('bbb\\IMG_9505.jpg', -3.2635773937819765)
('bbb\\IMG_9506.jpg', -5.651817185424136)
('bbb\\IMG_9507.jpg', -2.753471502815845)
('bbb\\IMG_9508.jpg', -3.8121214882616536)
('bbb\\IMG_9509.jpg', -26.205841649254836)
('bbb\\IMG_9510.jpg', -23.63581839582564)
('bbb\\IMG_9511.jpg', -23.118314382561646)
('bbb\\IMG_9512.jpg', -22.938313261785844)
('bbb\\IMG_9513.jpg', -36.05354723350469)
('bbb\\IMG_9514.jpg', -38.77126239844558)
('bbb\\IMG_9515.jpg', -39.88590291420077)
('bbb\\IMG_9516.jpg', -37.11500235809945)
('bbb\\IMG_9517.jpg', -17.69788419957331)
('bbb\\IMG_9518.jpg', -21.413713122626607)
('bbb\\IMG_9519.jpg', -21.526997593158786)
('bbb\\IMG_9520.jpg', -22.817017567919233)
('bbb\\IMG_9521.jpg', 0.781726069580847)
('bbb\\IMG_9522.jpg', 0.5405049730371262)
('bbb\\IMG_9523.jpg', 2.0940234014197063)
('bbb\\IMG_9524.jpg', -3.0668760430492013)
('bbb\\IMG_9525.jpg', 20.879615424853867)
('bbb\\IMG_9526.jpg', 15.832601774230172)
('bbb\\IMG_9527.jpg', 17.282937568218323)

('bbb\\IMG_9528.jpg', -25.107305130156448)
('bbb\\IMG_9529.jpg', -20.127743710052275)
('bbb\\IMG_9530.jpg', -21.256029040649878)
('bbb\\IMG_9547.jpg', -7.339425927811699)
('bbb\\IMG_9548 (1).jpg', -6.112797956275976)
('bbb\\IMG_9549.jpg', -72.50073373017183)
('bbb\\IMG_9550.jpg', -72.85033798063809)
('bbb\\IMG_9551 (1).jpg', -24.63562513034915)
('bbb\\IMG_9552.jpg', -22.971962435943997)
('bbb\\IMG_9553.jpg', -14.41495656914183)
('bbb\\IMG_9554.jpg', -18.13408570512549)
('bbb\\IMG_9555.jpg', -54.340477496238)
('bbb\\IMG_9556.jpg', -53.00411776297623)
('bbb\\IMG_9557.jpg', -53.169861978345615)
('bbb\\IMG_9558 (1).jpg', -56.98517711273162)
('bbb\\IMG_9567.jpg', -1.3685455363676517)
('bbb\\IMG_9568.jpg', -10.259872218503297)
('bbb\\IMG_9569.jpg', -5.941691698494493)
('bbb\\IMG_9570.jpg', -42.228645361537126)
('bbb\\IMG_9571.jpg', -17.808812668792974)
('bbb\\IMG_9572.jpg', 8.100782614480442)
('bbb\\IMG_9573.jpg', -47.17541016707047)
('bbb\\IMG_9574.jpg', -53.477509394581126)
('bbb\\IMG_9576.jpg', -10.873611955609286)
('bbb\\IMG_9577.jpg', -2.5183416747662)
('bbb\\IMG_9578.jpg', 68.51004794249272)
('bbb\\IMG_9579.jpg', 30.467971123197476)
('bbb\\IMG_9580.jpg', -7.186201148978147)
('bbb\\IMG_9581.jpg', -57.10336689551811)
('bbb\\IMG_9582.jpg', -8.04770297670894)
('bbb\\IMG_9583.jpg', -66.4014861964089)
('bbb\\IMG_9584.jpg', -23.300151320850006)
('bbb\\IMG_9585.jpg', -46.96927805451133)
('bbb\\IMG_9586.jpg', -27.884012133084944)
('bbb\\IMG_9587.jpg', -26.88377470291774)
('bbb\\IMG_9588.jpg', -4.314544658716122)
('bbb\\IMG_9589.jpg', 17.0527126282053)
('bbb\\IMG_9590.jpg', -20.046874986534707)
('bbb\\IMG_9591.jpg', -29.13542715968362)

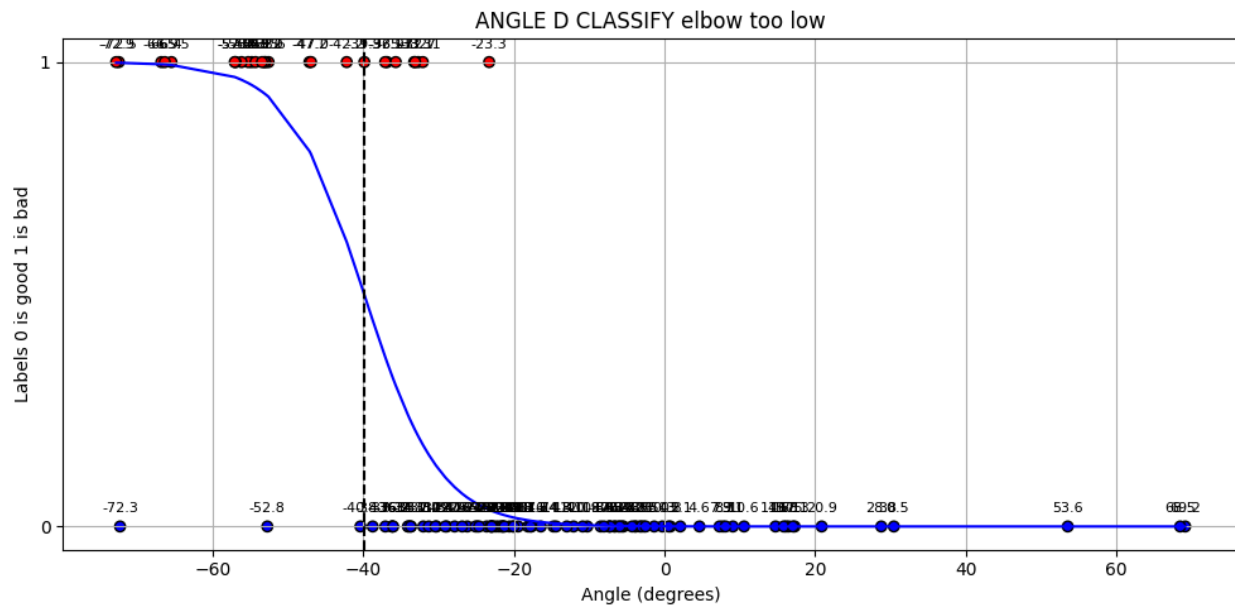
splitting into two:

- trying ML classification on Angles C and D
- maybe going to do a pre-trained CNN to detect whether the arm is opened or closed



11/29

I tried to classify Angle D if elbow is too low with current data and it works. Could see some of the outliers here and relabel if necessary.



I finished my supervised learning course in coursera and decided to try and apply “knowledge”. I don't think we need scikit learn for other angles too but the arm not open is still a big problem. It was sort of fun to write the functions to learn the topic better and not just be reliant on chat gpt entirely lol

```
Cutoff Angle [-39.90636683]
Iteration 9999: Cost      0.25
```

Anglec data is sort of messed up so some need to be relabeled to try the same thing. Angle c needs more of a quadratic model too. I haven't checked how to do that.

<https://www.youtube.com/watch?v=fSJj16UM990>

Learn basics of react

Then use other sources to put realtime camera in react

<https://medium.com/@dev-john-nguyen/integrating-mediapipe-hand-detection-in-react-native-f57c46aaab3a>

Maybe this idk.

Racket arm opened
Racket arm opened
Racket arm opened
Racket arm opened
Racket arm opened
Racket arm not opened
Racket arm opened
Racket arm opened
Racket arm opened
Racket arm opened
Racket arm not opened
Racket arm opened
Racket arm not opened
Racket arm opened
Racket arm not opened
Racket arm opened
Racket arm opened
Racket arm not opened
Racket arm not opened
Racket arm not opened
Racket arm not opened
Racket arm not opened
Racket arm opened
Racket arm not opened
Racket arm opened

[illegible]

Racket arm opened
Racket arm opened
Racket arm opened
Racket arm opened
Racket arm opened
Racket arm opened
Racket arm opened
Racket arm opened
Racket arm opened
Racket arm opened
Racket arm opened
Racket arm opened
Racket arm opened
Racket arm opened
Racket arm opened
Racket arm opened
Racket arm opened
Racket arm opened
Racket arm opened
Racket arm opened
Racket arm opened
Racket arm opened
Racket arm not opened
Racket arm not opened
Racket arm opened
Racket arm not opened
Racket arm not opened
Racket arm opened
Racket arm opened
Racket arm opened
Racket arm opened
Racket arm opened
Racket arm not opened
Racket arm opened
Racket arm not opened
Racket arm opened
Racket arm not opened
Racket arm opened
Racket arm opened
Racket arm opened
Racket arm opened

Racket arm opened
Racket arm opened

12/01

 openedOrClosed.ipynb im putting a lot of comments next to the lines of code to explain to myself what it does

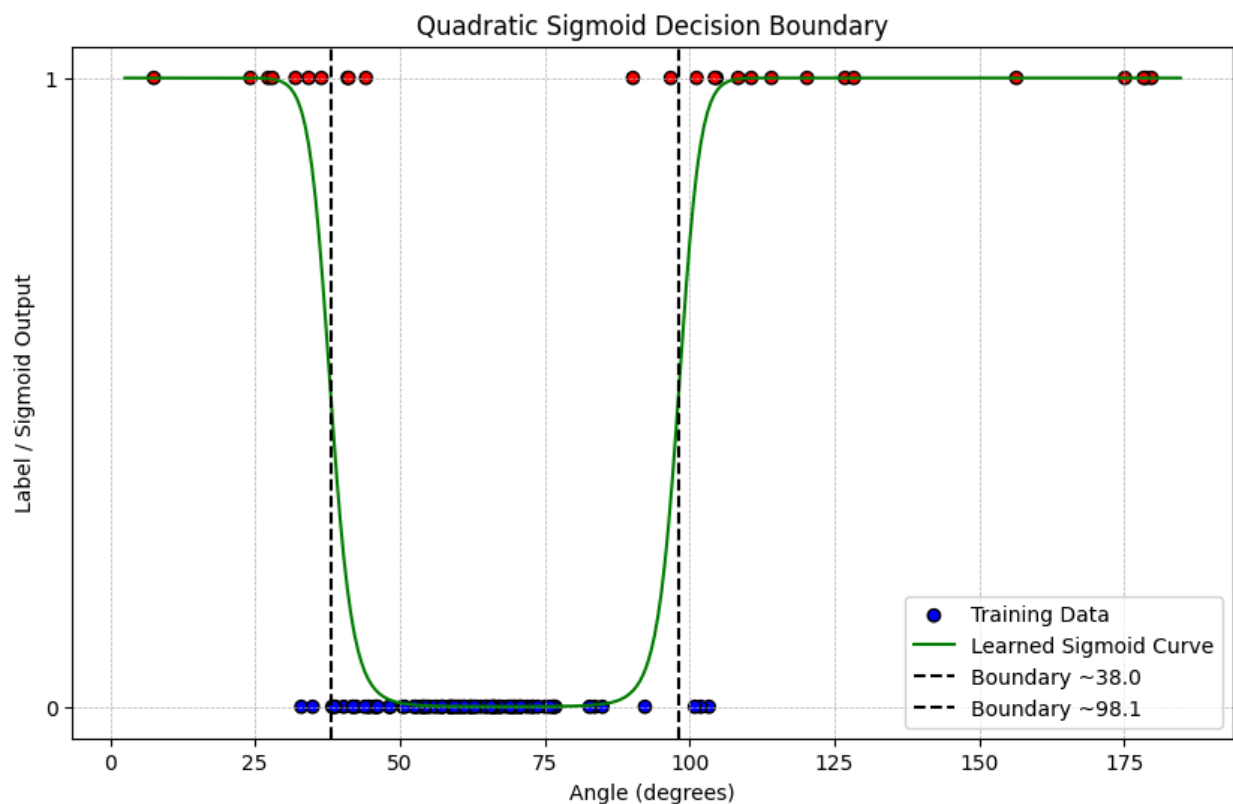
Thanks. I will make a google sheet later for the labels. Too chaotic with stuff everywhere lol.
I'll watch a cnn video sometime to get a jist of what is happening because i won't be starting or finishing the coursera one that soon.

I'm using PyTorch to build this CNN—I used Tensorflow and Keras for my generative AI model last time so it is a little bit different
Okie

<https://docs.google.com/spreadsheets/d/1kunlqvag6vXkwClYrXMOI8vKZKrQTVlrXMRTmVU4Izw/edit?usp=sharing>

I labeled my new images. I'll relabel the mislabeled ones today and try classification again.

If you copy you might have to get rid of the formatting its in a table format



OMG ok im done today lol.
Things to try tomorrow:

- Some arm too low angles are mislabelled again. But we already preformed classification and got a reasonable angle. I won't be doing it again and labeling again for now.
- Try a test with a mediapipe. Plug in the trained angles and see if it does some cool stuff
- Cnn
- Bro that thumbs up guy ruined the dataset for like 20 minutes. I couldn't figure out why it was not aligned by like 4 images.

12/08

Too high 16.7

Too low -39.3

Angle C (not good): less than 35 or above 98

HAVE TO START THINKING ABOUT UI / UX

▶ Sharing My AI-Powered Coding Workflow To Build Apps REALLY Fast
magicpatterns.com

- AI-powered UI generation

Cursor AI

- AI-powered code editor
- when at the home page click "Import from VS Code"

📁 Dataset for Beep Boop Bloop

ok hi i concatenated the new image folders and now there is a tuple problem but I literally can't work on it anymore today I'll continue tomorrow

- tried making the label into a list and that didn't work either
- goodbye i am going to do video stuff now

Can you send an img of the problem later. Also I really don't get this app dev stuff at all yet lol. I'm trying to figure that out.

12/11

trying to figure out CNN connection to mediapipe...

notes to kind of organize my head:

- CNN class we created has to be put into a SINGLE & SEPARATE python file
 - ex: [model.py](#) on VSCode
- also copy the .pth file we downloaded after training the model (store it in the same directory?)
- PyTorch (2.6.0+cu124), torchvision, and Pillow (PIL) installed
 - pip install torch==2.6.0+cu124 torchvision pillow
 - might need to adjust it based on CUDA or CPU-build
- loading the model

```
from model import openedOrClosedCNN

# load it:
model = openedOrClosedCNN()
state = torch.load("racket_arm_cnn.pth", map_location=device)
model.load_state_dict(state)
model.eval()
```

I ditched expo and now im just starting with react-native setting up a basic project first. And then hopefully can easily copy page code that I wrote last week without that much issue. I had too many conflicting dependencies or something.

12/13

ok i fixed it!!!

CODE: [NEW_openOrClosed.ipynb](#)

```
IMG_9555.jpg      → predicted: 1, actual: 1
IMG_9525.jpg      → predicted: 1, actual: 1
IMG_2429.jpg      → predicted: 1, actual: 1
IMG_2724.jpg      → predicted: 1, actual: 1
IMG_2735.jpg      → predicted: 1, actual: 1
IMG_9512.jpg      → predicted: 1, actual: 1
IMG_9516.jpg      → predicted: 1, actual: 1
IMG_9489.jpg      → predicted: 1, actual: 1
IMG_9476.jpg      → predicted: 1, actual: 1
IMG_9556.jpg      → predicted: 1, actual: 1
IMG_9517.jpg      → predicted: 1, actual: 1
IMG_2709.jpg      → predicted: 0, actual: 0
IMG_2729.jpg      → predicted: 1, actual: 1
IMG_9504.jpg      → predicted: 1, actual: 1
IMG_0077.jpg      → predicted: 1, actual: 1
IMG_9491.jpg      → predicted: 1, actual: 1
IMG_2726.jpg      → predicted: 1, actual: 1
IMG_9526.jpg      → predicted: 1, actual: 1
IMG_2452.jpg      → predicted: 0, actual: 0
IMG_9553.jpg      → predicted: 1, actual: 1
IMG_9496.jpg      → predicted: 1, actual: 1
IMG_9495.jpg      → predicted: 1, actual: 1
IMG_9502.jpg      → predicted: 1, actual: 1
IMG_9480.jpg      → predicted: 1, actual: 1
IMG_2720.jpg      → predicted: 0, actual: 1
IMG_2725.jpg      → predicted: 1, actual: 1
IMG_0076.jpg      → predicted: 1, actual: 1
IMG_9482.jpg      → predicted: 1, actual: 1
IMG_9499.jpg      → predicted: 1, actual: 1
IMG_2430.jpg      → predicted: 1, actual: 1
IMG_9589.jpg      → predicted: 1, actual: 1
IMG_9506.jpg      → predicted: 1, actual: 1
IMG_2721.jpg      → predicted: 0, actual: 1
IMG_9588.jpg      → predicted: 1, actual: 1
IMG_9573.jpg      → predicted: 0, actual: 0
Validation Accuracy: 94.29%
```

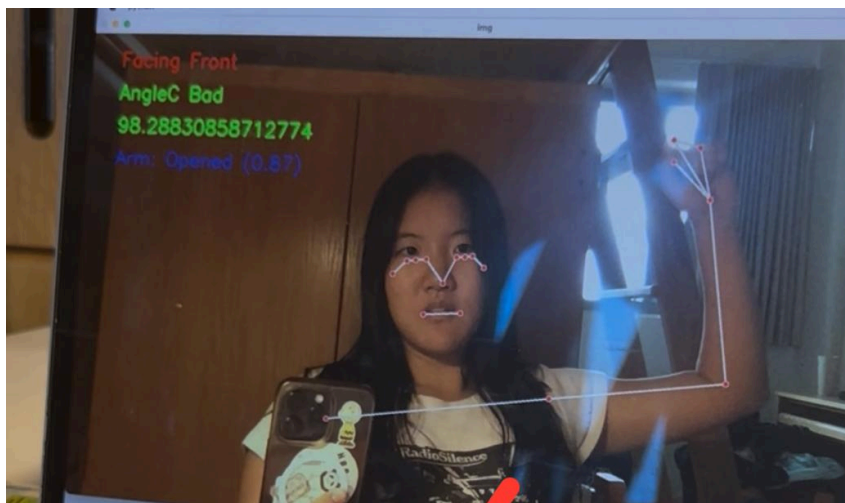
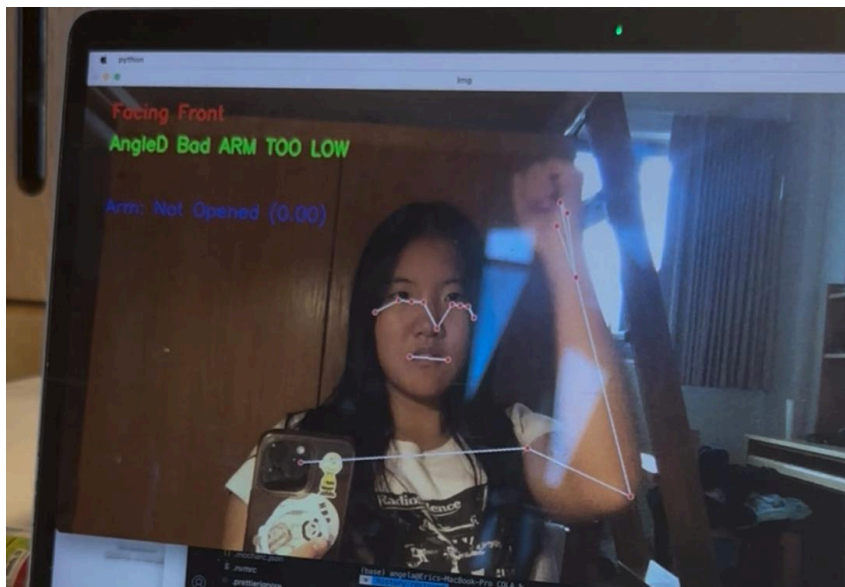
IMG_2720.jpg & IMG_2721.jpg were the ones that were incorrect (predicted not opened, but was opened) but i think IMG_2721 is supposed to say not opened

📺 How I Built a Full App in ONE Week Using AI (My Actual Workflow + Advice)

- should we lowkey try cursor 😭

- Yes i am going to crashout i got nothing done in the last two days

12/20



blue text outputs “Opened” or “Not Opened” along with a probability

- there needs to be a toggle on the app so the user can select “Left Hand” or “Right Hand”

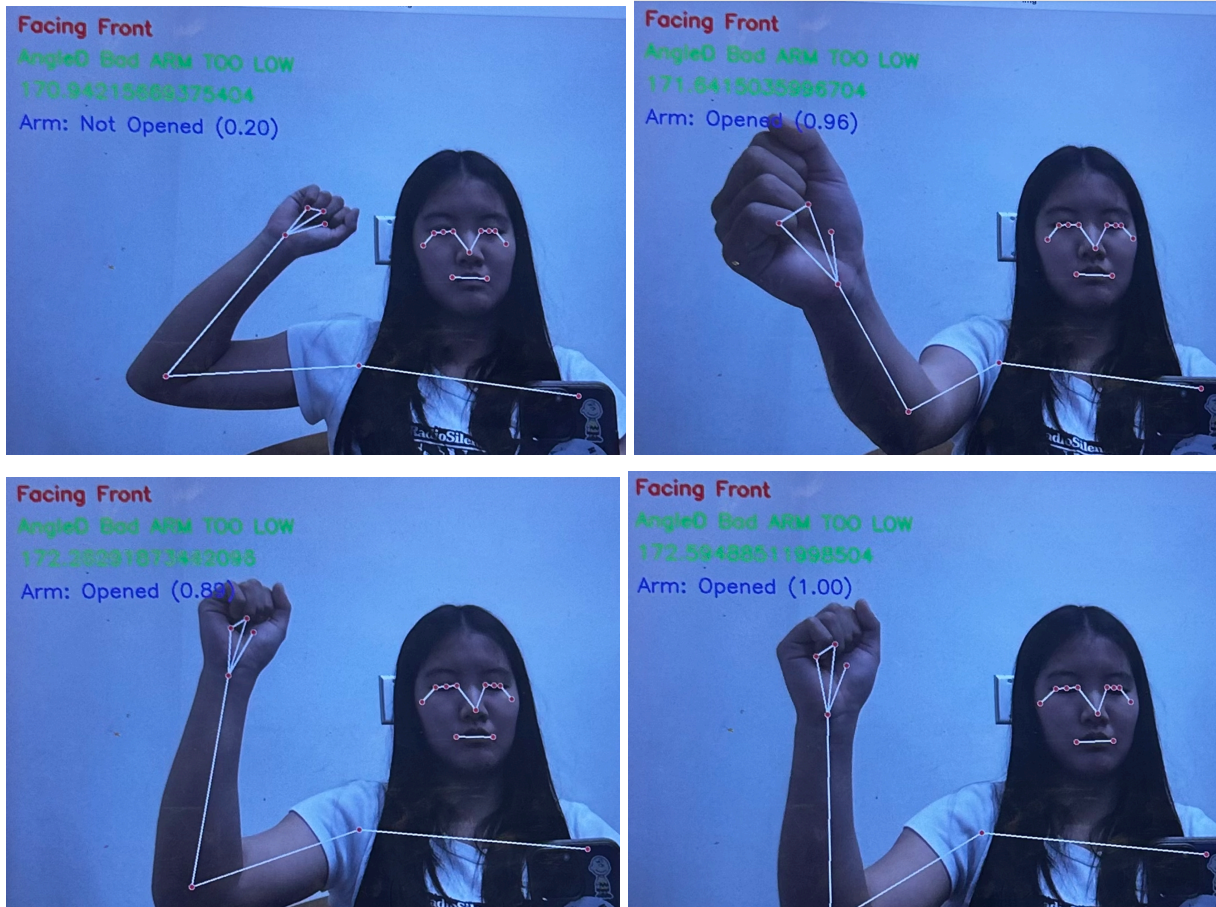
regarding mirroring:

cropped = cv2.flip(cropped, 1)

- reflects the image so the right arm appears as the left arm?
- ok so that doesn't work
- no it does work i did it wrong but it kind of works

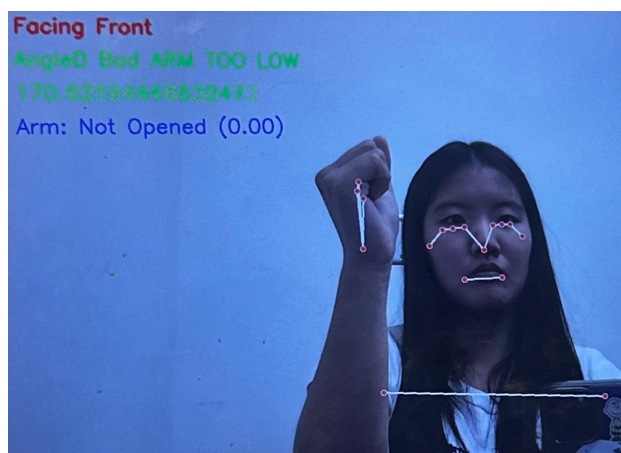
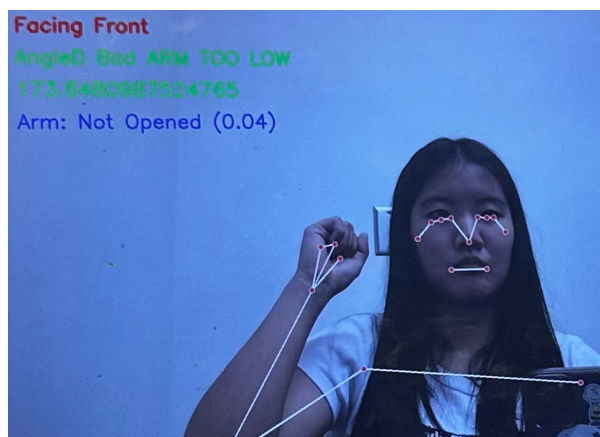
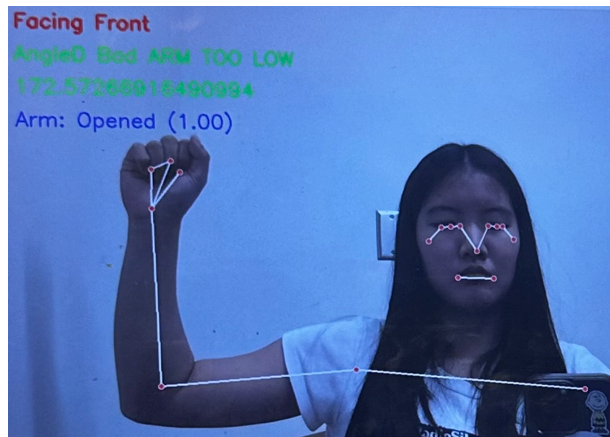
CASES THAT **AREN'T** CLASSIFYING CORRECTLY:

- we are going to have to check the labels of the data again 🤔



- model is definitely thinking the angle between upper arm and forearm (angle D?) is connected to the arm being opened or closed (look at upper left corner)
- need to look at data and labels again and make new data
- Ok. wait like sorry i dont get.

CASES THAT ARE CLASSIFYING CORRECTLY:



01/02

New images labeled

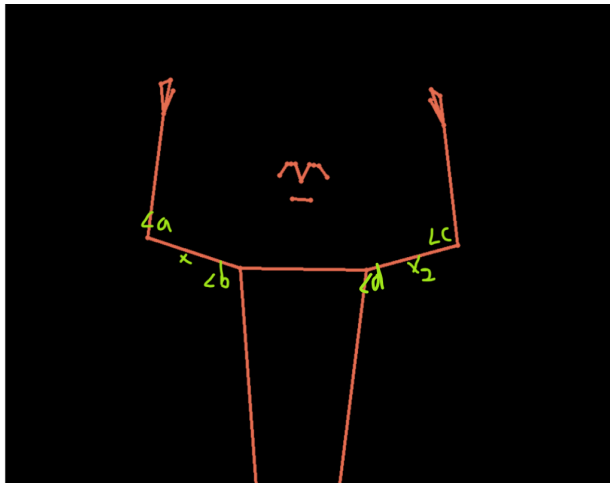
Ill be testing more on my moms phone rn adding in the other angle

I will have to reclassify angle d too low

-39.3 is way too much

01/04

I have 4 angles working. I made it do the right arm as a dominant hand not flipped like all our images.



Angles Abcd

I'll try to make a left-right hand option another day and fix a problem where you can't rotate camera sideways and c an only use it vertically. Other than that yeah.

01/08

updated CNN

```
Epoch 1, Loss: 6.5302
Epoch 2, Loss: 3.9096
Epoch 3, Loss: 1.9696
Epoch 4, Loss: 1.3275
Epoch 5, Loss: 0.7569
Epoch 6, Loss: 0.5192
Epoch 7, Loss: 0.2828
Epoch 8, Loss: 0.2448
Epoch 9, Loss: 0.1341
Epoch 10, Loss: 0.1373
Epoch 11, Loss: 0.0673
Epoch 12, Loss: 0.0719
Epoch 13, Loss: 0.0593
Epoch 14, Loss: 0.0397
Epoch 15, Loss: 0.0411
Epoch 16, Loss: 0.0433
Epoch 17, Loss: 0.0221
Epoch 18, Loss: 0.0326
Epoch 19, Loss: 0.0194
Epoch 20, Loss: 0.0193
```

loss is converging

I just saw this lol.

01/10

In node modules you should have react-native-human-pose put this in index.tsx

After clone idk if need npm or npx install. There are some other stuff too to get it working.

There are some commands here that could help. I did the android ones maybe you need to run the ios ones

<https://medium.com/@hs913271/human-pose-detection-in-react-native-develop-a-mobile-application-to-count-squats-in-real-time-6096b29330f9>

If you pull from github, pull from yes-new branch for now. IDK why when I commit it sends there atm, will fix soon.

I think this will work.

Git checkout yes-new

Git pull origin yes-new

```
import React from 'react';
import { WebView } from 'react-native-webview';
import { Dimensions } from 'react-native';

interface HumanPoseProps {
  width?: number;
  height?: number;
  enableSkeleton?: boolean;
  enableKeyPoints?: boolean;
  color?: string;
  mode?: 'single' | 'multiple';
  scoreThreshold?: number;
  isBackCamera?: boolean;
  flipHorizontal?: boolean;
  onPoseDetected?: (pose: any) => void;
  armProb?: (prob: any) => void;
  isFullScreen?: boolean;
  model?: 'blazePose' | 'poseNet';
}

export default function HumanPose(p: HumanPoseProps) {
  /**
   * Callback function that is invoked when a pose is detected by the
   * WebView.
   * @param pose - The pose data received from the WebView.
   */
}
```

```

    * @param prob
    */
const onPoseDetected = (pose: any) => {
    if (p.onPoseDetected) {
        p.onPoseDetected(pose);
    }
};

const armProb = (prob: any) =>{
    if (p.armProb){
        p.armProb(prob);
    }
}

// URLs for the different pose detection models hosted on the web.
const blazePoseUrl = 'https://beepose.vercel.app/';
const poseNetUrl = 'https://react-native-human-pose.web.app/';

// Determine the correct URL based on the selected model prop.
const modelUrl = p.model === 'blazePose' ? blazePoseUrl : poseNetUrl;

// Construct the query parameters to pass to the web app.
const queryParams = new URLSearchParams({
    enableSkeleton: String(p.enableSkeleton ?? false),
    enableKeyPoints: String(p.enableKeyPoints ?? false),
    color: p.color || '',
    mode: p.mode || '',
    scoreThreshold: String(p.scoreThreshold || ''),
    isBackCamera: String(p.isBackCamera ?? false),
    flipHorizontal: String(p.flipHorizontal ?? false),
    isFullScreen: String(p.isFullScreen ?? false),
}).toString();

const finalUri = `${modelUrl}?${queryParams}`;

return (
    <WebView
        source={{ uri: finalUri }}
        style={{
            width: p.width ?? 200,

```

```

        height: p.height ?? 200,
        borderRadius: 5,
    }}

    mediaPlaybackRequiresUserGesture={false}

    // Allows the camera video to play directly within the WebView on
iOS.
    allowsInlineMediaPlayback={true}

    // Standard props to enable necessary web features.
    javaScriptEnabled={true}
    domStorageEnabled={true}

    // This prop was removed in newer versions of react-native-webview.
    // Navigation can be controlled with `onShouldStartLoadWithRequest`
if needed,
    // but for this component's simple use case, it can be safely
removed.
    // limitsNavigationsToAppBoundDomains={true}

    /**
     * A message is sent from the WebView when a pose is detected.
     * This handler parses the message and calls the onPoseDetected
callback.
     */
    onMessage={event => {
      try {
        const msg = JSON.parse(event.nativeEvent.data);
        //console.log("bruh" + msg.type)
        if (msg.type==='cnn'){
          console.log("CNN PROBABILITY: "+msg.armProb)
          armProb(msg.armProb)
        }
        else if (msg.type == 'landmarks'){

          onPoseDetected(msg.poseLandmarks)
        }
        else{
          console.log("Unknown json message")

```

```

    }
  } catch (e) {
    console.error('Failed to parse pose data:', e);
  }
}
// Added an error handler for better debugging.
onError={ (syntheticEvent) => {
  const { nativeEvent } = syntheticEvent;
  console.warn('WebView error: ', nativeEvent);
}}
/>
);
}

```

01/13

In ios/<projectfoldername>/Info.plist

add

```

<key>NSCameraUsageDescription</key>
<string>$(PRODUCT_NAME) needs access to your Camera.</string>

```

01/15

Run before each time coding.

git checkout -b feature/camera

git pull origin main

git checkout -b feature/homepage

git add .

git commit -m "angela UI + max camera"

git push -u origin feature/homepage

Ignore

Keeps breaking

Copy beepboopblop1 folder to new xcode proj delete all else.

Info.plist

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE plist PUBLIC "-//Apple//DTD PLIST 1.0//EN" "http://www.apple.com/DTDs/PropertyList-1.0.dtd">
<plist version="1.0">
<dict>
    <key>NSCameraUsageDescription</key>
    <string>$(PRODUCT_NAME) needs access to your Camera.</string>
    <key>NSMicrophoneUsageDescription</key>
    <string>$(PRODUCT_NAME) needs access to your Camera.</string>
    <key>UIBackgroundModes</key>
    <array>
        <string>remote-notification</string>
    </array>
</dict>
</plist>
```