Yoga Pose Detection with Similarity Scoring

Final Project

CSCE 5222

Feature Engineering

Dr. Yuan

Yingsuan Lo

Jongwook Yoon

R. Cooper Snyder

Problem statement

Have you ever experienced attending a physical workout session without fully understanding the movement instruction yet not brave enough to ask the trainer's attention? Have you ever tried to follow some remote class due to the pandemic, but it is hard to grasp the trainer's movement correctly through the screen? Have you heard of some great trainers but always too far from you and remote classes always frustrate you? Have you ever tried to start a workout session, but it's hard to follow along because of laziness, and you really need a real trainer? Have you ever finally insisted on work-out by some app, but not knowing which direction or which strength of movements can lead you to the next level?

Here is our solution to the problems above -- we have provided a prototype that connects you and trainers directly, facilitating the productivity of in person training by analyzing the similarity between your yoga pose and the trainer's pose. On the platform screen, you can see both you and your trainer's pose analysis and see where they differ in order for you to fix your posture. The system will automatically give you some suggestions when you cannot reach the standard pose that your instructors perform so that you can build up your confidence and have a goal to work toward immediately. This project could be expanded into a phone application platform that trainers and yoga enthusiasts can use around the world to assist in better self led sessions or better crafted group sessions by leveraging the database of all users' uploaded poses.

Method and Implementation

The core problem for pose estimation is keypoint detection and tracking. Mediapipe's BlazePlaze model, developed by Google offers state of the art, real time pose detection and tracking. It detects and tracks 33 different 3D key points or landmarks on the body which can be used for a variety of use cases. In our case, we are using the landmarks to calculate the euclidean differences between a yoga instructor's gold standard poses and the poses of our group members and willing participant volunteers, both in still images and through time via videos of poses since the transition between poses matters just as much as being able to hold certain yoga poses.

MediaPipe's BlazePose is packaged in various different software libraries for versatile development. The solution offers packages in Python, JavaScript, and C++. The library we used in this project is the Python package, 'mediapipe' which can be installed with the command 'pip install mediapipe' into a Python virtual environment. From there you can call the different models offered, such as the Face Detection, Face Mesh, Hands, Holistic, Objectron, Pose and Selfie Segmentation models. We obviously chose to use the Pose model.

When called upon to process an image, the Pose model builds a Landmarks object which stores the 3D coordinates of each of the 33 keypoints. At this moment, the Z coordinate is experimental, and is not a 'true' Z coordinate. It is estimated by assigning the origin approximately at the center point between the hips, and choosing the Z axis as perpendicular to the camera. Positive Z coordinates are towards the camera, positive values and away from it.

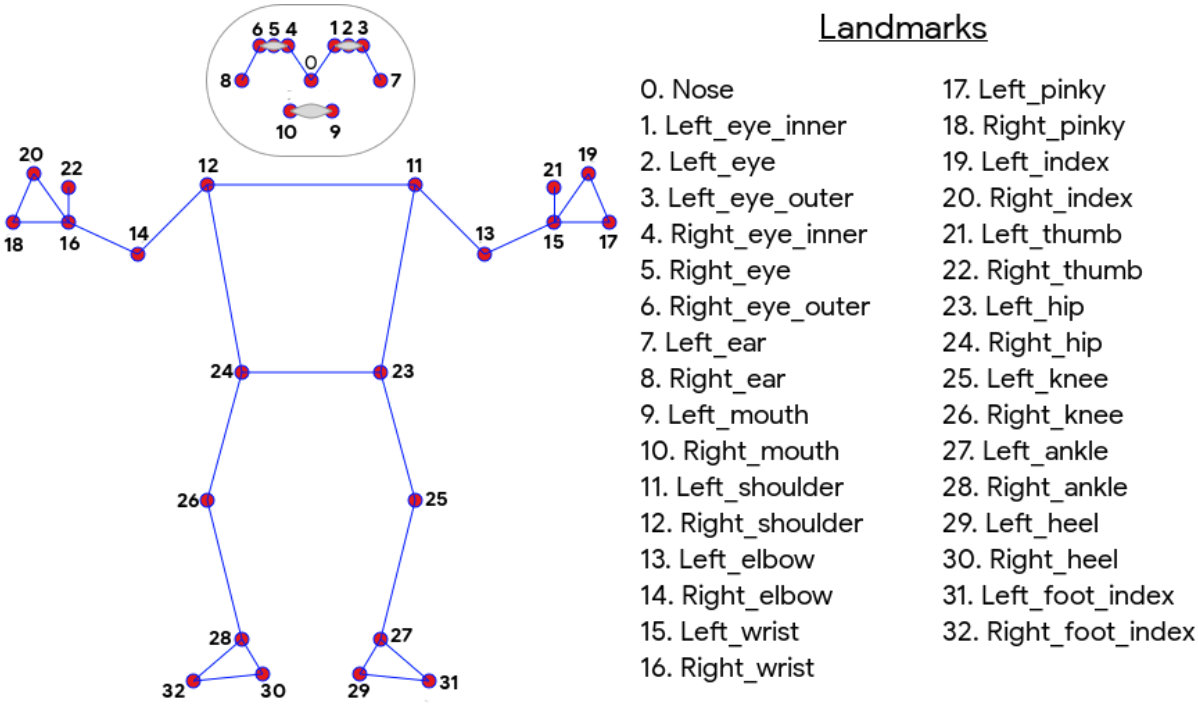


Figure 1: Landmarks

For our use case, we normalized the poses extracted to have the same torso size and vertical torso orientation by dividing those coordinates by the image width and image height. This resulted in landmark coordinates between 0 and 1.0. This does not solve all of the image similarity problems though. One source of a problem was the fact that our raw images and videos were taken with variable lengths between the subject and the camera. Additionally, the cameras used were different phones which all have different field of view angles. Additionally, when cropping the images, the distance between the subjects' head or feet and the edge of the image varies. To remedy this, we attempted multiple approaches to segment the humans from the background to only have the humans in the image.

Experimental Results

Description of the data used

The data we used as the gold standard yoga instructors came from two videos of beginner yoga tutorials on Youtube listed here.

- https://youtu.be/6hZIzMpHl-c
- https://youtu.be/qq2-5waRsVA

We extracted 6 short clips, three from each video, of three different yoga poses and transitions.
The short video clips can be downloaded and viewed at this Github URL:

- https://github.com/LoriSchuan-dev/feature_engineering_project2021
- Videos are listed under the 'pose_1_3' and 'pose_4_6' directories.

From those short clips, we extracted screen shots of the final poses to build an instructor set of still images of 6 different poses listed below.

Gold Standard Instructor Images

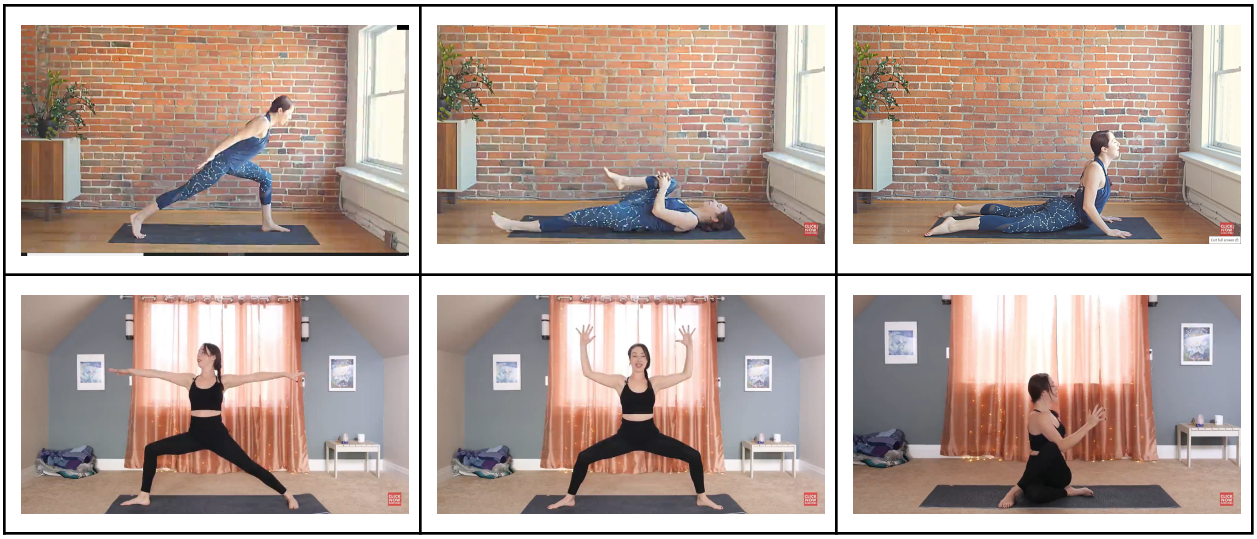


Figure 2: instructor poses

Participant 1 Data

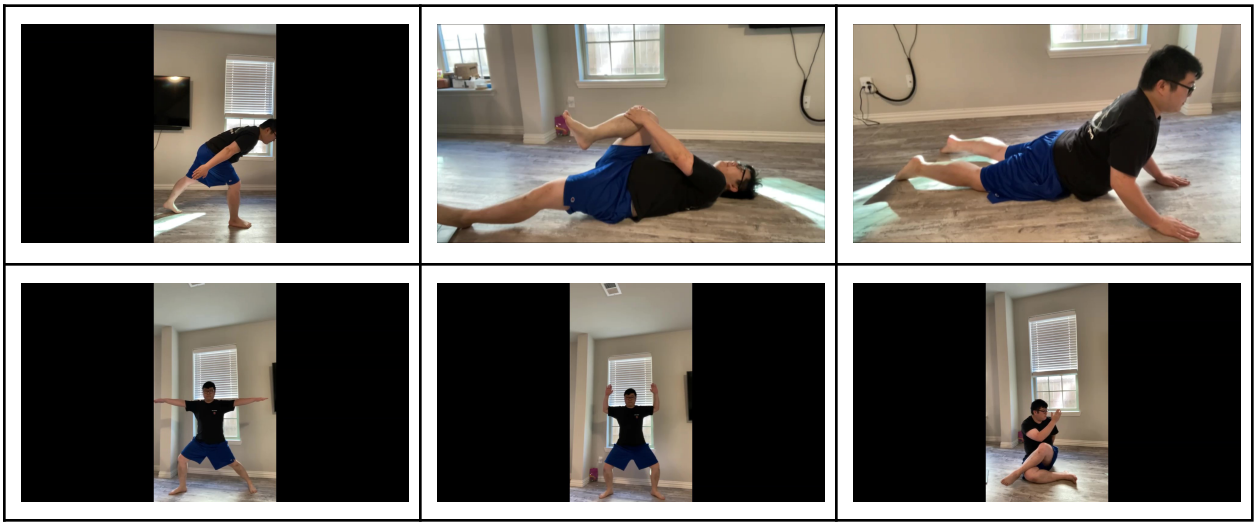


Figure 3: participant1 poses

Participant 2 Data

Figure 4: participant2 poses

Participant 3 Data

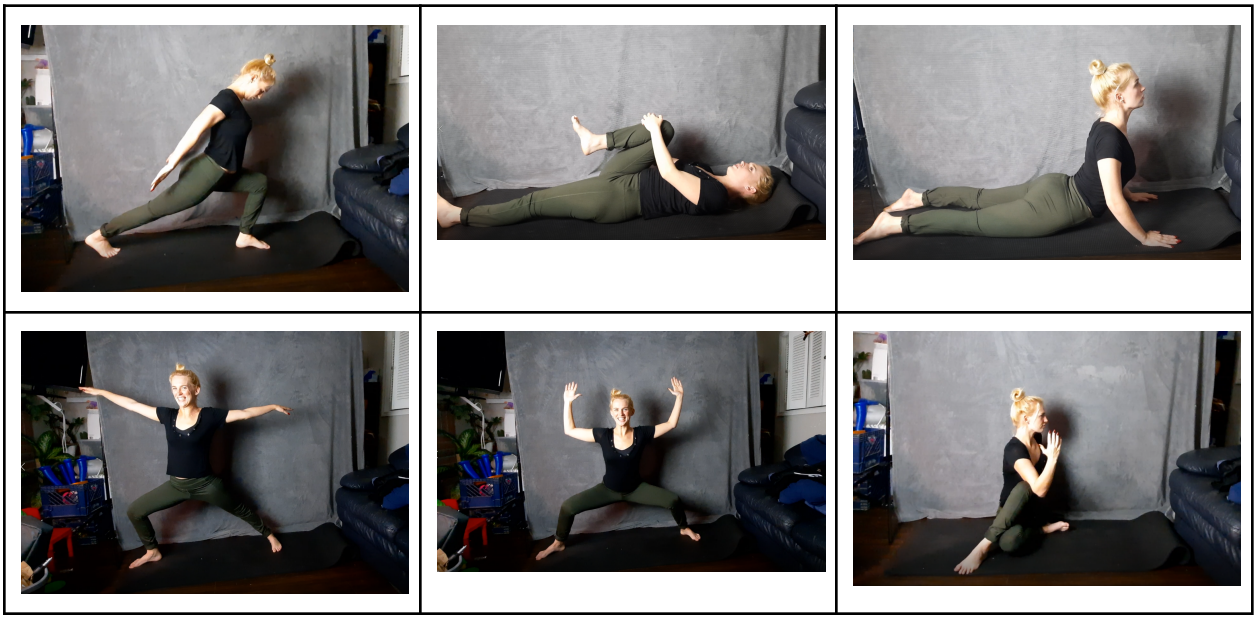


Figure 5: participant3 poses

Initial Experimentation

We experimented with BlazePose's landmark extractor by processing just a few images and visualizing what the extracted key points look like. These are visualized in the tables below.

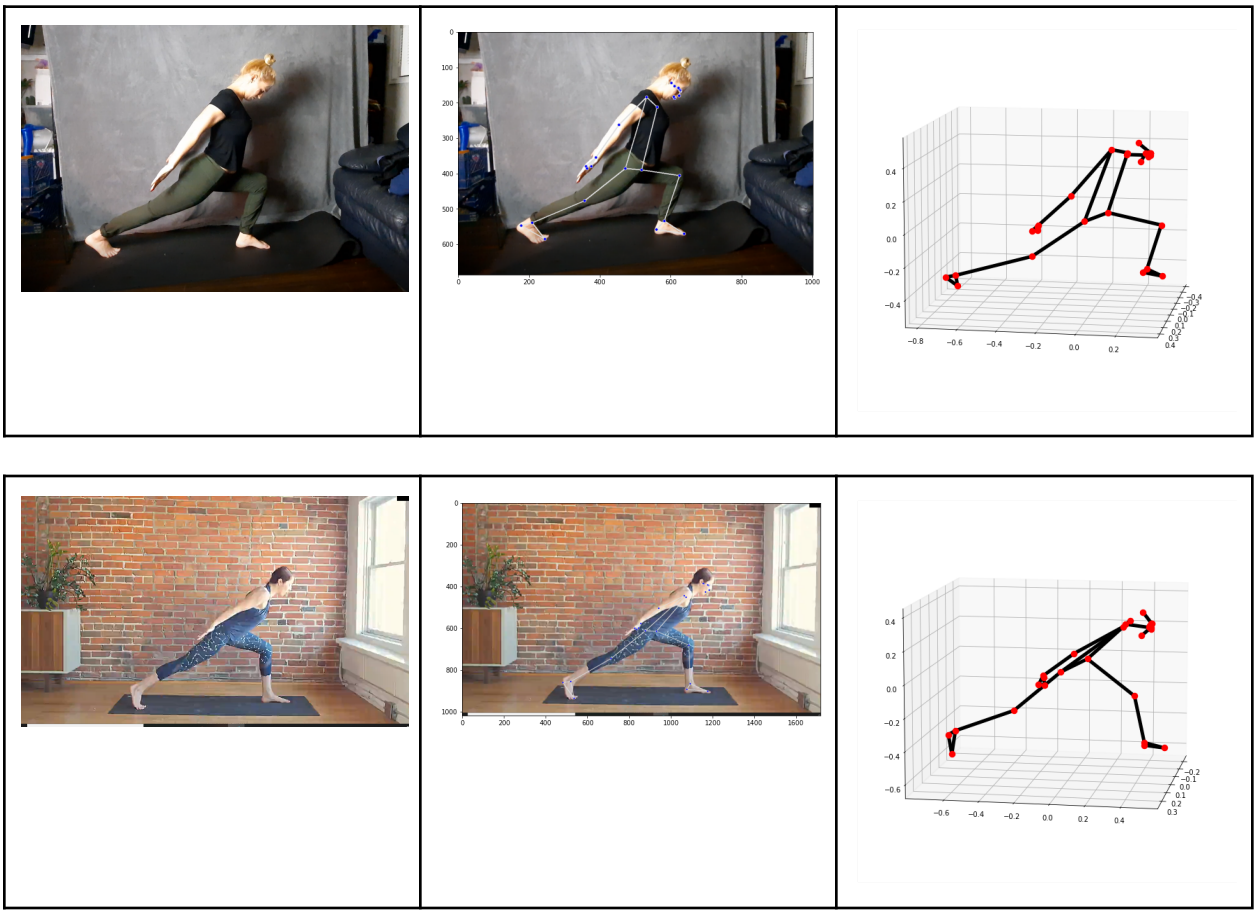
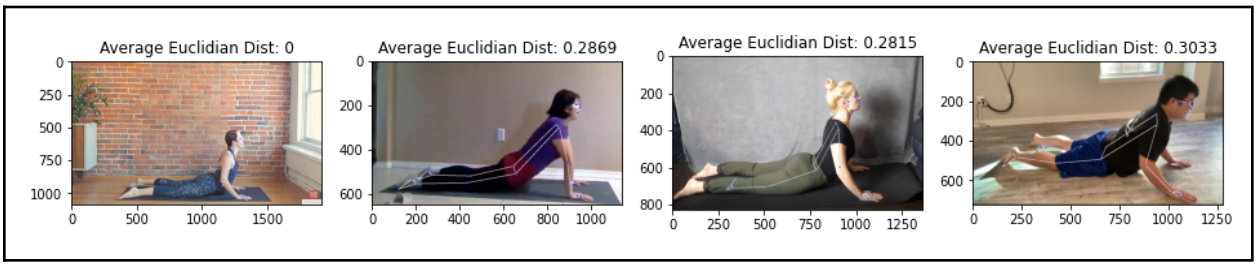


Figure 6: poses image and its 3D representation in BlazePose's landmark

Initial Similarity Score Results on Still Images

The initial similarity score pipeline on the still images extracted the keypoint landmarks from each image, calculated the Euclidean distance or L2 Norm of each respective keypoint, and averaged the differences all together. The results are shown in the figures below with the average euclidean distance scores above the images. Each image is compared to the one on the far left which is the gold standard image.



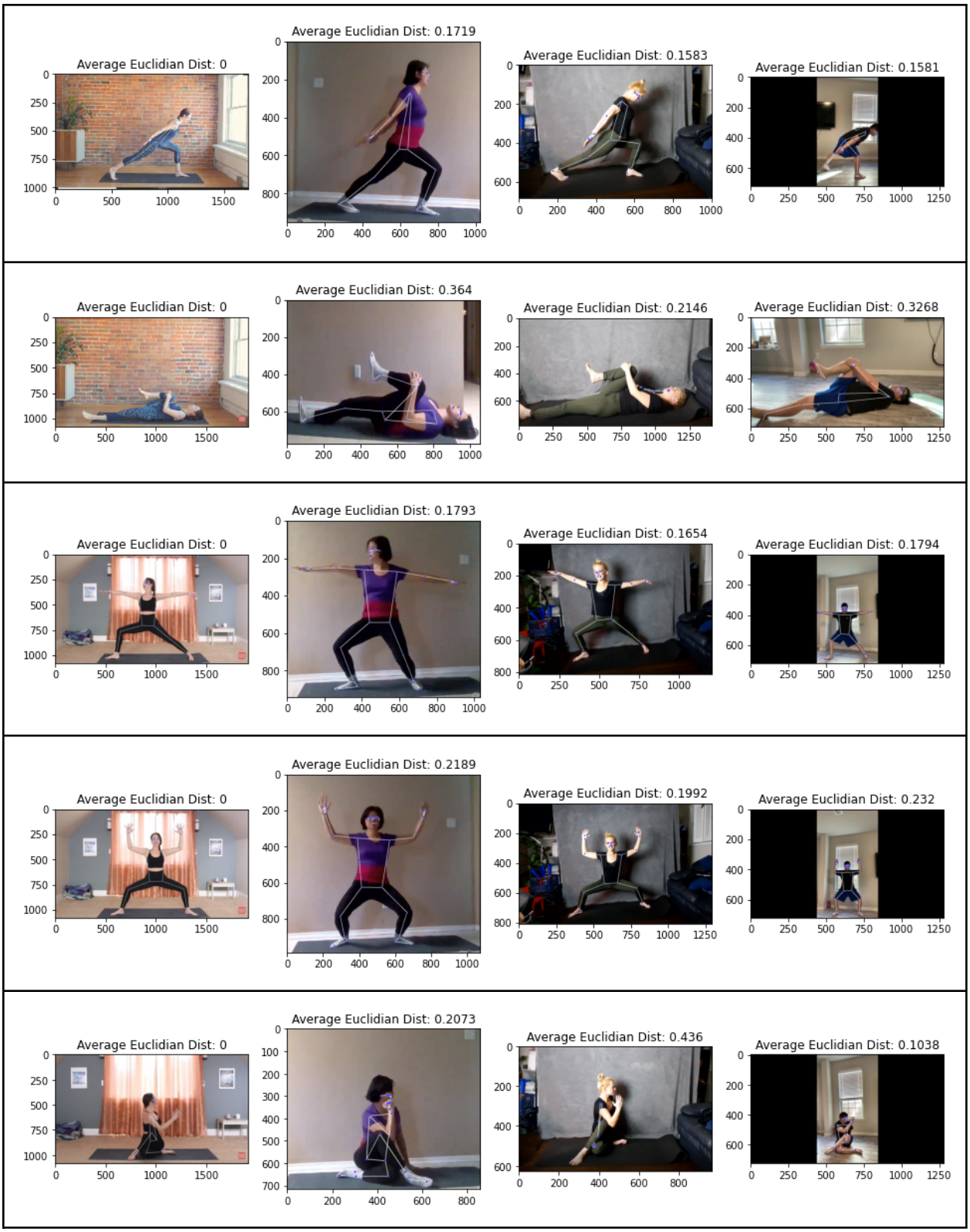


Figure 7: initial similarity calculation before image segmentation

Since the landmark coordinates are normalized to the length and height of the images, they range from 0 to 1. Therefore the average distances are indicative of the entire pose being off by that number's percentage. The scores range from 15.8% difference to up to 36.4% difference from the gold standard data.

Image segmentation trials and result

We tried several methods to segment figures in each image, remove the background and adjust image size. Because of the photo condition, floor color and our leg color would be too similar for threshold setting, so we tried Gabor filter bank for the segmentation. Among several setting trials, we came up with the better outcome listed below:

Gabor filter

Gabor filter: we use gabor filter bank to try on image segmentation and removing background using k means.

Gabor setting:
Sigma: 0.2
Orientation = [0 10 15 20 25 30 35 40 45 50 55 60 65 70 75 80 90];

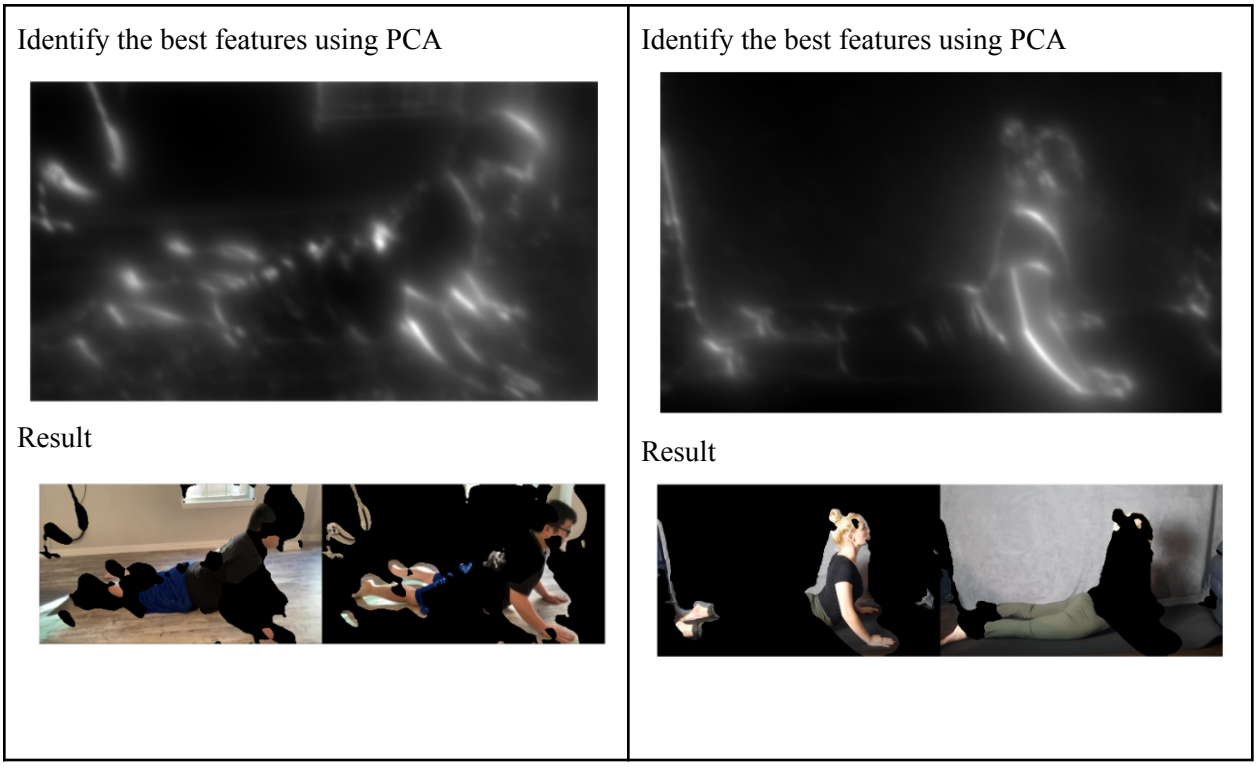


Figure 8: lighter tone and darker tone images for gabor filter bank result comparison

Gabor setting:
Using Gaussian filter first to get rid of high frequency components: Iblur1 = imgaussfilt(gImg,4);
Sigma: 0.2
Orientation = [0 10 15 20 25 30 35 40 45 50 55 60 65 70 75 80 90];
We get a better result on the right image compared to the previous.

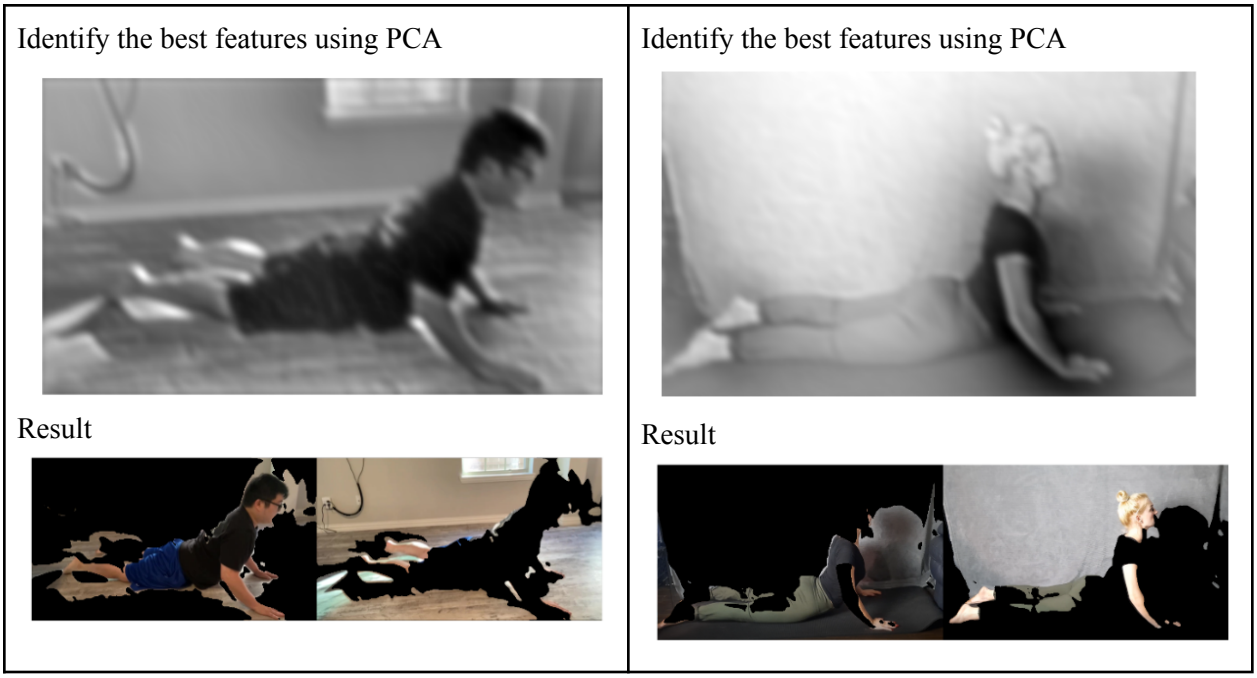
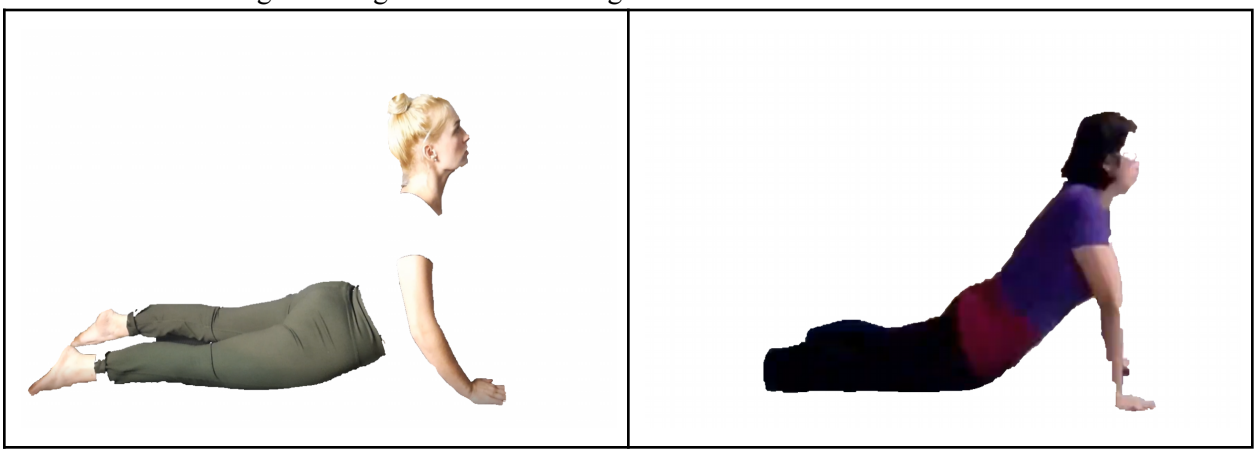


Figure 9: Applied gaussian filter before applying gabor filter bank on brighter tone image and darker tone image

Grabcut

We also tried the cv2 grabcut algorithm to do the segmentation.



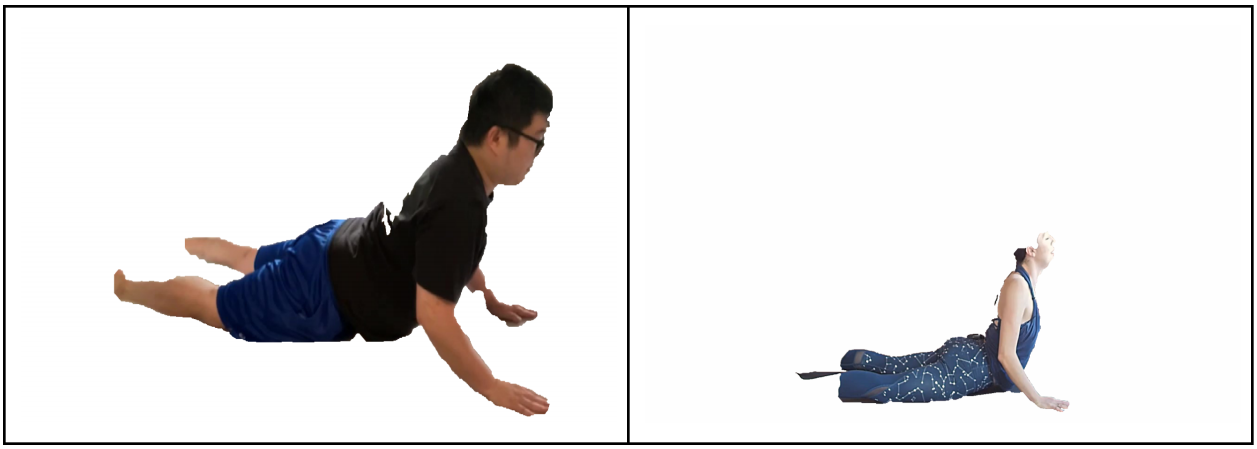


Figure 10: grabcut algorithm applied on image segmentation

Mask RCNN

Here we download pretrained Mask RCNN model weights to do image segmentation.

Figure 11: Mask RCNN applied on image segmentation

After extracting the segmented images of each participant, we reran the euclidean distance pipeline to get much closer numbers in similarity shown in the comparison table below. The distances are much closer ranging from .0699 to .2195 which shows the poses are between 7% and 22% similar to the gold standard instructor poses.

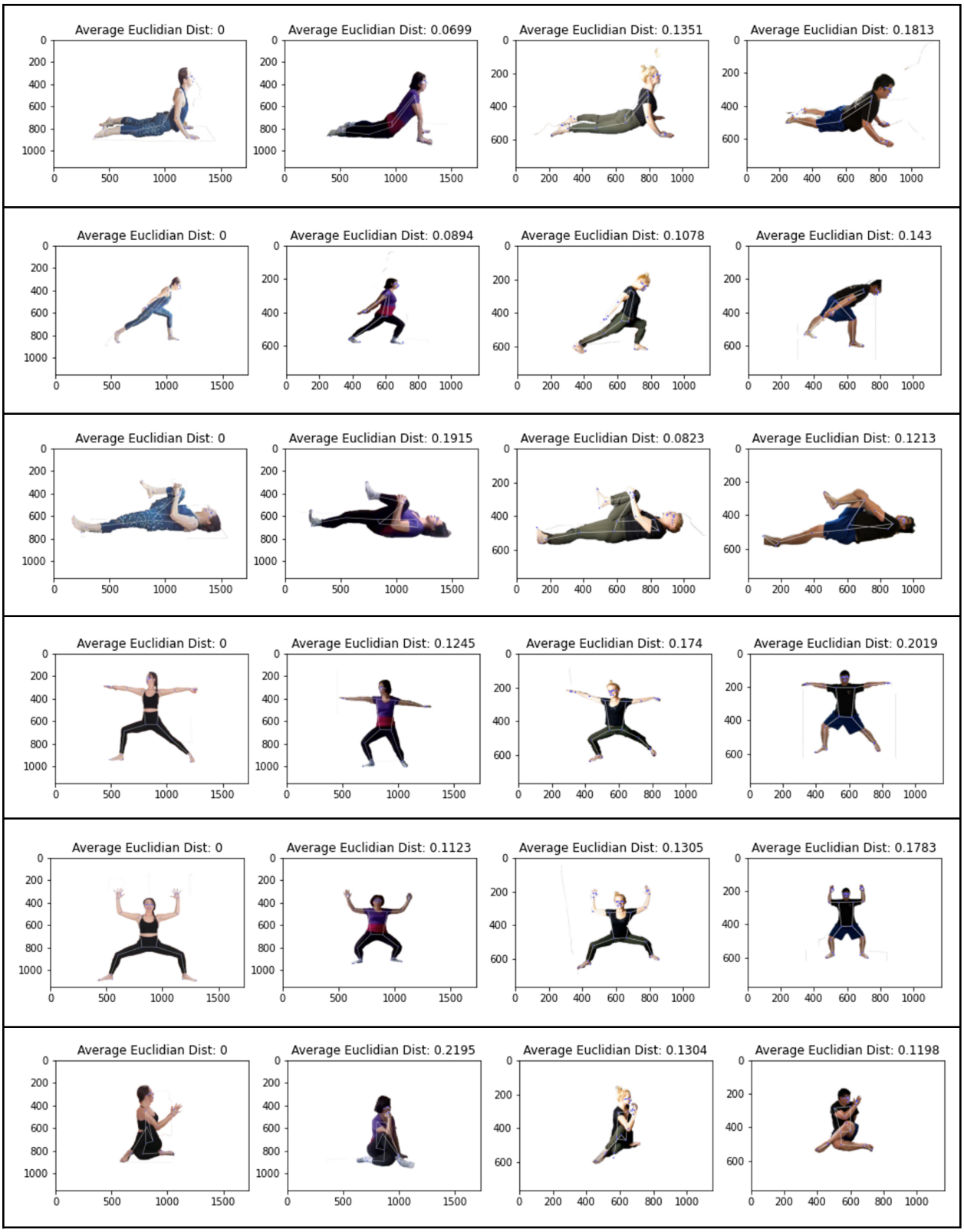


Figure 12: images after applied Mask RCNN

Discussion on image segmentation

From previous examples of segmentation, we can clearly notice that both gabor filter bank and mask RCNN perform better compared to the grabcut algorithm. Gabor filters are used to look for strong responses of frequency components. Because a single filter only extracts features in a certain direction, we tried filter banks and included various orientations from zero degree to 90 degree for a fine extraction. Our frames show diversity in the image : cloth color of each individual; the color difference between cloth and body part(leg, hands); color tone or brightness of background color, floor color; each part may also be different depending on the light effects on each individual which we consider as normal when users upload their self recording videos. The effort to deal with these diversities is closer to reality. And for some video frames that contain an overall darker tone performs worse than ones with brighter color tones, even the color of clothes are distinct from its background. In such cases, to eliminate some high frequency components by using a gaussian filter solves some of these problems, but it sometimes would be confusing for a brighter color tone frame because it would make shadows of the body clearer. To apply gabor filter banks for extracting people in the frame requires categorization of image or apply some techniques such as adjusting the image color tone to brighter ones or to tone down the image, and adjust gabor filter setting accordingly. Alternatives also include explicitly recommending users be aware of the recording environment for a more precise result.

For the second method, we tried the grabcut algorithm. It estimates the color distribution of the person and its background using a Gaussian mixture model which constructs pixel labels connecting regions with the same labels. After observation of the resulting images, we found it would lose body parts that have distinct color from other connected parts, and it often fails to include a complete body part such as face or leg. (As you may notice in our example images above)

For another method, we tried a deep learning mask RCNN pretrained model. We downloaded the pretrained model weight and used it to do image segmentation. It performs really well without adjusting the image beforehand. It recognizes objects as targets in its list. Then, select the target object with a mask and a bounding box. Here we get rid of the bounding box in the setting, so that we can simply do the image segmentation task and remove the background accordingly. In the practice, we eliminate other targets from the default target list and only leave people labels so it will not mask other targets.

```
1: 'person',
2: 'bicycle',
3: 'car',
4: 'motorcycle',
5: 'airplane',
6: 'bus',
7: 'train',
8: 'truck',
9: 'boat',
10: 'traffic light',
11: 'fire hydrant',
13: 'stop sign',
14: 'parking meter',
```

```
31: 'handbag',
32: 'tie',
33: 'suitcase',
34: 'frisbee',
35: 'skis',
36: 'snowboard',
37: 'sports ball',
38: 'kite',
39: 'baseball bat',
40: 'baseball glove',
41: 'skateboard',
42: 'surfboard',
43: 'tennis racket',
```

```
61: 'cake',
62: 'chair',
63: 'couch',
64: 'potted plant',
65: 'bed',
67: 'dining table',
70: 'toilet',
72: 'tv',
73: 'laptop',
74: 'mouse',
75: 'remote',
76: 'keyboard',
77: 'cell phone',
```

15: 'bench', 16: 'bird', 17: 'cat', 18: 'dog', 19: 'horse', 20: 'sheep', 21: 'cow', 22: 'elephant', 23: 'bear', 24: 'zebra', 25: 'giraffe', 27: 'backpack', 28: 'umbrella',	44: 'bottle', 46: 'wine glass', 47: 'cup', 48: 'fork', 49: 'knife', 50: 'spoon', 51: 'bowl', 52: 'banana', 53: 'apple', 54: 'sandwich', 55: 'orange', 56: 'broccoli', 57: 'carrot', 58: 'hot dog', 59: 'pizza', 60: 'donut',	78: 'microwave', 79: 'oven', 80: 'toaster', 81: 'sink', 82: 'refrigerator', 84: 'book', 85: 'clock', 86: 'vase', 87: 'scissors', 88: 'teddy bear', 89: 'hair drier', 90: 'toothbrush',

Table1: default target list of Mask RCNN

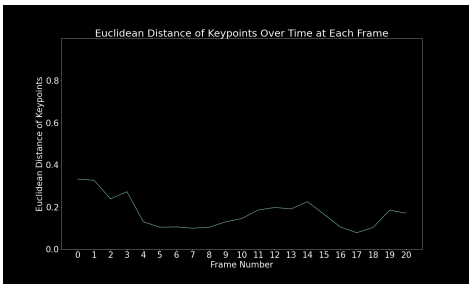
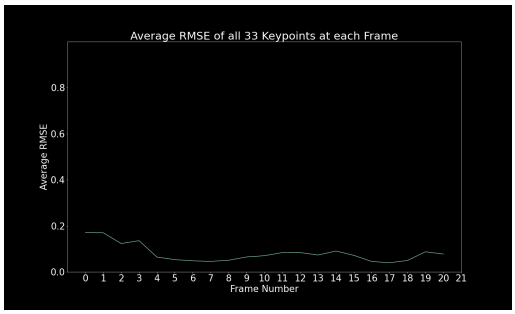
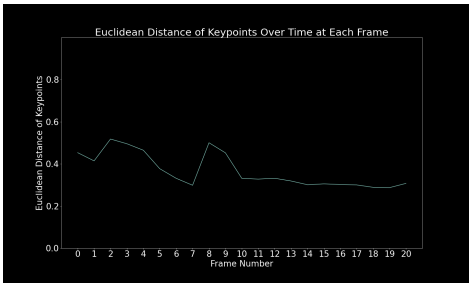
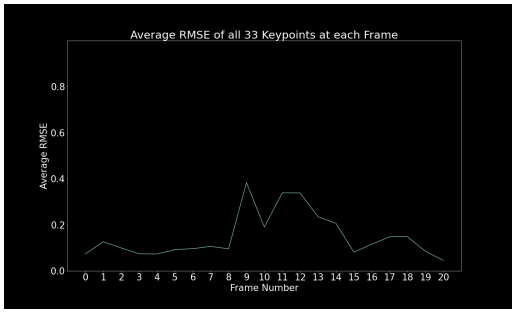
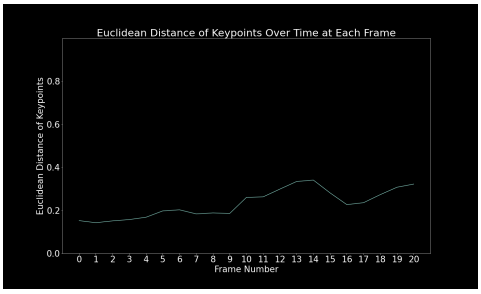
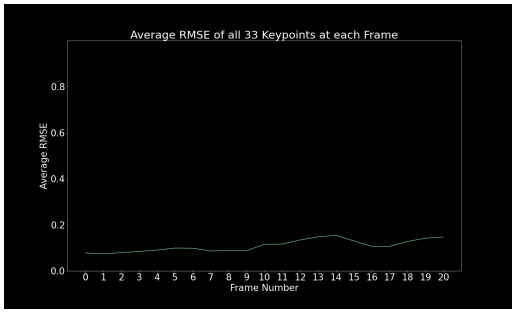
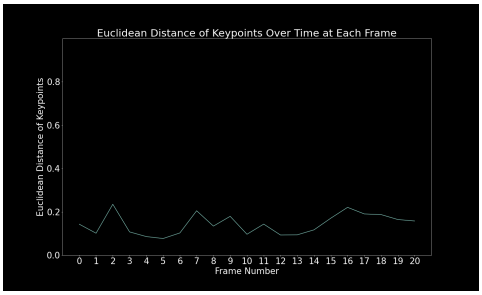
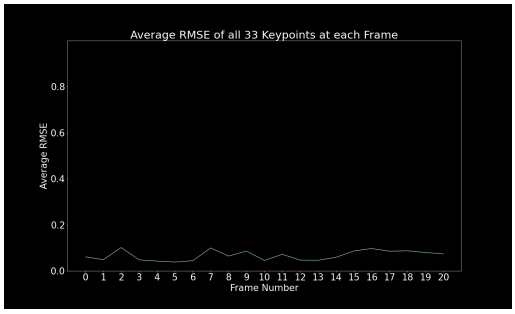
Discussion on Video Similarity Score Results

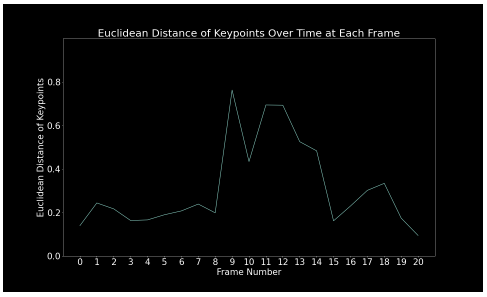
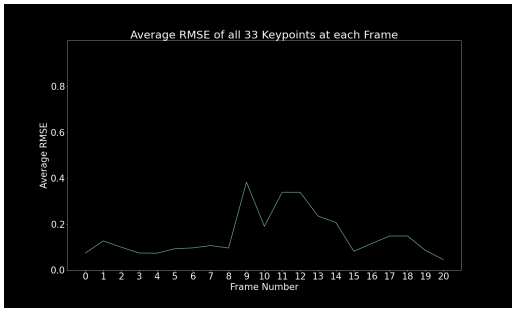
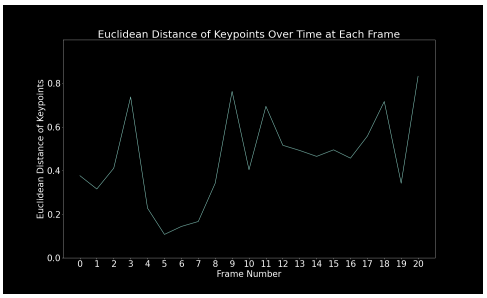
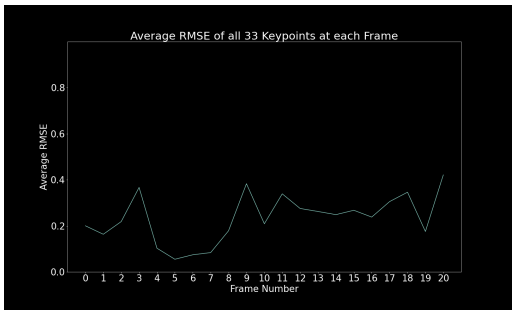
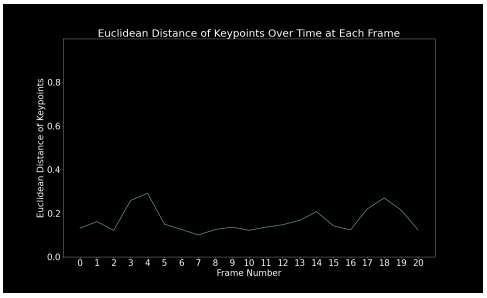
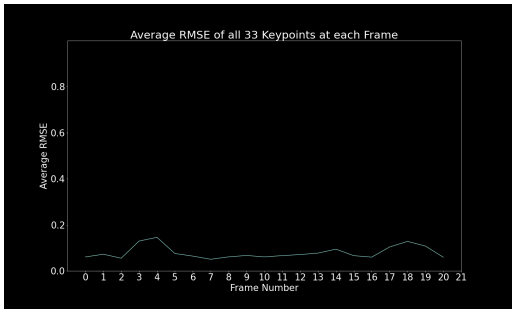
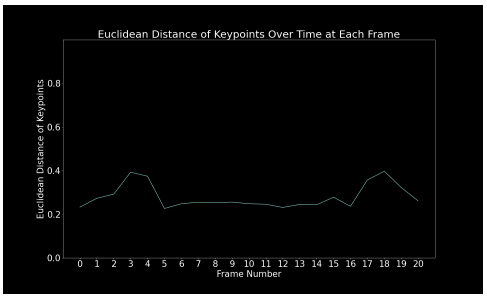
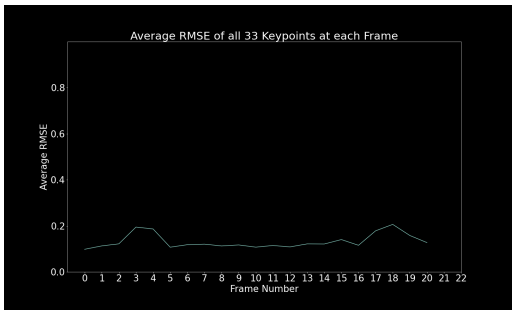
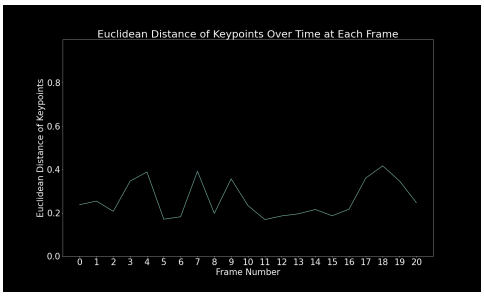
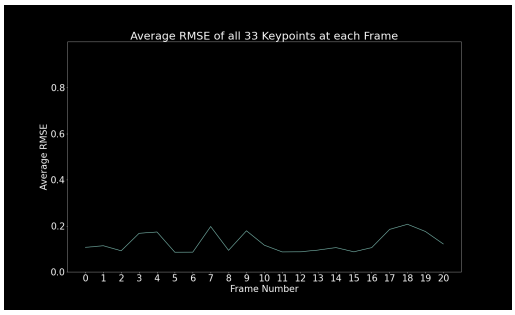
The videos all have different amounts of frames, so it is difficult to do a one to one euclidean distance comparison of poses. Pictured in the figure below are the frame counts of each of the 6 videos across the teacher and all participants.

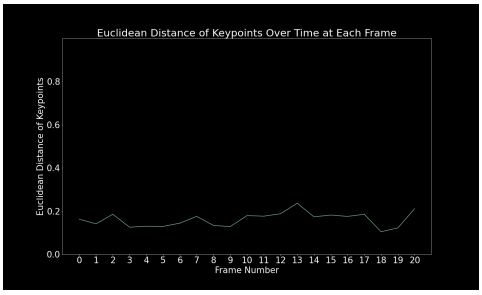
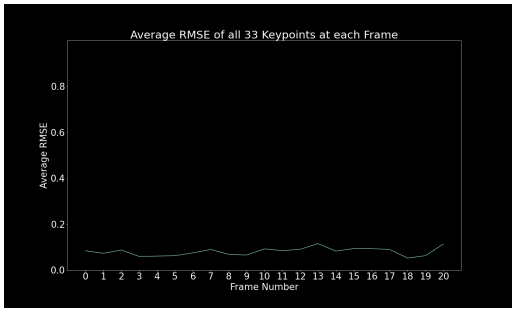
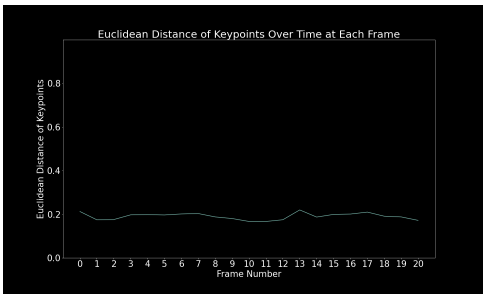
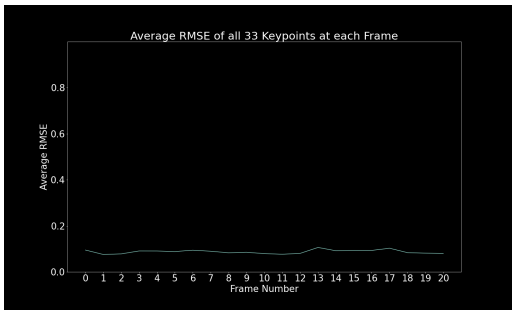
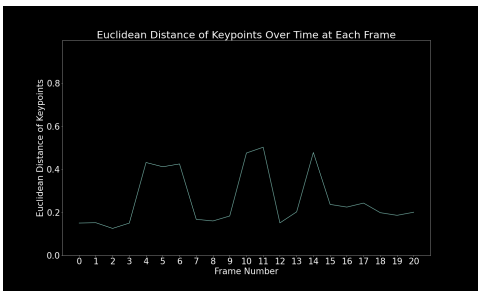
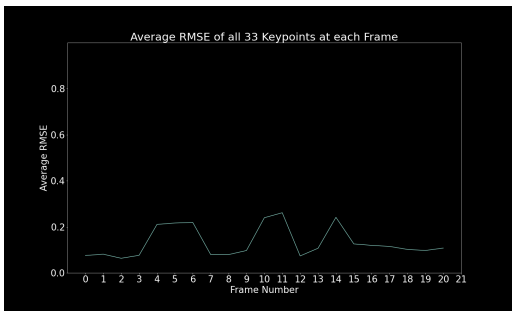
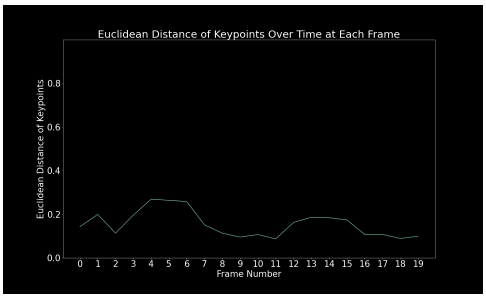
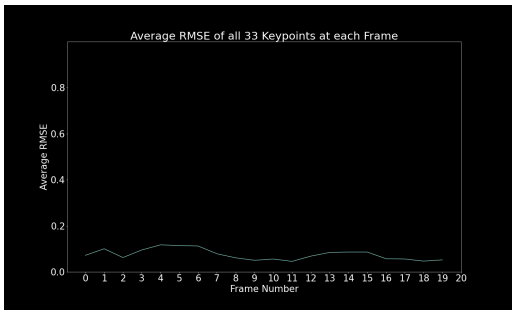
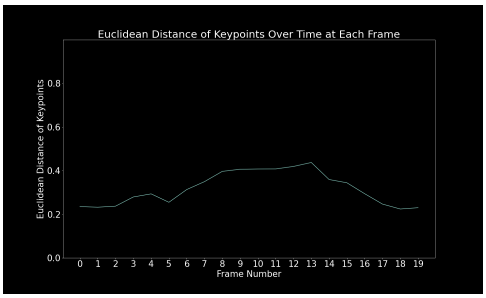
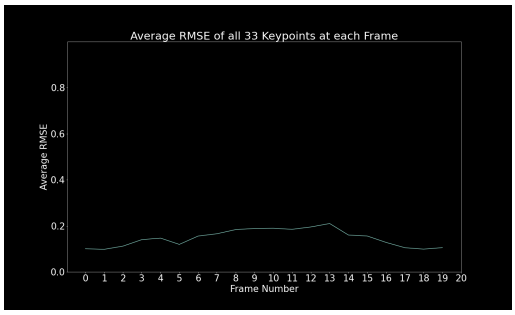
	teacher	lori	mikaela	jong
0	478	337	482	309
1	1152	313	804	600
2	229	195	207	158
3	379	165	275	290
4	370	220	380	181
5	487	295	306	469

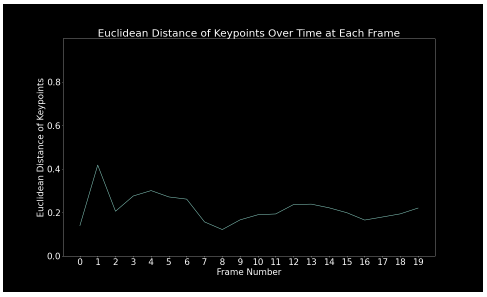
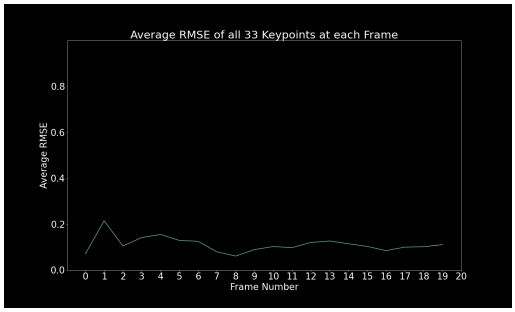
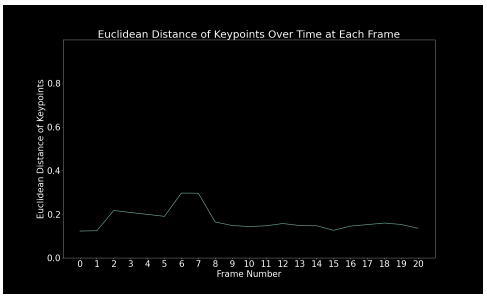
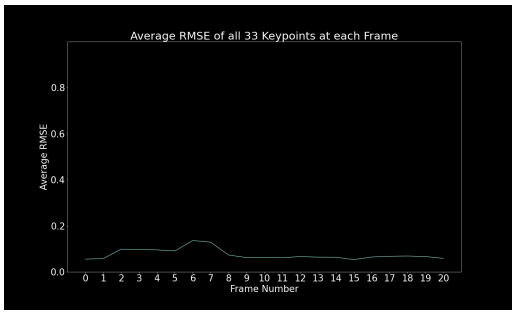
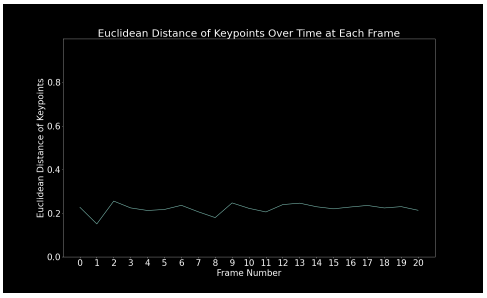
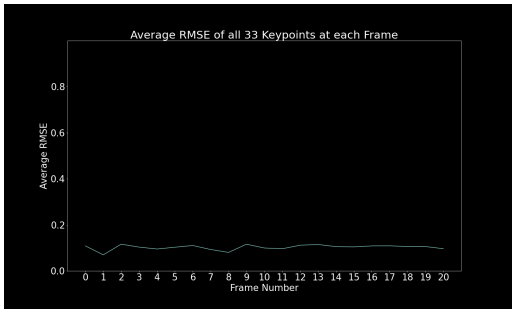
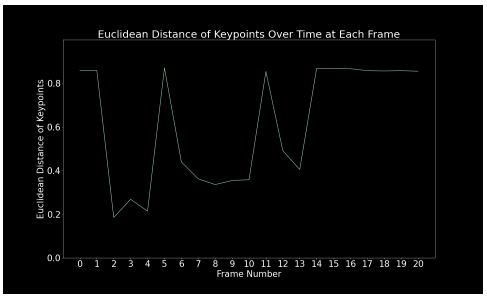
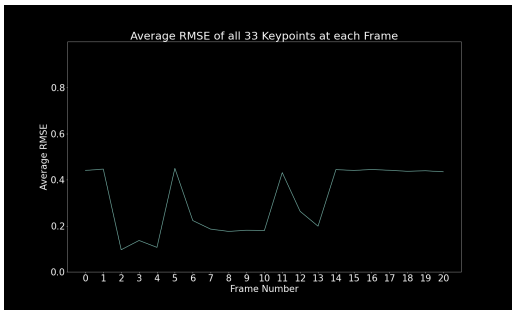
Our settled solution samples each video at their respective five percentiles. So every 5th percent of the total frame count gets sampled into a new video and in those frames the euclidean distance is calculated across all 33 keypoints. This ensures each video has the same number of frames to compare against. We also calculated the root mean squared error to see the absolute distance between each of the 33 keypoint pairs at each frame of the two videos.

Some videos had errors when extracting the frames so those have been excluded in these comparisons. The table below shows the graphs for the euclidean distance and the root mean squared error at each frame for each participant vs. the teacher video.

Pose #	Subj Name	Euclidean Distance Across Frames	RMSE Across Frames
1	Lori		
1	Jong		
1	Mikaela		
2	Lori		

2	Jong		
2	Mikaela		
3	Lori		
3	Jong		
3	Mikaela		

4	Lori		
4	Jong		
4	Mikaela		
5	Lori		
5	Jong		

5	Mikaela		
6	Lori		
6	Jong		
6	Mikaela		

All video comparisons with the sampled videos and the graphs in the same frame are listed in this directory with the pattern in the title <pose_number>_<subject_name>_comparison.mp4 There were errors extracting some landmark key points from some frames, and those landmarks got replaced with (0, 0, 0)'s for the (x,y, z).

Discussion of Results

Taking the still frames of the subjects' poses and comparing them to the gold standard instructor pose image and comparing the 33 extracted keypoints using MediaPipe's BlazePose model resulted in average Euclidean distances between 15.8% difference to up to 36.4% difference for the unsegmented images. With the segmented images we got better differences between 6% to 22%. This however is us manually choosing the still image that most matches the gold standard, instructor pose. To compare the entire videos of the pose and transition poses by calculating the Euclidean distance at each frame, results in varying results which may or may not reflect the actual reality. Using that metric alone to determine similarity between two videos would produce a highly dissimilar evaluation of a video of a participant doing the pose exactly but just a few frames before or after the instructor. The speed of the pose matters as well. This could be mitigated by carefully and manually editing the videos to start at the exact frame of the beginning of the pose, however that would require special engineering to do at scale for any number of videos.

The settled solution of taking every nth% frame of each video gets at least a relatively similar frame slice of the video and reduces computation time. Even with only ~25 frames for each sub-sampled video, it took several minutes to run the processing pipeline, so if we were to compare every single frame of each video it would take many times longer to calculate but would result in a much clearer picture of the similarity of the videos.

[Future task] Improving the algorithm for calculating pose similarity

The MediaPipe BlazePose model builds a Landmarks object which stores the 3D coordinates of each of the 33 keypoints. It means we can calculate angles between adjacent keypoints by using an inner (dot) product.

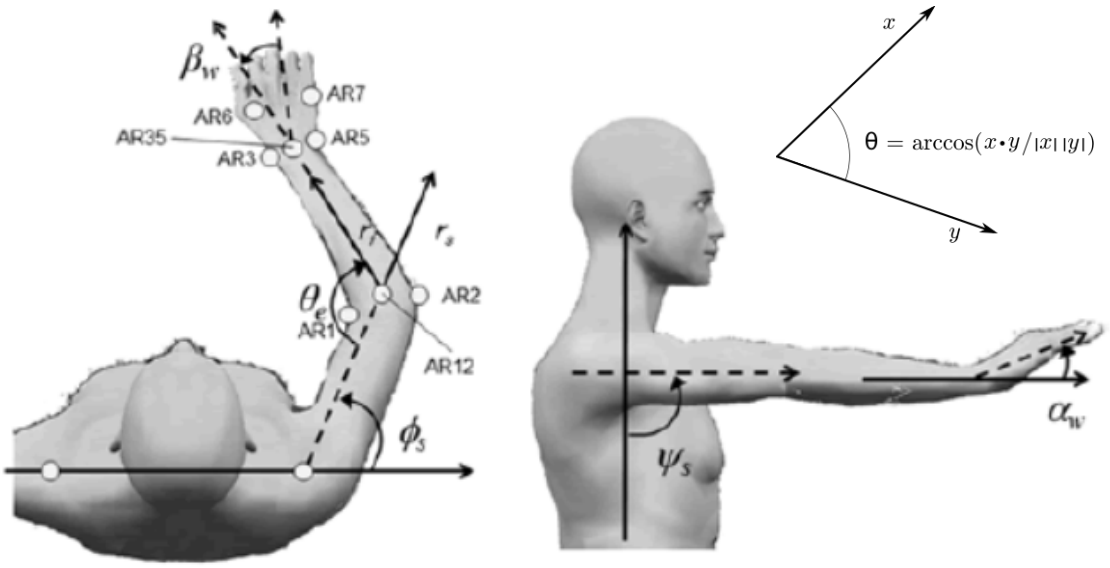


Figure 13: Angles of shoulder, elbow and wrist

Below figure shows the angle similarities of 6 poses.

Figure 14: Angle similarity calculation

Definitely, similarity calculation is the important task in yoga pose detection application, because the similarity means how well users are following a yoga teacher's posture. Therefore, our future task is improving the algorithm to get a more accurate similarity score, and it will help users to learn yoga postures quickly and accurately.

Reference

- Our coach video
https://youtu.be/6hZIzMpHl-c

- Google's MediaPipe
https://github.com/google/mediapipe

- Overview of Pose Detection:
https://www.fritz.ai/pose-estimation/#part-resources

- Datasets resources:
https://www.fritz.ai/pose-estimation/#part-resources

- Google's resources on their MediaPipe BlazePose, pose detection research
https://google.github.io/mediapipe/solutions/pose#python-solution-api
https://google.github.io/mediapipe/solutions/models.html#pose
https://blog.tensorflow.org/2021/08/3d-pose-detection-with-mediapipe-blazepose-ghum-tfjs.html (3D)

- Overview and in depth review of Google's MediaPipe BlazePose, pose detection research.
https://www.youtube.com/watch?v=brwgBf6VB0I

- Openpose dataset
https://github.com/CMU-Perceptual-Computing-Lab/openpose

- React App resource
https://github.com/ReactNativeNews/React-Native-Apps

- Ignite - the hottest React Native boilerplate
https://github.com/infinitered/ignite

- Colorwaver
https://github.com/mrousavy/Colorwaver

- Euclidean distance processing
https://www.mdpi.com/2073-4433/11/4/376/pdf

- Overview and in depth review of Google's MediaPipe BlazePose, pose detection research.
https://www.youtube.com/watch?v=brwgBf6VB0I