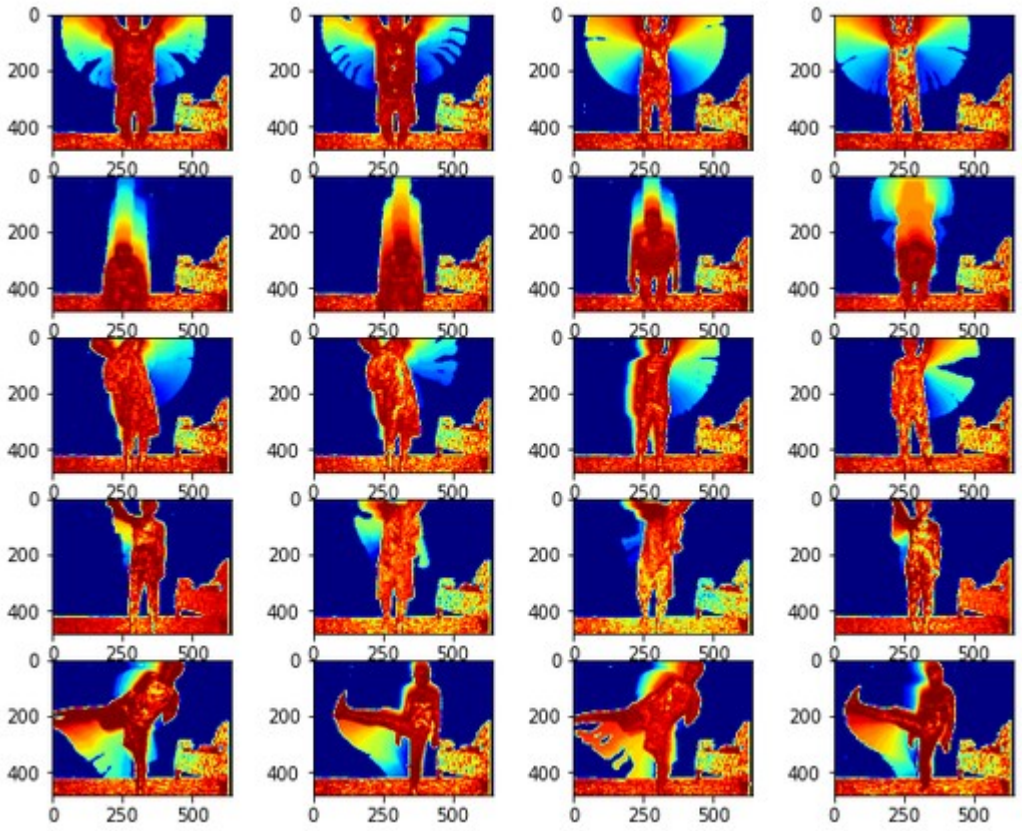
Project 5 – Computer Vision

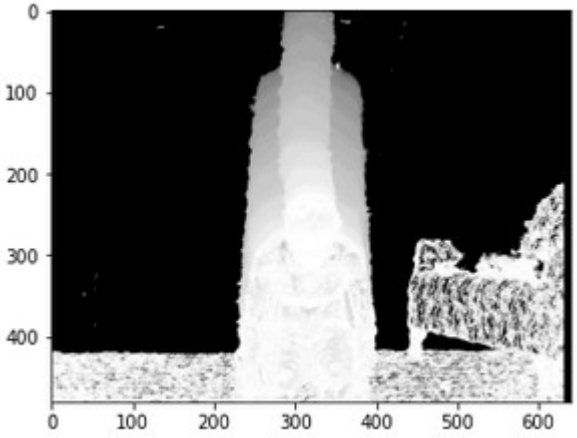
Part 1:

Display 5*4 grid

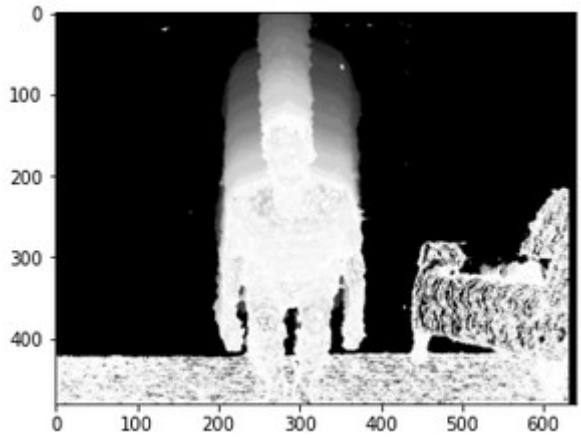


Part 4:

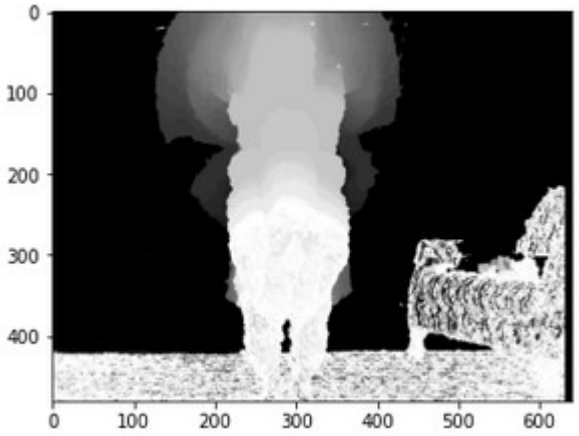
Test Example 1:



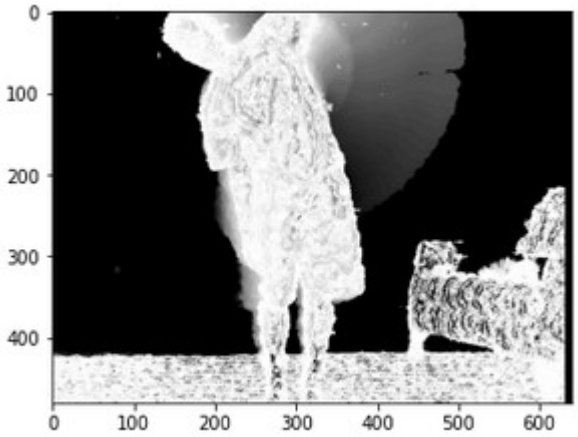
Top 4 most similar MHIs:



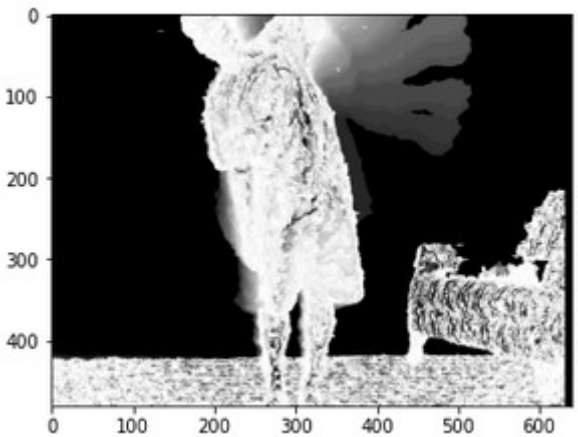
1st most similar



2nd most similar

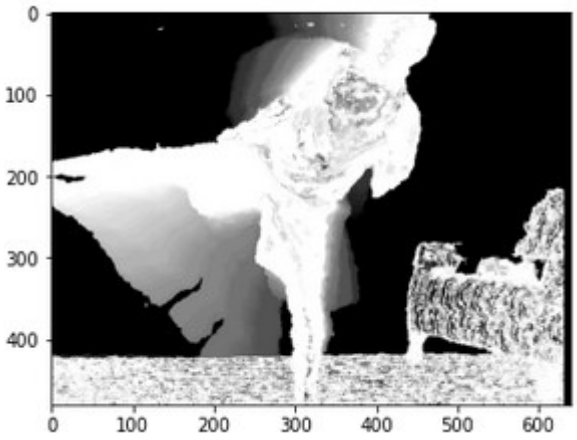


3rd most similar

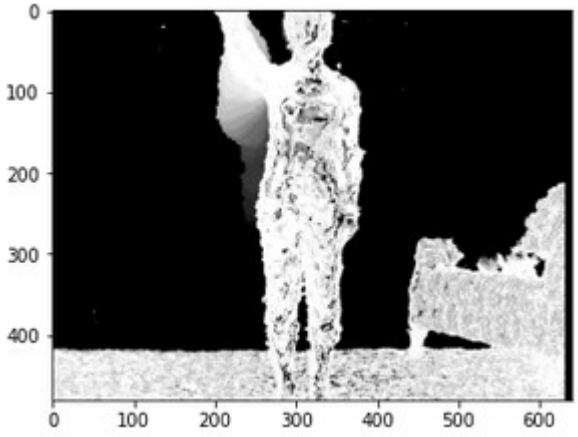


4th most similar

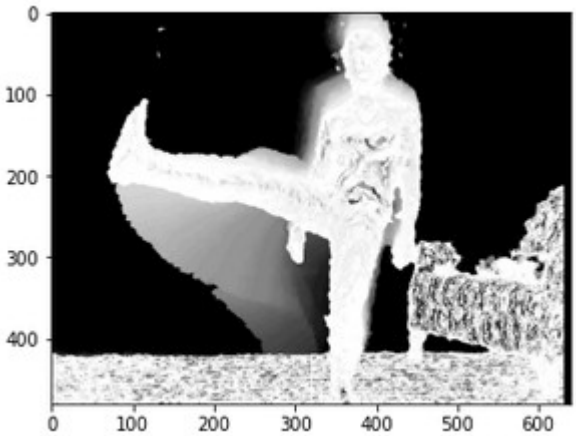
Test example 2:



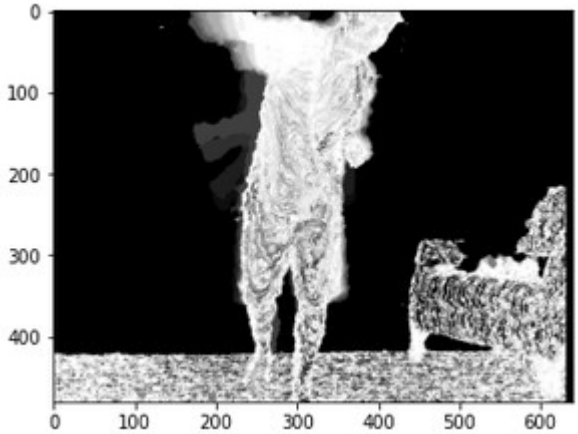
Top 4 most similar MHIs:



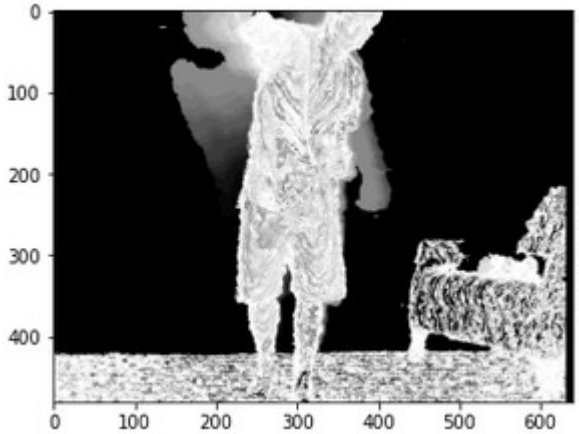
1st most similar



2nd most similar



3rd most similar



4th most similar

part 5: Classify All Actions

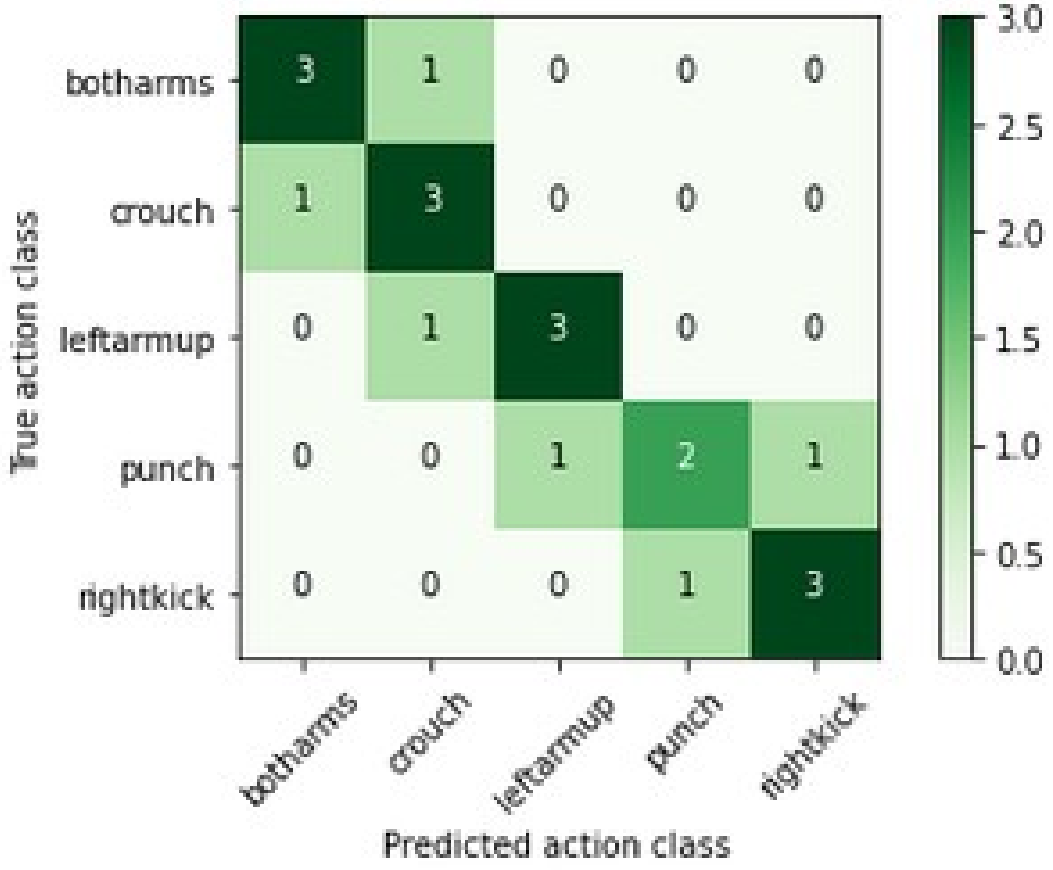
1. Overall Recognition Rate is 0.75

2. Mean Recognition Rate Per Class

```
botharms  :  0.75
crouch  :  0.75
leftarmup  :  1.0
punch  :  0.5
rightkick  :  0.75
```

3. Display a 5x5 Confusion matrix

	botharms	crouch	leftarmup	punch	rightkick
botharms	3	1	0	0	0
crouch	1	3	0	0	0
leftarmup	0	0	4	0	0
punch	0	0	1	2	1
rightkick	0	0	0	1	3



4. Discuss the performance and the most confused classes

The overall recognition rate is as high as 75%. Most classes archived 75% recognition rate. This is because the photos for “both arms”, “crouch”, “left arm up”, “right kick” are very similar within class. I was expecting “both arms” class perform better, since the MHI is very different from other classes.

For “both arms”, 1 in 4 is recognized as “crouch”. I think this is because both arms up and crouch share some similarity in MHI (as we can see from Part 1 5*4 matrix, crouch (2,4) looks a bit like both arms up. Notice for “crouch”, 1 in 4 is recognized as “both arms”, this could support my reasoning.

For “left arm up”, 1in 4 is recognized as “crouch”. This could be caused by crouch(2,4) as well.

For “right kick”, 1 in 4 is recognized as punch, this is because “right kick” have some similarity with “punch”, as both are asymmetric on MHI. The rotation invariant hu moments have the lowest Normalized Euclidean Distance to crunch(4,2). Crunch(4,2) has a radial area on the left (as to viewers). The body and head are both tilted a bit. This might confuse my program to think it’s a “right kick”.

And for “punch”, the recognition rate is only 2 in 4. This is because the MHI of “punch” is similar to left arm up (3,2) and rightkick (5,3). The crouch is classified because of the change in arm movement (lean to the left).

I believe as the amount of data increases, the program will have a higher recognition rate.

Extra credit:

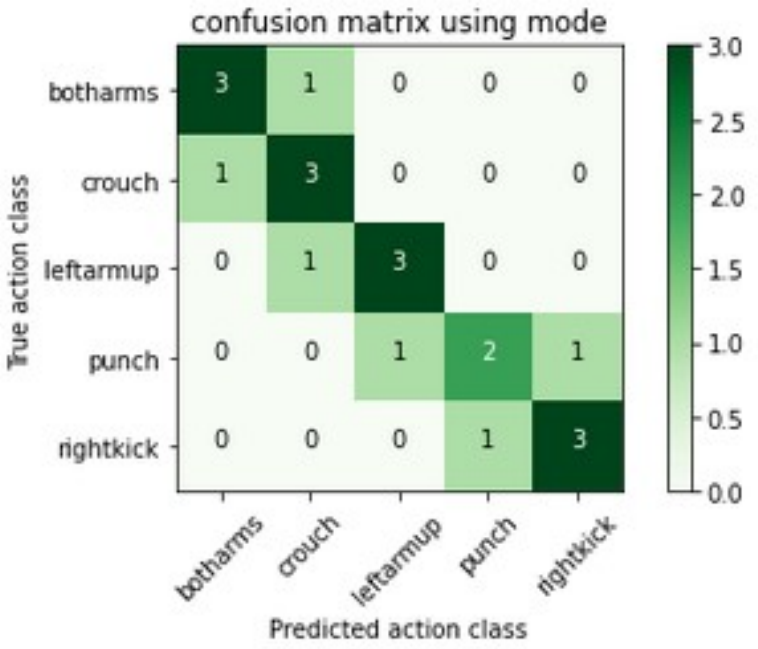
To improve the performance, I changed the predictAction using different normalization method, and other implementation of closest distance.

1. Using mode.
Mode is another thing that came in mind when finding the close matches. If we have more matches from the same class, we will more likely to get more actions from that class, thus achieving better performance. The result 0.7 is not as good as 1-NN method. This might be caused by the number of instances when hu moments are closer.

The function is shown in the jupyter notebook extra credit part.

Results:

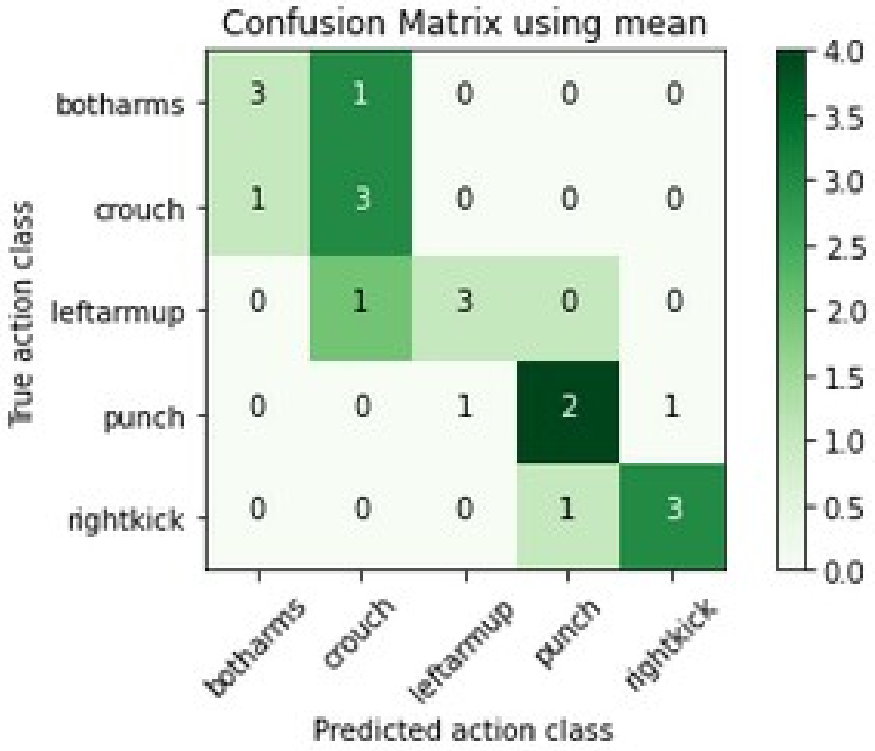
```
Mean Recognition Rate Per Class
botharms  :  0.75
crouch  :  0.75
leftarmup  :  0.75
punch  :  0.5
rightkick  :  0.75
Overall Recognition Rate
0.7
```



2. Using mean.
Similarly, I'm also thinking about using mean of the first 4 labels to improve the performance. The result is even worse than using mode. I think this is because the labels are very close and taking the average won't necessarily generate the best matches.

Result:

```
Mean Recognition Rate Per Class
botharms  :  0.25
crouch  :  0.75
leftarmup  :  0.25
punch  :  1.0
rightkick  :  0.75
Overall Recognition Rate
0.6
```



3. I also tried using other normalization methods. Instead of variance, I used square of variance. This method surprisingly produces better results. As the data size increases, the matching should be more accurate.

```
Mean Recognition Rate Per Class
botharms  :  1.0
crouch  :  0.75
leftarmup  :  0.75
punch  :  0.5
rightkick  :  1.0

Overall Recognition Rate
0.8
```