Project Report - 18.03.2018
Mehmet Ali ATICI

Project Title: HAND LANDMARK DETECTOR FOR AUTOMATIC HAND ANALYSIS

Motivation:

Hand research is one of the hot topics and currently there exists a number of works [1] in literature that investigate the correlation between digit ratio (2D:4D) and some other traits of a person. In Fig. 1, mostly used parameters in hand research are given:

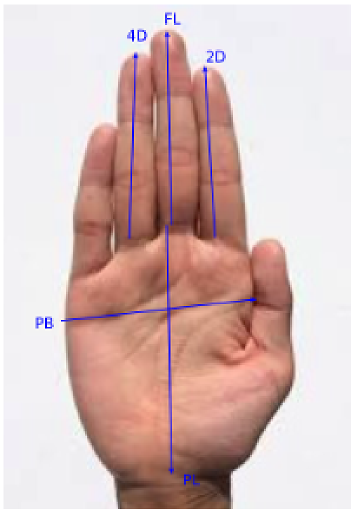


Fig. 1. Hand Research Parameters

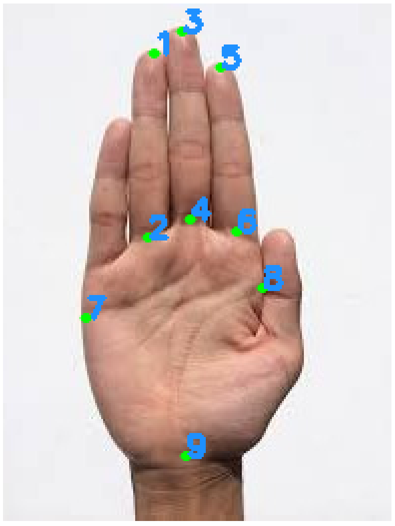


Fig. 2. Hand Landmarks

In Fig. 1;

4D is length of the fourth digit,
2D is length of the second digit,
PB, Palm Breadth, is width of the palm,
PL, Palm Length is height of the palm that is distance between bottom of the third digit and bottom of the palm, and
FL, Finger Length is the length of the third digit.

In hand research, the ratios of 2D/4D, FL/PB, FL/PL and PB/PL are commonly used.

Objective:

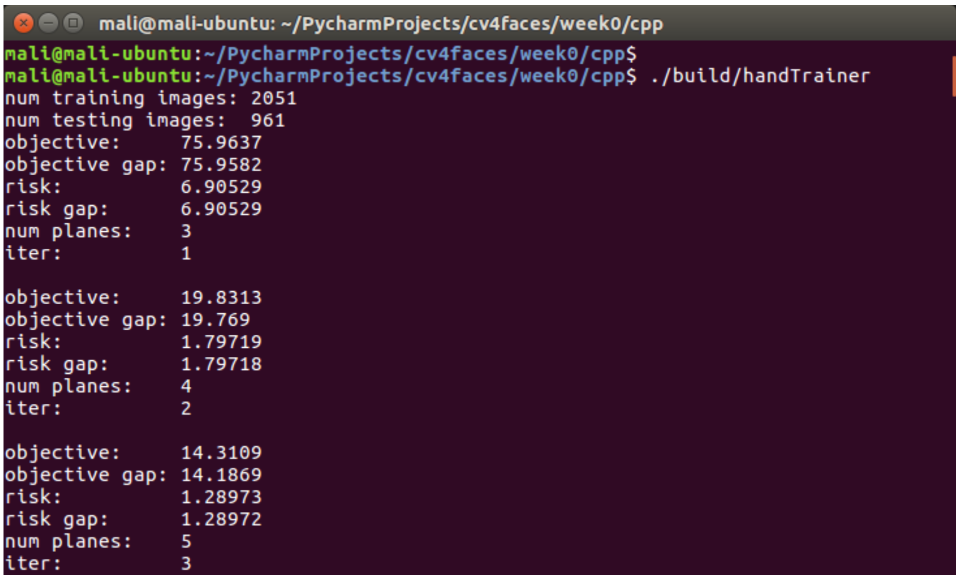
Main objective of this project is detecting the hand and calculating the ratios above automatically. For this, I have designed models that detect the hand box and 9 landmarks of the hand in images. The landmark positions are given in Fig. 2.

Dataset:

I have used the hand images from some publicly available datasets [2,3,4] for training and testing the hand detector and hand landmarks detector models. For hand detector, I have annotated hand boxes in 3012 images (2051 for training, 961 for testing) and for hand landmarks detector I have annotated 22 landmarks for each image of 931 images (691 for training, 240 for testing). The hands used in these datasets are belong to about 250 individuals. I have also designed a landmark detector that detects 22 landmarks however in this repository I share only the 9-landmarks detector model. It took about 40 hours to prepare these datasets. I used Dlib imglab tool for preparing the datasets.

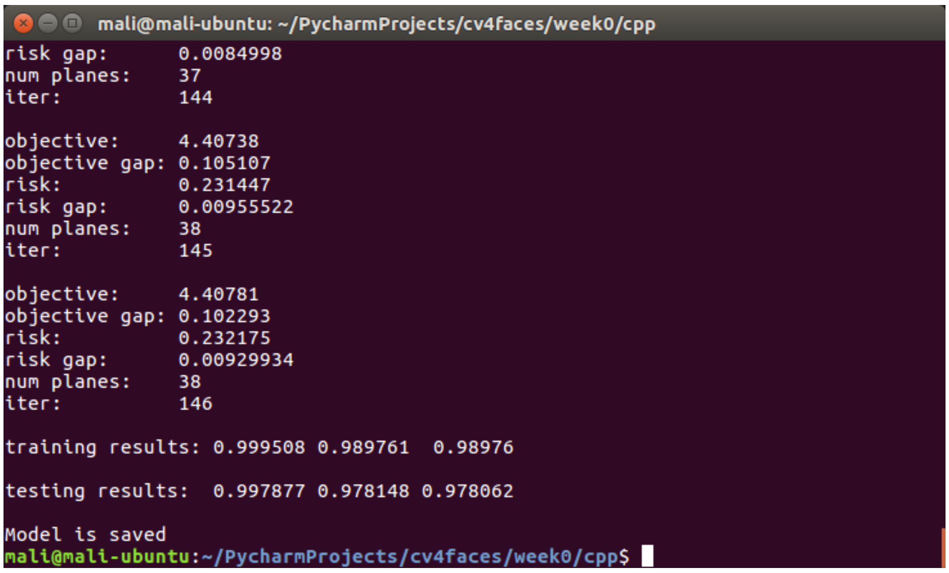
Methods:

In the project I have used OpenCV and Dlib libraries. For hand detector, I used Dlib general object detection implementation that applies HOG + SVM method in [5] for training the model. The code that produces HandDetector.svm model exists in handTrainer.cpp file. Secreenshots for training the hand detector are given in Fig 3. and Fig 4.



```
mali@mali-ubuntu: ~/PycharmProjects/cv4faces/week0/cpp
mali@mali-ubuntu:~/PycharmProjects/cv4faces/week0/cpp$
mali@mali-ubuntu:~/PycharmProjects/cv4faces/week0/cpp$ ./build/handTrainer
num training images: 2051
num testing images:  961
objective:     75.9637
objective gap: 75.9582
risk:          6.90529
risk gap:      6.90529
num planes:    3
iter:          1

objective:     19.8313
objective gap: 19.769
risk:          1.79719
risk gap:      1.79718
num planes:    4
iter:          2

objective:     14.3109
objective gap: 14.1869
risk:          1.28973
risk gap:      1.28972
num planes:    5
iter:          3
```

Fig 3. Start of handTrainer

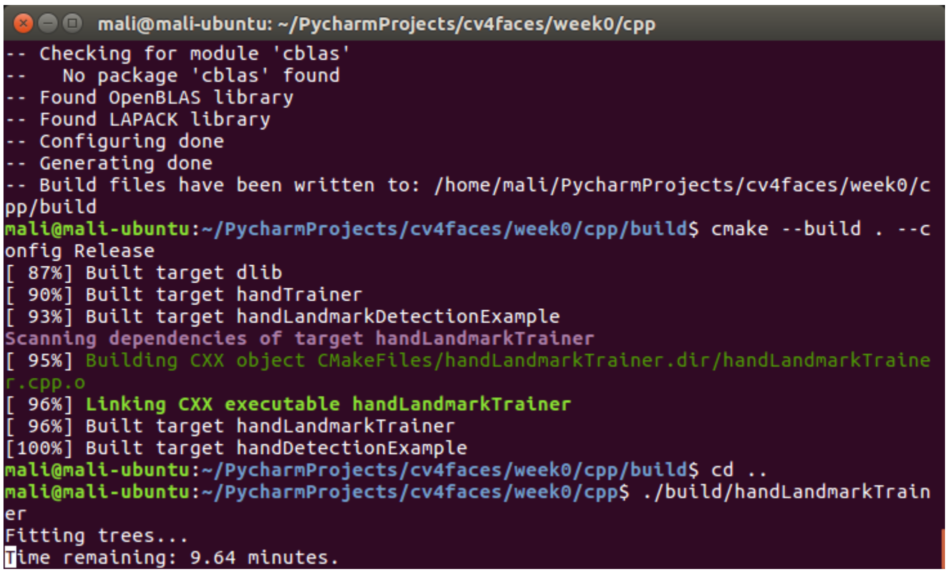


```
mali@mali-ubuntu: ~/PycharmProjects/cv4faces/week0/cpp
risk gap:      0.0084998
num planes:    37
iter:          144

objective:     4.40738
objective gap: 0.105107
risk:          0.231447
risk gap:      0.00955522
num planes:    38
iter:          145

objective:     4.40781
objective gap: 0.102293
risk:          0.232175
risk gap:      0.00929934
num planes:    38
iter:          146

training results: 0.999508 0.989761  0.98976

testing results:  0.997877 0.978148 0.978062

Model is saved
mali@mali-ubuntu:~/PycharmProjects/cv4faces/week0/cpp$
```

Fig 4. Results of handTrainer

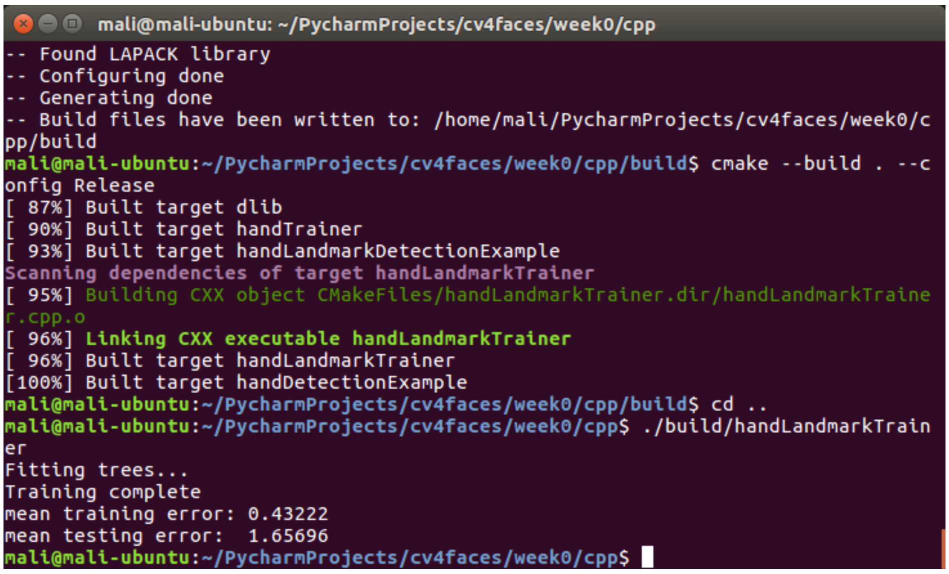
While training hand detector, I used sliding window size of (80,120) and I set the C value of SVM to 11. To improve the results I also enlarged the images 3 times. For detecting the small hands in the images, if no hand detected, I rescale the image as factor of 3 before detecting and resize the bounding box by shrinking 3 times. I didn’t apply any data augmentation because of both the results are already satisfying and the memory limitations.

For training the hand landmarks detector I used Dlib general shape trainer implementation that applies the method in [6]. The code for hand landmarks trainer model is given in handLandmarkTrainer.cpp file. I played with all parameters defined in the Kazemi's paper but only the **nu** and **tree_depth** parameters changed the accuracy to some extent. Screenshots of hand landmarks trainer are given in Fig 5. and Fig 6.



```
mali@mali-ubuntu: ~/PycharmProjects/cv4faces/week0/cpp
-- Checking for module 'cblas'
--   No package 'cblas' found
-- Found OpenBLAS library
-- Found LAPACK library
-- Configuring done
-- Generating done
-- Build files have been written to: /home/mali/PycharmProjects/cv4faces/week0/c
pp/build
mali@mali-ubuntu:~/PycharmProjects/cv4faces/week0/cpp/build$ cmake --build . --c
onfig Release
[ 87%] Built target dlib
[ 90%] Built target handTrainer
[ 93%] Built target handLandmarkDetectionExample
Scanning dependencies of target handLandmarkTrainer
[ 95%] Building CXX object CMakeFiles/handLandmarkTrainer.dir/handLandmarkTraine
r.cpp.o
[ 96%] Linking CXX executable handLandmarkTrainer
[ 96%] Built target handLandmarkTrainer
[100%] Built target handDetectionExample
mali@mali-ubuntu:~/PycharmProjects/cv4faces/week0/cpp/build$ cd ..
mali@mali-ubuntu:~/PycharmProjects/cv4faces/week0/cpp$ ./build/handLandmarkTrain
er
Fitting trees...
Time remaining: 9.64 minutes.
```

Fig 5. Start of handLandmarkTrainer



```
mali@mali-ubuntu: ~/PycharmProjects/cv4faces/week0/cpp
-- Found LAPACK library
-- Configuring done
-- Generating done
-- Build files have been written to: /home/mali/PycharmProjects/cv4faces/week0/c
pp/build
mali@mali-ubuntu:~/PycharmProjects/cv4faces/week0/cpp/build$ cmake --build . --c
onfig Release
[ 87%] Built target dlib
[ 90%] Built target handTrainer
[ 93%] Built target handLandmarkDetectionExample
Scanning dependencies of target handLandmarkTrainer
[ 95%] Building CXX object CMakeFiles/handLandmarkTrainer.dir/handLandmarkTraine
r.cpp.o
[ 96%] Linking CXX executable handLandmarkTrainer
[ 96%] Built target handLandmarkTrainer
[100%] Built target handDetectionExample
mali@mali-ubuntu:~/PycharmProjects/cv4faces/week0/cpp/build$ cd ..
mali@mali-ubuntu:~/PycharmProjects/cv4faces/week0/cpp$ ./build/handLandmarkTrain
er
Fitting trees...
Training complete
mean training error: 0.43222
mean testing error:  1.65696
mali@mali-ubuntu:~/PycharmProjects/cv4faces/week0/cpp$
```

Fig 6. Results of handLandmarkTrainer

I have tried tens of different parameter combinations for getting the best accuracy. It took about ten hours to achieve the best landmark trainer model. I have augmented the dataset by applying small rotations to the images. Trainer code files are given just to show the optimized parameter values and are useful when somebody has own annotated datasets.

Installation and Running the Codes:

This project is developed by using:

1. OpenCV 3.3
2. Dlib 19.7

After successful installation of OpenCV;

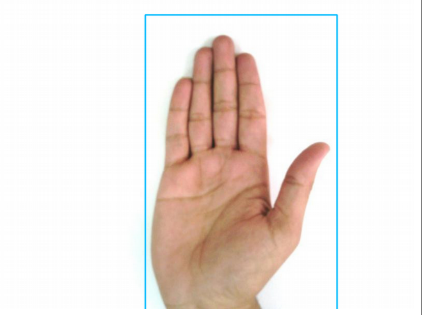
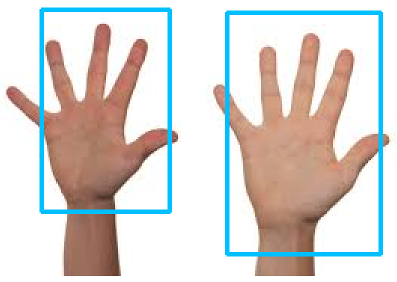
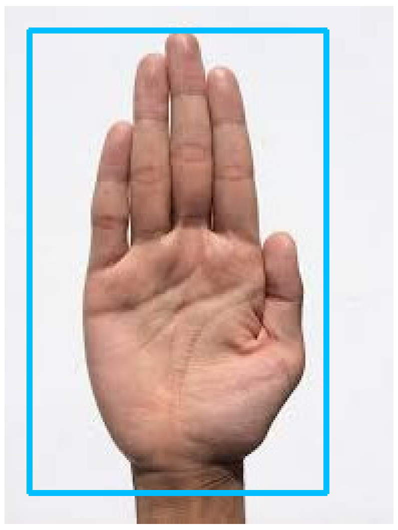
Download Dlib from http://dlib.net
Download this repository
Edit CmakeLists.txt according to your Dlib directory

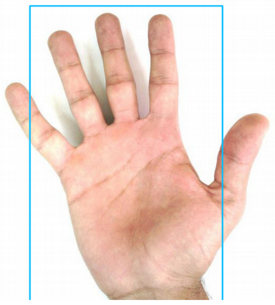
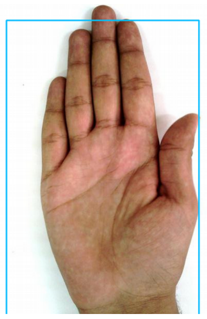
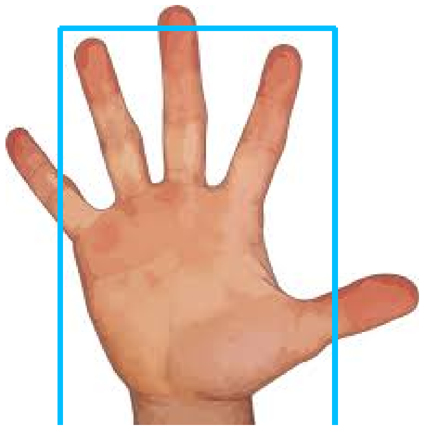
Open a terminal and change the directory to this repository and run following commands:

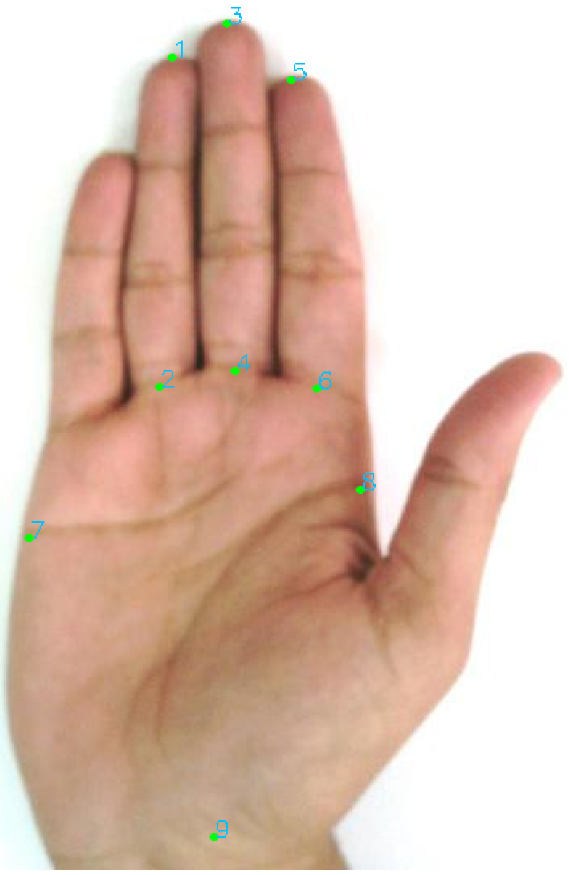
```
mkdir build
cd build
cmake ..
cmake --build . --config Release
cd ..
./build/handDetectionExample path/to/hand/images
./build/handLandmarkDetectionExample path/to/hand/images
```

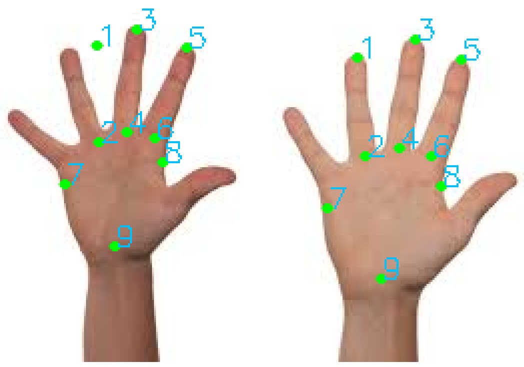
Results:

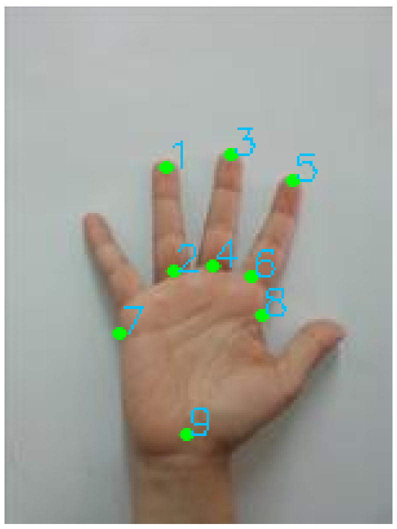
To show the detection of hand boxes and hand landmarks, I developed the handDetectionExample.cpp and handLandmarkDetectionExample.cpp files. These require the user to give hand images path when running. Sample images that show detection of hand boxes and detection of hand landmarks are given below:

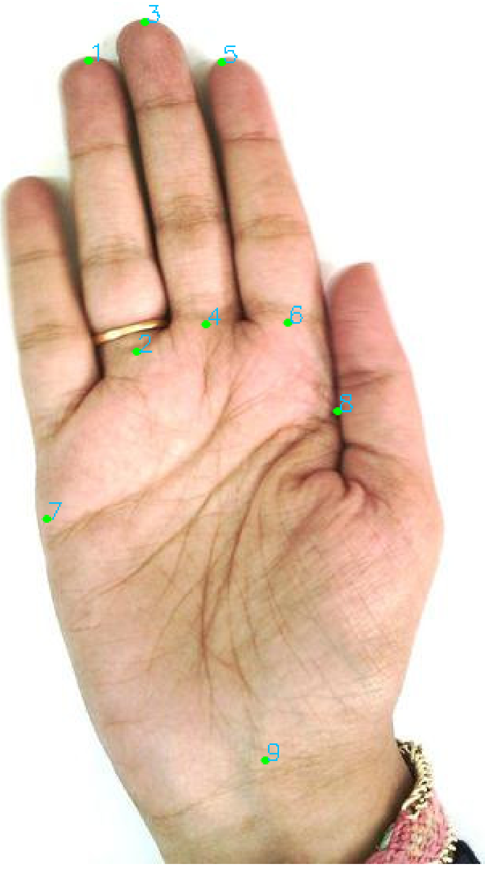
		

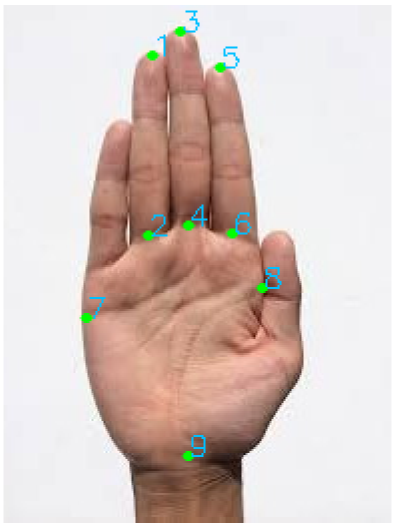
		
		

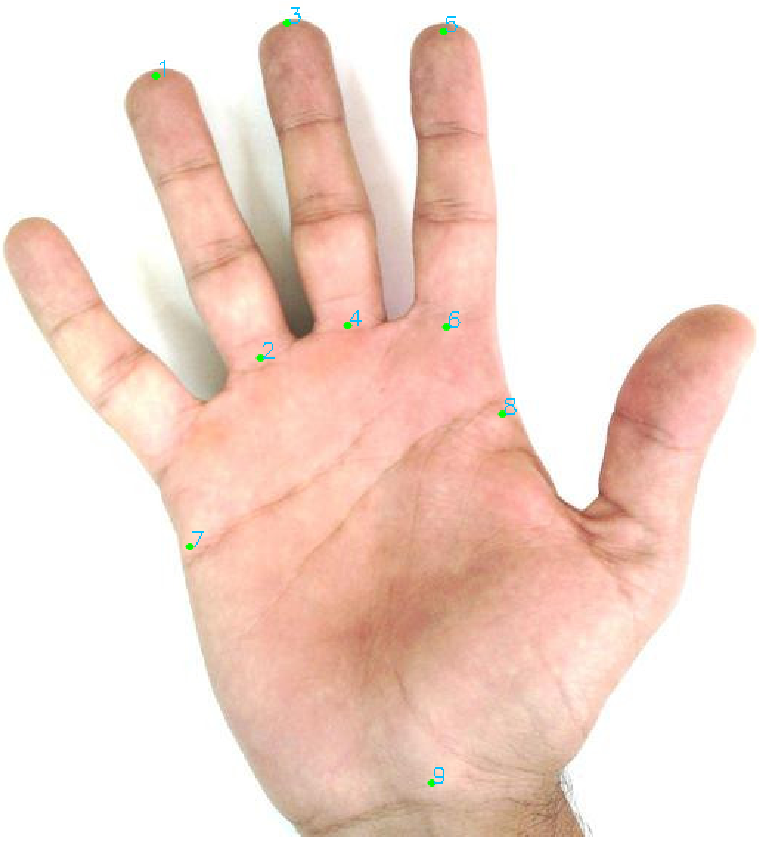

1
2
3
4
5
6
7
8
9


1
2
3
4
5
6
7
8
9
1
2
3
4
5
6
7
8
9

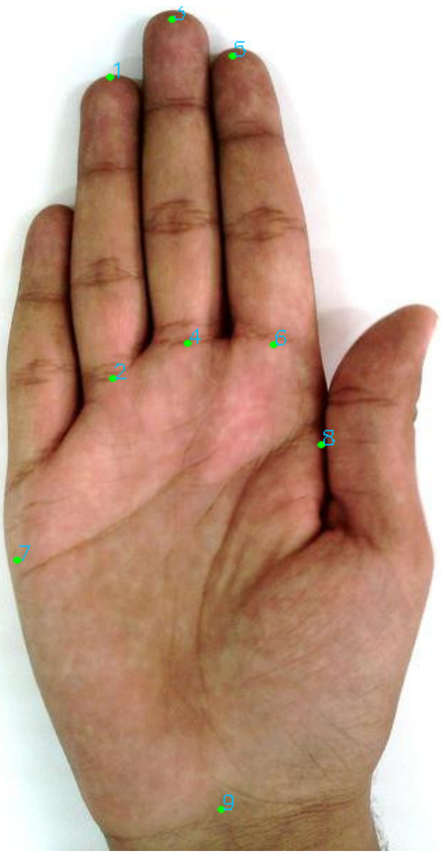

1
2
3
4
5
6
7
8
9


1
2
3
4
5
6
7
8
9

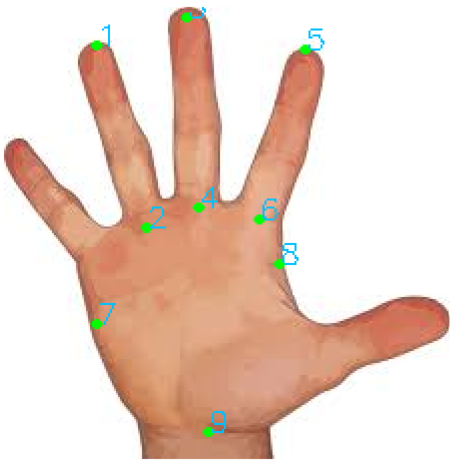

1
2
3
4
5
6
7
8
9

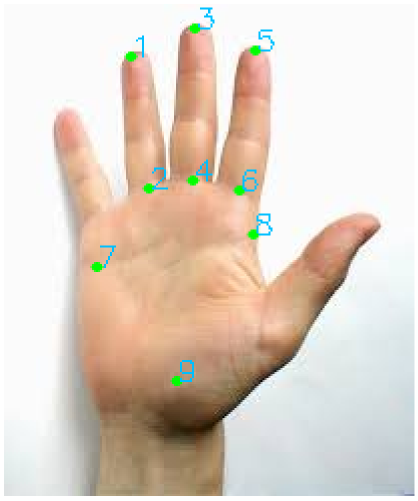


1
2
3
4
5
6
7
8
9



1
2
3
4
5
6
7
8
9





Conclusion and Future Works:

The main objective of this project is to detect 9 landmarks of hands automatically for using I hand research. The hand detector are able to detect hands successfully but hand landmarks detector sometimes results in landmark positions that deviate from the real positions. Nevertheless, success of both detectors are really promising. As for the future work, I will design a deep learning model for hand detection, I will try to improve the accuracy of hand landmarks detector by obtaining hand images from more individuals and finally I will develop a automatic gender detector based on hand landmarks. In addition to these, hand detector is for detecting right hands but it also detects left hands too. So to make the hand detector more robust, I will retrain the detector by annotating the left images as negative examples.

References:

[1] http://ieeexplore.ieee.org/document/7033184/
[2] https://sites.google.com/view/11khands
[3] https://www.ece.nus.edu.sg/stfpage/elepv/NUS-HandSet/
[4] https://www.mutah.edu.jo/biometrix/hand-images-databases.html
[5] http://ieeexplore.ieee.org/document/1467360/
[6] http://ieeexplore.ieee.org/abstract/document/6909637/