Unknown Face Recognition and Tracking

Zeb Barry
University of Canterbury
Christchurch, New Zealand
zba24@uclive.ac.nz

Supervisor: Dr Richard Green
University of Canterbury
Christchurch, New Zealand
richard.green@canterbury.ac.nz

***Abstract*—This paper presents a method of recognising and tracking previously unknown faces in real-time video using the 720-pixel MacBook Pro Facetime HD camera. To identify faces in an image a Histogram of Oriented Gradients (HOG) and Support Vector Machine (SVM) based method was used. Embeddings for each face were then generated using a modified ResNet network. These embeddings were then compared using the Euclidean distance and if the distance was below a set threshold of 0.6, then the two embeddings were considered a match. If no match was found, then the embedding was recorded as a new face and used to recognise that face in later images. This method was tested on the Yale Faces dataset and had an accuracy of 98.18 %. In order to improve the runtime of the method, the image was reduced in size. The optimal image reduction was 60 % of the original size as this had no significant effect on the accuracy of the method but was able to achieve 7.7 fps with no display.**

Keywords—Facial Recognition, HOG, Facial Landmarks, Facial Identification, ResNet, dlib

I. Introduction

Facial recognition has long been a popular research topic in computer vision with many different applications from security to social networking and recently with contact tracing [1] [2]. It is one of the least intrusive biometrics and is estimated to be worth over $9.6 billion by 2022, with a compound annual growth rate (CAGR) of 21.3% [3]. However, the technique does not lack challenges. By nature, we are all different and our faces are no exception. They range greatly in their shape, facial hair, eye colour, complexion and many more ways. While this makes it easier to identify one from another it presents a challenge to identify a face from the background [4]. This is also compounded by the occlusion of faces by hair and other obstacles, the resolution of the image, lighting, and the angle of the face to the camera. The final challenge is implementing the technique efficiently enough that it meets the performance requirements to run in real time.

In spite of all this there have been significant advances in recent years using neural networks with several methods proving more accurate than humans with an accuracy of 99.87% [5]. The limitations to these methods, however, is that they require large datasets of training images for each person. This is not practical for many applications such as attendance recording and tracking consumers in shops as there would not be any sample images.

This paper seeks a solution to this problem recording every face in an image and comparing it to all previously recorded images to find a match. If no match is found, then the face is added to the set of recorded images and used to recognise the face in later images. This allows for continuous recording and tracking of people and their movement. While not the primary goal, the method is also optimised to run in real-time.

II. Related Work

A. Face Identification

In order to identify a face in an image there are two common methods, either a Viola Jones Haar Cascade Classifier (VJ) or a Histogram of Oriented Gradients (HOG). The VJ method uses Haar features to scan an image and if sufficient features are found then the current scan window is said to contain a face. While this method is effective at identifying clean images of faces facing the camera in real time, it struggles with occlusion such as glasses and can produce false positives [6]. Using a grey scale image, the HOG method operates by analysing each pixel and its surrounding pixels to generate a gradient for that pixel [7]. These gradients are then averaged over a 16x16 pixel range to produce a simplified output. This resulting image is then compared to another HOG pattern of a "standard" face and regions that have similar gradient patterns are identified as faces [8]. This gradient method is very effective as it reduces the impact of lighting as the individual pixel values no longer matter, instead the change in pixel value is considered [8]. An example output of a HOG based detector is shown in Fig. 1 demonstrating how the output from the image is matched against a general pattern of a face.

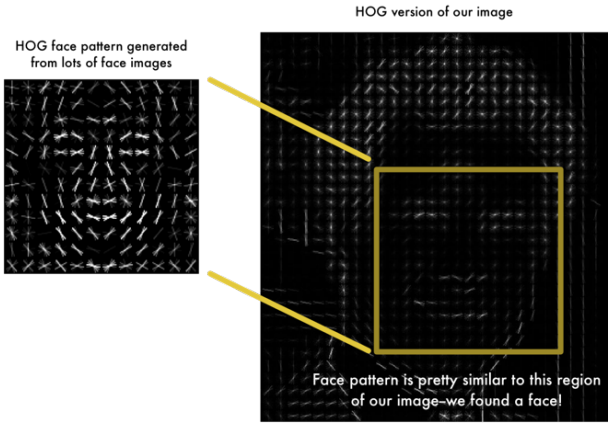


Fig. 1: Comparison of general HOG pattern of a face to example HOG output generated by a face [8].

B. Facial Landmark Detection

Once the region of interest (ROI) around each face has been found, a series of facial landmarks are located. The most common implementation of this was design by J. S. Vahid Kazemi and uses the Ensemble of Regression Trees method which can predict these landmarks within milliseconds [9]. These landmarks are then used to align each face so that the eyes and nose are in similar locations for easier comparison. There are two common models using either a 68-point model or a 5-point model, both included in the dlib library by Davis King [10]. The 68-point model provides the ability to track the jaw, mouth and other features. Conversely, the 5-point model

only tracks the inside and outer corners of each eye and the tip of the nose, shown in Fig. 2. This method is ~10% faster than the 68-point method and still provides sufficient information to align the image [11].

Fig. 2: Comparison of 68-point facial landmark model (top) to the 5-point model (bottom) showing landmark locations [11].

C. Embedding Generation

In order to compare each face, they are converted into a series of 128 measurements called embeddings. Researchers at Google proposed the use of a Convolutional Neural Network to develop these embeddings using a "triplet loss" function [12]. A "triplet loss" training step is the embeddings of two images of the same person and one image of a different person are created. The algorithm is then trained to make the embeddings for the same person closer together and the distance between the two different people greater. The equation for this triplet loss is shown below (1).

$$Loss = \sum_{l=1}^{N}\left[||f_i^a - f_i^p||_2^2 - ||f_i^a - f_i^n||_2^2 + \alpha\right]_+ \quad (1)$$

While the exact FaceNet model is not open source, there has been a lot of other research into the field and one of the most successful attempts is a modification of a ResNet network with 29 convolutional layers that was trained on over 3 million images. This model was developed by Davis King, author of the widely used dlib library and has an accuracy of 99.38% when used on the LFW benchmark for facial verification [13].

D. Recognition

Once the embeddings have been generated, they are used to compare and differentiate faces. Primarily this has been achieved through the use of machine learning algorithms such as Linear Support Vector Machines (LSVM) or Convolutional Neural Networks (CNN) [8] [14]. These methods involve training the algorithm on a set of sample images for each person so that the algorithm is able to correctly classify each face in the dataset. It is then tested on a separate test set of images of the same people to determine its accuracy. These methods are very accurate at over 99.87% accuracy on the widely used Labelled Faces in the Wild (LFW) dataset, better than human results on the same dataset (99.2%) [5]. However, this method is not suitable for recognising and tracking previously unknown people as the algorithm has not been trained to classify them. An alternative method is to compare the embeddings of each face using the Euclidean distance between them. From this the embeddings can be considered a match if the Euclidean distance is below a set threshold. This method is presented by Adrian Rosebrock, however his method requires comparing the embedding from the test image against a large set of training embeddings of the same person and if there are sufficient matches then the person is recognised [15]. This method will also not work for tracking previously unknown faces as their embeddings will not be in the training set.

III. Proposed Method

A. System and Applications

The proposed method involves implementing automatic facial recognition of unknown faces on a 2019 MacBook Pro with a 2.4 GHz Quad-Core Intel i5 Processor using the built-in 720-pixel, 1.2-megapixel (MP), 30 frames per second (FPS) Facetime camera. The method was designed to be run in real-time and therefore operation runtime is a significant factor. The software was developed using Python 3.7.6 and used OpenCV 4.1.2 to record and process images with no CUDA or GPU support. The program was run using the command line and edited in Sublime Text.

B. Identification and Embedding

To detect faces in an image, a model using a combination of HOG and a SVM designed by Davis King was used to identify faces in the image was used as it was more accurate than Haar Cascade methods and over 10 times faster than a CNN method (0.2 s compared to 3.3 s per image) [6] [14]. The faces were then aligned using the 5-point facial landmark ensemble of Regression Trees model described in section 3.2 as it provided sufficient detail to correctly align the image and able to run 10% faster than the alternative 68-point model [11] [9]. Once aligned, each face was then converted into an 128d embedding modified ResNet network but with an alteration such that the model does not resample the image 100 times, as was used in the LFW benchmark test. King provided functionality for this adjustment when designing the model and it increases the speed of the model by a factor of 100 while maintaining an accuracy of 99.13% on the LFW dataset. [16].

C. Recognition

In order to identify each embedding, they are compared to other previously identified embeddings in a matching process. This process is similar the one developed by Adam Geitgey where the Euclidean distance between two embeddings is calculated using the Frobenius norm [17] [18]. The Euclidean distance between two points is calculated for each of the 128 dimensions in the embeddings (2). Then the Frobenius norm of the resulting matrix is calculated (3), resulting in a single value that represents how similar the two embeddings are. The smaller the resulting distance, the higher the similarity of the two embeddings and the likelihood that they are of the same person. The threshold value used to determine if the encodings are a match or not is a parameter that can be tuned to improve accuracy, however, Gietgey recommends a value of 0.6 (a perfect match would be 0.0).

$$Euclidean\ Distance(a,\ b) = \sqrt{\sum_{i=1}^{n}(q_i - p_i)^2} \quad (2)$$

$$||A||_F = \sqrt{\sum_{i,j} |a_{ij}|^2} \quad (3)$$

However, this method still requires the use of a set of sample images of each person in order to identify each face. In the method developed by Aidan Rosebrock using Geitgey's work, if the resulting distance between an embedding and the embeddings from the sample images is above 0.6, it is simply labelled "Unknown". To remedy this, if an embedding does not match any of the existing embeddings in the sample dataset, it is considered "new" and added to the dataset along with a unique ID. Therefore, if this "new" face reappears in later images, its embedding will match the embedding in the dataset, allowing the face to be recognised and identified with its unique ID. This solves the issue of being unable to differentiate between several "Unknown" faces in an image and allows the facial recognition and tracking of people with no sample set of images, addressing the shortcomings of both Rosebrock's method and other CNN based methods.

D. Runtime Optimisation

Due to the linear brute force method implemented to match embeddings, the runtime of the method will increase significantly with an increase of unique faces. In order to reduce the impact of this several techniques were implemented to reduce the processor load. Firstly, the images were downsized to reduce the pixels per frame and therefore the calculations per frame. However, this has an inverse effect on the accuracy of the method as it reduces the quality of the image. This would increase the difficulty of identifying and recognising faces at a distance. To determine the optimal size reduction that still provides a high level of accuracy the method was tested at a variety of size reductions. These were all a percentage of the original 720p image as follows: 100%, 75%, 50%, 40%, 30%, and 25%. The testing method involved analysing images of a face at increasing distances from the camera and testing if the method was able to recognise the face based of a reference image. The distances tested ranged from 0.5 m to 2.1 m with a 0.2 m interval. The 0.5 m image was used as a reference to compare against the embeddings from the other images. This was tested with only one face in the frame and a fixed set of 10 previously identified faces so that the level of computation per frame would be consistent. The framerate of the method was computed by running the program on a pre-recorded video and then recording the number of times the recognition method was run (4). The framerate was tested both with and without the output from the camera displayed to determine the impact of processing the image and labelling the faces.

$$FPS = \frac{Frames\ Analysed}{Total\ Time} \quad (4)$$

E. Evaluation

In order to evaluate the accuracy of the recognition method, the program was tested on the Yale Faces dataset, consisting of 15 subjects with 11 greyscale photos each. This provided a wide variety of input varying in age, skin colour, hair colour and each of the 11 photos was different with lighting changes, glasses and facial expressions. The exact list of photo types is as follows: glasses, no glasses, normal, happy, sad, sleepy, surprised, winking, centred light, right light, left light. This dataset was used as it was anticipated that the method would struggle with large datasets due to the linear search method required to find a match. However, it still provided a wide range of environments and sample images. The major downfall with this dataset was its limited gender variety with only one female subject. To test the methods ability to differentiate new faces and recognise previously seen ones, the program was given each image individually and told to recognise any faces in the image. This meant that it had no reference to how many individual subjects there were, nor how many faces were in each image. If the face was not a match for any of the previously recorded faces then it was recorded as a new face and used to identify any recurrences in later images as described in section 3.C.

IV. Results

A. Recognition

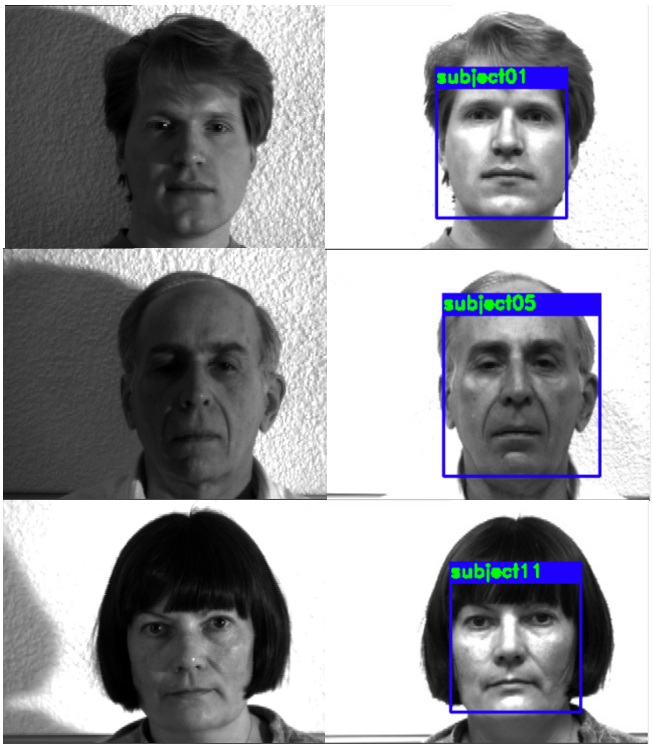
The results from the recognition test on the Yale Faces dataset for each subject are shown in TABLE I using the true positive rate, for an average accuracy of 98.18% (5). The true positive rate is determined by the number of correctly identified images over the total images of that subject.

$$True\ Positive\ Rate = \frac{Correctly\ Identified\ Images}{Total\ Images} \quad (5)$$

There has been previous research on this dataset by Patrick Wang, applying multiple different face recognition techniques such as eigenfaces, support vector machines (SVM) and independent component analysis (ICA). Wang determined that the ICA method was the most accurate with an accuracy of 93.13%, with the SVM method next at 91.1% [19]. This means that the proposed method has achieved a higher accuracy than all of the method's analysed in Wang's research at 98.18% compared to 93.13%, a difference of 5.05%. Although Wang's research was published in 2010, there have been later attempts on the same dataset such as in 2013 by Pradipta Maji et al., where they also used the Euclidean distance to compare embeddings [20]. However, they were only able to achieve an accuracy of 92.22% and so this was not considered the current benchmark. The three images that this method was unable to recognise were of subjects 1, 5 and 11. Subject 1 was an adult male, subject 5 was an elderly male and subject 11 was a middle-aged female. This shows minimal similarity between the three subjects; however, all three images were of the "right light" type. In this type there is a bright light pointed at the face from the right side, casting shadows across the left side of the image. These images failed due to the HOG based detector failing to identify a face in the image. These three images are shown alongside the "normal" image of each subject in Fig. 3. This means that of the 162 images where a face was detected, the proposed method was able to correctly identify 100% of them.

TABLE I: Accuracy results of proposed method for facial recognition when tested on Yale Faces dataset.

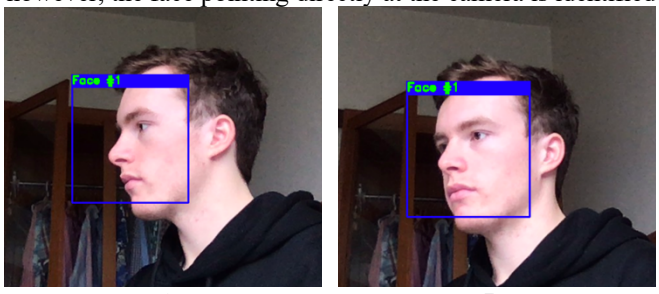
Subjects	*True Positive Rate*
2-4, 6-10, 12-15 (12 total)	100.00%
1, 5, 11 (3 total)	90.91%
Average	**98.18%**
Benchmark	93.13%



A B

Fig. 3: Three "right light" images where a face was not detected by the HOG based detector (A) alongside the labelled "normal" images of the same subjects (B). Images are as follows: Subject 1 (top), Subject 5 (middle), Subject 11 (bottom).

Aside from the instances where a face is not detected by the HOG detector, another instance where the method commonly failed to correctly identify a face was when the initial embedding was poor. When a new face enters the frame and the method attempts to recognise it, if no match is found for the embedding, it is saved and used to recognise that same face in later images. However, this method is significantly less effective if the initial embedding recorded is generated from an image where the face is partially obscured or at an angle. An example of this is shown in Fig. 4A, where the initial image that the method analyses is of a face angled at 90° to the camera such that only half of the face is visible. The face is detected, embedding generated and then saved. Following this, two more images are then analysed, one with a face at 45° to the camera and another facing directly at the camera, Fig. 4B and Fig. 5 respectively. The face angled at 45° to the camera is correctly recognised as the same as in Fig. 4A, however, the face pointing directly at the camera is identified as a different person. This is because the embeddings of the face angled sideways and the face pointing forwards are sufficiently different such that the Euclidean difference between the two is above the threshold of 0.6.



A B

Fig. 4: A) Results from method showing how face angled at 90° is identified. B) Results from method showing how face at 45° is recognised as the same face.

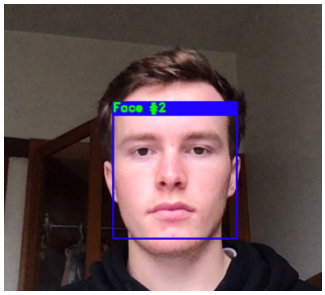


Fig. 5: Image of face pointing forwards showing how it is identified as a different face than Fig. 4A

B. Runtime Optimisation

In order to determine the optimal downsizing percentage of the original 720p image from the 2019 MacBook Pro Facetime HD camera, two methods were used. Firstly, a series of images of the same subject were taken in well-lit conditions with their face at an increasing distance from the camera ranging from 0.5-2.1 m in 0.2 m intervals. These images were then run through the program and encodings generated for any faces identified in the images. As there was only one face in the image and it is in all images, the expected output would be only one face identified and recognised as the same person. The first image run through the program was with the face located 0.5 m from the camera. As this was a previously unknown face, its embedding was recorded and used to recognise the same face in the following images. The Euclidean distance between the face at 0.5 m and the faces at greater distances for the different image sizes can be seen in Fig. 6. The threshold for recognising a face used in this method was 0.6. From this it is clear that all images where a face was identified, it was correctly recognised as all distances are below the threshold of 0.6. However, some of the smaller image sizes (25%, 30% and 40%) were not able to identify a face in the image as the distance increased (failed at 1.3 m, 1.5 m and 2.1 m respectively). This is why there is no values shown for these image sizes at the greater distances. Following this it was determined that any image size below 50 % is not viable as it reduced the level of detail in the image such that the method was unable to identify any faces. It can also be noted from Fig. 6 that while the downsizing of the image did impact the embedding distance calculated, the biggest impact came from the increase in distance to the camera. This is clear from the Euclidean distance for each of the image sizes, as they are very similar in each grouping, but increase significantly from one grouping to the next.

Once the viable image sizes for reliable face detection had been determined, they were each tested on a pre-recorded video to determine the average rate at which the method was able to process frames. This pre-recorded video was recorded on the 720p Facetime HD camera in well-light conditions similar to the distance tests. It involved a person moving around the frame, starting close to the camera and moving further backwards. The person also stood still at 1 m from the camera and turned the head left, right, up and down, to test the effects of angle on recognition. This video was then analysed using the method described in this report and any identified faces recognised. To determine the effects of downsizing the video on frames per second (fps) processed, the video was

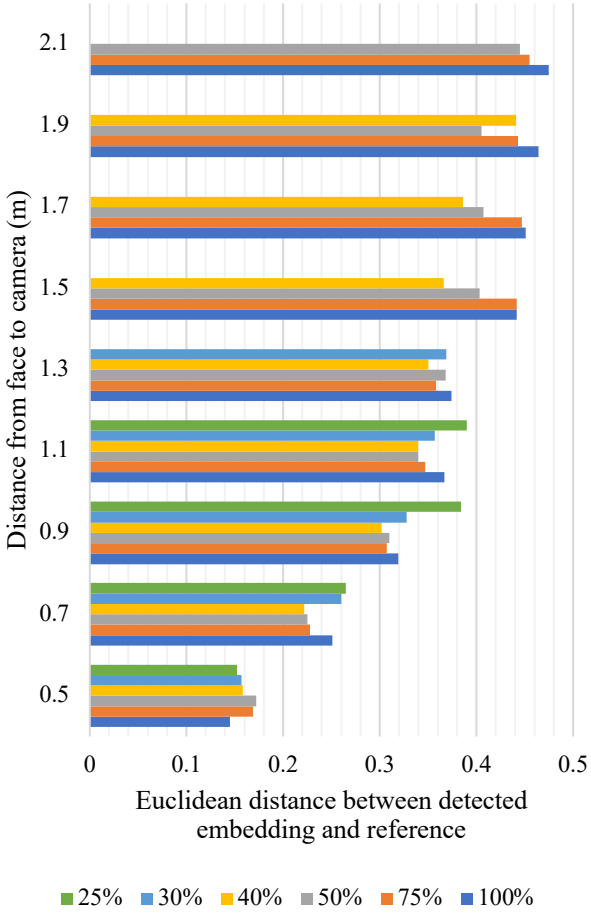


Fig. 6: Euclidean distance between sample face at 0.5 m and face at increasing distances for different image sizes (distance < 0.6 is a match).

analysed at image sizes ranging from 100 % (original size) to 50 %, as these were the sizes determined viable in the distance test. As displaying the video is also processed by the same program that analyses each frame, the method was tested first without displaying the video, and then with the video displayed at 50 % of the original size so that the computations were consistent for each test. As there was only one face in the video, the accuracy of detection method in each video was determined using the true positive rate on all frames where a face was detected (5). The results of these tests are recorded in TABLE II. These results show that the image could be reduced to 60 % of its original size without any significant change in the accuracy of the method. Even at a 50 % reduction the accuracy only reduced by 5 % compared to the accuracy at 100 %. However, reducing the image size from 100 % to 50 % increased the frames per second processed by a factor of three, increasing from 3.09 to 10.43 fps (with no image displayed). The impact of displaying the image did increase as the image size decreased, a difference of 0.33 fps at 100 % compared to 3.03 fps at 50 %. While there was no investigation into the reasoning behind this increasing delay there is a potential explanation. As the time required to display the image should be constant for all image sizes, it is always displayed at 50 %, the increasing impact is likely to be due to the increased frame rate. The faster the framerate the more frequently the program has to update the image and therefore the greater the difference between displaying and not displaying the video.

TABLE II: Average frames per second (fps) processed at different image sizes on a test video (with and without displaying the video) and the resulting recognition accuracy.

Image Size (%)	*Average Frames Per Second (fps)*		*Accuracy (%)*
	No Display	*With Display*	
100	3.09	2.76	84
90	3.72	3.25	82
80	4.62	3.98	84
70	5.94	4.82	83
60	7.7	5.87	84
50	10.43	7.40	79

C. Limitations

As mentioned, this method has several limitations. The primary limitation to the described method is the implementation of the recognition method. To recognise a face in an image, its embedding must be compared to the embeddings of all previously recorded unique faces. This linear search method is then repeated for all faces in the image. Therefore, the computation load becomes extremely high when there is a lot of faces in an image or a large dataset of previously recorded faces. Use of a better search algorithm such as the Fast Library for Approximate Nearest Neighbours for matching embeddings would significantly improve this failing [21]. Secondly, the proposed technique of recording the embeddings of new faces and using them to identify the same face in later images can fail if the original image is of poor quality. For example, in Fig. 4A where only half of the face is visible to the camera. Other possible cases include if the face is a long way away and therefore there is little detail as well as if the face is partially obscured.

V. Conclusion

This paper presented a method of recognising previously unknown faces in a real-time on a 2019 MacBook Pro using a HOG based face detector and modified ResNet network to create embeddings. The proposed method compares facial embeddings using the Euclidean distance. Unlike previous methods, this implementation requires no sample images to be able to recognise and track faces. It achieved an accuracy of 98.18 % on the Yale Faces dataset, outperforming existing recognition methods on the same dataset by 5.05 % [19], [20]. By reducing the size of the image to 60% of the original size, this method was able to operate at 7.7 fps with no display without any significant impact on its accuracy.

A. Future Work

This method of facial recognition and tracking is a promising approach for tracking unknown faces with accuracy comparable to existing methods. However, as described there are limitations to the current design and potential improvements to reduce these shortfalls are listed below.

1) Embedding matching

The current implementation involves matching an embedding by using a linear search on all unique recorded embeddings. This method becomes computationally more expensive as the number of faces in an image and the number of previously identified faces increases. In future work, this could be improved by using a more efficient search method or

library such as the FLANN library for approximate nearest neighbour searches.

2) Recording initial embeddings

As this method is able to track new faces by recording the embedding generated by the first image of the face, the ability of the method to recognise that face in later images is dependent on the quality of that first embedding. If the first embedding is generated from an image where the face is at an angle, partially obscured or far away, the method may fail to recognise that face when it is facing directly at the camera or from another side. In future research, a method of improving the recorded embedding based on later images of the same face could be investigated.

VI. References

[1] M. D. Kelly, "Visual Identification of People by Computer," Stanford University California Department of Computer Science, 1970.

[2] Z. X. Z. Z. Thomas Huang, "Face Recognition Applications," in Handbook of Face Recognition, London, Springer, 2011, pp. 617-638.

[3] R. Singh, "Facial Recognition Market by Technology (2D Facial Recognition, 3D Facial Recognition and Facial Analytics), Component (Hardware and Software) and Application (Homeland Security, Criminal Investigation, ID Management, Physical Security, Intelligent Signag," Allied Market Research, 2016.

[4] A. Piper, "About Face: The Risks and Challenges of Facial Recognition Technology," Risk Management, 1 November 2019.

[5] University of Massachusetts, "Labelled Faces in the Wild: Results," [Online]. Available: http://vis-www.cs.umass.edu/lfw/results.html#ozf. [Accessed 18 May 2020].

[6] C. a. A. R. a. P. D. a. D. I. a. D. H. a. M. I. Rahmad, "Comparison of Viola-Jones Haar Cascade Classifier and Histogram of Oriented Gradients (HOG) for face detection," IOP Conference Series: Materials Science and Engineering, vol. 732, pp. 012-038, 2020.

[7] M. M. Dehshibi, "HOG and LBP: Towards a Robust Face Recognition System," in International Conference on Digital Information Management, Tehran, 2015.

[8] A. Geitgey, Machine Learning is Fun! Part 4: Modern Face Recognition with Deep Learning, Medium, 2016.

[9] J. S. V. Kazemi, "Face Alignment with an Ensemble of Regression Trees," in The IEEE Conference on CVPR, 2014.

[10] D. King, "Dlib C++ Library," 14 December 2019. [Online]. Available: http://dlib.net. [Accessed 14 April 2020].

[11] A. Rosebrock, "(Faster) Facial landmark detector with dlib," PyImageSearch, 2018.

[12] D. K. J. P. Florian Schroff, "FaceNet: A Unified Embedding for Face Recognition and Clustering," Google Inc., 2015.

[13] D. King, "High Quality Face Recognition with Deep Metric Learning," Blogger, 12 February 2017. [Online]. Available: http://blog.dlib.net/2017/02/high-quality-face-recognition-with-deep.html. [Accessed 19 May 2020].

[14] A. Ponnusamy, "CNN based face detector from dlib," Towards Data Science, 2018.

[15] A. Rosebrock, "Face recognition with OpenCV, Python, and deep learning," PyImageSearch, 18 June 2018. [Online]. Available: https://www.pyimagesearch.com/2018/06/18/face-recognition-with-opencv-python-and-deep-learning/. [Accessed 15 April 2020].

[16] D. King, "GitHub," 19 February 2019. [Online]. Available: https://github.com/davisking/dlib/blob/master/python_examples/face_recognition.py. [Accessed 19 May 2020].

[17] A. Gietgey, "GitHub," 19 December 2019. [Online]. Available: https://github.com/ageitgey/face_recognition/blob/master/face_recognition/api.py. [Accessed 19 Mau 2020].

[18] The SciPy Community, "numpy.linalg.norm," 5 February 2020. [Online]. Available: https://numpy.org/doc/1.18/reference/generated/numpy.linalg.norm.html. [Accessed 19 May 2020].

[19] P. S.-P. Wang, "Experiments," in Pattern Recognition and Machine Vision, Aalborg, River Publishers, 2010, pp. 142-145.

[20] A. G. M. N. M. K. G. S. K. P. Pradipta Maji, "Monogenic Scale Space Based Facial Recognition," in Pattern Recognition and Machine Intelligence: 5th International Conference, Kolkata, Springer, 2013, pp. 181-165.

[21] D. L. Marius Muja, "FLANN - Fast Library for Approximate Nearest Neighbors," 24 January 2013. [Online]. Available: https://www.cs.ubc.ca/research/flann/uploads/FLANN/flann_manual-1.8.4.pdf. [Accessed 31 May 2020].