Delay compensation in mobile face tracking

G. van Spil
Student Bachelor Electrical Engineering
University of Twente, Enschede, The Netherlands
Supervisor: Dr.Ir. L.J. Spreeuwers

***Abstract*—Facial landmark localization is not possible in real-time using only facial landmark detectors. A tracker can be used to follow the landmarks in real-time. When a tracker is used the computation time of the landmark detector still has an influence on the results of the tracker since it will use older data. In this paper a method is proposed to handle this delay that is created by the detector. A buffer will be used to store images during the detection time, which will then be used by a tracking algorithm to catch up with the real-time incoming frames. An experiment has been performed to compare the performance of this method with other methods. The proposed method does outperform the other methods in some occasions.**

***Keywords*—Facial landmark tracking, real-time**

I. Introduction

Face recognition is a subject that has been studied very frequently during the past decades. It is used in a lot of instances, like identification. For accurate face recognition the face has to be normalized to make them comparable. In order to do this facial landmarks have to be localized. This localization is a slow process when a detector without prior knowledge is used for every frame. A study has been performed to research the feasibility of using tracking for facial landmark detection [1]. This paper focused purely on selecting the best tracking algorithm and did therefore not take into account the effects of the non-ideal facial landmark detector.

When using tracking for facial landmarks the detector is still of great influence. It does not only determine the maximum achievable accuracy with its results, but it also takes some time to localize the facial landmarks and therefore introduces a delay between acquiring the image and outputting the results. This delay will have an influence on the performance of the tracker and therefore the main questions are:

1) What are the effects of this delay?
2) What is the best way to handle the delay time created by the facial landmark detector.

The aim of this paper is to study the effects of the delay from the detector and to propose a solution in order to reduce the effects of this delay on the tracking accuracy.

The remainder of the paper is organized as follows: in section II related work on tracking algorithms, delay handling and detection algorithms is discussed. In section III a solution is proposed to handle the delay from the detector. The experiment to compare the tracking approaches is discussed in section IV. This is followed by section V were the results are presented. In section VI these results are discussed. Finally the paper will be concluded in section VII.

II. Related work

A. Tracking algorithms

A study has been performed on different tracking algorithms by van Wettum [1]. In this paper the speed, accuracy and robustness of four different tracking algorithms (Lucas-Kanade point tracker (LK), Discriminative Scale Space Tracker (DSST), Kernelized Correlation Filters (KCF), Structured Output Tracking with Kernels (STRUCK)) were tested. The paper concluded that the Lucas-Kanade is the fastest tracking algorithm while also being one of the most accurate tracking algorithms together with the DSST algorithm. Since the Lucas-Kanade tracker is able to perform in realtime it was deemed the best tracking algorithm.This research also included two state-of-the-art facial landmark detectors of which the DLIB facial landmark detector (DFLD) is selected as the landmark detector because of its accuracy.

B. Delay handling

The focus in this paper is placed on handling the delay that is caused by the time it takes the facial landmark detector. No other literature on this topic was found. Therefore no prior knowledge was used to construct a method to compensate for this delay.

C. Benchmarking

For facial landmark tracking a popular benchmark in the form of the 300-VW database exists [2]. The videos in this database have all been annotated using a 68-point markup. An example of this markup is shown in figure 1. This markup is used by both DFLD and the 300-VW database [2].

D. Lucas-Kanade tracker

The OpenCV library [3], a library for computer vision, includes an implementation of Lucas-Kanade point tracker with pyramids based on [4]. It is an implementation of an optical flow based tracker. The pyramidal structure is added to keep the performance high while it is still able to detect and register larger movements, because the region of interest (ROI) can be kept low.

E. DLIB Facial Landmark Detector

The DLIB [5] Facial Land Detector is based on the classic Histogram of Oriented Gradients [6]. It makes also use of a shape model, which can be trained by the provided tools. In this case the in [6] suggested shape model is used.

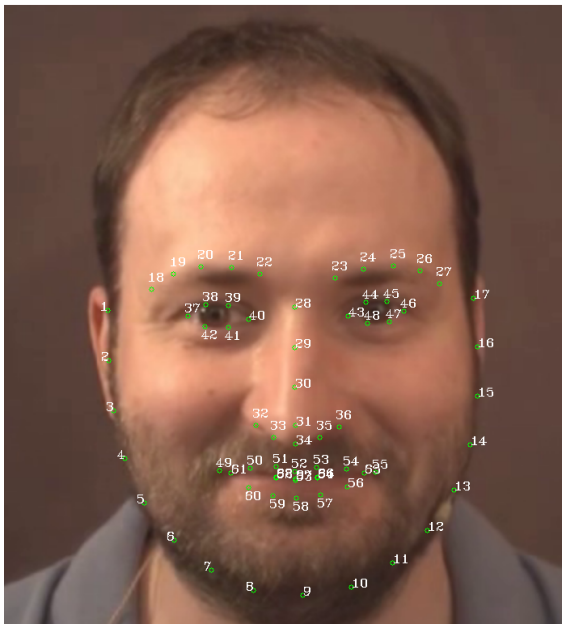


Figure 1. 68 point markup as used by both the 300-VW database and DLIB shown on a frame from the 300-VW database [2]

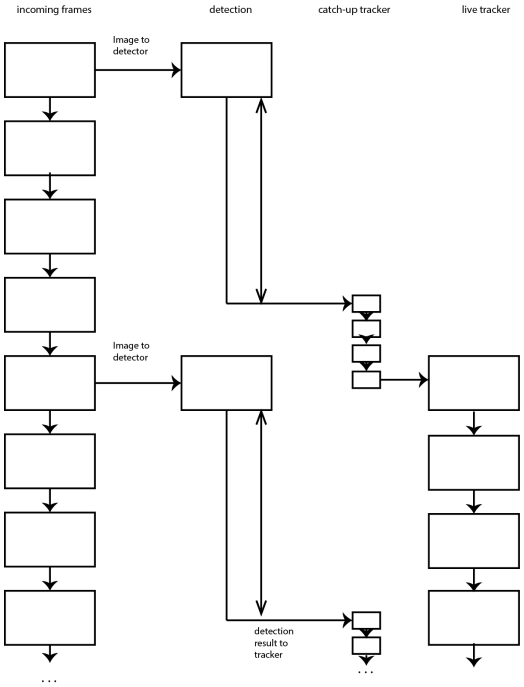


Figure 2. Time-based schematic of the frame processing with the delay compensating system.

III. Proposed solution

The main purpose of the proposed solution is to handle the delay created by the time it takes for the detector to find the facial landmarks. In order to do this the following is proposed: during the detection time frames will be stored in a buffer. When the detection is finished the tracker will use the frames in the buffer to catch up. When frames in the buffer have been used to catch up they are no longer necessary and are therefore erased. As soon as the tracking mechanism has caught up with the real-time frames it will continue tracking to label the landmarks on the new incoming frames. Meanwhile the detector will start again and the frames will again be put into the buffer. This process will repeat itself indefinitely.

In figure 2 a schematic is shown of the processing flow of the delay compensating system. A frame will be fed into the facial landmark detector, which will take the time of multiple frames to find the landmarks. During the detection a buffer will be filled with frames that are acquired in the meantime. Once the detector is finished the result is fed into the tracking system which will use the framed in the buffer to catch-up with the real time frames. Once the catch-up tracker has caught up with the real-time frames the tracking is handed off to a tracker that will now track only the real-time frames. At this point the detector will also start detection using a new frame. This process will repeat continuously.

The benefit of this method is that all frames will be used for the tracking, this prevents large differences between the frames. This method is possible because the tracking is fast enough to process multiple buffered frames in the time that one frame is acquired from the camera [1]. The downside could be that since more tracking operations are done more drift could be introduced by the tracking algorithm

IV. Experiment

In this section the performed experiments are described. Different methods have been tested to compare the performance and to determine to what extent the tracking with delay compensation has an advantage. Three methods will be compared, they will be as follows:

1) DFLD with added delay: this will simulate the facial landmark detection without tracking. The delay is added to simulate the computational time it takes to find the facial landmarks.

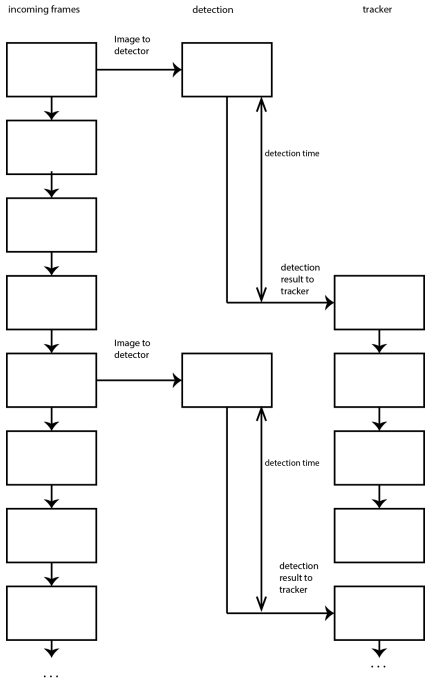


Figure 3. Time based schematic of the frame processing for the system without delay compensation.

2) DFLD and tracking: this will use DFLD as above but will now start tracking as soon as the result of the landmark detector is ready. A schematic view of this method can be seen in figure 3.
3) The proposed method.

A. Database

The used database for the evaluation is the 300-VW database [2]. This databases consists of videos with people in various conditions. Category 1 consists of people in well-lit conditions with few occlusions. Category 2 videos contain videos in unconstrained conditions (for instance poorly lit or over-exposed videos) but with no large occlusions. Category 3 videos are completely unconstrained videos and can therefore also have large occlusions. To limit the size of the experiment a subset of 12 videos were selected. This includes 8 videos from category 1 and 4 from category 2. These categories are chosen because the intended application do not have large occlusions but can have poor illumination conditions. All chosen videos have the face of the person relatively large in the frame.

B. Test method

The performance of the tracking methods will be compared using a framework written in C++ which is realized in Microsoft Visual Studio 2017. The experiment is conducted on a pc to simplify the execution of the experiment and to speed up the test procedures. In this framework the three different methods were implemented for comparison. To make the experiment relevant for mobile devices the timing for

Figure 4. Example frame of a category 1 video of the 300-VW database [2].

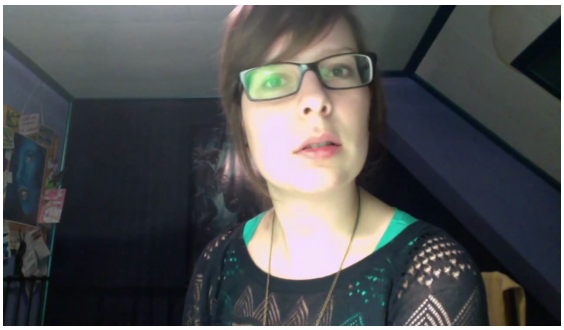


Figure 5. Example frame of a category 2 video of the 300-VW database [2] with non-ideal illumination.

detection speed and track speed can be set. This way the speed of the host device does not influence the results. An Android app was made to test the speed of the mobile device. The app implemented both the DLIB facial landmarks detection and the Lucas-Kanade tracking algorithm.

For the accuracy evaluation the calculated center for both eyes and the mouth are used. The center is calculated by taking the average of the corresponding point for both eyes and the mouth. The used equation is shown in equation 1. Furthermore the nose tip is considered for the accuracy evaluation. An example of the used points can bee seen in figure 6.

$$(x_{average}, y_{average}) = (\frac{\sum_{i=1}^{N} x}{N}, \frac{\sum_{i=1}^{N} y}{N}) \quad (1)$$

The error is calculated using the normalized root mean square method based on the method proposed in [2].

$$NRMSE = \frac{\sqrt{(x^f - x^g)^2 + (y^f - y^g)^2}}{d_{outer}} \quad (2)$$

In equation 2 the root mean square error (NRMSE) of a single landmark point is calculated. The superscripts f and g denote the track result and the ground truth respectively. The inter-ocular distance d_{outer} is the euclidean distance between

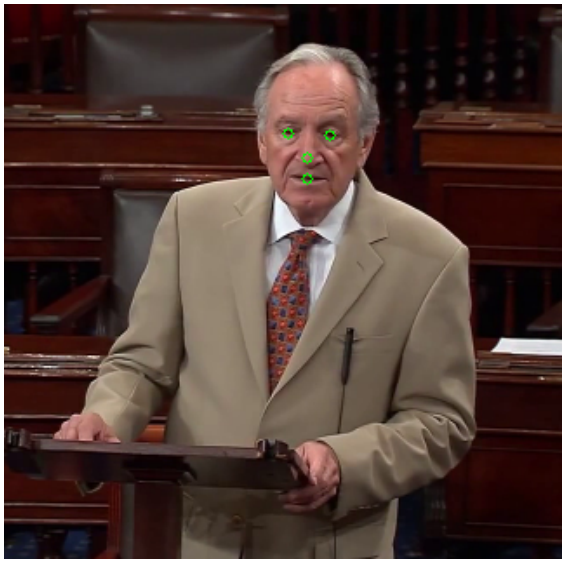


Figure 6. Example of a tracked frame with the centre points drawn on the face. The image is obtained from the 300-VW database [2]

the two outer points of the eyes [2]. The inter ocular distance is calculated using equation 3.

$$d_{outer} = \sqrt{(x_r^g - x_l^g)^2 + (y_r^g - y_l^g)^2} \tag{3}$$

The outer point of the left eye is defined is point 37 in the used markup, while the outer point of the right eye is point 46.

C. Testing parameters

As mentioned earlier the 'timing' parameters will be fixed during the experiment with the framework, an android app was used to determine the speed of a mobile device. From these results viable parameters for the experiment were chosen. This is important since the delay time determines how many frames have to be caught up and will therefore determine the accuracy once the tracker has caught up with the frames. The test was performed on a Samsung Galaxy S7. The frames were scaled to a resolution of 640 by 360 before they were fed to the facial landmark detector and the tracker. This resulted in detection times of 250ms and tracking times of 10ms. Assuming a frame rate of 25 frames/second it takes 7 frames to do a facial landmark detection and 4 frames can be tracked during one frame. These values have been used during the performed experiment. The experiment will also be performed with a detection time of 14 frames to see the impact of varying the delay time on the performance of the tracking methods.

V. Results

The accuracy of all methods are presented using Cumulative error distribution (CED) curves, these are presented on a different page. The curves show the results for all four different landmarks. Figure 7 show the performance of the different methods in category one videos. The performance of the different methods in category two videos is shown in figure 9. The results in figure 7 and 9 are both obtained when the delay time of the detector is set to seven frames. In figures 8 and 10 the results for category one and two are shown respectively but now with a delay time for the detector of fourteen frames.

VI. Discussion

Based on the CED curves the proposed solution works better in some cases. When an upper limit of the normalized error of 0.05 is taken as suggested in [2] the proposed method performs only slightly better than landmark detection without tracking on the nose tip with a delay of seven frames on the category one videos. It does however still outperform tracking without delay compensation in all cases. When a higher maximum error is considered the proposed method does perform better than the other tested methods, but it has to be noted that at this point the result is already quite off. When the length of delay is increased to fourteen frames the proposed method sometimes even performs worsen than tracking without compensation.

In category two videos the proposed method always performs better than the other two tested methods, even when the delay is increased to fourteen frames. It therefore has to be noted that the proposed method performs relatively better in category two videos. Also in this category tracking without any form of the delay compensation performs relatively poor compared to the other solutions.

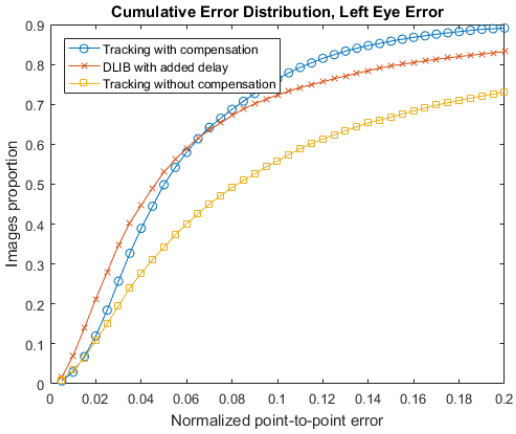
It also clearly noticeable that the accuracy of the proposed method drops when the delay length is increased. This is expected since this method will now perform tracking on more frames to catch up reducing the accuracy because this algorithm does not perform particularly well in long term tracking [1].

VII. Conclusion

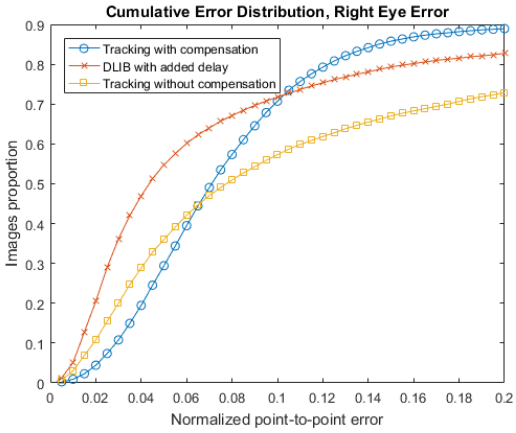
Three methods for facial landmark tracking have been tested. The focus was placed on how these different methods behave when a non-ideal detector is used and therefore a delay.

The time delay created by the facial landmark detector does have an influence on the accuracy of the tracking, lowering the accuracy in all cases when the delay length is increased.

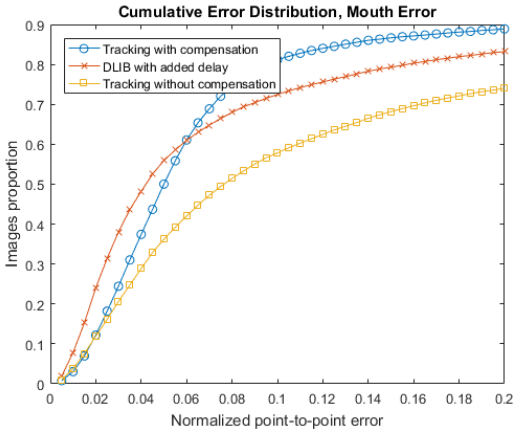
In category 1 videos the proposed solution does not work better than facial landmark detection without tracking, but it does have a better accuracy than tracking without any form of compensation. In category 2 videos the proposed solution does however outperform the other solutions on accuracy. Therefore it can be concluded that the proposed solution works better than tracking without any compensation, while it only has an advantage over no tracking in some conditions.



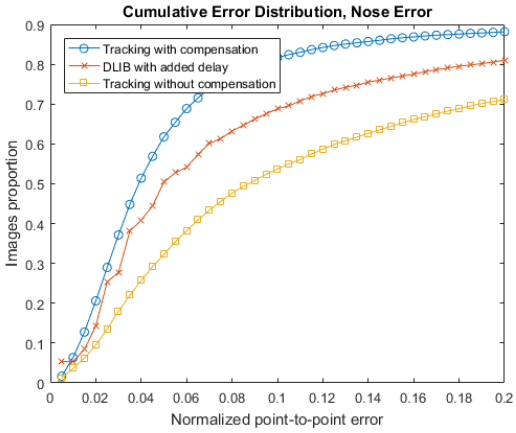
(a) CED curves of the left eye



(b) CED curves of the right eye

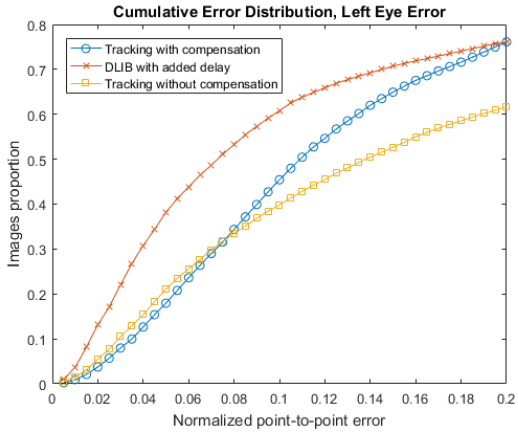


(c) CED curves of the mouth

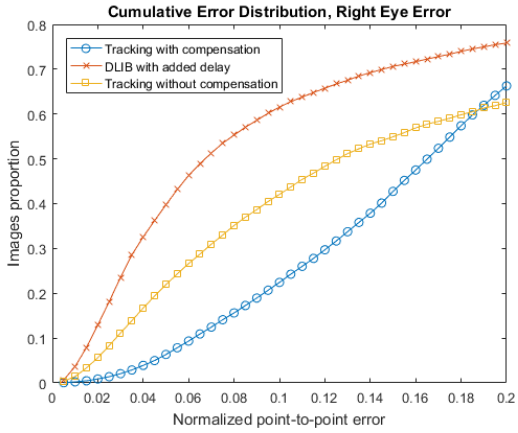


(d) CED curves of the nose

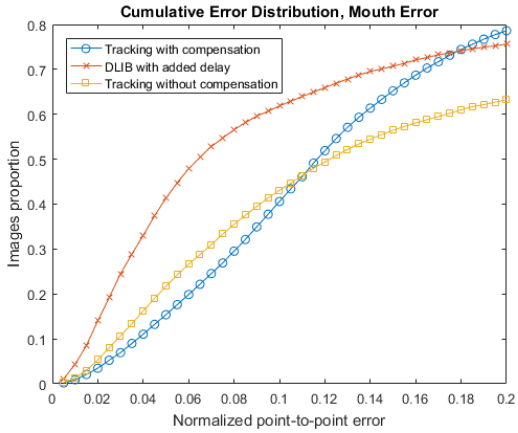
Figure 7. CED curves of the category one videos with a delay of 7 frames



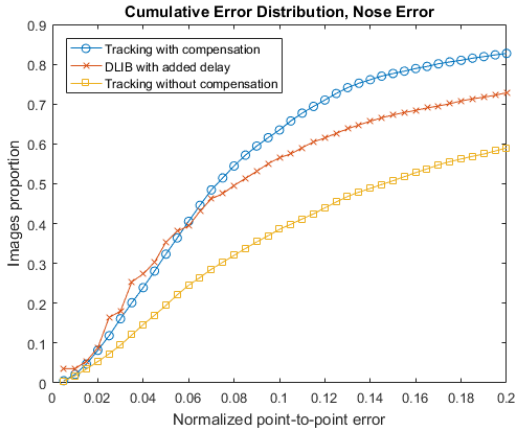
(a) CED curves of the left eye



(b) CED curves of the right eye

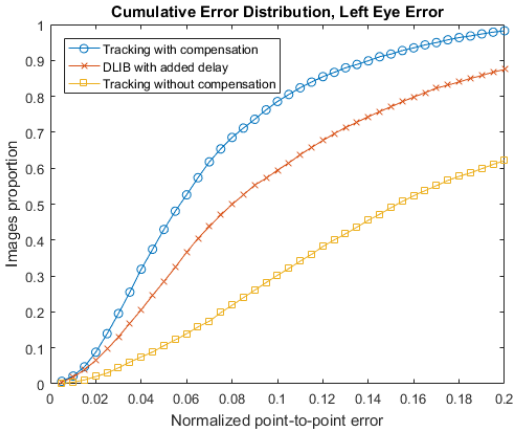


(c) CED curves of the mouth

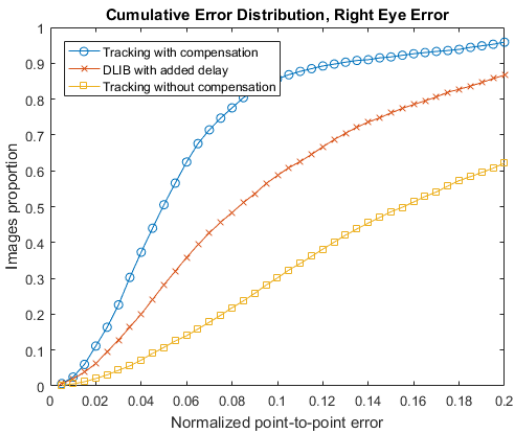


(d) CED curves of the nose

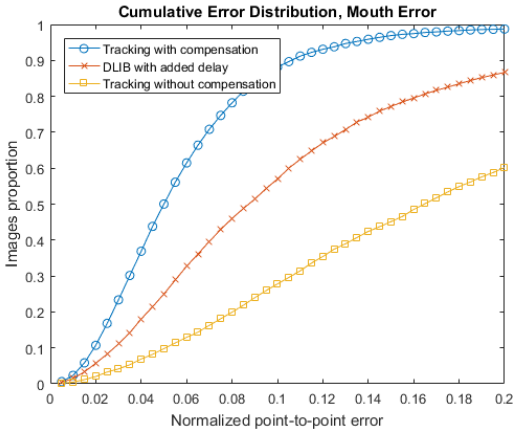
Figure 8. CED curves of the category one videos with a delay of 14 frames



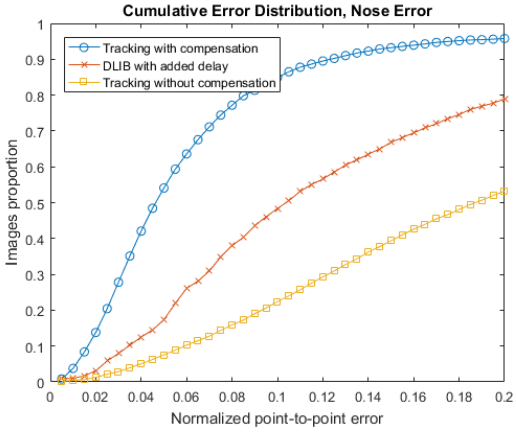
(a) CED curves of the left eye



(b) CED curves of the right eye

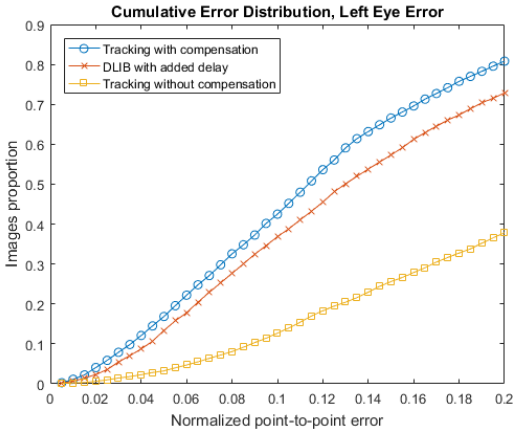


(c) CED curves of the mouth

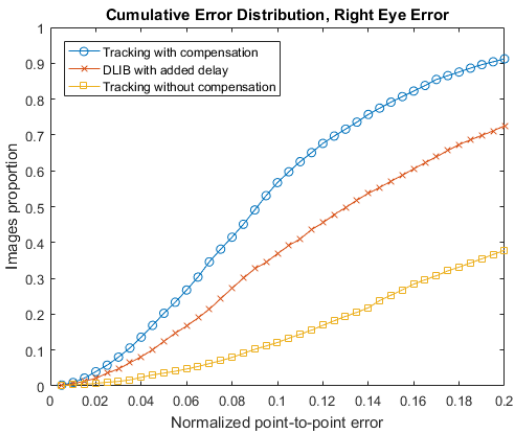


(d) CED curves of the nose

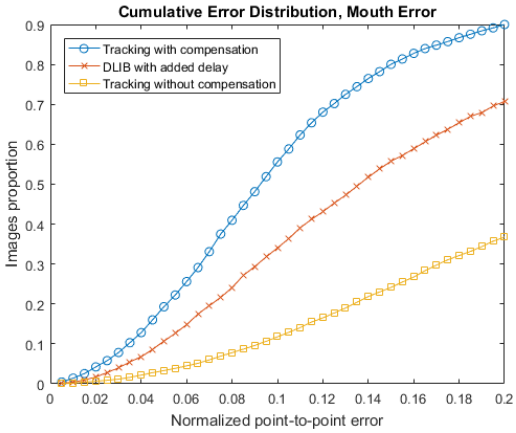
Figure 9. CED curves of the category two videos with a delay of 7 frames



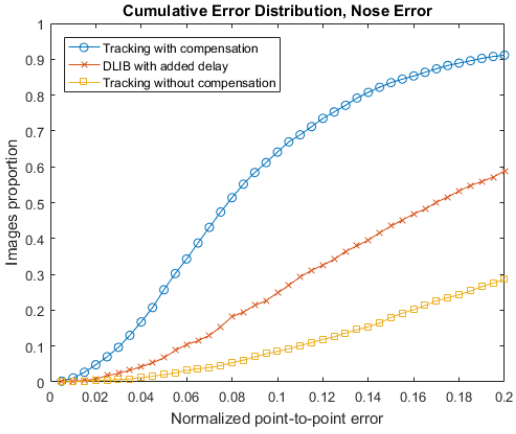
(a) CED curves of the left eye



(b) CED curves of the right eye



(c) CED curves of the mouth



(d) CED curves of the nose

Figure 10. CED curves of the category two videos with a delay of 14 frames

REFERENCES

[1] Y. van Wettum, "Facial landmark tracking on a mobile device," January 2017. [Online]. Available: http://essay.utwente.nl/71696/

[2] J. Shen, S. Zafeiriou, G. G. Chrysos, J. Kossaifi, G. Tzimiropoulos, and M. Pantic, "The first facial landmark tracking in-the-wild challenge: Benchmark and results," in *2015 IEEE International Conference on Computer Vision Workshop (ICCVW)*, Dec 2015, pp. 1003–1011.

[3] G. Bradski, "The opencv library," *Dr. Dobb's Journal of Software Tools*, 2000.

[4] J. yves Bouguet, "Pyramidal implementation of the lucas kanade feature tracker," *Intel Corporation, Microprocessor Research Labs*, 2000.

[5] D. E. King, "Dlib-ml: A machine learning toolkit," *Journal of Machine Learning Research*, vol. 10, pp. 1755–1758, 2009.

[6] dlib c++ library - face_detection_ex.cpp. http://dlib.net/face_landmark_detection_ex.cpp.html. (Accessed on 29-06-2017).