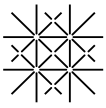


University of Basel

An Empirical Comparison of Deep Learning and 3DMM-based Approaches for Facial Occlusion Segmentation

Bachelor-Thesis

Natural Science Faculty of the University of Basel
Department of Mathematics and Computer Science
Computer Graphics and Vision (Gravis)
http://gravis.dmi.unibas.ch/

Examiner: Prof. Dr. Thomas Vetter
Supervisor: Dr. Adam Kortylewski

Elias Arnold
elias.arnold@stud.unibas.ch
14-930-770

10.08.2018

Acknowledgments

I would like to thank Dr. Adam Kortylewski for his extraordinary support and guidance in this thesis process. He supported me greatly and was always willing to help me. Thanks to Prof. Dr. Thomas Vetter for his expert advice and the opportunity to write this thesis in his research group.

This project would have been impossible without the excellent tutorials of the Gravis Group of the University of Basel. These tutorials have made it much easier for me to get started with Scalismo. The results in this work have been generated on hardware of the Gravis Group.

Abstract

Fittig is the process of finding the optimal parameters for a 3D Morphable Model (3DMM) so that its as close as possible to a given 2D-Object. In our our case, these objects are facial images of which only the face should be reconstructed. The Fitting-Algorithm has somehow to determine whether a pixel of the given image belongs to the face or not. One approach is to label each pixel if it shows a part of the face. The process of finding these labels is called *segmentation* and all labels together make up the *mask*. In facial images, faces are often not completely visible or occluded by a variety of objects. Therefore, an appropriate segmentation should detect occlusions and exclude them from the facial region.

In this thesis, we use the face segmentation-network proposed by Nirkin et al. [2]. It is claimed to be very fast and to make accurate segmentations. In a first step, we evaluated the mask of the network by comparison with other segmentations. In a next step, we integrate the mask of the network into the combined segmentation and parameter adaption process of Egger et al. [7] by treating our segmentation as a mask for the fitting process. We compare these fits with results of other masks by calculating the difference of the parameters.

Table of Contents

1 Introduction

Fitting is the process of generating a 3D model of a face from a 2D image. There are different approaches on how to do that. The gravis group of the University of Basel has developed an MCMC (Markov Chain Monte Carlo) algorithm that makes random changes to a 3DMM (3D Morphable Model) and accepts the proposal face whenever the new 3D face has a greater probability on the image that has to be fitted than the previous one (Apart from a certain amount of randomness which allows steps in less probable regions). For this Thesis, we use the popular Basel Face Model 2017 [1]. To allow the algorithm to determine the likelihood of a 3D facial proposal relative to a given 2D face, the algorithm's evaluator needs to know, which pixels to include in the probability calculation. Therefore we have to label each pixel whether it is part of the face and should be regarded by the evaluator, or if it is background and hasn't any importance for the construction of the 3D Model. Nirkin et al [2] claim, that this is possible with a standard Fully Convolutional Network (FCN).

1.1 Artificial neural networks

The idea of Artificial Neural Networks was heavily influenced by biology. These Networks consist of a variety of neurons which are grouped in layers. The way each neuron works is very simple. It takes multiple inputs of varying strength from other neurons, sums them up and decides depending on the sum whether it should send a stimulus itself and if so, in which strength. Each layer is somehow connected to the next layer. Some layers are fully connected (each neuron of a layer is connected to every other neuron in the next layer) while others are convolutional. Convolutional means that a neuron only gets input of its neighbors in a previous layer. There are many different architectures which mainly differ in the number of layers, number of neurons per layer and the interconnectivity of the neurons. A classical Convolutional Neural Network (CNN) is depicted in Figure 1.1.

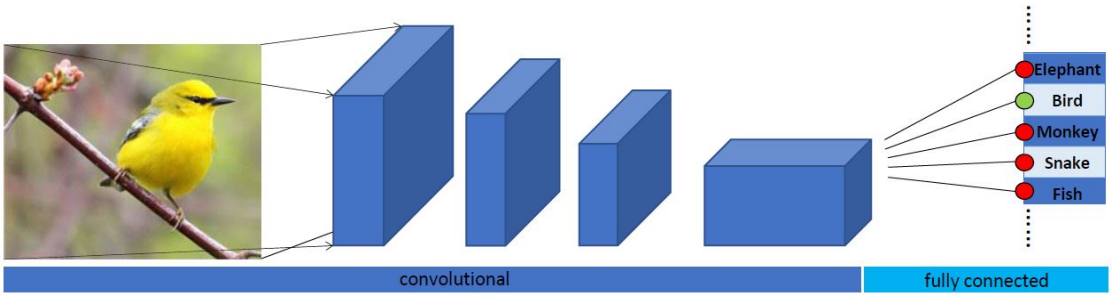


Figure 1.1: An example of an classical Convolutional Neural Network (CNN). After a certain number of convolutions, a pooling layer extracts the most important information of the image and writes it into the next layer. That is why the picture is getting smaller and smaller until just a vector is left. In our case (FCN) the picture is up sampled again to the original size.

Already in 1943 Warren McCulloch and Walter Pitts [3] showed that even simple networks of this kind can simulate every possible logical formula. For this, they used a neuron model that consisted of simple logic gates and could only process only binary input and output signals.

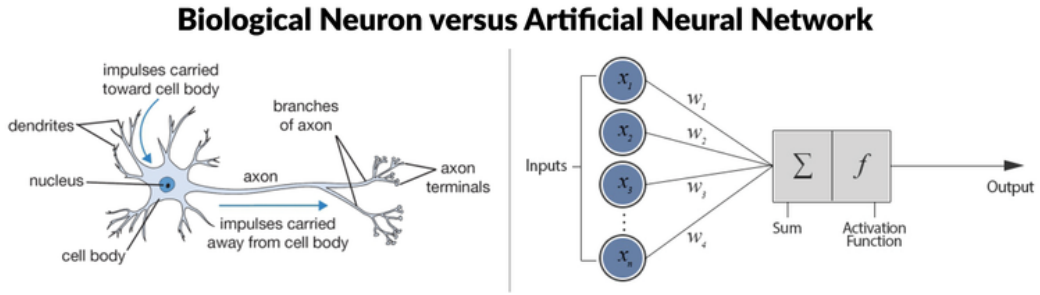


Figure 1.2: The left-hand side of the image [1]shows a biological neuron. It is a nerve cell that occurs in almost every animal. On the right-hand side is an artificial neuron. It sums up the stimuli of the previous neurons, applies an activation function to this sum and forwards the output of this function itself.

1.2 Adjustment of the edgeweights

The algorithm for this is called *backpropagation*. To train a network, the desired output must be known. Depending on the difference between the actual output and the desired output, the new weight of each edge is determined by derivation using the chain rule. The further away an edge is

[1] Source: Google.com

from the output, the less the edges weight gets updated. The chain rule approaches the desired output in very small steps. This bypasses the problem of Overfitting and prevents a neural network from learning the respective input by heart. Usually, many *backpropagation*-Steps are required to train a network. For details see section 1.2.1

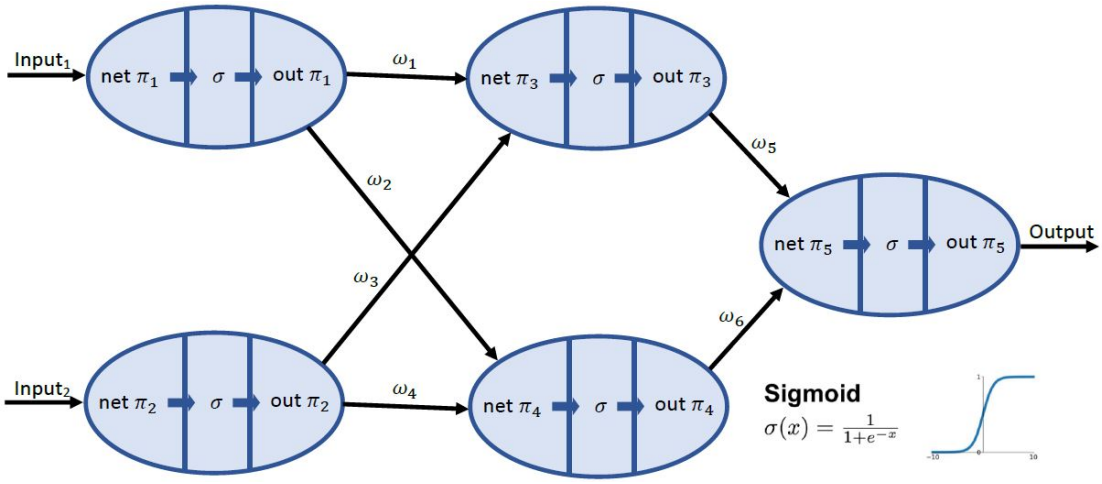


Figure 1.3: A simple neural network consisting of three layers. Neurons π_1 and π_2 are in the input layer. π_3 and π_4 make up the only hidden layer and π_5 is in the output layer. Each neuron sums up the input signals ($net\pi_i$), applies an activation function to it (σ) and forwards the output of this function as its own signal ($out\pi_i$). Mostly a sigmoid function is chosen as the activation function (right lower corner). The edgeweights are annotated with ω_i.

1.2.1 A simple claculation example (on Figure 1.3)

For this example, we assume that the neurons (and the edges connected to it) labeled with π_1 and π_2 are in the input layer and the π_5 neuron is in the output layer. The layer where π_3 and π_4 are in, is called *hidden layer*. For this example, the inputs and the edge-weights (ω_i) are the following:

- $Input_1$: 1
- $Input_2$: 0
- $\omega_i = \frac{i}{10} \forall i \in \{1, ..., 6\}$

First, let's calculate the output ($out\pi_5$) of the network:

$$out\pi_1 = \frac{1}{1+e^{-1}} = 0.731 \qquad out\pi_3 = \frac{1}{1+e^{-0.223}} = 0.556$$
$$out\pi_2 = \frac{1}{1+e^{-0}} = 0.5 \qquad out\pi_4 = \frac{1}{1+e^{-0.346}} = 0.586$$
$$net\pi_3 = 0.1 * 0.731 + 0.3 * 0.5 = 0.223 \qquad net\pi_5 = 0.5 * 0.556 + 0.6 * 0.586 = 0.630$$
$$net\pi_4 = 0.4 * 0.5 + 0.2 * 0.731 = 0.346 \qquad out\pi_5 = \frac{1}{1+e^{-0.630}} = 0.652$$

The output of the network is 0.652, but we want it to be 0.1. Since $\sigma^{-1}(0.1) = -ln(9)$ we expect both ω_5 and ω_6 to get smaller. For the *backpropagation* we need an error-function. We choose the MSE (mean squared error). So $MSE = (0.652 - 0.1)^2 = 0.305$. We start the *backpropagation* calculation with applying the chain rule to the derivative of the error with respect to the edge-weight which has to be updated:

$$\omega_i^* = \omega_i + \Delta\omega_i = \omega_i + \frac{\partial MSE}{\partial \omega_i} = \omega_i + \underbrace{\frac{\partial MSE}{\partial out\pi_5} * \frac{\partial out\pi_5}{\partial net\pi_5}}_{\delta_{\pi_5}} * \frac{\partial net\pi_5}{\partial \omega_i} \text{ for } i \in \{5, 6\}$$

- $\delta_{\pi_5} = \frac{\partial MSE}{\partial out\pi_5} * \frac{\partial out\pi_5}{\partial net\pi_5} = 2 * (0.652 - 0.1) * -1 * 0.652 * (1 - 0.652) = -0.2505$
- $\omega_5^* = \omega_5 + \delta_{\pi_5} * \frac{\partial net\pi_5}{\partial \omega_5} = \omega_5 + \delta_{\pi_5} * out\pi_3 = 0.361$
- $\omega_6^* = \omega_6 + \delta_{\pi_5} * \frac{\partial net\pi_5}{\partial \omega_6} = \omega_6 + \delta_{\pi_5} * out\pi_4 = 0.453$

Now we move on with the $\Delta\omega_i$'s for the *hidden layer*:

$$\frac{\partial E}{\partial \omega_i} = \underbrace{\underbrace{\frac{\partial MSE}{\partial out\pi_5} * \frac{\partial out\pi_5}{\partial net\pi_5}}_{\delta\pi_5} * \underbrace{\frac{\partial net\pi_5}{\partial out\pi_3}}_{\omega_5} * \frac{\partial out\pi_3}{\partial net\pi_3}}_{\delta_{\pi_3}} * \frac{\partial net\pi_3}{\partial \omega_i} \text{ for } i \in \{1, 3\}$$

$$\frac{\partial E}{\partial \omega_i} = \underbrace{\underbrace{\frac{\partial MSE}{\partial out\pi_5} * \frac{\partial out\pi_5}{\partial net\pi_5}}_{\delta\pi_5} * \underbrace{\frac{\partial net\pi_5}{\partial out\pi_4}}_{\omega_6} * \frac{\partial out\pi_4}{\partial net\pi_4}}_{\delta_{\pi_4}} * \frac{\partial net\pi_4}{\partial \omega_i} \text{ for } i \in \{2, 4\}$$

$$\delta_{\pi_3} = \delta_{\pi_5} * \omega_5^* out\pi_3 * (1 - out\pi_3) = -0.022 \qquad \delta_{\pi_4} = \delta_{\pi_5} * \omega_6^* out\pi_4 * (1 - out\pi_4) = -0.028$$
$$\omega_1^* = \omega_1 + \delta_{\pi_3} * out\pi_1 = 0.084 \qquad \omega_2^* = \omega_2 + \delta_{\pi_4} * out\pi_1 = 0.180$$
$$\omega_3^* = \omega_3 + \delta_{\pi_3} * out\pi_2 = 0.289 \qquad \omega_4^* = \omega_4 + \delta_{\pi_4} * out\pi_2 = 0.386$$

If we now evaluate the network, we get a $out\pi_5$ of 0.613. This is a very small change in absolute values because we have made only one iteration of the *backpropagation* algorithm. However, the MSE improves from 0.305 to 0.263!

1.3 The network used

For this thesis, we used a pretrained fully convolutional network from [2]. A Fully Convolutional Network (often called: FCN) is basically a CNN but with a modified architecture. An FCN does not have the fully connected layers which are usually found at the end of an CNN (see Figure 1.1). These layers would enable the Network to make decisions based on global information. A CNN for example can be used for classification. But for image analysis we want local information of the input image (we don't want to know if there is a face in the image, but where the face is in the image). Therefore a FCN uses only convolutional and pooling layers. In the whole Fully Convolutional Network only the following structure is repeated: One or more convolution layers and a pooling layer which downsamples the picture. This constellation is repeated several times.

The assembly of the network used for this thesis follows the FCN-8s-VGG architecture with extensions of [4]. The first part 'FCN' stands for 'Fully Convolutional Network', '8s' means that the result gets eight times upsampled (because of the pooling layers) and 'VGG' means the popular 16-layer network used by Oxford's Visual Geometry Group [5]. The original task of the network was to find the name of an object in an input image. The network could distinguish between 1000 different objects. Each cell in the final vector (1*1000 in size) was a boolean variable for one specific object.

1.4 Related Work

For our experiments a pretrained Fully Convolutional Network (FCN) of [2] was used. They showed that even with a widespread network, you can do good segmentation and that the network does not have to be specially tailored for the future purpose. But the network must have been trained with a large enough data set. For more details about the used FCN and the architecture see section 1.3. They used the FCN for intra- and inter-subject face swapping on the Labeled Faces in the Wild (LFW) dataset and showed that intra-subject swapped faces remain as recognizable as before the swap and that in the inter-subject version better face swapping leads to less perceptibility.

They used a semi-supervised approach to produce training data in order to train the FCN. To produce large quantities of them, they used 2'043 face videos of the IARPA Janus CS2 dataset. To avoid searching for the face in every frame of the video they used motion queues which tracked the face given an initial segmentation based on [6] which enriched their training set to 9'818 images. To enlarge the collection of images, they rendered 3D Shapes of of various objects (e.g. sunglasses, hands) into existing images. Each occlusion adds 9'500 images to their training set.

Egger et al. of the gravis group of the University of Basel propose a fully automated, probabilis-

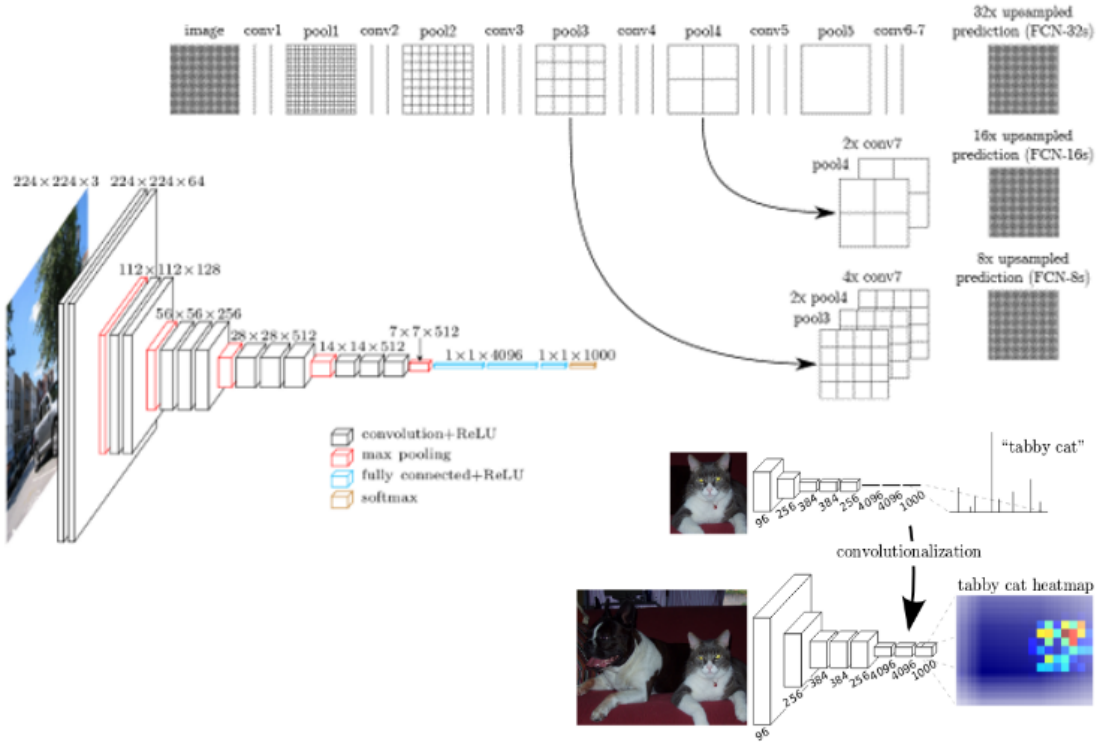


Figure 1.4: In the lower-left corner, the 16 layers of the well known VGGnet are shown. In the top-right, you can see the meaning of the '8s' term of the FCN-Name. It means that the resulting image has to be 8x upsampled, to get an image which is in size equal to the input image. J. Long et al [4] take the network and transform the fully connected layers into convolution layers (bottom-right).

tic and occlusion-aware 3D morphable face model adaptation framework [7]. These Methods use an iterative approach to generate the z-labels, namely to label each pixel whether it belongs to the face or to the background. Each label is described by its own model. This approach can handle multiple labels and differentiate between multiple occlusion types. For example face specific ones (eg. beards) and background. For updating the z-labels they use an algorithm which classifies a pixel based on the probabilities for each possible label. But for our experiments we limited ourselves to two. We only need to distinguish between face (including skin and beard) and background.

The algorithm does two things at the same time. In addition to creating a face mask (segmentation), it estimates the parameters of the popular Basel face Model to 3D reconstruct the given face. It's an EM-algorithm like method to solve two problems simultaneously. In the E-steps they updated the z-labels and in the M-steps they update the face model parameters. For face model adaptation they apply a stochastic sampling strategy based on the Metropolis–Hastings algorithm (Markov Chain Monte Carlo). The likelihood is split up in a background

model for each pixel (X_{BG}) and a foreground model (X_{FG}).

$$\underbrace{p(\Theta|I) = p(I|\Theta) * p(\Theta)}_{\text{posterior brobability of face model parameters } \Theta \text{ given an image } I} \qquad \text{with: } p(I|\Theta) = \prod_{X \in \text{pixels}} p(X_{BG}|\Theta)^{z-1} * p(X_{FG}|\Theta)^{z}$$

Conventional approaches for segmentation often fail on important parts of the face such as the eyes, eyebrows or the oral region because of their strong variability in color and shape. The segmentation of Egger et al. has difficulties with these aspects too as Figure 1.5 shows. The algorithm starts with an initial guess and then alternating updates the Parameters Θ and the z-labels. From the updated Parameter Set (M-Step) the algorithm updates the z-labels (E-Step) and vice versa (see Figure 1.6).

Figure 1.5: This picture shows the development of the labels after 0, 10 and 20 iterations. Striking in this sample image are not only the eyes as mentioned before but also the shadow of the nose, which is first segmented as a background. Only after a certain number of iterations these errors are partially recognized and provided with the correct label.

An approach using convolutional neural networks to segment occluded faces has already been described by [9]. The big difference to our approach is that multiple frames are needed for the final segmentation. An other approach of [10] of the University of Basel is to use random forests to detect facial-occlusions by hair to integrate occlusion into the fitting process in order to model uncertainty. Unlike us, they integrate the occlusion in the face likelihood of the evaluator.

1.5 Expectations of the FCN

Because [2] trained the FCN on a very large and diverse dataset, we expect the network to succeed in the task of segmenting a variety of faces. They claim that a standard segmentation network is sufficient to segment as well as a network which was especially tailored for this task and outperforms state of the art methods for face segmentation. The testing of the network is

[2] Figure 1.6 and it's description are one to one copyed from Fig.4 of the "Occlusion-aware 3D Morphable Models and and Illumination Prior for Face Image Analysis" paper of Egger et al. [7]

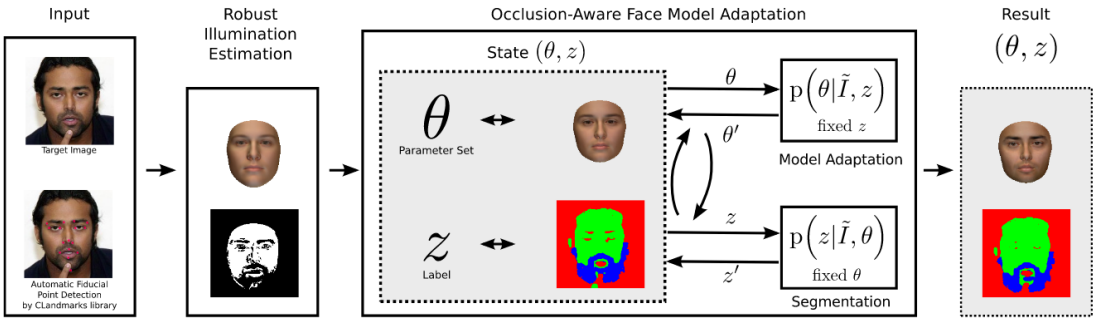


Figure 1.6: [2]Algorithm overview: As input, we need a target image and fiducial points. We use the external Clandmark Library for automated fiducial point detection from still images (Uřičář et al (2015) [8]). We start with an initial face model fit of our average face with a pose estimation. Then we perform a RANSAC-like robust illumination estimation for initialization of the segmentation label z and the illumination setting (for more details see Figure 9). Then our face model and the segmentation are simultaneously adapted to the target image I. The result is a set of face model parameters Θ and a segmentation into face and non-face regions. The presented target image is from the LFW face database (Huang et al. (2007))

done on real-life images, as well as on synthetically generated facial images which are occluded by an artificial object. We expect the FCN to do well on usual facial-occlusions (eg. hands, microphones) because it was trained on them and especially on the difficulties which are shown in Figure 1.5.

As already mentioned before, the approach of Egger et al. is iterative. Due to the high resolution of the Basel Face Model and the use of a software renderer they need up to 25 minutes per image. The performance of the code was less important to them. According to Egger et al. the runtime for the segmentation is about 2 minutes. Nirkin et al. claim that the FCN they trained surpasses previous results in both accuracy and speed. Our measurements showed an average speed of 3.2 seconds per image, including the initialization of the FCN (on an Intel Xeon E7 v3/Xeon E5 v3/Core i7 DMI2 (rev 02)). So we expect the segmentations of the FCN to be qualitatively nearly as good as the segmentations of Egger et al. but to be much faster.

2
Evaluation

2.1 Evaluation on Datasets

The face segmentation network was tested on two different datasets with real-life images: 1.) The Caltech occluded faces in the wild (COFW) by [11] and 2.) parts-labeled LFW Dataset of the University of Massachusetts [12]. Both datasets are designed to present faces in real-world conditions. The COFW dataset provides 29 landmarks and a bounding box for all 507 images. The original LFW dataset contains 13'000 images of 1'680 different subjects. Each face is labeled with the name of the subject. This database is actually meant to test facial recognition/verification algorithms on it. Nevertheless, we tested the segmentation of the FCN on the 500 image parts-LFW validation set. For each image there is a ground truth segmentation, which makes it possible to measure the quality of the FCN's segmentation in numbers.

2.1.1 COFW

Since on the COFW dataset, only landmarks and bounding-boxes are given, the segmentation had to be evaluated qualitatively. We were't able to count the correctly labeled pixels of the FCN with respect to a ground-truth mask. To increase the precision of the network, we cropped the images according to the given bounding-boxes. Because the bounding-boxes were only including the eyes and the mouth of the subject, we had to add an offset which was determined by eye. Nevertheless, we tried to reconstruct the graphic on Nirkis [2] github-repository 'face_segmentation' (Figure 2.1). In the next Figure (Figure 2.2) the same 18 images are segmented by the iterative method of Egger et al. This algorithm outputs both, a model and a segmentation. In Figure 2.3 are the fits of 9 images with both masks (segmentations) used. A matrix of the other 9 fits can be found in the Appendix.

Figure 2.1: 18 images of the COFW Dataset overlayd with the FCN output (in red). The segmentation results are very similar to those on Nirkin's github page.

Figure 2.2: Exactly the same images as in Figure 2.1, but this time with the (final) segmentation of the occlusion-aware method of Egger et al. [7]. Often the eyes are not segmented or the segmentation includes skin other than the face (eg. hands).

2.1.2 Parts-LFW

In the Parts-LFW dataset, we had a ground-truth mask for every image. The mask distinguishes between hair, skin and background. We looped trough all the provided ground-truth masks and overlaid them with the segmentations of the FCN. Now, we can compare the labels of both masks. The evaluation is quite impressive. The FCN performs very good in segmenting only pixels which belong to the face. In average over the 500 images of the Parts LFW dataset, there are 98.5% right non-segmentations (only 1.5% False Positives). On the other hand, only 85.4%

Figure 2.3: Tuples of facial images. In every tuple, the first image shows the fit with the mask of the algorithm of Egger et al. itself (Figure 2.2). The second shows the fit with the FCN mask which are depicted in Figure 2.1.

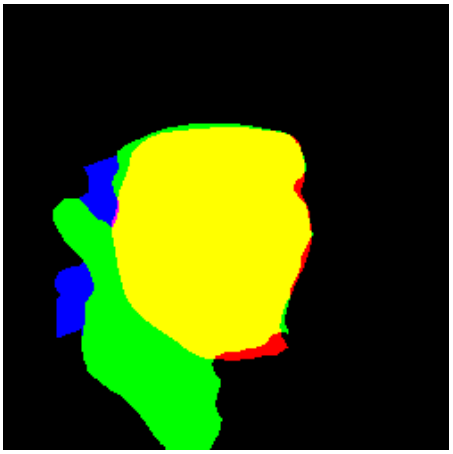
of all the pixels which belong to the face are segmented as face (4.6% False Negatives) which is not a good but acceptable result. Figures 2.4(a) and 2.4(b) depict such a face image and it's mask. Unfortunately, on the given mask, no distinction is made between face and other parts of the body, but everything is segmented as skin (Sub-figures 2.4(a) and 2.4(b)). However, more problematic is that some faces have beards. The FCN of [2] segmented facial hair which was excluded on the provided labels. So we had to manually remove these images (Sub-figures 2.4(c) and 2.4(d)). To reduce the effort, we took the Parts-LFW validation set containing 500 images. After removing the ones with a beard or mustache, we were left with 447 images. The results are summarized in Figure 2.5.

2.2 Evaluation on Synthetic-Data

In order to evaluate the FCN on synthetic-data, we used the Parametric-Face-Image-Generator of [13] to produce images of a random face in a given pose (see Figure 2.6). We extended the software so that it now renders occlusions over the face. Further, we changed the Parametric-Face-Image-Generator so it now generates a Ground-Truth-Mask, which classifies every pixel either as part of the face or as non-face.

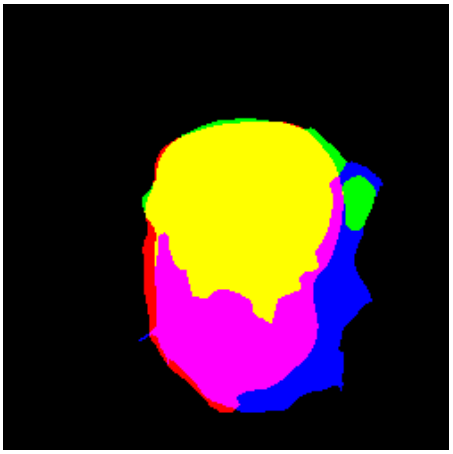
Nirkin et al. [2] claim that both the face itself an the context of the face play an important role for the outcome of the segmentation. To measure this effect and reduce the impact of outliers, we repeat each experiment multiple times with a different face and a different background image. All images we use for the following tests were generated with the same illumination. Each experiment with the synthetic face images is based on the average outcome of 10 datasets with

(a) An facial-image with the segmentation of the FCN highlited in red



(b) The provided ground-truth segmentation of the image 2.4(a) overlaid with the mask of the FCN (red)

(c) An facial-image with the segmentation of the FCN highlited in red



(d) The provided ground-truth segmentation of the image 2.4(c) overlaid with the mask of the FCN (red)

Figure 2.4: Plots of four Turing machines

false-positives (hair)	7.04%
false-positives (background)	0.68%
right-segmentations	85.87%
right non-sementations	99.32%

Figure 2.5: The averages over the 447 images of the Parts-LFW evaluation set of [12]. The FCN recognizes (almost) only pixels, which belong to the face (little false positives). By reducing from 500 to 447 images, we were able to reduce them. Unfortunately, there are many false negatives (skin pixels labeled as background).

different subjects.

Figure 2.6: Five examples of the synthetic face images. The same face is shown with yaw angles -45°, -25°, 0°, 25°, and 45°. The occlusion is randomly choosen and in a random orientation and position.

2.2.1 Dependence of the Euler angles

Because we can now create synthetic face images in any desired pose, we first wanted to find out the accuracy of segmenting the FCN for the angles: Yaw, Roll, and Pitch. In order to do that, we produced with the tool of [13] 101 face images for every integer angle from $-50^{\circ}$to 50°. In every picture, the face is turned one degree further. We evaluated each angle itself and every possible combination of the angles in order to create a hierarchy under the angles (results in Figure 2.7).

From the graphs of Figure 2.7 we can conclude that the roll angle is the most relevant for the FCN, that the pitch-angle is less important and that the accuracy of the FCN is still good even with high yaw angles (least important).

2.2.2 random boxes as occlusions

With the parametric-face-image-generator of [13], we produced about 5.2 thousand images splitted up in three datasets. Every dataset contains images for 20 occlusion levels, where one angle is in the range from $-40^{\circ}$to $40^{\circ}$and the other two angles are set to 0. To reduce the impact of one individual face, each dataset is reproduced 5 times with a different face. The results are in Figure 2.9 and Figure 2.10

In the given table of Figure 2.8, all provided images show a face, from which 20% of the pixels are occluded by a randomly colored box. We can optically verify, that 1) the yaw-angle, despite the occluding box, has not much of an effect. 2.) In both situations, $-40^{\circ}$and 40°, of the roll angle, the result is not satisfying. That's interesting, because no matter how big the roll-angle is, the information (the face) stays the same. 3.) In the third column, the segmentation with a negative pitch angle is much worse than with a positive one! This supports our assumption that the roll-angle plays a big role, followed by the (asymmetric) pitch-angle. The yaw-angle is less

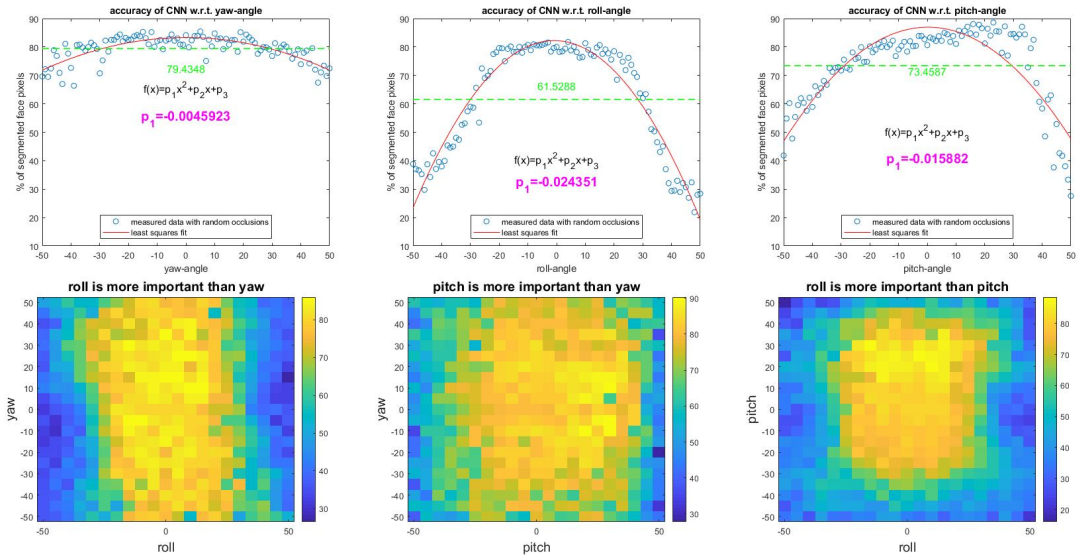


Figure 2.7: In the plots on the top row we see the segmetation accuracy in percent (on the y-axis) for every single image (with face angle from $-50°$to $50°$on the x-axis). The point cloud is approximated by a quadratic function via a least squares fit (red curve / $f(x)$). The first parameter of this function determines the opening angle (p_1). In the bottom row the colors indicate the segmentation accuracy. The brighter the color, the better the segmentation. In every plot, there is a cluster of high accuracy segmentations centered in the origin. The angle on whose axis the cluster has the smaller extent is the more important of the two. We call an angle 'important', when a small change of this angle leads to a failure of the FCN.

important because even at large angles much of the face is still segmented.

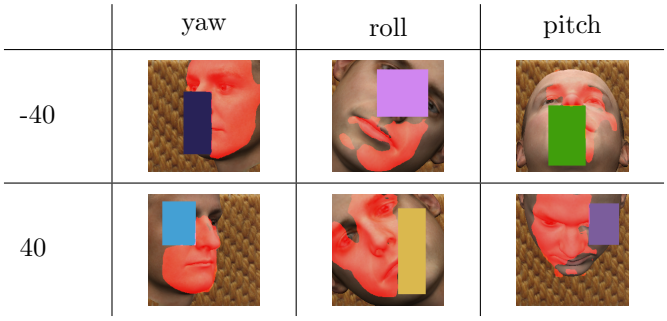


Figure 2.8: Based on these images, we can see that the roll angle (rotation) is the most sensitive, followed by the pitch angle (tilt), where the segmentation works better on positive angles than on negative ones. The most stable detection is at the yaw angle. It has the least influence on the segmentation.

Although boxes as occlusions are very simple and we're in control of the rectangle's size, we can occlude a given amount of the face region with this method, but in practice, exact rectangles

are very rare. That's why we left it with the rectangles and used real-world objects (e.g. hands, microphones or sunglasses) as synthetic occlusions. Unlike Nirkin[1], we didn't use the landmarks of the face for this task. We placed them randomly on the image instead. Thats the reason why we use a second plot to show the percentage of occluded face pixels in the following.

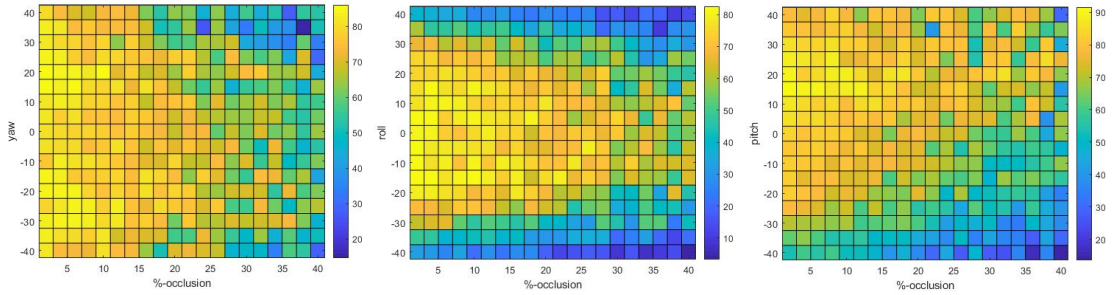


Figure 2.9: The color of each grid cell indicates the accuracy of the segmentation of the FCN on a set of faces turned by the corresponding angle and occluded with a square, so that the corresponding amount (on the x-axis) of the face is hidden. The brighter the color, the better the segmentation. On each plot, we would expect to see a triangle pointing to the right. This means that the combination of a large angle and a large occlusion make the face even more unsegmentable.

We see that the segmentation is very sensitive to roll-angles and the FCN is not trained to segment faces in every rotation! Surprisingly, the yaw angle plays a subordinate role here. The right most plot of Figure 2.10 tells us, that the sign of the pitch angle plays a significant role! Since we aren't able to determine at which angles exactly the FCN begins to fail,we repeated the experiment with a higher angle range:

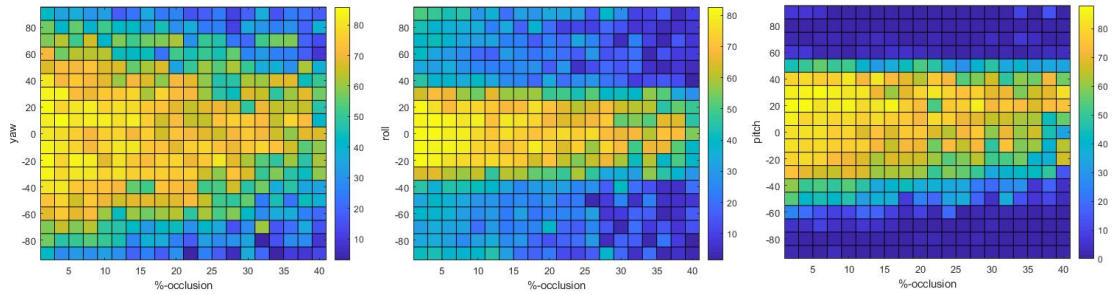


Figure 2.10: This plot shows the outcome of a similar expreiment as snown in Figure 2.9 with less resolution. All the angles (yaw, pitch and roll) range from $-90°$to 90 °. The scale on the "%-occlusion" stays the same as in Figure 2.9 We can clearly see the limits of the FCN even with a occlusion of 2%! Very interesting is the hard transition from good segmentations to bad segmentations in the two right plots.

2.3 Parametric-Face-Image-Generator

As metioned at the beginning of this chapter, we extended the Parametric-Face-Image-Generator of [13]. In our version the option "occlusionMode" in the configuration files can now be set to:

- **"eyes"**: A circle occludes the picture centered on the pupil centers of the face.
- **"random-1"**: A hand hides the picture in a random place in a random orientation.
- **"random-2"**: A microphone hides the picture in a random place in a random orientation.
- **"random"**: A randomly chosen occlusion image hides the picture in a random place.
- **"box"**: Boxes, filled with an arbitrary color hides the image.
- **"box-whiteNoise"**: Boxes, filled with Gaussian white noise hides the image.
- **"box-skinColor"**: Boxes, filled with the color at the tip of the chin hide the image in a random place.
- **"box-[0-100]"**: Boxes, filled with a random color sized that they occlude the specified amount of face pixels.
- **"loop"**: Produces 20 copies of the same image, each wit a box occluding 2,4,6,...,40 % of the face.
- **"texture"**: Fulls a randomly placed box with a texture image specified in the folder 'testures'.

The software provides csv-Files, rps-Files, tlms-Files, ground truth masks, images with occlusions and images without occlusions for both the 'bfm' version and for the tailored 'face12' version of the Basel Face Model. If an occlusion gets rendered over a landmark, it gets disabled, which was of use in the next chapter (see Figure 3.1).

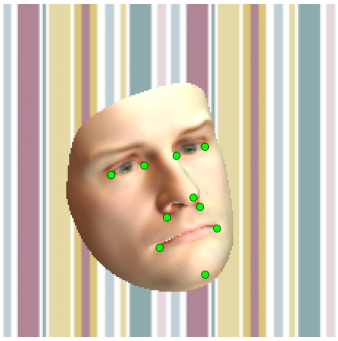
3

Evaluation of the Fitting with the Segmentation of the FCN

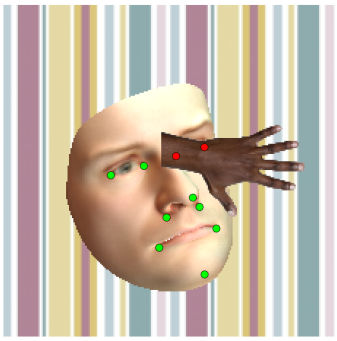
The EM-algorithm like method of [7] uses a binary mask to determine whether a pixel is relevant for the fitting-process or if it's not. We changed the EM-like method of Egger et al. to not estimate a mask at all, but use the same initial mask in each iteration. But we let this EM-algorithm estimate the parameters for a 3DMM (3D Morphable Model) in order to get a 3D reconstruction of the 2D face. As the Morphable Model we use the popular Basel Face Model of 2017 [1] which was originally proposed in 1999 by Blanz and Vetter [14]. The Parametric-Face-Image-Generator of [13] outputs renderings of sample facial images, provides the ground truth parameters of the face and provides a ground truth segmentation. Since we want to measure the fitting accuracy with the masks of Egger itself and the FCN-Segmentation only, we render occlusions over the facial image as in the previous chapter.

The experiments in this cchapter will only be explained with one specific type of occlusions (hands) so as not to overwhelm the reader. Of course, we repeated the experiments on all occlusions that the FCN from Nirkin et al. was trained on. Please find these fits and the error-plots in the appendix.

The fitting method of Egger et al. is guided by landmarks on the face (eg. nose-tip, mouth-corner). Each landmark has a boolean flag which determines whether the landmark is visible for the observer or not. Because the Parametric-Face-Image-Generator was originally developed to render only a face in a certain pose and without any occlusions, all landmarks were labeled as visible per se. Therefore, we had to turn off those who were covered by an occlusion (see Figures 3.1(a) and 3.1(b)).



(a) A facial image of the Parametric-Face-Image-Generator with all landmarks activated.



(b) The same image as in 3.1(a) but with an occlusion and two underlying landmarks which are disabled because they're not visible anymore.

Figure 3.1: The original landmarks (green dots) and the landmark which had to be disabled because of the occlusion (red dots)

For the evaluation of the fitting, we generated faces with 150 parameters. 50 of them were for the shape of the Basel Face Model, another 50 for the color and the last 50 for the expression. We now render different faces with various occlusions, various angles/poses, a random illumination and cover them with different occlusions.

The aim of this chapter is to reconstruct the generated synthetic faces with the EM-like algorithm of Egger et al. which estimates fitting parameters and a segmentation itself. But we manipulate the algorithm that it skips the E-step and takes the segmentation of the FCN, the ground truth segmentation or no mask in every iteration. To get acceptable results we fix the number of EM steps to 20. In this experiment, we measure the fitting accuracy and speed, without letting the fitting algorithm produce it's own z-labels. The FCN is trained to cut hands, glasses and microphones out of the picture. That's why in the following experiments we use these three classes of occlusions and test the segmentation of the FCN an a dataset without any occlusions too. With the mask of the FCN, the fitting should also be much faster, because the time-consuming job of updating the z labels is eliminated. After every iteration, we get information about pose, probability of the current fit and about the 3DMM parameters of the fit. We compare those to the ground truth information from the Parametric-Face-Image-Generator. To calculate the error, we use the RMSE (Root Mean Squared Error).

3.1 Evaluation on a tailored face

In this setting, we use the 'face12' version of the Basel Face Model. It has in contrast to the original no neck, no ears and no hairline. Figure 3.4 shows the error of the parameters, the angles, illumination and the probability in every iteration. The experiments are repeated 10 times with different faces. We see that in all the parameters (shape, color and expression) the quality of the fit with the FCN segmentation is very close to the one of the iterative approach of Egger et al. The differences in the Euler angles (yaw, pitch and roll) are negligible, since all errors move in a very small order. The plot named 'EnvironmentMap' shows the deviation of the illumination parameters.

For updating the 3DMM-Parameters in every iteration (M-Step), the algorithm of Egger et al. does a Metropolis-Hastings like Random-Walk in the parameter space. The walk is guided by the probability of the proposal and tries to maximize it. Unfortunately, you can not compare the curves of the last plot (posterior) directly with each other, as the value of the posterior depends very much on the segmentation used. The more pixels are segmented as face, the more terms the evaluator adds to the posterior. Therefore it can vary very strongly. It is added up because the negative logarithm is applied to the probability value for each pixel. $P(\Theta) = \prod_{x_i \in \text{facepixels}} p(x_i; \Theta) \rightarrow L(\Theta) = \sum_{x_i \in \text{facepixels}} -ln(p(x_i; \Theta))$. This is valid because the logarithm is a monotonic increasing function.

(a) original facial image (b) facial image with occlusion

Figure 3.2: These pictures show the faces that should be modeled with different masks. We fit with the method Egger et al. [7] from Figure 3.2(b) a face as similar as possible to the face in Figure 3.2(a).

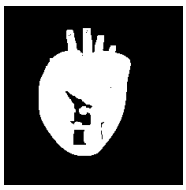
	ground truth mask	egger	FCN	no mask
z-labels				
fits				

Figure 3.3: In the top row are the different segmentations we use for our experiments. The segmentation of Egger et al. (second image from left) was calculated iteratively. Given is only the final mask. In the second row are the particular fits, when the above mask is used as segmentation. You can see that the fits without any mask (NO MASK) and the fit with the mask according to Egger et al. (EGGER) are far away from the true face depicted in Figure 3.2(a). However, the fit with ground truth mask of the Parametric-Face-Image-Generator (GROUDTRUTH) and the one with the FCN-Segmentation (FCN) come very close to the original.

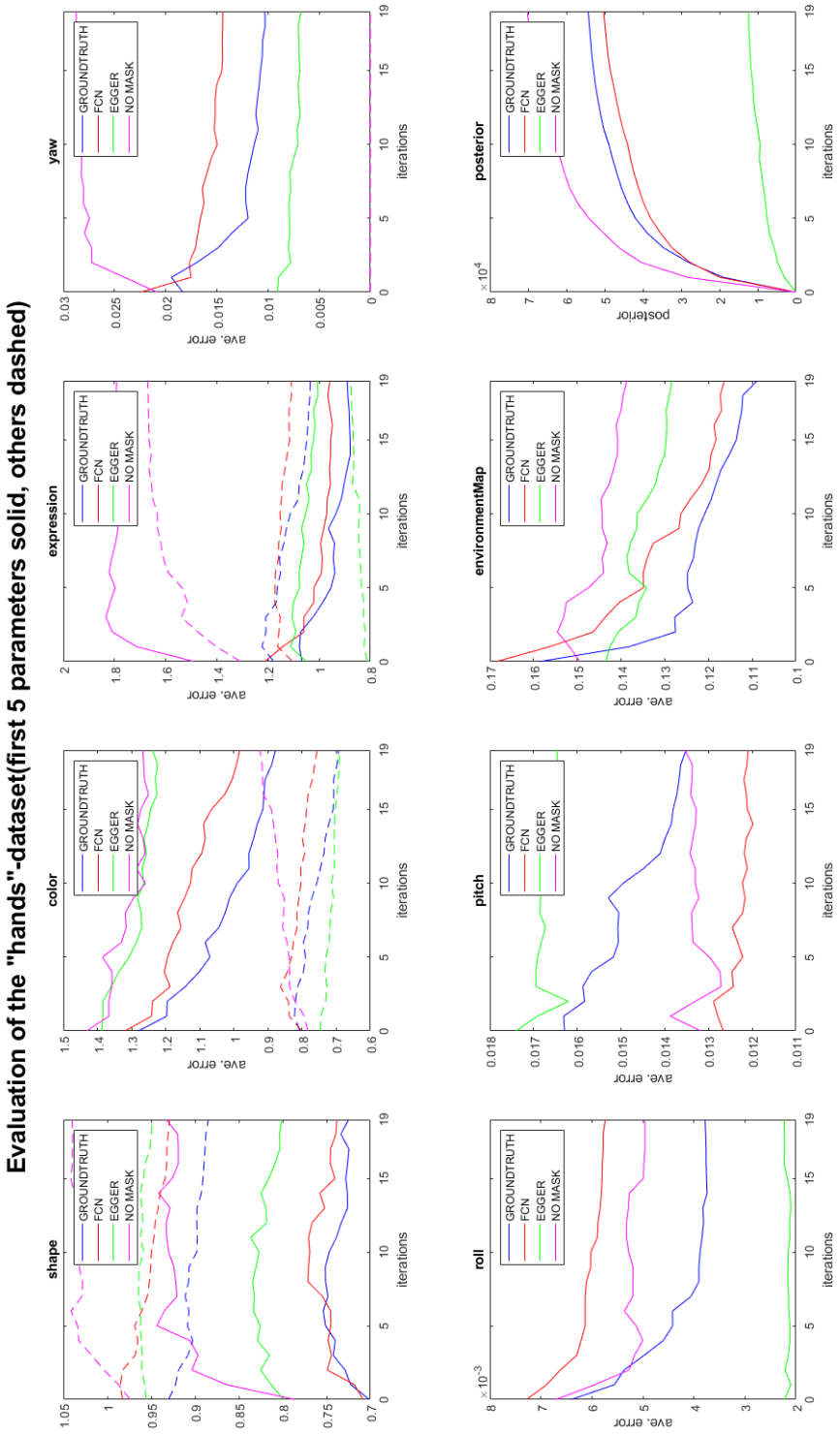


Figure 3.4: The first three plots show the errors of the first 50 shape, color and expression parameters of the Basel Face Model(in standard deviations). The next three show the errors af the Euler Angles in radian. The 'EnvironmentMap'-plot shows the illumination chosen by all methods. In the last plot we see the unnormalized posterior-probability, with which a MCMC model was accepted.

In most cases, the fit with the FCN mask is not as different from the Fit with egger mask itself as in Figure 3.3. However, we find that Egger's method tends to segment all skin pixels, whether they belong to the face or not. That's why in a next step we create faces with the 'bfm' version of the Basel Face Model. These faces are not tailored and include ears and neck. We expect the segmentation of Egger et al. to segment these areas too.

(a) original facial image (b) facial image with occlusion

Figure 3.5: These pictures are different from the ones in Figure 3.2 because we now use the 'bfm' version of the Basel Face Model which contains ears and neck. The fitting algorithm takes 3.5(b) as input and 'tries' to approximate 3.5(a).

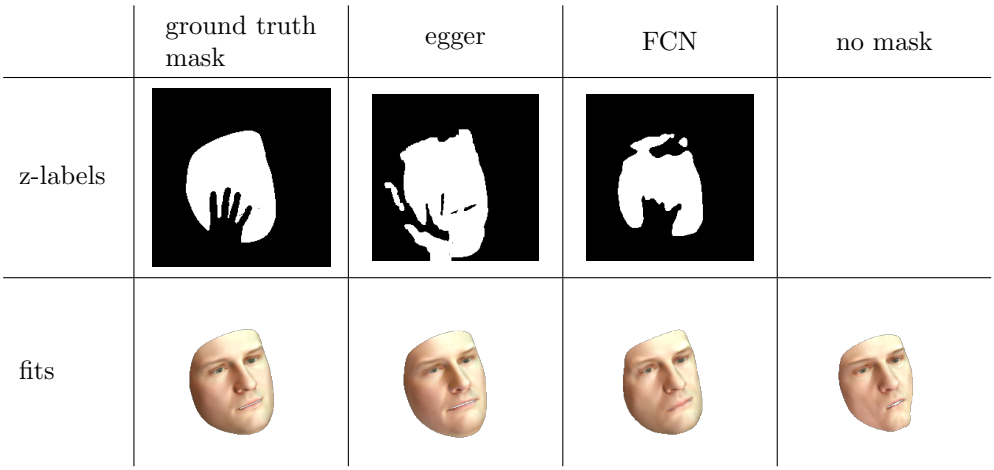


Figure 3.6: Due to the over-segmentation by Egger, a clear difference in the error of the 'color' parameters can be seen. The mask of the FCN models the color significantly better.

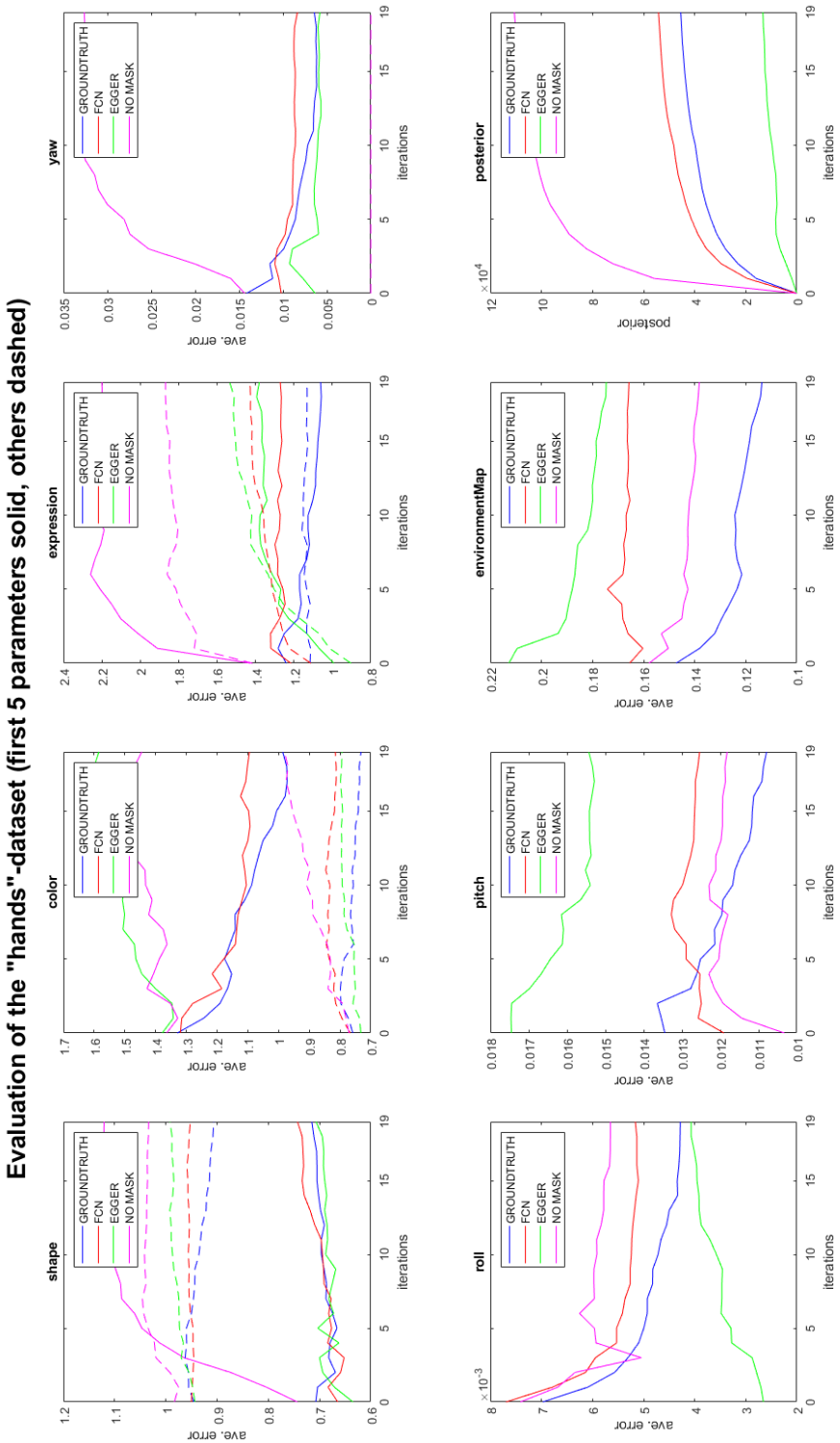


Figure 3.7: Due to the over-segmentation by Egger, a clear difference in the error of the 'color' parameters can be seen. The mask of the FCN models the color significantly better.

From these plots we see that the method of Egger et al. oversegments the synthetic face and segments regions which are not visible on the final fit. Since these regions are also evaluated and have a slightly different color from the rest of the face, the final color is wrong. This is easy to see in the error-plots for the color-parameter (second plot) where the spread between the FCN and the Egger mask is approximately .5 standard deviations.

3.1.1 Runtime

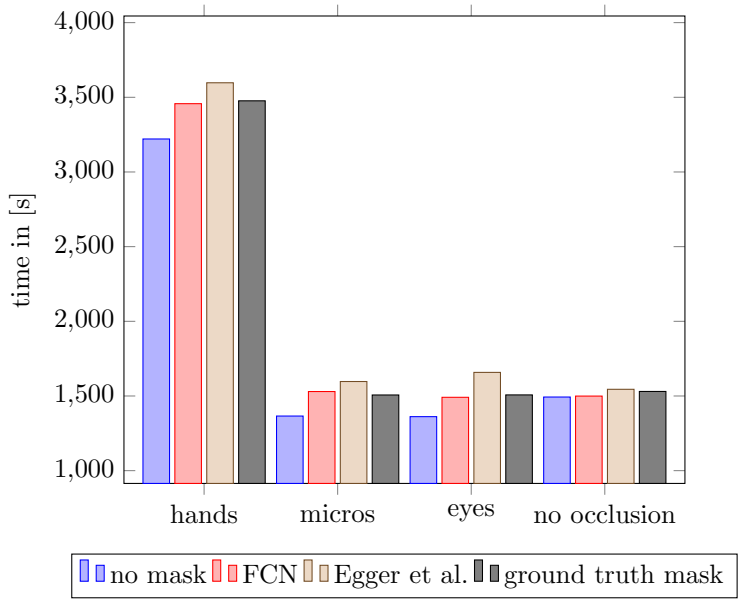


Figure 3.8: Comparison of the Wall-Clock Time for the fitting process. The times were measured with the tailored 'face12' mask and the 'bfm' rendering.

Since we evaluated our experiments on a compute-server where we had no control of the priority of the process, we can't make a general statement about the absolute duration. Within a dataset all fits for the different segmentations were computed simultaneously. From this we can conclude that the fit with the FCN mask tends to require less time especially if the face is occluded by objects which can be recognized by the FCN.

However, it is very difficult to make a general statement about the runtime behavior, because we cannot control whether the algorithm writes results in the cache and just loads them when desired, or computes them. As already mentioned, the Metropolis-Hastings algorithm proposes a randomly chosen face fit to be next one. If the evaluator rejects it, the algorithm uses the old fit as the next and just has to load it from the cache.

4

Integration of the FCN into the original work of Egger et al.

The aim of this chapter is to combine the advantages of the iterative segmentation and fitting method of Egger et al. and of the segmentation by the FCN. In the previous chapters, we saw that both approaches have their strengths. Lets summarize them:

- strenghts of the segmentation of the FCN:
  - Although the borders of the segmentation are sometimes a bit spongy, (almost) only skin pixels are segmented (very few false positives).
  - As long as the face is not rotated (roll angle), the FCN is very sturdy against pitch and yaw.
  - Even if the occlusion has approximately the same color as the face, the FCN may recognize the occlusion (see Figure 3.3)
- strenghts of the segmentation of Egger et al.
  - It is an iterative algorithm, the segmentation is always improved, based on the always improving 3DMM fit.
  - The algorithm can find occlusions that are thin and small and the borders of the segmentation are very sharp (See Figure 4.1.
  - The method of Egger et al. uses a beard prior to exclude beards from the segmentation (see image in the top-right corner of Figure 2.2 and [7])

We combined both approaches so that the fitting algorithm of Egger et al. takes the segmentation of the FCN as the initial mask and refines these during the following iterations in E-Step. The biggest weakness of the FCN is its inability to detect small or thin occlusions (eg in Figure 4.1).

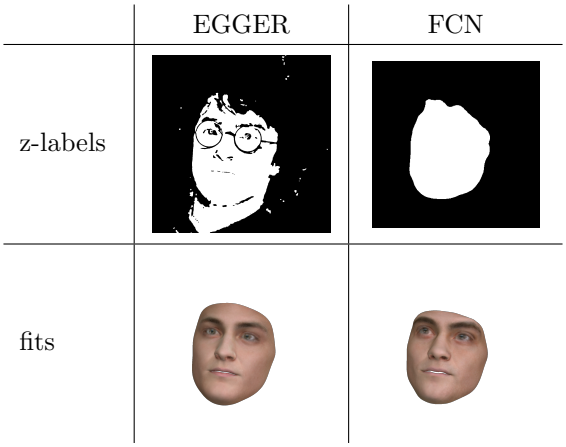


Figure 4.1: This example illustrates a major advantage and a disadvantage of Egger's approach. The FCN can't find the thin glass, but segments the eyes while Egger does not.

It is precisely this weakness of the FCN that is a strength of the approach of Egger et al. It is therefore obvious to take the FCN's mask as a first segmentation, which contains (almost) only facial pixels and then refine this segmentation and search thin occlusion in it.

However, our assumption is, that final segmentation does not differ much from the previous mask according to Egger et al. because of the Metropolis Hastings Sampling in every iteration. The Random Walk always ends in the same place (or same region), regardless of the starting point.

In order not to show too many new pictures, we refer to sample images of the last chapter. We first show an example with an tailored (face12) target face (Figure 4.2 and Figure 4.3), then an example where the target face is a 'bfm'-Version face (Figure 4.5 and Figure 4.6).

Both experiments in this chapter show that the error of a fit with a combined mask is very close to the error of a fit with the iterative segmentation of Egger et al. It is interesting that the model parameters get a little better with the combined segmentation but the lighting gets worse in both cases.

(a) initial mask of the FCN (b) segmentation of Egger et al. (c) combined segmentation (d) fit with the combined mask (Subfigure (c))

Figure 4.2: An example of the combination of the two masks. Figure 4.2(a) shows the segmentation of the FCN while Figure 4.2(b) shows the segmentation by Egger et al. The combination is depicted in Figure 4.2(c). The target image for this example is the same as in Figure 3.2. We see our fears confirmed and that the initial segmentation only makes a tiny difference. Figure 4.2(d) shows the resulting fit with the combined mask.

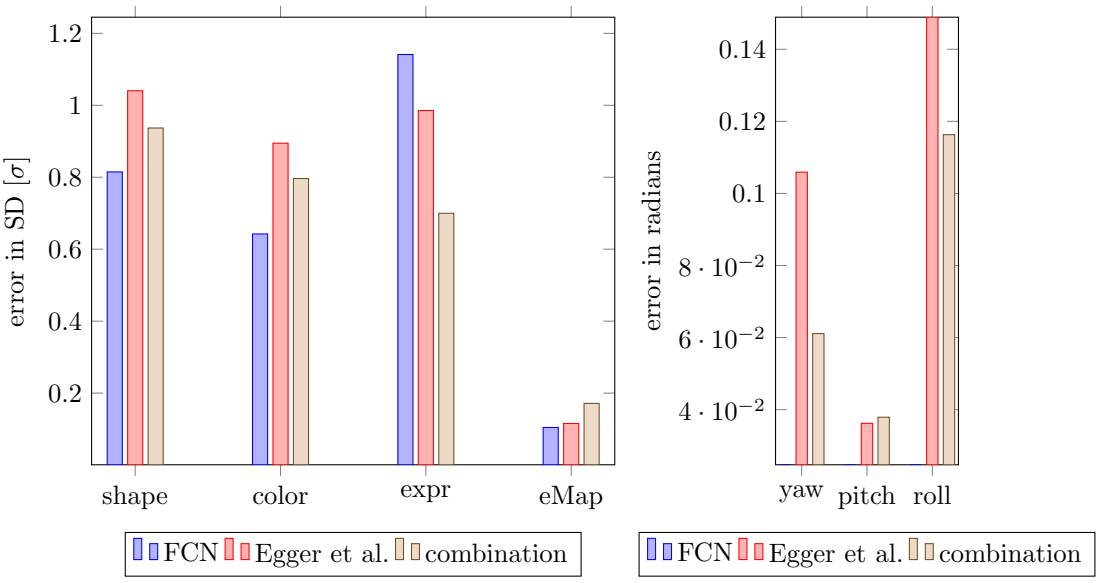
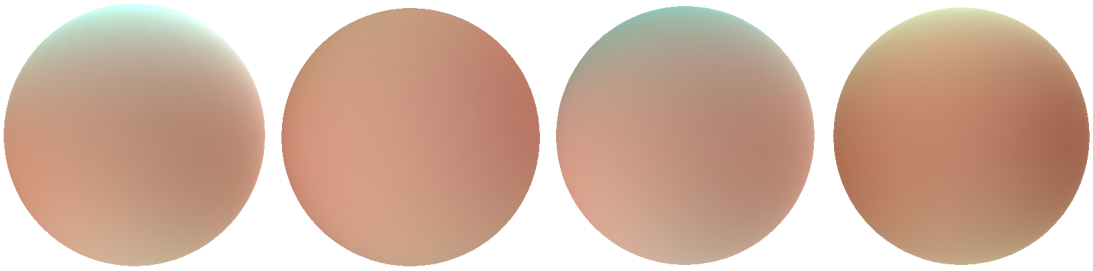


Figure 4.3: A comparison of the Basel Face Model parameters. In this plot the fits of egger and the fcn of Figure 3.3 and 4.2(d) are compared. Per parameter, only the first 5 dimensions are considered. In all considered parameters, the version with the combined mask has a lower error than the slower fit with the mask of Egger et al. alone!



(a) Egger et al. (b) FCN (c) combination (d) groundtruth

Figure 4.4: The illuminations of the last example rendered on a sphere. It seems like the illumination with the FCN mask is dull and has no specular term (only ambient). However, the illumination with the combined mask is strongly based on illumination (a).

(a) initial mask of the FCN (b) segmentation of Egger et al. (c) combined segmentation (d) fit with the combined mask (Subfigure (c))

Figure 4.5: An example of the combination of the two masks. Figure 4.5(a) shows the segmentation of the FCN while Figure 4.5(b) shows the segmentation by Egger et al. The combination is depicted in Figure 4.5(c). The target image for this example is the same as in Figure 3.5(b). Figure 4.5(d) shows the resulting fit with the combined mask.

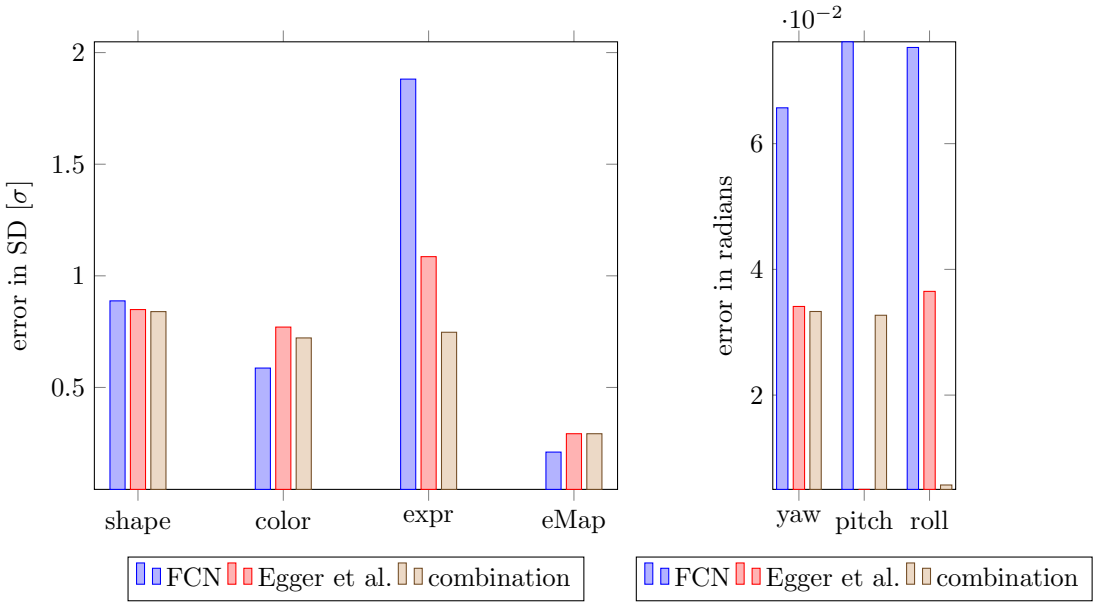
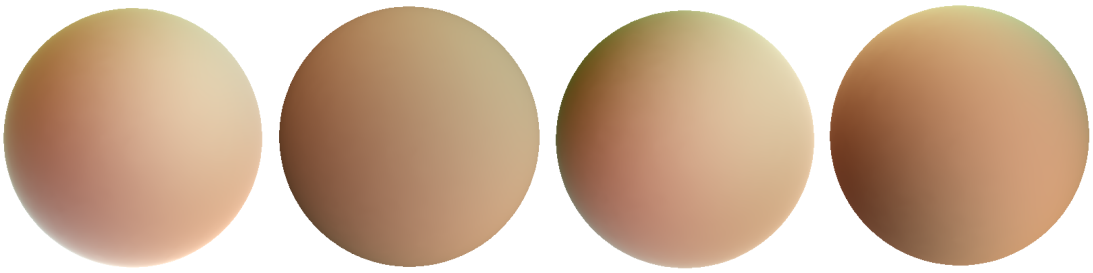


Figure 4.6: A comparison of the Basel Face Model parameters. In this plot the fits of egger and the FCN of Figure 3.6 and the fit with the combined mask 4.2(d) are compared. For each parameter, all dimensions are considered. With the combined segmentation, the fit is a tiny bit better than with the segmentation of Egger et al. alone.



(a) Egger et al. (b) FCN (c) combination (d) groundtruth

Figure 4.7: The illuminations of the last example rendered on a sphere. It seems like (as in the previous example) the illumination with the FCN mask is dull and has no specular term (only ambient). Again, the illumination with the combined mask is strongly based on illumination (a).

5
Conclusion

This thesis tested and analyzed a Fully Convolutional Network (FCN) for segmenting facial images under occlusion. This standard network was trained by Nirkin et al. [2] on a rich and diverse dataset. They claim that the speed and the accuracy of this segmentation method outperforms other approaches that were made especially for this task.

We evaluate the segmentation accuracy of the network on two real-life datasets. In one of them, every image was supplied with the ground truth labels, which determine whether a pixel belongs to the face or not. A comparison of both segmentations showed that the segmentation of the FCN contained very few false positives. However, the FCN only recognizes about 4/5 of the actual facial region (false negatives). In a further step, we evaluated the network on synthetic images with a to be able to be in control of all parameters ourselves. The results show that there is a hierarchy among the angles that determine the position of the face. There is a hierarchy among the angles that determine the position of the face. Regardless of whether the face is hidden or not, the FCN the most vulnerable to large roll-angles, then pitch-angles and least important are the yaw-angles. That's a strange result, because no matter how the face is rolled, the information does not change. We fear that the reason for this is the training of the FCN.

Figure 5.1: Three examples of training images used by Nirkin et al. These are pictures of the recent Janus CS2 dataset. It looks like the face was not turned on any of the faces. The first two images are overlayd with synthetic occlusions. The third one depicts the interface used for semi-supervised labeling.

Egger et al. [7] of the University of Basel propose an EM-like method to simultaneously segment a face out of a given image and reconstruct it. We compare the segmentations of Egger et al. with the segmentation of the FCN and find that (1) The approach of Egger et al. oversegments the image, (2) The method of Egger et al. tends to exclude important details like the eyes due to they strong variability in color and shape, (3) The FCN often fails in recognizing and segmenting thin occlusions.

The runtime of the FCN is much faster compared with the iterative segmentation of Egger et al. On our GPU, the segmentation with the FCN takes approximately 3 seconds, while the algorithm of Egger et al. takes 2 minutes (according to [7]).

(a) original (b) segmentation of Egger et al. (c) segmentation of FCN

Figure 5.2: Comparison of the two segmentation ((b) and (c)) of the same facial image (a).

Both approaches have their weaknesses and strengths, which we try to combine. Therefore we take the iterative algorithm of Egger et al. and give it the FCN-Segmentation as an initial segmentation. Since this algorithm uses a Metropolis-Hastings method, the final mask looks very similar to Egger et al.'s mask without the integration. However, the illumination estimation is improved a lot, because an initial mask exists (the original approach of egger et al. does not use a mask/segmentation to determine the illumination parameters). This is an interesting starting point for further research on how to optimally combine both segmentations.

Bibliography

[1] Gerig, T., Forster, A., Blumer, C., Egger, B., Lüthi, M., Schönborn, S., and Vetter, T. Morphable Face Models - An Open Framework. *CoRR*, abs/1709.08398 (2017). URL http://arxiv.org/abs/1709.08398.

[2] Nirkin, Y., Masi, I., Tran, A. T., Hassner, T., Medioni, and Medioni, G. On Face Segmentation, Face Swapping, and Face Perception. In *IEEE Conference on Automatic Face and Gesture Recognition* (2018).

[3] McCulloch, W. and Pitts, W. A logical calculus of the ideas immanent in nervous activity. *Bulletin of Mathematical Biology*, 5(4):115–133 (1943).

[4] J. Long, E. S. and Darrell, T. Fully convolutional networks for semantic segmentation. In *Proc. Conf. Comput. Vision Pattern Recognition*, page 3431–3440 (2018).

[5] K. Simonyan, A. Z. Very Deep Convolutional Networks for Large-Scale Image Recognition. In *arXiv:1409.1556*.

[6] M. Grundmann, M. H., V. Kwatra and Essa, I. Efficient hierarchical graph-based video segmentation. In *Proc. Conf. Comput. Vision Pattern Recognition* (2010).

[7] Bernhard Egger, A. S. A. K. A. M.-F. C. B. T. V., Sandro Schönborn. Occlusion-aware 3D Morphable Models and an Illumination Prior for Face Image Analysis (2018).

[8] Uricar, M., Franc, V., Thomas, D., Sugimoto, A., and Hlavac, V. Real-time multi-view facial landmark detector learned by the structured output SVM. In *2015 11th IEEE International Conference and Workshops on Automatic Face and Gesture Recognition (FG)*, volume 02, pages 1–8 (2015).

[9] Saito, S., Li, T., and Li, H. Real-Time Facial Segmentation and Performance Capture from RGB Input. *CoRR*, abs/1604.02647 (2016). URL http://arxiv.org/abs/1604.02647.

[10] Morel-Forster, A. *Generative shape and image analysis by combining Gaussian processes and MCMC sampling*. Ph.D. thesis, University of Basel, Faculty of Science (2016).

[11] X. P. Burgos-Artizzu, P. P. and Dollár, P. Robust face landmark estimation under occlusion. CCV 2013, Sydney, Australia (2013).

[12] Kae, A., Sohn, K., Lee, H., and Learned-Miller, E. Augmenting CRFs with Boltzmann Machine Shape Priors for Image Labeling (2013).

[13] B. Egger, A. M.-F. A. S., A. Kortylewski. parametric-face-image-generator (2017).

[14] Blanz, V. and Vetter, T. A Morphable Model for the Synthesis of 3D Faces. In *Proceedings of the 26th Annual Conference on Computer Graphics and Interactive Techniques*, SIGGRAPH '99, pages 187–194. ACM Press/Addison-Wesley Publishing Co., New York, NY, USA (1999). URL http://dx.doi.org/10.1145/311535.311556.

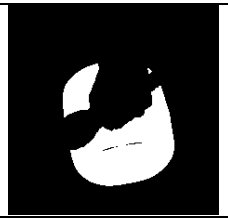
A
Appendix

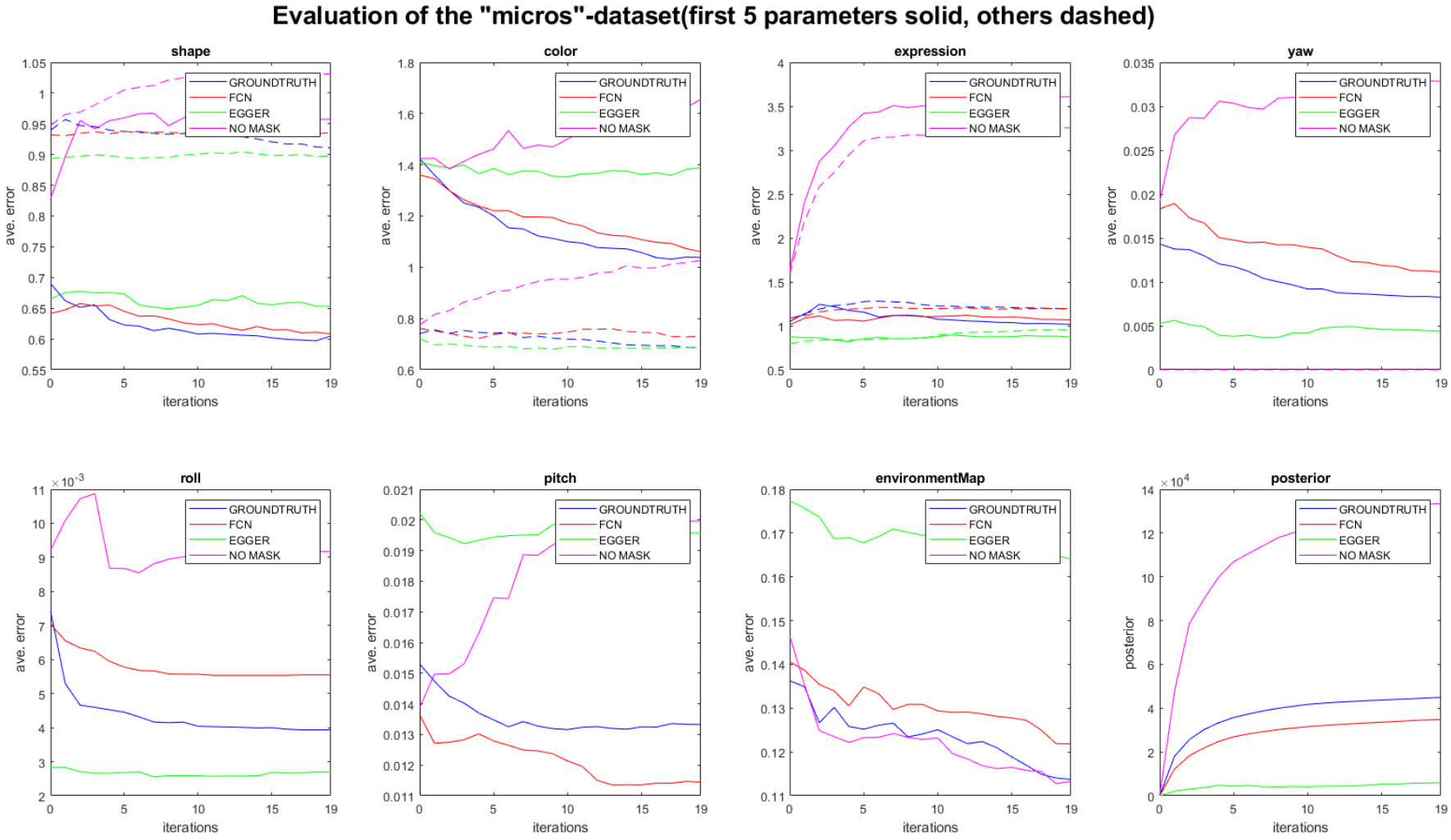
A.1 COFW-Images and Fits

Figure A.1: The additional images of Figure 2.3. In each tuple the first plot shows the fit with the mask of egger et al. and the second plot is made with the segmentation of the FCN.

A.2 Datasets other than Hands(which are shown in the thesis)

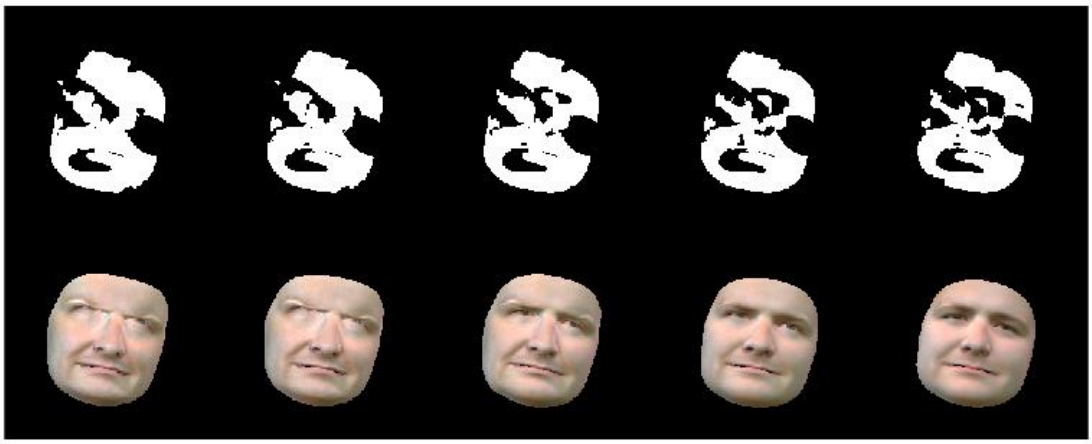
micros (mask: face12, rendering: face12):

	Groundtruth	Egger	FCN	No mask
Mask				
Fit				



segmentation and mask of test3 in every 5th iteration with mask: GROTRU(from right to left)

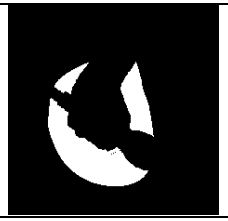
segmentation and mask of test3 in every 5th iteration with mask: EGGER(from right to left)



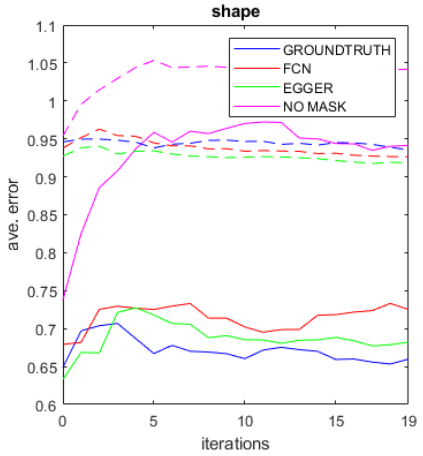
segmentation and mask of test3 in every 5th iteration with mask: FCN(from right to left)

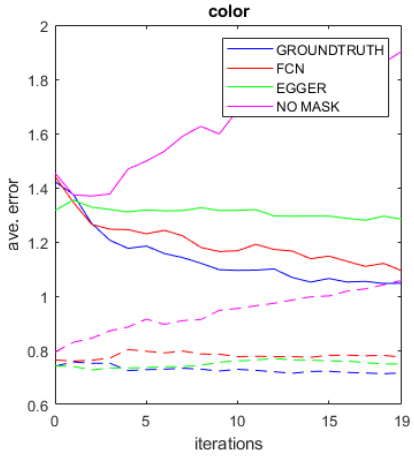
segmentation and mask of test3 in every 5th iteration with mask: NO_OCCLUSION(from right to left)

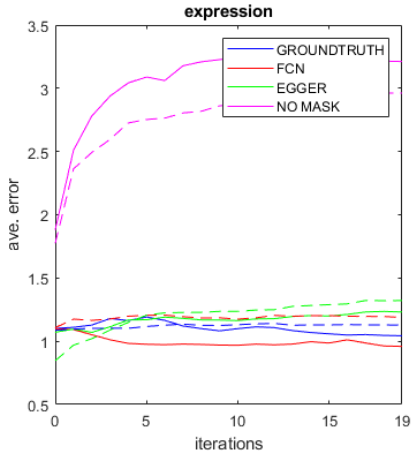
micros (mask: face12, rendering: bfm):

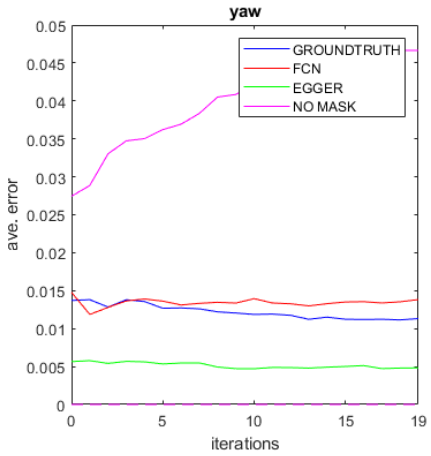
	Groundtruth	Egger	FCN	No mask
Mask				
Fit				

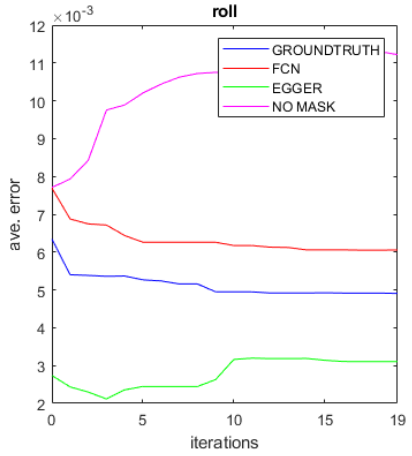
Evaluation of the "micros"-dataset(first 5 parameters solid, others dashed)

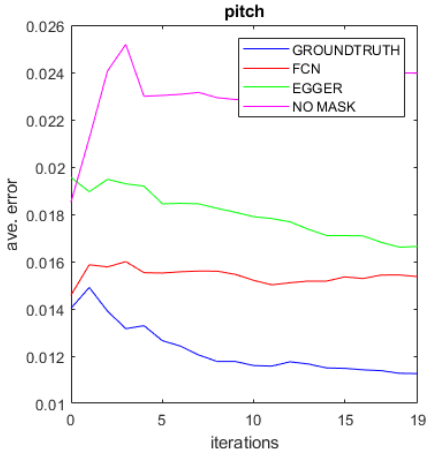


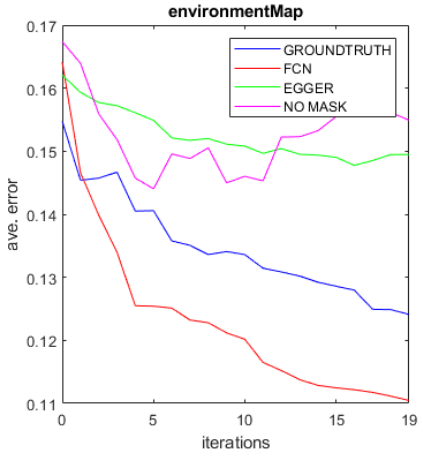


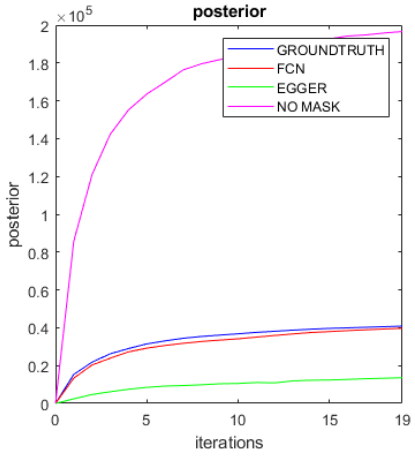








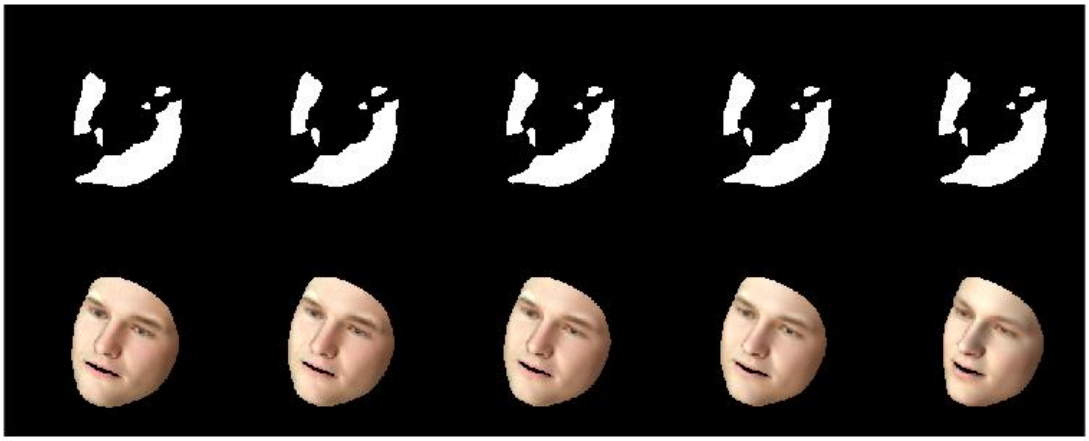




segmentation and mask of test0 in every 5th iteration with mask: GROTRU(from right to left)

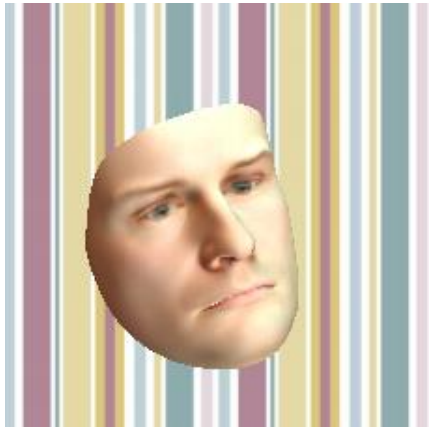
segmentation and mask of test0 in every 5th iteration with mask: EGGER(from right to left)

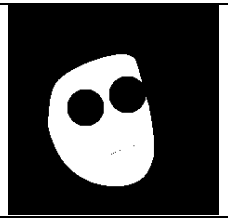
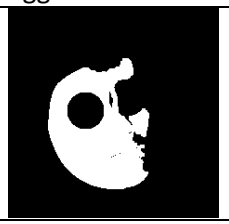
segmentation and mask of test0 in every 5th iteration with mask: FCN(from right to left)



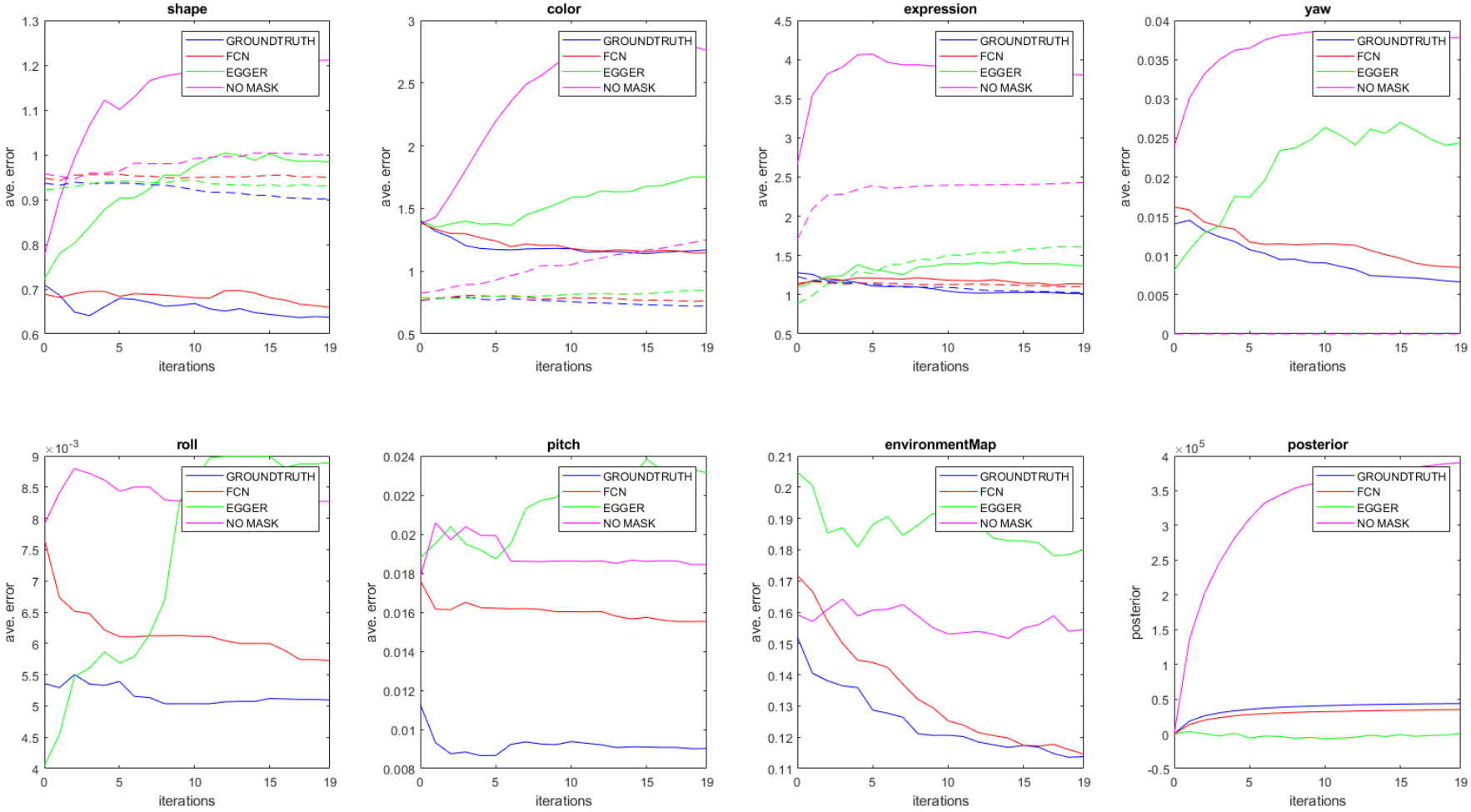
segmentation and mask of test0 in every 5th iteration with mask: NO_OCCLUSION(from right to left)

glasses (mask: face12, rendering: face12):

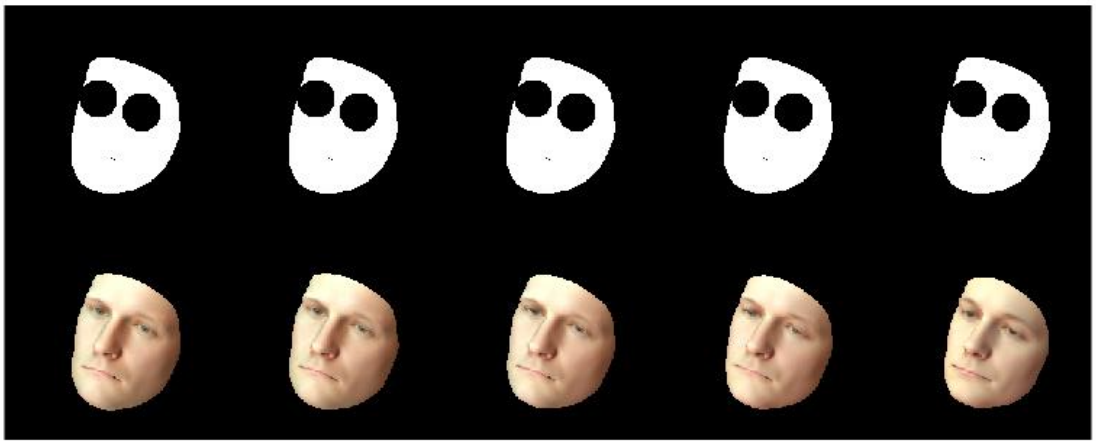


	Groundtruth	Egger	FCN	No mask
Mask				
Fit				

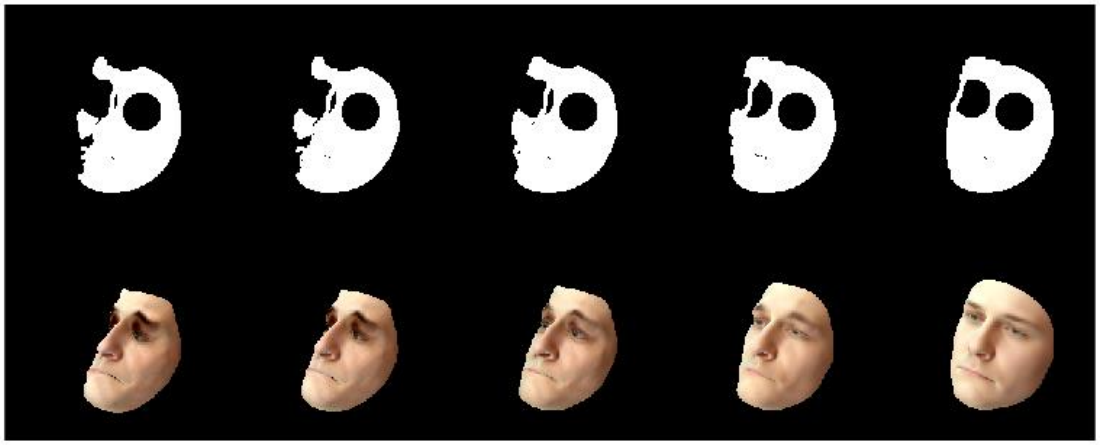
Evaluation of the "glasses"-dataset(first 5 parameters solid, others dashed)



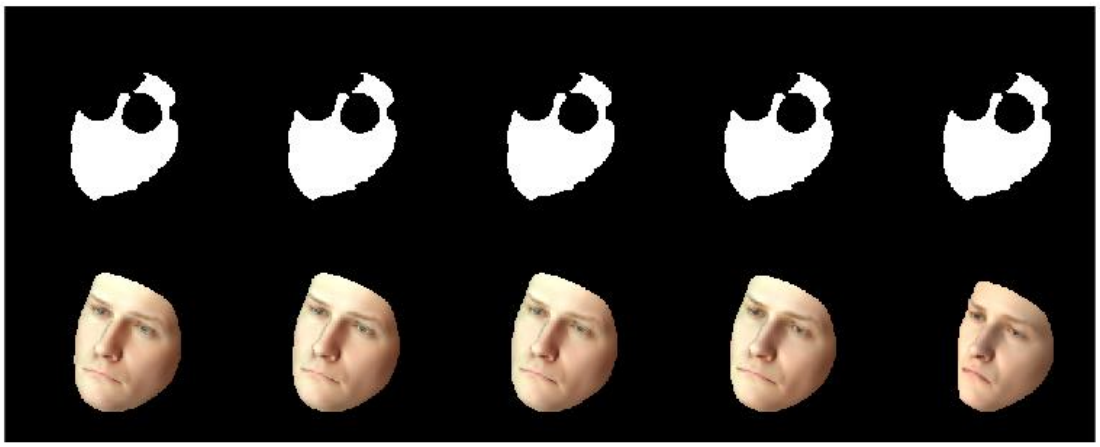
segmentation and mask of test0 in every 5th iteration with mask: GROTRU(from right to left)



segmentation and mask of test0 in every 5th iteration with mask: EGGER(from right to left)

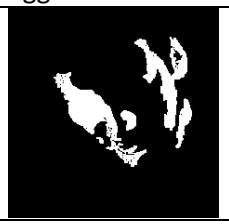


segmentation and mask of test0 in every 5th iteration with mask: FCN(from right to left)

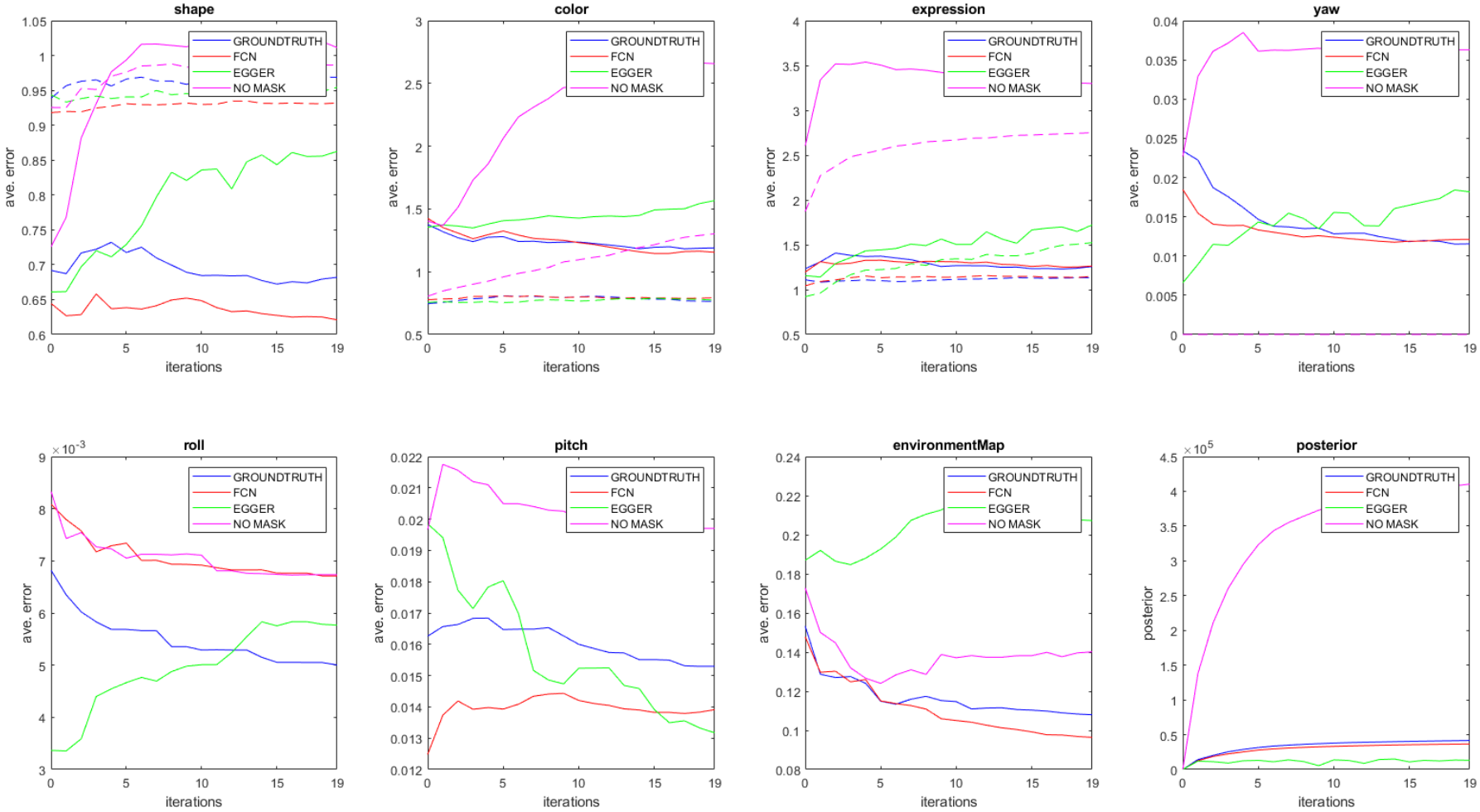


segmentation and mask of test0 in every 5th iteration with mask: NO_OCCLUSION(from right to left)

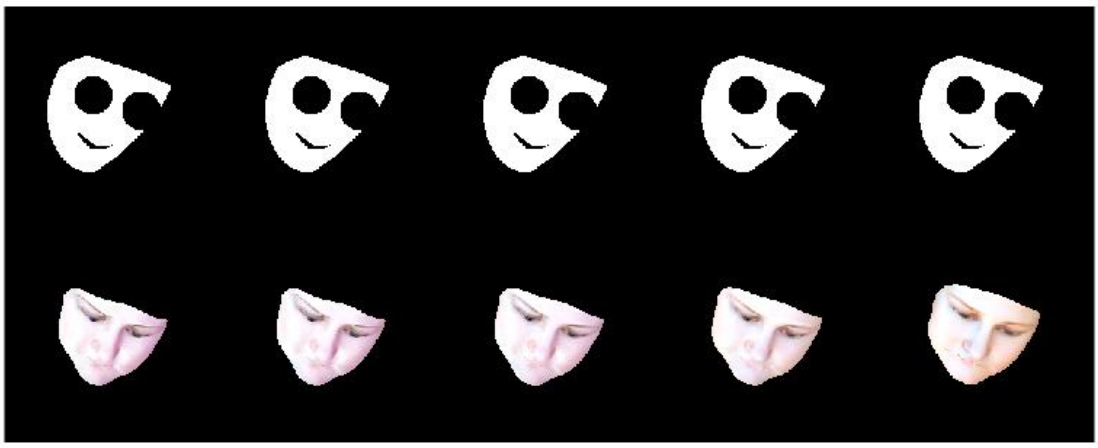
glasses (mask: face12, rendering: bfm):

	Groundtruth	Egger	FCN	No mask
Mask				
Fit				

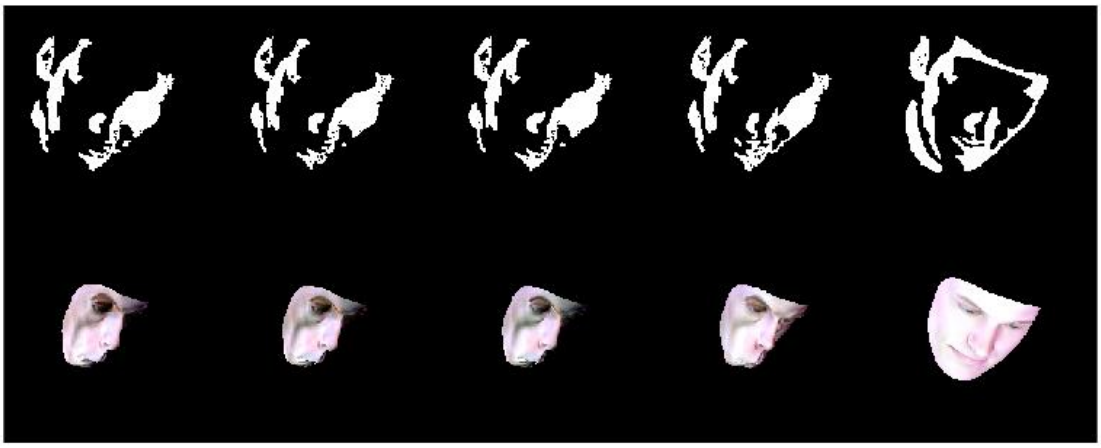
Evaluation of the "glasses"-dataset (first 5 parameters solid, others dashed)



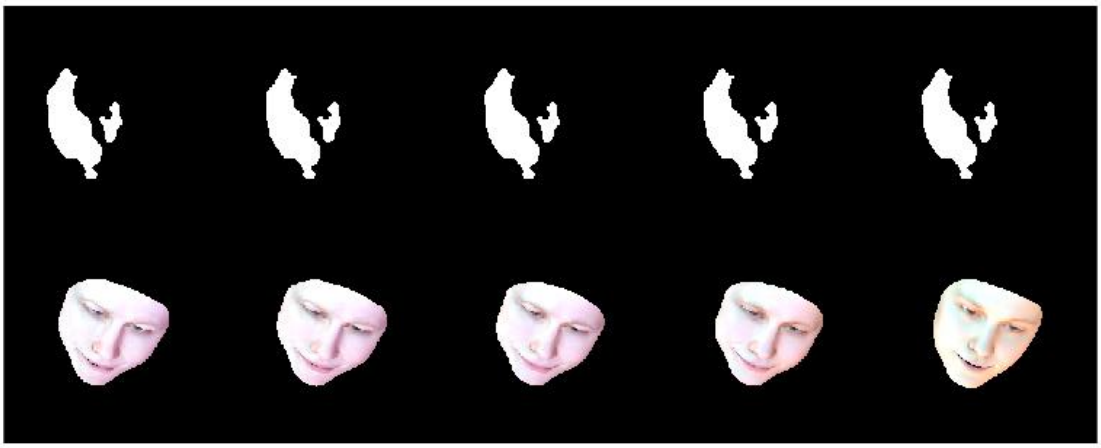
segmentation and mask of test2 in every 5th iteration with mask: GROTRU(from right to left)



segmentation and mask of test2 in every 5th iteration with mask: EGGER(from right to left)

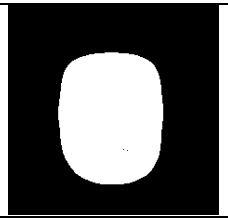
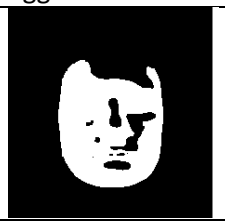


segmentation and mask of test2 in every 5th iteration with mask: FCN(from right to left)

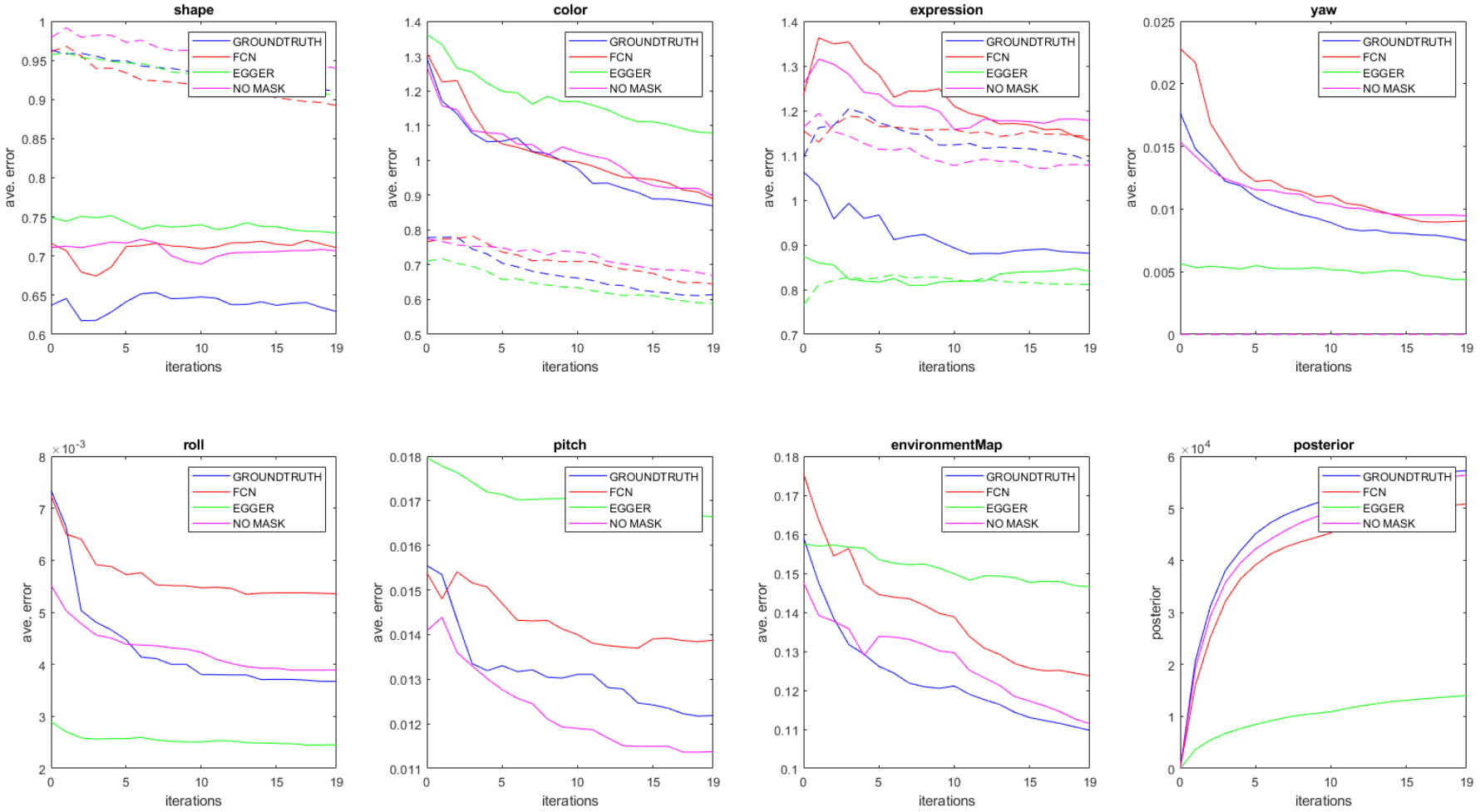


segmentation and mask of test2 in every 5th iteration with mask: NO_OCCLUSION(from right to left)

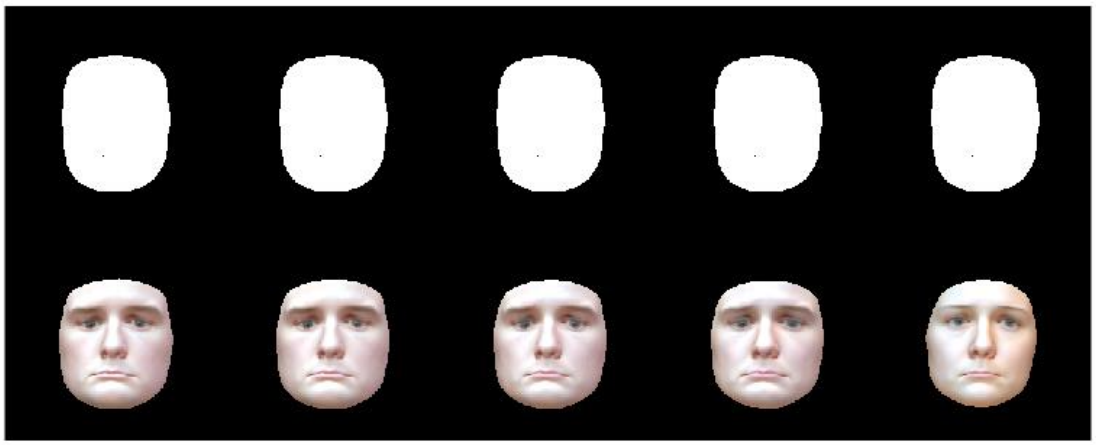
no occlusion (mask: face12, rendering: face12):

	Groundtruth	Egger	FCN	No mask
Mask				
Fit				

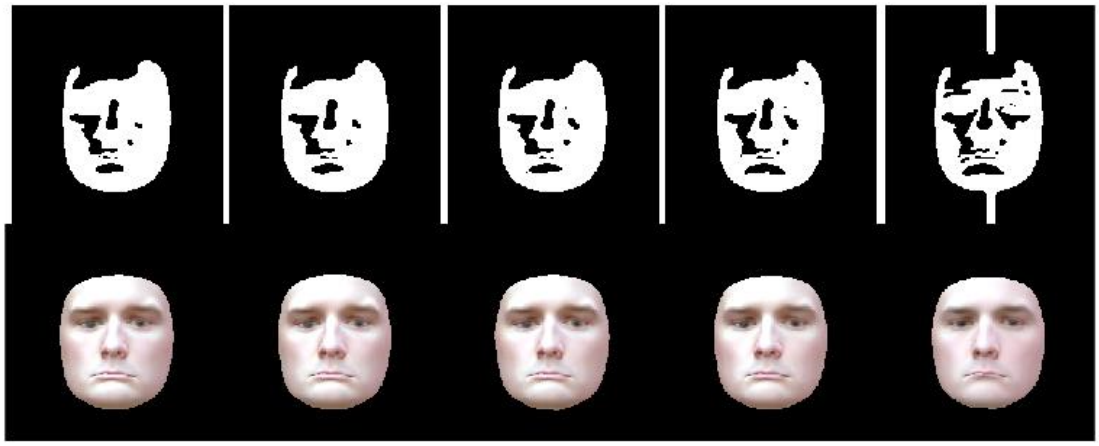
Evaluation of the "no occlusions"-dataset(first 5 parameters solid, others dashed)



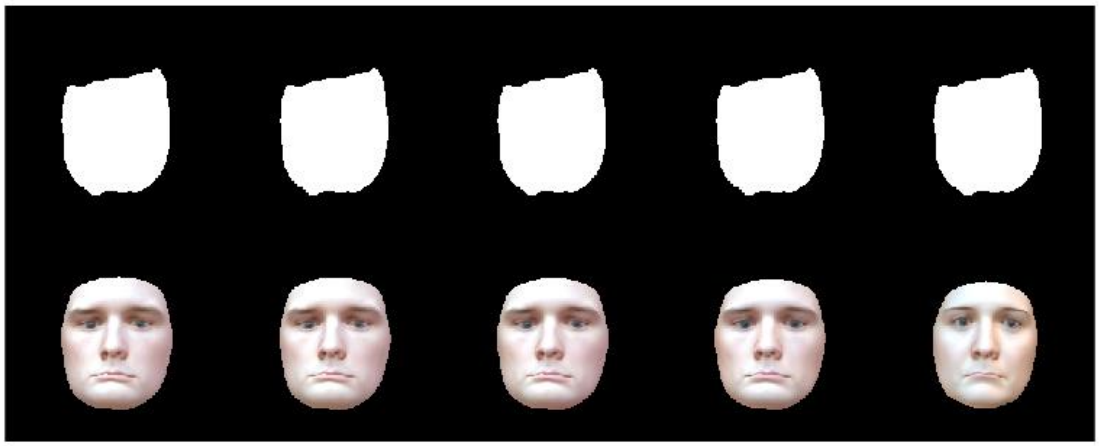
segmentation and mask of test7 in every 5th iteration with mask: GROTRU(from right to left)



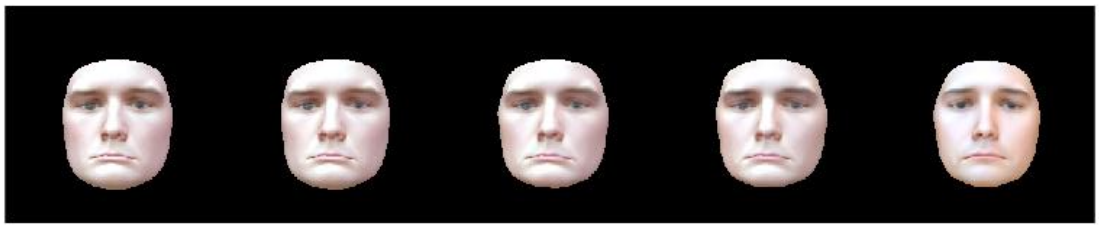
segmentation and mask of test7 in every 5th iteration with mask: EGGER(from right to left)



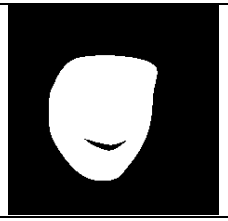
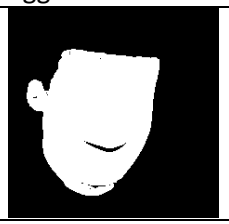
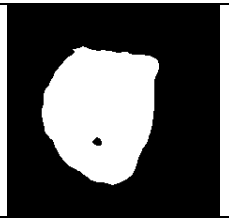
segmentation and mask of test7 in every 5th iteration with mask: FCN(from right to left)



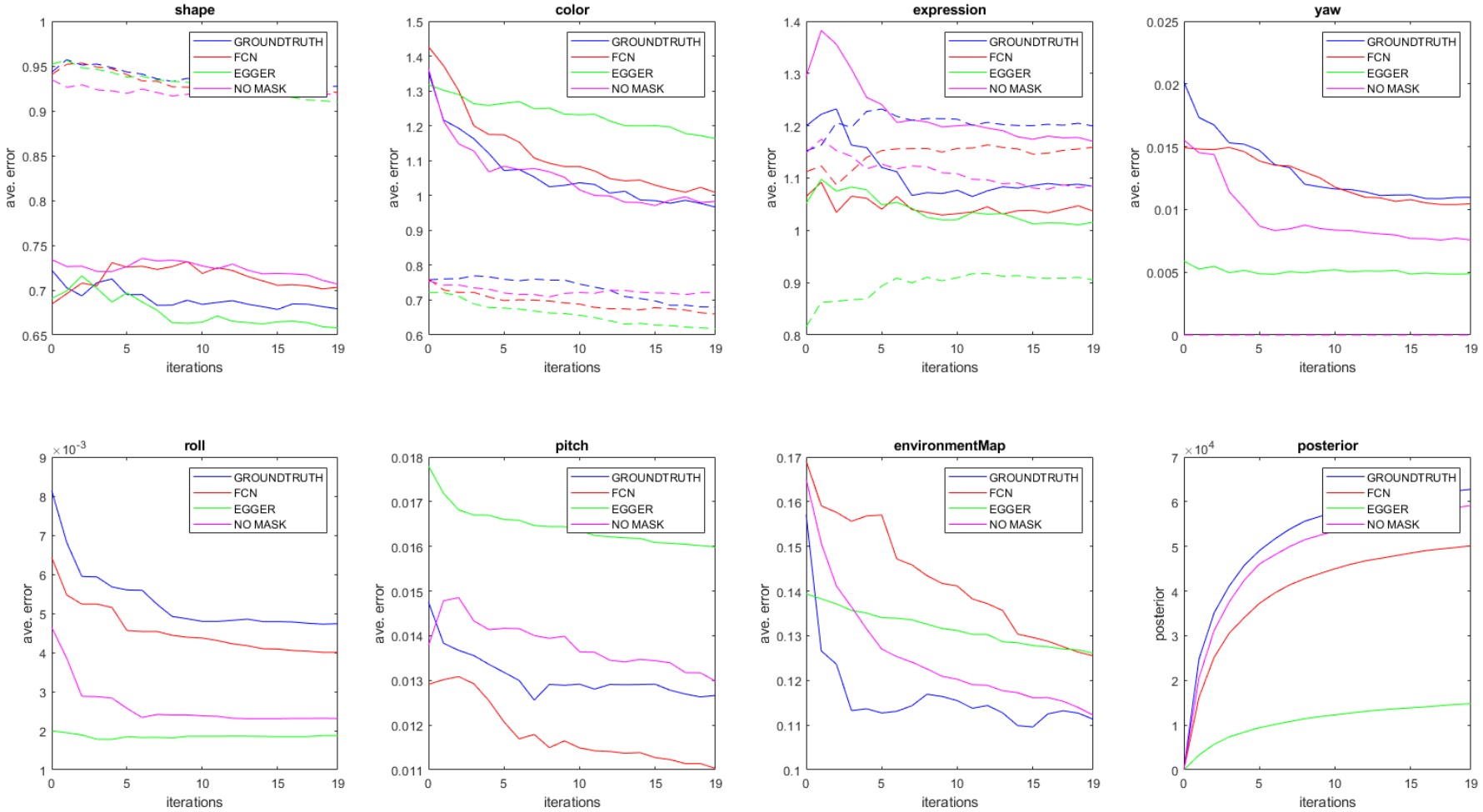
segmentation and mask of test7 in every 5th iteration with mask: NO_OCCLUSION(from right to left)



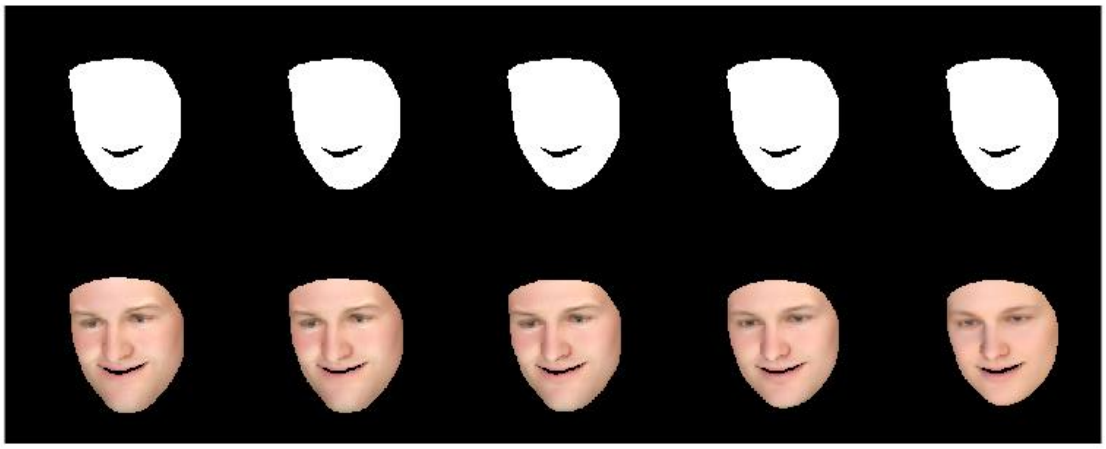
no occlusion (mask: face12, rendering: bfm):

	Groundtruth	Egger	FCN	No mask
Mask				
Fit				

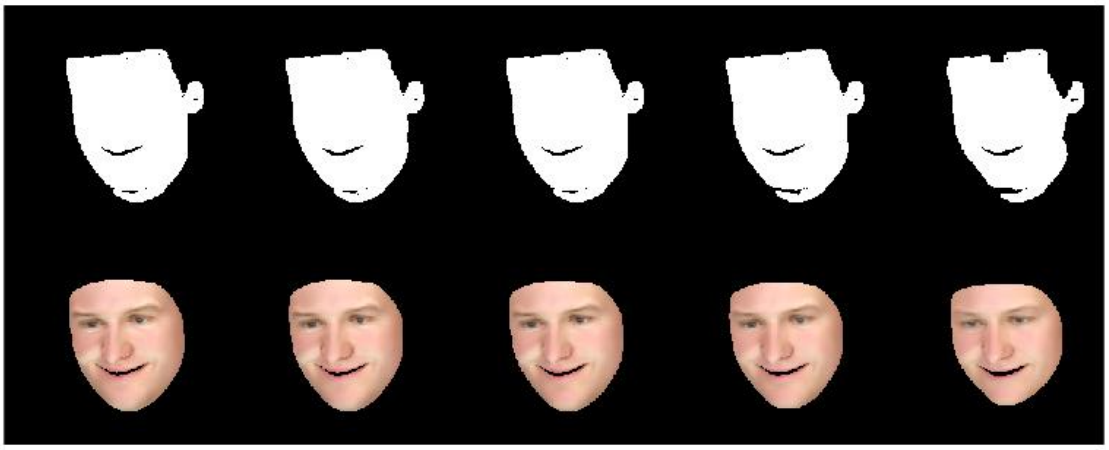
Evaluation of the "no occlusions"-dataset(first 5 parameters solid, others dashed)



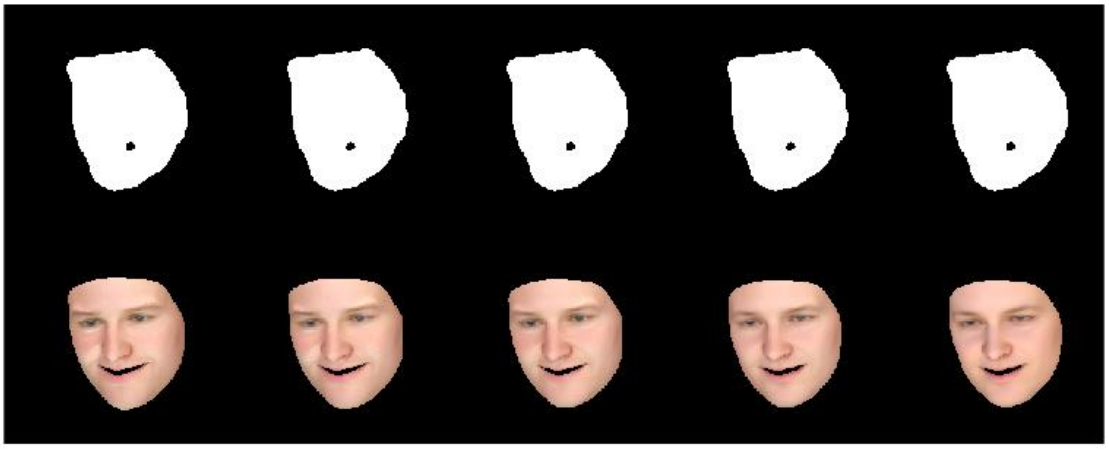
segmentation and mask of test6 in every 5th iteration with mask: GROTRU(from right to left)



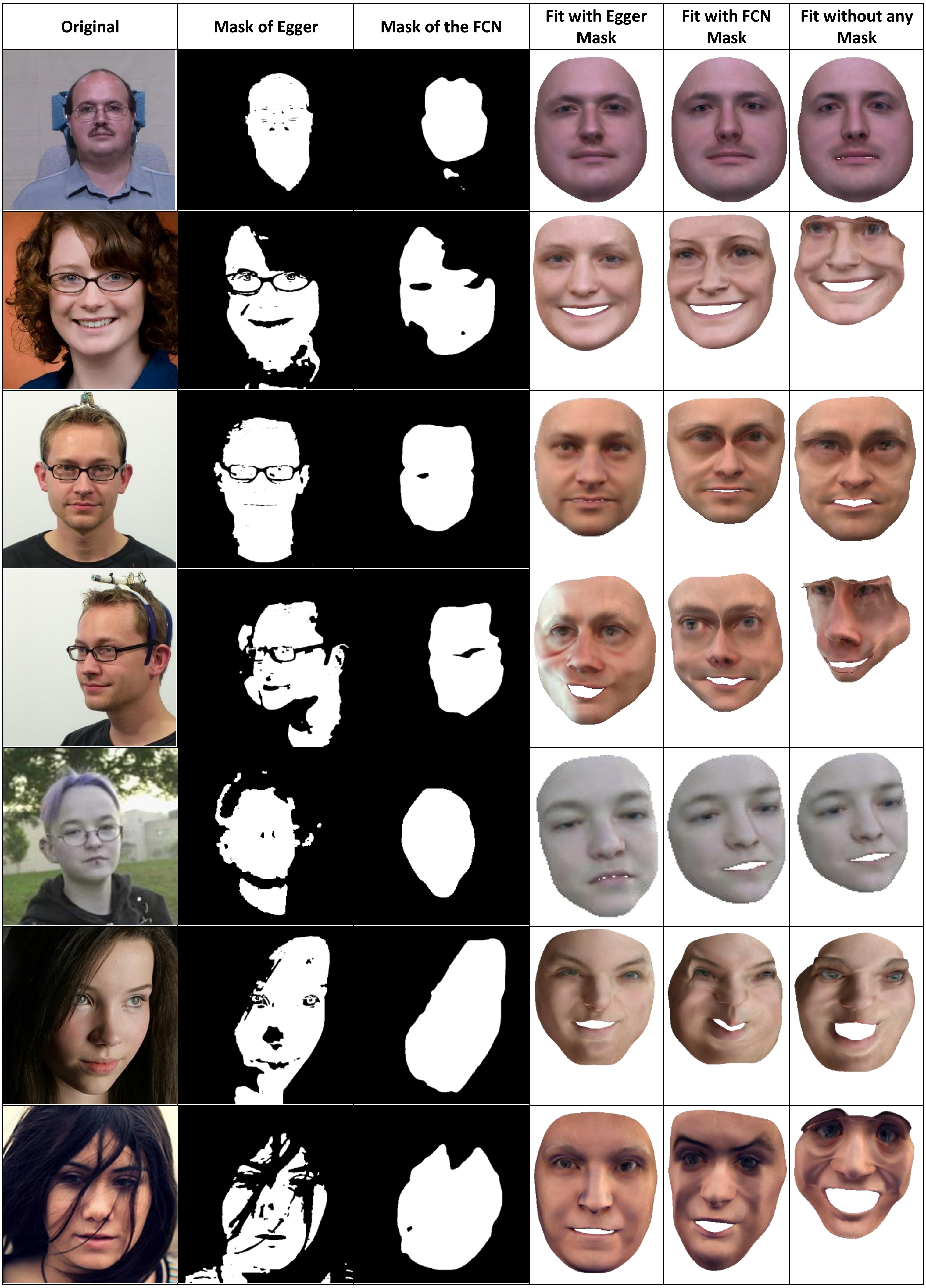
segmentation and mask of test6 in every 5th iteration with mask: EGGER(from right to left)

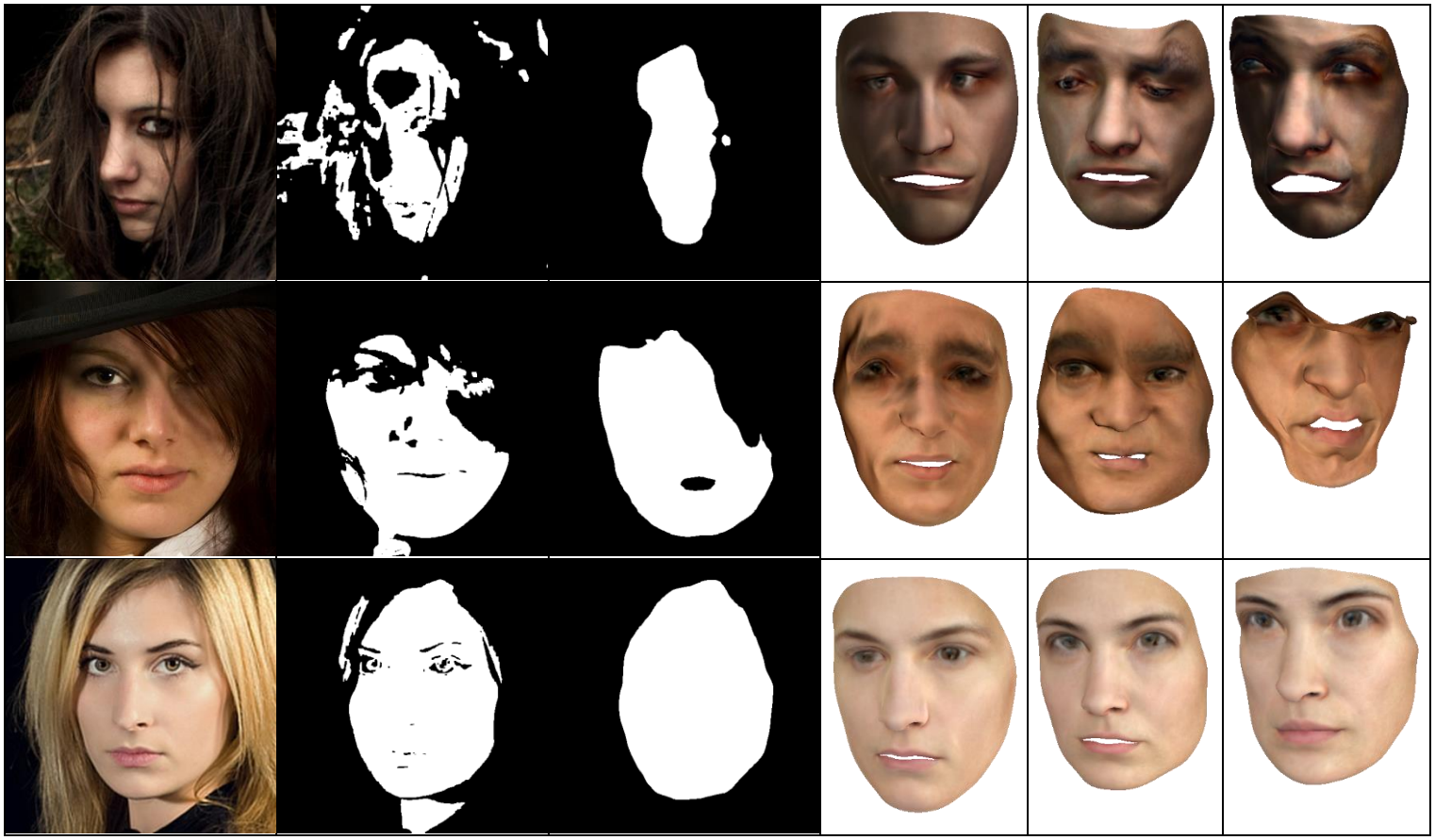


segmentation and mask of test6 in every 5th iteration with mask: FCN(from right to left)



segmentation and mask of test6 in every 5th iteration with mask: NO_OCCLUSION(from right to left)


Original
Mask of Egger
Mask of the FCN
Fit with Egger Mask
Fit with FCN Mask
Fit without any Mask



Declaration on Scientific Integrity
Erklärung zur wissenschaftlichen Redlichkeit

includes Declaration on Plagiarism and Fraud
beinhaltet Erklärung zu Plagiat und Betrug

Author — Autor
Elias Arnold

Matriculation number — Matrikelnummer
14-930-770

Title of work — Titel der Arbeit
An Empirical Comparison of Deep Learning and 3DMM-based Approaches for Facial Occlusion Segmentation

Type of work — Typ der Arbeit
Bachelor-Thesis

Declaration — Erklärung
I hereby declare that this submission is my own work and that I have fully acknowledged the assistance received in completing this work and that it contains no material that has not been formally acknowledged. I have mentioned all source materials used and have cited these in accordance with recognised scientific rules.

Hiermit erkläre ich, dass mir bei der Abfassung dieser Arbeit nur die darin angegebene Hilfe zuteil wurde und dass ich sie nur mit den in der Arbeit angegebenen Hilfsmitteln verfasst habe. Ich habe sämtliche verwendeten Quellen erwähnt und gemäss anerkannten wissenschaftlichen Regeln zitiert.

Basel, 10.08.2018

Signature — Unterschrift