Link to data:
https://atreus.informatik.uni-tuebingen.de/seafile/d/8e2ab8c3fdd444e1a135/?p=%2FSegmentationEye&mode=list

# The applicability of Cycle GANs for pupil and eyelid segmentation, data generation and image refinement

Wolfgang Fuhl
University Tübingen
Sand 14, Tübingen, Germany
wolfgang.fuhl@uni-tuebingen.de

David Geisler
University Tübingen
Sand 14, Tübingen, Germany
david.geisler@uni-tuebingen.de

Wolfgang Rosenstiel
University Tübingen
Sand 14, Tübingen, Germany
Wolfgang.Rosenstiel@uni-tuebingen.de

Enkelejda Kasneci
University Tübingen
Sand 14, Tübingen, Germany
enkelejda.kasneci@uni-tuebingen.de

## Abstract

*Eye tracking is increasingly influencing scientific areas such as psychology, cognitive science, and human-computer interaction. Many eye trackers output the gaze location and the pupil center. However, other valuable information can also be extracted from the eyelids, such as the fatigue of a person. We evaluated Generative Adversarial Networks (GAN) for eyelid and pupil area segmentation, data generation, and image refinement. While the segmentation GAN performs the desired task, the others serve as supportive Networks. The trained data generation GAN does not require simulated data to increase the dataset, it simply uses existing data and creates subsets. The purpose of the refinement GAN, in contrast, is to simplify manual annotation by removing noise and occlusion in an image without changing the eye structure and pupil position. In addition 100,000 pupil and eyelid segmentations are made publicly available for images from the labeled pupils in the wild data set (DOWNLOAD). These will support further research in this area.*

## 1. Introduction

The commonly used pupil recognition algorithms are based on rule-based methods [12, 16, 40] using extracted edges or specially designed filters. The reason for this is the resource saving computations required to apply them. Disadvantages of these methods are a non-constant runtime as well as weaknesses in certain challenges. For edge based methods these are blurred images, only partially visible pupils, poor light conditions, strong reflections as well as low resolution images.

Since eye tracking is constantly moving into new areas such as driver observation [30, 3], virtual reality [19, 34, 5], augmented reality [22, 35], microscopy [33, 6] and many more, the requirements for image processing are also becoming ever higher. Newer methods for pupil recognition use machine learning methods to be usable in a variety of applications as well as to be adapted to new challenges. A good example from industry is the Pupil Invisible [25, 51, 47] eye tracker. It uses modern convolutional neuronal networks and offers the possibility to store data on a server to further improve the detection. This allows the eye tracker to adapt to new scenarios without the need of new algorithm development.

In this work, we propose a framework to train Generative Adversarial Networks (GANs) [18] using the cyclic loss function. The reason for this is that GANs can be used for a wide range of applications in the field of image-based eye tracking. The first purpose is the image segmentation for usage in data post-processing and dataset generation. The segmented data allows to improve the accuracy of eye tracking experiments offline and can also be used to train resource saving and realtime applicable machine learning algorithms, e.g. random forests for online usage [36]. Since there is a plethora of possible camera configurations, perspectives, and light conditions (RGB, NIR), we expect that trained network are not always applicable to all eye tracking data. Therefore, the second GAN is trained to refine images. This includes removing occlusions, noise, and adjusting the image contrast without changing the eye shape or the pupil position. This will help detection algorithms and also support annotation of ground truth data. The third GAN is used to generate additional training data with annotations based on provided data. This data can be generated

using rendering [48].

## 2. Related works

The main focus of the eye tracking community in the past years was a robust and reliable pupil signal. Therefore, a plethora of approaches have been proposed and summarized in [17, 42] for head-mounted eye trackers. Since the resolution, and the light and contrast conditions for stationary cameras (remote eye trackers) differ strongly from those of head-mounted eye trackers, both types were considered separately for a very long time. A summarization for their respective pupil detection algorithms can be found in [11]. The major advances in the field of pupil extraction where achieved by edge detection [40]. This approach was further improved by edge filtering [12, 16] and edge combination [16, 37]. Since edge detection fails for blurred images, other approaches using adaptive thresholds [20] and segment selection [23] where proposed. Then, machine learning also lead to CNN based pupil detection methods [15, 43], where multi-stage approaches are applied to achieve high accuracies. Other approaches, like random ferns [10] and oriented edge boosting [9], where also applied for pupil center detection, but only the latter was extended by an ellipse fit to segment the pupil.

The field of the eyelid and eye-opening extraction has received only a small amount of attention. In [46], the first attempt based on edge detection and approximation with parabolas was made. For iris recognition, an improved approach was proposed in [4]; based on the iris location, the eyelids are searched as curvilinear edges. Afterward, a spline was fitted to these edges. Another approach used the largest edges in the image [1] after anisotropic diffusion. Since the eyelid edges can be covered by eyelashes or blurred through motion, a pure intensity-based approach partitioning the image in regions was proposed [39]. This approach was further refined by computing a likelihood map based on texture patches for the eye corners and the central point of the upper and lower eyelid [49]. Since the likelihood map had proven to be robust, another approach used image patch statistics for the computation in combination with edge detection [13].VASIR [29] an open source tool developed by the National Institute of Standards and Technology, uses the linear Hough transform for iris segmentation followed by a third order polynomial fitting for eyelid extraction. An optimization that searches for four eyelid points based on the optimal oriented edge value was proposed in [14]. Modern machine learning algorithms for landmark detection [36] where also applied for eyelid extraction [8] together with histograms of oriented gradients and support vector machines.

In recent years, CNNs achieved a considerable breakthrough for image segmentation. The transposed convolution filters where proposed in [31], which allowed to scale the output information of a network and remove the fully connected layers. An alternative to this approach is encoder and decoder networks [2], which up-sample based on pooling indices from the encoder. Both use a softmax loss function to predict the labels. For further improvement of segment borders, a region loss function was proposed [21]. They used aligned region of interests for loss computation, which eliminates inter-class competitions. With the uprising of image generation using a generator and discriminator [18] and the cycle loss function [52]. CNNs attained the ability to transfer styles between images, which was already used to perform a semantic segmentation [52] and can be used for data generation using simulated data [38]. In this work, we use GANs with the cyclic loss function for image segmentation, data generation, and image refinement.

## 3. Method

Generative Adversarial Network (GAN) consists of two competing networks. The generator (Figure 1) attempts to create the most authentic representation of the output distribution. Therefore, it learns to transform the input distribution into the output distribution ($G(A) \rightarrow B$), which is also known as style transfer [18]. The discriminator, tries to find out whether it is a generated image or a true picture of the output distribution ($D(G(A)) \in B$) [18].

Therefore, the discriminator, it has to minimize its classification accuracy on the data distribution B ($log(D(B))$) [18]. In contrast, the generator has to maximize the error of the discriminator based on the generated data ($log(1 - D(G(A)))$) [18]. Since this approach is difficult to train because the discriminator tends to overfit and thereby rejects everything, the cyclic loss was proposed [52]. The difference to the GAN is that both directions are considered ($G_1(A) \rightarrow B$ and $G_2(B) \rightarrow A$), which allows an additional loss formulations between the two generators and is called the cyclic loss function ($A - G_2(G_1(A))$ and $B - G_1(G_2(B))$). It has to be mentioned that for the cyclic loss, two generators and two discriminators are used. One pair of generator and discriminator generates a new image and the other pair reconstructs the input.

Figure 1 shows the used architecture for our evaluation. It consists of three convolution blocks with batch normalization and the rectifier liner unit (ReLU). Instead of pooling, our network uses the stride parameter for downscaling. These layers are responsible for feature extraction. The style transfer or distribution transformation is performed by consecutive residual blocks with equal block depth. Each residual block consists of three convolution layers with batch normalization and the rectifier linear unit. Afterward, the new image is generated using transposed convolution layers.

For our discriminator, we used four convolution layers

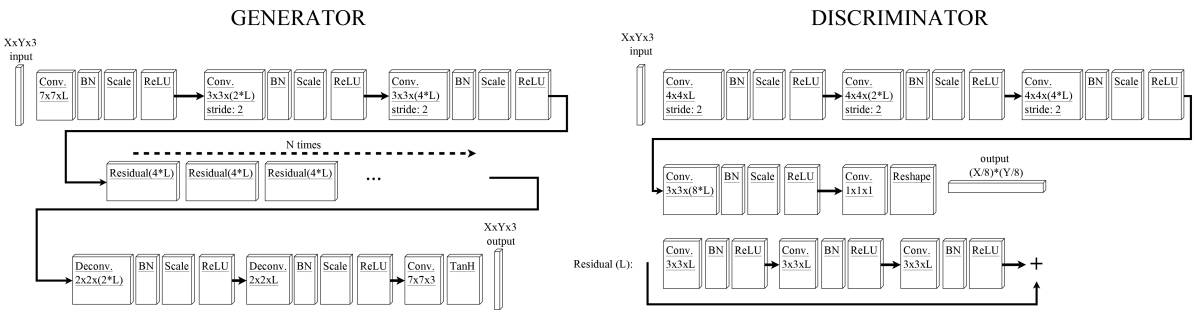


Figure 1. General architecture of the used generator and discriminator for all CNNs.

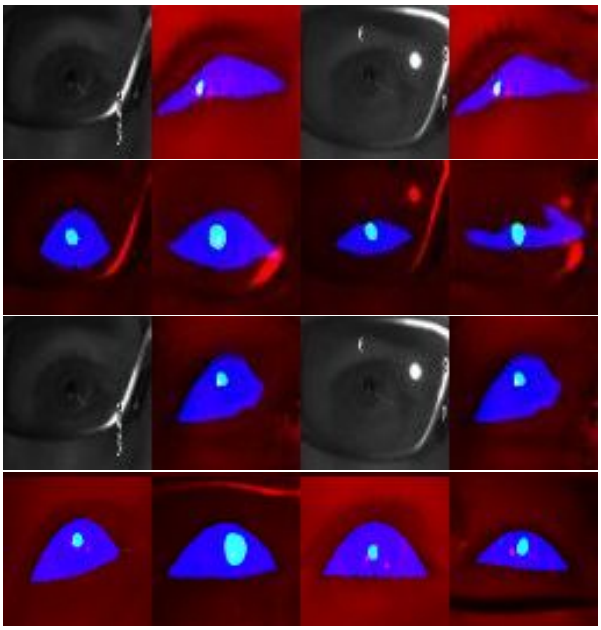

Figure 2. Results after one (first row) and ten epochs (second row) for data segmentation (first four columns) and data generation (last four columns).

with batch normalization and the stride parameter for downscaling. The last layer is a convolution to produce a one-dimensional output. These architectures follow the original structure proposed in [52]. The parameter $L$ specifies the block depth of convolution layers in the network. With the parameter $N$, the amount of residual block in the central part of the generator are specified. For our models we used $N = 5$ and $L = 64$. In addition, the input size of the network can be adjusted. For the runtime and memory consumption evaluation we used the input resolutions $32 \times 32$, $64 \times 64$, $96 \times 96$, $128 \times 128$ and $256 \times 256$ (Table 4). In the segmentation task (Table 1) we used $64 \times 64$ and for the pupil center detection evaluation (Table 2) we used a resolution of $128 \times 128$. The refinement GAN uses the highest resolution ($256 \times 256$) to reduce the impact of the upscaling operation to the original size of the image. We used Caffe [24] for training and execution of our models. The least square loss function was used as it stabilizes the training for GANs in comparison to the negative log likelihood [32]. In addition we used a buffer of 100 generated images to reduce oscillation during training as proposed in [38]. The used optimizer was Adam [28] with momentum set to 0.5 and a fixed learning rate of $0.0001$.

For the data augmentation image flips in horizontal and vertical direction were used as well as gaussian blur with a factor of $1 - 1.2$. Additionally the image was shifted and the resulting margin was filled with random noise. Further data augmentations were random noise up to 20%, squares and ellipses inserted at random positions and reflections. The squares and ellipses were also filled with random values. For the reflections, images from the ImageNet imagenetcvpr09 dataset were randomly selected and placed over the original image wan2017benchmarking.

Translated with www.DeepL.com/Translator

### 3.1. Image segmentation task

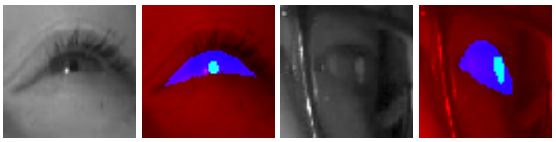

Figure 3. Input and output pairs of the finally trained cycle GAN.

For the training of the image segmentation, we used paired images. While it is also possible to use unpaired training, our results improved $\approx 10\%$ using paired image examples (Table 1). The input to the network is a grayscale image inserted into all three channels with a resolution of $64 \times 64$ pixels. As output, we used the grayscale image in the red, the visible part of the eye in the blue and the pupil in the green channel. It makes it easier to evaluate the segmentation based on the output and also allows the network to learn the reconstruction of the input image similar to a deep autoencoder. Row one and two in Figure 2 show some examples for the first and tenth epoch. While these are early stages in training, it can be seen that the segmentation of the tenth epoch significantly improved, whereas the reconstruction is still incomplete. Figure 3 shows the segmentation

results of our cycle GAN after ≈200 epochs. As can be seen, the image was reconstructed and segmented.

### 3.2. Image generation task

For data generation, we separated the input dataset into two sets. Each image was constructed equally to the output image of the segmentation. It means that in the red channel we inserted the grayscale image, in the blue channel the eyelid and in the green channel the pupil area. Therefore, input and output contained the image and the segmentation. The training itself was done with unpaired image examples; Figure 4 shows some generated images. More interesting is that the GAN learned to add a bright pupil (third image in the top row) as well as eyelashes (last image top row) or glass frames with reflections. Therefore, our approach we used the same data distribution we can use the results of both generators as new training data.

### 3.3. Image refinement task

The data generator cycle GAN can increase the amount of training data for human related challenges like eyelashes, eye shape, pupil size, moles in the eye area, and many more. Challenges related to the image acquisition such as noise, reflections, and illumination are not covered if they are not frequently available in the training data. In addition, new images with challenges that have to manually annotated are usually difficult to segment by hand. Therefore, we propose to use the cycle GAN for image refinement, which gives us a generator for image augmentation too. Our data augmentation includes random noise, image patch covering, blurring, contrast variations, and reflections. For adding reflections, we used the approach from [45], where the reflection is assumed to be a blurred additive of a second image. Examples of our data augmentation and the results of the refiner are shown in Figure 5. In the first column of the figure, the input images are shown. The second column shows when changes in the image contrast, blur, and noise is added. For the third column, we added a cat image a reflection. The fourth column shows the input image with reflections, noise, blur, and contrast change. For training, we used paired images, where the images from the dataset were used multiple times with different challenges.

## 4. Evaluation

For the training of our cycle GANs, we used the dataset proposed in [8] which consists of $16,200$ hand-labeled images with a resolution of $1280 \times 752$ pixels from six subjects. For the pupil center detection evaluation we used additionaly $\approx 25,000$ images from nine subjects which are not publicly available. The recording system was a near-infrared remote camera in a driving simulator setting. Figure 6 shows images from the dataset. In the first row, the recordings are shown. In the second row, the eye regions are shown with annotations. As can be seen, the dataset contains images with reflections as well as open and closed eyes together with head rotations.

For the comparison to the state-of-the-art, we made a cross-subject evaluation. Therefore, we trained our network on all but one subject and used this left out subject for evaluation. We repeated this procedure until each subject was evaluated once. The same was done for the landmark detection algorithm [26] implemented in DLIB [27]. For [14], we only need to evaluate all subjects since it does not have to be trained. As metric, we used the Jaccard index ($\frac{GT \cap DT}{GT \cup DT}$), which is the cut between the detected area ($DT$) and the ground truth area ($GT$) divided by their union. This metric is a common metric for segmentation quality analysis, where 0.5 can be seen as a good result.

Table 1. Average Jaccard index per algorithm cross validated on the dataset from [8]. Best result in bold.

| Class: | Eyelid | Pupil |
|---|---|---|
| [26] | 0.52 | - |
| [14] | 0.52 | - |
| ElSe remote [11] | - | 0.33 |
| BORE ellipse [9] | - | 0.65 |
| DeepVOG [50] | - | 0.22 |
| real data unpaired | 0.7 | 0.62 |
| real data paired | 0.79 | 0.72 |
| gen. data unpaired | 0.71 | 0.63 |
| gen. data paired | 0.78 | 0.72 |
| paired with gen. and real data | **0.84** | **0.78** |

Table 1 shows the results of our segmentation GAN without data generation and with additional data from the generator GAN. As can be seen, our approach outperforms the state-of-the-art by a large margin. The main error stems from the lower resolution ($64 \times 64$) in comparison to the original ($100 \times 74$). For the pupil segmentation, the accuracy of our approach drops, which is due to one pixel having a higher impact on the error. While the first four algorithms run on a single CPU core in real time, our approach requires $\approx 18ms$ on an NVIDIA GTX 1050Ti with a resolution of $64 \times 64$. For the DeepVOG [50] and our model we used a threshold parameter of $0.9$ for the heat map output.

In Table 2, the results for pupil center detection in comparison to the state-of-the-art on puplicly available datasets are shown. The evaluation metric is the euclidean distance to the ground truth annotation. If the distance is below or equal to $5px$ the position is seen as accurate which was proposed in [12] to compensate for annotation inaccuracies. The values in Table 2 represent the percentage of images where the detected position was equal or below the $5px$. Our approach requires $\approx 33ms$ on an NVIDIA GTX 1050Ti with a resolution of $128 \times 128$. The other approaches exception of DeepEye [44], [7], and DeepVOG [50] run in real time on a CPU. For DeepVOG [50] and our model we used the same threshold parameter ($0.9$) as in the segmentation evaluation. The extraction of the pupil center position

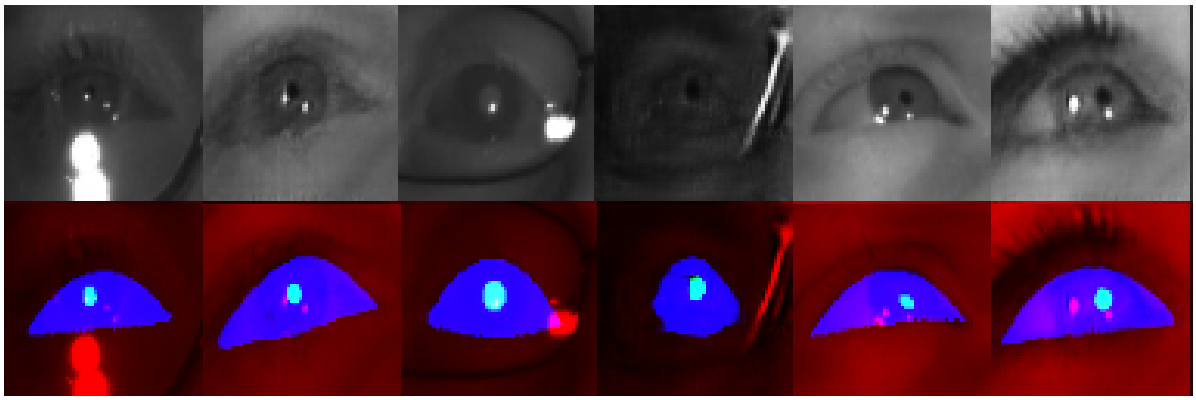

Figure 4. Generated images using our trained cycle GAN for data generation.

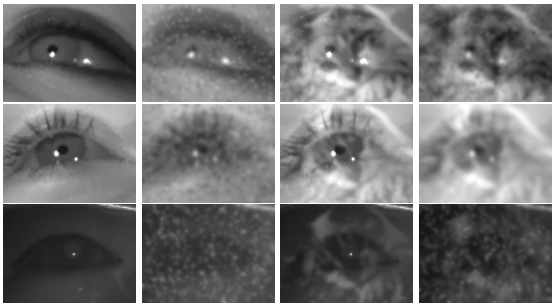

Figure 5. Augmented images for refinement and augmentation training.

Table 2. Average detection result over all subjects from the datasets provided with ElSe [12], ExCuSe [16], PNET [15], Świrski [40], and labeled pupils in the wild [42]. The five pixel euclidean distance was used to compensate for inaccurate annotations as proposed in [12, 16]. Best result in bold.

| Datasets: | [12, 16, 15] | [40] | [42] |
|---|---|---|---|
| ElSe [12] | 0.67 | 0.81 | 0.54 |
| ExCuSe [16] | 0.54 | 0.86 | 0.50 |
| Świrski [40] | 0.30 | 0.77 | 0.49 |
| PURE [37] | 0.72 | 0.78 | 0.73 |
| CBF [10] | **0.91** | - | - |
| PNET [15] | 0.76 | - | - |
| DeepEye [44] | 0.83 | 0.54 | 0.50 |
| [7] | 0.79 | 0.74 | 0.84 |
| DeepVOG [50] | 0.43 | 0.52 | 0.62 |
| Prop. ($128 \times 128$), paired, real data | 0.87 | 0.89 | 0.85 |
| gen.&real data | **0.91** | **0.93** | **0.89** |

out of the segmentation we used the center of mass from the segmented area.

As can be seen in Table 2 and Table 1 the trained GANs are applicable to the segmentation and pupil center detection task. In addition, the generated data using the generator GAN improves the results for both experiments.

In Table 3, the improvement using refined images for pupil center detection on publicly available datasets is shown. As can be seen the results of all algorithms was

Table 3. Average detection result over all subjects from the datasets provided with ElSe [12], ExCuSe [16], PNET [15], Świrski [40], and labeled pupils in the wild [42]. The five pixel euclidean distance was used to compensate for inaccurate annotations as proposed in [12, 16].

| Datasets: | [12, 16, 15] | [40] | [42] |
|---|---|---|---|
| Original | | | |
| ElSe [12] | 0.67 | 0.81 | 0.54 |
| Świrski [40] | 0.30 | 0.77 | 0.49 |
| PURE [37] | 0.72 | 0.78 | 0.73 |
| Down & upscaling of the image | | | |
| ElSe [12] | 0.58 | 0.79 | 0.69 |
| Świrski [40] | 0.14 | 0.78 | 0.48 |
| PURE [37] | 0.65 | 0.77 | 0.72 |
| Refined images | | | |
| ElSe [12] | 0.74 | 0.81 | 0.73 |
| Świrski [40] | 0.69 | 0.79 | 0.71 |
| PURE [37] | 0.74 | 0.79 | 0.74 |

improved on the refined images. This improvement does not rise from the down and upscaling as can be seen in the central part of the table. It is even the case that scaling down and up worsens the results. An exception to this is the labeled pupils in the wild data set in which ElSe has improved by 15%. Another very interesting result is the Swirski data set in which only a minimal or no improvement (ElSe) was achieved. This is due to the fact that part of the data set contains occluded pupils through the eyelid. The second step of ElSe is effective for such images but does not benefit from the refinement. All evaluated algorithms use edges as a feature for pupil detection, which shows that the refinement GAN also improves the fine structure of an image.

Table 4 shows the runtime and memory usage for different input resolutions. For our architecture, we set the parameters $N = 5$ and $L = 64$ as shown in Figure 1. The input resolution of $256 \times 256$ exceeded the memory capacity of our GPU, which forced our framework to only partially load the models on the GPU. This is the reason of the increased runtime for the training of this model.

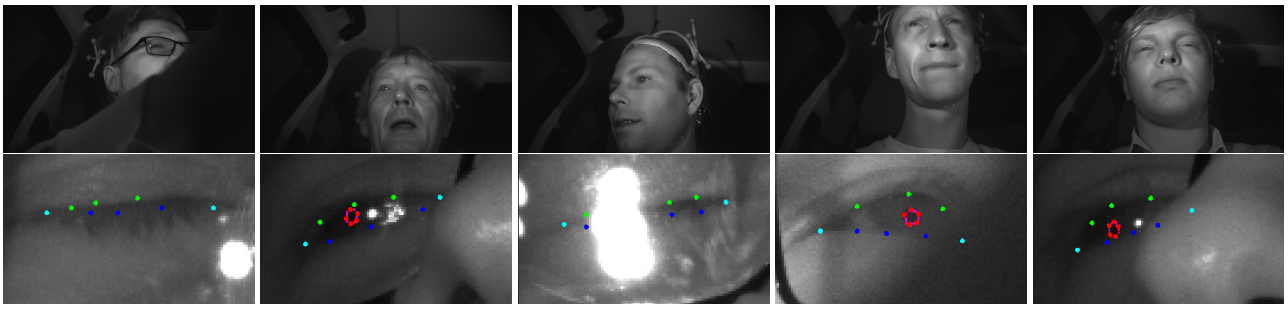

Figure 6. Dataset for training of our GANs [8].

Table 4. Average runtime (10000 samples) on a NVIDIA GTX 1050ti (4GB Ram) for training and execution using different input and output resolutions with a batch size of one and the same architecture.

| Resolution | Execution(Memory) | Training(Memory) |
|---|---|---|
| $32 \times 32$ | 16.7ms (800MB) | 341ms (1.5GB) |
| $64 \times 64$ | 17.4ms (850MB) | 418ms (1.7GB) |
| $96 \times 96$ | 24ms (900MB) | 515ms (2.0GB) |
| $128 \times 128$ | 33ms (1GB) | 657ms (2.3GB) |
| $256 \times 256$ | 91.4ms (1.5GB) | 1715ms (5GB) |

## 5. Labeled pupils in the wild segmentations

For further research in the field of pupil and eyelid detection we have segmented 100,000 images of the well-known "Pupils in the wild" data set. These segmentations are made publicly available. In the selection of images, we have focused on challenges that often overtax the current state of the art algorithms. In addition, some images are also from simple data sets, so that the basic task is covered by the annotations without additional image-based challenges. The original dataset contains 130,856 images separated into 66 videos from 22 participants. We have always selected whole videos so that tracking algorithms can be evaluated and trained on them. For the selection of the particularly challenging videos we have used the results of the state-of-the-art algorithms (Table 5).

As can be seen, the videos 3, 4, 5, 9, 10, 11, 13, 15, 18, 19 and 22 are especially challenging. These videos contain different challenges. Figure 7 shows pictures of these videos and others with segmentations. These challenges include reflections, pupil occlusion, poor lighting conditions and motion blur. Especially those challenges make it difficult to extract clean edges which is the reason for the poor detection rate of the state-of-the-art algorithms. While this evaluation shows how accurate the pupil center can be detected, it does not show how accurately the pupil area and shape was extracted.

The selected videos for our segmentation are 2, 3, 4, 5, 7, 9, 10, 11, 13, 15, 18, 19 and 22 as well as 1, 6, 8 and 12 which represent the images without image-based challenge. Therefore, our annotations are exactly for 101,125 images from the pupils in the wild dataset. Figure 7 shows some examples of the segmentations as well as the images from the dataset. For the annotation process, we used all three GANs. The generator GAN for the extension of our training data as well as our segmentation GAN for the initial annotation. In some images it was difficult to check and correct the annotation without improving the image quality. For this purpose the refinement GAN was used.

These annotated videos now also offer the possibility to use other metrics such as segmentation quality in addition to the detection rate. In the case of the pupil this is especially important for 3D eyeball creation [41] and therefore, to estimate a 3D gaze position as well as to compensate eye tracker drifts in case of head mounted eye trackers. The segmentation results can be seen in Table6. As metric we used

Table 5. Detection rates for the state-of-the-art algorithms per video on the labeled pupils in the wild dataset [42]. The five pixel euclidean distance was used to compensate for inaccurate annotations as proposed in [12, 16]. Each video $[1, 3]$ per subject $[1, 22]$ is evaluated separately for each algorithm. Best result in bold.

| | ElSe | | | Świrski | | | PURE | | |
|---|---|---|---|---|---|---|---|---|---|
| | 1 | 2 | 3 | 1 | 2 | 3 | 1 | 2 | 3 |
| 1 | 0.88 | 0.95 | 0.81 | 0.92 | 0.9 | 0.7 | **0.93** | **0.98** | **0.83** |
| 2 | 0.41 | 0.82 | 0.86 | 0 | 0.37 | 0.89 | **0.56** | **0.86** | **0.93** |
| 3 | 0.21 | 0.58 | 0.94 | 0 | 0.03 | 0.91 | **0.3** | **0.59** | **0.96** |
| 4 | 0.09 | **0.53** | 0.51 | **0.13** | 0.03 | 0.32 | 0.09 | 0.38 | **0.81** |
| 5 | **0.31** | 0.35 | **0.02** | 0.22 | 0.02 | 0.01 | 0.25 | **0.43** | 0.01 |
| 6 | 0.83 | 0.8 | 0.89 | 0.67 | 0.33 | 0.88 | **0.84** | **0.87** | **0.93** |
| 7 | 0.67 | 0.94 | 0.6 | **0.79** | **0.97** | 0.23 | **0.79** | 0.94 | **0.72** |
| 8 | 0.88 | 0.87 | 0.69 | **0.94** | 0.68 | 0.75 | 0.93 | **0.9** | **0.76** |
| 9 | **0.41** | 0.5 | 0.94 | 0.17 | **0.56** | **0.95** | 0.4 | 0.49 | **0.95** |
| 10 | 0.91 | 0.37 | 0.9 | **0.95** | **0.59** | 0.58 | 0.94 | 0.47 | **0.96** |
| 11 | 0.51 | 0.84 | 0.79 | 0.13 | 0.38 | 0.43 | **0.54** | **0.89** | **0.83** |
| 12 | 0.96 | 0.92 | 0.81 | 0.6 | 0.94 | 0.61 | **0.97** | **0.97** | **0.88** |
| 13 | **0.34** | **0.69** | 0.51 | 0.16 | 0.27 | 0.44 | 0.32 | 0.68 | **0.54** |
| 14 | 0.83 | 0.49 | 0.79 | **0.95** | 0.41 | **0.92** | 0.87 | **0.59** | 0.83 |
| 15 | 0.5 | 0.58 | 0.53 | 0.23 | 0.31 | 0.6 | **0.55** | **0.63** | **0.59** |
| 16 | **0.97** | 0.59 | 0.88 | 0.96 | 0.53 | 0.71 | 0.94 | **0.62** | **0.89** |
| 17 | **0.5** | **0.84** | **0.85** | 0.49 | 0.77 | 0.77 | **0.5** | 0.81 | **0.85** |
| 18 | 0.54 | 0.87 | **0.95** | 0.46 | 0.8 | 0.58 | **0.64** | **0.92** | 0.94 |
| 19 | 0.83 | 0.79 | **0** | 0.29 | 0.44 | **0** | **0.93** | **0.84** | **0** |
| 20 | 0.78 | 0.78 | 0.94 | 0.69 | 0 | 0.55 | **0.86** | **0.97** | **0.96** |
| 21 | 0.88 | 0.96 | 0.82 | 0.19 | 0.96 | 0.54 | **0.92** | **0.98** | **0.84** |
| 22 | 0.63 | 0.63 | 0.85 | 0 | 0 | 0.18 | **0.66** | **0.71** | **0.92** |

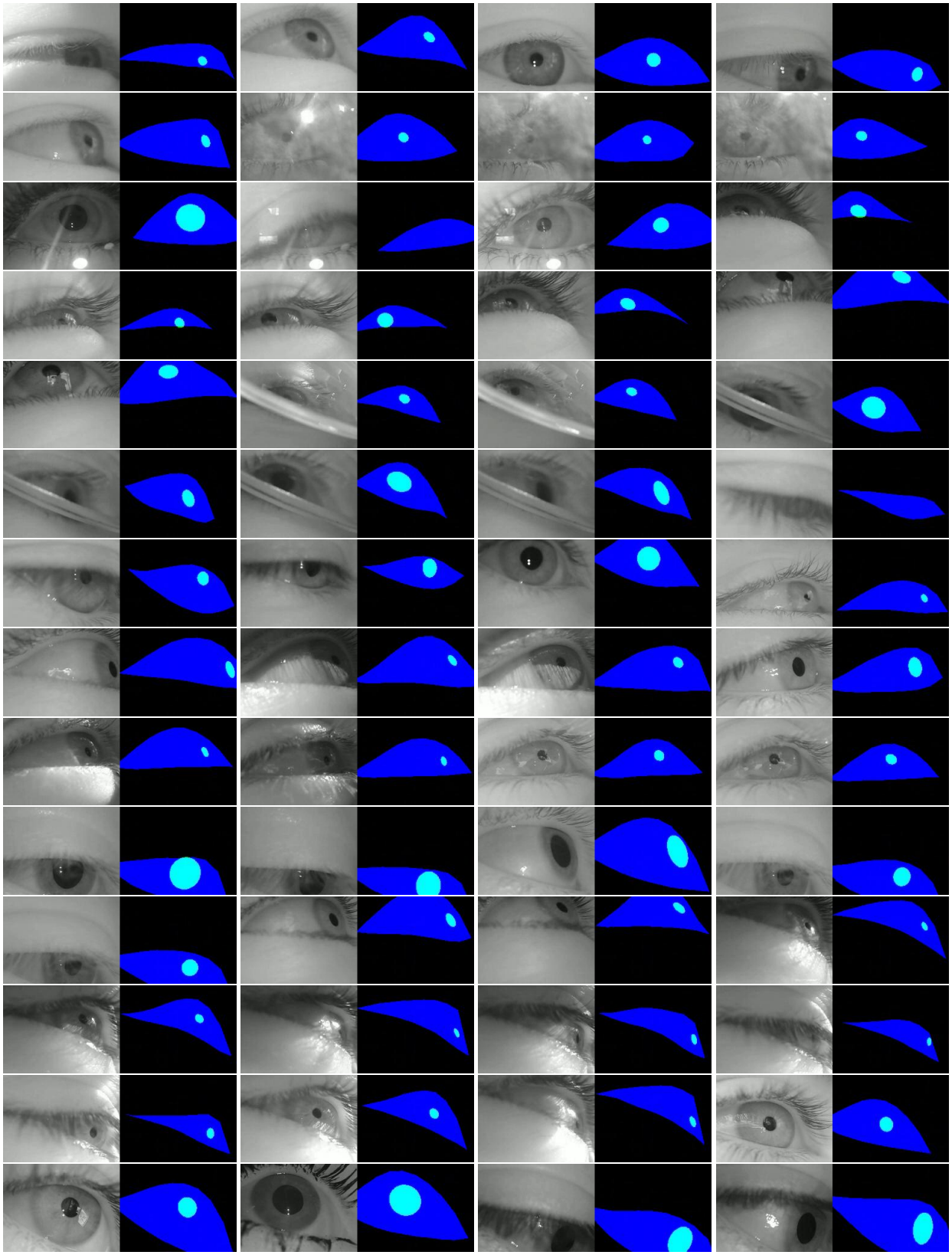

Figure 7. Segmented images from the pupils in the wild dataset [42].

the Jaccard index averaged over all images per video.

Table 6. Segmentation results using the average Jaccard index for the state-of-the-art algorithms per video on the labeled pupils in the wild dataset [42]. Each video $[1, 3]$ per subject is evaluated separately for each algorithm. Best result in bold.

| | ElSe | | | Świrski | | | PURE | | |
|---|---|---|---|---|---|---|---|---|---|
| | 1 | 2 | 3 | 1 | 2 | 3 | 1 | 2 | 3 |
| 1 | **0.92** | **0.93** | **0.92** | 0.91 | 0.91 | 0.83 | 0.9 | 0.91 | 0.9 |
| 2 | 0.32 | 0.83 | **0.89** | 0.01 | 0.51 | 0.78 | **0.56** | **0.85** | **0.89** |
| 3 | 0.21 | **0.92** | **0.9** | 0 | 0.11 | 0.82 | **0.31** | **0.92** | 0.88 |
| 4 | 0.29 | **0.9** | 0.73 | **0.33** | 0.5 | 0.54 | 0.31 | 0.61 | **0.85** |
| 5 | 0.49 | 0.22 | **0.36** | 0.17 | 0.06 | 0.02 | 0.5 | **0.36** | **0.08** |
| 6 | **0.87** | **0.91** | **0.92** | 0.76 | 0.61 | 0.91 | 0.84 | **0.91** | 0.91 |
| 7 | 0.55 | 0.89 | 0.5 | **0.75** | **0.9** | 0.25 | 0.69 | 0.87 | **0.62** |
| 8 | 0.91 | **0.94** | 0.65 | **0.93** | 0.79 | **0.7** | 0.91 | 0.93 | **0.7** |
| 9 | 0.82 | 0.81 | **0.94** | 0.46 | 0.76 | 0.93 | **0.83** | **0.82** | 0.92 |
| 10 | **0.86** | 0.37 | 0.9 | 0.85 | **0.56** | 0.79 | 0.84 | 0.44 | **0.92** |
| 11 | 0.55 | 0.88 | **0.91** | 0.12 | 0.64 | 0.58 | **0.64** | **0.89** | **0.91** |
| 12 | **0.95** | 0.91 | 0.87 | 0.72 | **0.92** | 0.79 | 0.93 | 0.91 | **0.88** |
| 13 | 0.67 | **0.91** | 0.64 | 0.44 | 0.43 | 0.55 | **0.74** | 0.9 | **0.72** |
| 15 | **0.79** | 0.82 | 0.73 | 0.55 | 0.6 | **0.77** | **0.79** | **0.83** | 0.74 |
| 18 | 0.61 | **0.89** | **0.96** | 0.59 | 0.82 | 0.74 | **0.65** | 0.86 | 0.94 |
| 19 | 0.9 | 0.88 | 0.19 | 0.6 | 0.65 | 0.15 | **0.91** | **0.9** | **0.21** |
| 22 | 0.76 | 0.69 | 0.88 | 0 | 0 | 0.18 | **0.78** | **0.76** | **0.91** |

An interesting result is provided by subject 1. By comparing the detection rate of ElSe in Table 5 to the results of PuRe, it can be seen that PuRe has always a higher detection rate. In the case of segmentation accuracy (Table 6), this is not true. Here ElSe is more accurate in comparison to PuRe on all three videos. This is due to the used convolution filters for edge detection. ElSe uses the differential of Gaussian whereas PuRe uses large Sobel filters. The Sobel filters are more robust to noise especially since they are used as 2D convolutions. This effects the edge pixel accuracy and therefore, the resulting ellipse of the fitting procedure. As can be seen in Table 6 from the best results (bold), that they are distributed equally across all algorithms.

In order to highlight this effect more clearly, we have conducted another experiment (Table 7). Here the average segmentation accuracy is shown for all pupil with a pupil center detection with less or equal to 5 pixels euclidean distance. As segmentation metric we used again the Jaccard index. All remaining pupils were ignored. This shows how well a correctly detected pupil based on the center estimation is segmented.

In Table 7 it can be seen that the results have changed entirely which supports our argument that the differential of Gaussian is a more accurate edge filter while less robust as separated 1D filters (Table 5). This is especially true since one of the main difference between ElSe and PuRe is the used filter for edge extraction. The inaccurate segmentation results from ElSe steam from iris edges which are selected as best ellipses. A disadvantage of this evaluation for PuRe could be the higher detection rate. However, in most cases where all algorithms are similarly good, ElSe still performs best in segmentation. This evaluation is only one of many possibilities and should show how it can help the algorithm developer to evaluate algorithms in different ways.

Table 7. Segmentation results using the average Jaccard index for images where the pupil center was detected with less or equal 5 pixels euclidean distance. The images are from the labeled pupils in the wild dataset [42] with the provided segmentations. Each video $[1, 3]$ per subject is evaluated separately for each algorithm. Best result in bold.

| | ElSe | | | Świrski | | | PURE | | |
|---|---|---|---|---|---|---|---|---|---|
| | 1 | 2 | 3 | 1 | 2 | 3 | 1 | 2 | 3 |
| 1 | **0.95** | **0.94** | **0.94** | 0.94 | 0.93 | 0.93 | 0.93 | 0.92 | 0.91 |
| 2 | 0.62 | **0.91** | **0.93** | 0 | 0.85 | 0.92 | **0.82** | 0.88 | 0.92 |
| 3 | 0.75 | **0.96** | **0.93** | 0 | 0.29 | 0.91 | **0.8** | 0.94 | 0.9 |
| 4 | **0.85** | **0.97** | **0.93** | 0.74 | 0.88 | 0.88 | 0.84 | 0.96 | 0.92 |
| 5 | **0.9** | **0.83** | **0.58** | 0.41 | 0.34 | 0.14 | 0.88 | 0.78 | 0.3 |
| 6 | **0.92** | **0.95** | **0.95** | 0.85 | 0.91 | 0.92 | 0.89 | 0.93 | 0.93 |
| 7 | 0.8 | **0.93** | **0.85** | **0.85** | 0.91 | 0.82 | 0.83 | 0.9 | 0.83 |
| 8 | **0.94** | **0.97** | 0.85 | **0.94** | 0.95 | **0.86** | 0.92 | 0.95 | 0.83 |
| 9 | **0.96** | **0.92** | **0.95** | 0.93 | 0.91 | **0.95** | 0.95 | 0.9 | 0.93 |
| 10 | **0.92** | 0.79 | **0.95** | 0.87 | 0.78 | 0.9 | 0.9 | **0.81** | 0.94 |
| 11 | **0.87** | **0.93** | **0.97** | **0.87** | 0.9 | 0.93 | 0.84 | 0.92 | 0.95 |
| 12 | **0.96** | **0.93** | **0.92** | 0.93 | 0.92 | 0.89 | 0.95 | 0.91 | 0.89 |
| 13 | **0.93** | **0.97** | 0.87 | **0.93** | 0.93 | **0.88** | 0.9 | 0.95 | 0.86 |
| 15 | **0.93** | **0.93** | **0.93** | 0.89 | **0.93** | 0.9 | 0.92 | **0.93** | 0.91 |
| 18 | **0.88** | **0.92** | **0.97** | 0.86 | 0.91 | 0.94 | 0.85 | 0.89 | 0.95 |
| 19 | **0.95** | **0.93** | 0.57 | 0.89 | 0.9 | 0.81 | 0.93 | 0.92 | **0.84** |
| 22 | **0.93** | **0.91** | **0.96** | 0 | 0 | 0.92 | 0.91 | 0.9 | 0.95 |

## 6. Conclusion

We have demonstrated the applicability of GANs for pupil and eyelid segmentation, data generation and data refinement. In all our experiments we were able to achieve state-of-the-art results as well as improving the results of state-of-the-art algorithms with data refinement. The runtime of our models is significantly higher compared to state-of-the-art algorithms and requires a modern GPU. However, our models can be used for off-line data preparation, which can be used for training smaller models or other machine learning methods such as random forest. In addition, the off-line data processing can improve the data quality of scientific experiments and eye tracking user studies for market research. Future work will go into the direction of dataset generation for eye tracking in the area of augmented and virtual reality. This will allow evaluations especially for these areas and support training of machine learning approaches.

## Acknowledgements

Work of the authors is supported by the Institutional Strategy of the University of Tübingen (Deutsche Forschungsgemeinschaft, ZUK 63). This research was supported by an IBM Shared University Research Grant including an IBM PowerAI environment. We especially thank our partners Benedikt Rombach, Martin Mähler and Hildegard Gerhardy from IBM for their expertise and support.

## References


[1] M. Adam, F. Rossant, F. Amiel, B. Mikovikova, and T. Ea. Eyelid localization for iris identification. *Radioengineering*, 17(4):82–85, 2008.

[2] V. Badrinarayanan, A. Kendall, and R. Cipolla. Segnet: A deep convolutional encoder-decoder architecture for image segmentation. *arXiv preprint arXiv:1511.00561*, 2015.

[3] C. Braunagel, W. Rosenstiel, and E. Kasneci. Ready for take-over? a new driver assistance system for an automated classification of driver take-over readiness. *IEEE Intelligent Transportation Systems Magazine*, 9(4):10–22, 2017.

[4] J. Daugman. How iris recognition works. In *The essential guide to image processing*, pages 715–739. Elsevier, 2009.

[5] A. T. Duchowski, V. Shivashankaraiah, T. Rawls, A. K. Gramopadhye, B. J. Melloy, and B. Kanki. Binocular eye tracking in virtual reality for inspection training. In *Proceedings of the 2000 symposium on Eye tracking research & applications*, pages 89–96. ACM, 2000.

[6] S. Eivazi, R. Bednarik, V. Leinonen, M. von und zu Fraunberg, and J. E. Jääskeläinen. Embedding an eye tracker into a surgical microscope: requirements, design, and implementation. *IEEE Sensors Journal*, 16(7):2070–2078, 2016.

[7] S. Eivazi, T. Santini, A. Keshavarzi, T. Kübler, and A. Mazzei. Improving real-time cnn-based pupil detection through domain-specific data augmentation. In *Proceedings of the 11th ACM Symposium on Eye Tracking Research & Applications*, page 40. ACM, 2019.

[8] W. Fuhl, N. Castner, L. Zhuang, M. Holzer, W. Rosenstiel, and E. Kasneci. Mam: Transfer learning for fully automatic video annotation and specialized detector creation. In *Egocentric Perception, Interaction and Computing Workshop (EPIC@ECCV)*, 09 2018.

[9] W. Fuhl, S. Eivazi, B. Hosp, A. Eivazi, W. Rosenstiel, and E. Kasneci. Bore: boosted-oriented edge optimization for robust, real time remote pupil center detection. In *Proceedings of the 2018 ACM Symposium on Eye Tracking Research & Applications*, page 48. ACM, 2018.

[10] W. Fuhl, D. Geisler, T. Santini, T. Appel, W. Rosenstiel, and E. Kasneci. Cbf: circular binary features for robust and real-time pupil center detection. In *Proceedings of the 2018 ACM Symposium on Eye Tracking Research & Applications*, page 8. ACM, 2018.

[11] W. Fuhl, D. Geisler, T. Santini, W. Rosenstiel, and E. Kasneci. Evaluation of state-of-the-art pupil detection algorithms on remote eye images. In *International Joint Conference on Pervasive and Ubiquitous Computing*, pages 1716–1725. ACM, 2016.

[12] W. Fuhl, T. Kübler, K. Sippel, W. Rosenstiel, and E. Kasneci. Excuse: Robust pupil detection in real-world scenarios. In *International Conference on Computer Analysis of Images and Patterns*, pages 39–51. Springer, 2015.

[13] W. Fuhl, T. Santini, D. Geisler, T. Kübler, W. Rosenstiel, and E. Kasneci. Eyes wide open? eyelid location and eye aperture estimation for pervasive eye tracking in real-world scenarios. In *PETMEI*, 09 2016.

[14] W. Fuhl, T. Santini, and E. Kasneci. Fast and robust eyelid outline and aperture detection in real-world scenarios. In *Winter Conference on Applications of Computer Vision*, pages 1089–1097. IEEE, 2017.

[15] W. Fuhl, T. Santini, G. Kasneci, and E. Kasneci. Pupilnet: convolutional neural networks for robust pupil detection. *arXiv preprint arXiv:1601.04902*, 2016.

[16] W. Fuhl, T. C. Santini, T. Kübler, and E. Kasneci. Else: Ellipse selection for robust pupil detection in real-world environments. In *Proceedings of the Ninth Biennial ACM Symposium on Eye Tracking Research & Applications*, pages 123–130. ACM, 2016.

[17] W. Fuhl, M. Tonsen, A. Bulling, and E. Kasneci. Pupil detection for head-mounted eye tracking in the wild: an evaluation of the state of the art. *Machine Vision and Applications*, 27(8):1275–1288, 2016.

[18] I. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. Courville, and Y. Bengio. Generative adversarial nets. In *Advances in neural information processing systems*, pages 2672–2680, 2014.

[19] B. Guenter, M. Finch, S. Drucker, D. Tan, and J. Snyder. Foveated 3d graphics. *ACM Transactions on Graphics (TOG)*, 31(6):164, 2012.

[20] A. Haro, M. Flickner, and I. Essa. Detecting and tracking eyes by using their physiological properties, dynamics, and appearance. In *Computer Vision and Pattern Recognition*, volume 1, pages 163–168. IEEE, 2000.

[21] K. He, G. Gkioxari, P. Dollár, and R. Girshick. Mask r-cnn. In *International Conference on Computer Vision*, pages 2980–2988. IEEE, 2017.

[22] S. Ishimaru, K. Kunze, K. Kise, J. Weppner, A. Dengel, P. Lukowicz, and A. Bulling. In the blink of an eye: combining head motion and eye blink frequency for activity recognition with google glass. In *Proceedings of the 5th augmented human international conference*, page 15. ACM, 2014.

[23] A.-H. Javadi, Z. Hakimi, M. Barati, V. Walsh, and L. Tcheang. Set: a pupil detection method using sinusoidal approximation. *Frontiers in neuroengineering*, 8:4, 2015.

[24] Y. Jia, E. Shelhamer, J. Donahue, S. Karayev, J. Long, R. Girshick, S. Guadarrama, and T. Darrell. Caffe: Convolutional architecture for fast feature embedding. In *International Conference on Multimedia*, pages 675–678. ACM, 2014.

[25] M. Kassner, W. Patera, and A. Bulling. Pupil: an open source platform for pervasive eye tracking and mobile gaze-based interaction. In *International Joint Conference on Pervasive and Ubiquitous Computing*, pages 1151–1160. ACM, 2014.

[26] V. Kazemi and J. Sullivan. One millisecond face alignment with an ensemble of regression trees. In *Computer Vision and Pattern Recognition*, pages 1867–1874, 2014.

[27] D. E. King. Dlib-ml: A machine learning toolkit. *Journal of Machine Learning Research*, 10(Jul):1755–1758, 2009.

[28] D. P. Kingma and J. Ba. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014.

[29] Y. Lee, R. J. Micheals, J. J. Filliben, and P. J. Phillips. Vasir: an open-source research platform for advanced iris recognition technologies. *Journal of research of the National Institute of Standards and Technology*, 118:218, 2013.

[30] X. Liu, F. Xu, and K. Fujimura. Real-time eye detection and tracking for driver observation under various light con-

ditions. In *Intelligent Vehicle Symposium, 2002. IEEE*, volume 2, pages 344–351. IEEE, 2002.

[31] J. Long, E. Shelhamer, and T. Darrell. Fully convolutional networks for semantic segmentation. In *Computer Vision and Pattern Recognition*, pages 3431–3440, 2015.

[32] X. Mao, Q. Li, H. Xie, R. Y. Lau, and Z. Wang. Multi-class generative adversarial networks with the l2 loss function. *arXiv preprint arXiv:1611.04076*, 5, 2016.

[33] I. T. Oltean, J. K. Shimmick, and T. N. Clapham. Eye tracking device for laser eye surgery using corneal margin detection, Oct. 9 2001. US Patent 6,299,307.

[34] A. Patney, M. Salvi, J. Kim, A. Kaplanyan, C. Wyman, N. Benty, D. Luebke, and A. Lefohn. Towards foveated rendering for gaze-tracked virtual reality. *ACM Transactions on Graphics (TOG)*, 35(6):179, 2016.

[35] T. Pfeiffer and P. Renner. Eyesee3d: a low-cost approach for analysing mobile 3d eye tracking data using augmented reality technology. In *Proceedings of the Symposium on Eye Tracking Research and Applications*, 2014.

[36] S. Ren, X. Cao, Y. Wei, and J. Sun. Face alignment at 3000 fps via regressing local binary features. In *Computer Vision and Pattern Recognition*, pages 1685–1692, 2014.

[37] T. Santini, W. Fuhl, and E. Kasneci. Pure: Robust pupil detection for real-time pervasive eye tracking. *Computer Vision and Image Understanding*, 2018.

[38] A. Shrivastava, T. Pfister, O. Tuzel, J. Susskind, W. Wang, and R. Webb. Learning from simulated and unsupervised images through adversarial training. In *Computer Vision and Pattern Recognition*, pages 2107–2116, 2017.

[39] M. Suzuki, N. Yamamoto, O. Yamamoto, T. Nakano, and S. Yamamoto. Measurement of driver's consciousness by image processing-a method for presuming driver's drowsiness by eye-blinks coping with individual differences. In *SMC*, volume 4, pages 2891–2896. IEEE, 2006.

[40] L. Świrski, A. Bulling, and N. Dodgson. Robust real-time pupil tracking in highly off-axis images. In *Proceedings of the Symposium on Eye Tracking Research and Applications*, pages 173–176. ACM, 2012.

[41] L. Świrski and N. A. Dodgson. A fully-automatic, temporal approach to single camera, glint-free 3d eye model fitting [abstract]. In *Proceedings of ECEM 2013*, Aug. 2013.

[42] M. Tonsen, X. Zhang, Y. Sugano, and A. Bulling. Labelled pupils in the wild: a dataset for studying pupil detection in unconstrained environments. In *Proceedings of the Ninth Biennial ACM Symposium on Eye Tracking Research & Applications*, pages 139–142. ACM, 2016.

[43] F. Vera-Olmos and N. Malpica. Deconvolutional neural network for pupil detection in real-world environments. In *International Work-Conference on the Interplay Between Natural and Artificial Computation*, pages 223–231. Springer, 2017.

[44] F. Vera-Olmos, E. Pardo, H. Melero, and N. Malpica. Deepeye: Deep convolutional network for pupil detection in real environments. *Integrated Computer-Aided Engineering*, 26(1):85–95, 2019.

[45] R. Wan, B. Shi, L.-Y. Duan, A.-H. Tan, and A. C. Kot. Benchmarking single-image reflection removal algorithms. In *Proc. ICCV*, 2017.

[46] R. P. Wildes. Iris recognition: an emerging biometric technology. *Proceedings of the IEEE*, 85(9):1348–1363, 1997.

[47] E. Wood, T. Baltrušaitis, L.-P. Morency, P. Robinson, and A. Bulling. Learning an appearance-based gaze estimator from one million synthesised images. In *Proceedings of the Ninth Biennial ACM Symposium on Eye Tracking Research & Applications*, pages 131–138. ACM, 2016.

[48] E. Wood, T. Baltrusaitis, X. Zhang, Y. Sugano, P. Robinson, and A. Bulling. Rendering of eyes for eye-shape registration and gaze estimation. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 3756–3764, 2015.

[49] F. Yang, X. Yu, J. Huang, P. Yang, and D. Metaxas. Robust eyelid tracking for fatigue detection. In *ICIP*, pages 1829–1832, Sept 2012.

[50] Y.-H. Yiu, M. Aboulatta, T. Raiser, L. Ophey, V. L. Flanagin, P. zu Eulenburg, and S.-A. Ahmadi. Deepvog: Open-source pupil segmentation and gaze estimation in neuroscience using deep learning. *Journal of neuroscience methods*, 2019.

[51] X. Zhang, Y. Sugano, M. Fritz, and A. Bulling. Appearance-based gaze estimation in the wild. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 4511–4520, 2015.

[52] J.-Y. Zhu, T. Park, P. Isola, and A. A. Efros. Unpaired image-to-image translation using cycle-consistent adversarial networks. *arXiv preprint*, 2017.