Available online at www.ijournalse.org

Emerging Science Journal

(ISSN: 2610-9182)

Vol. 6, No. 5, October, 2022

A YOLO Detector Providing Fast and Accurate Pupil Center Estimation using Regions Surrounding a Pupil

Wattanapong Kurdthongmee [1*], Piyadhida Kurdthongmee [2], Korrakot Suwannarat [1], Jeremy K. Kiplagat [3]

[1] *School of Engineering and Technology, Walailak University 222 Thaibury, Thasala, Nakornsithammarat 80160 Thailand.*

[2] *Center for Scientific and Technological Equipment, Walailak University 222 Thaibury, Thasala, Nakornsithammarat 80160 Thailand.*

[3] *International College, Walailak University 222 Thaibury, Thasala, Nakornsithammarat 80160 Thailand.*

Abstract

Eye-tracking technology has many useful applications, including Virtual Reality (VR) devices, Augmented Reality (AR) devices, and assistive technology. The main objective of eye-tracking technology is to detect eye position and track eye movements. It is possible to determine the eye position when the pupil center is detected. In this paper, a deep learning-based approach to the detection of pupil centers from webcam images is presented. As opposed to all previous approaches to object detection based on training the detector with objects to be detected, our object detector was trained with both the region surrounding a pupil and the region between an eye and the region surrounding a pupil. The latter set of regions has been found to increase the overall detection accuracy. A novel post-processing algorithm is also presented to estimate the pupil center from all the detected regions. To achieve real-time performance, we have adopted the tiny architecture of YOLOv3, which has 23 layers and can be executed without requiring a GPU accelerator. To train the detectors, different variations of regions covering a pupil and an eye were used, as well as expanding regions surrounding a pupil and an eye. The PUPPIE dataset was used as the primary input for training the detector. The setting with the best detection accuracy was applied to all publicly available datasets: I2Head, MPIIGaze, and U2Eyes. In terms of accuracy, the results indicate that pupil center estimation is comparable to the state-of-the-art approach. It achieves pupil center estimation errors below the size of a constricted pupil in more than 98.24% of images. Furthermore, the detection time is 2.8 times faster than the state-of-the-art approach.

Keywords:

Eye Tracking;

Pupil Center Detection;

Convolutional Neural Networks;

You-Only-Look-Once.

Article History

Received:	11	March	2022
Revised:	12	June	2022
Accepted:	01	July	2022
Available online:	31	July	2022

1- Introduction

As a critical component of Virtual Reality (VR), Augmented Reality (AR) devices, and assistive technology, eye-tracking requires high accuracy and real-time performance. The purpose of eye-tracking technology is to locate and track eye movements accurately. The pupil center estimation is the starting point to providing the information to locate an eye. Two main types of pupil center estimation can be distinguished based on two different systems: a head-mounted system and a head-free or remote system. The gaze direction is measured relative to the head when a head-mounted system is used. A three-dimensional head pose (position and orientation) must be estimated to calculate the pupil center in space. Various types of transducers can be used to measure head position, of which the magnetic position transducer is the most commonly used. Another approach involves using a head-mounted camera that records scenes in front of the subject. Visual cues are then extracted from images captured by the scene camera to determine the head pose relative to the observed scene [1].

* **CONTACT**: kurdthongmee.wattanapong@gmail.com

DOI: http://dx.doi.org/10.28991/ESJ-2022-06-05-05

Scleral coils or Electro-Oculography (EOG) [2] and Infrared Oculography (IRO) are also used for eye tracking. Both devices are suitable for controlled environments where the user uses head-mounted devices or whose movements are limited, resulting in highly accurate systems [3]. In a head-free or remote system, the pupil center estimation can be performed by Video-Oculography (VOG) in a less invasive manner. A Convolution Neural Network (CNN) has been proven to provide accurate pupil center estimation for such systems effectively. It has been demonstrated that CNN can correctly operate in challenging images with defects associated with poor illumination, reflections, or pupil occlusion [4, 5]. In recent years, less invasive technologies have been proposed that do not require body attachment. A popular method that utilizes a webcam to provide the image for CNN to perform pupil center estimation is presented by Choi et al. [6]. An inherent limitation of CNN-based approaches is that they require a large dataset representing the variability of the problem to adapt to the solution and generalize. Additionally, a framework is required to learn and deploy the model effectively.

Herein, we review the related work in pupil center estimation, focusing specifically on neural networks with several layers, Deep Neural Networks (DNN). In their study, Xia et al. [7] applied Fully Convolutional Networks (FCN) to segment pupil regions. Using a shallow structure with a large kernel convolutional block, they could transfer their performance from semantic segmentation to the localization of the eye centers by carefully selecting hyperparameters. A heterogeneous CNN model was proposed by Choi et al. [6] for the problem of pupil center estimation. In their approach, faces and landmarks on a face in an image were detected. These landmarks were then used to determine the eye region. When glasses were present in an image, they were removed by a Generative Adversarial Network (GAN) to locate the proper eye region. Next, pupil regions were segmented using an FCN. Finally, the pupil center estimation was performed. The proposed approach outperformed the previously proposed approaches on public datasets, such as BioID and GI4E. A robust alternative to wearing glasses was proposed by Lee et al. [8]. It utilized appearance-based pupil center estimation inspired by the work of Choi et al. [6]. The perceptual loss was also employed to mitigate the blur phenomenon resulting from glass removal. Mutual information maximization was additionally integrated to enhance the representation quality of the segmentation network. It was claimed that the latency of the proposed approach was significantly reduced by employing the concept of non-local block and self-attention block with a large receptive field. Kitazumi and Nakazawa [9] proposed a CNN-based approach for pupil segmentation and pupil center estimation. The dlib [10] was used to detect the eye region. A five-layer architecture called U-Net [11] was exploited to perform pupil segmentation. Zdarsky et al. [12] presented an approach for gaze estimation. Landmarks on a face were estimated using DeepLabCut [13], an open-source toolkit for estimating the pose of body parts based on deep learning. The algorithm uses a subset of the DeeperCut feature detectors [14] which connects a pre-trained ResNet-50 network with deconvolution layers to sample the visual information and produce spatial probability densities. Various layers of deconvolution are applied to different parts of the body. Probability density represents "evidence" that this specific body part is located in a particular area [15]. However, Zdarsky et al. [12] failed to provide information regarding the accuracy of the estimated landmarks.

There are also several publications on real-time pupil center estimation on a low-performance platform, i.e., on a platform without GPU acceleration or with low performance. Kim et al. [16] presented the approach which was based on using a cascade deep regression forest instead of DNN. It provided an accurate real-time pupil center estimation running on the CPU platform. Cai et al. [17] also proposed an approach for pupil center estimation with a low computational cost based on hierarchical adaptive convolution. Convolution of the eye image was performed using different hierarchical kernels. A kernel was selected for each image based on the user's 3D head pose as part of the localization process. Lee et al. [18] presented the network that uses mutual information during the training stage to increase the accuracy of pupil center estimation. A general-purpose computer platform that operated the proposed approach achieved a real-time performance of 52 frames per second. At the same time, precisions of 96.71 percent, 99.84 percent, and 96.38 percent were obtained for the BioID, GI4E, and Talking Face Video datasets, respectively.

Blousseau et al. [19] introduced a new center-of-mass output layer to their CNN to increase the accuracy of pupil center estimation. The estimation accuracy was 400% greater than the traditional approach. Unfortunately, only eight subjects were used for benchmarking. Ou et al. [20], and Poulopoulos et al. [21] proposed an approach to avoid using large datasets for training the network to perform pupil center estimation. The approach performed image-to-heatmap translation and trains them on an adversarial training framework. It was reported to achieve an accuracy of 96.86% (for normalized error less than 0.05) only on the BioID dataset with the maximum processing time per image of 34 ms. In order to meet two critical requirements of low complexity and algorithm performance, Kang and Chang [22] proposed a pupil-tracking approach applicable to both drivers with and without sunglasses. To generate bare faces, a regression-algorithm-based approach was employed that utilizes scale-invariant feature transforms. A supervised regression-based pupil center estimation approach was additionally proposed to estimate pupil centers in eyes covered by sunglasses. The approach achieved high accuracy and speed, with a precision error of less than 10 mm and a 5 ms processing time on a limited proprietary test dataset. Larumbe-Bergera et al. [23] provided the research community with an improved version of the facial landmark dataset. They also proposed a novel pupil center estimation approach that extends the Resnet-50 backbone by incorporating a pooling layer and four additional fully connected layers to detect eye landmarks used to estimate a pupil. The approach was validated using both realistic and synthetic datasets. It achieved an estimation error below the size of an unconstricted pupil in more than 95 percent of the images. It was claimed to be 3 to 8 times faster than current state-of-the-art approaches. The preliminary result of real-time pupil center estimation to the problem of Chinese reading tests was presented by Lin et al. [24]. The detection accuracy of eye movement events was as high as 97% on a limited-size test dataset.

Upon reviewing this literature, it becomes apparent that almost all previously proposed approaches share the following limitations:

- Using a limited-size dataset for creating DNN models for pupil center estimation
- Increasing the number of layers of processing on DNN
- Testing/validating the model on a small and proprietary dataset
- Having solutions optimized for high-performance platforms, such as those with GPU acceleration

It has long been perceived that the best DNN detector, or the detector from now on, requires training on a large dataset, which is not practical. This paper presents a novel approach to creating the pupil detector that is, in turn, used for pupil center estimation from a limited-size dataset. We propose that training the detector can be accomplished not only by using the region covering the object to be detected, the pupil in this context, but also by using the surrounding regions. The reason for this is that a region surrounding the object contains a limited set of features and a limited variability that DNN can effectively extract and learn. Accordingly, the detector is limited in its applicability. Fortunately, additional features can be enhanced from the surrounding regions. A noteworthy contribution of this paper is that it proposes a novel approach to creating a pupil detector by training it on both the region surrounding the pupil and the region between an eye and the region surrounding a pupil. Additionally, a lightweight post-processing algorithm is presented that estimates pupil centers by using all the detected regions and their geometric relationships. Using a shallow layer DNN detection framework, we demonstrate that our approach generates an effective pupil detector that is comparable to the current state-of-the-art one in terms of detection accuracy and error.

This paper is organized as follows. Materials and methods are presented in Section 2. The experimental results and discussion follow in Section 3. Finally, the paper is concluded in Section.

2- Materials and Methods

2-1- Dataset

2-1-1- The Datasets of Training the Detector

In the course of this research, we used the pupil annotations from the PUPPIE dataset and the facial point annotations from the datasets referred to by PUPPIE. In total, there are 1,791 images in the PUPPIE dataset. The following describes the steps to prepare the training dataset. On each image, the pupil annotations are retrieved first for the left and right eyes. After this, the point annotation files associated with the image in the original datasets are visited. Only the relevant members of the point annotations between 37 and 42 and 43 and 48 are extracted. These points correspond to the point annotations surrounding the left and right eyes, respectively. A Python script that we developed is used to automatically create the bounding boxes for these two sets of point annotations. The bounding boxes for the left and right eyes, respectively, are designated as K_L and K_R. As a pupil annotation is defined as a 2D point, in contrast to a bounding box surrounding the pupil, our Python script has the additional responsibility of creating the bounding box around each pupil annotation.

The bounding box is defined as the area whose center is at the annotation point of a pupil. Rather than restricting the bounding box size, it can be adjusted; to be detailed later, to cover a more prominent region around a pupil. The left and right pupil bounding boxes are designated as P_L and P_R, respectively. The K_L and K_R bounding boxes and P_L and P_R bounding boxes are used to automatically create a set of bounding boxes surrounding a pupil BB. Considering the left eye, the BB_L consists of all boxes within the K_L, but outside the P_L. By using these approaches to build and annotate BB, a Python script was developed by our team member to define BB with different classes, which include UL, UM, UR, ML, MR, LL, LM, and LR. For these classes, the left alphabets indicate the vertical locations of the bounding box relative to the bounding box surrounding a pupil, i.e., U, M, and L indicate upper, middle, and lower, respectively. Similarly, the right alphabets indicate the horizontal positions of the bounding box relative to the bounding box surrounding the pupil, i.e., L, M, and R indicate left, middle, and right. As an example, a bounding box whose class is UM corresponds to the upper middle region of a bounding box surrounding a pupil. Additionally, two additional parameters are defined to control the expansion of the bounding boxes surrounding a pupil and an eye in order to provide flexibility when creating the appropriate training datasets. These parameters are designated α and β, respectively. Increasing the α parameter expands the surrounding area of a pupil P. Similarly, increasing the β parameter expands the bounding boxes surrounding an eye *K*. There is only one restriction on the choice of β: it must not make *P* larger than *K*. Otherwise, the complete set of *BB* cannot be constructed. According to preliminary results on the PUPPIE dataset, the α parameter can be chosen between 1 and 5. The α = 1 indicates that *P* is equal to the greater between the width (*W*) and height (*H*) of the input image divided by 200; i.e.:

$$BB_P = \frac{max(W,H)}{200} \tag{1}$$

The size of *K* is controlled by adjusting its β parameter in the range of 1.0000 to 1.8175 with an increment of 0.0625. That is to say, the maximum size of *K* is 1.8175/2 expanded from its original size.

By extending both *P* and *K*'s surrounding areas, a variety of features are created for the pupil and its immediate surroundings. Figure 1 exhibits some sample images along with a set of non-overlapping bounding boxes *BB* where α equals 1 in (a) and (c) and α equals 3 in (b) and (d). In addition, the figure presents two different sizes of *BB*: (a) and (c) the original size and (b) and (d) with the β of 1.125. In the figure, small red circles indicate the members of point annotations around the left and right eyes. The blue and white rectangles represent *P* and *K*, respectively. *BBs* are automatically generated by our Python script and displayed in green rectangles. Increasing the α and β parameters also produces a variety of features in a pupil and out of the pupil, but within an eye.

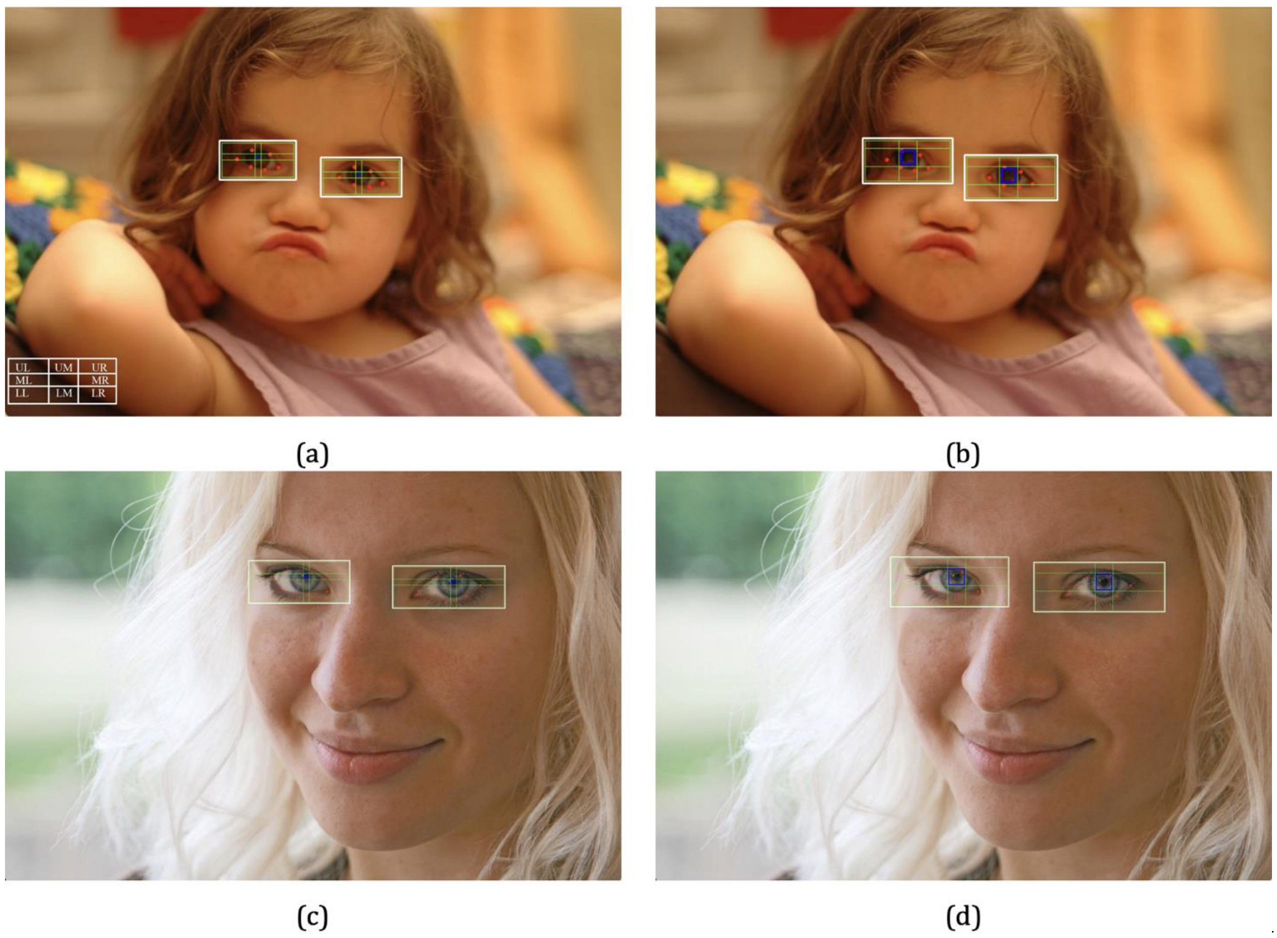

Figure 1. Sample images along with the definitions of a set of non-overlapping bounding boxes with two sizes of P: α = 1 in (a) and (c) and α = 3 in (b) and (d), and two sizes of K: original size in (a) and (c) and the β of 1.125 (b) and (d). These images are part of the 300W dataset, which is freely available for non-commercial use.

Following the steps previously described to prepare the training dataset, several datasets were produced by varying the α and β parameters as indicated. Each dataset was used to train the detector in a ratio of 80:20 between the training and testing datasets. In terms of the number of images, this is equivalent to 1433:358.

2-1-2- The Validation Datasets

To compare our approach to the state-of-the-art one proposed by Larumbe-Bergera et al. [23], all the datasets used in their publication were also used in our experiments. Briefly, these datasets include:

- ***GI4E:*** The GI4E dataset consists of 103 users looking at 12 different points on a screen under standard laboratory conditions. The dataset includes annotations for the pupil center and eye corner. It is considered to be the best dataset and has accurate annotations
- ***I2Head:*** The I2Head dataset consists of a ground-truth dataset for head pose, gaze, and a simplified head model for 12 individuals. The annotations of this dataset have been improved by Larumbe-Bergera et al. [25].
- ***MPIIGaze:*** The MPIIGaze dataset comprises 213,659 images collected from 15 participants during natural everyday laptop use over a period of more than three months. There are some images which include the entire face, while others only include a cropped version of the eye region. Annotations in the dataset include eye corners, pupil centers, and specific facial landmarks.
- ***U2Eyes:*** The U2Eyes dataset consists of binocular images that reproduce real gaze tracking scenarios. The publicly accessible version of the dataset consists of images from 20 users. A user examines two grids of 15 and 32 points, respectively, with 125 different head poses, resulting in 5,875 images per user. The annotated data includes head pose information, gaze direction information, and 2D/3D landmarks.

Table 1 summarizes the number of images in these datasets and those we used during our validation process.

Table 1. A summary of the total number of images within the datasets and the images used during our validation process

Dataset	Total size	Validation size
GI4E	1,236	1,236
I2Head	2,784	2,784
MPIIGaze-subset	10,848	585
U2Eyes	117,500	117,500

2-2- Method

2-2-1- The Detector

The YOLO version 3 (YOLOv3) [26] framework was used in this experiment to train a detector with nine classes of *P* and *BB*. Within the framework, we did not intend to modify or increase the number of layers and additional image enhancements. To investigate the effectiveness of the detectors, we have employed all default configurations except for the number of classes, *n*, which is 9 (*UL, UM, UR, ML, MR, LL, LM, LR,* and *P* (for the pupil class)) in all experiments, dependent parameters, i.e.:

- The number of iterations for training the detector, which is defined by: $n \times 2{,}000$, and;
- The number of filters in the layers preceding the YOLO-layer, which is determined by: $(n + 5) \times 3$.

It was decided to use the tiny configuration of YOLOv3 with 23 layers (tinyYOLOv3). Because of its shallow network architecture, this configuration has demonstrated real-time detection performance even without GPU acceleration on the computer platform [26]. It should be noted that the state-of-the-art approach presented by Larumbe-Bergera et al. [23] makes use of the Resnet-50 backbone with the integration of five additional layers. Layers are more than double the number of YOLOv3 in the tiny configuration. Our training method was based on transfer learning. As a starting point, darknet53.conv.74 was provided to the detector. In order to train the detector with all datasets, the open-source darknet was utilized. We used the Google Colab platform with GPU-assisted acceleration to speed up the training stage. The training stage of each detector took approximately four hours on average. A loss function was recorded for all detectors during the training stage. A model checkpoint was saved every 200 iterations for the first 2,000 iterations and every 1,000 iterations thereafter.

Figure 2 illustrates the steps involved in producing the training dataset from the PUPPIE dataset and training the detector with respect to the α and β parameters.

After each detector was trained, it was validated against the validation datasets described in Sec. 2-1-2. Our research team developed a Python program that utilizes the detection results and a post-processing algorithm, which will be described in Sec. 2-2-2, in order to estimate a pupil center. For the implementation of the detector and the post-processing algorithm, we utilized the OpenCV library [27]. The additional parameters passed to the OpenCV library functions that facilitate the task of object detection, the *blobFromImage*, and *NMSBoxes* functions, are as follows: the scale factor for the *blobFromImage* function is 1/512, and the *confThreshold* and *nmsThreshold* values are 0.625 and 0.875 respectively. Lastly, the detection confidence, which is used to remove non-relevant bounding boxes, was set to 0.25.

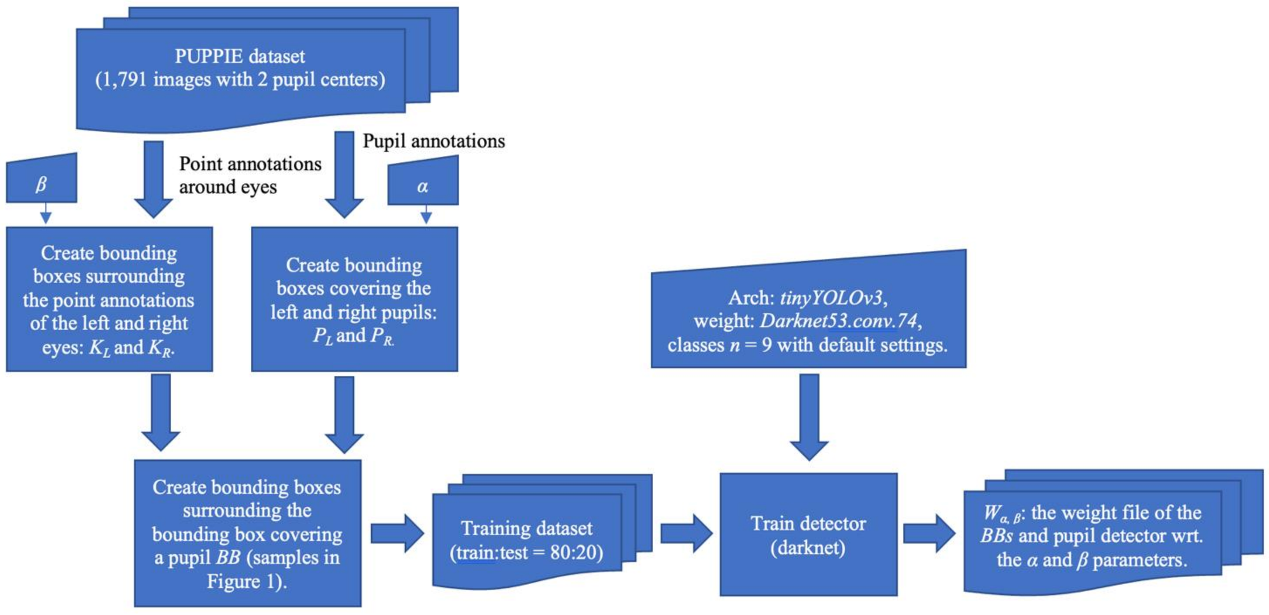
PUPPIE dataset (1,791 images with 2 pupil centers)
Point annotations around eyes
Pupil annotations
β
α
Create bounding boxes surrounding the point annotations of the left and right eyes: K_L and K_R.
Create bounding boxes covering the left and right pupils: P_L and P_R.
Arch: tinyYOLOv3, weight: Darknet53.conv.74, classes $n = 9$ with default settings.
Create bounding boxes surrounding the bounding box covering a pupil BB (samples in Figure 1).
Training dataset (train:test = 80:20)
Train detector (darknet)
$W_{\alpha,\beta}$: the weight file of the BBs and pupil detector wrt. the α and β parameters.

Figure 2. **Summary of the details of the dataset preparations for training the detector**

2-2-2- The Post-Processing Algorithm

The results of the detector for a single eye are expected to consist of a single *P* and a set of *BB* as shown in Figure 1. For best-case scenarios, all nine bounding boxes are expected to be detectable. These detected bounding boxes fall into the following classes: *UL*, *UM*, *UR*, *ML*, *MR*, *LL*, *LM*, *LR*, and *P*. However, our initial experimental results indicated that in some cases the detector failed to detect *P*. This may be due to the fact that *P* is relatively small compared to the total image size. In addition, it was only able to detect some members of *BB*. As a result, in order to leverage the accuracy of detection for a pupil, a post-processing algorithm is required to approximate the pupil from some detected members of *BB*. The algorithm is only activated if and only if *P* cannot be detected. In this way, it means that we give higher priority to approximating the location of a pupil to *P*. In this instance, the location of a pupil is taken to be the center of *P*. Otherwise, the algorithm utilizes the geometric relationships between the detected members of *BB* to estimate a pupil center. Let us define the horizontal extension of the bounding box of *B*, or *B′*, as the box with the same height as *B*, while its width is increased to be equal to the width of the image. In addition, the vertical extension of the bounding box of *B* is the bounding box with the same width as *B*, but its height is extended to match the height of an image. Additionally, we define two functions, *max(a,b)* and *min(a,b)*, which produce as their output the minimum and maximum values between a and b, respectively. Assume that the coordinates of a bounding box *X* are (L_X, T_X, *RX*, B_X). Moreover, all detected members of *BB* have already been clustered into the appropriate group; that is, all detected members of *BB* around the left eye pupil are all clustered within the same group. In practice, these members of *BB* can be processed trivially without adding the additional processing time to the algorithm. This is due to the fact that all members of *BB* appear to be grouped separately. Several are found in close proximity to the corresponding eye within the image.

To estimate *P* using a set of detected *BB*, the following rules can be drawn:

- For a pair of *UM* and *LM* bounding boxes, *P* is estimated by: $P^* = (min(L_{UM},L_{LM}),B_{UM},max(R_{UM},R_{LM}),T_{LM})$ (see Figure 3-a),
- For a pair of *ML* and *MR* bounding boxes, *P* is estimated by: $P^* = (R_{ML},min(T_{ML},T_{MR}),L_{MR},max(B_{ML},B_{MR}))$ (see Figure 3-b),
- For a pair of *UL* and *LR* bounding boxes, *P* is estimated by: $P^* = (R_{UL},B_{UL},L_{LR},T_{LR})$ (see Figure 3-c),
- For a pair of *UR* and *LL* bounding boxes, *P* is estimated by: $P^* = (R_{LL},B_{UR},L_{UR},T_{LL})$ (see Figure 3-d),
- For a pair of *UL* and *MR* bounding boxes, one possible *P* is: $P^* = (R_{UL},B_{UL},L_{MR},B_{MR})$ (see Figure 3-e),
- For a pair of *UL* and *LM* bounding boxes, *P* is estimated by: $P^* = (R_{UL},B_{UL},R_{LM},T_{LM})$ (see Figure 3-f),
- For a pair of *UR* and *LM* bounding boxes, *P* is estimated by: $P^* = (L_{LM},T_{UR},R_{LM},T_{LM})$ (see Figure 3-g),
- For a pair of *UR* and *ML* bounding boxes, *P* is estimated by: $P^* = (R_{ML},T_{ML},L_{UR},B_{ML})$ (see Figure 3-h),
- If a pair of closed neighbor members of *BB* is either *UM* - *ML* or *UM* - *MR* or *LM* - *ML* or *LM* - *MR*, *P* is estimated by the intersection region between the appropriate horizontal and vertical extensions of these bounding boxes. The examples for the cases of *UM* - *ML* and *LM* - *MR* are illustrated in Figures 3-i and 3-j,

- Otherwise, no *P* can be estimated.

In practice, a set of *BB* gives rise to multiple *P**. Our algorithm finds the intersection between all members of *P** and the result is finally used to represent a single *P*.

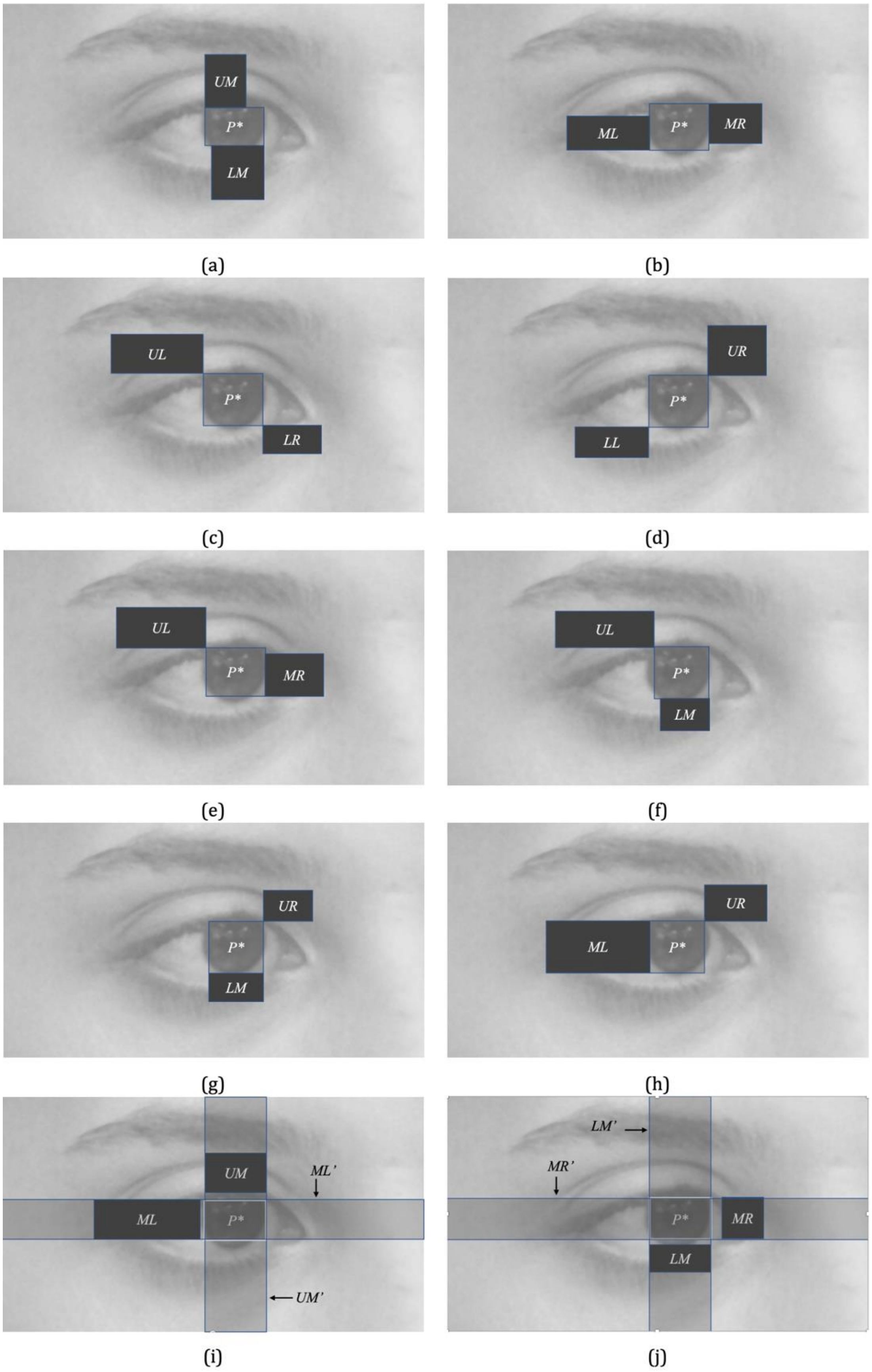

Figure 3. Different cases to estimate P based on using the detected set of *BB*

2-2-3- Evaluation

Our proposed approach is benchmarked against the state-of-the-art approaches using the relative error measure e_{max}, a measure proposed by Jesorsky et al. [28]. The error is defined as follows:

$$e_{max} = \frac{max(d_l,d_r)}{IPD} \tag{2}$$

Here, d_l and d_r are the Euclidean distances between the ground truth and the approximated locations of the left and right pupils, respectively. The *IPD* can be calculated as follows:

$$IPD = \sqrt{\left(x_{(p,l)} - x_{(p,r)}\right)^2 + \left(y_{(p,l)} - y_{(p,r)}\right)^2} \tag{3}$$

where $(x_{(p,l)},y_{(p,l)})$ and $(x_{(p,r)},y_{(p,r)})$ correspond to the centers of the left and right pupils, respectively. Accuracy is calculated as a percentage of images for which this error falls below certain thresholds. State-of-the-art approaches, especially the one benchmarked by our approach [23], are commonly compared using $e_{max} \leq 0.025(2.5\%)$, $e_{max} \leq 0.050(5\%)$ and $e_{max} \leq 0.100(10\%)$.

Contrary to the previously proposed approaches that derived the accuracy of detection from the ratio of a true positive (TP) to the sum of a true positive and a false positive (FP), we used the relative error as a basis for determining the accuracy of detection. The reason for this is that a single detection of the pupil cannot be used as a basis for determining the e_{max}. Therefore, it should not be regarded as a positive result. The accuracy of detection is redefined in this context as a function of the contribution of detection results derived from an image in which left and right pupils are detected.

3- Results and Discussion

Different experiments were carried out to create the detectors from the PUPPIE dataset with variations of the α and β parameters: α was between 1 and 5, and the β parameter was in the range of 1.0000 to 1.8175 with a 0.0625 increment. Figure 4 presents a summary of the steps involved in finding the best detector from a set of detectors that are trained with different combinations of α and β parameters using a variety of datasets. Our observations of the loss values recorded during the detectors' training stage found adequate detectors with α = 3 and β between 1.1250 to 1.8175 with a 0.0625 increment. These detectors were likely to produce a similar characteristic of loss curves that sharply drops after the first 800 iterations and gradually decay to around 0.0 at the end of the training stage. Figure 5 illustrates the comparative loss curves of these detectors. Obviously, concerning the training dataset, these detectors seemed to learn to detect all the classes successfully. These detectors were elaborately compared to find the most effective one regarding the accuracy of detection and the relative errors: $e_{max} \leq 0.025$, $e_{max} \leq 0.050$, and $e_{max} \leq 0.100$. The GI4E dataset was used to serve this purpose. The experiment results for the top ten most effective detectors are presented in Table 2. From the results, the winner detector was the one that was trained with α = 3 and β = 1.8125 datasets at the checkpoint of 6,000 because this detector gave rise to the best accuracy of detection and almost all the best relative errors.

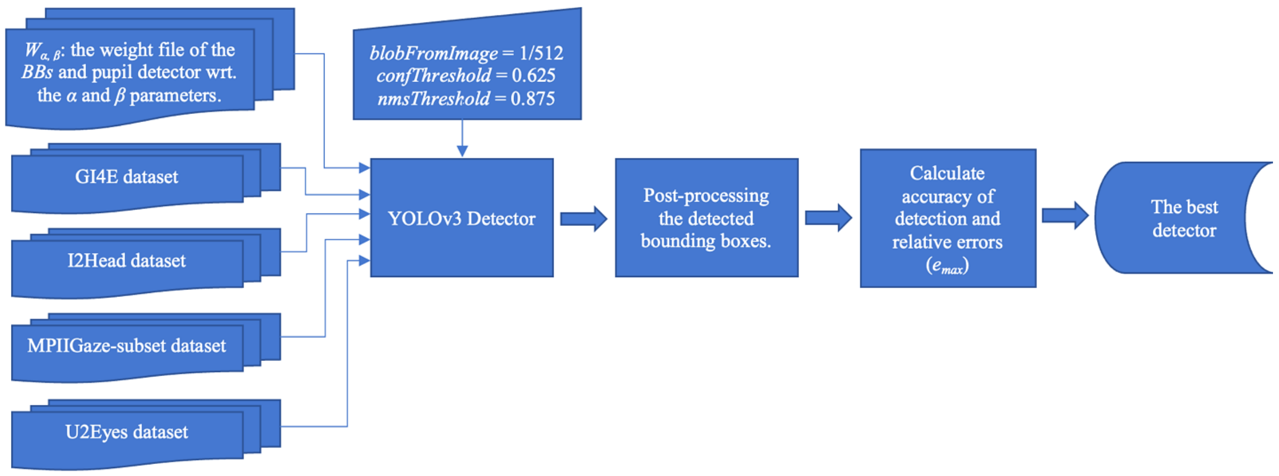
$W_{\alpha,\beta}$: the weight file of the BBs and pupil detector wrt. the α and β parameters.
GI4E dataset
I2Head dataset
MPIIGaze-subset dataset
U2Eyes dataset
blobFromImage = 1/512
confThreshold = 0.625
nmsThreshold = 0.875
YOLOv3 Detector
Post-processing the detected bounding boxes.
Calculate accuracy of detection and relative errors (e_{max})
The best detector

Figure 4. A summary of the steps involved in finding the best detector from a set of detectors that are trained with different combinations of α and β parameters using a variety of datasets

Table 3 compares the relative error for pupil center estimation on the GI4E dataset between the winner detector against the previous approaches, which produced the relative error $e_{max} \leq 0.025$ greater than or equal to 79.50% [23]. It is noticeable that all previous approaches employ DNN to create the detectors. However, they mainly focus on adding new processing layers or increasing the number of layers to the architectures to extract more valuable features and train their detectors. Our winner detector is superior to almost all these approaches. It is only less effective but comparable to the state-of-the-art approach of Larumbe-Bergera et al. [23]. It can be explained that since the approach of Larumbe-Bergera et al. [29] relies on using the deeper ResNet-50 backbone with additional five layers for pupil detection. It also requires an external library to perform face detection, the MTCNN, before performing a pupil center estimation. It seems to be less effective when the computation time is considered. The experiment results in terms of the computational time will be provided shortly.

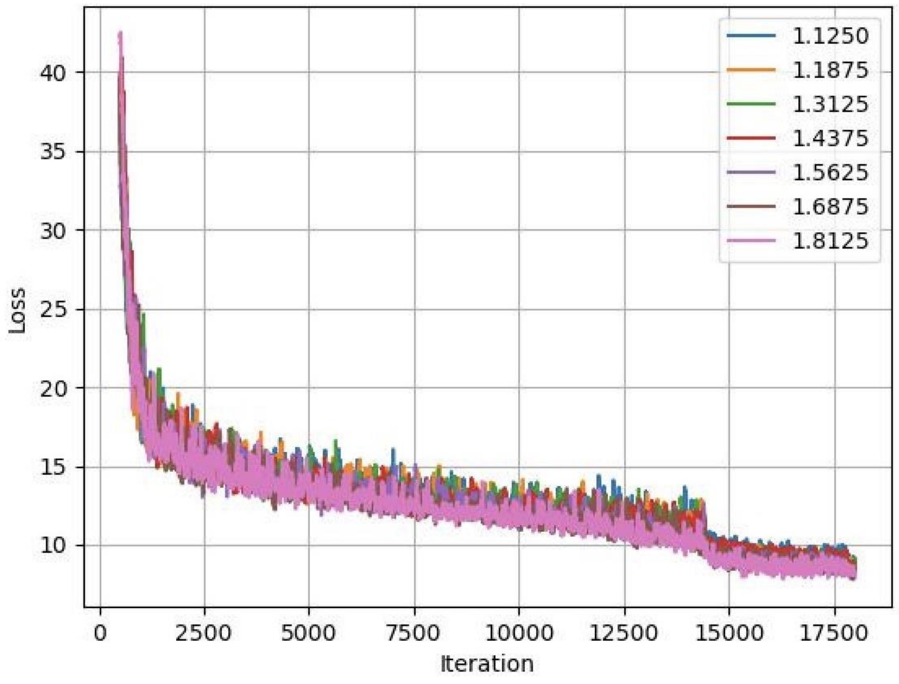

Figure 5. **Comparison of the detectors' loss curves trained from the datasets with α = 3 and β between 1.1250 to 1.8175 with a 0.0625 increment**

Table 2. **Comparison of experiment results between the ordinary detector and several detectors produced by training with our datasets. Acc. of det. is the accuracy of detection**

Eye Expansion (%)	Iteration	Acc. of Det.	Relative errors (e_{max})		
			$e_{max} \leq 0.025$	$e_{max} \leq 0.050$	$e_{max} \leq 0.100$
1.5625	4000	98.14	93.73	97.61	98.19
1.3125	4000	97.49	91.29	92.37	92.37
1.8125	5000	97.41	97.59	99.67	99.92
1.8125	**6000**	**96.84**	**98.24**	**99.75**	**99.92**
1.3125	3000	96.52	97.49	98.58	98.66
1.5625	5000	96.52	96.81	99.66	99.75
1.8125	8000	95.23	97.45	99.07	100.00
1.1250	8000	95.15	99.57	100.00	100.00
1.1875	15000	94.74	98.21	99.66	99.74
1.6875	9000	94.74	95.47	97.01	97.01

Table 3. **Relative errors comparison for pupil center location on the GI4E database for the approaches which produced the relative error $e_{max} \leq 0.025$ greater than or equal to 79.50% [18].**

Publications	Relative errors (e_{max})		
	$e_{max} \leq 0.025$	$e_{max} \leq 0.050$	$e_{max} \leq 0.100$
Kim et al. [16]	79.50	99.30	99.90
Lee et al. [8]	79.50	99.84	99.84
Cai et al. [17]	85.70	99.50	-
Larumbe et al. [23]	87.67	99.14	99.99
Levinshtein et al. [30]	88.34	99.27	99.92
Choi et al. [6]	90.40	99.60	-
Kitazumi et al. [9]	96.28	98.62	98.95
Larumbe-Bergera et al. [25]	98.46	100.00	100.00
Our approach (α = 3, β = 1.8125, 6000)	98.24	99.75	99.92

The winner detector with the configuration of $\alpha = 3$ and $\beta = 1.8125$ at the checkpoint of 6,000 was then further used to estimate a pupil center on the I2HEAD, MPIIGaze, and U2Eyes datasets. Figure 6 illustrates some samples of randomly selected output images from the winner detector on these datasets; the GI4E dataset is also included, arranged on the dataset basis from top to bottom: GI4E, I2Head, MPIIGaze, and U2Eyes; respectively. In each column within the figure, the images demonstrate the cases where pupils are estimated from (1) both detected pupils, (2) one detected pupil and the other from operating the post-processing algorithm, and (3) both from operating the post-processing algorithm; respectively. It is noted that all the white bounding boxes indicate the member of *BB*. All the green boxes indicate the member of *P*. The white and the green plus signs are the ground truths and the estimated pupil centers, respectively. The results from benchmarking between the winner detector and the state-of-the-art approach of Larumbe-Bergera et al. [23] are summarized in Table 4. Considering the $e_{max} \leq 0.025$, it is apparent that the winner detector produces comparable relative errors to the state-of-the-art one on all the datasets. It is significantly more effective to operate on the U2Eyes dataset. Overall, the winner detector and the state-of-the-art one produce a similar trend of relative errors. That is to say, they are arranged in the following order: GI4E, MPIIGaze, I2Head, and U2Eyes. The relative errors of the first three datasets are comparable between the winner detector and the state-of-the-art one. It can be explained that the nature of the U2Eyes dataset consists of synthesized close-up images. All features within the pupils and eyes are apparent, sharp, and seem to be free from any disturbances. Some natural features inherent in an image, which the detectors require to indicate a pupil correctly, are failed to imitate. It results in a significant failure to correctly locate a pupil. The table shows that the winner detector lags behind the state-of-the-art counterpart on the MPIIGaze dataset. It can be explained that the nature of the MPIIGaze dataset consists of a high percentage of persons wearing eyeglasses (see one example in Figure 6-g)). In contrast, the PUPPIE dataset used to create the winner detector contains only a few samples. It could affect the features of eyeglasses' appearance unknown to the winner detector. It is especially true when parts of eyeglasses partially cover a person's eye.

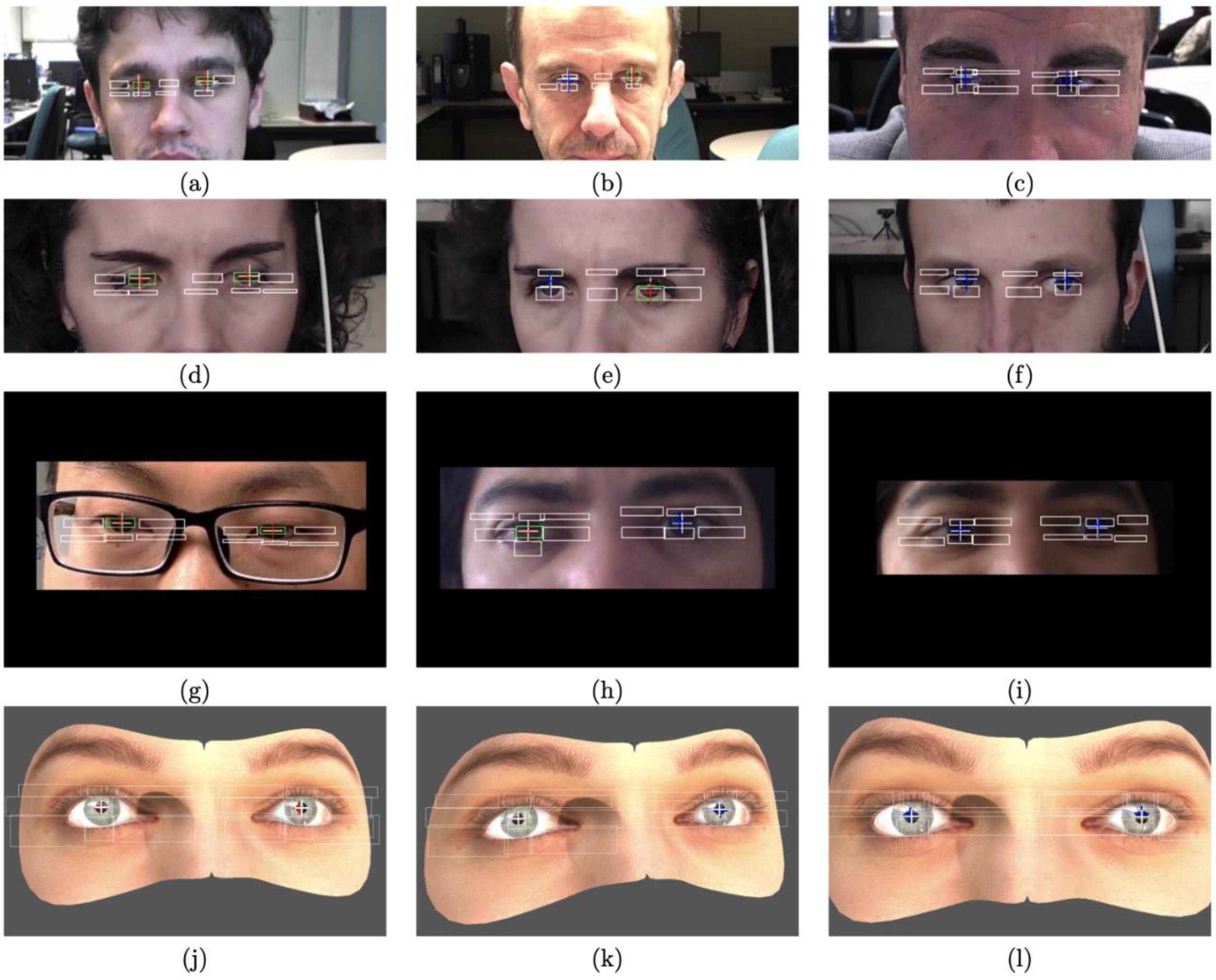

Figure 6. Randomly selected samples output images from the proposed approach on a dataset basis from top to bottom: GI4E, I2Head, MPIIGaze and U2Eyes. Each column demonstrates the cases where pupils are estimated from both detected pupils, from one detected pupil and from the result of the post-processing algorithm, and both from the post-processing algorithm, respectively. All the white bounding boxes are the members of BB. All the green boxes are the members of *P*. The white and the green plus signs are the ground truths and the estimated pupil center locations, respectively.

Table 4. Accuracies of detection (Acc. of det.) and relative errors for pupil center location on GI4E, I2HEAD, MPIIGaze and U2Eyes datasets

Approaches	Datasets	Acc. of Det.	Relative errors (e_{max})		
			$e_{max} \leq 0.025$	$e_{max} \leq 0.050$	$e_{max} \leq 0.100$
Larumbe-Bergera et al. [18]	GI4E		98.46	100.00	100.00
	I2Head		96.88	100.00	100.00
	MPIIGaze-subset		97.09	99.83	100.00
	U2Eyes		93.44	99.93	100.00
Present Study	GI4E	96.84	98.24	99.75	99.92
	I2Head	99.14	96.68	98.00	98.00
	MPIIGaze-subset	84.27	96.84	97.62	98.41
	U2Eyes	94.34	94.70	97.37	98.41

Finally, the average detection times between our detector and the state-of-the-art one were measured on the common computer platform. These are compared and shown in Table 5. The state-of-the-art one was claimed to take about 2.00 ms to operate on an Intel Xeon E5-1650 v4 CPU with an Nvidia Titan X (Pascal) GPU and about 5.00 ms using an Intel i7-6700k CPU and Nvidia GTX 960 GPU. Our winner detector takes 0.80 and 1.97 ms for the same procedure on the former and latter platforms. Our approach improves computational time performance by up to 2.5 times. The average detection time of our detector was also tested on a popular low-performance platform; the Raspberry Pi 4, with the following specifications: Broadcom BCM2711, Quad-core Cortex-A72 (ARM v8) 64-bit SoC running at 1.5GHz with Broadcom VideoCore VI. The average detection time is 158.95 ms. It is equivalent to the frame rate of 6.25. It is a promising performance for the detector to be employable on a popular low-performance platform. It would open further applications that rely on a less invasive camera-based eye-tracking technology.

Table 5. Comparison of the average detection time between the winner detector and the state-of-the-art one

Approaches	Detection times (ms)		
	Xeon E5-1650 + Titan X	i7-6700k + GTX 960	Raspberry Pi
Larumbe-Bergera et al. [18]	2.00	5.00	NA
Present Study	0.80	1.97	158.95

At this point, it can be seen that our proposed approach has been proven to be more effective in creating a pupil center estimator regarding the accuracy of detection, the relative errors, and the detection time. The detected regions surrounding a pupil and the algorithm to post-process these regions significantly enhance the effectiveness of the detector. Our proposed approach contradicts the previous understanding that (1) a detector must be trained from a large-sized dataset and (2) a detector can be tailored to be more effective either by incorporating more processing layers into its architecture or fine-tuning some training hyperparameters. Our proposed approach also produces a comparable detection accuracy on a shallower layer of detector compared to the state-of-the-art one. The enhanced effectiveness of our detector comes from the fact that it learns a variety of features from the surrounding regions of a pupil. These are valuable for being post-processed to determine the pupil center if the detector fails to detect a pupil directly.

4- Conclusion

In this paper, a novel approach to pupil center estimation is presented. As opposed to increasing the size of the dataset for training or adding more layers of processing to a DNN detector, this approach involves training the detector with the region surrounding the pupil and the region between the pupil and the eye. This latter set of regions is divided into eight subregions. Each subregion has an annotation indicating its class and its position relative to the pupil's region. In order to determine the pupil center, the proposed post-processing algorithm is used in conjunction with the detected results. The size of a pupil and an eye were varied during experimentation to generate several different detectors. These detectors were derived from the PUPPIE dataset. A tiny-YOLOv3 with 23 layers was trained to make real-time performance available even on computer systems without GPU processing power. A winner detector has been selected by validating the detectors against the GI4E dataset in order to achieve the best detection and estimation accuracy, which is determined by the percentage of pupil center estimation errors below the size of a constricted pupil e_{max}. Additionally, the winner detector was examined with the following datasets: I2Head, MPIIGaze, and U2Eyes. The results indicate that the accuracy of pupil estimation is comparable to that of the state-of-the-art approach. It achieves pupil center estimation errors that are below the size of a constricted pupil in more than 98.24% of the images. In addition, the average detection time of the winner detector is 2.8 times faster than the state-of-the-art approach. A low-performance platform was tested on the Raspberry Pi 4 and achieved 6.25 frames per second. It would allow for further applications involving less intrusive camera-based eye-tracking technology that can be processed on a low-performance computing platform. Additionally, the approach is highly expected to be applicable to the DNN detection problem, whose objects share similar characteristics to a pupil.

5- Declarations

5-1- Author Contributions

Conceptualization, W.K.; methodology, W.K.; software, K.S., and G.K.; validation, W.K., and P.K. writing original draft preparation, W.K., P.K., and G.K. All authors have read and agreed to the published version of the manuscript.

5-2- Data Availability Statement

Pupil annotations from the PUPPIE dataset has been used (available at *https://www.unavarra.es/gi4e/databases/elar* (accessed on March 2022).

5-3- Funding

The authors received no financial support for the research, authorship, and/or publication of this article.

5-4- Institutional Review Board Statement

Not applicable.

5-5- Informed Consent Statement

Not applicable.

5-6- Conflicts of Interest

The authors declare that there is no conflict of interest regarding the publication of this manuscript. In addition, the ethical issues, including plagiarism, informed consent, misconduct, data fabrication and/or falsification, double publication and/or submission, and redundancies have been completely observed by the authors.

6- References

[1] Guestrin, E. D., & Eizenman, M. (2006). General theory of remote gaze estimation using the pupil center and corneal reflections. IEEE Transactions on Biomedical Engineering, 53(6), 1124–1133. doi:10.1109/TBME.2005.863952.

[2] Duchowski, A. T. (2003). Eye Tracking Techniques. Eye Tracking Methodology: Theory and Practice, 55–65, Springer, London. doi:10.1007/978-1-4471-3750-4_5.

[3] Cognolato, M., Atzori, M., & Müller, H. (2018). Head-mounted eye gaze tracking devices: An overview of modern devices and recent advances. Journal of Rehabilitation and Assistive Technologies Engineering, 5, 1-13. doi:10.1177/2055668318773991.

[4] Fuhl, W., Santini, T., Kasneci, G., & Kasneci, E. (2016). Pupilnet: Convolutional neural networks for robust pupil detection. arXiv preprint arXiv:1601.04902. doi:10.48550/arXiv.1601.04902.

[5] Fuhl, W., Santini, T., Kasneci, G., Rosenstiel, W., & Kasneci, E. (2017). Pupilnet v2. 0: Convolutional neural networks for CPU based real time robust pupil detection. arXiv preprint arXiv:1711.00112. doi: 10.48550/arXiv.1711.00112.

[6] Choi, J. H., Il Lee, K., Kim, Y. C., & Cheol Song, B. (2019). Accurate Eye Pupil Localization Using Heterogeneous CNN Models. 2019 IEEE International Conference on Image Processing (ICIP). doi:10.1109/icip.2019.8803121.

[7] Xia, Y., Yu, H., & Wang, F. Y. (2019). Accurate and robust eye center localization via fully convolutional networks. IEEE/CAA Journal of Automatica Sinica, 6(5), 1127–1138. doi:10.1109/JAS.2019.1911684.

[8] Lee, K.I., Jeon, J.H., Song, B.C. (2020). Deep Learning-Based Pupil Center Detection for Fast and Accurate Eye Tracking System. Computer Vision – ECCV 2020. ECCV 2020. Lecture Notes in Computer Science. Springer, Cham, Switzerland. doi:10.1007/978-3-030-58529-7_3.

[9] Kitazumi, K., & Nakazawa, A. (2019). Robust Pupil Segmentation and Center Detection from Visible Light Images Using Convolutional Neural Network. 2018 IEEE International Conference on Systems, Man, and Cybernetics (SMC). doi:10.1109/SMC.2018.00154.

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

[11] Ronneberger, O. (2017). Invited Talk: U-Net Convolutional Networks for Biomedical Image Segmentation. Bildverarbeitung für die Medizin 2017. Informatik Aktuell. Springer, Berlin, Heidelberg. doi:10.1007/978-3-662-54345-0_3.

[12] Zdarsky, N., Treue, S., & Esghaei, M. (2021). A Deep Learning-Based Approach to Video-Based Eye Tracking for Human Psychophysics. Frontiers in Human Neuroscience, 15. doi:10.3389/fnhum.2021.685830.

[13] Mathis, A., Mamidanna, P., Cury, K. M., Abe, T., Murthy, V. N., Mathis, M. W., & Bethge, M. (2018). DeepLabCut: markerless pose estimation of user-defined body parts with deep learning. Nature Neuroscience, 21(9), 1281–1289. doi:10.1038/s41593-018-0209-y.

[14] Insafutdinov, E., Pishchulin, L., Andres, B., Andriluka, M., Schiele, B. (2016). DeeperCut: A Deeper, Stronger, and Faster Multi-person Pose Estimation Model. Computer Vision – ECCV 2016. ECCV 2016. Lecture Notes in Computer Science. Springer, Cham, Switzerland. https://doi.org/10.1007/978-3-319-46466-4_3.

[15] He, K., Zhang, X., Ren, S., & Sun, J. (2016). Deep Residual Learning for Image Recognition. 2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR). doi:10.1109/cvpr.2016.90.

[16] Kim, S., Jeong, M., & Ko, B. C. (2020). Energy Efficient Pupil Tracking Based on Rule Distillation of Cascade Regression Forest. Sensors, 20(18), 5141. doi:10.3390/s20185141.

[17] Cai, H., Liu, B., Ju, Z., Thill, S., Belpaeme, T., Vanderborght, B., & Liu, H. (2018). Accurate eye center localization via hierarchical adaptive convolution. 29th British Machine Vision Conference. British Machine Vision Association, 3-6 September 2018, Newcastle, United Kingdom.

[18] Lee, K.I., Jeon, J.H., Song, B.C. (2020). Deep Learning-Based Pupil Center Detection for Fast and Accurate Eye Tracking System. Computer Vision – ECCV 2020. Lecture Notes in Computer Science. Springer, Cham, Switzerland. https://doi.org/10.1007/978-3-030-58529-7_3.

[19] Brousseau, B., Rose, J., & Eizenman, M. (2020). Hybrid eye-tracking on a smartphone with CNN feature extraction and an infrared 3D model. Sensors (Switzerland), 20(2), 543. doi:10.3390/s20020543.

[20] Ou, W. L., Kuo, T. L., Chang, C. C., & Fan, C. P. (2021). Deep-learning-based pupil center detection and tracking technology for visible-light wearable gaze tracking devices. Applied Sciences (Switzerland), 11(2), 851.

[21] Poulopoulos, N., Psarakis, E. Z., & Kosmopoulos, D. (2021). PupilTAN: A Few-Shot Adversarial Pupil Localizer. 2021 IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops (CVPRW). doi:10.1109/cvprw53098.2021.00350.

[22] Kang, D., & Chang, H. S. (2021). Low-complexity pupil tracking for sunglasses-wearing faces for glasses-free 3d Huds. Applied Sciences (Switzerland), 11(10), 4366. doi:10.3390/app11104366.

[23] Larumbe-Bergera, A., Garde, G., Porta, S., Cabeza, R., & Villanueva, A. (2021). Accurate pupil center detection in off-the-shelf eye tracking systems using convolutional neural networks. Sensors, 21(20), 6847. doi:10.3390/s21206847.

[24] Lin, Z., Liu, Y., Wang, H., Liu, Z., Cai, S., Zheng, Z., … Zhang, X. (2022). An eye tracker based on webcam and its preliminary application evaluation in Chinese reading tests. Biomedical Signal Processing and Control, 74, 103521. doi:10.1016/j.bspc.2022.103521.

[25] Larumbe-Bergera, A., Porta, S., Cabeza, R., & Villanueva, A. (2019). SeTA. Proceedings of the 11th ACM Symposium on Eye Tracking Research & Applications. doi:10.1145/3314111.3319830.

[26] Redmon, J., & Farhadi, A. (2018). Yolov3: An incremental improvement. arXiv preprint arXiv:1804.02767. doi:10.48550/arXiv.1804.02767.

[27] Bradski, G. (2000). The openCV library. Dr. Dobb's Journal: Software Tools for the Professional Programmer, 25(11), 120-123.

[28] Jesorsky, O., Kirchberg, K.J., Frischholz, R.W. (2001). Robust Face Detection Using the Hausdorff Distance. Audio- and Video-Based Biometric Person Authentication. AVBPA 2001. Lecture Notes in Computer Science, Springer, Berlin, Heidelberg. doi:10.1007/3-540-45344-X_14.

[29] Larumbe, A., Cabeza, R., & Villanueva, A. (2018). Supervised descent method (SDM) applied to accurate pupil detection in off-the-shelf eye tracking systems. Proceedings of the 2018 ACM Symposium on Eye Tracking Research & Applications. doi:10.1145/3204493.3204551.

[30] Levinshtein, A., Phung, E., & Aarabi, P. (2018). Hybrid eye center localization using cascaded regression and hand-crafted model fitting. Image and Vision Computing, 71, 17–24. doi:10.1016/j.imavis.2018.01.003.