Detectable Object-sizes Range Estimation based Multi-task Cascaded Convolutional Neural Networks in the vehicle Environment

Whui Kim, Hyun Kyun Choi
Intelligent Robotics Research Division
Electronics and Telecommunications Research Institute
Daejeon, Korea
{khsunkh, choihk}@etri.re.kr

Min Jung Shin
Department of Electronics & Communications Engineering
KwangWoon University
Seoul, Korea
smj139052@kw.ac.kr

***Abstract*—In many studies regarding driver monitoring system, they directly employed face detection algorithms for general-purpose in an unconstrained environment. These algorithms are generally not suitable for limited resources of vehicles. Unlike the general unconstrained environment, the range of detectable face sizes and locations can be estimated in the vehicle environment. In this paper, we propose the Detectable Object-sizes Range Estimation algorithm (DORE) to estimates the range of detectable face sizes through specific information in the vehicle environment. The DORE algorithm makes images, in which a face is likely to be detected in the in-vehicle environment, to be fed into a face detection algorithm, such as Multi-task cascaded Convolutional Neural Networks (MTCNN) which stably detect faces rather than others. Our experiment shows that DORE applied MTCNN not only had the same performance as MTCNN in terms of accuracy but also had relatively low processing time in the vehicle environment.**

***Index Terms*—Multi-task Cascaded Convolutional Neural Networks, MTCNN, Detectable Object-sizes Range Estimation, DORE, Pyramid image**

I. Introduction

Face detection is one of the most active research areas in a wide range of industries, and many applications of the automotive industry also have used that. In-vehicle applications such as driver expression, emotion, and status recognition typically involve face detection module as well as face tracking and recognition modules. These modules are sequential in order of detection, tracking, and recognition. Face detection corresponding to the pre-processing of the entire system inevitably affects subsequent modules. Although problems such as error transition or performance degradation may arise, many studies directly employ face detection algorithms developed for general purpose in an unconstrained environment.

In-vehicle datasets have different characteristics from unconstrained environments. For general purpose face detection in an unconstrained environment, widely used face detection datasets consist of gathered images from websites such as google.com, yahoo.com, and flickr.com [1]. Images These datasets have various image sizes and face locations. In the in-vehicle environment, datasets consist of images acquired from a limited area, such as the drivers seat, using a fixed camera mounted on dashboard, room mirror, or pillars. For each driver in a vehicle, the distance between the driver and the mounted camera is comparatively stable. Therefore, the variation of the face size in the driver's image is not significant.

The general-purpose face detection algorithms typically use the pyramid image that includes variously resized images. Those algorithms demand a larger size range of resized images to detect various size faces. Because the variation of the driver's face size is comparatively stable, using a larger size range of the pyramid image cause a waste of computing resources.

If computing resources are large, such as using a cloud server or a high-performance PC, the waste of computing resources may not be a severe problem. However, it is still difficult to use a cloud server and high-performance PCs in vehicles due to the following constraints. First, driver fatigue or drowsiness is mainly caused by driving for a long time on a highway with poor communication infrastructure, so it is necessary to use resources in the vehicle. Next, the use of relatively low priced and performance embedded boards is demanded to commercialize in-vehicle applications.

A GPU equipped embedded board which is relatively inexpensive can slightly alleviate the problems caused by the above constraints. In the in-vehicle applications, a recognition module consumes as much computing resources as the face detection module. The computing resource of the GPU equipped embedded board is still insufficient to detect and recognize the face simultaneously. Therefore, unnecessary waste of computing resources in the face detection needs to be minimized because the computing resource is limited in the vehicle.

The general-purpose face detection algorithms can be configured to suit the vehicle environment with a constant value considering the characteristics of all vehicles. However, they are still not adaptable to a variety of vehicles or drivers. To alleviate this, we propose a Detectable Object-sizes Range Estimation (DORE) algorithm, which estimates the range of object sizes through the information of the driver and the vehicle. With DORE, a pyramid image can consist only of

978-1-7281-1220-6/19/$31.00 ©2019 IEEE

images, in which a face is likely to be detected. It can reduce the waste of computing resources caused by attempting to find faces of non-existent sizes. Experiments show that the Multi-task cascaded Convolutional Neural Network (MTCNN) [2] face detector can improve in terms of processing time without accuracy reduction by using DORE.

II. Related Work

Traditional face detection algorithms including Haar-Cascade [3], LBP-Cascade [4], HoG-SVM [6], mainly rely on hand-crafted features and shallow learning-based classifiers. These algorithms have simple architecture and work almost real-time. However, these can degrade in real-world applications with large visual variations of faces and multiple detectors are used to detect multi-view or occluded faces [2].

Research of convolutional neural networks (CNN) over the past few years resolve these problems with a CNN model [5] such as the CNN version of MMOD [6], SSD [7], Faster RCNN [8], and YOLO [9]. CNN based face detection algorithms have significant performance [7]. However, these algorithms aim at large scale visual recognition and require a large number of computing resources.

Unlike these algorithms, which aim at large scale visual recognition, MTCNN was proposed to detect a face and has a lightweight CNN architecture. In terms of accuracy as well as processing time, MTCNN outperforms other state-of-the-art methods of face detection and has been widely used for face localization and alignment. However, MTCNN is still not easy to be implemented for real-time face detection [10].

Many studies have been carried out recently to improve MTCNN. These studies mention that among PNet, RNet, and ONet included in MTCNN, PNet accounts for most of the processing time. These studies attempt to improve the processing time of PNet by adding multi-task classifiers of different facial parts or branches proposing face locations to PNet. These methods reduce or eliminate the level of the pyramid image [10]–[12].

However, these algorithms still aim at general-purpose in the unconstrained environment and are attempting to detect faces of non-existent sizes in the vehicle environment. DORE reduces the level of the pyramid image by the camera specifications and the comparative distance between the driver and the mounted camera. DORE enables MTCNN to detect faces only at the level of the pyramid image where faces can exist.

III. Detectable Object-sizes Range Estimation Based Multi-task Cascaded Convolutional Neural Networks in the Vehicle Environment

A. Face Detection in Driver Monitoring System

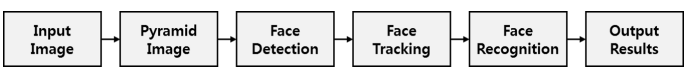

Fig. 1. Example of Driver Monitoring System

General driver monitoring systems consist of face detection, tracking, and recognition, as shown in Fig. 1. Due to this sequential structure, the performance of face detection can impact on subsequent modules and the entire system. In this paper, a driver monitoring system, which consists of MTCNN face detector, Kernelized Correlation Filter tracker [13], and Xception classifier [14] for recognition, was used to analyze the effect of face detection on subsequent modules and the entire system in terms of processing time.

In the structure shown in Fig. 1, although face detection help to improve processing efficiency and accuracy of face recognition by feeding the higher resolution face images into recognition module, it takes a relatively large amount of computation as much as recognition. We experimented with this structure on about $2,500$ images. Processing times were measured when a face was not detected and tracked in about 10 percent of the images. The experiment environment details are in section IV-A. As shown in Table I, detection consumes more than half of the entire processing time. Especially, Proposal Network and additional procedure consume about 20 percent of the entire processing time, respectively.

TABLE I
Processing time of detection, tracking, and recognition

Type		Processing time(ms)		Proportion(%)
MTCNN Detection	Proposal Network(PNet)	13.357	31.987	52.59
	Refine Network(RNet)	4.447		
	Output Network(ONet)	2.098		
	Additional Procedure	12.085		
KCF Tracker		7.605		12.50
Xception Recognition		21.230		34.90
Total		60.822*		100.0

* Frames Per Second is 16.442

B. Detectable Object-sizes Range Estimation in the Vehicle Environment

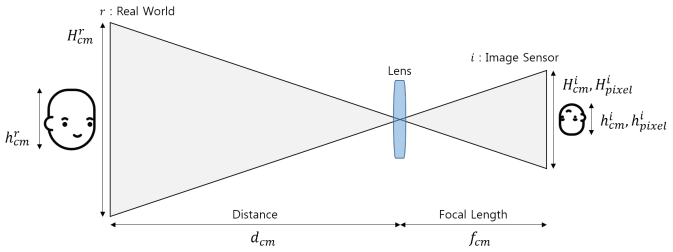

Fig. 2. A geometry of a pinhole camera as seen from the horizontal axis

Equation (1)-(3) describe the DORE algorithm. Fig. 2 describes the notations of the equations. Each of ${H^r}_{cm}$, ${h^r}_{cm}$, and d_{cm} means the size of the field of view, the size of the object, and working distance in the real world, respectively. As the specifications of the camera, ${H^i}_{cm}$, ${H^i}_{pixel}$, and f_{cm} means the size of the image sensor, the pixel size of the image, and the focal length of the lens, respectively. ${h^i}_{cm}$ means the size of the objects projection onto the image sensor plane, and ${h^i}_{pixel}$ means the pixel size of the projected object on the image. ${h^i}_{pixel}$ is directly proportional to ${h^r}_{cm}$ and inversely proportional to d_{cm}. α is proportionality constant of ${h^i}_{pixel}$ and the ratio ${h^r}_{cm}$ to d_{cm} and is calculated by Equation (2).

$$\frac{h^i{}_{pixel}}{h^i{}_{cm}} = \frac{H^i{}_{pixel}}{H^i{}_{cm}} \qquad \frac{h^i{}_{cm}}{h^r{}_{cm}} = \frac{H^i{}_{cm}}{H^r{}_{cm}} = \frac{f_{cm}}{d_{cm}}$$
$$h^i{}_{pixel} = \frac{H^i{}_{pixel}}{H^i{}_{cm}} \times h^i{}_{cm} \qquad h^i{}_{cm} = \frac{f_{cm}}{d_{cm}} \times h^r{}_{cm} \tag{1}$$

$$\begin{aligned} h^i{}_{pixel} &= \frac{H^i{}_{pixel}}{H^i{}_{cm}} \times \frac{f_{cm}}{d_{cm}} \times h^r{}_{cm} \\ &= \alpha \times \frac{h^r{}_{cm}}{d_{cm}} \ (\text{ if } \alpha = \frac{H^i{}_{pixel}}{H^i{}_{cm}} \times f_{cm}) \end{aligned} \tag{2}$$

$$\alpha \times \frac{\max(h^r{}_{cm})}{\min(d_{cm})} \geq h^i{}_{pixel} \geq \alpha \times \frac{\min(h^r{}_{cm})}{\max(d_{cm})} \tag{3}$$

To estimate the range of detectable face sizes, we need to calculate each size of detectable face sizes according to the camera specifications and the distance between the driver and the mounted camera. The size of detectable faces can be calculated by Equation (2). $h^i{}_{pixel}$ is the size of detectable faces in the in-vehicle environment.

We can estimate a range of detectable object sizes if we know the max and min values of $h^i{}_{pixel}$. The calculation procedure of these values is as follows. First, calculate α with camera specifications by Equation (2). For example, the α is 306.383 when utilizing the Raspberry Pi camera with specifications of Table II [15].

TABLE II
SPECIFICATION OF THE RASPBERRY PI CAMERA MODULE V1

Specifications	Notation	Width	Height
Image Resolution	$H^i{}_{pixels}$	320 pixels	240 pixels
Sensor Image Area	$H^i{}_{cm}$	0.376 cm	0.274 cm
Focal Length	f_{cm}	0.360 cm	

Next, the pixel size of objects on the image can be calculated using the following prerequisites. For adults in the United States, the length of head breadth is from $10.2cm$ to $17.4cm$, and the size of the end of jaw to the top of the head is from $19.0cm$ to $29.1cm$ as shown in Table III [16]. Assuming that PNet of MTCNN can detect a face of which the ratio of height to width is one to one, the size of the detectable face is from $19.0cm$ to $29.1cm$.

TABLE III
HUMAN HEAD SHAPES IN THE UNITED STATES

Length	Minimum	Maximum
Head breadth	10.2 cm	17.4 cm
Menton to top of head	19.0 cm	29.1 cm

Finally, for the safety of drivers and passengers, NHTSA recommends placing themselves more than $25cm$ from the airbag. The maximum distance between the dashboard and the headrest of our driving simulator is $100cm$. Then, the range of detectable face sizes on the image is from $40pixels$ to $267pixels$ by Equation (3). In this way, the approximate range of detectable face sizes can be estimated using the vehicles indoor information.

C. Detectable Object-sizes Range Estimation based MTCNN

The MTCNN detector, as well as most face detectors, uses the pyramid image to detect objects of various sizes. Algorithm 1 is the procedure for calculating a vector of scales to make a pyramid image in [20]. In Algorithm 1, the min_layer represents a minimum value between the width and height of the smallest resized image of the pyramid image. The formula for calculating each resizing scale of pyramid image layers can be simplified, as shown in Equation (4).

Algorithm 1 Default Calculation of Scales

Require: $PNet\_kernel\_size = 12, min\_face\_size = 20, scale\_factor = 0.709$

$m \leftarrow PNet\_kernel\_size / min\_face\_size$
$min\_layer \leftarrow \min(image.width, image.height) \times m$
$scale \leftarrow empty\_list$
$factor\_count \leftarrow 0$
while $min\_layer >= PNet\_kernel\_size$ **do**
  $scale.append(m \times scale\_factor^{factor\_count})$
  $min\_layer \leftarrow min\_layer \times scale\_factor$
  $factor\_count \leftarrow factor\_count + 1$
end while
return scale

$$scale_l = \frac{PNet\_kernel\_size}{min\_face\_size} \times scale\_factor^l \tag{4}$$

In Equation (4), l means the layer number of pyramid image from 0. The image size and the detectable face size of each layer image are calculated by Equation (5). The size of the detected object is larger on the smaller image.

$$\begin{aligned} scaled\_image\_size_l &= input\_image\_size \times scale_l \\ detectable\_face\_size_l &= PNet\_kernel\_size \div scale_l \end{aligned} \tag{5}$$

Depending on the size of the images of the pyramid image layers, the time taken to search the locations of candidate faces with PNet of fixed size may vary. It is possible to estimate the calculation costs of each layer from the output area of PNet. As shown in Table IV, the output area of PNet from the first and second layers is much bigger than the others.

TABLE IV
PNET'S INPUT&OUTPUT SIZE OF EACH LAYER IMAGE IN PYRAMID IMAGE

Type		Pyramid Image (input size is 320 × 240 pixels)							
Layer Number		**0**	**1**	**2**	**3**	**4**	**5**	**6**	**7**
Scale**		0.60	0.43	0.30	0.21	0.15	0.11	0.08	0.05
Pyramid Image*	**Width**	192	137	97	69	49	35	25	18
	Height	144	103	73	52	37	26	19	13
PNet Output*	**Width**	91	64	44	30	20	13	8	4
	Height	67	47	32	21	14	8	5	2
PNet Output Area*		6097	3008	1408	630	280	104	40	8
Detectable Face Size*		20	28	40	56	79	112	157	222

* pixels, ** ratio

However, faces maybe not detected in the first and second layer image of the pyramid image when the range of face

size on the image is from $40pixels$ to $267pixels$, as mentioned in section III-B. For real-time face detection, setting a global constant such as minimum face size to get rid of the unnecessary layer is not appropriate because the face detection environment is different for each vehicle or driver. Instead, the scale range of the pyramid image is limited with minimum and maximum scales calculated Equation (6) with the estimated range of detectable face sizes. Therefore, PNet can search the faces only in face detectable layers of the pyramid image.

$$\begin{aligned} min\_scale &= \frac{PNet\_kernel\_size}{\max(\{detectable\_face\_size_l\})} \\ max\_scale &= \frac{PNet\_kernel\_size}{\min(\{detectable\_face\_size_l\})} \end{aligned} \tag{6}$$

IV. Experiments

A. Experiment Environment and Evaluation

The evaluation dataset consists of NTHU-DDD $(173,299)$ [17] and our dataset $(40,940)$, totaling $214,239$ images. The images were resized to 320×240 to unify the sizes. A laptop used in the experiment has Intel® Core™ i7-6700HQ, 8GB RAM, and NVIDIA GTX 960m, and a used camera is BlasterX Senz3D of Creative corporation. Our dataset consists of images from the camera on the dashboard of the driving simulator in Fig. 3.

Fig. 3. The driving simulator

To make facial ground-truth of the dataset, we first roughly annotated bounding boxes of drivers face using the MTCNN face detector and the KCF tracker. Points of the bounding boxes are corrected to include only the face because areas other than the face are unnecessary for recognition.

We used the range of facial ground-truth instead of Equation (3) since each dataset does not describe details such as sensor size, focal length, and working distance. For each dataset, experiments were conducted using each range of face sizes. Fig. 4 shows the distribution of face sizes for each dataset.

To evaluate DORE-MTCNN, we compared it with Haar-Cascade, LBP-Cascade, HoG-SVM, SSD, MTCNN based face detectors. To implement these detectors, we used python libraries such as Dlib, OpenCV, and MTCNN [18]–[20]. Detection bounding boxes are different depending on the characteristic of each face detector. The results of face detection are true when the area of intersections over the area of the unions, Equation (7), is over 0.6. The r_{gt} and r_{rst} mean the bounding boxes of ground-truth and detection, respectively. Accuracy is calculated by Equation (8).

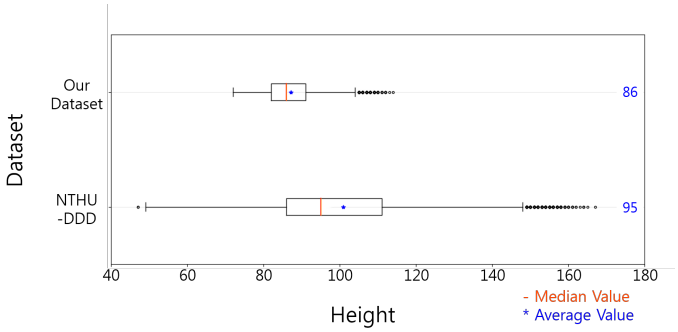

Fig. 4. The distribution of face size for each dataset

$$IoU(r_{rst}, r_{gt}) = \frac{Area(r_{rst} \bigcap r_{gt})}{Area(r_{rst} \bigcup r_{gt})} \tag{7}$$

$$Accuracy = \frac{True}{True + False} \tag{8}$$

B. Experiment Results

As shown in Table V, the MTCNN detector has the highest performance in terms of accuracy over both datasets. The LBP-cascade detector has the lowest processing time, but it has the lowest accuracy. The SSD detector relatively low processing time and has the highest accuracy in our dataset composed mostly of the frontal face. In the NTHU-DDD dataset composed largely of profile face, it has lower accuracy rather than the MTCNN detector. In profile face detection, the others except the MTCNN detector detected faces with areas other than the face and were lower accuracy rather than the MTCNN detector. In frontal face detection, the MTCNN detector has relatively high accuracy.

TABLE V
Comparison of Detection Results

Type	Processing Time millisecond (ms)		NTHU-DDD Accuracy (%)		Our dataset Accuracy (%)	
	Front	+Profile	Front	+Profile	Front	+Profile
Haar-Cascade	11.90	32.18	82.39	85.23	91.41	92.67
LBP-Cascade	**6.77**	10.51	70.93	77.02	90.58	92.35
HoG SVM	41.50		88.53		97.75	
CNN ver. MMOD	34.62		72.89		98.56	
SSD	16.27		89.07		99.46	
MTCNN	31.99		95.98		96.56	
DORE-MTCNN	15.98		95.98		96.56	

By comparing Tables I and VI, the DORE algorithm can improve the overall performance of MTCNN in terms of processing efficiency. We conducted this experiment in the

identical environment of Table I. By applying the DORE algorithm to MTCNN, the number of input images decreased. The processing time of PNet decreased by more than 50 percent. The number of candidate facial windows, which are results of PNet and are targets of non-maximum suppression and Refine Network (RNet), decreased. The processing time of not only the RNet but also the additional procedure, which occupies the most processing time of MTCNN, decreased. As a result, the ratio of face detection in the total processing time of the sequential structure deceased by 16.82 percent point.

TABLE VI
PROCESSING TIME OF DETECTION, TRACKING, AND RECOGNITION WITH DORE-MTCNN

Type		Processing time(ms)		Proportion(%)
MTCNN Detection	Proposal Network(PNet)	5.985	15.983	35.77
	Refine Network(RNet)	2.202		
	Output Network(ONet)	2.277		
	Additional Procedure	5.519		
KCF Tracker		7.027		15.73
Xception Recognition		21.671		48.50
Total		44.681*		100.0

* Frames Per Second is 22.381

Furthermore, Tables V and VI show that DORE-MTCNN not only had the same performance as MTCNN in terms of accuracy but also had relatively low processing time. That is, the PNet results of unused pyramid image layers did not affect the final results. As a result, it is reasonable to use the DORE-MTCNN detector because both frontal and profile faces can be stably detected and processed relatively quickly.

V. CONCLUSION

In this paper, we have proposed the Detectable Object-sizes Range Estimation (DORE) algorithm which restricts layers of pyramid image with specifications of camera and drivers seat. We tested DORE-MTCNN on NTHU-DDD, which is mostly composed of profile faces, and our dataset, which is mainly composed of front faces. Experimental results showed that the original MTCNN outperforms other detectors in both profile and front face detection in terms of accuracy. By using the DORE algorithm, MTCNN outperforms others, which have reliable accuracy, in terms of processing time as well as accuracy. Additionally, the processing time of face detection in the sequential structure of our driver monitoring system could decrease by 16.82 percent points.

Our proposed algorithm can also apply to other face detection algorithms that use the pyramid image. In future work, we will apply our method to other detection algorithms and experiment DORE applied detection algorithms in terms of accuracy and processing time. Additionally, our method was also tested on YawDD [21], one of the datasets used for driver monitoring. However, the accuracy of the original MTCNN was significantly low when the face was located at the top area of the image. To solve this problem, we will fine-tune and retrain the deep-learning models of MTCNN and test our method on YawDD in further research.

ACKNOWLEDGMENT

This work was supported by Electronics and Telecommunications Research Institute (ETRI) grant funded by the Korean government and Ulsan Metropolitan City. [19ZS1300, The development of smart context-awareness foundation technique for major industry acceleration] [19AS1100, Development of smart HSE system and digital cockpit system based on ICT convergence for enhanced major industry]

REFERENCES

[1] Peter N. Belhumeur, David W. Jacobs, David J. Kriegman, and Neeraj Kumar, "Localizing parts of faces using a consensus of exemplars." IEEE transactions on pattern analysis and machine intelligence 35.12 (2013): 2930-2940.
[2] Kaipeng Zhang, Zhanpeng Zhang, and Zhifeng Li, "Joint face detection and alignment using multitask cascaded convolutional networks." IEEE Signal Processing Letters 23.10 (2016): 1499-1503.
[3] Paul Viola, and Michael Jones. "Rapid object detection using a boosted cascade of simple features." Proceedings of IEEE Conference on Computer Vision and Pattern Recognition (CVPR 2001). IEEE, 2001.
[4] Timo Ahonen, Abdenour Hadid, and Matti Pietikainen. "Face description with local binary patterns: Application to face recognition." IEEE Transactions on Pattern Analysis & Machine Intelligence 12 (2006): 2037-2041.
[5] Davis E King, "Easily Create High Quality Object Detectors with Deep Learning." http://blog.dlib.net/2016/10/easily-create-high-quality-object.html
[6] Davis E King, "Max-margin object detection." arXiv preprint arXiv:1502.00046 (2015).
[7] Wei Liu, Dragomir Anguelov, Dumitru Erhan, Christian Szegedy, Scott Reed, Cheng-Yang Fu, and Alexander C. Berg, "SSD: Single shot multibox detector." European conference on computer vision. Springer, Cham, 2016.
[8] Shaoqing Ren, Kaiming He, Ross Girshick, Jian Sun. "Faster r-cnn: Towards real-time object detection with region proposal networks." Advances in neural information processing systems. 2015. 91-99.
[9] Joseph Redmon, Santosh Divvala, Ross Girshick, Ali Farhadi. "You only look once: Unified, real-time object detection." Proceedings of the IEEE conference on computer vision and pattern recognition. 2016. 779-788.
[10] Zhang, H., Wang, X., Zhu, J., and Kuo, C. C. J., Fast face detection on mobile devices by leveraging global and local facial characteristics. Signal Processing: Image Communication (2019), Volume 78, 1-8.
[11] Zeng, D., Zhao, F., Ge, S., and Shen, W., Fast cascade face detection with pyramid network. Pattern Recognition Letters (2019), Volume 119, 180-186.
[12] Zeng, D., Liu, H., Zhao, F., Ge, S., Shen, W., and Zhang, Z., Proposal pyramid networks for fast face detection. Information Sciences (2019), Volume 495, 136-149.
[13] Francois Chollet. "Xception: Deep learning with depthwise separable convolutions." Proceedings of the IEEE conference on computer vision and pattern recognition. 2017.
[14] Joao F. Henriques, Rui Caseiro, Pedro Martins, and Jorge Batista, "High-speed tracking with kernelized correlation filters." IEEE transactions on pattern analysis and machine intelligence 37.3 (2015): 583-596.
[15] The Raspberry Pi Foundation, "Camera Module." https://www.raspberrypi.org/documentation/hardware/camera/
[16] Jin-hee Lee, Su-jeong Hwang Shin, Cynthia L. Istook, "Analysis of human head shapes in the united states." International Journal of Human Ecology 7.1 (2006): 77-83.
[17] Ching-Hua Weng, Ying-Hsiu Lai, and Shang-Hong Lai. "Driver drowsiness detection via a hierarchical temporal deep belief network." Asian Conference on Computer Vision. Springer, Cham, 2016.
[18] Davis E. King, Dlib, https://github.com/davisking/dlib
[19] OpenCV team, OpenCV, https://github.com/opencv/opencv
[20] Ivn de Paz Centeno, MTCNN face detection implementation for TensorFlow, https://github.com/ipazc/mtcnn
[21] Shabnam Abtahi, Mona Omidyeganeh, Shervin Shirmohammadi, and Behnoosh Hariri, "YawDD: A yawning detection dataset." Proceedings of the 5th ACM Multimedia Systems Conference. ACM, 2014.