EasyChair Preprint

№ 8871

Eye Detection For Drivers Using Convolutional Neural Networks with Automatically Generated Ground Truth Data

Sorin Valcan and Mihail Gaianu

September 23, 2022

Eye Detection For Drivers Using Convolutional Neural Networks With Automatically Generated Ground Truth Data

1st Sorin Valcan, 2nd Mihail Găianu
Department of Computer Science
West University of Timişoara
and
Continental Automotive Romania - User Experience UX
Timişoara, Romania
sorin.valcan96@e-uvt.ro, mihail.gaianu@e-uvt.ro

***Abstract*—Eye detection is an essential feature for driver monitoring systems acting as a base functionality for other algorithms like attention or drowsiness detection. Multiple methods for eye detection exist. The machine learning based methods involve a manual labeling process in order to generate training and testing datasets. This paper presents an eye detection algorithm based on convolutional neural networks trained using automatically generated ground truth data and proves that we can train very good machine learning models using automatically generated labels. Such approach reduces the effort needed for manual labeling and data preprocessing.**

***Index Terms*—labeling automation, infrared camera, driver monitoring, eye detection, convolutional neural networks**

I. Introduction

Eye detection is the base functionality required for multiple driver monitoring features like attention or drowsiness detection. A good eye location is required to continue the pupil and eye opening analysis. The algorithms must be fast enough to be used for real time processing in real life scenarios.

For machine learning models a good quality training and testing dataset must exist. There are specific tasks where datasets exist or can be obtained automatically from the internet without too much processing required. For example, a price prediction model dataset containing the history of product prices exists by default or can be obtained in time by observing the price evolution. In this paper, the eye labels on the infrared images with drivers are not automatically available and usually it requires manual effort for marking locations of the eye.

In [1] we proposed a ground truth data generation algorithm created specific for labels on infrared driver recordings. This paper proposes the usage of such automatically generated datasests for training machine learning models in order to completely remove the manual labeling effort. When datasests with millions of frames need to be labeled, the effort for creating a labeling team and organizing it becomes long, complex and expensive. The effort needed for a few developers to implement an automatic algorithm is more feasible.

In 1960s was the start for eye detection algorithms [3]. In this section we aim to learn about some of these useful models with large learning capacity, able to handle the complexity of tasks involving problems that are too large to be fully specified in any dataset [4]. Convolutional neural networks for eye tracking is a newer algorithm that has been researched [5] in the last years. Because of how little has been done in this area, our project focuses on this eye tracking algorithm in order to add knowledge to the development of convolutional neural networks for eye tracking and gaze.

The aim of this work was to develop a convolutional neural network based eye zone estimation for automotive applications that monitors the driver during his trip from one location to another, ensuring a safer driving environment for him and other traffic participants. In the early 2000's, A. T. Duchowsky [6] points that eye detection tracking could be the basis for one of the most promising type of Human Machine Interface (HMI).

From the beginning, lots of time and effort were put into the research of eye tracking using different head-mounted systems to be able to measure the eye more accurately. However, this kind of systems are not of interest anymore for consumers in general or for the automotive industry because it is impractical to wear inaccurate and uncertain head wear. The problems generated by the head pose and orientation in regard with eye tracking were tackled using either model based or appearance based methods, e.g. the work of J. G. Wang [7] or Y. Sugano [8]. Other researches have opted to use near-infrared (NIR) cameras [9], stereo imaging [10], zoom cameras in combination with wide-angle cameras [11], or a combination of these to increase coverage, so that they will be able to allow a larger head movement.

Due to the remarkable performances of Deep Neuronal Networks (DNNs) in visual computing tasks, the deep learning based solutions for gaze estimation have gained an increased popularity. For example, S. Vora et. al. compared the performances of several convolution neural network (CNN) architectures: AlexNet, VGG16, ResNet50 and SqueezeNet in pre-

dicting 6 gaze zones plus eyes closed case [12]. A Recurrent-CNN network architecture that combines appearance, shape and temporal information for video based gaze estimation is introduces in [13]. In order to overcome the problem of head rotation, H. S. Yoon et. al. are proposing a combination of single image and dual near-infrared cameras [14].

In the last years, due to the remarkable performances of Deep Neuronal Networks (DNNs) in visual computing tasks, the deep learning based solutions gained an increased popularity [15], [16].

II. Methods

This paper presents a system that uses two convolutional neural networks for computing the eye locations on infrared driver recordings. The first neural network is used as a regression model to compute the pixel location of the bounding box that should contain the eye and the second one is a classification model used to confirm the computed bounding box is a correct eye detection.

A. Base dataset

The dataset used for training the neural networks is described in [1]. The labels are automatically computed on infrared driver recordings with the resolution of 1280x800. This dataset has labels for bounding boxes that must contain the eye inside it.

Because of the automatic process for labels generation that is not perfect, the dataset contains less than 1% of frames where eyes are marked incorrect. There are more than 250.000 frames with eye labels. For the neural network training, subsets for training and testing are generated using the automatic preprocessing system described in [2]. This system reduces the chances of encountering the wrong labels in the generated subsets.

B. Eye detection method

The proposed method for eye detection is a system of two convolutional neural networks as described in Figure 1. The first one has the purpose to output 8 values representing two bounding boxes which should correspond to the left and right eye. The seconds one should classify each eye patch to check if it is indeed an eye.

The main reason for the eye patch classifier is to remove the false positive detections on frames where the driver is not behind the steering wheel. The eye location neural network will provide an output for any given image and it has not been trained to directly return invalid values for frames where eyes are not present.

C. Eye location neural network

The eye location is a convolutional neural network regression model that estimates two bounding boxes where the eyes should be present. The architecture of the network is described in Figure 2.

In the first part of the preprocessing we execute four consecutive average pooling layers in order to reduce the input image from 1280x800 to 80x50. This drastic downsizing has been used because of the similar performance in terms of precision compared to networks where we start to run convolutional layers directly on the input image. The big advantage is the reduction of time needed to process one frame, mainly because executing average pooling on big feature maps is much faster that the convolution operations.

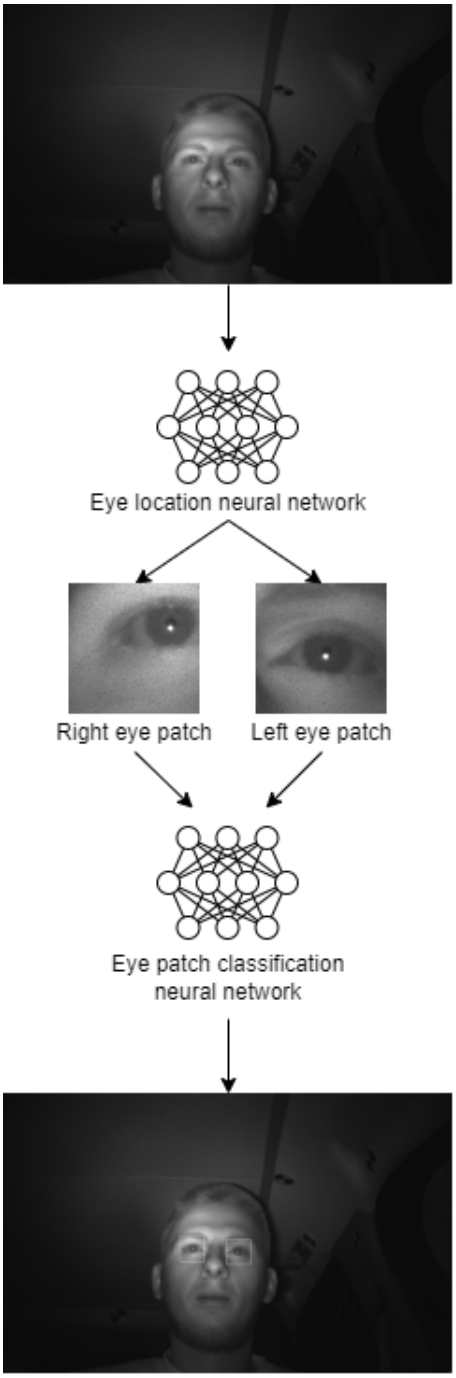

Fig. 1. Neural network system for eye detection

After the downsizing we have 2 series of convolutional, parametric relu and max pooling layers used for the feature

extraction. In the convolutional layers, we used 3x3 kernels initialized with random values and a stride of 1 used for traversing the input feature maps. We also use gradients clipping factor of 0.3 which adds more stability to the training process and makes the model converge to a solution much faster. The clipping of the gradients is also necessary because we don't normalize the input values from the image which makes the initial phase of the training very unstable and the model has problems in converging to a good solution.

After obtaining the last feature maps resulted from the preprocessing of the image, we use one hidded layer with 500 neurons and L2 regularization with lambda parameter of 0.1 to avoid overfitting. The gradients clipping factor of 0.3 is used also for the hidden layer. All values from this layer are activated using the parametric relu activation with an alpha factor of 0.1. The second hidden layer will compute the 8 output values that represent the output of the neural network. For the second hidden layer the L2 regularization factor has been removed.

For the output layer we use the mean squred error loss function to start computing gradients in the backpropagation process.

We used also variantions for the architecture presented above, with less average pooling and more convolutional layers and also tried to add one more hidded layer. The results were very similar in terms of precision, which resulted in the simplest and smallest version of the architecture, in order to keep the computing performance as good as possible.

In the training of the neural network we used subsets of data generated by the automatic system presented in [2]. We generated subsets of training data of approxiamtely 12.000 frames and 2.000 frames for testing. These number may vary slightly depending on the availability of frames for every specific subject. Increasing the number of samples in the training subset has no positive influence on the precision of the resulted model. For each test and subset generated, we keep the subjects in the training dataset separate from the ones used for testing the precision of the model so there is no chance to have a network learning a specific person and generating misleading good performance.

D. Eye patch classification neural network

The input for the eye patch classification network is one eye patch defined by the bounding boxes from the eye location network. This network is used to make sure that both bounding boxes are computed correct for the eye locations. The architecture of the network is defined in Figure 3.

In the preprocessing part of the neural network there are two series of convolutional, parametric relu and max pooling layers used for feature extraction. The parameters of these layers are the same as for the previous network. The preprocessing results in a flatten array of 1024 inputs for the hidden layer.

The first hidden layer has 100 neurons and the same L2 and gradients clipping parameters like in the previous network. The parametric relu activation is also present here. The second hidden layer will output 2 neurons representing the classification of the eye patch. The softmax activation function is used to normalize the outputs of the network before the mean squared error loss function starts to compute the gradients in the training phase.

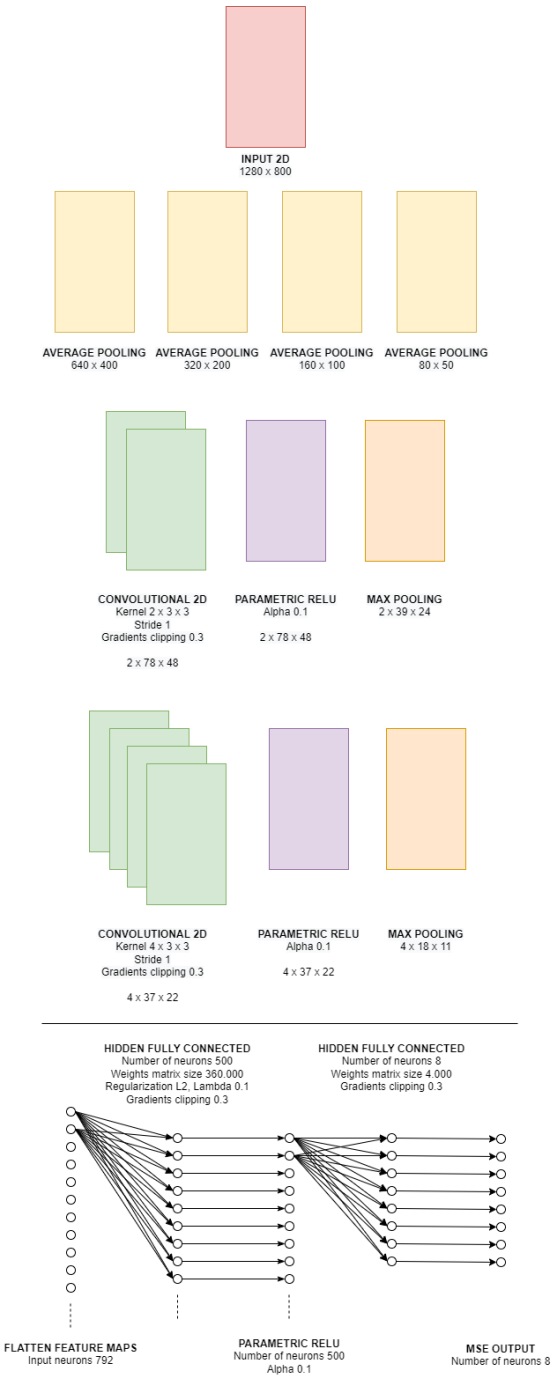

Fig. 2. Convolutional neural network for eye location

The dataset used for training this classification neural network is generated based on the eye labels from [1]. Using these location, we used the eye patches from the generator as positive samples that should be classified by the network as

valid. We also take the original bounding boxes and shift them in all directions by maximum 20% of the patch size in order to make the valid patches more varied. A shift of 20% is small enough to keep the eye inside the bounding box and use it as a valid training sample.

To generate invalid eye patches we move the base bounding box from the existing valid labels in all directions by 60% of the patch size. Random locations of the moved patches are selected. We also generate some invalid eye patches from random locations for frames where the eye labels were not detected in order to have various invalid cases available(e.g. when driver is not behind the steering wheel, has the head rotated or when the eyes should have been detected but the generator missed that frame).

Fig. 3. Convolutional neural network for eye patch classification

Because of the big variety especially for the invalid eye patches generated, we use approximately 40.000 patches for training subsets and 6.000 frames for testing.

E. Eye patch searching process

In order to improve the number of detections from the neural network system described above, we use the eye patch classification network with a search mechanism around the bounding box locations given by the regression network.

This system is used because the location of the bounding boxes for multiple subjects have a constant shift in one direction which makes the classification of the patches to become invalid. To solve this problem, we do a search around the given patch location, and and run the classification network on each one.

The search is characterized by a percent defining how much around the input location the program should iterate and a stride used for traversing the entire window. All locations for the patches that are classified as valid by the second neural network are saved, and the median value for both x and y coordinate will be the resulted location of the eye.

Another safety feature is to use the median value only if we have at least two possible patches classified as valid. This helps avoiding some false positive outputs given by the classifier for various random patches.

This process is performed separately for each eye. We output the location only if both eyes were found using the search described above.

III. Obtained results

In this section we will present the results obtained for each individual component of the system and for the system as a whole.

The KPI we consider here are the same as in [2]. The focus in on the overlap between the detected bounding boxes resulted from the neural networks and the bounding boxes from the base dataset generated with [1]. The definition of a correct bounding box is to have the eye contained in it.

For the eye location regression neural network the average overlap obtained on multiple tests is maximum 65%. This percent is the result of the processing for multiple testing subjects. For some of the subjects the overlap percentage goes up to 90% but for other it can be as low as 10%. This means for some subjects the overlap is too small to have the eye contained within the generated bounding box. If we consider a correct bounding boxed moved with 20% of the patch size for both x and y coordinate that remains a valid detection, we would have an overlap of 64% between the two. For the resolution of 1280x800 and the eye patch size of 70x70 from the base dataset, it means a detection shifted with 14 pixels from the original ground truth bounding box is still valid. This is not necessarily true for all bounding boxes, because if one of them is already at a limit and gets shifted even more it may not contain the eye anymore.

Considering all these, the eye location neural network is used to get an approximate location of the eyes on the image, in some cases with a very good precision but for some subjects with the bounding box close to the eye or containing the eye only partially. The network also outputs two bounding boxes for any image, even if the driver is not behind the steering

wheel or the eyes are not in the image because of various movements of the head.

The eye patch classification neural network testing precision is usually around 99%, with the worst cases around 96%. Performance is mostly influenced by the subjects with glasses in the testing data. The types of glasses in the base recordings dataset are very diverse, with various frame patterns for each individual subject. The reflections from the frame makes the neural network to generate false positive detections.

The two neural networks with their individual performance described above work together in a system like in Figure 1. There are further analysis we can extract to understand the performance of the entire eye detection system.

Using the second neural network with the searching mechanism for eye patches from II-E we obtain an average overlap on all the recordings with the testing subjects of 80%. The main problem with the glasses for the eye patch classification network has an important influence on the KPI. If we remove from the resulted KPI the recordings with glasses, the average overlap between the ground truth labels and the system detections goes up to 87%. Used together as a system, we have major overlap improvement compared to the single eye location neural network that was 65%.

Another important analysis is the availability of the eye dections from the neural networks system compared to the availability from the ground truth dataset. The ground truth data generator described [1] is a very precise but not necessarily a constant eye detection algorithm. It's purpose is to generate correct labels even with the risk of missing multiple frames from a processed recording.

The availability of detections from the neural network system has increased significantly compared to the automatic ground truth data. An analysis for 90 recordings with testing subjects never seen by the neural networks in the training phase is presented in Figure 4.

For almost all recordings there are more detections from the neural networks compared to the dataset that was used to train them. Except for the glasses recordings where detections can jump from the eyes to the frame and additional to the average overlap KPI that is computed for the common frames between the ground truth and the neural network detections we can estimate that eyes are located inside the computed bounding boxes in more than 93% of the cases.

The system works well on recordings with drivers from different ethnicities and is very flexible, not being affected by different heights in the seating position. The neural network outputs correct eye locations for various scenarios especially when the driver gets in the car and very unusual head positions occur. There are also multiple recordings with drivers wearing surgical masks. In these cases the eye detection works very well.

IV. Future work

The future work of this project will first focus on detection of mouth and nostrils. The procedure will remain the same, but there will be some small adjustments. First we will update

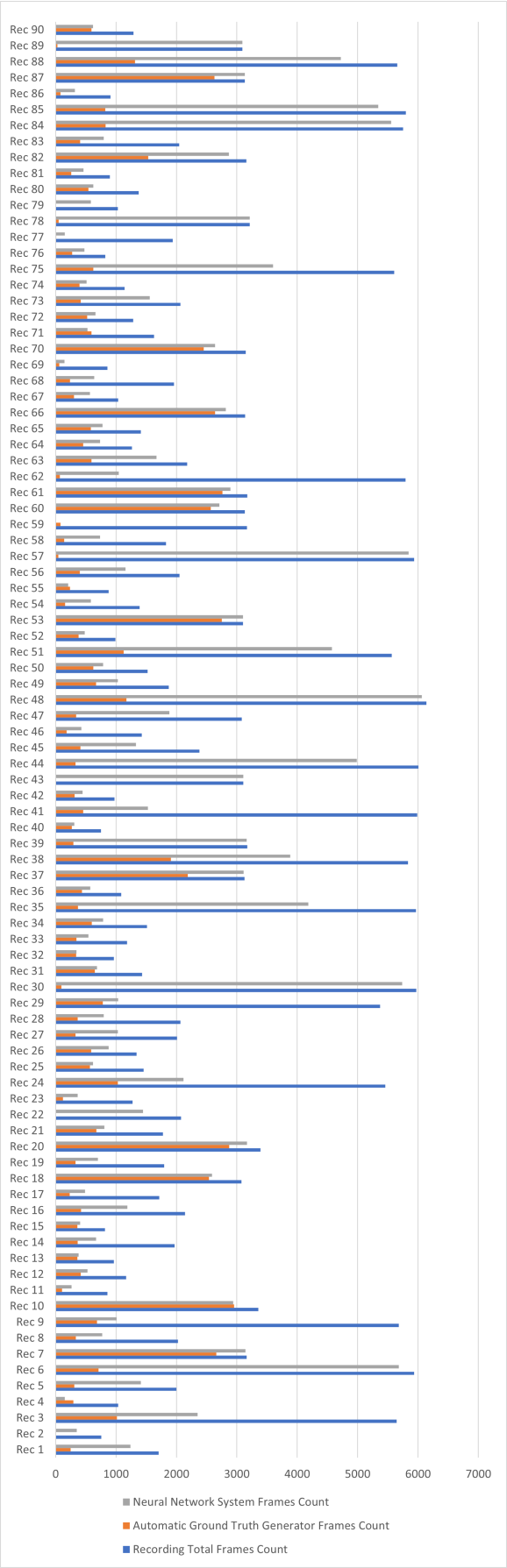

Fig. 4. Number of eye detections comparation between the automatic ground truth generator and the neural network system

the automatic ground truth data generator to include a mouth and nostrils detection algorithm and the resulted labels will be used to train neural networks.

The neural network architecture from Figure 2 will probably be updated to include eight additional output values, representing two new bounding boxes. Once the regression neural network is functional, a new classification neural network for mouth and nostril confirmation will be trained in similar way with the one for eyes.

In this point we expect to prove that we can train a complete face feature detection neural network system using only automatically generated ground truth data. Once this is achieved, we can continue with other driver monitoring algorithms for pupil and eye opening analysis used in attention or drowsiness detection.

As possible further improvements and research directions we could mention:

- Dealing with occlusions and images that contain only partial data by employing better face detectors.
- Speeding up inference time using ML accelerators.
- Experiments on more near infrared higher resolution.

V. Conclusions

This paper presented the improvements obtained using a convolutional neural networks system trained with automatic ground truth data. Eye tracking is one of the key technologies for future driver assistance systems since human eyes contain much information about the driver's condition such as gaze, attention level, fatigue level and drowsiness. One problem common to many eye tracking methods proposed so far is their sensitivity to lighting condition change. This tends to significantly limit their scope for automotive applications. This paper describes eye detection that works under variable and realistic lighting conditions. It is based on a hardware system for the real-time acquisition of a driver's images using IR illuminator and the software implementation for monitoring eye that can avoid the accidents.

Without any human labeling effort and using inconstant ground truth data, we managed to train convolutional neural networks that offer a very good precision and improved availability for eyes location than the data used to train them.

The main advantage is the automatic nature of this system which allows development to go on and remove all the human effort needed for labeling and selection of the data.

VI. Acknowledgment

The authors are thankful to Mr. Cosmin Moisa from User Experience @ Continental Automotive Timişoara for fruitful discussions and constructive suggestions.

References

[1] Valcan, S.; Gaianu, M. Ground Truth Data Generator for Eye Location on Infrared Driver Recordings. J. Imaging 2021, 7, 162. https://doi.org/10.3390/jimaging7090162.

[2] S. Valcan, "Convolutional Neural Network Training System For Eye Location On Infrared Driver Recordings Using Automatically Generated Ground Truth Data," 2021 23rd International Symposium on Symbolic and Numeric Algorithms for Scientific Computing (SYNASC), 2021, pp. 222-226, doi: 10.1109/SYNASC54541.2021.00045.

[3] C. Kleinke. Gaze and eye contact: A research review, 1986.

[4] Krizhevsky, Sutskever, and Hinton. Imagenet classification with deep convolutional neural networks. International Conference on Neural Information Processing Systems, 2012.

[5] V. Sitzmann, A. Serrano, A. Pavel, M. Agrawala, D. Gutierrez, B. Masia, and G. Wetzstein. How do people explore virtual environments. IEEE Transactions on Visualization and Computer Graphics, 2017.

[6] A. T. Duchowski, "A breath-first survey of eye-tracking applications", Behavior Research Methods Instruments Computers, vol. 34, no. 4, pp. 455-470, 2002.

[7] J. G. Wang and E. Sung, "Study on eye gaze estimation", IEEE Trans. Syst. Man and Cybernetics – Part B, vol. 32, no. 3, pp. 332-350, 2002.

[8] Y. Sugano, Y. Matsushita and Y. Sato, "Appearance-Based Gaze Estimation Using Visual Saliency", IEEE Transactions on Pattern Analysis and Machine Intelligence, vol. 35, no. 2, pp. 329-341, Feb. 2013.

[9] J. Wu, W. Ou and C. Fan, "NIR-based gaze tracking with fast pupil ellipse fitting for real-time wearable eye trackers", 2017 IEEE Conference on Dependable and Secure Computing, pp. 93-97, 2017.

[10] S. W. Shih and J. Liu, "A novel approach to 3-D gaze tracking using stereo cameras", IEEE Trans. Syst. Man and Cybernetics – Part B, vol. 34, no. 1, pp. 234-245, 2012.

[11] D. H. Yoo and M. J. Chung, "A novel non-intrusive eye gaze estimation using cross-ratio under large head motion", Computer Vision and Image Understanding, vol. 98, no. 1, pp. 25-51, Apr. 2005.

[12] S. Vora, A. Rangesh and M. M. Trivedi, "Driver Gaze Zone Estimation Using Convolutional Neural Networks: A General Framework and Ablative Analysis", IEEE Transactions on Intelligent Vehicles, vol. 3, no. 3, pp. 254-265, Sept. 2018.

[13] C. Palermo, J. Selva, M. A. Bagheri and S. Escalera, "Recurrent CNN for 3D Gaze Estimation using Appearance and Shape Cues", British Machine Vision Conference, 2018.

[14] H. S. Yoon, N. R. Baek, N. Q. Truong and K. R. Park, "Driver Gaze Detection Based on Deep Residual Networks Using the Combined Single Image of Dual Near-Infrared Cameras", IEEE Access, vol. 7, pp. 93448-93461, 2019.

[15] R. Mîrșu, G. Simion, C.-D. Căleanu and I.M. Pop-Calimanu, "A PointNet-Based Solution for 3D Hand Gesture Recognition", Sensors, vol. 20, no. 11, 2020, [online] Available: https://www.mdpi.com/1424-8220/20/11/3226.

[16] R. Mîrșu, G. Simion, C.-D. Căleanu and O. Ursulescu, "Deep Neural Networks vs Bag of Features for Hand Gesture Recognition", Telecommunications and Signal Processing (TSP), July 1-3, 2019, [online] Available: https://doi.org/10.1109/TSP.2019.8768812.