Enhancing Attention through the Eye Tracking

Ondrej Kainz
Department of Computers and Informatics
Technical University of Kosice
Kosice, Slovakia
ondrej.kainz@tuke.sk

Marek Gera
Department of Computers and Informatics
Technical University of Kosice
Kosice, Slovakia
marek.gera@student.tuke.sk

Rastislav Petija
Department of Computers and Informatics
Technical University of Kosice
Kosice, Slovakia
rastislav.petija@tuke.sk

Miroslav Michalko
Department of Computers and Informatics
Technical University of Kosice
Kosice, Slovakia
miroslav.michalko@tuke.sk

Dávid Cymbalák
Slovak Centre of Scientific and Technical Information
Bratislava, Slovakia
david.cymbalak@cvtisr.sk

Frantisek Jakab
Department of Computers and Informatics
Technical University of Kosice
Kosice, Slovakia
frantisek.jakab@tuke.sk

Roman Vápeník
Slovak Centre of Scientific and Technical Information
Bratislava, Slovakia
roman.vapenik@cvtisr.sk

2020 18th International Conference on Emerging eLearning Technologies and Applications (ICETA) | 978-1-6654-2226-0/20/$31.00 ©2020 IEEE | DOI: 10.1109/ICETA51985.2020.9379229

***Abstract*—The contribution describes an implementation of a system capable of measuring and possibly improving the human attention through the use of eye-tracking techniques. The used techniques achieve high accuracy for non-head mounted cameras with low resolution in visible spectrum. In order to achieve this, an analysis of existing methods was carried out. The implementation of the system is created through the utilization of Python-compatible library for face detection as well as using Timm and Barth algorithm for pupil detection. The main focus of the testing phase is the accuracy of the face detection and pupil detection algorithms in various lighting conditions as well as under different view angles. Additional tests are performed on participants with different eye-colors. The experimental solution may be used also for tracking of the attention of students or in various educational games.**

Keywords—attention, dlib, eye detection, eye tracking, Python, Timm and Barth, webcamera

I. INTRODUCTION

The use of computer algorithms for object detection has been a highly targeted goal for many decades. In the recent years the importance of such methods increased even more drastically. This has happened due to the fact that people acquired easier access to hardware capable of efficiently utilizing the benefits of said algorithms. Whether we take into consideration the ever-improving cameras, mobile phones and speed of calculating abilities of hardware, all of these have wildly contributed to taking object detection methods seriously in everyday life. Some of the most influential methods, and those which this contribution focuses on, are the methods of eye tracking and detection.

A great importance of eye tracking in the modern world is apparent for many reasons. Examples of this may be seen in the use of eye-tracking techniques for medical purposes, e.g. in order to detect and measure patient's attention, reaction times, tiredness and etc. In 2018, Junli Xu et al. [1] have used head-mounted eye-tracking set to measure driver's fatigue. Systems like these are not new, and they have been researched and developed widely in the last few years and decades. However, these techniques do not have to be applied only to car driving. The analysis from 2014 by T. Blascheck et. al [2] describes how the eye tracking data can be visualized and thus used for various purposes of attention measuring and more.

The importance of discussing the issue of measuring and improving attention through the use of eye tracking techniques stems from several points. The first is that in commercially and medically used systems, the eye tracking is mostly based on head mounted cameras. The obvious advantage of such systems is accuracy since the camera is much closer to the eye. Moreover, in most of the cases it allows the user free head rotation which further simplifies the use of such system. This methodology comes with setbacks however, with most of the systems requiring special hardware and software which may prove to be expensive and difficult to acquire. In this contribution we offer a simple and highly inexpensive alternative capable of working with non-head mounted cameras. This includes low resolution web cameras working fully in the visible spectrum. Furthermore, we briefly touch on the problematic of face detection and facial landmark detection which are a necessary requirement for an accurate eye locating and tracking. Outputs of the research may be used in the novel IoT solutions [3], [4].

II. FACIAL LANDMARK DETECTION

The first necessary step before the use of eye detection algorithms involves the acquisition of an image frame containing the eye on which the detection is performed. The reason for this mainly exists due to the way eye detection algorithms recognize a pupil or the center of an eye. In general, the algorithms rely on a comparison between the pupil and the surrounding area, e.g. by attempting to find the roundest or the darkest part of the image. Therefore, the less clutter a frame contains the more accurate eye detection algorithms become. The reduction in frame size generally also means that there is less data for an algorithm to consider and process. The most reliable way to achieve this is through the use of full-face detection. This process assigns so-called facial landmarks (see Fig. 1) to specific points in the image frame, which usually correspond to some of the most prominent features of the human face, e.g. the tip of the nose, the leftmost and the rightmost point of the eye, the chin outline, etc. These can be used as reference points to accurately bound the area containing a human eye. Some of the most widely used methods are AAM (Active Appearance Models) and CLM (Constrained Local Methods) surveyed by Wu et al. [5].

978-0-7381-2267-7/20/$31.00 ©2020 IEEE

A. Active appearance methods

There have been many attempts to create fast and accurate face detectors. Among the earliest examples may be one of the most famous and influential methods using eigenfaces from 1991 paper of Turk et al. [6]. However, this method and many other methods of that period lacked sufficient robustness to work under different lighting conditions or head poses and angles. In response to this need, models like AAM were designed which widely utilize machine learning methods. Machine learning algorithms can be taught how to detect or recognize a face by analyzing a quantity of images containing faces. This is almost exclusively performed on grayscale versions of the original frames, which greatly reduces the dimensionality of data and speeds up the learning process. Through the analysis of these images, AAM acquires two models. The first one is the model of face shape variation, which simply predicts the general shape of a human face. The second one is the model of the appearance variation. This model is acquired by normalizing the shape of a face and subsequently measuring the global pixel intensity. The fact that both models are measured globally is what sets AAM apart from most of the other methods. Using both models, appearance model and shape model, it is possible to use the gathered shape and appearance of an image section to calculate the likelihood of a face being contained within it. This is done by simply comparing the similarity of the trained models to the models gathered during face detection.

B. Constrained Local Methods

Reference [7] describes the nature of CLM. The main significant way they differ from their AAM counterparts is the use of local rather than global measurement of the pixel intensities. In short, this means that CLMs calculate the model of the face shape variation as a collection of pixel intensities, each of them belonging to a particular facial landmark. Rather than acquiring one global appearance model, CLM allows to measure each part of the face separately. This greatly increases the robustness of detection under various lighting, poses, angles etc. The most widely used implementation of these methods involves the use of SVM (Support vector model) presented by C. Cortes and V. Vapnik [7] in 1995, which is a highly efficient and famous machine learning model for group classification. SVM is generally coupled with the use of HOG (Histogram of oriented gradients) presented by N. Dalal and B. Triggs [8] in 2005, serving as a filter improving the algorithm efficiency. In short, it uses gradients of intensity gathered from the image frame. The gradients are represented in the form of vectors describing the flow of intensity in a certain direction. The image is divided into several smaller regions. A gradient with a particular direction can be assigned to each of the regions to create a complete histogram usable by the learning algorithm. The use of CLMs, especially models using SVM and HOG, have been very efficient and popular in the recent years. They are also present in the model of choice for this contribution.

III. Pupil detection

There is another crucial step after having located eyes in an image frame. It is the ability to detect the eye pupils, hence being able to use them to predict the gaze direction. The important requirements we have set for ourselves are the ability of algorithms to work with non-head mounted and low-resolution cameras. For this reason, we have compared some of the most promising algorithms.

A. Ellipse selection algorithm

One of the most successful algorithms for pupil detection is ElSe (Ellipse Selection) [9]. This method uses an edge filter to detect ellipsoid edges in an image. Based on the idea that the color of the center of the eye should be visibly distinguishable from the rest of the eye, the algorithm is capable of finding the roundest features in the image. As expected, a problem arises due to the fact that the pupil is generally not the only round object in a frame. Algorithm therefore searches the locations where several ellipses cross and proceeds to remove the pixels at the cross-section. The lines with a several pixel widths are reduced to 1 pixel width. Afterwards, the least round lines are removed, only keeping the roundest objects. To do this, a threshold value of accepted roundness is used. If the result of the filtering is one remaining round object, it is assumed to be the pupil. Otherwise the algorithm enters the second phase which is guaranteed to return the most pupil-like position, even if no pupil is contained within the image. This is done through the use of convolution for surface difference filtering and the use of median filter.

ElSe is a highly promising method of pupil detection. It has been shown [10] to provide a high level of accuracy under a wide variety of conditions. However, the method has been mainly developed for head mounted cameras. One of the downsides of the method is that the use of a non-head mounted camera generally causes the main algorithm phase to fail. This means that the algorithm is not only wasting computing potential but is forced to use the less efficient secondary method of detection. Furthermore, the visibility of a pupil in higher distances from the camera may prevent it from appearing round or being distinguished enough.

B. Timm and Barth algorithm

Timm and Barth algorithm [11] is based on the use of gradients of color intensity. This is possible due to the fact that there is generally a strong difference in color intensity between the center of the eye and the surrounding area. This allows the algorithm to locate the center of the eye by either finding the pupil or the center of the entire inner area of the eye including iris. The center of the eye is found as the roundest dark object in an image. For increased accuracy, this algorithm also includes the use of Gaussian filter. The big advantage of this method in comparison to ElSe is that the algorithm is specifically designed for the use by non-head mounted cameras. Unlike ElSe, Timm and Barth algorithm [11] does not require a second phase as it is guaranteed to always find the darkest roundest object on the screen. This makes the algorithm highly efficient. Moreover, its robustness [10] under different lighting conditions as well as head rotations provide the reason for the use of the algorithm in this contribution.

IV. Experimental Solution for Eye Tracking

The implementation of the system has been designed in the Python programming language and has been specifically tailored for Windows operating system. The system relies on the use of dlib library for constrained local face detection based on HOG [8] and SVM [7]. This is followed by pupil detection using the Timm and Barth algorithm [11]. Using this data, it is possible to calibrate the system for the use on a computer screen which allows the system to predict the specific gaze location, or gaze point, of a user. This can in turn

978-0-7381-2267-7/20/$31.00 ©2020 IEEE

be used to compare the gaze point location to the expected gaze point location and measure the accuracy of a user's gaze.

A. Implementation of the facial landmark detection

The face detection is implemented as a part of the main detection cycle, which is continuing to process new frames from the camera for as long as the calibration or the attention enhancement game is active. This was achieved through the use of OpenCV library video capture as well as frontal face detector contained in dlib library. In order to acquire the facial landmark positions, a 68-point predictor has been utilized. This, in short, means that it assigns 68 different facial landmark points to a single face in an image, as seen in Fig. 2. However, the detector is capable of finding multiple faces in an image, therefore only one of the faces is selected and processed. The landmarks are stored as an array of points, which allows a program to use them as reference points in order to only select the necessary part of the image. This is especially important for the next stages of pupil detection, where the correct area containing eyes must be found. The predictor presupposes a mostly frontal view of the user's face through the camera.

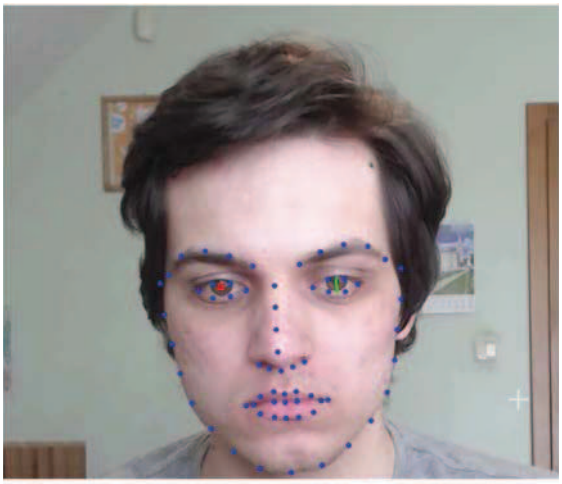

Fig. 1. Example of landmark assignment in blue color.

B. Implementation of the pupil detection

As previously mentioned, the pupil detection algorithm of choice for this contribution is the Timm and Barth algorithm [11]. It uses gradients of color intensity to differentiate between dark and round objects, for example the pupil, and the surrounding eye area. In order to make sure that the detected point is indeed the pupil, an eye region is first selected using the landmark points around the eye as a reference (see Fig. 3). The region is best selected based on the leftmost and the rightmost points of the eyes. Vertically the topmost and the bottommost points may be used. However, for the future calibration and increased accuracy it is more efficient to select the area of an eye in a form of a square. The height of the square is then equal to the width of the selected square. This is because the upper and lower eyelids may not always be detected perfectly, and a significant part of the eye may be covered. In order to mitigate this it is therefore highly beneficial to increase the vertical size of the frame containing the eye, in spite of the fact that it may contribute to a slightly slower algorithm performance due to a larger computational area.

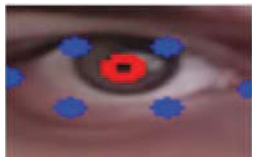

Fig. 2. Selected eye region with landmarks (blue) and the pupil (red).

The implementation of Timm and Barth algorithm [11] is contingent upon several input values which affect its speed and accuracy. The values of these parameters are closer analyzed in the results and testing section. In short, the values are:

- accuracy,
- sigma value of Gauss filter,
- alpha value of contrast.

Accuracy specifies the amount of image pixels processed by the detector, e.g. 1 for every pixel, 2 for every second pixel etc. The higher values cause the algorithm to run faster at a certain cost in precision. Sigma is the value of the Gauss filter specifying the amount of blur applied to an image frame in pixels. This value should be as low as possible in order not to overly blur the image. The recommended value is 2. Finally, the alpha specifies the level of contrast applied to the image. This can be advantageous as the pupil detection algorithm relies on finding the darkest objects. The step of adding contrast causes the darkest parts of the image to distinctly stand out from the lighter parts. However, the value chosen was 1, in other words no applied contrast. The explanation of this is further touched on in the results and testing section.

C. Direction offset calculation

After the detection of the face landmarks and subsequent detection of the eye pupils, it is necessary to select an efficient way to calculate the gaze direction of the user. This can be calculated in the form of offsets relative to specific facial landmark points. The raw data acquired can then be taken into consideration during calibration and eye tracking. In order to calculate the non-calibrated gaze direction, several parts are taken into consideration:

- blink detection,
- horizontal gaze offset,
- vertical gaze offset,
- offset combination from both eyes.

The detection of blinking is an important part of the system. The eye detection algorithm is always guaranteed to find the most pupil-like object in the image frame. This is also true when the pupil is not present, i.e. it is covered by the eyelids or eyelashes etc. Thus, we define value R, which is a ratio of eye width W and eye height H (1):

$$R = W / H \tag{1}$$

The higher the ratio R the higher the chance of the eye being closed or almost closed. If a blink is detected, the system takes it into account. Furthermore, gaze offsets are computed. The horizontal offset Kx (2) is calculated as the horizontal distance of the pupil Zx from the leftmost point of the eye Lx in relation to the eye width W:

978-0-7381-2267-7/20/$31.00 ©2020 IEEE

$$Kx = (Zx - Lx) / W \quad (2)$$

Lastly, the vertical offset *Ky* (3) is calculated as the vertical distance of the pupil *Zy* from the eyebrow midpoint *Oy* in respect to the eye width. The eyebrow midpoint is the midpoint of the leftmost point of the left eyebrow and the rightmost point of the right eye. Using this as a reference point, the system retains accuracy even under slight head rotation. However, it requires the combination of the data from both eyes in order to calculate the offset properly.

$$Ky = (Zy - Oy) / W \quad (3)$$

The reason for using eye width in the calculation stems from the need to keep the consistency of calculation even under different distances from the camera. This naturally comes at a cost of reduced accuracy due to providing lower quality input for the camera. In general, this step makes the vertical offset calculation more stable.

D. Calibration

The calibration consists of displaying calibration points on a monitor. The user must then proceed to look at the currently displayed point for as long as it is shown. The values of horizontal and vertical offsets are recorded which are then used to normalize the offset for that particular screen. There are many different ways calibration points can be displayed. The implementation of this contribution uses 5 different calibration points. One in the middle of the screen and 4 for each vertical and horizontal direction. An example of this directly from the program UI can be seen in Fig. 4. The values are then translated into specific pixels on the screen when evaluating the accuracy. Horizontally (4), this can be calculated using the distance of calibrated center *Xc* from the current non-calibrated offset *Kx*, in respect to the calibrated maximum *Xmax*. When looking to the right, the value of *Xmax* corresponds to the lowest calibrated horizontal value and vice versa.

$$KoefX = 0.5 + (Xc - Kx) / (2 * \text{abs}(Xc) - Xmax) \quad (4)$$

The same method can be applied to the vertical offset calculation (5) by replacing the horizontal values with the vertical values. These values are the vertical calibration center *Yc*, the non-calibrated vertical offset *Ky* and finally the calibrated maximum *Ymax*. This value is contingent upon the direction of gaze as well.

$$KoefY = 0.5 + (Yc - Ky) / (2 * \text{abs}(Yc) - Ymax) \quad (5)$$

To improve the accuracy under standard use conditions it may be beneficial to use the same *Ymax* value for horizontal and vertical gaze direction. This is due to the fact that vertical gaze direction generally experiences less change as users generally move their head less in the vertical direction compared to the horizontal direction. The implementation therefore averages the *Ymax* values for both vertical directions and uses them interchangeably.

E. Game for attention measuring and enhancement

Using the implemented system for eye tracking, a program for attention measuring and enhancement is created. The program consists of several main parts:

- calibration
- the attention enhancement game
- measured data result
- the ability to display previous data

The calibration consists of 5 calibration points using the method previously described. The points are being displayed after each other. After the last point is calibrated, the system allows the user to run the game for attention improvement. During the game, a point is displayed at a random screen location. The point changes its location after several seconds and the user is tasked with directing their attention onto it for as long as the game is running. Meanwhile, the user is rated based on time spent looking at the target location and the time spent looking elsewhere. The user can stop the game at any point in time after which a result screen appears with the previously mentioned metrics. The system provides the option to save data. These can be later viewed or deleted from the save file.

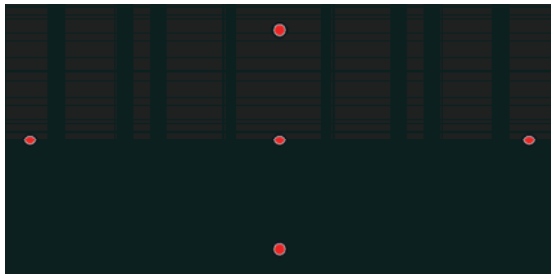

Fig. 3. Calibration using 5 points.

V. Results

The implemented system was tested under standard and non-standard conditions which may occur during its use. The tests included environments with various levels of lighting, several test subjects with different eye colors, under several head positions and rotations. The camera used for this testing was Logitech C930e with 1080p recording quality. Various resolutions were tested in order to select the most robust and accurate settings for a wide variety of recording hardware. This was specifically focused on low cost web cameras working in visible spectrum.

A. Face detection testing

The accuracy of face detection algorithm was tested under several different angles, ranging from direct camera facing to atypical angles which are not expected to occur during the system use. The accuracy has proven to be highly acceptable with the predicted facial landmarks accurately placed onto the respective facial features. This result can be seen with the face directly aligned to the camera as well as when leaning backwards. Fig. 5 shows this with the directly aligned face on the left and the leaning face on the right.

Fig. 4. Facial detection - direct facing (left), backward leaning (right)

978-0-7381-2267-7/20/$31.00 ©2020 IEEE

Similar results can be seen with face angled to the side. Although such angles are not recommended for accurate eye tracking, the face detection stayed mostly unaffected. A visible inaccuracy of the detection was only seen after angling the face steeply downwards (Fig. 6), thus causing the algorithm to have difficulties with distinguishing the neck from the rest of the face. This is mostly a non-issue as these angles are not to be expected in standard use of the system. The detection has furthermore proved to be highly robust during most of the lighting conditions including daylight and environments with under artificial light sources.

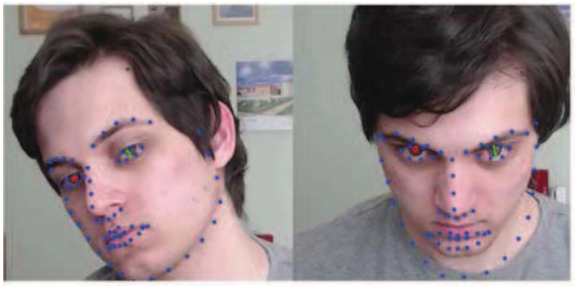

Fig. 5. Facial detection - side lean (left), frontal lean (right)

B. Pupil detection testing

The correct pupil detection is one of the most crucial requirements for accurate eye tracking. As such, testing consisted of various lighting conditions and eye colors. The system has proven to be highly robust when provided with non-extreme light conditions. One of the leading issues with incorrect light conditions was the appearance of light reflection on the eye. Since the eye detection algorithm relies on finding the darkest and roundest objects in an image, such a reflection on the pupil or the iris may cause the algorithm to slightly misplace the predicted pupil location. The algorithm is therefore accurate for as long as no significant eye reflection is caused. This is especially important for dark-eyed users whose pupil and iris color may be similar. An example of a misplacement can be seen in Fig. 7 in the bottom right frame.

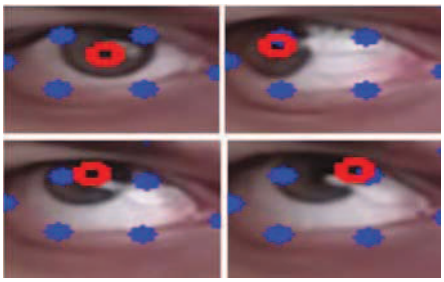

Fig. 6. Pupil detection under various angles.

Timm and Barth algorithm [11] has proved to be highly accurate especially for the users with light eye color. This is due to the fact that the pupil is visibly darker than the rest of the eye and the detection is easier.

The algorithm is surprisingly accurate even with nearly closed eyes as long no eyelashes or other objects block the view of the pupil. However, the accuracy of the eye tracking algorithm cannot be guaranteed under these conditions. Hence the algorithm uses width to height ratio of the eye in order to detect a blink. The results have shown value 5.7 as good threshold for a blink detection. This means that if the ratio is 5.7 or higher, a blink has been detected and the system takes blinking into account. Direct camera view onto face is recommended, especially for reducing the negative effect on accuracy caused by eyelashes covering parts of the eyes.

C. The variables for the detection system

Several combinations of variables were tested in the way they affect precision. These values are accuracy, frame resolution and alpha value of the contrast filter. Each of them were assigned a dispersion score which is based on the horizontal offset value defined in this contribution. It is therefore providing a relative comparison to the accuracy attained by using the different settings rather than an objective unit of measurement. Fig. 1 shows the most promising combinations of variable values.

TABLE I. COMBINATIONS OF DETECTION VARIABLES

Dispersion	Variables		
	Accuracy (px)	*Resolution (px)*	*alpha*
0.0233	1	1920x1080	1.0
0.0902	1	1920x1080	1.5
0.0226	1	960x540	1.0
0.0753	2	1920x1080	1.0

The first combination of values in Table 1 is clearly the most accurate, attaining the lowest dispersion value. However, what needs to be taken into consideration is also the speed of the algorithm. With the accuracy and the resolution set to the highest values, the algorithm is capable of outputting less than 3 frames per second. This is especially problematic since the game for attention enhancement needs to be played in real time. Although it does not affect the game itself, it does affect the amount of data which can be gathered in a certain time frame. Another important information shown to us by the Table 1 is that increased alpha values for contrast do not provide expected benefit to accuracy. In this case, the dispersion increased by almost 0.07 from the original value with 1.0 alpha. The first two options in Table 1 are thus not advisable. It is therefore noted that the alpha used should be 1.0 and either the accuracy or the resolution need to be changed. Two tests were conducted. The first one with lowered resolution to 960x540 and the second one with worsened accuracy to 2 pixels. The results have shown the first option to be faster, capable of outputting 8.35 frames per second in testing conditions, while the second option was only capable of outputting 3.99 frames per second. Further reason for lowering the resolution rather than algorithm accuracy was the fact that it allows the system to operate on cameras with much lower quality. Although the precision was lowered, it was still capable of producing satisfying results while gaining a high amount of speed. On other hand, further tests have shown larger decrease in vertical accuracy. For this reason, the final variables have been set to accuracy 1px, resolution 960x810 and alpha 1.0. The increase in vertical resolution proved to retain virtually all of the speed (around 8.3 frames per second) and increased the vertical accuracy.

D. Testing results

The testing has shown the robustness of the system for use in standard indoor environment. The system is capable of accurately detecting the face and the eyes. The accuracy on test subjects with light eye color has proved to be more than adequate. The system also retains high accuracy when used on testing subjects with dark eyes. In this case, the detection is

978-0-7381-2267-7/20/$31.00 ©2020 IEEE

more sensitive to light reflection caused by incorrect lighting. Although the accuracy is still mostly sufficient, it is advised to use good lighting source and angles in order to reduce the risk of reflection. The use of the system also presupposes mostly static position of the head with minimal movement and rotation. This is due to the fact that the program does not contain a system for calculating the head rotation, as it would cause lower accuracy of the system. It is also capable of working accurately under slight angles of the camera. However, placing camera directly in front of the user is still recommended for optimal accuracy. The resulting system is sufficiently capable of measuring user's attention and tracking gaze location. Further improvements to the system accuracy can be made in the form of better gaze direction calculation, calibration and speed optimization.

CONCLUSION

The experimental system for eye tracking, attention measurement and enhancement was presented. Face detection using dlib face detector followed by Timm and Barth pupil detection algorithm was utilized for the purpose of eye tracking. Furthermore, an original system for gaze direction calculation and calibration was developed. Combination of these steps has allowed the fulfillment of the main goal, providing a simple and effective program capable of working with low resolution cameras in visible spectrum. The system is capable of calibrating, measuring and displaying attention metrics.

Due to its robustness, the program can provide a good alternative to other, mostly commercial and expensive, methods of eye tracking, all the while having virtually no special requirements for computational and recording hardware. Further, it may be used for educational games or for tracking of the students' attention.

ACKNOWLEDGMENT

This publication has been published with the support of the Operational Program Integrated Infrastructure within project: Research in the SANET Network and Possibilities of Its Further Use and Development (ITMS code: 313011W988), co-financed by the ERDF.

REFERENCES

[1] Xu J, Min J, Hu J. Real-time eye tracking for the assessment of driver fatigue. *Healthc Technol Lett*. 2018;5(2):54-58. Published 2018 Jan 31. doi:10.1049/htl.2017.0020

[2] Blascheck, Tanja, et al. "State-of-the-Art of Visualization for Eye Tracking Data." *EuroVis (STARs)*. 2014.

[3] J. Mocnej, A. Pekar, W. K. Seah, P. Papcun, E. Kajati, D. Cupkova, J. Koziorek, and I. Zolotova, "Quality-enabled decentralized IoT architecture with efficient resources utilization," *Robotics and Computer-Integrated Manufacturing*, vol. 67, p. 102001, 2021.

[4] A. Pekar, J. Mocnej, W. K. G. Seah, and I. Zolotova, "Application Domain-Based Overview of IoT Network Traffic Characteristics," *ACM Computing Surveys*, vol. 53, no. 4, pp. 1–33, 2020.

[5] Turk, Matthew, and Alex Pentland. "Face recognition using eigenfaces." Proceedings. 1991 IEEE computer society conference on computer vision and pattern recognition. 1991.

[6] Wu, Yue, and Qiang Ji. "Facial landmark detection: A literature survey." *International Journal of Computer Vision* 127.2 (2019): 115-142.

[7] Cortes, Corinna, and Vladimir Vapnik. "Support-vector networks." *Machine learning* 20.3 (1995): 273-297.

[8] Dalal, Navneet, and Bill Triggs. "Histograms of oriented gradients for human detection." *2005 IEEE computer society conference on computer vision and pattern recognition (CVPR'05)*. Vol. 1. IEEE, 2005.

[9] Fuhl, Wolfgang, et al. "Else: Ellipse selection for robust pupil detection in real-world environments." Proceedings of the Ninth Biennial ACM Symposium on Eye Tracking Research & Applications. 2016.

[10] Fuhl, Wolfgang, et al. "Evaluation of state-of-the-art pupil detection algorithms on remote eye images." Proceedings of the 2016 ACM International Joint Conference on Pervasive and Ubiquitous Computing: Adjunct. 2016.

[11] Timm, Fabian, and Erhardt Barth. "Accurate eye centre localisation by means of gradients." Visapp 11 (2011): 125-130.

978-0-7381-2667-7/20/$31.00 ©2020 IEEE