Universal Machine-Learning Processing Pattern for Computing in the Video-Oculography

Albert Śledzianowski(✉), Jerzy P. Nowacki, Konrad Sitarz, and Andrzej W. Przybyszewski

The Faculty of Information Technology, Polish-Japanese Academy of Information Technology, 02-008 Warsaw, Poland
asledzianowski@gmail.com

Abstract. In this article, we present a processing pattern dedicated to video-oculography. It is a complete solution that allows for checking the external conditions accompanying the video oculography, conducting oculometric measurements based on different test models, estimating eye movement (EM) angles and detecting EM type in the stream of coordinates, and calculating its parameters. We based our architecture on neural networks (NN), machine-learning (ML) algorithms of different types, parallel/asynchronous computing, and compilations of the models, to achieve real-time processing in oculometric tests.

Oculometric tests provide significant insight into central neuro-motor states, but machine-learning methods are needed to estimate their meaning. A limitation of this cognitive-analytical trend was the reliance on dedicated measuring devices, such as eye-trackers, which are usually separate and expensive equipment. Presented approach goes beyond these limitations and was developed to use standard home computer equipment, with an internet connection and computer camera. Our set of dedicated algorithms embedded in the software compensates for hardware limitations.

We tested the results of the presented solution on reflexive saccades (RS) and a standard 30 frames per second (FPS) web-camera, and we were able to distinguish between young and old persons and between healthy and prodromal neurodegenerative (ND) subjects by analyzing RS parameters. Visual processes are connected to many brain structures and, by using ML methods, we are trying to dissect them. The development of solutions like the one presented in this article gives hope for the general availability of screening tests connected to ND risk and for collecting extensive data outside the laboratory.

We hope that this direction will contribute to the development of ND analytic means in computational health and consequently, to the faster development of new ND preventive measures.

Keywords: machine learning · deep learning · eye moves · eye tracking · eye move computations · video oculometry · computer vision · reflexive saccades · saccade detections · neurodegenerative diseases · Parkinson's disease

J. Mikyška et al. (Eds.): ICCS 2023, LNCS 14075, pp. 200–212, 2023.
https://doi.org/10.1007/978-3-031-36024-4_15

1 Introduction

The video-oculography supported by computer vision algorithms introduces a non-invasive and economically efficient method of eye tracking based on standard cameras, rather than dedicated devices. It allows for measuring the same components of the EM, and thanks to the possibility of using standard devices such as computer webcams, it brings oculometry to widespread use.

Many researchers experiment with models and new approaches to estimate EM and gaze direction. It has already been confirmed that web-based eye tracking is suitable for studying fixation, pursuit eye movements, and free viewing with repeatability of well-known gazing patterns with slightly less accuracy and higher variance of the data compared to laboratory devices [1]. Furthermore, in the case of RS, such webcam systems are able to calculate parameters with sufficient results, allowing for clinical diagnosis and identification of neurological disorders such as Multiple Sclerosis (MS) [2]. Other researchers have proved that in terms of accuracy, webcam systems can be equal to infrared eye-trackers with 1000 Hz Hz sampling frequency [3].

The core elements of each video-oculographic system are algorithms associated with eye position estimation. There are many different approaches; Lin et al. experimented with estimation based on the appearance-features of eyes, Fourier Descriptors, and the Support Vector Machine, combining a position criterion, gaze direction detection, and lighting filtering [4]. Xu et al. experimented with grayscaling and histogram equalization, creating a 120D feature set for linear regression [5]. However, for the last period of time, for both computer vision and video-oculography, the golden standard has been Convolutional Neural Network (CNN) models. CNNs are based on convolution operations, which, in short, sample every possible pixel arrangement to find specific patterns. In case of video-oculography, this pattern will include the eye component, including the iris and the most important pupils. Akinyelu et al. based their estimation on the face component to extract the gaze features from the eyes, and a 39-point facial landmark component was used to encode the shape and location of the eyes into the network. Experiments confirmed better results of CNN in comparison to the Visual Geometry Group (VGG) NN [6]. Meng et al. experimented with CNN and webcams for an eye-tracking method based on detecting 6 eye features [7]. Gunawardena et al. explored this subject in terms of the efficiency of 4 lightweight CNN models for eye tracking: LeNet-5, AlexNet, MobileNet, and ShuffleNet. They indicate that MobileNet-V3 and ShuffleNet-V2 are the most suitable CNN models [8]. Harenstam-Nielsen experimented with adding Long-Short Term Memory (LSTM) cells to the convolutional layers and Recurrent Neural Network (RNN) cells to the feature layers to increase eyetracking performance [9]. Here, the importance of "memory" modeling in the context of the CNN should be emphasized. For example, the convolutional layer of a model trained for face detection might encode the information that "eyes" are present in the input image, but the receptive field will not have any information to explain whether the eyes are above the nose or not. This is because each convolutional kernel will not be large enough to process many features at one trial.

The cost of tracing wide-range dependencies within an image is ineffectiveness and weight of the model, which is most important for real-time processing. This is why several separate models are used for eyetracking based on the CNN architecture. Usually, this process starts from face detection, then facial landmarks are estimated to determine the location of the eyes. Finally, the image of the eyes is entered into the final network to determine gaze directions.

This issue seems to be already been resolved by the self-attention approach introduced first in transformer-based architectures, which are well known from neural language processing. In computer vision, this architecture takes a feature map as an input parameter and computes relations between each feature based on its weights, resulting in shared information about each object in the image. However, vision transformers in video-oculography are still a very fresh subject and will be elaborated in the next work, as CNN, despite its disadvantages, with the right approach allows for very fast and efficient video-oculographic processing. This is the subject of this text and general purpose was to propose a universal processing pattern based on our best knowledge, which would cover and resolve well-known issues and allow for easy embedding and conducting all standard oculometric tests. We hope that the proposed solution will contribute to the dissemination of oculometric tests also in the context of ND diagnosis.

2 Methods

We developed a pattern for general use in video-oculographic processing constructed of 3 base steps,

1. Analysis of information about sufficient/insufficient equipment or environmental conditions
2. Face detection and facial components estimations including estimation of gaze direction
3. EM estimation including signal smoothing and EM type detection and computations of its parameters.

The description of this processing pattern has been presented in subsequent sections and on Fig. 1.

2.1 Video Signal Quality Check

The environmental factors such as poor lighting or equipment, glasses on the subject's face, or insufficient concentration on the stimulus can bring an unacceptable quality of the signal and recordings of the EM test. This is why we decided to introduce several methods estimating quality components and established threshold values to validate the EM signal. Firstly, we set the frequency level threshold to be ∼25 FPS, as our previous experiments showed that in lower frequencies it is impossible to calculate the parameter within the acceptable error range, due to an enormous temporal sample error. As explained in [3], in the case

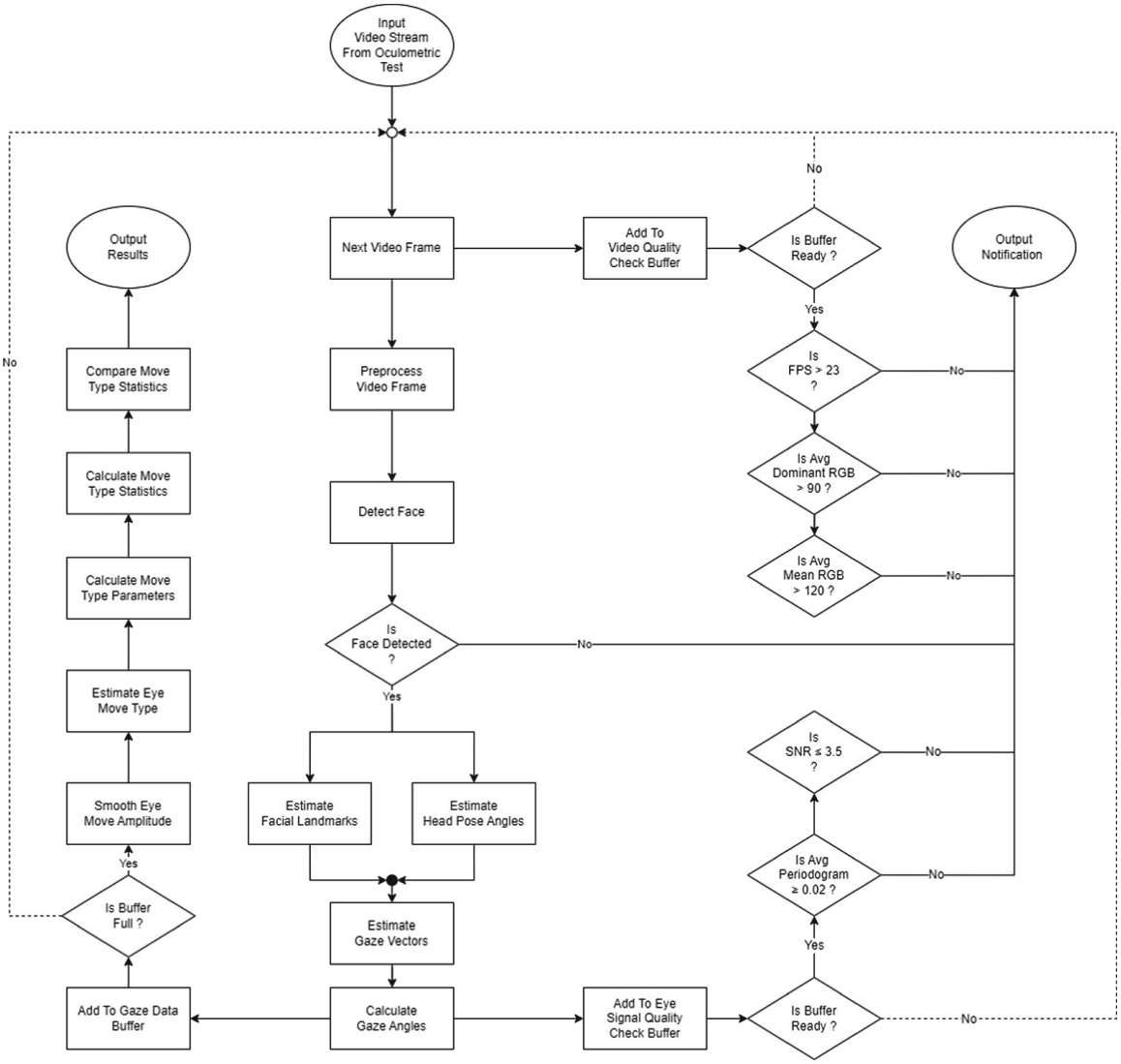

Fig. 1. The process pattern schema for video-oculometric test.

of web cameras, 30 FPS would be ideal, but many factors, such as insufficient light, can affect the video frequency.

This is why we determine lighting conditions from video signals by calculating the dominant and average colors in an image converted to grayscale. The dominant color is estimated with clustering by the k-means function implemented in the OpenCV library with 5 colors and selecting random initial centers in each attempt [10]. The mean color is just calculated as the mean value along the bitmap columns. These two parameters indicate acceptable lighting conditions in presented approach. We have experimentally determined that the average value obtained from the RGB of the dominant color should not be lower than 90 and the mean color should not be lower than 120. Similarly, for video signal estimation, we decided to use two parameters for noise level measurement of EM signals: Signal to Noise Ratio (SNR) as a simple relationship between the horizontal axis EM mean and its statistical deviation (SD). We also calculated the Periodogram of the horizontal axis EM, which estimates the spectral density

of a signal providing statistical average including noise. We used the function implemented in the SpectralPy library with the default "boxcar" window, density scaling, and "constant" detrend. Both parameters support the evaluation of EM noise level and, during experiments, we established acceptable thresholds for EM signals of ≤ 3.5 for SNR and ≥ 0.02 average from the Periodogram.

We propose to use the presented parameters with a small tolerance for quality that is only slightly different from the threshold values. We also propose to perform a quality check during calibration, as this is the moment when we have enough data to perform quality analysis, yet the actual test has not been done, making it easy to stop the study and notify the examined person of insufficient conditions.

2.2 Face Detection and Facial Components Estimations

We decided to base this part of the process on trained neural networks mostly based on convolutional architecture compiled into functions, which allows for the best performance of estimations. We used the Intel OpenVINO™ [11] framework, which uses this method for optimizing deep learning performance together with a pipeline that allows for running many asynchronous estimations in parallel, including image pre-processing. The Fig. 2 presents the schema of parallel processing, which has been used for nested computations, where inference results of the ancestor model (i.e. face detection) are passed to descendant models (i.e. head pose and facial landmarks estimators).

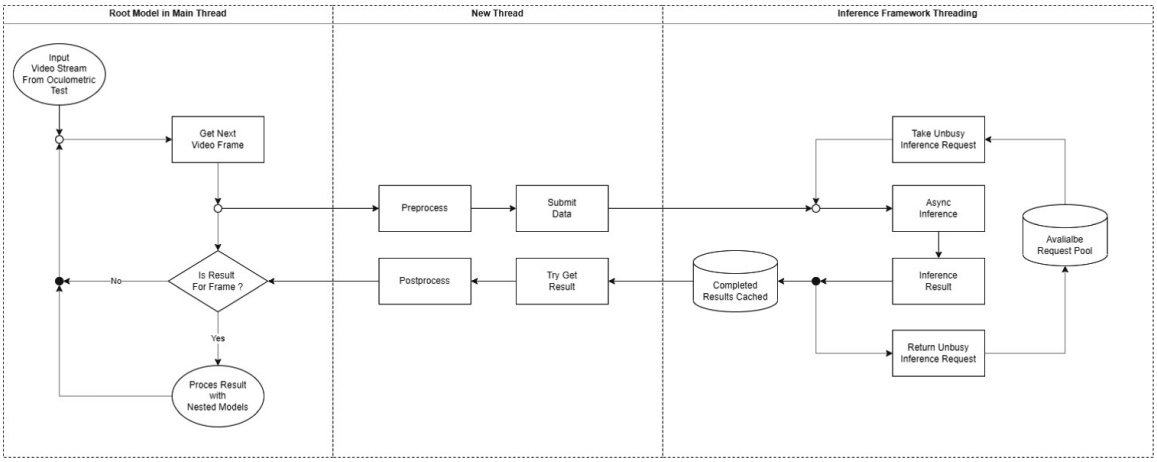

Fig. 2. The schema of parallel processing of nested models.

Additionally, we tried to select the lightest and quickest NN architectures we could find to minimize estimation latency. In order to estimate the eye angles, we connected four models working together in one sequence. The face detector finds the face in the main image from the camera. Next, we smooth the face bounding box coordinates to reduce jittering caused by the detector's uncertainty range with Weighted Moving Average (WMA) with a weight value set to 0.5.

Then, the cropped face image is sent in parallel to the next two models to estimate the position of the head relative to the camera lens and to estimate facial landmarks to find the eyes in the face image. Finally, gaze estimation is performed based on the images of both eyes and the head position angles. Pre-processing includes mostly resizing and rotating the image and reshaping the bitmap matrix into the model input [10,12], post-processing includes mostly reshaping the model outputs into a more convenient format and calculations, i.e. trigonometric conversion from gaze vectors into gaze angles.

For face detection, we used the lightweight "face-detection-0205" model, based on the MobileNetV2 scaffolding and Fully Convolutional One-stage Object Detection (FACOS) as a head [13]. The model was trained with indoor and outdoor scenes shot by a front-facing camera and outputs bounding boxes with 93.57% of average precision (AP) for faces larger than 64×64 pixels. For facial landmarks estimation, we used the "facial-landmarks-35-adas-0002" model [14]. The CNN was trained with a 1000-sample random subset of different facial expressions and outputting a row-vector of 70 floating point values (x, y) for normalized 35 landmarks coordinates with a Normed Error (NE) of 0.106 [13]. For head pose estimation, we used the "head-pose-estimation-adas-0001" model [15]. The CNN had ReLU activation function, batch normalization and angle regression layers as convolutions, fully connected with one output returning Tait-Bryan angles describing rotation in Euclidean space with an accuracy of: yaw 5.4 ± 4.4, pitch 5.5 ± 5.3 and roll 4.6 ± 5.6 [15]. For gaze estimation, we used the "gaze-estimation-adas-0002" model [16]. The Visual Geometry Group CNN model for gaze direction estimation outputs a 3-D vector in a Cartesian coordinate system (in which the z-axis is directed from the mid-point between eyes to the camera center, y-axis is vertical, and x-axis is perpendicular to both z and y) with a Mean Absolute Error (MAE) of 6.95 ± 3.58 of angle in degrees [16].

2.3 Eye Move Estimations

For the purposes of this study, we chose RS as one of the most common types of eye movement performed by humans thousands of times a day, which is also of great importance in oculometric research related to ND [17–25]. In presented approach, RS is stimulated by alternately displaying the fix-point and one of the left/right peripheral targets (markers) in random time intervals (in a step/no-delay model). All are green ellipses in the horizontal axis at a distance of about 8°, and the subjects sat at a distance of about 55–70 cm from the camera/monitor. The study included 20 RS tests performed by each subject, Fig. 3 presents the RS test pattern used in our experiment.

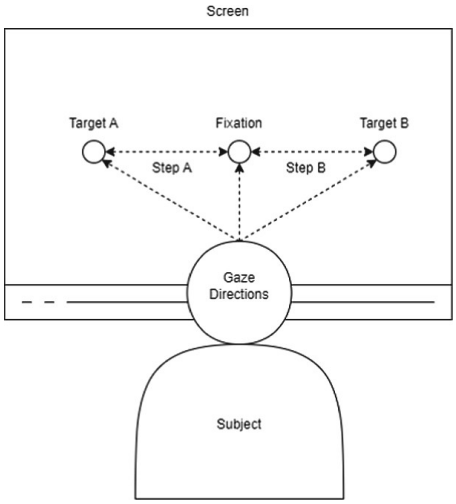

Fig. 3. The reflexive saccades test pattern schema.

For a more accurate estimation of the RS, the signal was smoothed. In presented approach, based on the EM fluctuation analysis, smoothing might be conducive to an accurate determination of the start and end of the amplitude peak. We smoothed the signal with a low-pass Butterworth 2nd order filter with a cutoff of 3.5, and for this purpose, we used the implementation of this function included in the SciPy library [26]. Smoothing was performed outside the RS estimated initially on the raw data, to prevent changing the shape of the RS amplitudes.

For RS detection, we implemented a rolling window function that estimates this type of EM in time series of gaze angles based on stabilization, deviation, and changes in the amplitude. In addition, we introduced a signal of markers to the algorithm for estimations of the RS latency and gain parameters. The algorithm first calculates fluctuation in the fixation state (as EM mean shift in the x-axis) in the small (300 ms) control window (fixation state window) preceding changes in the state of the marker. Then, the size of the next window is calculated with the starting point set between the moment when the peripheral target appears and below the possible minimal RS latency (40 ms) depending on the data frame rate. From the starting point, the window end is set to the maximum RS length (500 ms for $\sim 8°$). Next, the window is iterated in search of longer amplitude change with minimal duration for longer RS (50 ms) towards the displayed target. For such amplitude change to be classified as start of the RS, the shifts between data points must be longer than the mean fluctuation in the fixation state window and must be $\geq 30\%$ of the distance between fix-point and the peripheral target. With the assumption of the min RS duration (50 ms) algorithm looks for the data-point in which position relative to the previous point will be $<30\%$ which is considered as minimal inhibition of the RS and also its ending point. If no starting or ending point was found, a trail is classified as failed. For each detected RS, we calculated the following parameters: latency as the time difference between the start of the marker movement and the start of the eye movement, duration as the total movement time period, amplitude as

the total EM angle, average and maximum velocity as the respectively average and maximum angular velocity, and gain as the ratio between the RS amplitude and fix-point-target distance.

2.4 Experimental Application

We evaluated the presented approach by creating a sample web application in which we implemented our processing pattern. Figure 4 presents its pipeline. In the first stage (Head Position Validation), we adjusted the head position using the camera view with landmarks markers applied to the subject's face. We estimated the distance between the head and the monitor by computing the distance between the subject's center of eyes, received from facial landmarks coordinates. We decided to accept an approximated range of 55–80 cm, allowing the subject to be able to use the mouse or keyboard, but not too far as the face image had to be large enough. We also estimated the perpendicular head position determined on the basis of the vertex positions of the face bounding box. We implemented this feature to obtain registration of subjects in the same head position. The next stage was a calibration. We decided on the simplest three-point schema mapping only the position of the fixation point and the peripheral targets in physical space. During calibration, we also computed the parameters of the video signal quality described in Subsect. 2.1. In all cases, the application reported poor or satisfactory parameter values, enabling corrections to be made. After successful completion of these stages, the main oculometric test for RS was performed, then RS were detected in the registered EM signal and their parameters were estimated as described in Sect. 2.3. Finally, the results were presented on the screen and recorded in a file system of a server.

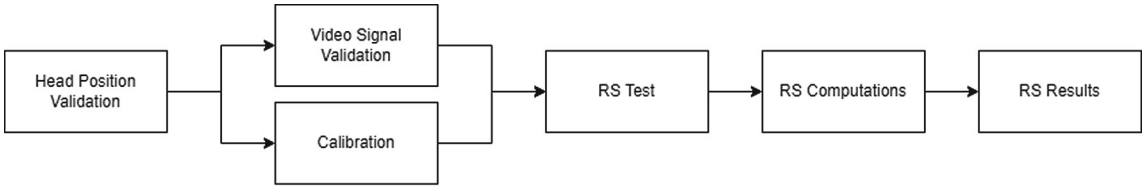

Fig. 4. The pipeline of the experimental application.

3 Results

We wanted to evaluate the presented processing pattern by using application described in previous section. We tested it with 8 subjects in range of 23–45 years old and different gender (158 RS samples). The group was small, but our goal was not to build a statistically large population of different individuals. Rather, we only wanted to collect data from sample equipment and examples of external conditions that may accompanying the study.

We also wanted to reference our results with laboratory measurements and laboratory-grade devices. We decided to compare our results with those published in the data sheet "Descriptive and Reproducibility Results of the Pro-Saccadic Task", where 8° horizontal RS were tested (28 subjects) [27]. All measurements in this research were performed with a professional infrared eye tracker, the Eyelink 1000 Plus, with a frequency of 1000 Hz Hz [27]. In our measurements we used a standard but good quality webcam, the Logitech C922 Pro Stream, at 30 FPS with Full HD mode.

For latency (ms), we obtained a mean of 192 ± 45, for gain (%), 1.16 ± 0.24, for average velocity we received means of 88 deg/sec ± 33.78, and for maximum velocity 145 (deg/s) ± 52. In the reference data, the latency mean was 176 ± 21 (a difference of 16 ms), the gain was 0.98 ± 0.05 (a difference of 0.18), and the maximum velocity was 342 deg/s ± 44 (a difference of approximately 200 ms) [27]. Table 1 presents comparison between experimental and reference statistics. All differences were statistically significant ($P < 0.05$), which is evident given the differences in frequency.

Table 1. Comparison between experimental and reference statistics.

Source	Latency (ms)	Gain (%)	Max Velocity (deg/s)
Webcam 30 Hz - experimental data	192 ± 45	1.16 ± 0.24	145 ± 52
Eye-tracker 1000 Hz - reference data	176 ± 21	0.98 ± 0.05	342 ± 44
Difference	16 ± 48.3	0.18 ± 0.25	197 ± 67.3

We can see that even with such a large difference in sampling rate (970 Hz), some parameters, such as latency, do not show proportionally large differences, while parameters like velocity clearly indicate the impossibility of capturing all values at low frequencies. Therefore, it can be argued that not for every parameter do large differences in frequencies proportionally translate into information loss between subsequent sampling points. This is particularly important when such differences are statistically insignificant when comparing two different groups of subjects, as indicated in the next section of this text.

However, in our system, the processing frequency depends only on the hardware capabilities. This is a fundamental limitation of an approach like ours that cannot be minimized by techniques such as synthetic oversampling, because no two oculomotor systems are the same and artificial data generated from statistical analysis will always be inadequate. In terms of eye tracking accuracy, with the very fast but not super accurate model we used, the results seem to be satisfactory. The difference in gain between our results (1.16 ± 0.24) and the results of the reference study (0.98 ± 0.05) does not seem to be great. Although gain expresses the differences between EM and marker amplitudes, which strongly depends on the subject, in the case of fundamental differences in accuracy, the difference in the resulting gains would certainly be much greater.

However, in terms of frequency, the results for a 100, 300, or 1000 FPS camera would be completely different due to temporal sampling error. The data are collected at some finite frequency, and if it is lower than 1000 Hz, changes in speed or direction of eye movement are not registered between two consecutive sampling points. As 1 ms is the maximal one-point temporal sampling error at a 1000 Hz [28] and the sampling error will be half of the sample time value $((1000/x)/2)$ we can calculate that 300 Hz this error would be 1.6, 100 Hz 5, and 30 Hz 16.6 ms, which corresponds to the difference between results from reference data we received for latency. It must be noted that the data frequency for the processing speed in the proposed pattern seems to be transparent, thanks to the use of parallel/asynchronous estimations and fast models with compiled form. This does not change the fact that in this approach, the frequency of data is independent from the processing, and in this context, the process may be responsible at most for notifications to the users of the system.

4 Discussions

The phenomenon of temporal sample error in lower frequencies, so clearly visible in the case of velocity results, is well described by Anderson et al. [28]. However, its effect has dissimilar influences on different parameters, as shown by the latency where, with a huge difference in sampling frequency, the nominal difference in parameter values is very small. When it comes to the classification of young and old people, or healthy people and ND patients, such an error margin may not affect the results. For example, in one of our previous studies on RS task with PD patients, specific disease stages (varying in progression) showed latency thresholds of 260.0 ms and 308.5 ms [29,30], while the healthy subjects for the RS show latency between 190–200 ms [31]. In this context, it is unlikely that any classifier will go wrong with such large differences. Similarly, for the classification of old and young, or other reasons for deviating from the average results of a regular person, high data rates may not be necessary, so standard computer hardware may be sufficient.

Therefore, it is important to be aware of the type of study and its requirements for sampling rate. Thus, one should be aware of the selection of the appropriate camera for an experiment or patient examination. For computational solutions such as ours, the sampling rate is transparent and depends on the equipment and lighting, which are also evaluated in presented approach by methods described in Sect. 2.

5 Conclusions

In this text, we presented a pattern for processing in modern video-oculometric applications, allowing us to get satisfactory results using consumer-grade computer equipment. We propose using this pattern also with standard web-cameras, where high frequency is not required and approximate results are enough to

make assumptions. Because of the limitations of consumer-grade equipment, it will never compete in accuracy with laboratory equipment.

However, we have proved that it can bring reliable results, such as for latency in PD, which could be used for screening tests of ND. Due to the availability of computer devices, applications using this pattern may disseminate oculometric examinations, making them universally available.

For more accurate applications, high frame rate equipment is needed, but our solution is fully scalable and, in this context, parallel processing adapts to the capabilities of the video device.

Declaration of Competing Interest

Software frameworks and libraries: OpenVino, Tensorflow, OpenCV, SciPy, Scikit Learn, Numpy, Pandas, contributed to the development of the software methods presented in this article. The authors declare no conflict of interests.

References

1. Semmelmann, K., Weigelt, S.: Online webcam-based eye tracking in cognitive science: a first look. Behav. Res. Methods **50**(2), 451–465 (2017). https://doi.org/10.3758/s13428-017-0913-7
2. Aljaafreh, A., Alaqtash, M., Al-Oudat, N., Abukhait, J., Saleh, M.E.: A low-cost webcam-based eye tracker and saccade measurement system **14**, 04 (2020)
3. Naruniec, J., et al.: Webcam-based system for video-oculography. IET Comput. Vis. **11**(2), 173–180 (2017)
4. Lin, Y.-T., Lin, R.-Y., Lin, Y.-C., Lee, G.C.: Real-time eye-gaze estimation using a low-resolution webcam. Multimedia Tools Appl. **65**, 543–568 (2012)
5. Xu, P., Ehinger, K.A., Zhang, Y., Finkelstein, A., Kulkarni, S.R., Xiao, J.: TurkerGaze: crowdsourcing saliency with webcam based eye tracking (2015)
6. Akinyelu, A.A., Blignaut, P.: Convolutional neural network-based technique for gaze estimation on mobile devices. Frontiers Artif. Intell. **4** (2022)
7. Meng, C., Zhao, X.: Webcam-based eye movement analysis using CNN. IEEE Access **5**, 19581–19587 (2017)
8. Gunawardena, N., Ginige, J.A., Javadi, B., Lui, G.: Performance analysis of CNN models for mobile device eye tracking with edge computing. Procedia Comput. Sci. **207**, 2291–2300 (2022). Knowledge-Based and Intelligent Information And Engineering Systems: Proceedings of the 26th International Conference KES2022
9. Harenstam-Nielsen, L.: Deep convolutional networks with recurrence for eye-tracking [internet] [dissertation] (2018)
10. Bradski, G.: The OpenCV library. Dr. Dobb's J. Softw. Tools (2000)
11. Intel®. Intel® distribution of openvino™ toolkit (2022). https://docs.openvino.ai/
12. Harris, C.R., et al.: Array programming with NumPy. Nature **585**, 357–362 (2020)
13. Intel®. Openvino™ toolkit - open model zoo repository, Intel's pre-trained models (2023). https://docs.openvino.ai/latest/omz_models_model_face_detection_0205.html

14. Intel®. Openvino™ toolkit - open model zoo repository, Intel's pre-trained models (2022). https://docs.openvino.ai/latest/omz_models_model_facial_landmarks_35_adas_0002.html
15. Intel®. Openvino™ toolkit - open model zoo repository, Intel's pre-trained models (2022). https://docs.openvino.ai/2019_r1/_head_pose_estimation_adas_0001_description_head_pose_estimation_adas_0001.html
16. Intel®. Openvino™ toolkit - open model zoo repository, Intel's pre-trained models (2022) https://docs.openvino.ai/2019_r1/_gaze_estimation_adas_0002_description_gaze_estimation_adas_0002.html
17. Chambers, J.M., Prescott, T.J.: Response times for visually guided saccades in persons with Parkinson's disease: a meta-analytic review. Neuropsychologia **48**(4), 887–899 (2010)
18. Turner, T., Renfroe, J., Delambo, A., Hinson, V.: Validation of a behavioral approach for measuring saccades in Parkinson's disease. J. Motor Behav. **49**, 1–11 (2017)
19. Stuart, S., et al.: Pro-saccades predict cognitive decline in Parkinson's disease: ICICLE-PD. Mov. Disord. **34**(11), 1690–1698 (2019)
20. Perneczky, R., Ghosh, B.C.P., Hughes, L., Carpenter, R.H.S., Barker, R.A., Rowe, J.B.: Saccadic latency in Parkinson's disease correlates with executive function and brain atrophy, but not motor severity. Neurobiol. Dis. **43**(1), 79–85 (2011). Autophagy and Protein Degradation in Neurological Diseases
21. Antoniades, C., Zheyu, X., Carpenter, R., Barker, R.: The relationship between abnormalities of saccadic and manual response times in Parkinson's disease. J. Parkinsons Dis. **3**, 10 (2013)
22. Yang, Q., Wang, T., Su, N., Xiao, S., Kapoula, Z.: Specific saccade deficits in patients with Alzheimer's disease at mid to moderate stage and in patients with amnestic cognitive impairment. Age (Dordrecht, Netherlands) **35** (2012)
23. Pereira, M.: Saccadic eye movements associated with executive function decline in mild cognitive impairment and Alzheimer's disease: Biomarkers (non-neuroimaging)/novel biomarkers. Alzheimer's Dement. **16** (2020)
24. Boxer, A.: Saccade abnormalities in autopsy-confirmed frontotemporal lobar degeneration and Alzheimer disease. Arch. Neurol. **69**, 509–517 (2012)
25. Śledzianowski, A., Szymanski, A., Drabik, A., Szlufik, S., Koziorowski, D.M., Przybyszewski, A.W.: Combining results of different oculometric tests improved prediction of Parkinson's disease development. In: Nguyen, N.T., Jearanaitanakij, K., Selamat, A., Trawiński, B., Chittayasothorn, S. (eds.) Intelligent Information and Database Systems, pp. 517–526. Springer, Cham (2020). https://doi.org/10.1007/978-3-030-42058-1_43
26. Virtanen, P., et al.: SciPy 1.0: fundamental algorithms for scientific computing in Python. Nat. Methods **17**, 261–272 (2020)
27. Bijvank, J.N., et al.: A standardized protocol for quantification of saccadic eye movements: demons. PLOS ONE **13**, e0200695 (2018)
28. Andersson, R., Nyström, M., Holmqvist, K.: Sampling frequency and eye-tracking measures: how speed affects durations, latencies, and more. J. Eye Move. Res. **3**, 1–12 (2010)
29. Szymanski, A., Szlufik, S., Koziorowski, D.M., Przybyszewski, A.W.: Building classifiers for Parkinson's disease using new eye tribe tracking method. In: ACIIDS (2017)

30. Przybyszewski, A., Kon, M., Szlufik, S., Szymański, A., Habela, P., Koziorowski, D.: Multimodal learning and intelligent prediction of symptom development in individual Parkinson's patients. Sensors **16**, 1498 (2016)
31. Ramsperger, E., Fischer, B.: Human express saccades: extremely short reaction times of goal directed eye movements. Exp. Brain Res. **57**(1), 191–5 (1984)