

Embedded System For Computer Vision Syndrome Relief: A Technical Note

Huy G. Nguyen*, Nhien L. Nguyen*, Quang N. Pham*, Bao T. Tran*

*: High School for The Gifted, VNUHCM

Abstract: Computer Vision Syndrome (CVS) is a globally widespread issue among individuals who spend long periods working on computers, leading to symptoms such as eye strain and blurred vision, affecting life quality and productivity. Treatments and prevention of CVS have been discussed in previous papers; however, their efficiency has been proven to be insignificant. Moreover, each current treatment and prevention technique of CVS only addresses one CVS-inducing factor. This research project aims to alleviate the effect of CVS by developing the Computer Vision Syndrome Relief System (CVSRS), an embedded system designed to simultaneously implement a range of the current CVS prevention techniques, effectively addressing multiple CVS-inducing factors at once. The system integrates multiple prevention techniques of CVS into a single system, including blink rate drop notification, 20-20-02 rule notification, automatic adjustment of screen brightness to ensure appropriate screen-environment brightness ratio, and automatic adjustment to ensure appropriate screen height/viewing angle. The CVSRS is expected to be more effective than current methods that address only individual factors of CVS. The development process of the CVSRS has reached the state of which all the applications are completed separately: a model for blink detection for low blink-rate notification, a gaze-lock detection model for the 20-20-20 notification for an embedded program for adaptive brightness purposes.

Keywords: Computer Vision Syndrome, blink detection, gaze-lock detection, automatic screen brightness adjustment

1. INTRODUCTION

Computer Vision Syndrome (CVS) is the medical term used to describe the ocular and visual symptoms resulting from prolonged computer usage. Symptoms include eye strain, eye fatigue, aches in and around the eyes (ocular symptoms), blurred vision near, blurred vision when looking from near to far (visual symptoms), etc [1]. These symptoms lead to discomfort for users during, and immediately after computer use; the presence of CVS is also associated with lowering life quality and work productivity [1]. CVS is prevalent among computer users all around the world, and the longer the duration of computer usage, the more severe the symptoms [1]. To be more precise, it was reported by a paper in 2015 that 64% to 90% of computer users reported experiencing CVS [1]. A 2023 meta-analysis revealed that 40 out of 45 global studies reported a prevalence of CVS exceeding 50% among participants regardless of age and occupation, with the highest percentage being 98.7% of the sample size [6]. It is clear that the problem of CVS is widespread, and with the growth of computer integration into our daily lives, its prevalence is expected to grow in the future without the development of effective treatments. The factors that are responsible for CVS are divided into 3 main categories and their current treatments/prevention techniques are summarized in Table 1 below.

Table 1. Summary of CVS-inducing factors and their treatments

Category	Inducing factors	Treatment/prevention techniques
Users' natural visual ability-related factors	Uncorrected refractive errors, presbyopia, oculomotor anomalies, etc [1]	<ul style="list-style-type: none">- Eyeglasses/spectacle lenses such as progressive addition spectacle lenses, and blue light-blocking spectacle lenses [2]- Surgery and other medical (optical) interventions- Orthoptic Exercises- Antioxidant and nutritional supplements [2]
	Dry eye [1, 2, 3]	<ul style="list-style-type: none">- Lubricating drops [5]- Artificial tears [2]- Blinking exercise: closing the eyelids for 2 seconds two times then squeezing eyelids for 2 seconds [3]- 20-20-20 rule: every 20 minutes of screen time, a user should look away at something at least 20 feet away for at least 20 seconds [3]- Utilize a desktop humidifier so as to keep the humidity of the working environment at or above 40% [3]
Environmental	Inappropriate lighting	<ul style="list-style-type: none">- The recommended maximum permissible luminance

and work factors	<ul style="list-style-type: none"> - Unequal luminances between the background and computer screen [1] - Glare [1, 3] - Reflections of computer screen [1] 	ratio to be 1:3 between the task and the immediate surroundings or 3:1 between the immediate surroundings and the task (the ratio is dependent on the distance of the glare source from the point of fixation) [1] - Anti-glare and anti-reflection glass
	Prolonged computer usage [3]	- 20-20-20 rule: every 20 minutes of screen time, you should look away at something at least 20 feet away for at least 20 seconds [3]
Device related factors	Screen's display position: <ul style="list-style-type: none"> - Display height [1, 3] - Display viewing distance [1, 3] 	<ul style="list-style-type: none"> - Improving workstation ergonomics of the display position, the optimal display position should be chosen so as to minimize both ocular and nonocular symptoms. The top display screen should be at or lower than 5 degrees below the line of sight and the center of the screen should be no more than 25 degrees below the line of sight. [1] - The center of the display screen should be between 15 to 20 degrees below the horizontal level of the user's eye [3] - The entire display screen must be positioned so that the angle to be viewed downwards does not exceed 60 degrees [3] - The optimal display viewing distance is dependent on screen size, character size, and users' visual ability. The recommended distance according to The Occupational Safety and Health Administration (OSHA) is between 45 - 60 cm. [1]
	Display resolution [1, 3]	- Users are advised to not use computers with too low resolution
	Text, image quality, presentation rate and refresh rate [1, 3]	- The Video Electronic Standards Association (VESA) has recommended a minimum refresh rate of 75 Hz that minimizes flicker at all brightness levels. [5]
Psychological factors	<ul style="list-style-type: none"> - Stress and Anxiety [1] - Sleep quality [1] 	- Psychological treatments

Although currently there are various treatments for CVS, their effectiveness has not been thoroughly assessed and proven as there are many more studies about CVS that need to be conducted in order to “uncover gaps in our understanding of the problem” [7, 8]. Furthermore, articles that propose prevention and treatment for computer vision syndrome often present

individual treatments. It is logical to reason that the more treatment one can carry out, the more efficient the treatment and prevention process will be; however, realistically speaking, many people who are preoccupied with a computer will struggle to maintain a myriad of such actions.

To this end, this research aims to implement a combination of CVS prevention and treatment methods simultaneously, using technology to make this process more convenient and feasible for individuals who may struggle to manage multiple actions while working on a computer. More specifically, we developed a prototype of the Computer Vision Syndrome Relief System (CVSRS), which incorporates a custom PCB board as well as software to address multiple factors contributing to CVS at once. The inducing factors of CVS that this project aims to address are dry eye, lighting-related factors, and prolonged computer usage by simultaneously implementing the treatments that target those factors presented in Table 1 above. More specifically, the functionalities of the final system include:

- (1) A blink detection model to calculate the blink rate and notify the user if the blink rate signifies a strong possibility of contracting computer vision syndrome.
- (2) An automatic gaze-lock detection model to enforce the 20-20-20 rule.
- (3) A custom PCB and program automatically adjust the screen brightness of the computer screen in order to maintain the optimal brightness ratio between the screen and the surrounding environment.

By integrating solutions such as screen brightness adjustments and blink rate monitoring into a cohesive system, this project proposes a more feasible and user-friendly approach to addressing CVS. Unlike traditional methods, which rely on the user manually performing multiple preventative actions, our system automatically enforces CVS prevention and treatment methods, reducing the burden on users and ensuring higher compliance when they are preoccupied with other tasks when working on the computer. To the best of our knowledge, this paper proposed the first system to implement a range of CVS prevention and treatment methodologies simultaneously.

2. MATERIAL & METHODOLOGY

2.1. Blink drop notification

It is proven that the blink rate correlates positively with the severity of CVS symptoms. The blink rate in normal circumstances ranges from 15 to 20 blinks per minute, and when working on a computer it decreases to 4 to 6 times per minute. Another study states that the blink rate decreases from 22 times per minute in the relaxed state to 7 times per minute when working on a computer [15]. To prevent and alleviate the severity of CVS symptoms, we implemented a blink detection model to calculate the number of blinks per minute in real-time so as to notify the user if their blink rate is a possible indicator of CVS.

The blink-drop notification function is carried out in various steps. First, a blink detection model is developed. Then, as the blink detection model is deployed to count the number of blinks in real time, the blink rate (blink per minute) is calculated in a 1-minute sliding window fashion. Lastly, for each minute, if the blink rate is below a threshold, a notification is created to notify the user.

2.1.1. Blink detection model

For blink detection, we implement the method suggested by [14], which utilizes the Eye Aspect Ratio (EAR):

$$EAR = \frac{||p2-p6|| + ||p3+p5||}{2||p1 - p4||}$$

In this formula, $p1$, $p2$, $p3$, $p4$, $p5$, and $p6$ are the coordinate pairs (x and y) of the landmarks of the eyes. The ratio is the total of the Euclidean distance between 4 vertical eye landmarks divided by 2 times the Euclidean distance between the 2 horizontal landmarks. The eye's landmarks are detected using Google's MediaPipe Face Landmark.

However, since MediaPipe is able to detect 16 eye landmarks (Figure 1), we suggest a variation of the original formula, which considers 6 coordinate pairs instead of 3. With additional landmarks in consideration, we could capture more subtle variations of the eye-area movements and thus help distinguish a blink from other facial expressions.

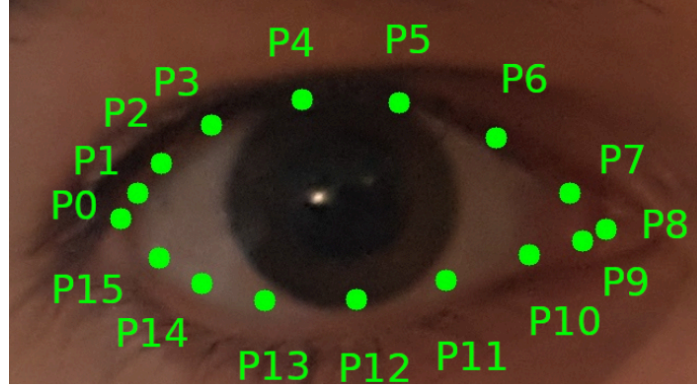


Figure 1. 16 eye landmarks coordinates detected by MediaPipe Facial Landmark Detector, denoted P0 to P15.

Our suggested formula is as follows:

$$EAR = \frac{||P4-P12||+||P5-P11||+||P3-P13||+||P6-P10||+||P2-P14||}{5*||P0-P3||}$$

The EAR value is calculated in real-time. In order to reduce the noise and reduce unwanted fluctuation in the retrieved data, we incorporate a moving average filter into the process. To that end, the EAR after raw calculations is then subjected to this formula (the EAR value after applying the moving average filter is denoted EAR_A):

$$EAR_A(t) = \frac{1}{w} \sum_{i=t-w+1}^t EAR_i$$

Where w is the width of the filter and t is the current frame. Our moving average filter has a width of 2 frames.

Subsequently, as proposed by [14], a Support Vector Machine (SVM) model is trained for the blink detection task. For each frame, the EAR value is calculated, and the value of the six preceding and subsequent frames of that frame is collected to form a 13-dimensional vector, which is then fed to the SVM as input. The model is trained on the EyeBlink8 dataset, which consists of 82600 frames spanning 8 videos of 4 individuals in different conditions [16].

After training, the model is able to identify in real time if a video frame is part of a blink sequence. Therefore, to detect a complete blink, a blink duration threshold is added. The duration of a blink is usually 0.1 seconds to 0.4 seconds - without special conditions any facial actions that are longer than that duration are not considered a blink. We propose a blink duration

threshold after the model classifies if a frame is part of a blink sequence. To be more specific, if the prediction is “1” for more than $0.4 * (FPS)$. FPS is the frame per second of the camera, which in this case is 30 - this means our blink duration value is $0.4 * 30 = 12$ frames, and a blink is counted. On the other hand, if the duration is more than the threshold, it is regarded as a non-blink action.

2.1.2. Blink rate calculation and blink drop notification function

As the model identifies and counts blinks in real time, the calculation of the blink rate (the number of blinks per minute) is carried out in a 1-minute window fashion. More specifically, for each minute, a blink rate value, which is the number of detected blink in that minute is recorded. If the blink rate falls below 6 blinks per minute, the user is notified via a pop-up notification on their computers; if the blink rate exceeds the threshold, the blink rate is calculated for the next 1 minute.

2.2. 20-20-20 notification

A Feed-Forward-Network(FFN) model is applied on top of Mediapipe’s landmark detection model for its simplicity and lightweight nature as the main point of the application is enclosed to differing gaze from non-gaze expression to classify the gaze-lock position of the user. Based on the change of face ratio from a 2D projection on screen compared to the user’s straight-looking face, we can estimate the head direction of the user. Another key feature is the iris position, which will decide which relative direction the user is looking to. The features to feed into the ANN are first normalized based on their ratio to the face's vertical and horizontal distance for the purpose of tuning out the face's distance to the screen and consequently, the camera focal length.

- **FaceRatio** - The division of the current face ratio and one when looking straight
- **minDistanceToNose** - Distance from nose to face center point
- **LVRatio** - Left eye vertical ratio
- **LHRatio** - Left eye horizontal ratio
- **RVRatio** - Right eye vertical ratio
- **RHRatio** - Right eye horizontal ratio
- **HDistRatio** - Left eye horizontal distance
- **noseDistanceX** - Current x position of the nose relative to the center point
- **noseDistanceY** - Current y position of the nose relative to the center point

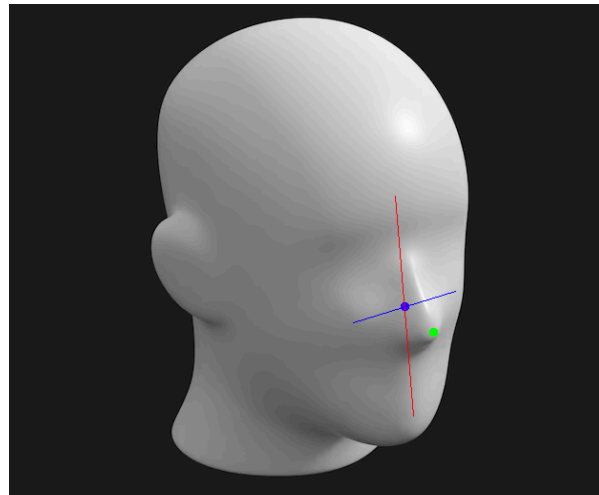


Figure 5. An illustration of the data extracted: the horizontal (blue) and vertical distance (red) of the face, their intersection, and the relative nose position in the image plane.

Different datasets were used in this study: The Columbia Gaze Data Set, Gaze 360, and a custom dataset. The gaze-lock detection model trained on the custom dataset has shown the best quality, this can be explained by the data collecting methods being tailored to the study. The dataset consists of videos from a webcam of 6 participants, these images were categorized into two classes: "Looking" (1) and "Not Looking" (0). Image frames from the dataset were processed through the Mediapipe face landmark detector, which returned numerical values corresponding to the desired features. These values served as inputs to the FFN model. The FFN was trained in mini-batches with an additional batch normalization process, a dropout rate of 0.25, using the Adam optimizer with L2 regularization to prevent overfitting. As of now, we are testing a model trained with a more prominent Gaze360 dataset, better results can be expected.

The user must calibrate to get the face ratio and the normal state of their eyes. By registering a new user. A notification will come up if, throughout 20 min, the users appear to be consistently staring at the screen without at least 20 seconds of looking away.

2.3. Automatic brightness adjustment

Overview: Automatic brightness adjustment technology enhances user experience by dynamically adjusting screen brightness based on ambient light conditions of the surrounding environments. This adaptation reduces eye strain, conserves energy, and optimizes screen visibility. This feature involved automatic brightness adjustment, exploring sensor-based and software-driven approaches. Sensor-based approaches use light sensors to measure ambient light levels and adjust the screen brightness accordingly, often aiming for an ideal brightness ratio of 3:1 with the surrounding light conditions. Software-driven approaches may further refine this adjustment, using algorithms that can adapt to a user's environment and preferences.

Implementation: Automatic brightness control on laptops that don't natively support it is a complex undertaking, as it requires interaction with hardware components such as ambient light sensors and potentially modifying the operating system's drivers. Only some of the advanced and hybrid laptops such as Macbook from Apple or Dell Inspiron or other gaming laptops have these features. Therefore, in this project, we will aim to create a USB head and embedded system to calculate the environment light and automatically adjust the screen brightness to optimize the screen visibility and reduce computer vision syndrome relief for other types of laptops.

2.3.1. Electrical Circuit:

1. Components:



ATMEGA 328P



CRYSTAL OSCILLATOR
16Mhz



USB UART CH340G



CAPACITOR 22pF



LDR GL5537



RESISTOR 10KΩ



BUTTON

2. Circuit Designing:

Figure 1 shows a setup for an ATmega328P microcontroller, configured to measure ambient light and communicate via USB. The ATmega328P, acting as the central controller, is connected to a CH340G USB-to-serial converter, which allows data transmission between the microcontroller and a computer over USB. A light-dependent resistor (LDR) is paired with a resistor to form a voltage divider, enabling the ATmega328P to detect changes in ambient light based on the LDR's variable resistance. Additionally, the circuit includes a crystal oscillator connected through two capacitors to stabilize the microcontroller's clock, ensuring precise timing for its operations. A push button with a pull-up resistor provides manual reset functionality by grounding the RESET pin when pressed. This configuration is suitable for applications where ambient light levels need to be sensed and processed, such as in an automatic brightness control system for screens or other light-sensitive devices.

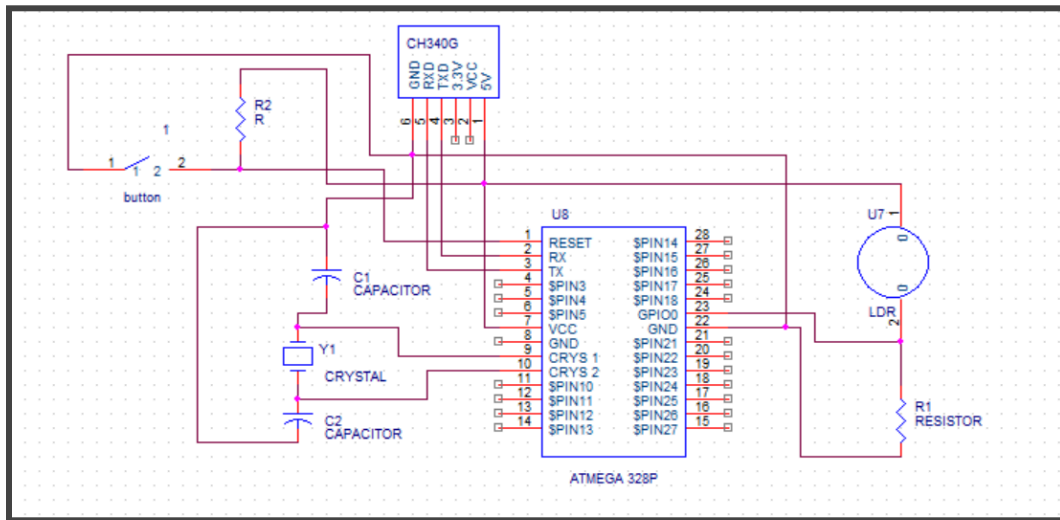


Figure 1: Electrical Circuit using ATMEGA 328P and Light Dependent Resistor.

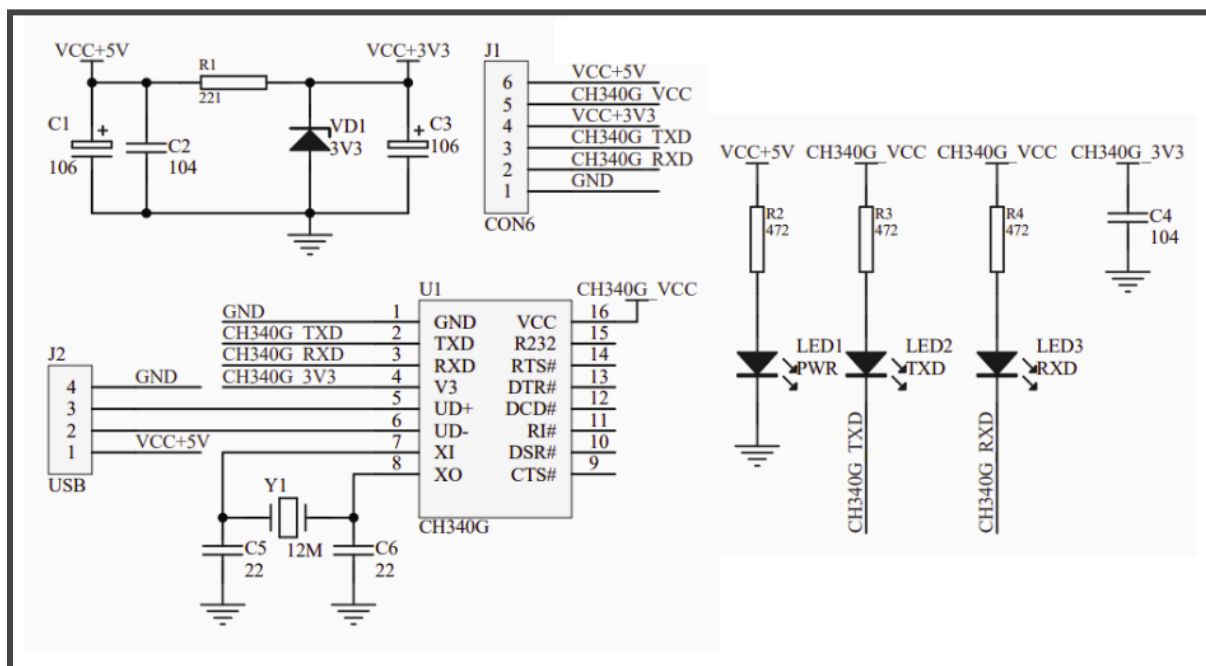


Figure 2: Electrical circuit of USB UART CH340G.

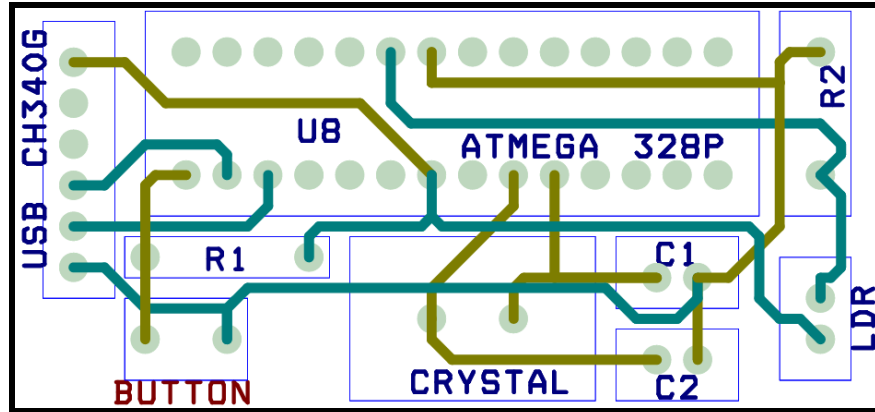


Image 1: PCB design of adaptive brightness

The adaptive brightness system works by capturing ambient light data through the LDR and using the ATmega to interpret these light levels. When the LDR detects increased ambient light, its resistance decreases, lowering the analog voltage read by the microcontroller. The ATmega uses these readings to adjust the screen brightness, either by controlling the laptop display directly (if supported) or adjusting an attached LED's brightness.

The microcontroller has a set range of brightness levels mapped to light intensities. Depending on the reading, the microcontroller sends a brightness adjustment signal via USB, creating a feedback loop for continuous brightness adjustment.

3. Lux transmission formula with analog signals

The provided information outlines the key principles and equations governing the behavior of a light-dependent resistor (**LDR**) and its relationship to light intensity. The circuit diagram shows an LDR connected in series with a resistor (**R2**), powered by a voltage source (**VCC**), and the output voltage (**Vout**) is determined by the ratio of the LDR resistance (**R2**) and the series resistor (**RS**). The equation for **Vout** demonstrates how the output voltage is calculated based on these resistance values, allowing for the determination of voltage output based on the LDR resistance.

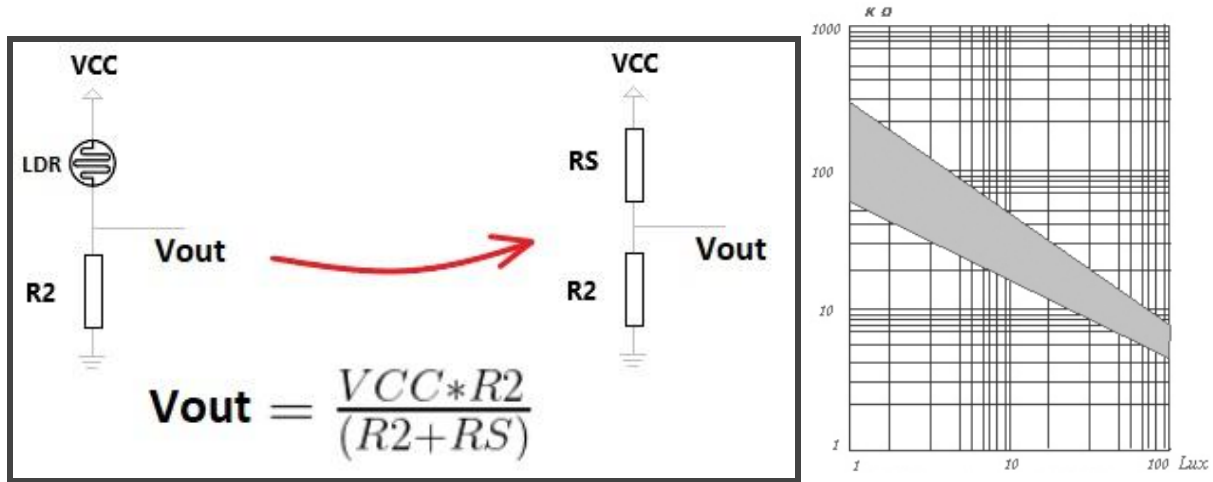


Figure 3: Equation for Light Dependent Resistor values and Lux values

The equation for lux in terms of resistance (R) is as follows:

$$Lux = \frac{10^c}{R^\alpha}$$

Where:

- c is a constant determined experimentally, with $c = 8.32899$
- $\alpha = \frac{5}{3}$ based on $\delta = \log\left(\frac{R_{10}}{R_{100}}\right) = 0.6$

This equation allows you to calculate the ambient light in lux based on the resistance R measured from a light-dependent resistor (LDR) or similar sensor. The values of c and α were determined experimentally to fit the observed relationship between resistance and light intensity.

$$y = \frac{\text{Lg}(R_{10}/R_{100})}{\text{Lg}(100/10)} = \text{Lg}(R_{10}/R_{100})$$

Figure 4 shows a logarithmic relationship between ambient light (measured in lux) and screen brightness (measured in percentage). The equation for this relationship is given as:

$$y = 9.9323 \ln(x) + 27.059$$

where y represents the screen brightness percentage and x is the ambient light in lux. This equation indicates that as ambient light increases, screen brightness also increases but at a gradually decreasing rate, approaching a maximum brightness level as lux levels rise.

The curve fits the data with an R square value of 0.9576, showing a high degree of correlation between measured data points and the model. Using this equation, once the ambient light level in lux is known (e.g., from an LDR sensor), it can be converted to an appropriate screen brightness

percentage. This model helps in achieving smooth and efficient automatic brightness control, adjusting the screen brightness according to surrounding light levels to optimize user comfort and power consumption.

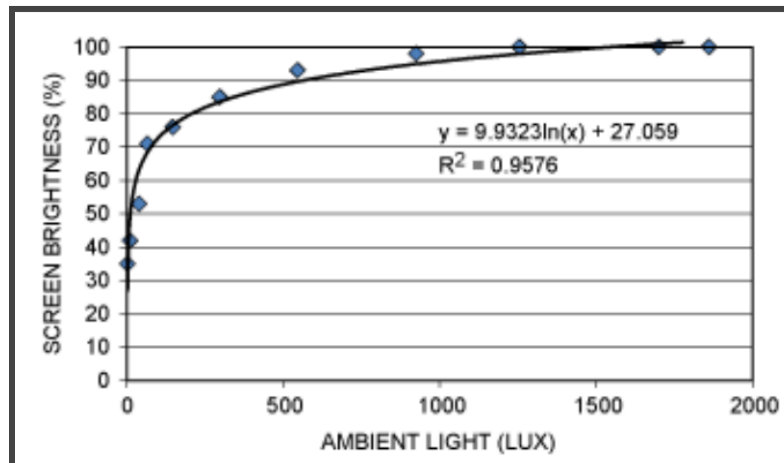


Figure 4: Exponential equation from lux to screen brightness percentage in computer

2.4. Evaluation process

2.4.1. Blink rate drop notification

For this function, we will evaluate the 2 following criteria:

- (1) Blink detection model's accuracy: test on a custom-created dataset called RealWorldBlink. See if the model is able to detect blinking accurately, differentiate between blink and other facial expressions such as squinting, detect blinking in different head poses, and accurately detect rapid blinking. The accuracy of the model is assessed by collecting values including precision, recall, F1, and AUC score.
- (2) Responsiveness of algorithm: test in real-world experiment - have users use the program for about 5-15 minutes, record the blink rate and the notification time stamp: does the algorithm notify the user at the right time? Calculate the percentage of times when the notification was on time and when it was not.

2.4.2. 20-20-20 rule notification

As the scope of this task is to differentiate gaze from non-gaze, the process is limited to evaluating the usability and how far away from real gaze the accumulated detected gaze time is:

- (1) Model's accuracy: test on a different set of unused images and labels.
- (2) Usability: test on webcam videos and see how far the 20-minute notification is from the real 20-minute gaze.
- (3) Effectiveness: Compare subjects who strictly followed the notification to those who didn't. How it affects their work rate and consciousness after prolonged screen time.

2.4.3. Automatic adaptive brightness

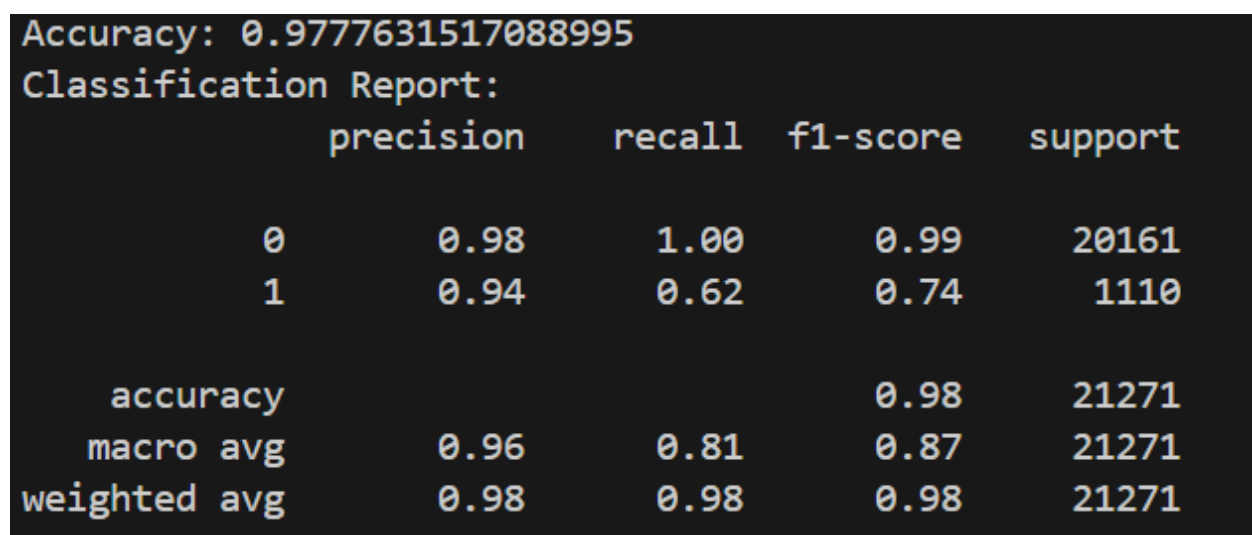
For this function, we will evaluate the 2 following criteria:

- (1) Reaction Time: Simulate different lighting environments (e.g., turning lights on and off and experimenting with different light intensities) and record the time the system takes to adjust. Measure how quickly the system adjusts the brightness when ambient light changes.
- (2) Accuracy: Follow the same experiment as above. Compare the system's adjusted screen brightness with the manually calculated ideal brightness level for the given ambient lighting condition.

3. RESULTS

3.1. Blink detection model

Our SVM model is trained using the Eyeblink8 database. After manipulating the data and training the SVM, our model achieved an accuracy of nearly 98% (Figure 2). However, due to the unbalance of our dataset, the performance scores for identifying positive samples are lower compared to when identifying negative samples. We attempted to solve this problem by applying SMOTE and adjusting our process of negative samples by adding a 5-frame spacing between each collected negative sample but the performance of those versions when deployed are lower compared to our previous version.



```
Accuracy: 0.9777631517088995
Classification Report:

```

	precision	recall	f1-score	support
0	0.98	1.00	0.99	20161
1	0.94	0.62	0.74	1110
accuracy			0.98	21271
macro avg	0.96	0.81	0.87	21271
weighted avg	0.98	0.98	0.98	21271

Figure 2. The performance report of our SVM is generated by Python’s Sklearn library, which is used to implement the SVM. Although we have achieved high accuracy, our data is still unbalanced regarding the number of positive and negative samples, which leads to lower performance scores when the model attempts to identify positive samples.

RealWorldBlink

According to our literature review, many papers evaluate their proposed blink detection algorithm using datasets. Many opted for popular datasets for blink detection algorithms such as EyeBlink8 and TalkingFaces [citation], while others created their own evaluation datasets to match their own preference [citation - EARM]. There is a concern that if we evaluate our algorithm on a previously created dataset, the calibration process might not return the most accurate EAR threshold, because our calibration process requires manual calibration. To this end, we designed our own experiment, and in the process created our own evaluation dataset called RealWorldBlink. Below is a detailed description of the RealWorldBlink dataset.

Evaluation videos: All the videos are collected by a web camera with 640x480 resolution and 30fps. Each video is from about 2 minutes to approximately 3 minutes long. The dataset contains 6 such videos. Each video is a selfie recording of a subject working on a computer with their full upper body and head present and no hair covering their eyes region of interest that can prevent the face detection process; there is only 1 face present in each video. The lighting is maintained well-lit but with different intensities and light colors. During the recording, the subjects blink naturally but are reminded to include facial expressions such as squinting and grimacing as well as change head pose ranging from 0 to 45 degrees from the screen. They are also reminded to include rapid blinks with 2 or 3 blinks in quick succession. The videos are annotated by the annotation tool version 3.2 developed by Folgeron [citation - his web?]. The details of each video in the dataset is presented in Table x.

Subjects: All of the subjects are Asian. There were 6 males, and 3 of the males wore glasses.

Video	Video 1	Video 2	Video 3	Video 4	Video 5	Video 6
Video info						
Resolution (pixels)	640x480	640x480	640x480	640x480	640x480	640x480
FPS	30	30	30	30	30	30
Frame count	8141	3011	5558	3576	3023	4990
Duration (s)	271.4	100.4	185.3	119.2	100.8	166.3
Number of frames processed	8141	3011	5558	3576	3023	4990
Ground truth blinks by manual notification	71	45	54	67	33	34
Ground truth double blinks by manual notification	6	6	12	6	6	5

Table x. Information about each video in the RealWorldBlink video dataset

The true positives (TP), false positives (FP), true negatives (TN), and false negatives (FN) of each evaluation model in each video are presented in Table X. Subsequently, the precision, recall, F1 score, support, as well as AUC is presented in table x.

		Video 1	Video 2	Video 3	Video 4	Video 5	Video 6
Model 2: With moving average filter	TP	51	31	10	54	22	12
	FP	47	4	12	4	2	1
	FN	21	12	37	8	14	26
	Precision	0.52	0.89	0.45	0.93	0.92	0.92
	Recall	0.71	0.72	0.21	0.87	0.61	0.32
	F1	0.60	0.8	0.29	0.9	0.73	0.47
	Accuracy	0.99	0.99	0.99	1.0	1.0	0.99
EAR Thresholding: 0.3	TP	8	0	10	5	0	0
	FP	3	1	33	6	18	30
	FN	66	38	43	63	34	35
	Precision	0.73	0	0.23	0.45	0	0
	Recall	0.11	0	0.19	0.07	0	0
	F1	0.19	Null	0.21	0.12	Null	Null
	Accuracy	0.99	0.99	0.99	0.98	0.98	0.99
EAR Thresholding: 0.2	TP	32	11	22	47	31	30
	FP	299	55	185	108	14	15
	FN	45	28	26	21	3	6
	Precision	0.12	0.17	0.11	0.3	0.69	0.67
	Recall	0.42	0.28	0.46	0.69	0.91	0.83
	F1	0.19	0.21	0.18	0.42	0.78	0.74
	Accuracy	0.97	0.97	0.96	0.96	0.99	1.0
EAR	TP	50	14	28	54	31	32

Thresholding: 0.18	FP	620	28	101	44	5	6
	FN	30	26	22	10	5	4
	Precision	0.07	0.33	0.22	0.55	0.86	0.84
	Recall	0.62	0.35	0.56	0.84	0.86	0.89
	F1	0.13	0.34	0.32	0.66	0.86	0.86
	Accuracy	0.92	0.98	0.98	0.98	1.0	1.0

Table x. Evaluation statistics of each evaluation model deployed on the RealWorldBlink video dataset.

3.2. Gaze-lock detection model

As multiple models trained on different datasets with different settings, the best performance belongs to the latest model trained on our custom-made dataset with 5 subjects, our current evaluation progress showed an accuracy of 99.7% across the training dataset and 93% on the test cases of the same training subjects. We continued collecting custom datasets from 5 different people looking into and away from the screen for evaluation. From the dataset, the average F1 score reached 0.91 with the model giving out high precision and recall (**Figure 3**).

	Subject 1	Subject 2	Subject 3	Subject 4	Subject 5
Accuracy	88.96%	93.12%	90.97%	91.16%	90.03%
Precision	0.78	0.87	0.9	0.89	0.88
Recall	1.0	0.97	0.95	0.96	0.95
F1 Score	0.88	0.92	0.92	0.92	0.92
Average F1 Score	0.91				
Average Accuracy	90.848%				

Figure 3. The performance report of our gaze-lock detection model.

Reference:

- [1] Gowrisankaran S, Sheedy JE. (2015). Computer vision syndrome: A review. *Work*, 52(2):303-14. <https://doi.org/10.3233/WOR-152162>
- [2] Singh S, McGuinness M, Anderson A et al. (2022). Interventions for the Management of Computer Vision Syndrome: A Systematic Review and Meta-analysis. *Ophthalmology*, (2022), 1192-1215, 129(10). <https://doi.org/10.1016/j.ophtha.2022.05.009>
- [3] Pavel I, Bogdanici C, Donica V et al. (2023). Computer Vision Syndrome: An Ophthalmic Pathology of the Modern Era. *Medicina* (Lithuania), (2023), 59(2). <https://doi.org/10.3390/medicina59020412>
- [4] Rita I, Michel S, Vanessa A. (2023). Association of Computer Vision Syndrome with Depression/Anxiety among Lebanese Young Adults: The Mediating Effect of Stress. *Healthcare* (Basel). 2023 Oct; 11(19): 2674. <https://www.ncbi.nlm.nih.gov/pmc/articles/PMC10572235/>
- [5] Blehm C, Vishnu S, Khattak A et al. (2005). Computer vision syndrome: A review. *Survey of Ophthalmology*, (2005), 253-262, 50(3). <https://doi.org/10.1016/j.survophthal.2005.02.008>
- [6] Anbesu E, Lema A et al. (2023). Prevalence of computer vision syndrome: a systematic review and meta-analysis. *Scientific Reports*, (2023), 13(1). <https://doi.org/10.1038/s41598-023-28750-6>
- [7] Rosenfield M. (2011). Computer vision syndrome: A review of ocular causes and potential treatments. *Ophthalmic and Physiological Optics*, (2011), 502-515, 31(5). <https://doi.org/10.1111/j.1475-1313.2011.00834.x>
- [8] Bali J, Neeraj N, Bali R. (2014). Computer vision syndrome: A review. *Journal of Clinical Ophthalmology and Research*, (2014), 61, 2(1). <https://doi.org/10.4103/2320-3897.122661>
- [9] ESP32 technical reference manual. (n.d.). https://www.espressif.com/sites/default/files/documentation/esp32_technical_reference_manual_en.pdf
- [10] Lapa, I., Ferreira, S., Mateus, C., Rocha, N., & Rodrigues, M. A. (2023). Real-time blink detection as an indicator of computer vision syndrome in real-life settings: an exploratory study. *International journal of environmental research and public health*, 20(5), 4569.
- [11] Divjak, M., & Bischof, H. (2009, May). Eye Blink Based Fatigue Detection for Prevention of Computer Vision Syndrome. In *MVA* (pp. 350-353).
- [12] Jennifer, J. S., & Sharmila, T. S. (2017, January). Edge based eye-blink detection for computer vision syndrome. In *2017 International Conference on Computer, Communication and Signal Processing (ICCCSP)* (pp. 1-5). IEEE.
- [13] Lapa I, Ferreira S, Mateus C, Rocha N, Rodrigues MA. Real-Time Blink Detection as an Indicator of Computer Vision Syndrome in Real-Life Settings: An Exploratory Study. *Int J Environ Res Public Health*. 2023 Mar 4;20(5):4569. doi: 10.3390/ijerph20054569. PMID: 36901579; PMCID: PMC10001854.

- [14] Tereza S, Jan C. (2016). Real-Time Eye Blink Detection using Facial Landmarks. *21st computer vision winter workshop*, Rimske Toplice, Slovenia (Vol. 2)
- [15] Lapa, I., Ferreira, S., Mateus, C., Rocha, N., & Rodrigues, M. A. (2023). Real-Time Blink Detection as an Indicator of Computer Vision Syndrome in Real-Life Settings: An Exploratory Study. *International journal of environmental research and public health*, 20(5), 4569. <https://doi.org/10.3390/ijerph20054569>
- [16] Drutarovsky, T., & Fogelton, A. (2014, September). Eye blink detection using variance of motion vectors. *In European conference on computer vision* (pp. 436-448). Cham: Springer International Publishing.

Evaluation methodology:

1. Blink detection model: → Deadline 24/11/2024

- Deploy the model on a custom-created video dataset. The information about the video dataset is as follows:
 - + The dataset consists of 7 videos, ranging from 100 seconds to approximately 5 minutes. The resolution of each video is 640x480. The fps of each video is 30fps.
 - + The dataset consists of 7 subjects. All of them are asian. 6 of them are males, 1 is female. 2 of them wore glasses. The videos are recorded in different lighting conditions.
- After the model is deployed on that dataset, the following metrics are collected: true positives, false positives, true negatives, false positives, precision, F1-score, recall, and accuracy.
- The model is compared to 2 other models: the EAR thresholding model with 0.2 as benchmark model (this benchmark model is also used by other researchers).

2. Gaze-lock detection model: → Deadline 24/11/2024

- Deploy the model on a custom-created video dataset. The information about this video dataset is as follows:
 - + The dataset consists of x videos, ranging from x seconds to approximately x minutes. The resolution of each video is AxB. The fps of each video is xfps.
 - + The dataset consists of x subjects. All of them are asian. x of them are males, x is female. x of them wore glasses. The videos are recorded in different lighting conditions.
- After the model is deployed on that dataset, the following metrics are collected: true positives, false positives, true negatives, false positives, precision, F1-score, recall, and accuracy.

3. Automatic Brightness Limiting (ABL) → Deadline 24/11/2024

- The function is tested in the different lighting conditions, for each condition, the brightness in lux is recorded and pictures of the testing environment is collected for reproducibility:
 - + Morning (8-10am) -indoor-lights on.
 - + Morning (8-10am) -indoor-lights off.
 - + Morning-outdoor
 - + Noon (12 am)-outdoor
 - + Afternoon (5-6pm) -indoor-lights on
 - + Afternoon (5-6pm) -indoor-lights off
 - + Night (7pm -) - indoor-lights on
 - + Night (7pm -) - indoor-lights off
- For each experiment, the following metrics are collected: the ratio of the screen brightness and the environment after applying ABL to assess the accuracy; the speed in which the brightness was adjusted by the ABL (starting from when it is deployed to when

the brightness of the screen is adjusted to the calculated brightness) in seconds (compare with a threshold that classify as “acceptable”)

4. Complete application with all the functionalities running simultaneously: → Deadline: 26/11/2024

- **To verify the technical feasibility:** The app is deployed on 4-12 users for approximately 15 - 20 minutes; the participants are ask to complete a task during the experiment to simulate real-life scenario where the user are working on the computer; the task include playing music, using a search engine (Google Chrome) to search for an article and typing that article into a text application (Microsoft Word); during the experiment the following are recorded:
 - + The blink rate and blink drop notifications timestamps are recorded to see if the blink drop notification function is working properly
 - + The gaze time and 20-20-20 notification timestamps are recorded to see if the 20-20-20 notification function is working properly
 - + The brightness of the screen after ABL adjustment and the brightness of the surrounding environment is recorded to see if the ABL function is working properly
 - + CPU usage when the app is running to test the workload of the whole system
 - + Any error, delay in functionalities, etc are recorded for further assessment