528 Project Proposal

Safe Screen Distance Detection & Total Screen Time Estimation

Disha Singh | Maanya Mishra

---

1. Problem At Hand

We propose to help people view a mobile/laptop screen from a safe distance and keep them aware of the total time they spend in front of a screen thus managing eye stress.

2. Motivation

It is estimated that around 50-90% of the people around the world have some if not all symptoms of the *Computer Vision Syndrome.* This is a huge range of various eye strains and discomforts. More often than not, screens keep eyes moving up and down, left to right and changing focus quicker than reading a book. We also tend to dry up our eyes as we blink less often in front of a screen. Given the current scenario, computer/mobile screen are an essential part of life and spending long hours in front of them cannot be omitted. Well, if we cannot get rid of the screens, ***we can use the screens more responsibly***. This thought led us to develop something to help people manage their eye stress by alerting them when they are not within a safe distance from the screen. Thus asking them to move further away from the screen to put less pressure on the eyes.

3. Benefits/Future Use

Our work is directly useful for the user and places computing, hence UbiComp at the perfect direct interaction with the user. Our approach can not only be used to alert user to move further away, it can be furthered to measure screen time of the user, increase/decrease resolution of the screen based on distance of the eyes from the screen creating a dynamic screen.

4. Solution

We are planning to develop a browser extension that will be capable of running on a laptop. Our platform will actively estimate the distance of the user's face from the screen and alert them using a pop-up if they are too close. We will **first find the focal length F** of the camera by using a mathematical concept of triangle similarity. P is the apparent width of a reference object in pixels at the device camera. D is the known distance between the camera and the object. W is the actual width of the object.

F (focal distance) = (P X D) / W

Second, we will click a picture of the user, **detect user's face** in it and **find the distance** D' between the camera and the middle of the Interpupillary Distance (IPD) using:

D' = (F X Mean IPD of Human Face) / Apparent IPD of face(in pixels)

After this we will **compare D' with the standard safe distance** using this standard chart for a laptop screen(~1 - 1.7ft).

As a secondary goal, we will find **total time spent on the screen by estimating for how long the user's face was detected** in the app.

5. Existing Solutions to find distance

Microsoft Research has used sound waves to detect distance to the objects near a computer. Ultrasonic waves are emitted from the built-in speaker and based on Doppler effect measured by the built-in microphone, distance to the object can be found.

Some other solutions are using LiDAR to measure distance. Although most laptops have the sensor embedded, only the high-end iPhones have this capability yet. Other approaches also take multiple photos causing security and battery concerns.

None of them though, have provided useful feedback to people and have not used the estimated distance for eye stress management.

6. Advantages of our approach

We will only take a picture when there is some motion detected to avoid multiple images thus reduce battery consumption. We are making a browser extension which will provide useful feedback to people and not just tell the distance of their face from the screen. We also propose to take care of multi-user scenario.

7. Technical Details

- Our main challenge is that we are using mean width of face and not the exact width of face. We will try to use the camera view angle to eliminate the need for user's actual face width as described in this paper here.
- Also, OpenCV can detect eyes and not the pupil, therefore finding the middle of the interpupillary distance seems challenging.
- For screen time estimation, camera needs to be on all the time leading to privacy concerns. So, we will try to click photos only when their is some movement detected by OpenCV.

Platform: We will use OpenCV's built-in face detector to do our face detection and integrate it onto a JavaScript based browser extension.

8. User Study Design and Evaluation

Evaluation: We will use measuring tape to measure the actual screen-face distance and compare it with the distances measured by our system. We will do it with 10-15 people with varying face widths (if possible to find). We are tolerating an **error of ∓5% of the safe distance**. To evaluate screen time we will measure the estimated time with actual time calculated by an inbuilt iPhone stopwatch as and when user moves away from screen and back. A **0-1 Accuracy > 90%** would be a good measure for both tasks.

Data Collection: Since we are using OpenCV's pretrained classifier for face detection, we do not need to collect data for any of our tasks.

Ease of adoption: A browser extension which is easily installed with one click is our way to go as that will cater to a huge number of users, professionals or students. ***Real world deployment is not a part of this project*** for the sake of time, but is proposed as future work.

9. Project Plan and Timeline

Maanya: Estimate camera focal length & set up face detection (By 4/15). Integrate entire system with browser extension (By 4/27).

Disha: Set up browser extension (By 4/10), finding face-screen distance, creating safe distance & total screen time alerts (By 4/20).

Both of us will collaborate for the write-up and the video demo. (4/20 - 5/02)