Real Time Webcam Video to Screen Click Position Mapping with Eye Tracking

Julian Chu, Sharon Cheng, Richard Cheung
[juliantc, scsharon, rich13] @stanford.edu

Advisor: Mihir Patel
mihirp@stanford.edu

Overview

Eye tracking has many applications, and one of which is cursor input. However, existing methods are either flawed or expensive, and there are no existing work on mapping eye tracking to mouse click using ML.
Our goal is to **build a real-time mouse click application based on webcam and screenshot images using ML and computer vision**.

Hypothesis

- The multi-frame webcam video approach will improve the performance of eye tracking cursor since the user might not always look at the exact location of the cursor at the instance of mouse click
- The addition of screenshot input can improve model performance since the model can learn mouse click patterns in relation with specific screen features and thus provide context of user eye gaze to mouse click

Dataset & Features

~31000 samples collected by running our own data collection pipeline which collects 10 webcam frames before click in 10 fps and a laptop screenshot captured at the instance of click shown in **Figure 1a**.

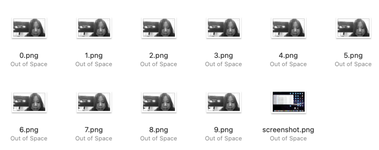


Figure 1a.

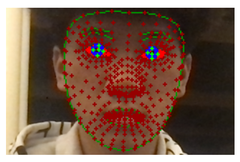
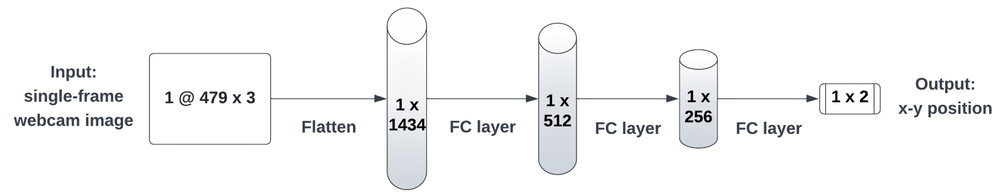


Figure 1b.

We extract facial landmarks using Google's MediaPipe Face Mesh + Iris model shown in **Figure 1b**, and RGB screenshot (3x80x50) is normalized. We generate normalized (x, y) mouse click coordinates as output.

Methods

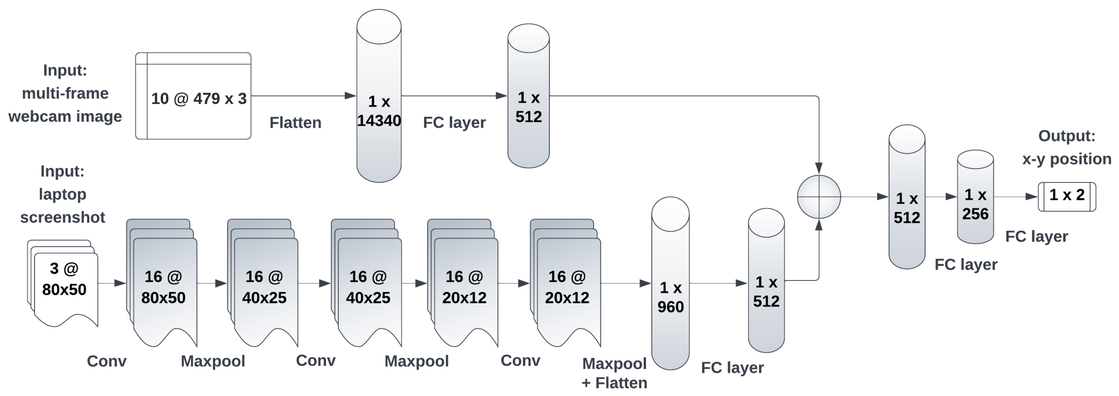
Baseline Model: Single-Frame Webcam with FC Net



Proposed Model 1: Multi-Frame Webcam with FC Net

- Same architecture as baseline but input is now a set of 10 frames of webcam images

Proposed Model 2: Multi-Frame Webcam with FC Net + Screenshot with CNN



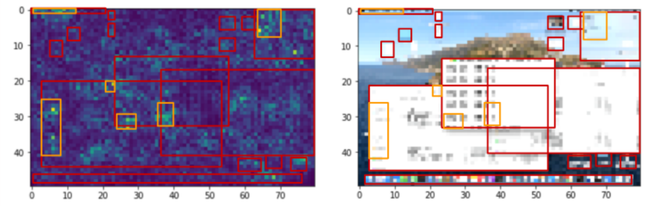
Experiemental Results & Analysis

Model	MSE Loss	avg. Dist (cm)
Baseline (M)	0.24	2.29
Multi FC (S)	0.26	2.3
Multi FC (M)	0.20	1.8
Multi FC (L)	0.16	1.95
With screenshot (S)	0.14	1.74
With screenshot (M)	0.11	1.5

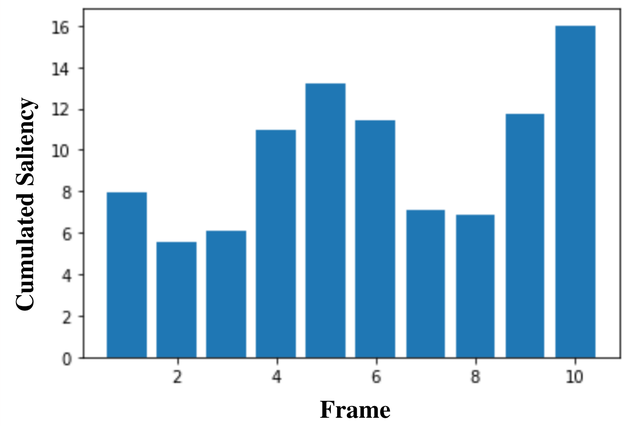
Figure 2. The mean squared error loss and average distance log for all 6 training instances of our 3 models: Baseline, Multi FC (Proposed Model 1), and With Screenshot (Proposed Model 2).

Training Logs of Models:
As shown in **Figure 2**, our baseline model generated the greatest lost and the greatest average distance whereas both our proposed models have enhanced performance with our proposed model 2 (with screenshots) demonstrating the best performance.

Discussion



Addition of screenshots:
Mouse click saliency map shows higher saliency in specific feature areas (e.g. window, files, menu bar)



A multi-frame model:
Cumulated saliency → higher accuracy of results.
Saliency difference in frame → user analytic.

$$\frac{1}{N}\sum_{k}^{N}\sum_{j}^{m}\left(\frac{\partial x_{l_{ij}}^{k}}{\partial y_1}+\frac{\partial x_{l_{ij}}^{k}}{\partial y_2}\right)$$

Figure 3. Equation for our saliency calculation

Future Work

- Improve screenshot processing modules (e.g. higher screenshot resolution, remove max-pooling)
- Preserving additional context about the screenshot (e.g. HTML content, buttons)
- Better ways of concatenating screenshot and eye tracking data (e.g. using attention)