EECS 225B Digital Image Processing
2022 Fall Class Project Proposal

Eye-Tracking System Under Different Lighting Conditions

Berkeley EECS
Chin-An (Daniel) Chen
Master of Engineering

Abstract

In this project, I am planning to implement an eye-tracking system that could detect the user's eyes in real-time under both normal and dim light environments.

Under normal light conditions, the overall pipeline would be first localizing the face within the image, then detecting the eyes within the face, and pointing out the center of the pupil in the end.

To further improve the system to support low-light conditions, I planned to implement a low-light enhancement module to be added right before doing the face localization, and the following modules are the same as the system under normal conditions.

Because this system targets detecting the user's eyes correctly in real-time, I would put more emphasis on the accuracy and the runtime to compare between different algorithms.

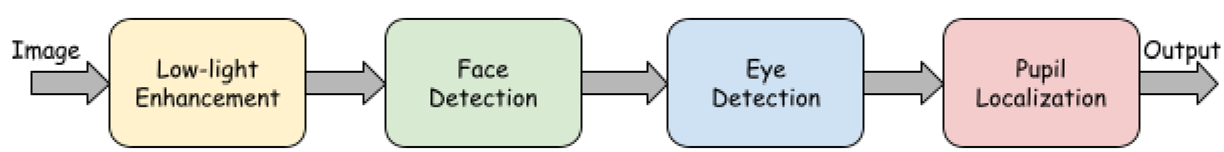
Backgrounds

I am working on the Vision Correcting Display(VCD) as my capstone project under the supervision of Professor Brain A. Barsky. By VCD, we aim to provide an image which will be seen in sharp focus by a viewer without the use of eyeglasses or contact lenses; that is, the text or image on the display is modified according to the measured vision problem of the viewer so that the transformed image will appear in sharp focus by the viewer. This could impact computer monitors, laptops, tablets, and mobile phones.

To realise VCD system, the first step is to find the user's eyes, as well as the gaze angles, for the following processes to render the re-focused display for the users. Hence, I decided to implement the real-time eye-tracking system using low-light enhancement models we learnt in this class to improve the VCD system to be supportive under different lighting conditions.

Algorithms

In this session, I divided the overall datapath into four stages shown below to realise eye-tracking system.



For each stage, there are several algorithms to satisfy the functionality. My plans would then be first implementing each of the algorithms, and connecting their inputs and outputs into the whole systems;

afterward, comparing their performances, such as runtime, accuracy for the face and eye detection, and how precise they could to point out the locations of the pupils, and deciding which combinations of eye-tracking system would be the most beneficial to the VCD system.
Below, I listed out all the candidates for each stage and the references.

- Low-Light Enhancement
  - Zero-Reference Deep Curve Estimation [8]
  - EnlightenGAN [9]
  - Dual Illumination Estimation [5]
  - LIME [6]

- Face Detection
  - Haar Cascade [1]
  - HOG frontal face detector with SVM [11]
  - MTCNN [2]
  - DNN frontal face detector [12]

- Eye Detection
  - 68 key points facial landmarks [13]

- Pupil Localization [4]
  1. Get the images of eyes
  2. Brighten the Iris and Pupil
  3. Binarization to isolate the pupil regions
  4. Contour the pupils
  5. Localize the centers of the pupils

Evaluation Metrix

To evaluate the different algorithm for the eye-tracking system, I also listed several metrics below.

- Runtime on CPU/ GPU
  - Because the system targets also on mobile devices, I would like to evaluate the runtime not using GPU but CPU, which could be closer to the real use cases.
  - Would focus more on CPU runtime.

- Face/ Eye Detect Accurary
  - Given several face/eye images, recording how many of them could be detected having a face/ eye within.

- Pupil Localization Accuracy

- Ablation Study
  - Comparison between adding low-light enhancement module and without the module

Reference

[1] P. Viola and M. Jones, “Rapid object detection using a boosted cascade of simple features,” Proceedings of the 2001 IEEE Computer Society Conference on Computer Vision and Pattern Recognition. CVPR 2001, 2001, pp. I-I, doi: 10.1109/CVPR.2001.990517.
[2] K. Zhang, Z. Zhang, Z. Li, Y. Qiao, “Joint Face Detection and Alignment using Multi-task Cascaded Convolutional Networks,” arXiv:1604.02878
[3] Face Detection Models: Which to Use and Why?
[4] Real-time eye tracking using OpenCV and Dlib
[5] Q. Zhang, Y. Nie, W. Zheng, “Dual Illumination Estimation for Robust Exposure Correction,” arXiv:1910.13688
[6] X. Guo, Y. Li and H. Ling, “LIME: Low-Light Image Enhancement via Illumination Map Estimation,” in IEEE Transactions on Image Processing, vol. 26, no. 2, pp. 982-993, Feb. 2017, doi: 10.1109/TIP.2016.2639450.
[7] Low Light Image Enhancement Python Implementation
[8] C. Guo, C. Li, J. Guo, C. Change Loy, J. Hou, S. Kwong, R. Cong, “Zero-Reference Deep Curve Estimation for Low-Light Image Enhancement,” arXiv:2001.06826
[9] Y. Jiang, X. Gong, D. Liu, Y. Cheng, C. Fang, X. Shen, J. Yang, P. Zhou, Z. Wang, “EnlightenGAN: Deep Light Enhancement Without Paired Supervision,” arXiv:1906.06972
[10] Barsky Lab
[11] CNN based face detector from dlib
[12] DNN-based Face Detection And Recognition
[13] V. Kazemi and J. Sullivan, "One millisecond face alignment with an ensemble of regression trees," 2014 IEEE Conference on Computer Vision and Pattern Recognition, 2014, pp. 1867-1874, doi: 10.1109/CVPR.2014.241.