Hough transform approach

# Outlines of the method

Given an input video the algorithm of our method is

1. Cature one frame of the video
2. Perform Viola-Jones eye localization
3. Discard the top 40% of the image
4. Perform binary thresholding using Otsu threshold
5. Perform edge detection on the thresholded image with Canny edge detection
6. Perform Hough circle detection and pick the most likely circle
7. Process the next frame (back to 1)

# Advantages & Disadvantages

The initial goal of our project is to create a algorithm that will perform in real-time and segment the iris from the face in the video. With this goal, our Hough transform-based method achieve the following advantages and disadvantages:

## Advantages

* Fast: the algorithm performs at almost real-time (30 frame per second) in Release mode without parallel implementation
* Easy implementation: eye detection and edge detection methods are readily available from OpenCV.

## Disadvantages

* Does not cope well with non-frontal irises (eye become elliptical instead of circle): this is an inherent drawback of our current Hough Transform step. Please see discussion for future work for our thoughts on fix this matter.

# Discussions of implementation

## Eye detection using Viola-Jones algorithm

The trained Haar classifier included in OpenCV allows for quick and easy detection of eyes. It works well under our indoor lighting video sequences. However, using it directly will include the eye-brows. One can employ a heuristic approach: removing the top 40% of the detected region in order to remove the eye brows. This approach works well for our video sequences. We did not experiment with re-training of the Haar classifier for only the eye. However, we hypothesize that eye-brow information is useful in detection of the eye. Thus, using an eye detection which are trained with eyes including eye-brows then removing the eye-brow section should be a preferred approach.

## Employing a tracker (e.g. Kalman filter)

We considered adding a Kalman filter step in our algorithm. However, since we want our algorithm to be real-time, adding an extra Kalman filter will slow it down below the real-time threshold. Adding a filter also mean our algorithm is not easily parallelizable as one frame need to be processed prior to the processing of the next.

## Using original images without thresholding

Our original implementation which results was showed during the demo requires quite a bit of parameter tuning which suggests that the approach is not robust to different images and settings. The original method does not take advantage of the skin color typically lighter/different from the eye.

## Using thresholding with Otsu theshold (targeting skin segmentation)

Otsu thresholding is a brute force method to search for the best threshold which minimizes the total intra-class variance. Otsu thresholding minimizes the following term:

Where weights are the probabilities of the two classes separated by the threshold t and sigma2 are the variance of the two classes.

Otsu thresholding removes the requirement of picking the right parameter for our binary thresholding step. However, it doesn't work well with skin under arbitrary lighting.

|  |  |  |
| --- | --- | --- |
|  | Good case | Bad case |
| Thresholded image with Otsu | D:\dc\Dropbox\My Dropbox\Courses\DIP\uhiris\ForReport\Skin segmentation\input_12.png | D:\dc\Dropbox\My Dropbox\Courses\DIP\uhiris\ForReport\Skin segmentation\input_1.png |
| Edge map of thresholded image | D:\dc\Dropbox\My Dropbox\Courses\DIP\uhiris\ForReport\Skin segmentation\edge_12.png | D:\dc\Dropbox\My Dropbox\Courses\DIP\uhiris\ForReport\Skin segmentation\edge_1.png |
| Found Hough circle | D:\dc\Dropbox\My Dropbox\Courses\DIP\uhiris\ForReport\Skin segmentation\iris_12.png | D:\dc\Dropbox\My Dropbox\Courses\DIP\uhiris\ForReport\Skin segmentation\iris_1.png |

One can see that bad results from thresholding using Otsu (i.e. including regions outside of the eye), do not mean the end result will be bad.

## Constraining the results of Hough transform

A typical image will return in several high peaks in the Hough space. It is important that we constrain the accepted peaks given our knowledge of the input. In our method, we employ the following constraints when searching for possible peaks in the Hough space:

## Using really high quality images

Using a very high quality image, we get a rather interesting (but bad) output (figure 1). Using a high quality mode in our Logitech Quickcam Orbit AF, we can get a really high resolution of the eye. In the image on the right, the user holds the camera as close to the eye as possible and still get the eye in focused. This creates a problem since the iris will reflect the scene in front of the subject. Such reflection is then detected by Viola-Jones detection algorithm. The algorithm will then give a bad region detection.

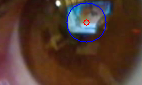


Figure : Eye detection get confused by reflection in the iris

# Future works

## Handling non-frontal irises

Using the average iris for human, then perform matching while considering iris as a patch lying on a perfect sphere.

### Handling higher quality images

We can use the information reflected from the iris for other purposes. (quote that paper) One can use artificial light in order to get information (i.e. artificial known lighting arrangements). Then we can use this information to detect the gaze of the user assuming that the light fixture does not move in space.

## Better estimation of detection result