|  |
| --- |
| COSC 6380 dIGITAL iMAGE pROCESSING |
| Iris Detection |
| Hough Transform and Topographic Approaches |
|  |
| **Dat Chu, Michael Fang, Paul Hernandez, Homa Niktab and Danil Safin** |
| **12/14/2009** |

|  |
| --- |
| This report describes the approaches we have tried and the problems we have encountered along the line towards realizing real-time iris detection and tracking |

Literature review

The problems of Iris detection, segmentation and tracking have been studied extensively, and there is a good amount of work that has been done towards this end. Hence we narrowed down our scope for literature review to more recent advancement and only those that do not require special instrument, i.e., IR illumination.

|  |  |  |
| --- | --- | --- |
| Year | Authors | Title |
| 2002 | Kashima et al. | **An Iris Detection Method Using the Hough Transform and Its Evaluation for Facial and Eye Movement** |
| 2003 | Kawaguchi et al. | Iris detection using intensity and edge information |
| 2004 | Cui et al. | An Appearance-Based Method for Iris Detection |
| 2004 | Perez et al. | Real-time Iris Detection on Faces with Coronal Axis Rotation |
| 2005 | Peng et al. | A Robust Algorithm for Eye Detection on Gray Intensity Face without Spectacles |
| 2006 | Niu et al. | 2D Cascated AdaBoost for Eye Localization |
| 2007 | Wang et al. | **Using Geometric Properties of Topographic Manifold to Detect and Track Eyes for Human-Computer Interaction** |
| 2007 | Akashi et al. | Using Genetic Algorithm for Eye Detection and Tracking in Video Sequence |
| 2008 | Chen et al. | A Robust Segmentation Approach to Iris Recognition Based on Video |
| 2009 | Xu et al. | Real Time Detection of Eye Corners and Iris Center from Images Acquired by Usual Camera |

\*The papers that we chose to implement are marked in bold.

Part I: The Hough Transform Approach

**Investigators: Dat Chu and Paul Hernandez**

# Outlines of the method

Given an input video the algorithm of our method is

1. Capture 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 an 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 achieves 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 fixing 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:

1. Only allow circles that are 10 pixel apart between their centers (to suppress spurious circles)
2. Only allow circles which radius is no more than 1/5 of the image height (remove big circles which correspond to the eye lids)

## 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.

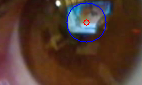


Figure : Eye detection get confused by reflection in the iris

# 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.

## Coping with non-frontal irises

Coping with non-frontal irises require a more flexible model than the Hough circle model. We are planning on investigating the ellipse Hough Transform approach in a future work. Another way is to treat the iris as the overlapped region of two circles and a line. Using this approach one can better segment iris in situation where the upper eye lid obstructs about half of the eye.

Part II: The TOPOGRAPHIC Approach

**Investigators: Michael Fang, Homa Niktab and Danil Safin**

# Outlines of the method

Given an input video the algorithm of our method is

1. Treat the intensity image as a 3D surface and fit a continuous function at each pixel location
2. Topographic labeling based on local gradient and principal curvatures
3. Apply SVM to identify eye-pairs
4. Keep track of the eye locations in the subsequent frame through mutual information matching

# Advantages & Disadvantages

## Advantages

* Does not rely on a face detector
* Handles certain amount of illumination/pose changes
* Fast once in tracking stage

## Disadvantages

* Initialization seems to be slow
* Difficult implementation comparing to the Hough transform method since pretty much everything needs to be written from scratch - no existing libraries that does topographic classification to date.
* Sensitive to parameter/threshold selection

# Discussions of implementation

## Least-square fitting

In order to perform topographic classification on the image we have to compute the local gradient and principal curvatures, which are information only available on a continuous surface. Therefore the first thing we want to do is to fit a continuous surface at each pixel location.



**Figure 2: input image and its gray values**

Assume that the continuous surface takes the form of two variable polynomial of some degree, a surprising fact is that the polynomial coefficients can be computed with a linear filter, independent of the actual data. The Savitzky-Golay filters are used towards this end. We found two implementations of Savitzky-Golay filters in MATLAB [Luo 2005][Krumm 2001] and implemented our own in C++ using Chebyshev polynomials [Meer and Weiss 1990]. Although they generate different filter values but all three give desirable surface fitting results. Figure 2 shows an input image and its corresponding gray-level image. Before computing the polynomial coefficients we first need to smooth the surface otherwise the presence of noise will make the surface fitting error-prone. Figure 3 shows a selected region with its surface before and after the smoothing. In our experiment, the Gaussian smoothing filter is 15-by-15, and=2.5, and is applied twice to get our smoothed result.

|  |  |
| --- | --- |
| C:\workspace\uhiris\ForReport\area.png | C:\workspace\uhiris\ForReport\noisy.png |
| C:\workspace\uhiris\ForReport\smooth.png |

**Figure 3: selected region and its surface before and after Gaussian smoothing**

The Savitzky-Golay filters we used are of size 5 and we assume the polynomials to be up to the second order, hence we have:

 (1)

We thus have:

,,,, (2-6)

Since we are evaluating the derivatives at the center of the patch, i.e.,=0,=0, we can analytically reconstruct the continuous surface (shifted by) from the first and second order partial derivatives, which can be obtained by simply convolving the smoothed image with the corresponding filters.

* Luo 2005: http://www.mathworks.com/matlabcentral/fileexchange/9123-2-d-savitzky-golay-smoothing-and-differentiation-filter
* Krumm 2001: http://research.microsoft.com/en-us/um/people/jckrumm/SavGol/SavGol.htm
* Meer and Weiss 1990: Smoothed Differentiation Filters for Images