IAT 455 – Final Project

Face Detection

Team 9

Jingya Huang | Ya-Chi Liu  
 301141913 | 301146403

April 4, 2014

Contents

[1 Overview 3](#_Toc384346338)

[2 Research 3](#_Toc384346339)

[2.1 Algorithm 3](#_Toc384346340)

[2.1.1 Edge Orientation Matching 3](#_Toc384346341)

[2.1.2 Modeling the face 4](#_Toc384346342)

[2.1.3 Distance Metrics 4](#_Toc384346343)

[2.1.4 Elastic Matching 4](#_Toc384346344)

[2.1.5 Hierarchical Image Search 5](#_Toc384346345)

[3 Work Distribution 6](#_Toc384346346)

[4 Final Result 6](#_Toc384346347)

[5 Conclusion 7](#_Toc384346348)

# Overview

We were working on face detection for this final project. Our primary goal was originally was to identify the number of people in a scene. We are achieving this by detecting the number of faces and followed the algorithm outlined in the conference paper, “Face Detection and Tracking using Edge Orientation Information” by Bernhard Froba and Christian Kublbeck. The project was built with Microsoft Visual Studio 2010 with OpenCV version 2.4.8.

# Research

As mentioned, we followed the algorithm outlined in Bernhard Froba and Christian Kublbeck’s paper. We also referenced OpenCV 2 documentation online and a couple introduction books on OpenCV from the library. We addressed our problems with OpenCV on stackoverflow and asked for some help from the TA.

## Algorithm

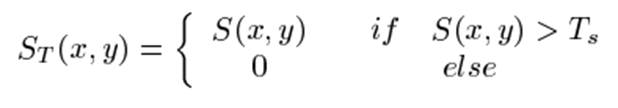
The algorithm addressed in the paper in a brief overview are:

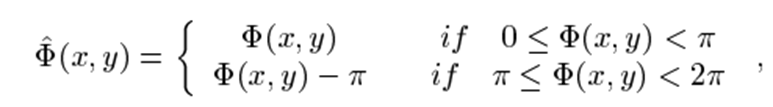
1. Edge Orientation Matching
2. Hierarchical Image Search
3. Training face model

### Edge Orientation Matching

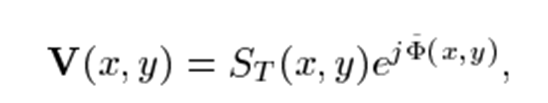
The paper uses Sobel edge operator for retrieving the edge strength and direction information. To calculate the edge strength and direction, they used the following formula:

After obtaining the edge information, they threshold on the edge strength to eliminate the noises and set the range of gradients to [0, π] to retain only the important information that they need.





Then they use this newly computed edge strength and direction, they store the edge information in a complex edge orientation vector **V**(x, y).



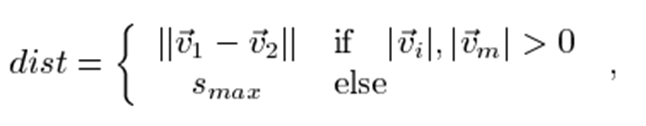
### Modeling the face

Our face model was extracted from “The face detection homepage” by Robert Frischholz (<http://www.facedetection.com/facedetection/datasets.htm>). The model that we chose for our program is shown below:



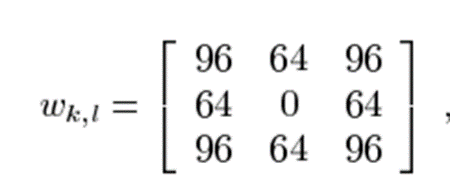
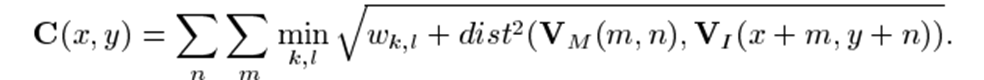
### Distance Metrics

To find the similarity between the original image and the matching image, the paper first calculate the distance function that compares the pixel value between the model and the image. The equation shown below is the function that we used in our program. We followed the direction of the paper and set smax=255 for our grey-scaled image.



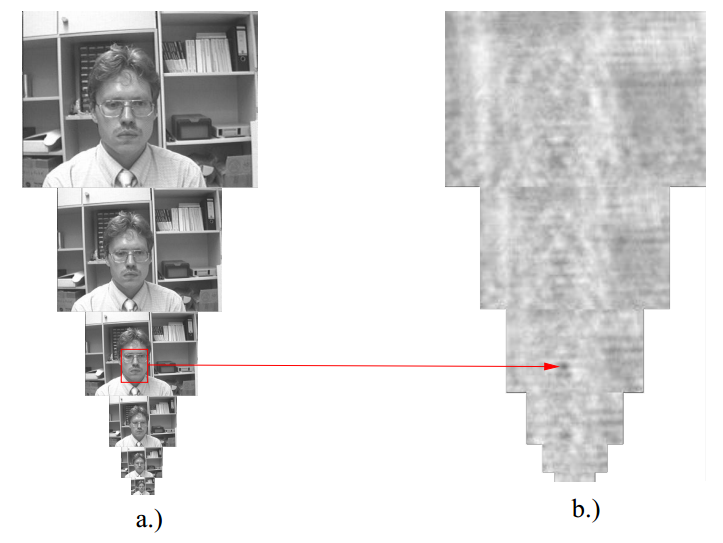
### Elastic Matching

The second step in finding the face from the given image is to find the minimum cost from the formula follows. The variables *k* and *l* are [-1 0 1] that finds the minimum between the current pixel and its other adjacent eight neighbours. wk,l is a weighting constant that has to be chosen manual with different images. We followed their convention to get the cost.

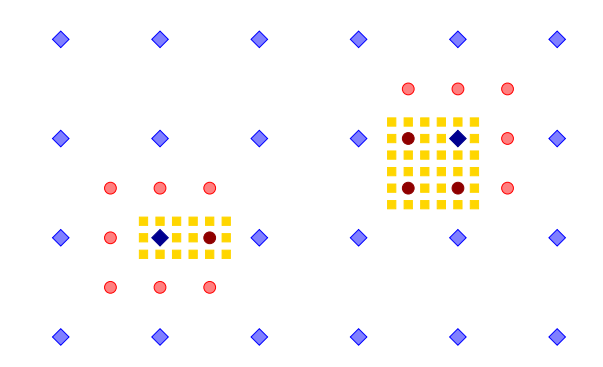


### Hierarchical Image Search

We didn’t have this part successfully implemented, but we found it interesting. To improve the performance on matching, the paper looks at different resolutions of the original image while the size of the model image stays constant. This allows us to detect faces of different sizes with a given scene.



Then to improve the speed up the image search, they perform matching on every 6th pixel (blue diamonds). If the matching result falls below a certain threshold, then the every 3rd pixel (red circles) will be examined. If again, the result is found below the threshold, the pixels along with all its 8 neighbours will be looked at.



# Work Distribution

Jingya worked on complex edge orientation vector and calculations on distance metric and elastic matching as described in the algorithm section.

Ya-Chi dealt with sobel edge information and failed to implement hierarchical image search.

A little github-ing: <https://github.com/yachi001/FaceDetection/network>

# Final Result

The source images and our model:

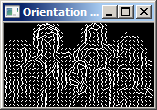
Our result of applying Sobel operator along the x and y direction using Kx=[-1 0 1; -2 0 2; -1 0 1] and   
Ky = [-1 -2 -1; 0 0 0; 1 2 1]. The kernel values is in reverse to the paper.



Result of the edge strength and direction from Sobel. Our direction information is most likely to be correct; we just couldn’t display it properly.



Our result of complex edge orientation vector as describe in the algorithm section.



Result of elastic matching



Our final result is obtained from thresholding elastic distance where the threshold value is set to 0.03. This thresholding step is in place of the section on hierarchical search.



[code attached: team9-code.zip]

# Conclusion

From the work of this project, we have experienced a face detection algorithm through implementations and that learned OpenCV may not have a solid documentation on their API. During the pair programming session, we also learned to use various debugging tools in Visual Studio 2010.