Oregon State University
College of Engineering

CS Capstone Technology Review and Implementation Plan

December 4, 2018

Pedestrian Counting and Privacy Preservation

Prepared for

Oregon State University

Dr. Fuxin Li

Prepared by

Group 9

Pavement Prometheus

Stephanie Allison Hughes

Abstract

The City of Portland is updating their data gathering system to better integrate data and technology into the decisions made by the city. They are hoping to detect pedestrian movement so future traffic paths can be created to increase safety. The task of my group is to provide data on, and hopefully a solution to, this issue. Mainly our concern is manipulation of data, so the collected data can be stored and analyzed without violating privacy portions of the city's social contract. My role in this project is to obtain facial detection data of pedestrians from street footage to be used to train a convolutional neural network.

CONTENTS

1 Introduction

The software described in this document using facial detection is a project under the advisement of Chanho Kim (Georgia Tech) and Dr. Fuxin Li (Oregon State University). The client for this project is the City of Portland, which wants a proof of concept for a way to transform the data from their traffic cameras so the city may store the data without storing identifying information about the citizens in the footage. My role in this project is to detect pedestrians through a computer vision program. Since I am in a larger group and my area of focus is more concise, I will be evaluating my role through different methods of facial detection and associated technology that runs under that methodology. This document will examine three possible methods of facial detection to gain data to train a convolutional neural network that can detect other pedestrians.

1.1 Individual Project Role

My individual project role will be to develop a computer vision program that can detect a pedestrian's face from video footage. I will be using video footage from a software simulation of a street seen with pedestrians to use as an initial data source. The data from the images will be used to train a convolutional neural network (CNN) along with my group members. I will be responsible for isolating the footage, getting the data points, and providing the detection data to help train the CNN.

1.2 Accomplishment Goals

The goal for the full project is to have a system that results in information on pedestrian movements that can be stored for open access by the public. The goal for my area of the project is to provide a program that can detect pedestrian faces through data from simulated video footage. A future stretch goal is the ability to provide data analysis on traffic patterns as well.

1.3 Method Criteria

The full pedestrian detection project is focused around detecting all people within video footage while also using low computing power. With this, the methods of facial detection will be judged primarily on their accuracy and simplicity of implementation. Bonus criteria that will be helpful to the development of the project will be speed and resources. The accuracy of a facial detection method will be centered around how well faces can be detected considering the strength of the detection evaluations and susceptibility to weakness in results. The simplicity of implementation will be evaluated on the method's ability to accomplish proper face detection with not too many different moving parts that could increase complexity and computing power. In the end, the best method will prioritize accuracy and simplicity, clearly fitting the needs for the project.

2 Knowledge-based

2.1 Description

Knowledge-based methods of facial detection are based off pointing out the relationship between different facial features [1]. A face is detected based on common features such as the eyes, nose, and mouth and their positions between each other. For example, if you have an image of a face, you would first identify the gradient intensity of the pixels on the face and the edges of the facial structure. From there the algorithm identifies certain features (eyes, nose, mouth, edges

of face etc.) from the pixel intensity values and stores the measurements between both eyes, the eyes and nose, the nose and mouth, etc. [1] Following this those measurements are compared with the measurements of different pictures of the same person. Once that person has been detected in different settings, then those assessments are tested to detect a face in a different person's image. Each time a new image is used, the calculation of the relationship between different facial features is refined to then be able to detect new unknown faces. The variables measured, facial featured used, and rules for refining data are all determined by the setup of the facial detection algorithm and those guidelines determine how faces are detected. This method is focused around human-determined rules for detecting facial features using the knowledge of specific distinguishing characteristics of human faces at the foundation of the algorithm [1].

2.2 Strengths & Weaknesses

Strengths of the knowledge-based facial detection method are that one has greater control over the parameters of the algorithm and detection guidelines are straight-forward. When developing and testing the facial detection program the programmers can determine what features of a face are more important to focus on [1]. When testing the program on different images there is a specific outcome to achieve - detect set facial features based on set guidelines. If there is a parameter that isn't being detected or is off in some way one can more quickly determine which guidelines are being violated. Since the regulations of the program are pre-set by the programmer, a knowledge-based method is more straight-forward to work with. Once the program is up and running, it should have a fast processing speed and be able to detect other faces if the rules are coded correctly.

Weaknesses of the knowledge-based method is that while the rules of the program are set initially, it can be difficult to fully translate those guidelines into code [2]. The algorithm is set based on a human's idea of the main facial features of a person's face, yet there are all kinds of different images to make those rules work well with. One can easily identify which rules might be violated but adjusting the code to have images more closely fit the guideline can be tricky. For example, an image of a face whose features are measured with the person looking straight versus a turned face or unclear setting may make the measurement guidelines not work as well. The knowledge-based facial detection approach gives a clear detection objective set by humans but turning those guidelines into a universal detection program may be more complicated than it appears.

2.3 Software

OpenCV is an open-source library that can be used within computer vision projects with many different machine learning tools, so it works well to aid in creating a knowledge-based facial detection program [3]. Developers can create a set of rules to determine the guidelines for specific facial features and work with over 2,500 algorithm resources from OpenCV to make those rules form facial detection standards in their program [3]. For the pedestrian detection project, OpenCV would allow me to focus on the refining a set of facial features to focus on when finding faces, then use different components of OpenCV libraries to ensure the Knowledge-based facial detection requirements are met.

3 Feature-based

3.1 Description

Feature-based method takes a bottom-up approach where the program detects facial features first and works to detect invariant features [1]. For this method, the features are detected through examining the edge, intensity, color, shape, etc.

of a facial feature [1]. The image is examined to find common variances in the face, grouping the components together to ensure they match. The facial feature groups are determined by spatial filters and the common grey level constraints [4]. For example, a nose may be detected from a face by examining the shape, intensity, and color of pixels centered in the middle of the face and compared with by the geometric constraints averaged by the sum of previous images. From there the probability that the feature we're examining is a nose is dependent on how similar its characteristics are to that of the faces before. Once all needed facial features are detected with a high likelihood of being correct, then that image is labeled as a face [4]. The Feature-based method is reliant on the common characteristics of facial features to piece together the detection of a face.

3.2 Strengths & Weaknesses

Major strengths of the Feature-based method are that facial features are more likely to be located despite the orientation of the face. If a person is faced to the side or ad a different direction, the program is not fazed as it is not looking at the overall face but the distinctive features [2]. This would make the speed of the program quicker as it would not need to run extra functions to detect faces from different angles, just the primary program. Since the method just examines the facial components and not the overall image it is simpler to detect a face versus other objects as you need all features to make the face.

Weaknesses of Feature-based facial detection is that the facial feature detection is dependent on a similar shape, color, intensity, etc. of the features so changes in the environment can affect results [4]. If a face is lit of in a different way or the image is grainy, the characteristics that determine the feature might be skewed. Also, if an environment is complex around the face then the program may have difficulty narrowing down the objects to examine and have a harder time distinguishing facial components.

3.3 Software

Face is software by Microsoft that uses the distinctive features of a face to mark a section of an image to be a person. The software examines 27 different elements of a face including the face edges, hair, eyes, eyebrows, mouth, nose, chin, and more [5]. Each facial feature is measured with an x and y value delivered in a JSON file [5]. This technology has a 70 percent accuracy, focusing on the facial features, making it a very accurate implementation [5]. For the pedestrian project, this software would be used to compare program results with and could be used as a base to develop a program that works well with the street footage given. Face compares an image with 20 other images per minute, which allows for its high accuracy [5]. The multiple data points from this software and the documentation within the Microsoft website gives the group plenty of resources to work with. The accuracy of Face is a major advantage to implementing a facial detection program with this software, yet it would need further evaluation to see if it's computing power usage is low enough.

4 Template-based

4.1 Description

Template-based facial recognition method detects faces using set measurements with all features plotted out together. The regions of the face are pre-defined instead of separately detected and are marked based on contours in the face [2]. The regions of the face form a set template that is hard-coded within the program and is compared to the shapes

found in the image [1]. For example, a template would consist of the shapes of the edge of a face, eyes, nose, mouth and other distinctive features. The template is then placed over the area of an image where the full set of features resembles those marked out in the guide. From there the measurements are compared between the different areas of the template face with the human face to see if the areas examined are close enough for the image to be marked as a human [1]. The Template-based method uses the correlation between the image and the set of features to perform facial detection, resulting in the comparison of a more complete human face.

4.2 Strengths & Weaknesses

Strengths of Template-based facial detection is that it is simple to implement and straight-forward in its usage. Since the template used to compare to the image is pre-defined, it is much easier to create versus a complicated multi-parameter comparison. The template distinguishes all different regions needed to create a full face, so it creates a set of standards for an image to be approved as a person [4]. The development of a template is clear to implement as once the measurements are set it is just down to the comparison aspect of the program versus constant re-evaluation.

Weaknesses of a Template-based approach are that it may not be very accurate and there are only so many templates that can cover all kinds of faces and poses [2]. The templates may work well in a stable environment where a face takes up more of an image, but with large images, it is more difficult for the template to accurately match with a face [2]. Also, humans have many different looking features and may be positioned in various ways, so generating a set of templates that universally covers these factors is not easy. With this, hard-coding enough templates take a very long time and does not seem realistic to generate quick enough results, also making the computing speed slow.

4.3 Software

The Facial Tracking DLL from Carnegie Mellon is an open-source project with many different facial detection libraries that use a Template-based approach [6]. The facial detection libraries contain different functions that can allow one to form their facial templates by setting the different region measurements [6]. With this I could develop a collection of facial templates that can detect faces with the street footage provided. There are also helper functions within the Facial Tracking DLL that would allow the templates to be optimized based on the images used to test the program. The Facial Tracking DLL would allow for simple implementation of a Template-based facial detection program, although the accuracy of the library functions is unknown and would need to be consistently verified.

5 Conclusion

5.1 Criteria Evaluation

1: Passes criteria **0:** Fails criteria

	Knowledge (OpenCV)	**Feature (Microsoft Face)**	**Template (Face Tracking DLL)**
Speed	1	1	0
Accuracy	1	1	0
Simplicity	0	1	0
Resources	1	1	1
Total	**3**	**4**	**1**

With this table we can see how the different scores the methods and their software stack up against the criteria evaluated. Feature-based methodology fulfilled the most requirements (4/4), followed by Knowledge-based (3/4) and last with Template-based (1/4).

5.2 Evaluation Discussion

In conclusion, Knowledge-based, Feature-based, and Template-based facial detection methods have various levels of accuracy and complexity, yet I consider the Feature-based approach to be the best fit for the project. The Knowledge-based approach allows for people to be detected from common guidelines although its implementation may be too complex for the time allotted, even with the OpenCV resources. The Template-based method could allow for quick implementation of a program to identify faces with the given Facial Detection DLL libraries, yet its accuracy is unknown. The Feature-based approach very accurately detects face and allows for simple implementation with the Microsoft Face detection resources. Although, the Feature-based method using Face may lead to more computing power depending on how quickly the images need to be processed. Overall, the Feature-based facial detection method's ease of development and validity are what influenced my decision to determine it as the best fit for the pedestrian detection project.

6 References

[1] D. Dwivedi, "Face detection for beginners," Available at https://towardsdatascience.com/face-detection-for-beginners-e58e8f21aad9 (2018/04/27).

[2] "Face detection (face recognition techniques) part 1," Available at http://what-when-how.com/face-recognition/face-detection-face-recognition-techniques-part-1/ (2017/06/14).

[3] "Opencv about," Available at https://opencv.org/about.html (2018/01/30).

[4] M. Chauhan and M. Sakle , "Study Analysis of Different Face Detection Techniques," *International Journal of Computer Science and Information Technologies*, 2014.

[5] "Microsoft face," Available at https://azure.microsoft.com/en-us/services/cognitive-services/face/ (2018/01/30).

[6] F. J. Huang and T. Chen, "Tracking of Multiple Faces for Human-Computer Interfaces and Virtual Environments," *IEEE Intl. Conf. on Multimedia and Expo*, 2000.