University at Buffalo
Department of Computer Science and and Engineering
CSE 473/573 - Computer Vision and Image Processing

Spring 2020
TuTh 9:30AM-10:50AM, Hoch 114

Project #3
Due Date: 5/13/20, 11:59PM

Contents

1 Projects # 3 Overview (100 pts Total)

The default topic for this project is face detection in the wild, described below in Section 3. You may work in groups of up to 2 students for the default project, and only need to submit the project once. If you do not pick one of the alternative projects, then you will be expected to turn in the default face detection project by the due date.

As an alternative we are offering a set of optional projects that require more computer vision implementation than the face detection project, but are not constrained to developing code from scratch. You may work in groups of up to 3 student on the optional projects. The projects will require you to build a computer vision application that can be demonstrated an must run in real time.

The list of possible projects include:

- A Touch-less Face Enabled Time-clock
- A Visual Welcome Center
- A Virtual Wall
- A Document Mobile Retriever

Each of them are described below in Section 4.

2 Project Selection: April 16th Deadline

You will have until April 16th to decide if you will choose one of these options or go with the default project. At the point that you decide, you will be committed the project you choose, so choose wisely!

To choose your project and partners, follow this link to a Google form.

473-573 Project #3 Selection Google Form

If you do NOT fill out the form, we will assume you are not teaming with anyone and that you will submit the default project.

3 Default Project: Face Detection in the Wild

The goal of this project is to have you implement the Viola-Jones face detection algorithm [1]. You should download the paper and understand its details. As discussed in class, the Viola-Jones algorithm is capable of detecting frontal faces in real time and is regarded as a milestone in the development of computer vision. Despite the fact that deep learning-based face detectors [2, 3] have gradually emerged as the preferred paradigm for face detection, variants of Viola-Jones algorithm still remain competitive in situations where speed is critical. A great introduction to Viola-Jones algorithm can be found at https://www.youtube.com/watch?v=uEJ71VlUmMQ.

3.1 Project Description

Given a face detection dataset composed of thousands of images, the goal is to train a face detector using the images in the dataset. The trained detector should be able to locate all the faces in any image coming from the same distribution as the images in the dataset. Figure 1 shows an example of performing face detection. We will use FDDB [4] as the dataset for this project. FDDB contains more than 2800 images and associated bounding box annotation, for more than 5700 faces.

Please keep in mind that Viola-Jones algorithm [1] is the ONLY method that you could use for this project. Using other methods - HOG / SIFT / SURF + SVM, deep learning-based methods, will result in a deduction of more than 50% of the maximum possible points for this project.

You must also use integral images to implement the feature extraction. You are free to use enhancements of the boosting algorithm such as cascades to obtain better results.

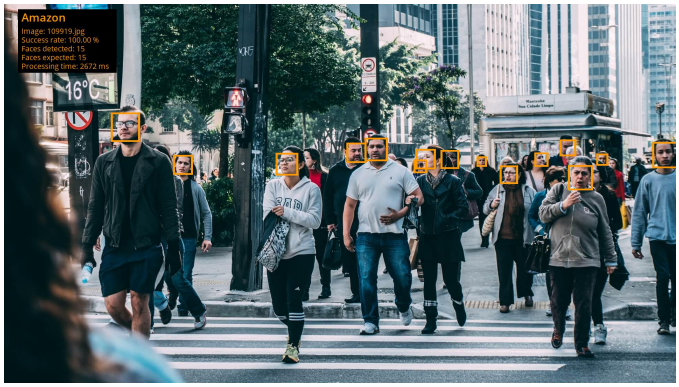

Figure 1: An example of performing face detection. The detected faces are annotated using orange bounding boxes.

3.2 Libraries permitted and prohibited

- Any Python Standard Library can be used.
- Any APIs provided by OpenCV that have "`cascade`", "`Cascade`", "`haar`" or "`Haar`" functionality can not be used.

- Using any APIs that implement part of Viola-Jones algorithm directly, *e.g.,* an API that computes integral image, will result in a deduction of $10\% - 100\%$ of the maximum possible points of this project[1].

3.3 Data and Evaluation

The FDDB [4] and the evaluation script should be downloaded directly from the following site: http://vis-www.cs.umass.edu/fddb/.

The following bash script will be used to evaluate the performance of the face detector you train:

```
unzip UBID-project3.zip
cd [UBID-project3]
python3 FaceDetector.py [data-directory]
python3 ComputeFBeta.py ./results.json [ground-truth.json]
```

`FaceDetector.py` contains YOUR python code that will able to detect faces in all the images in `[data-directory]` and generate a json file, *i.e.*, `results.json`, that stores all the bounding boxes of the detected faces. The generated json file shall be located in the same folder where the images located. The bounding boxes of the detected faces should be stored in a list using the following format:

```
[{"iname": "img.jpg", "bbox": [x, y, width, height]}, ...]
```

`"img.jpg"` is an example of the name of an image; `x` and `y` are the row index and column index of the top-left corner of the bounding box; `width` and `height` are the width and height of the bounding box, respectively. `x`, `y`, `width` and `height` should be integers.

`ComputeFBeta.py` contains YOUR python code that computes f_β using `results.json` and the ground truth in `[ground-truth.json]`.

You can refer to the sample json file posted under resource section on piazza for more information. There is also a piece of example code regarding how to create json file.

3.4 Code and Report

Your code should be zipped up in a directory called Project3. As discussed in class, unlimited number of submissions is allowed and only the latest submission will be used for grading. Since this project allows more flexibility in the use of libraries, please also remember to include a file with your resources your code requires to make it easier on the grader. This should be in the form of a txt file named "requirements.txt". The file named "requirements.txt" should specify the libraries you used and the version of the libraries, and our grader should be able to install all the libraries you used by using the command: pip install -r requirements.txt

In addition to your code, a report is required with this project. It should contain

- Your name and your UBID, and the names and UBID of your project partners.

[1]Please do not ask if a specific API could be used on Piazza. You need to judge whether an API implements part of Viola-Jones algorithm directly on your own.

- An overview of the Viola-Jones algorithm,
- A description of your implementation,
- Results of your face detector on FDDB (from the program above) and
- An analysis of the results (failure cases, possible improvements, etc),

The report should be a pdf file. You will be graded on both the report and an evaluation on a held out test dataset that you currently do not have access to.

4 Alternative Projects Options

Congratulations! You have just been hired as an image processing and computer vision engineer at Victor E. Bull Inc. This is a startup company that has cool ideas for vision products that they want to use to change the world. Since you've taken computer vision at UB you have been selected to lead a team to develop, implement and test one of the projects listed below. It's your choice on which ones to start with but you must work with the company president, Dr. Doermann, during office hours or extended office hours to develop the specifications, agree on capabilities, and design test and evaluation criteria. Dr. Doermann has very limited knowledge of how this should be done, so that part is up to you, but he has a vision for what the industry wants. Check out the following projects and see if you would like to choose one of these alternative project options. In the end, the project will require some basic description of what the functionality is, the users and technical integrators document, working source code, and a qualitative evaluation. The nice thing is that since you are engineers and you've already passed the class, you can use any capabilities that are in the public domain two implement your system. You just cannot use any code that specifically accomplishes all of the goals of the project.

You must submit the project and demonstrated your new system to the president by the last day of final exams in May. Choose carefully, because your future with this company depends on it.. . .

4.1 Touch-less Face Enabled Time-clock

In this project you will develop a time clock and time recording system that relies on the recognition of individuals enrolled in the system to keep track of their hours. The system will not have any touch based interaction, but may allow interaction of a person with voice commands, head movement or hand gestures such as thumbs up or down.

System should consist of an interface that allows the enrollment of a person, and a back-end database that keeps track of information about the person themselves as well as their time records.

When a person approaches the clock the system should attempt to recognize them by detecting the face and matching them against previously enrolled individuals in the database. Once the system recognizes a person, it should display the identity of the person and wait for the person's confirmation using voice commands, head movement, hand gestures or other touch-less interaction methods. The system should clock in-out only after receiving the person's confirmation.

The system should then ask them to verify that they are clocking in or clocking out as appropriate, and record other attributes such as breaks for coffee or lunch. If they want to clock out without clocking in, they should be allowed to record an "exception" that will be checked later through the interface.

It should also keep a picture with a time of each interaction.

Other features of such a system might include:

- The ability to analyze the database to look for anomalies, such has
- The ability to report and time summarized by day/week (or period)

4.2 Visual Welcome Center

In this project, you will develop an interface to interact with visitors to the Department of Computer Science and Engineering at UB. The system should have back end and where people can be enrolled manually, along with he information such as their title, affiliation, and email address, website etc.

When a visitor approaches the system they should be identified if they are already in the system and one of a set of welcome messages that are already in the system should be displayed along with any other information you deem appropriate.

If they're not in the system it should also welcome them and asked them if it is ok if you enroll them in the system. You could capture this information, for example with a hand gesture (thumbs up or down) or a verbal command from a limited vocabulary (yes or no for example. If possible you could also capture a business card and enter this information in the record.

It will be up to you to define other features of such a system, but some such features might include:

- Each time a user is recognized and verified by the system should make a record of the time and date when they came in, and a new photo to eventually enroll
- You might consider an OCR component that could recognize names on business cards held up or captured by a desk camera.

4.3 Virtual Wall

This project will involve the creation of a virtual wall the users can interact with. You will be provided with a stereo camera that can take a picture of a wall at an acute angle. From the stereo camera it is your job to write the software to track a person's hand and allow them to point at particular locations on this wall.

For this project the virtual wall can be simply a tapped off rectangle region that sometime in the future could be the projection screen. At a minimum that screened area should be divided into nine parts and you should be able to recognize when anyone approaches and gets within a fixed distance of the wall, and points to a particular element/cell. The closer you can get to an exact relative XY coordinates on the virtual wall the better.

You may want to interact with the user in some meaningful way, such as playing a sound when they touch the wall, or asking them to touch a block numbered one through nine, so you can test the system.

The higher the resolution and the more reliably the system works, the bigger your bonus!

Here is a link to the stereo camera that will be available to your team is described here: https://www.mynteye.com/products/mynt-eye-stereo-camera

Here are some other features you may consider:

- Some way to provide feedback to the users through sounds or confirmation
- The ability to know the position**s** where two or more people are pointing at the **same** time.

And here are some things you can assume:

- You can assume that the camera is fixed at an acute angle to the wall (from the side is fine).
- You can calibrate the scene by clicking on the 4 corners of the virtual wall before starting.

4.4 Mobile Retriever

In a recent class at UB, Prof. Doermann talked about a mobile retriever system that was implemented by one of his graduate students over a decade ago. Although this technology was never really commercialized, it's a very interesting idea. The idea is that when documents are created electronically and posted or published in hard copy form, there may be a desire to retrieve the original content, for example to be bred back as a podcast, or simply to find the original document electronic form.

It is your job to take this paper and implement the major components of the system. This includes taking a large set of documents such as PDFs of scientific literature, and converting them to images. You must then implement the image processing routines needed to index it as described in the paper. You will then implement the techniques for retrieval including the triplet verification.

Although the original paper talked about an implementation on a mobile phone, any type of WebCam will work for this project. But it must be integrated you must be able to take a snapshot of a document and find it in a collection of say a minimum of 1000 pages.

If you are eager enough, you can find another way to do this has techniques and computer vision have advanced tremendously in the last decade. But please keep in mind that the goal here is a real-time retrieval and there are constraints on the problem that you should take advantage of.

Here is a link to the original publication and it will also be provided online in case you have trouble retrieving.

Don't underestimate the challenges that may arise. You should be an excellent programmer to take this one on.

X. Liu and D. Doermann,"Mobile Retriever: access to digital documents from their physical source," International Journal of Document Analysis and Recognition (IJDAR), vol. 11, no. 1, pp. 19–27, 2008.

https://link.springer.com/content/pdf/10.1007/s10032-008-0066-4.pdf

4.5 Requirements for Alternative Project options

- For the optional project, they are all expected to run in real time to demonstrate the functionality, not just on a recorded video.
- You should submit the following:
  - A cover page with your name and your UBID, and the names and UBID of your project partners.
  - A 1-2 page description of your system suitable for advertisements - ie a description of the capabilities
  - All source code
  - Good documentation of code or an integrators document (what software needs to be installed and how to use it)
- A short video of your system in action
- A short description on what you felt where the major challenges you overcame and what you would have improved if you had more time.

- Your project should be uploaded to UBLearns.

5 Grading and Assessment

The grading of this project will be graded in part on the approach and design of the project and in part on the performance.

5.1 FaceDetection

- Performance (50%) - FScore of > 0.8 will receive full credit
- Code Base (25%) - Implementation of Training, Haar Features, Integral Image, etc will receive at least 15 of these points
- Report (25%)- A detailed report including discussion of what worked and did not work, even if the project does not perform well will receive full credit. A short few line report or no report will be docked significantly,

5.2 Alternative Projects

- Performance (50%) - The system works in real time as designed.
- Code Base (25%) - Documentation and choice of modules for creation of the system are expected
- Report (25%)- Three documents are required for these projects as indicated above - Design, Integrator and after project. Detailed reports including discussion of what worked and did not work, even if the project (code) does not perform well, you will still be able to receive full credit for your report. Short few line reports or no reports will be docked significantly,

6 Academic Integrity

There are more than a dozen open source python implementations of Viola-Jones algorithm, including but not limited to those listed on https://github.com/topics/viola-jones?l=python. Copying and pasting code from open source implementations is strictly prohibited. For details about academic integrity policy of UB, please refer to https://catalog.buffalo.edu/policies/integrity.html.

References

[1] P. Viola and M. Jones, "Rapid object detection using a boosted cascade of simple features," in *IEEE Conference on Computer Vision and Pattern Recognition (CVPR'01)*, 2001.

[2] Z. Liu, P. Luo, X. Wang, and X. Tang, "Deep learning face attributes in the wild," in *IEEE International Conference on Computer Vision (ICCV'15)*, December 2015.

[3] Y. Sun, X. Wang, and X. Tang, "Deeply learned face representations are sparse, selective, and robust," in *IEEE Conference on Computer Vision and Pattern Recognition (CVPR'15)*, June 2015.

[4] V. Jain and E. L. Miller, "Fddb: a benchmark for face detection in unconstrained settings," 2010.