Software Specification
Document

for

Crowd_Count++

Version 1.0
November 2015

Prepared by:
Juan Mejia   Michael Safdieh   Rosario Antunez
Table of Contents

1. Abstract & Keywords
   1.1 Abstract
   1.2 Keywords
2. Overall System Description
   2.1 System Perspective
      2.2.1 Backend Components
      2.2.2 RESTful API and User Interface - Front End Components
3. Related Work
   3.1 Multi-Source Multi-Scale Counting in Extremely Dense Crowd Images [4]
   3.2 Fast Crowd Segmentation Using Shape Indexing [3]
   3.3 OpenCV for Object Move Detection [1]
4. Benefits for Society
   4.1 Overall Benefits
   4.2 Real-World Applications
      4.2.1 MTA
      4.2.2 DOT
5. High-level Description
   5.1 Testing and Approach
      5.1.1 Testing of Crowd Counting
      5.1.2 Object Detection Approach
   5.2 Backend Components
      5.2.2 Image Processing & Indexing
      5.2.3 Database
   5.3 RESTful API & User Interface - Front End Components
      5.3.1 RESTful API
      5.3.1 User Interface - Front End
6. Collaborations & Future Task Distributions
   6.1 Collaboration
   6.2 Future Task Distributions
7. Concluding Remarks
8. Bibliography
1. Abstract & Keywords

1.1 Abstract

The problem of counting the number of people in images and videos arises in several real world applications including crowd management, design and analysis of buildings and spaces, and safety and security. In certain scenarios such as public rallies, marathons, public parks, and transportation, the number of people in the given scenario is of direct importance. This process usually involves fixed cameras, allowing the use of background modelling and change detection algorithms which produce a change measure at each pixel in a video stream. The resulting distance images, along with the original images, can be used to detect humans and their movements in the scene. While isolated humans can be easily detected, groups of people close together and partially occluding each other are more difficult to localize and track. We plan to unit test Crowd_Count++ by counting the number of people in a classroom using images provided by the City College of New York. Counting individual humans in a crowd forms the focus of this Senior Design Project.

1.2 Keywords

Image Processing, Machine Learning, Image Indexing, OpenCV Algorithms, Object Detection, Crowd Counting, RESTful API

NOTE:
This report adapted some layout ideas from “Software Requirements Specification for Splitpay” [5].

2. Overall System Description

This section focuses on the basic description of the Crowd_Count++ software system.
2.1 System Perspective

Crowd_Count++ is aimed at counting the amount of people in a crowd, in a room, at a bus station, or in almost any other situation. Crowd_Count++ is primarily a back end system, and, as such, the description of the system primarily focuses on the different backend components. The system will have a web-based front end side to it as well.

2.2 System Functionality

2.2.1 Backend Components

The backend of Crowd_Count++ is to be comprised of three main modules: the Standardized Image Generation module, the Image Processing & Indexing module, and the Database module. The raw data to be fed into the Standardized Image Generation module is video footage of a crowd of people. The module will take the footage frame by frame, and from it produce a collection of standardized images (with the size being 1024 x 1024 pixels).

These standardized images will then be fed into the Image Processing & Indexing module. It is here where the system will use indexing and image processing techniques to count the number of people in a given image. The number of people detected in the images, along with other important information will then be sent to the Database module.

The Database module will contain all the information and results produced by the system. It will be used as a center for all the newest data gathered by the system, and will be available for querying by the users.

2.2.2 RESTful API and User Interface - Front End Components

In order for the front end of the system to be able to display the results of the video footage, an API is used as a communicator between the front end (i.e. the user interface) and the database (backend). When a request is sent by the front end to access information from the database, the API connects to the database, retrieves the result of the request, and sends it back to the front end of the system.

The front end of Crowd_Count++ will provide a simple, user-friendly interface for displaying and querying requested data. It will include a real-time feed of a crowd count
of a specified location in the form of a map. Additionally, the front end will include historical counts of crowds on past dates, and a help section containing documentation on how to use the application and information on the RESTful API that will be used to retrieve information.

3. Related Work

Some of the existing literature relevant to the product to be developed and applications is briefly reviewed in this section.

3.1 Multi-Source Multi-Scale Counting in Extremely Dense Crowd Images [4]

This article is about computing an estimate of the number of people present in an extremely dense crowd having a single image. Because of the image density, the problems including perspective, occlusion, clutter, and the fact that there are only few pixels assigned to a person, traditional techniques such as counting by human detection is almost impossible. Haroon Idrees and Imram Saleemi from the Center of Research in Computer Vision at University of Central Florida and Cody Seibert and Mubarak Shah from the Department of EECS at the same university designed a solution to the current problem. Their approach relies on multiple sources such as low confidence head detections, repetition of texture elements, and frequency-domain analysis to estimate counts. Additionally, the authors employ a global consistency constraint on counts using Markov Random Field. This caters for disparity in counts in local neighborhoods and across scales. The authors tested their approach on a new dataset of fifty crowd images containing 64K annotated humans, with the head counts ranging from 94 to 4543. This is in stark contrast to datasets used for existing methods which contain not more than tens of individuals. Subsequently the authors demonstrate through experiments the efficacy and reliability of the proposed approach by quantifying the counting performance.

3.2 Fast Crowd Segmentation Using Shape Indexing [3]

This paper discusses a fast method for the problem of estimating the number of humans and their positions from background differenced images obtained from a single camera where inter-human occlusion is significant. There are a few factors that make
the problem challenging and difficult, such as the state space formed by the number, positions, large articulations of people, noisy background differencing, imprecise processes, and output that is far from ideal (holes, fill-ins, irregular boundaries). Authors Lan Dong from the Department of Electrical Engineering at Princeton university and Vasu Parameswaran, Visvanathan Ramesh, and Imad Zoghlami from Real-Time Vision and Modeling Department at Siemens Corporate Research propose an example-based algorithm which maps the global shape feature by Fourier descriptors to various configurations of humans directly. They use locally weighted averaging to interpolate for the best possible candidate configuration. The inherent ambiguity resulting from the lack of depth and layer information in the background difference images is mitigated by the use of dynamic programming, which finds the trajectory in state space that best explains the evolution of the projected shapes. The key components of the solution are simple and fast. They use unit testing to demonstrate the accuracy and speed of the approach on real image sequences.

3.3 OpenCV for Object Move Detection [1]

Computer vision, an area of image processing, has grown rapidly in recent years since hardware (cameras) is cheaper and processing power is faster and affordable. OpenCv has contributed to the advancement of computer vision by offering learning algorithms which previously were not possible due to hardware constraints. Today OpenCv with real-time processing power is the main software used by the science and research community. OpenCv enables the detection of objects by using different known algorithms such as convolution/filters, threshold, histogram and matching, contours, and efficient nearest neighbour matching. At Carleton University in Ottawa, Canada, OpenCv has been used to find moving objects from a video camera. The image is captured as a frame and converted into an 8-bit grayscale image. Then, the movement of certain pixels is analyzed in different frames in order to know the current location of the object. OpenCv will continue to contribute in the development of better algorithms that have a great impact in real world applications such as video indexing on the web, product inspection, medical imaging and robotics perception.
4. Benefits for Society

4.1 Overall Benefits

Understanding consumer behavior is crucial for today’s world. It is of great importance to have a tool to assist the knowledge of crowds in different settings. Having this system implemented will make a difference in the quality of a many services being used by a considerable amount of people such as public parks, transportation, libraries, marathons, etc. The software assists entities to estimate the number of people present in some location. The use of image processing techniques to be able to index a person in a unique way as well as the use of machine learning techniques to be able to study previous data and new data to establish and estimate the number of people are beneficial to computer science and engineering in general. Moreover, through the implementation of this system, we aim to make an efficient use of all what we have learned in college so far to serve our community.

4.2 Real-World Applications

Our system will be built specifically upon footage from the Senior Design I classroom, and, with permission, footage from the City College of New York’s administration building. Although this is limited raw data, the Crowd_Count++ system can be adapted to for many different applications that require counting the number of people in a given location.

4.2.1 MTA

One such example is the MTA buses in New York City. Each MTA bus has several video cameras taking footage of its passengers. Being able to feed that footage into a software system that counts the amount of passengers on a bus between stops is very useful. Knowing the amount of people at any given time on the bus would enable the MTA to make better decisions when dispatching buses from their depots, and thus creating a more efficient transportation system for the citizens of NYC. This method would also be cheaper than installing hardware sensors on each bus.
4.2.2 DOT

Another example for adaptation is the Department of Transportation (DOT) of New York City [2]. The DOT organizes counts of pedestrians in 114 locations two times every year. Currently the counts are carried out manually by enumerators. Implementing an automated pedestrian counter by using video cameras, could extend to taking counts every day and have available data every time they need it.

5. High-level Description

In this section, a more detailed description of the software system is given than the one in Section 2 (along with a general diagram of the system), among other important information.

Below is a diagram of the main components (and the interactions between them) of Crowd_Count++:
5.1 Testing and Approach

It is important to mention the steps that will be taken to retrieve the raw video data as well as the approach used.

5.1.1 Testing of Crowd Counting

Testing will be done using a control group and an uncontrolled group.

Control Group: A camera will be installed by the door of the Senior Design I classroom where the implemented system will estimate how many people are in the room. In order to test the accuracy of the system, a physical count will also be taken and compared against the count given by the Crowd_Count++ system in order to verify correctness.

Uncontrolled Group: Video footage from the CCNY administration cameras will be used (if permission is granted) to see how the system would perform in an uncontrolled environment. Crowd_Count++ will take continuous counts through the day and update the database.

5.1.2 Object Detection Approach

An object detection algorithm will be implemented to detect a human, rather than a facial recognition algorithm. Every person will be treated as an object, where the algorithm can detect unique features associated with the detection of a human-being. The use of this approach has a twofold reason. The first is that tracking facial features infringes on people’s privacy. Crowd_Count++ is only intended to count the number of people in a given area, not identify them. The second reason is that facial recognition is very complex in itself, and would therefore add a great deal of unnecessary work to an already difficult task.

5.2 Backend Components

5.2.1 Standardized Image Generation

- Raw data from video camera footage is input into the Standardized Image Generation module.
The module breaks up the video frame by frame into individual images to be processed.
Each image is converted into a standard size of 1024x1024 pixels, and into a uniform image format (such as .png or .jpeg). This is done in order to ease the process of counting the number people per image. If the images are not standardized, the Image Processing & Indexing module would have to process each image differently in order to correctly count the number of people in it.
These uniform images are then collected and sent to the Image Processing & Indexing module.

5.2.2 Image Processing & Indexing

This module is the core of Crowd_Count++, as this is the component where the actual counting takes place. The module also computes other data about the images given. The processing will work in the following manner:

- The images from the Standardized Image Generation module are received.
- Images that appear to be redundant are disposed of in order to process more significant images efficiently.
- The images selected are to undergo thorough processing. The module will use an object detecting algorithm as well as various image processing and indexing techniques in order to best detect the presence of human beings in each photo.
- A number is given to each image corresponding to the number of people detected in each respective image.
- Additional data, such as the time of the image, will be associated with each image appropriately.
- All of the data relating to each image is then sent to the Database module.

5.2.3 Database

- The database is to house all of the data collected from the images processed from the previous modules.
- The amount of people detected per image is to be stored, as well as the time of each image.
- The database is to be updated as new information is processed.
● Additional information about general trends of certain time intervals will be stored in the database, such as long periods of time when there was no change in the number people in a room, periods of time when the number of people in the room changed often, and more.
● The database will comprised of many tables in order to store all of this data in an organized and efficient manner.
● The database will answer queries received from the front end through the API.

5.3 RESTful API & Front End Components

5.3.1 RESTful API

The API will act as a bridge between the front end and the backend of the Crowd_Count++ system. It will work in the following manner:

● The API will receive a request for information from the database. This request comes from the user at the front end.
● The API translates this request and queries the database for the requested information.
● The API then receives a response from the database.
● Finally, the API translates this response, and sends it to the user at the front end where it is to be displayed.

5.3.1 User Interface - Front End

The front end of the system is relatively simple and user friendly.

1. Homepage (Real Time Feed)
   ○ On the homepage, the user can make a request for the most current information from the database.
   ○ Additional options such as setting time intervals can be selected in order to show the number of people in the given room at those times.
   ○ After the request has been made, the page would display the results in a map using a geolocation format.

2. Historical Counts Page
   ○ The user can also view a page that archives past days counts of people and times.
The user would select a day and time and the page would display the corresponding records from the database in the same format of the homepage.

This page is useful to gather statistics and draw conclusions about the data.

3. Help Page

The user can also choose to view the help page.

This page contains documentation on how to use the application and information on the RESTful API used to retrieve information.

It is important to note that the details of the system are subject to additions and modifications. These modifications will likely take place during the design and implementation phases of the Crowd_Count++ system.

6. Collaborations & Future Task Distributions

6.1 Collaboration

First we discussed the entire document and agreed on what to include in each section. All the team members had some input into the specification that is being presented. The work on this report was divided among the group members in the following way:

Juan: Usages of OpenCv in Image processing, object detection, Benefits for society, Front End General Overview and Communication to the Back End Using Restful APIs, Unit Testing and Future task distributions.
Michael: Overall description of the system, High level description of the system, Benefits for society, Collaboration, Bibliography

Additionally, each group member had a hand in editing the document.

6.2 Future Task Distributions

Rosario, Juan and Michael: Installation of Video Camera in our Senior Design Class.
Request Video footage for educational purposes from The City College of New York Administration.

**Michael and Rosario:** Detailed Design of the Crowd Counting Database (Backend Development).

**Juan:** Detailed Design of the Frontend.

**Rosario, Juan and Michael:** Standard Implementation of RESTful APIs using Java.

**Juan, Michael:** Implementation of the Frontend using HTML, CSS and Javascript with AngularJS.

**Juan:** Unit testing for the Front End using Protractor.

**Rosario, Michael:** Unit testing for robustness of the Backend to accept multiple connections.

**Rosario, Juan, Michael:** Testing of algorithm for counting people.

**Rosario, Juan and Michael:** Detailed Documentation about the application.

### 7. Concluding Remarks

While writing this document, we learned that the product specification is of great importance when developing a software system. The fact that we have to put in writing all the details required helps the team have a better understanding of the design as a whole. In order to write this document we had to exchange ideas of what to include in each section, which forced the team to communicate effectively. Some important lessons that we learned from developing this document are: research about the subject is crucial, and incorporating the input of all the teammates is a big challenge as each teammate has to be understood by the other and an agreement must be reached on how to proceed.

Some of the future tasks to be carried out include: buying a camera, get video footage from CCNY cameras and permission to use it as part of the project, continue reading about the subjects that are being addressed, continue updating the design as changes come along the way, and, finally, implement the system.

### 8. Bibliography

Additionally, some ideas were adapted from: