Progress Report of Final Year Project

Project Title: Design and implement a face-tracking engine for video

William O’Grady
08339937

Electronic and Computer Engineering,
College of Engineering and Informatics,
National University of Ireland, Galway

Project Supervisor:
Dr. Peter Corcoran

Submission Date: 15th December 2011
# Table of Contents

List of Figures ........................................................................................................4  
Glossary ..................................................................................................................4  
Abstract ...............................................................................................................5  
Chapter 1 Project Outline ......................................................................................6  
  1.1 Project Title ..................................................................................................6  
  1.2 Project Overview .........................................................................................6  
  1.3 Project Aims ...............................................................................................6  
  1.4 Conclusion ..................................................................................................6  
Chapter 2 Progress to Date .....................................................................................8  
  2.1 Face Detection Research ............................................................................8  
    2.1.1 Introduction ..........................................................................................8  
    2.1.2 LBP based Classifiers .........................................................................8  
    2.1.3 Census Transform ..............................................................................8  
    2.1.4 Viola-Jones Algorithm ......................................................................8  
    2.1.5 Results ...............................................................................................9  
  2.2 Viola Jones Algorithm ..................................................................................9  
    2.2.1 Introduction ........................................................................................9  
    2.2.2 Integral Image ....................................................................................9  
    2.2.3 Cascaded Classifier ..........................................................................10  
  2.3 Face Detection Implementation ....................................................................11  
    2.3.1 Introduction ........................................................................................11  
    2.3.2 Initial Algorithm ...............................................................................11  
    2.3.3 Improvements in Initial Algorithm ......................................................12  
    2.3.4 Current Algorithm Development .......................................................13  
  2.4 FYP Website ...............................................................................................14  
  2.5 Conclusion ..................................................................................................14  
Chapter 3 Tasks Remaining ..................................................................................15  
  3.1 Introduction ...............................................................................................15  
  3.2 Remaining Tasks and Proposals for Completion ...........................................15  
  3.3 Development Plan ......................................................................................16  
  3.4 Conclusion ..................................................................................................16
## List of Figures

<table>
<thead>
<tr>
<th>Figure</th>
<th>Description</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>Figure 2.1</td>
<td>Operation of basic LBP [1]</td>
<td>8</td>
</tr>
<tr>
<td>Figure 2.2</td>
<td>Example of Haar like features</td>
<td>9</td>
</tr>
<tr>
<td>Figure 2.3</td>
<td>Cascaded Classifier</td>
<td>10</td>
</tr>
<tr>
<td>Figure 2.4</td>
<td>Flowchart of basic algorithm</td>
<td>11</td>
</tr>
<tr>
<td>Figure 2.5</td>
<td>Example of face tacking sub-algorithm</td>
<td>12</td>
</tr>
<tr>
<td>Figure 2.6</td>
<td>Improvements of face tacking sub-algorithm</td>
<td>13</td>
</tr>
<tr>
<td>Figure 3.1</td>
<td>Current Development Plan</td>
<td>16</td>
</tr>
</tbody>
</table>

## Glossary

<table>
<thead>
<tr>
<th>Abbreviation</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>FYP</td>
<td>Final Year Project</td>
</tr>
<tr>
<td>FPS</td>
<td>Frames per Second</td>
</tr>
<tr>
<td>LBP</td>
<td>Local Binary Pattern</td>
</tr>
<tr>
<td>ROI</td>
<td>Region of Interest</td>
</tr>
</tbody>
</table>
Abstract

This report documents the project outline of my Final Year Project (FYP), the progress to date and future project plans. It also documents how the future plans will be undertaken in the most efficient way possible to meet all criteria of the project outline.
Chapter 1  Project Outline

1.1  Project Title

Design and implement a face-tracking engine for video.

1.2  Project Overview

The goal of this project is to design and implement an effective and efficient face-tracking algorithm using OpenCV, Matlab or equivalent. The system should be able to detect faces in a video frame, ideally in real time. The system should be able to locate and show the position of the face in the following frames and thus also detect when the face has left the frame. It should also be able to detect and locate multiple faces in a single frame and function accordingly.

This system initially can be initially operated on a PC and does not have to operate in real time. However the end prototype should be able to operate in real-time and dynamically track faces off a live video stream.

1.3  Project Aims

The main aims of the project are as follows:

Research into different techniques for object detection which for this project will be face detection should be researched. Evaluation of this research should be undertaken to determine best method of face detection for faces in a real time video stream. On conclusion of this best method a software architecture and its main subsystems required should be determined. A method of implementing these subsystems should also be evaluated.

A review of the current “state-of-art” face detection algorithm should be undertaken. With the knowledge taken from this review and also the prior research, a working algorithm should be developed. This working algorithm should then be initially tested on a webcam live feed and if needs be the feed can be slowed down to 10 Frames per Second (FPS). Research should then be undertaken to improve on this algorithm through methods such as image pre-processing and using multiple acquired images to provide an effective tracking system. An evaluation should be undertaken on improvements in object tracking through pattern matching and/or pre-filtering.

The algorithm should then be improved through improvements in frame-to-frame tracking over single-frame face detection. The design should then be implemented on an embedded device or FPGA, for the purpose of this project it will probably be an Android based mobile phone. An evaluation methodology should then be developed which enables a comparison in terms of improved tracking and a reduction in terms of false positives and false negatives. A successful
algorithm should also be developed which provides real-time tracking at higher frame rates (25+ fps) or on HD video streams (720p or higher).

Finally these techniques of the improved speed processing, real-time tracking and analysis should then be integrated on a real-time video feed.

1.4 Conclusion
The project brief has been detailed very clearly of the criteria which are to be met. The project initially began mid September and many of these objectives have being met which will be discussed in the next chapter.
Chapter 2  Progress to Date

2.1  Face Detection Research

2.1.1  Introduction
There are many forms of object detection algorithms existing now that the use of cameras has become integrated into our everyday lives. A few of these algorithms which have being researched are Local Binary Pattern (LBP) based classifiers, the Census based transform algorithm and also the more well known, Viola-Jones algorithm.

2.1.2  LBP based Classifiers
LBP is a texture based descriptor which divides a sub image in to a 3x3 matrix in which it thresholds each pixel of this matrix against the center pixel. In Figure 2.1 below we can see how this algorithm takes its threshold values and outputs them in a binary form. Then the histogram of the labels can be used as a texture descriptor [1]. With this texture descriptor as well as others, they can then be compared in a classifier to determine whether or not an object if a face or not.

2.1.3  Census Transform
Census based transforms are very similar to LBP but the main difference is where the LBP compares 8 pixels with the centre pixel (8bit result), Census compares all 9 pixel values with their average (9bit result) [2].

It appears that both approaches in terms of face detection do not seem to be widely implemented. A major issue of concern with both algorithms would be memory location as to get an accurate texture description of one’s face then it would require a huge amount of 3x3 windows in which the threshold is calculated. Not only that but it would make the processing speed of the detector very slow, which for a real time feed is simply not suitable. If we compare these two algorithms to Viola-Jones algorithm, we can clearly see why Viola-Jones algorithm is the most popular.

2.1.4  Viola-Jones Algorithm
Viola-Jones algorithm is by far the most dominant algorithm when it comes to face detection. It uses a combination of integral images, cascaded classifiers and a learning algorithm called Adaboost which increases the performance of the algorithm through improvements on the classifiers. Not only is this algorithm the most popular and suitable for detection on a real time feed but OpenCV libraries
have its own built methods to allow easy development of this algorithm. This makes it the most suitable algorithm for a FYP out of the three.

2.1.5 Results
From the above research it is clear to see that the most suitable algorithm for this project would have to be Viola-Jones Algorithm. However a further comparison of the three algorithms could make for an interesting post graduate project, unfortunately though it is not suitable for this FYP due to time constraints.

Before development can begin, it is essential that a more detailed review of Viola-Jones algorithm and its three main features is undertaken.

2.2 Viola Jones Algorithm

2.2.1 Introduction
As described above this algorithm has three main features; integral images, cascaded classifiers and its learning algorithm. These three features which make it so efficient will now be reviewed in a detailed manner.

2.2.2 Integral Image
Before an integral image can be described, its main process can be described as Haar basis functions [3]. A Haar like feature is essentially getting an image, splitting it up into rectangles, summing up the pixel intensities in these regions and then getting their differences. Figure 2.2 below gives us an example of how these regions work. By using these different features and also through the use of rotation, an accurate threshold pixel intensity of a certain area can be calculated.

![Figure 2.2 Example of Haar like features](image)
However Haar like features is only one process of the integral images implementation in which the algorithm uses. The actual integral image involves taking these images and “integrating” them together i.e. adding more sub images to allow for a whole image to be scanned using these Haar features in a very effective way. This is one of the main features which allows the algorithm to be implemented in real time.

### 2.2.3 Cascaded Classifier

A classifier is to categorize pixels in a digital image into some class [4] or in this case a face. A cascaded classifier is thus a classifier in step like (cascaded) form. In this case the classifier comes in xml format and has already being precompiled in OpenCVs’ package. Cascaded classifiers are used to optimize the detection rate. If a sub image being passed through a classifier and it fails at any of the points it is then rejected and in this case labeled “not a face”. Figure 2.3 below depicts the operation of a cascaded classifier.

![Cascaded Classifier](image)

The initial classifier removes a large amount of the negative sub images with very little processing. The lower layers of classifier eliminate additional negative sub images but this takes more computational requirement [3].

However it’s not just a normal classifier that is being used, it is a cascaded classifier that has being trained using the Adaboost learning algorithm so that it can quickly reject negative sub images.

### 2.3.4 Adaboost Algorithm

Adaboost is a self-adaption algorithm in the sense that it trains itself. It is used to train classifiers to make them more efficient and quicker to reject negative sub images. It works as follows [5]:

- It calls a weak classifier repeatedly in series of rounds $t = 1 ... T$ (total T classifiers)
- After each call a distribution of weights $D_t$ is updated which indicates the importance of the data set in the classification.
- On each round weights of incorrect classified examples increase while weights of correct classified examples decrease.
2.3.5 Conclusion
Each separate feature of Viola-Jones algorithm combines to make the overall algorithm very efficient and suitable for face detection on a real time video feed. This conclusion furthermore strengthens the decision to use this algorithm as opposed to LBP or Census based transform. With this information an effective algorithm can now be designed. The most suitable software to implement this algorithm with would have to be OpenCV libraries as its package already contains numerous cascaded classifiers all trained using an AdaBoost algorithm. The package as explained earlier also contains inbuilt functions to specifically implement integral images and Haar based detection.

2.3 Face Detection Implementation

2.3.1 Introduction
Viola-Jones algorithm has thus being determined to be used as the main algorithm for the program. As outlined before OpenCV libraries come equipped with the necessary functions and features to successfully implement it as well. A decision must now be made as to what language to code this algorithm with. The choices for OpenCV libraries are Python, C, C++ and Android. On the basis that it is best to “keep to what you know”, the language C was decided upon as the main language of implementation for this project while C++ may also be needed if editing of the OpenCV libraries is undertaken. However the final code will hopefully be in the end ported over to Android, as it is one of the aims to implement the project on an embedded device.

2.3.2 Initial Algorithm
Before a complex algorithm can be decided it is essential to fully understand and develop a basic algorithm. Figure 2.3 below is the flowchart for this initial algorithm which was successfully developed. The initial algorithm uses a standard integrated webcam to provide the live feed.
The program firstly initializes variable and loads necessary data such as the frontal face classifier. The program then enters a while loop to continually get new frames, it then calls a method which scans these frames for faces and displays the location of these faces through the use of a square around the persons head.

2.3.3 Improvements in Initial Algorithm

The first basic improvement was simply to halve the size of the image that was scanned while still retaining quality of the image. This improvement alone yielded impressive results.

From there it was clear that to really optimize this algorithm than it would be necessary to implement a face tracking algorithm to improve processing speed through frame-to-frame tracking over single-frame face detection. The initial algorithm was to first detect a person’s face in a frame. If a face was found then it records its location. In the next frame, if a face had been found in the previous frame then the algorithm sets the area where the face was found plus a buffer (to allow for head movement) as a Region of Interest (ROI). Now that a ROI has been set the algorithm only needs to search a small area for a face as opposed to a whole image. Figure 2.5 below illustrates this.

However it is clear that there are issues in this algorithm when it comes to detecting multiple faces, but these issues are taken care in the refining steps of this sub-algorithm.

The solid red square in the above Figure 2.5 shows what would be displayed when the algorithm detects a face. The dotted red square around this red box shows that the program searches for a face in. This sub-algorithm has also being developed and provided a huge increase in processing speed in FPS however due to the major flaw mentioned above, these results aren’t exactly valid.

However it is now essential that we develop the sub algorithm further so it includes the ability to detect other faces outside of this ROI. The algorithm that was developed to solve this problem was simple enough. If the first face was found (it being the largest, found by setting certain flags in Haar detector) in a certain area, then that face is tracked. The program then searches all other parts of the image for more faces. So for example the largest face was found to the left of the image centre, then the program searches everywhere outside the area in which the face was found. The same
happens for a face found on the right side of the image. However if the largest face was found in the middle of the image then the program searches to the left and right of this face for “smaller” faces.

The algorithm now allows for more than one face to be detected, however it still only tracks the main (largest) face. Figure 2.6 below illustrates what happens when the largest face is found in the centre area of the image.

![Figure 2.6 Improvements of face tracking sub-algorithm](image)

The areas on either side of the green lines in Figure 2.6 are the regions in which the algorithm searches for more faces. It can be seen clearly how the smaller faces to the left and right of the main face have been detected.

This algorithm has been successfully coded and has yielded an increase in performance.

2.3.4 Current Algorithm Development
Currently the main focus is now on trying to improve on performance through the use of combining multiple images to provide a better tracking system. An example of this would be the tracker knows the direction in which to expect the face to move in. Of course this depends on the camera either being stationary and a person moving or if the camera is rotating about an axis.

Another feature being developed is not trying to detect all faces in one “sweep” of a frame but using multiple frames to detect all faces. This is based on the assumption that faces won’t disappear in the space of one or two frames.

An important pre-processing algorithm could be if an area of a face was known, then it may be more performance efficient to use a filter to look for skin colors which will tell us that a face is still there.

These ideas are still being developed and are not yet finalized, however if these are successfully implemented then it could improve on performance of the algorithm immensely.
2.4 FYP Website
A website has been created for this project. The main purpose of this website is to allow the public to view progress made on the project throughout its cycle. This website can be found with the following link:

http://williamogradyfyp.wordpress.com

2.5 Conclusion
Below is a checklist of the tasks that have been completed. It clearly shows that progress-wise, the project is ahead of schedule when compared to what else is required from the project.

✓ Research different techniques for face detection.
✓ Evaluate and determine best method.
✓ Determine main algorithm and subsystems based on this evaluation
✓ Review current “state-of-art” face detection algorithm
✓ Develop a working algorithm.
✓ Tested on live webcam feed (not slowed down in this case).
✓ Research into improving algorithm through further pre-processing and/or using multiple images to provide an effective tracking system.
✓ Develop a tracking algorithm for single-frame face detection
Chapter 3  Tasks Remaining

3.1  Introduction
Below are a list of tasks that are yet to be completed. The project is ahead of schedule which means that either more tasks can be added on or more time can be delegated to the more difficult tasks that await. It is vital that the initial objectives outlined in the brief are met first before any extra tasks are tempted, therefore it is probably best that the original development plan doesn’t differ much and if time allows at the end, more features can be added to the project.

3.2  Remaining Tasks and Proposals for Completion
The following list illustrates the remaining tasks due for completion and proposals for tackling them.

- Evaluate research of object tracking through pattern matching and/or pre-filtering
  - Now that research has being undertaken for the use of multiple acquired images to provide a more effective tracking algorithm and also the use of pre-processing to detect skin in a known area of a face, it is essential that these methods be tested to determine most suitable algorithm. It may be best to try each of these subsystems in the main design and record results and finally use a combination of these algorithms to determine the most suitable system.

- The design should be implemented on an embedded device or FPGA.
  - The embedded device that has being chosen is an Android based mobile phone. There are many strong reasons for this choice, such as Android is very development friendly due to its open source background and also there is a huge amount of documentation for it.
  - This task may take more time than planned due to the need of porting code into android based OpenCV (unsure as of yet the complexity involved)
  - As discussed above the project is ahead of schedule so if time allows develop this Android app should be further developed so that a face in real time could be altered in a certain way, such as placing different articles of clothing on it i.e. a hat or something similar. This is just an initial idea but altering of the face in some way will be the end goal.

- An evaluation methodology should be developed which enables a comparison in terms of improved tracking and a reduction in terms of false positives and false negatives.
  - As of now there is no defined method of proposal for this task. Further research and discussion with the project supervisor will need to be undertaken to determine best method of implementation.

- Develop algorithm which provides real-time tracking at 25+ FPS or on HD video streams (720p or higher)
  - Continue research and evaluations of tracking subsystem as described above and test on HD video stream or a camera which operates in 25+ FPS.
- Develop final design and test on a real-time video feed.
3.3 Development Plan
The following plan is very similar to the original plan. The timeline has being updated to the current status of the project and the added Android app feature has being included at the end of this plan.

<table>
<thead>
<tr>
<th>Date</th>
<th>Task Description</th>
<th>Events</th>
</tr>
</thead>
<tbody>
<tr>
<td>01/01/2012</td>
<td>Continue research and evaluate research of object tracking through pattern matching and/or pre-filtering</td>
<td>Oral Presentation</td>
</tr>
<tr>
<td>08/01/2012</td>
<td>Implement decision subsystem and test</td>
<td></td>
</tr>
<tr>
<td>15/01/2012</td>
<td>Implement design on an Android based mobile phone</td>
<td></td>
</tr>
<tr>
<td>22/01/2012</td>
<td>Develop an evaluation methodology of improved tracking</td>
<td>Project Review Meeting</td>
</tr>
<tr>
<td>29/01/2012</td>
<td></td>
<td></td>
</tr>
<tr>
<td>05/02/2012</td>
<td></td>
<td></td>
</tr>
<tr>
<td>12/02/2012</td>
<td>Develop an algorithm which provides real-time tracking at 25+ FPS</td>
<td></td>
</tr>
<tr>
<td>19/02/2012</td>
<td></td>
<td></td>
</tr>
<tr>
<td>26/02/2012</td>
<td>Integrate this subsystem into final design and test</td>
<td></td>
</tr>
<tr>
<td>04/03/2012</td>
<td>Update Android app and add extra feature</td>
<td>Bench Demonstrations</td>
</tr>
<tr>
<td>11/03/2012</td>
<td></td>
<td></td>
</tr>
<tr>
<td>18/03/2012</td>
<td></td>
<td>Final Version of Web Page</td>
</tr>
<tr>
<td>25/03/2012</td>
<td></td>
<td>Project Log Book Submission</td>
</tr>
<tr>
<td>01/04/2012</td>
<td>End of Project</td>
<td>Final Project Thesis</td>
</tr>
</tbody>
</table>

Figure 3.1 Current Development Plan

3.4 Conclusion
The project is ahead of schedule, all required tasks and events to date have being met on time. The project should be finished on time and have a working demonstration available for the week of 11th March 2012.
References

doi: 10.1109/ISKE.2008.4731094

doi: 10.1109/AFGR.2004.1301514

doi: 10.1109/CVPR.2001.990517

[4] TANGJAITRONG, Supichai “Image Classification”, Enviromental Remote Sensing, Chulalongkorn University: Faculty of Science [online], available:
http://www.sc.chula.ac.th/courseware/2309507/Lecture/remote18.htm [accessed 18th Nov 2011]