

## Real-Time Face Detection and Tracking for High Resolution Smart Camera System

Y. M. Mustafah <sup>a,b</sup>, T. Shan <sup>a</sup>, A. W. Azman <sup>a,b</sup>, A. Bigdeli <sup>a</sup>, B. C. Lovell <sup>a,b</sup>  
<sup>a</sup> NICTA Ltd, 300 Adelaide Street, Brisbane, QLD 4000, Australia  
<sup>b</sup> ITEE, University of Queensland, Brisbane, QLD 4072, Australia

### Abstract

Smart Cameras are becoming more popular in Intelligent Surveillance Systems area. Recognizing faces in a crowd in real-time is a key features which would significantly enhance Intelligent Surveillance Systems. Using a high resolution smart camera as a tool to extract faces that are suitable for face recognition would greatly reduce the computational load on the main processing unit. This processing unit would not be overloaded by the demands of the high data rates required for high resolution video and could be designed solely for face recognition. In this paper we report on a multiple-stage face detection and tracking system that is designed for implementation on the NICTA high resolution (5 MP) smart camera.

### 1. Introduction

Smart camera has recently emerged to become a popular alternative for the traditional digital camera with the advances in both machine vision and semiconductor technology. With the smart camera concept, a camera will have the ability to extract specific information from the images that it has captured. So far there does not seem to be an established definition of what a smart camera is. In this paper, we define a smart camera as a vision system which can extract specific information from images for other devices such as a PC or a surveillance system without the need for an external processing unit. Figure 1 shows a basic structure of a smart camera. Just like a typical digital camera, a smart camera captures an image using an image sensor, stores the captured image in the memory and transfers it to another device or user using a communication interface. However, unlike the simple processor in a typical digital camera, the processor in a smart camera will not only control the camera functionalities but it is also able to analyse the captured images so that extra information can be obtained from them.

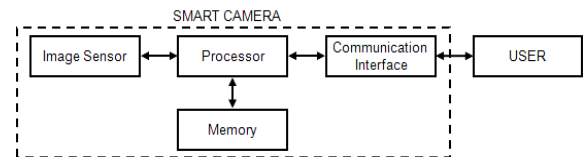


Figure 1: Basic Smart Camera Architecture.

There are many smart camera products available in the market today. However, it is still a very active area of research because of the wide range of capabilities of smart camera that could be improved. One of the most well-known works on the smart camera was by Wolf et al. [1]. They introduced a smart camera system that could recognize various gestures made by a person. Another popular work on smart camera was by Bramberger et al. [2] where they built a smart camera prototype called SmartCam. Their prototype camera is a fully embedded smart camera system targeted for various surveillance applications such as traffic control.

In [3], we described a smart camera design that can be used as an aid for face recognition in crowd surveillance. The camera is designed to utilize a high resolution CMOS image capture device and an FPGA based processor for ROI extraction. Crowd surveillance usually surveys a very wide area with several objects of interest in its view, thus requiring a high resolution camera.

### 2. Face Detection on High Resolution Image

In 2001, Viola and Jones [4] proposed an image-based face detection system which can achieve remarkably good performance in terms of detection accuracy as well as speed. The main idea of their method is to combine weak classifiers based on simple binary features which can be computed extremely quickly. Simple rectangular Haar-like features are extracted; face and non-face classification is done using a cascade of successively more complex classifiers which are trained by the AdaBoost learning algorithm.

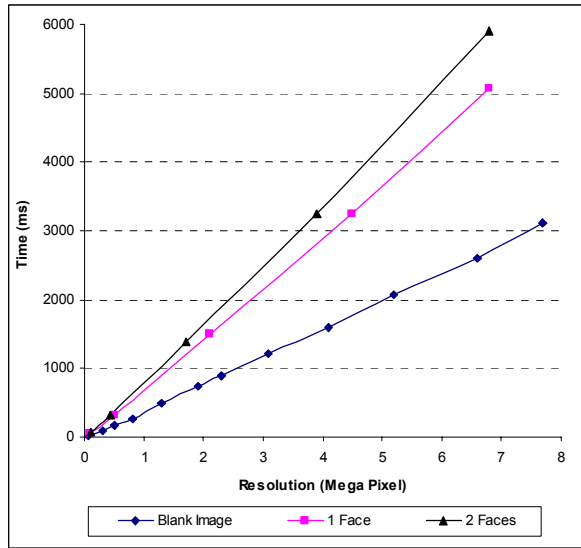


Figure 2: Face Detection Time vs Image Resolution

High resolution images provide much more detailed information regarding objects in view. However, processing a high resolution images would require higher processing speed and more memory. For example, images captured by our 5MP smart camera system would require 15 times more memory compared to the standard VGA resolution images with the same bit depth. While a Viola-Jones system can achieve almost real-time performance with low resolution images, it would require much longer processing time as the resolution of the image gets higher. The graph in Figure 2 shows the time taken by the OpenCV [5] based Viola-Jones face detection module to detect a face in several test images with resolutions ranging from 320 x 240 to 3200 x 2400. The time taken for the module to detect faces increases linearly with the resolution of the images due to the number of the detection sub-windows that need to be generated by the module increasing proportionally with image resolution. As the complexity of the images increases (i.e. more faces are present in the images), the time taken will also increase as more detection sub-windows manage to pass through the cascade classifier stages.

High resolution images also have the tendency to yield more false positive detections due the extra details in the image. Figure 3 shows the face detection results on VGA resolution image (a) and 4MP resolution image (b). For both tests, the same standard OpenCV module was used. The tests assume that faces in the image could be in any size from the minimum size of 30x30 pixels. Even though the face can be correctly detected in the image, many false positive detections could cause a major problem if the results are to be used for a face recognition system.

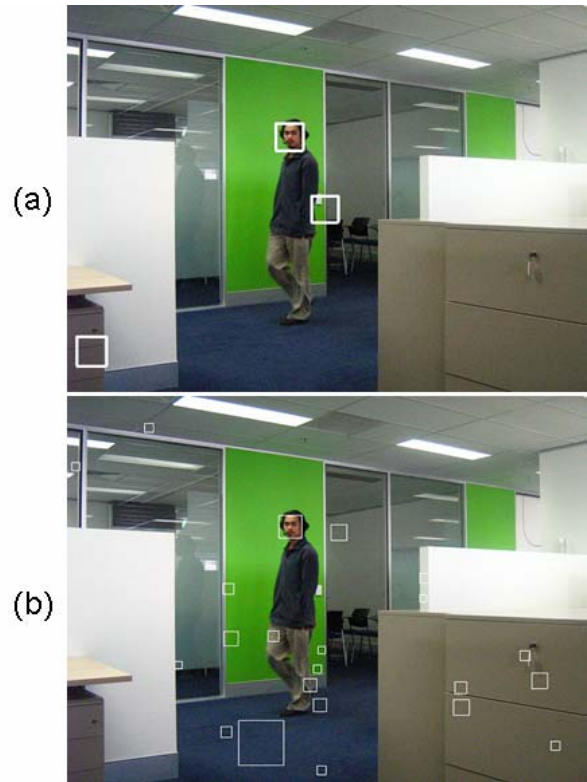


Figure 3: Face Detection on VGA Image (a) and 4MP Image (b)

### 3. Real-Time Face Detection and Tracking System

The Viola-Jones face detector, depending on its training data, can produce a very accurate face detector. In surveillance activity, cameras are usually fixed to one location. Therefore, background subtraction could be used to reduce the Region-of-Interest (ROI) in the captured image. Focusing on the face detection problem in colour image, the ROI could be reduced even further by using a skin colour detector. With the ROI greatly reduced, it would be possible to implement a Viola-Jones face detector for a high resolution system in real-time.

Taking the above mentioned points as our motivation, we propose a multiple-stage face detection and tracking system that is suitable to be implemented on a high resolution smart camera. Figure 4 shows the design of the proposed system. It consists of a background subtraction stage for object tracking, a skin colour detection stage for human skin tracking, and two-step Viola-Jones face detection stage for face detection and tracking.

Most CMOS image sensor devices capture images by sending only one or several pixels at a time as their

output. On the embedded vision platform, if the pixels can be processed as they come out from the image sensor in ‘pixel-based’, processing speed could be greatly increased and memory resource usage would be greatly reduced. Pixel-based processing means that the pixel can be processed individually, independently from its neighboring pixels. Pixel-based processing would allow more flexible parallelism in the hardware design. While, it is impossible to use the pixel-based processing approach for Viola-Jones face detection, we have managed to design the earlier stages of the proposed system so they can be processed in pixel-based.

In the next sections we will present the proposed face detection and tracking system and show how the stages are designed for the real-time implementation.

#### 4. Background Subtraction Stage

Background subtraction is a widely used approach for detecting moving objects in a sequence of frame images from static cameras. In this approach, moving objects are detected from the difference between the current frame image and a reference image (background image). The background image is an image that represents the view of the camera with no moving objects. As a static camera view is usually affected by various changes such as the varying luminance and noise, the background image must be frequently updated so that the difference between the image and the current detected frame image is minimal.

Many different background subtraction methods have been proposed over the years such as Running Gaussian Average, Temporal Median Filter, Mixture of Gaussians, Kernel Density Estimation, Sequential Kernel Density Estimation, Co-occurrence of Image Variations and Eigen-backgrounds [6]. At this early stage of implementation, we have chosen to use basic or direct background subtraction as this method requires the minimum memory usage and it is possible to use pixel-based implementation.

In our implementation, direct subtraction is performed on every incoming pixel data against the corresponding background pixel. If the subtraction result is less than the preset threshold value, the pixel is considered as background and will be used to update the background pixel data. Otherwise, the pixel will be considered as an object pixel. Higher threshold values would produce a cleaner background subtraction result, but would compromise the overall accuracy of background subtraction result. Figure 5 shows the results from the background subtraction implemented in software, with two different threshold values; (a) very low value and (b) very high value. It is noticeable that the noise error can be reduced by setting the threshold value higher in (b) and that as a result, detection accuracy is poor.

#### 4.1. Noise Filter

In order to obtain a cleaner background subtraction result out of the chosen background subtraction method, a simple Noise Filter stage is introduced. In this stage, we eliminate all the pixels that is classified as a noise pixel from the background subtraction result. We defined the pixel,  $p$  as a noise, if all the pixels surrounding  $p$ , within a certain threshold range,  $T$ , are not an object pixels. Higher value of  $T$  would produce a much cleaner result but would cost a higher memory usage in hardware implementation. The noise filter stage will allow the threshold value for background subtraction to be set to a lower value, thus maintaining its accuracy. Figure 6 shows the result of background subtraction before the noise filter stage (a) and after the noise filter stage (b).

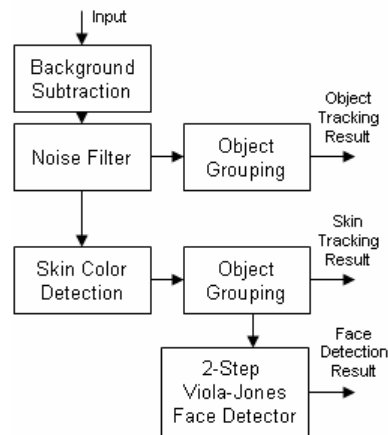


Figure 4: Proposed face detection system



Figure 5: Background Subtraction using Low Threshold Value (a) and High Threshold Value (b)

## 5. Skin Colour Detection

Colour has proven to be a useful tool for robust object detection. For human skin, colour characteristics can be used to very easily determine if a image pixel belongs to a human skin or otherwise. Numerous techniques for skin colour detection and recognition have been reported in the literature. In [7], skin colour detection methods are categorized into three classes; 1) methods that use explicitly defined skin cluster boundaries, 2) non-parametric methods and 3) parametric methods.

Due to its simplicity and pixel-based implementation, the explicitly defined skin cluster boundaries method is chosen. We chose RGB as the colour space for the skin detection to avoid further pixel data processing as the pixels are generated in RGB from the CMOS image sensor. Equation (1) shows the skin colour boundaries defined by Peer et. al. [7].

$$\begin{aligned} & \mathbf{(R,G,B) \text{ is classified as skin if:}} \\ & \mathbf{R > 95 \text{ and } G > 40 \text{ and } B > 20 \text{ and}} \\ & \mathbf{\max\{R,G,B\} - \min\{R,G,B\} > 15 \text{ and}} \quad (1) \\ & \mathbf{|R-G| > 15 \text{ and } R > G \text{ and } R > B} \end{aligned}$$

Having a skin detection combined with background subtraction improves the skin detection result significantly. Figure 7 shows the results of the skin detection module alone (a) and skin detection combined with the background subtraction module (b). Many false positive detections in (a) are removed in (b) by the background subtraction module.

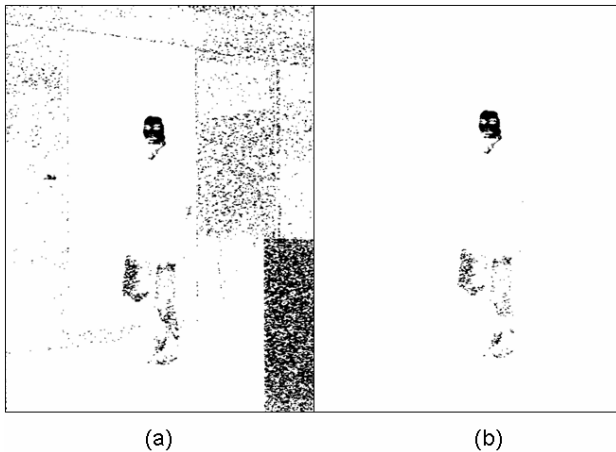


Figure 7: Skin Detection alone (a) and Skin Subtraction with Background Subtraction (b)

## 6. Object Grouping

For an automated system, objects must be automatically identified and grouped as an ROI after the background subtraction process or the skin colour detection process. In our proposed face detection system, the ROI windows need to be generated from the skin colour detection result automatically to be sent as an input to the Viola-Jones face detector.

We implement a method for grouping objects into ROIs that can be done in pixel-based. In this method, we simply group the pixels together if they are connected within a certain range. Similar to the noise filter in 4.1, we defined a pixel,  $p$  as part of an object,  $O_I$ , if one or more pixels surrounding  $p$ , within a certain threshold range,  $T$ , are part of  $O_I$ . Otherwise  $p$ , will be initialized as the first pixel of a new object,  $O_2$ . Again here, the value of  $T$  plays an important role in producing good results. However, for this particular stage,  $T$  must be selected depending on the minimum scale of the objects of interest. Figure 8 shows the ROIs generated from the background subtraction result in Figure 6-b (a) and from the skin detection result in Figure 7-b (b). A suitable threshold,  $T$ , value was used in the test.

## 7. Two-step Viola-Jones Face detection

With the implementation of the previous stages to greatly reduce the ROI, it is possible to achieve real-time performance from the Viola-Jones face detector on a high resolution image.

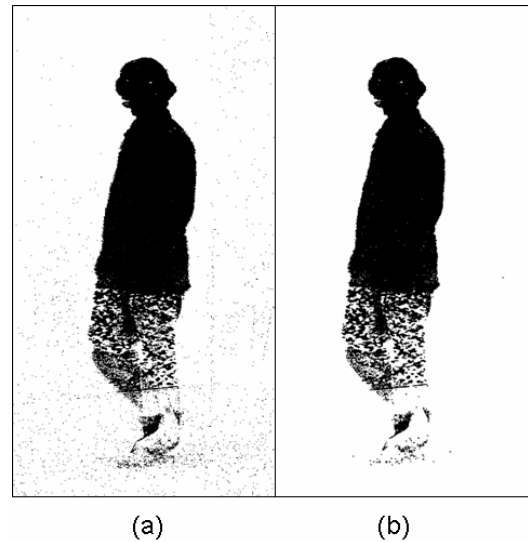


Figure 6: Background Subtraction without Noise Filter (a) and with Noise Filter (b)

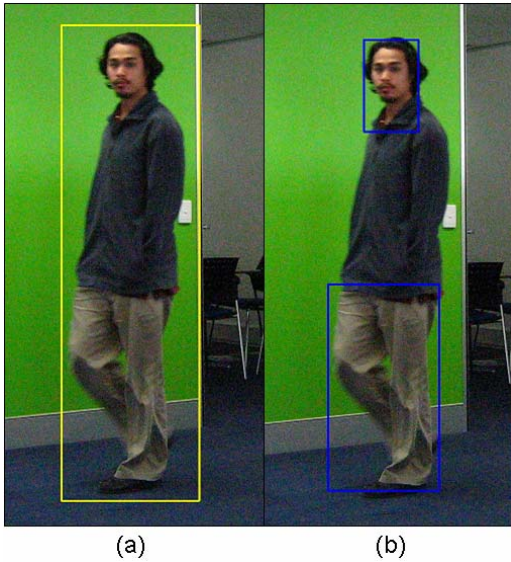


Figure 8: ROI generated from Background Subtraction Result (a) and from Skin Detection Result (b)

In surveillance activity, it is very important to track all the faces that are available in the scene. However, most of these faces usually are not suitable for face recognition. Therefore in order to reduce the load on the face recognition system, having a second detector to filter out all the unsuitable faces is a necessity. Therefore we introduce the idea of two-step face detector that is required to enable our system to track all the faces in the image as well as to detect faces that are suitable for recognition to be sent to the face recognition system. Figure 9 shows the result of the two-step face detection module. First, skin detection ROI (a) is tested for faces. Then, if a face is found, the ROI of the face (b) will be further detected for faces suitable for face recognition (c).

### 7.1. Finding All the Faces in the Image

The purpose of the first face detector is to detect all the face in the image regardless of their orientation or rotation. Very robust face detection is very important so that the result can be used for a tracking and identity labeling. To do this, the face detection module could be trained using the any possible faces image such as the faces in Figure 10. Notice the training data consist of variety of faces pose and orientation.

### 7.2. Finding Faces for Face Recognition

The second stage of the face detector must be able to filter all the faces leaving only the faces that are suitable for face recognition. Most face recognition systems currently available are very good at recognizing perfectly

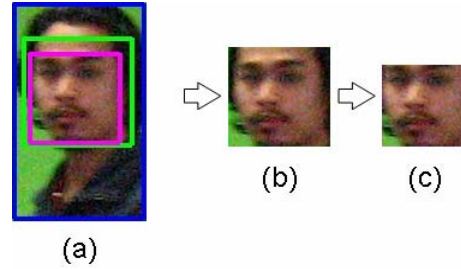


Figure 9: (a) Skin Region, (b) First Step Face Detection, (c) Second Step Face Detection



Figure 10: Training Image for the First Step of the Face Detector



Figure 11: Training Image for the Second Step of the Face Detector

aligned face. Hence, at this stage, we trained our second stage of the face detector using mostly frontal face images. Figure 11 shows some of the faces that were identified for suitability to be used for the second step face detector training. Most of the faces used for the training were taken from the MIT CBCL Face Database [8].

## 8. Results and Discussions

The system proposed was implemented using C and the OpenCV library. We tested the implementation on a sequence of high resolution (4MP) frame images and Figure 12 shows the results on 3 sequential frames. With this method we managed to track moving objects, human skin, and human faces and to detect suitable faces for recognition based on the Viola-Jones face detection module at a very high speed. The time taken to perform the two-step face detection on a system with 3.00GHz Pentium 4 CPU for the result in Figure 9, is approximately 20.2ms. A time of 17.9ms is taken in first face detector and only 2.3ms is taken in the second

detector. Compared to the time taken to detect a face in Figure 3 on the same system, which is  $5449ms$ , our system is significantly faster and more accurate, producing much fewer false positive results. It is believed, the hardware implementation would be much faster as most of the stages were designed for pixel-based processing implementation.

Implementing the proposed face detection system in hardware is not really a straight forward task. While the earlier stages seem quite easy to implement on hardware, there are many issues to be considered. For instance, for the noise filter and object grouping stages, even though they can be processed pixel-based, some pixels will need to be buffered depending on the threshold values selected. Hence threshold values for these stages need to be carefully determined so that adequate memory resources are utilized on the hardware.

Due to the recursive nature of the Viola-Jones face detector module, a good design of the face detector is necessary for hardware implementation. Reducing the classifier stages and the features of the module would ease the hardware implementation difficulties, but it would lower the accuracy of the detector. Hence, suitable values have to be determined.

## 9. Conclusions

Smart Cameras are being introduced in emerging surveillance systems. In [3], we proposed a high resolution smart camera system to aid the crowd surveillance activity. In this paper, we now propose face detection and tracking system that is suitable to implement on the smart camera system. The system consists of a background subtraction stage, a skin colour detection stage, and two-step Viola-Jones face detection stage. The face detection was successfully implemented in software using C and the OpenCV library. The implementation shows that the proposed system is able to speed up and increase the accuracy of face detection on high resolution images. Our future work would involve implementing this robust face detection algorithm in the NICTA smart camera system.

## 10. Acknowledgement

This project is supported by a grant from the Australian Government Department of the Prime Minister and Cabinet and by the Australian Research Council through the Research Network for Securing Australia. NICTA is funded by the Australian Government's Backing Australia's Ability initiative, in part through the Australian Research Council.



Figure 12: Software Implementation Result on Sequence of Frames

## 11. References

- [1] W. Wolf, B. Ozer, and T. Lv, "Smart cameras as embedded systems," *Computer*, vol. 35, pp. 48-53, 2002.
- [2] M. Bramberger, A. Doblender, A. Maier, B. Rinner, and H. Schwabach, "Distributed embedded smart cameras for surveillance applications," *Computer*, vol. 39, pp. 68-75, 2006.
- [3] Y. M. Mustafah, A. W. Azman, A. Bigdeli, B. C. Lovell, "An Automated Face Recognition System For Intelligence Surveillance: Smart Camera Recognizing Faces In The Crowd" to be published in International Conference on Distributed Smart Camera Conference, Vienna, Austria, September 2007.
- [4] P. Viola and M. Jones, "Rapid object detection using a boosted cascade of simple features", in *IEEE Conference on Computer Vision and Pattern Recognition*, pp. 511-518, Kauai, USA, December 2001.
- [5] *Open Computer Vision Library*. Retrieved on August 7, 2007 from <http://sourceforge.net/projects/opencvlibrary/>.
- [6] M. Piccardi, "Background subtraction techniques: a review" in *IEEE International Conference on Systems, Man and Cybernetics*, pp. 3099- 3104, vol.4, The Hague, Netherlands, October 2004.
- [7] Vezhnevets V., Sazonov V., Andreeva A., "A Survey on Pixel-Based Skin Color Detection Techniques". *Proc. Graphicon-2003*, pp. 85-92, Moscow, Russia, September 2003.
- [8] CBCL Face Database #1, MIT Center For Biological and Computation Learning, <http://www.ai.mit.edu/projects/cbcl>, last accessed on August 7, 2007.