An ARM Based Hardware Architecture for Image Processing

Vikash Kumar Singh

Abstract- Image processing and Object detection applications are often associated with real-time performance constraints that stem from the embedded environment that they are often deployed in. The aim of the project is to design and develop a hardware implementation using advanced features of ARM 9 architecture.

The proposed embedded system uses ARM 32 bit RISC CPU. It has features of image/object detection and video processing by using various features and classification algorithms for object detection. Image processing is a form of signal processing for which the input is an image, such as a photograph or video frame; the output of image processing may be either an image or a set of characteristics or parameters related to the image. The used Algorithms and Hardware architecture overcomes the performance in terms of sensors and hardware cost is also low. So, our designed embedded system that detects partially visible human being faces just as they enter the camera view, with low false alarm rate and high speed.

This system capture image by means of camera connected to ARM9 microcontroller through USB and the image is detected and processed using image processing technique like Haar object detection and Viola-Jones Detection Algorithm. The captured image undergoes spatio temporal reference samples in terms of both back ground and fore ground estimation, evaluation and spatial Gaussian kernel to provide high quality image of object detection that detected image is continuously displayed on display unit and data is stored in pen drive connected to it. The detected image is compared with the stored face.xml in ARM, if a face is detected in image, the count is of detected face is increased and the image is stored in the USB storage device for further analysis. On connected the USB with PC with internet, an automated email system will take the images, attach it with an email and the email will be sent automatically to a given email id.

Index Terms-: Image Processing using ARM9, face Detection, counting objects with low level features, CCTV Replacement, Haar Object detection on ARM9, Viola-Jones Algorithm on ARM9, Automated Email Attacher

1 *INTRODUCTION*

Since the appearance of face Recognition technology, in the 80's of 20th century, and it has rapidly developed, especially in the 90's. Nowadays, it has achieved some results periodically, and has been used in safeguard area and so on. Automatic face recognition is a widely used biological recognition technology. In comparison with other detection methods, face detection and image storage has direct, friendly and convenient features.

Now the market share of face recognition is just less than the fingerprint recognition and the proportion is increasing, which have broken the situation that fingerprint recognition monopolized the market in the international biological recognition market. Therefore, it is very important for the research of face recognition technology.

This paper, which is based on ARM9 and Linux operating system, is a portable device and meets the requirements of up to date for face detection; it also can be used in many other areas.

- *Vikash Kumar Singh is currently pursuing master's degree program in Embedded System and VLSI – D in Lords Institute of Engineering and Technolgy, JNTU, Hyderabad, India, PH-+91-9000764778. E-mail: VKSinghMailBox@GMail.com*

The system is based on the ARM9 processor for face detection systems. The Embedded System uses the common USB camera for image acquisition, Linux-based operating system software, and ARM9 S3C2440. Based on this hardware platform, Embedded Linux operating system and drivers are developed firstly, and then face detection system is achieved on the operating system.

A low-cost, small face detection f system which has important practical significance is developed. The system uses ARM9 as the system control chip, the system achieved low cost, portability, miniaturization requirements. Different from the run on PC machine face recognition algorithm, the system identification algorithm must take into account the computing capacity and ARM speed, so that detection accuracy and recognition time is an acceptable range.

The proposed embedded system uses ARM 32 bit RISC CPU. It has features of image/object detection and video processing by using various features and classification algorithms for object detection. Image processing is a form of signal processing for which the input is an image, such as a photograph or video frame; the output of image processing may be either an image or a set of characteristics or parameters related to the image. The used Algorithms and Hardware architecture overcomes the performance in terms of sensors and hardware cost is also low. So, our designed embedded system that detects partially visible human being faces just as they

enter the camera view, with low false alarm rate and high speed.

This system capture image by means of camera connected to ARM9 microcontroller through USB and the image is detected and processed using image processing technique like Haar object detection and Viola-Jones Detection Algorithm. The captured image undergoes spatio temporal reference samples in terms of both back ground and fore ground estimation, evaluation and spatial Gaussian kernel to provide high quality image of object detection that detected image is continuously displayed on display unit and data is stored in pen drive connected to it. The detected image is compared with the stored face.xml in ARM, if a face is detected in image, the count is of detected face is increased and the image is stored in the USB storage device for further analysis.

2 ARM 9 AND RTOS

ARM9 is an ARM architecture 32-bit RISC CPU family. With this design generation, ARM moved from a von Neumann architecture (Princeton architecture) to a Harvard architecture with separate instruction and data busses (and caches), significantly increasing its potential speed. Most silicon chips integrating these cores will package them as modified Harvard architecture chips, combining the two address busses on the other side of separated CPU caches and tightly coupled memories.

There are two subfamilies, implementing different ARM architecture versions.

1. ARM9TDMI based cores
2. ARM9E based cores

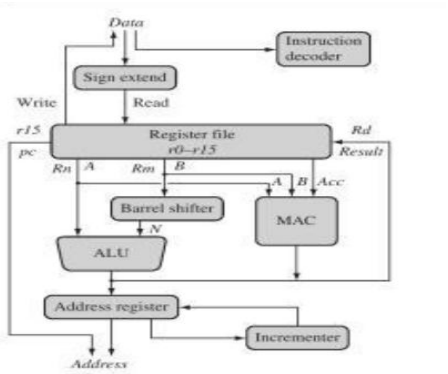

Figure 1- ARM core dataflow model

2.1 Differences from ARM7 Core

Key improvements over ARM7 cores, enabled by spending more transistors, include:

- Decreased heat production and lower overheating risk.
- Clock frequency improvements. Shifting from a three stage pipeline to a five stage one lets the clock speed be approximately doubled, on the same silicon fabrication process.
- Cycle count improvements. Many unmodified ARM7 binaries were measured as taking about 30% fewer cycles to execute on ARM9 cores. Key improvements include
- Faster loads and stores; many instructions now cost just one cycle. This is helped by both the modified Harvard architecture (reducing bus and cache contention) and the new pipeline stages.
- Exposing pipeline interlocks, enabling compiler optimizations to reduce blockage between stages.
- Additionally, some ARM9 cores incorporate "Enhanced DSP" instructions, such as a multiply-accumulate, to support more efficient implementations of digital signal processing algorithms.
- Switching to Harvard architecture entailed a non-unified cache, so that instruction fetches do not evict data (and vice versa). ARM9 cores have separate data and address bus signals, which chip designers use in various ways. In most cases they connect at least part of the address space in von Neumann style, used for both instructions and data, usually to an AHB neither interconnect connecting to a DRAM interface and an External Bus Interface usable with NOR flash memory. Such hybrids are no longer pure Harvard architecture processors.

Analog-to-digital conversion consists two distinct operations: sampling and quantiza-tion. Sampling converts a continuous time, continuous amplitude analog signal into a discrete time, continuous amplitude signal, while quantization converts the continuous amplitude of this sampled signal into a set of discrete levels. Figure 2.1 [1] shows the principle of A/D conversion.

2.2 RTOS Linux

A real-time operating system (RTOS) is an operating system (OS) intended to serve real-time application requests. It must be able to process data as it comes in, typically without buffering delays. Processing time requirements (including any OS delay) are measured in tenths of seconds or shorter.

Linux refers to the family of Unix-like computer operating systems using the Linux kernel.

3. IMAGE PROCESSING IN ARM9 AND LINUX

3.1 Image Processing

An image is defined in the "real world" is considered to be a function of two real variables, for example, a(x,y) with a as the amplitude (e.g. brightness) of the image at the real coordinate position (x,y).

Modern digital technology has made it possible to manipulate multi-dimensional signals with systems that range from simple digital circuits to advanced parallel computers. The goal of this manipulation can be divided into three categories:

- Image processing (image in -> image out)
- Image Analysis (image in -> measurements out)
- Image Understanding (image in -> high-level description out)

3.2 Object Detection

The process of image object detection deals with determining whether an object of interest is present in an image/ video frame or not. An image object detection system receives an input image/video frame, which will subsequently search to find possible objects of interest. This search is done by extracting smaller regions from the frame, called search windows, of m x n pixels (m can be equal to n), which go through some form of preprocessing (histogram equalization, feature extraction), and are then processed by a classification algorithm to determine if they contain an object of interest or not. However, the object of interest may have a larger size than that of the search window, and given that the classification algorithm is trained for a specific search window size, the object detection system must have a mechanism to handle larger objects.

To account for this, an object detection system can increase the size of the search window and rescan the image, which implies that different classifiers are used for each window size. Alternatively, the size of the input image can be decreased (downscaling); consequently, the size of the object of interest will be reduced so it can be "enclosed" within the search window. All subsequent downscaled versions of the input image, are therefore reexamined using the same search window size. The downscaling process is often preferred as it is computationally less expensive [42]. Downscaling is done in steps to account for various object sizes, down to the size of the search window. Hence, many downscaled images are produced from a single input image/video frame, each in turn producing a number of search windows, which increases the amount of data that must be processed by the classification algorithm. Search windows are extracted every few pixels, and the number of pixels that are skipped is called the window overlap.

3.3 Viola Jones Detection Algorithm

The object detection procedure classifies images based on the value of simple features. There are many motivations for using features rather than the pixels directly. The most common reason is that features can act to encode ad-hoc domain knowledge that is difficult to learn using a finite quantity of training data. For this system there is also a second critical motivation for features: the feature-based system operates much faster than a pixel-based system. The simple features used are reminiscent of Haar basis functions which have been used by Papageorgiou et al. More specifically, we use three kinds of features. The value of a two-rectangle feature is the difference between the sum of the pixels within two rectangular regions. The regions have the same size and shape and are horizontally or vertically adjacent . A three-rectangle feature computes the sum within two outside rectangles subtracted from the sum in a center rectangle. Finally a four-rectangle feature computes the difference between diagonal pairs of rectangles. Given that the base resolution of the detector is 24x24, the exhaustive set of rectangle features is quite large, 45,396.

Viola Jones is an object detection technique which minimizes computation time while achieving high detection accuracy. The approach was used to construct a face detection system which is approximately 15 faster than any previous approach. Preliminary experiments, which will be described elsewhere, show that highly efficient detectors for other objects, such as pedestrians, can also be constructed in this way. It is new algorithms, representations, and insights which are quite generic and may well have broader application in computer vision and image processing.

In order to achieve true scale invariance, almost all object detection systems must operate on multiple image scales. The integral image, by eliminating the need to compute a multiscale image pyramid, reduces the initial image processing required for object detection significantly. In the domain of face detection the advantage is quite dramatic. Using the integral image, face detection is completed before an image pyramid can be computed. While the integral image should also have immediate use for other systems which have used Harr-like features, it can foreseeably have impact on any task where Harr-like features may be of value. Initial experiments have shown that a similar feature set is also effective for the task of parameter estimation, where the expression of a face, the position of a head, or the pose of an object is determined.

It is a technique for constructing a cascade of classifiers which radically reduce computation time while improving detection accuracy. Early stages of the cascade are designed to reject a majority of the image in order to focus subsequent processing on promising regions. The key point is that the cascade presented is quite simple and homogeneous in structure. Previous approaches for attentive filtering, such as Itti et. al., propose a more complex and heterogeneous mechanism for filtering [8]. Similarly Amit and Geman propose a hierarchical structure for detection in which the stages are quite different in structure and processing. A homogenous system, besides being easy to implement and understand, has the advantage that simple tradeoffs can be made between processing time and detection performance

3.4 Haar Object Detection

Haar-like features are digital image features used in object recognition. They owe their name to their intuitive similarity with Haar wavelets and were used in the first real-time face detector.

Historically, working with only image intensities (i.e., the RGB pixel values at each and every pixel of image) made the task of feature calculation computationally expensive. A publication by Papageorgiou et al. discussed working with an alternate feature set based on Haar wavelets instead of the usual image intensities. Viola and Jones adapted the idea of using Haar wavelets and developed the so-called Haar-like features. A Haar-like feature considers adjacent rectangular regions at a

specific location in a detection window, sums up the pixel intensities in each region and calculates the difference between these sums. This difference is then used to categorize subsections of an image. For example, let us say we have an image database with human faces. It is a common observation that among all faces the region of the eyes is darker than the region of the cheeks. Therefore a common haar feature for face detection is a set of two adjacent rectangles that lie above the eye and the cheek region. The position of these rectangles is defined relative to a detection window that acts like a bounding box to the target object (the face in this case).

In the detection phase of the Viola–Jones object detection framework, a window of the target size is moved over the input image, and for each subsection of the image the Haar-like feature is calculated. This difference is then compared to a learned threshold that separates non-objects from objects. Because such a Haar-like feature is only a weak learner or classifier (its detection quality is slightly better than random guessing) a large number of Haar-like features are necessary to describe an object with sufficient accuracy. In the Viola–Jones object detection framework, the Haar-like features are therefore organized in something called a classifier cascade to form a strong learner or classifier.

The key advantage of a Haar-like feature over most other features is its calculation speed. Due to the use of integral images, a Haar-like feature of any size can be calculated in constant time (approximately 60 microprocessor instructions for a 2-rectangle feature).

Rectangular Haar-like features

A simple rectangular Haar-like feature can be defined as the difference of the sum of pixels of areas inside the rectangle, which can be at any position and scale within the original image. This modified feature set is called 2-rectangle feature. Viola and Jones also defined 3-rectangle features and 4-rectangle features. The values indicate certain characteristics of a particular area of the image. Each feature type can indicate the existence (or absence) of certain characteristics in the image, such as edges or changes in texture. For example, a 2-rectangle feature can indicate where the border lies between a dark region and a light region.

Fast computation of Haar-like features

One of the contributions of Viola and Jones was to use summed area tables,[3] which they called integral images. Integral images can be defined as two-dimensional lookup tables in the form of a matrix with the same size of the original image. Each element of the integral image contains the sum of all pixels located on the up-left region of the original image (in relation to the element's position). This allows to compute sum of rectangular areas in the image, at any position or scale, using only four lookups:

$$\text{sum} = I(C) + I(A) - I(B) - I(D).$$

where points A, B, C, D belong to the integral image I, as shown in the figure.
Each Haar-like feature may need more than four lookups, depending on how it was defined. Viola and Jones's 2-rectangle features need six lookups, 3-rectangle features need eight lookups, and 4-rectangle features need nine lookups.

Tilted Haar-like features

Lienhart and Maydt introduced the concept of a tilted (45°) Haar-like feature. This was used to increase the dimensionality of the set of features in an attempt to improve the detection of objects in images. This was successful, as some of these features are able to describe the object in a better way. For example, a 2-rectangle tilted Haar-like feature can indicate the existence of an edge at 45°.

Messom and Barczak extended the idea to a generic rotated Haar-like feature. Although the idea sounds mathematically sound, practical problems prevented the use of Haar-like features at any angle. In order to be fast, detection algorithms use low resolution images, causing rounding errors. For this reason, rotated Haar-like features are not commonly used.

Haar-like features are digital image features used in object recognition. A Haar-like feature considers adjacent rectangular regions at a specific location in a detection window, sums up the pixel intensities in each region and calculates the difference between these sums. This difference is then used to categorize subsections of an image. For example, let us say we have an image database with human faces. It is a common observation that among all faces the region of the eyes is darker than the region of the cheeks. Therefore a common haar feature for face detection is a set of two adjacent rectangles that lie above the eye and the cheek region. The position of these rectangles is defined relative to a detection window that acts like a bounding box to the target object (the face in this case. This difference is then compared to a learned threshold that separates non-objects from objects. Because such a Haar-like feature is only a weak learner or classifier (its detection quality is slightly better than random guessing) a large number of Haar-like features are necessary to describe an object with sufficient accuracy. In the Viola–Jones object detection framework, the Haar-like features are therefore organized in something called a classifier cascade to form a strong learner or classifier. The key advantage of a Haar-like feature over most other features is its calculation speed. Due to the use of integral images, a Haar-like feature of any size can be calculated in constant time (approximately 60 microprocessor instructions for a 2-rectangle feature).

OpenCV's face detector uses a method that Paul Viola and Michael Jones published in 2001. Usually called simply the Viola-Jones method, or even just Viola-Jones, this approach to detecting objects in images combines four key concepts:

- Simple rectangular features, called Haar features
- An Integral Image for rapid feature detection
- The AdaBoost machine-learning method

- A cascaded classifier to combine many features efficiently.

The features that Viola and Jones used are based on Haar wavelets. Haar wavelets are single wavelength square waves (one high interval and one low interval). In two dimensions, a square wave is a pair of adjacent rectangles - one light and one dark.

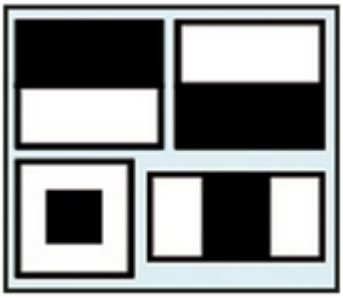

Fig2: Example of Haar feature used in OpenCV

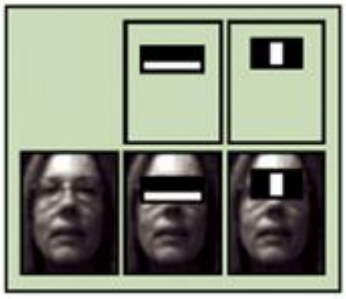

Fig3: The first two Haar features in the original viola Jones cascode

The actual rectangle combinations used for visual object detection are not true Haar wavlets. Instead, they contain rectangle combinations better suited to visual recognition tasks. Because of that difference, these features are called Haar features, or Haarlike features, rather than Haar wavelets. Figure 1 shows the features that OpenCV uses.

The presence of a Haar feature is determined by subtracting the average dark-region pixel value from the average light-region pixel value. If the difference is above a threshold (set during learning), that feature is said to be present.

To determine the presence or absence of hundreds of Haar features at every image location and at several scales efficiently, Viola and Jones used a technique called an Integral Image. In general, "integrating" means adding small units together. In this case, the small units are pixel values. The integral value for each pixel is the sum of all the pixels above it and to its left. Starting at the top left and traversing to the right and down, the entire image can be integrated with a few integer operations per pixel.

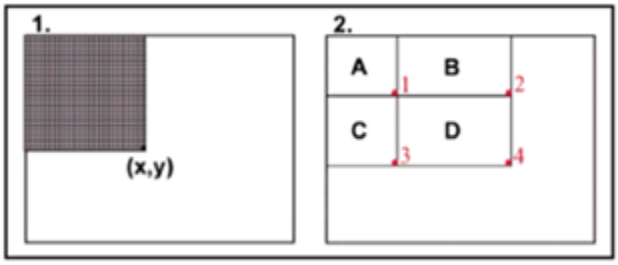

Fig4: Pixel view - Haar Algorithm Example

As Figure 4 shows, after integration, the value at each pixel location, (x,y), contains the sum of all pixel values within a rectangular region that has one corner at the top left of the image and the other at location (x,y). To find the average pixel value in this rectangle, you'd only need to divide the value at (x,y) by the rectangle's area.

But what if you want to know the summed values for some other rectangle, one that doesn't have one corner at the upper left of the image? Figure 2b shows the solution to that problem. Suppose you want the summed values in D. You can think of that as being the sum of pixel values in the combined rectangle, A+B+C+D, minus the sums in rectangles A+B and A+C, plus the sum of pixel values in A. In other words,

$$D = A+B+C+D - (A+B) - (A+C) + A.$$

Conveniently, A+B+C+D is the Integral Image's value at location 4, A+B is the value at location 2, A+C is the value at location 3, and A is the value at location 1. So, with an Integral Image, you can find the sum of pixel values for any rectangle in the original image with just three integer operations: (x4, y4) - (x2, y2) - (x3, y3) + (x1, y1).

To select the specific Haar features to use, and to set threshold levels, Viola and Jones use a machine-learning method called AdaBoost. AdaBoost combines many "weak" classifiers to create one "strong" classifier. "Weak" here means the classifier only gets the right answer a little more often than random guessing would. That's not very good. But if you had a whole lot of these weak classifiers, and each one "pushed" the final answer a little bit in the right direction, you'd have a strong, combined force for arriving at the correct solution. AdaBoost selects a set of weak classifiers to combine and assigns a weight to each. This weighted combination is the strong classifier.

Viola and Jones combined a series of AdaBoost classifiers as a filter chain, shown in Figure 3, that's especially efficient for classifying image regions. Each filter is a separate AdaBoost classifier with a fairly small number of weak classifiers.

The acceptance threshold at each level is set low enough to pass all, or nearly all, face examples in the training set. The filters at each level are trained to classifiy training images that passed all previous stages. (The training set is a large database of faces, maybe a thousand or so.) During use, if any

one of these filters fails to pass an image region, that region is immediately classified as "Not Face." When a filter passes an image region, it goes to the next filter in the chain. Image regions that pass through all filters in the chain are classified as "Face." Viola and Jones dubbed this filtering chain a cascade.

The order of filters in the cascade is based on the importance weighting that AdaBoost assigns. The more heavily weighted filters come first, to eliminate non-face image regions as quickly as possible. Figure 4 shows the first two features from the original Viola-Jones cascade superimposed on my face. The first one keys off the cheek area being lighter than the eye region. The second uses the fact that the bridge of the nose is lighter than the eyes.

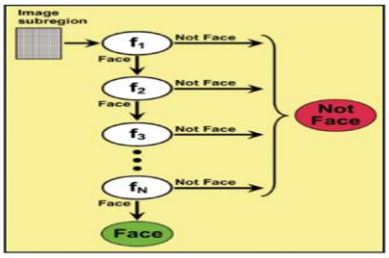

Fig 5: Haar Classifier cascade

4. Implementation of Haar Object Detection of ARM9 Board

4.1 Block Diagram

The ARM 9 friendly ARM board is the main component of this embedded system. The power supply is connected to Friendly ARM Board using an adaptor. The FriendlyARM Board is connected to a USB Hub. From USB HUB we can connect multiple devises with the Board. Here we have connected the Web CAM and Pen Drive with the USB HUB. When power supply is switched on FriendlyARM board starts up. The Webcam starts capturing the video and starts displaying on the Display screen. When a person or a face is detected the system will draw rectangle on the face on the display and will display count of number of faces detected on top of the rectangle faces. It can detect multiple faces. Once the system detects the face it captures the image and saves in the pen drive. This all happens on a real time basis. The Embedded system will keep on monitoring for occurrence of face through the Webcam and save the captured images with faces on a real time basis.

Once we have a data/images capture for a complete day, we can unplug the pen drive and connect to a PC having internet connection and run the SendMail java executable to take the images from the pen drive and send it to a remote email using GMAIL smtp.

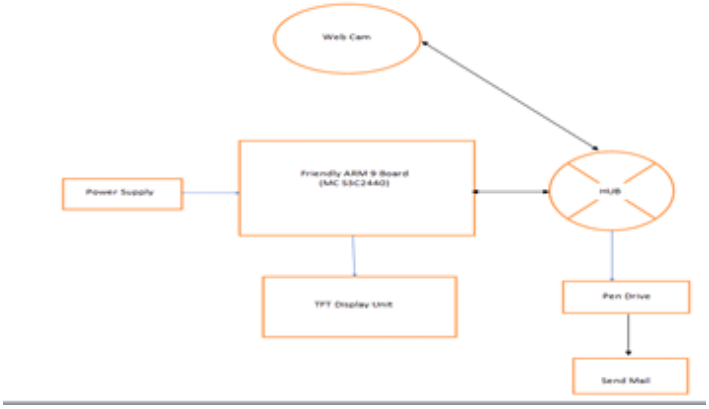

Fig 6: Block Diagram of the System

4.2 OpenCV Software

OpenCV (Open Source Computer Vision Library: http://opencv.org) is an open-source BSD-licensed library that includes several hundreds of computer vision algorithms. The document describes the so-called OpenCV 2.x API, which is essentially a C++ API, as opposite to the C-based OpenCV 1.x API. The latter is described in opencv1x.pdf.

OpenCV has a modular structure, which means that the package includes several shared or static libraries. The following modules are available:

- core - a compact module defining basic data structures, including the dense multi-dimensional array Mat and basic functions used by all other modules.
- imgproc - an image processing module that includes linear and non-linear image filtering, geometrical image transformations (resize, affine and perspective warping, generic table-based remapping), color space conversion, histograms, and so on.
- video - a video analysis module that includes motion estimation, background subtraction, and object tracking algorithms.
- features2d - salient feature detectors, descriptors, and descriptor matchers.
- objdetect - detection of objects and instances of the predefined classes (for example, faces, eyes, mugs, people, cars, and so on).
- highgui - an easy-to-use interface to video capturing, image and video codecs, as well as simple UI capabilities.
- gpu - GPU-accelerated algorithms from different OpenCV modules.
- some other helper modules, such as FLANN and Google test wrappers, Python bindings, and others.

The further chapters of the document describe functionality of each module. But first, make sure to get familiar with the common API concepts used thoroughly in the library.

The haar Algorithm can vbe easily implemented on ARM9 board using openCV.

4.3 Implemenation on Friendly ARM board

Hardware equipments

- Target Device: S3C2440 (Friendly ARM)
- Camera: Web Cam
- USB: Pen Drive, USB HUB
- Adaptor

Software equipments

- Programming Language Used: Embedded C++, Java
- Software's Used: OpenCV, QT
- Operating System: Linux
- Drivers: USB drivers, Display drivers.

Fig 7: Hardware Connection

5. COMPARISIONS

5.1 Comaprision between ARM9 and FPGA Implemenation

S.No.	ARM9 Implementation	FPGA Implementation
1	Microcontroller : ARM 9	Implemented on FPGA
2	Project is Implemented on friendly ARM 9 Board	FPGA, interfaced with an embedded Xilinx Micro blaze soft processor for I/O purposes
3	Form IO, ARM9 input output ports are used	Xilinx Micro blaze soft processor for I/O purposes
4	Processor : ARM 9 pipelined processor	Xilinx Micro blaze
5	Operating system : Linux	FPGA interfacing is used
6	Detects the face, counts the number of detected faces, and saves the image to an external memory	Detects the face
7	No External RAM	external DDR2 DRAM with a capacity of 256MB
8	Less power consumption	more power consumption
9	Cheaper	Costlier
10	Coded in Embedded C++	Coded in MATLAB
11	Highly extensible with the help programming	Not Much extensible based on programming.
12	Parallel processing is supported by default	Parallel processing is not supported by default
13	Slower when compared to FPGA Implementation	Faster when compared to ARM Implementation
14	External memory can be connected to store data	External memory cannot be connected to store data
15	Provision to transfer data to a remote location	No Provision to transfer data to a remote location

5.1 Comaprision haar Algorithm and SVM Implemenation

S.No.	Haar Algorithm	SVM Algorithm
1	Viola-Jones Detection and Haar Algorithm	Support Vector Machines (SVM)
2	Implemented on ARM based Microcontroller	Implemented on FPGA.
3	Easily and programmatically Extensible to many application.	For Every enhancement Hardware needs to be added,
4	Object detection Slower when compared to SVM	Faster
5	Easy to implement	Need a lot of training to implement
6	Detection accuracy is Very high. Near to 100%.	Detection accuracy < 80%
7	Can be implemented on any type of Hardware	Can only be implemented on FPGA.
8	It is based on Haar like features of objects	It is based on array of objects.
9	It is a well-tested and Proven Object detection algorithm	The Algorithm is still under revision due to hardware constraints and implementation cost
10	Low cost of implementation	High cost of implementation.

6. RESULTS

1. Successfully implemented and tested basic image processing on ARM9 Hardware.
2. The ARM 9 architectural support to Haar and Viola Jones algorithm is explored and implemented successfully in

the project

3. Captured the images and displayed it on TFT screen on real-time basis using ARM9, WebCam.
4. Detected multiple faces with low level features in the image and drawing rectangle on the detected faces.
5. Counted the number of detected faces on real-time basis and displayed it on the screen.
6. Successfully stored the images in a pen drive for every detected image by the ARM.
7. Scheduled automatic attachment of images stored in pen drive and able to send automated email using GMAIL SMTP when PC is connected to Internet.

Fig 8: Output Image stored in pendrive

6. Advantages And Applications

1. Image detection and counting of faces is performed using ARM9 and CMOS camera. ARM 9 uses 5 stages pipelining with significantly high frequency rates. It is in advanced technology and hence it is very reliable, fast and easy to implement.
2. The proposed embedded system can be used effectively as CCTV camera replacement. It stores only images rather than storing video. In this way memory used to store large videos is saved. And it becomes easy to maintain the system.
3. The maintenance cost is less; the system is comparatively easy to design hence engineering cost is less. The bulk production of the system is cheaper and hence the overall cost of the system decreases drastically. Hence it is a very cost effective approach of image processing and detection.
4. The system supports counting faces with low lever features. With a high resolution camera this system can be used at traffic lights controller to increase/ decrease the time duration based on number of human faces detected. The system can be effectively used while gathering data for auditing the traffic of human at airport, bus stand etc.
5. The system is designed and developed in such a way that it is easily extendable to many applications like sending the stored image via email to concerned authority in case of thief detection. The system stores the detected image in pen drive. The system is easily extendable to detect other objects.

7 Future Scope

We can aim to further explore the scalability of the system in terms of various parameters such as the input image size, parallel processing etc. by porting it to an ARM 11. The system can be made capable of detecting multiple objects by modifying the code and adding additional xml files.

8 Conclusion

This Paper presents an ARM based image processing engine for object detection which can achieve real-time performance while maintaining high detection accuracies. The architecture scales linearly to the hardware budget taking full advantage of ARM 9 modular and regular design, while providing true parallel processing for detection of face, counting and storing the image. The detected images which are stored in USB drive are automated to be attached in an email and on providing the email the images will be delivered to the email. Furthermore, the same ARM structure can be used for different applications regardless of the amount of objects to be detected. The ARM 9 architectural support to Haar and Viola Jones algorithm is explored and implemented successfully in the project. Additionally, using the enhanced version of the proposed architecture it can be configured to operate in a variety of modes and is able to adapt to different application demands (such as multiclass applications). As an immediate follow up to this work we aim to further optimize the ARM architecture for processing multi-dimensional Images. Overall the proposed architecture is capable of real-time ARM-based object detection while providing a configurable detection platform that can operate in a variety of embedded object detection scenarios, and adapt to specific application and designer demands with a very low cost of implementation.

References

[1] A Parallel Hardware Architecture for Real-Time Object Detection with Support Vector Machines, Christos Kyrkou, Student Member, IEEE, and Theocharis Theocharides, Senior Member, IEEE, IEEE TRANSACTIONS ON COMPUTERS, VOL. 61, NO. 6, JUNE 2012
[2] He Dong-feng, Ling Jie. Face Recognition : A Survey[J] of Mirocomputer Development, 2003,13(12):0075-0078
[3] P. Viola, M. Jones. Rapid object detection using a Boosted cascade of simple features. In: IEEE Conference on Computer Vision and Pattern Recognition, pp. 511-518, 2001.
[4] He Dong-feng, Ling Jie. Face Recognition : A Survey[J] of Mirocomputer Development, 2003,13(12):0075-0078
[5] Chen Jian-zhou, Li Li, Hong Gang, Su Da-wei. An Approach of Face Detection In Color Images Based on Haar Wavelet [J]. Microcomputer Information, 2005, 21(10-1):157-159.
[6] Cao Lin, Wang Dong-feng, Liu Xiao-jun, Zou Mou-yan. Face Recognition Based on Two-Dimensional Gabor Wavelets [J]. Journal of

Electronics & Information Technology, 2006, 28(3): 0490-0494.

[7] Chellappa R,Wilson C L,Sirohey S.Human and machine recognition of faces:A survey[J]. Digital Object Identifier,1995,83(5):705-741.