Ann. Data. Sci. (2015) 2(3):317–333
DOI 10.1007/s40745-015-0064-6

Modular Real-Time Face Detection System

Kaiyu Wang[1] · Zhiming Song[2] · Menglin Sheng[2] · Ping He[2] · Zhenan Tang[2]

Received: 2 December 2015 / Revised: 10 December 2015 / Accepted: 16 December 2015 /
Published online: 23 December 2015
© Springer-Verlag Berlin Heidelberg 2015

Abstract In this paper, a novel system architecture of face detection in possession of modular characteristic is proposed, and the corresponding face detection method is described, to match with the proposed architecture. First of all, the proposed architecture of face detection consists of two modules, namely, the coprocessor module of face detection based on FPGA and target system module, which hopes to implement finial face detection, based on general purpose CPU, and USB bus is used as the communication bridge between the two modules. Secondly, taking the characteristics of FPGA and general purpose CPU into consideration, face detection algorithm can be divided into two layers. The first layer of face detection algorithm based on skin color and eyes' graylevel variation is implemented in the FPGA, and then the corresponding detection results and image are transmitted to the second module by USB bus so as to further detect face using the algorithm combining principle component analysis with support vector machine, which is referred to as the second layer of algorithm. Because the second layer of the algorithms are operations of float-point and loop, it implemented in the general purpose CPU. This architecture enables face detection to be implemented not only in high performance computing platform in possession of USB bus interface, but also in small terminal products and low-end embedded systems, where the performance of processor and the resource of hardware are limited. Actual testing results show that the proposed system architecture can implement real-time face detection for the images with 640×480 resolution, and the detection accuracy is about 89 %.

✉ Kaiyu Wang
wkaiyu@dlut.edu.cn; challenge10760@163.com

1 Dalian University of Technology, Dalian 116024, China

2 School of Management, Harbin Institute of Technology, Harbin 150001, China

Keywords Modular face detection system · FPGA · The face detection based on SVM

1 Introduction

In practical face analysis systems such as face recognition system [1], expression recognition system [2], audio-visual speech processing system [3] and video surveillance [4], real-timely searching for face in the sequential images containing face and background is seen as the first step work of the systems, and the corresponding tasks of face analysis are implemented after that. The process of searching for face is called as face detection. Face detection is to search for faces with different expressions, sizes and angles in images in possession of complicated light and background, and then feeds back parameters of face. Clearly, the computation burden of face detection is considerable, so it will have an influence on the real-time performance of the face analysis systems.

Especially, with along the development of small terminal products and low-end embedded systems, the market share of the products can be broadened by transplanting face analysis applications into them. However, in terms of the products, real-time face analysis is almost impossible, resulting from that their performance of processor and resource of hardware are limited. Therefore, to make up the disadvantages, an efficient system architecture of face detection is indispensable.

In the past, most of the studies of face detection focused on software algorithms [5,6], and it was difficult to turn the study results into actual products as a result of those algorithms' complexity and real-time performance. Now, due to a wide range application of face detection, people begin with paying their attention to system architecture out of pure software algorithm, for example, the hardware architecture of face detection based on ASIC is designed [7], resulting in that face detection performance in speed is superior to the same algorithm version implemented in pure software. Re configurable hardware is taken as an efficient choice to implement face detection system, for example, the architecture composed of multi-FPGA chip is proposed to speed up face detection. Although pure hardware implementation for face detection can obtain an excellent result in speed, some weaknesses are inevitable, such as a long development cycle and a bad flexibility. So, hardware-software co-design is an alternative method for the reason that the features from software and hardware can be used sufficiently so as to speed up selectively partial algorithm using hardware. A system architecture combining FPGA soft core with user-defined hardware instruction is designed [9] to speed up the face detection algorithm, Adaboost. Similarly, an acceleration method of face detection combining FPGA soft core with hardware IP core is also proposed [10]. Although all the system architectures mentioned above can speed up the process of face detection to some extent, they are not suitable to small terminal products and low-end embedded systems for the reasons that [7] need consume 7 Watts of power [8], is expensive in cost [9,10], are in possession of high customization and their communication interface between software and hardware are constrained within FPGA chip, making it difficult to transplant the systems and conduct secondary developments. In light of the reasons, we explore a new system architec-

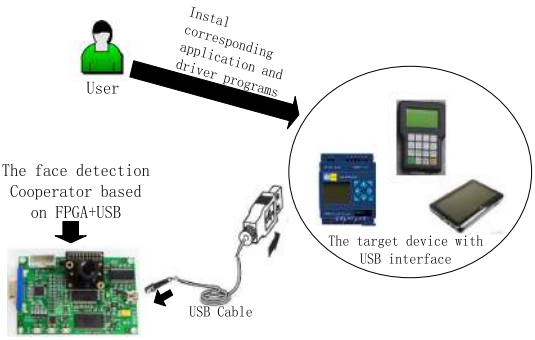

Fig. 1 The modular system architecture of face detection

ture of face detection totally different from the architectures mentioned above, and its design inspiration is from the modular design of robot [11]. In the modular design of robot, the robot system consists of independent function modules, which are reconfigurable and can exchange information by using standard interfaces. Therefore, we divide our face detection system into the two modules, face detection coprocessor module based on FPGA (the first module) and target system module based on general purpose CPU (the second module), and then use USB bus as the communication bridge between the two modules. This makes the first module based FPGA become a coprocessor of face detection with the USB characteristic, Plug and Play, so as to assist the second module with USB bus interface based on general purpose CPU in implementing real-time face detection. Figure 1. shows our proposed modular system architecture.

On the other hand, in accordance with the fact that FPGA has much greater performance than general purpose CPU in the data procession stream-oriented and the integer and fixed-point applications while general purpose CPU is better at performing complicated floating-point computation, long inner loops and irregular, indirect addressing than FPGA is [12], our proposed architecture adopts a face detection method, which is hierarchical, from coarse to fine, from part to whole. The face detection method consists of two layers. they are the first layer of algorithm based on skin color and eyes' graylevel variation, which is realized on FPGA, and the second layer of algorithm combining principle component analysis (PCA) with support vector machine (SVM), which is implemented on general purpose CPU, respectively. This makes it possible for the first layer of algorithm that after a frame entire image is sampled, the location of face is obtained, and then the second layer of face detection algorithm can acquire the location of face after reading the signals from the first layer so as to directly locate the area of face and further detect face. It is clear that this method can alleviate the computation burden of the second layer of algorithm and is efficient for some target system needed to realize real-time face detection, especially for small terminal products and low-end embedded systems.

The next section will be organized as follows: Sect. 2 will describe face detection algorithms and corresponding backgrounds related with this paper. Our proposed architecture and corresponding algorithm units are specified in detail in Sect. 3. Testing results are shown in Sect. 4 and conclusions are provided in Sect. 5.

2 Face Detection Algorithms and Backgrounds Related with this Paper

It is so easy for human beings to identify face in a complex context. However, with respect to computer system, this task is difficult as a result of many factors, such as unstable face features, uncertain face angle and gesture and changeful environment. Therefore, the robust and real-time algorithms of face detection always are seen as research focus. Many algorithms of face detection have been proposed, and herein we classify them as four different kinds [5,6]: face detection based on knowledge, face detection based on feature invariant, face detection based on template matching and face detection based on machine learning. The first two kinds of algorithms take advantage of facial features to detect face, such as geometrical characteristics, color characteristics. The third kind of algorithm predefines a template of facial features by manual, and face is detected by searching input image and comparing correlation between image searched and template predefined. The last kind of algorithm trains a classifier using face samples and non-face samples, and then the face is detected using the classifier. For the algorithms mentioned above, they have their own characteristics, for example, the face detection based on knowledge and the face detection based on feature invariant have relatively low detection accuracy, but their burdens of calculation is small and detection time is short. In contrast with the two kind of algorithms, face detection based on template matching and face detection based on machine learning are much more accurate, but they have a high calculation complication and a long detection time. So, in practical real-time systems of face detection, to detect face combining various algorithms is a great choice, for example [13], denotes that before using the face detection algorithm based on neural network, taking advantage of the face detection algorithm based on skin color can eliminate most of non-face domains to speed up the algorithm based on neural network, which is excellent in accuracy, but terrible in speed.

Our proposed face detection method combining numerous algorithms mentioned above consists of two layers of algorithms. First of all, the first layer of algorithm based on FPGA is described. For the purpose of simplicity, we just discuss about that there exits single front face in input images and resolution of face is about 200×200 pixels in the images. Herein, skin color of face is used to take input image as binary image, and then the number of pixels belonging to connected skin domain is counted using the labeling algorithm of connected domain to acquire a signal (skin_ok) of indicating whether the skin domain equivalent to face in size exits. On the other hand, the central coordinate of face can be acquired parallelly, and at the same time, the row coordinate which eyes locate also can be obtained on the basis of the characteristic that horizontal graylevel image complexity in the eyes' domain of face inner is the biggest. Next, another a signal (eye_ok) of indicating whether eyes exist is obtained by judging whether the distance between the row coordinate of face center and the row coordinate which eyes locate is within the range of a given threshold value. Finally, skin_ok and eye_ok are used to jointly acknowledge whether face exit (face_ok) . It is obvious that the process mentioned above is parallel and is suitable for implementing using FPGA. Therefore, after a frame image stream-oriented is inputted, the acknowledge signal (face_ok) of indicating whether face exist and corresponding coordinates surrounding facial rectangular frame are obtained.

Secondly, in our the second layer of face detection algorithm, the algorithm combining PCA with SVM is used, and it is an algorithm based on machine learning. It is well-know that for SVM, the burden of calculation is considerable, and the operations of float-point number and circular addressing is needed. Therefore, the second layer of algorithm is realized in general purpose CPU. What's more, the implementation of the second layer of algorithm is based on the results from the first layer of algorithms, which are transmitted by USB. This enables the second layer of algorithm to directly locate to the domain of face and then detect face using SVM. Clearly, the burden of calculation for the second layer of algorithm is decreased. On the other hand, SVM will be implemented in the method, off-line learning and on-line detection, and PCA is used to reduce the dimension of face eigenvector used by SVM. This is also beneficial to alleviate the burden of calculation for the second layer of algorithm. So, the second layer of algorithm can be easily implemented, making it possible for the second module to be realized in small terminal products and low-end embedded systems.

To verify the validity of the proposed algorithms composed of two layers, we just implement the second layer of algorithm in the general purpose PC. In the future, the second layer of algorithm is able to be transplanted into small terminal products and low-end embedded systems. In that situation, as shown in the Fig. 1, user only needs to install corresponding programs related to the second layer of algorithm in the second module and plugs the first module based on FPGA into the second module. The real-time face detection can be implemented in the second module.

3 The Architecture of our Proposed System

Figure 2 shows our proposed system architecture, it is obvious that the system consists of two modules isolated by the USB cable, and the two layers of algorithms mentioned above are implemented in the two modules, respectively.

In our system, image process stream-oriented is used. Namely, in the first module based on FPGA, face detection results can be obtained by single scanning image. So, Instead of storing intermediate results, SDRAM shown in Fig. 2 just is used to store the input original image. In this way, after a whole image is processed, the corresponding processed image still exists, which is beneficial to the display of VGA and insures that real-time face detection can be implemented without the loss of input image.

In the first module based on FPGA shown in Fig. 2, the first layer of face detection algorithm is the part surrounded by dot dash line, and it consists of the four units: coarse identification unit of face based on skin color, the calculation unit of face center point coordinate based on skin color, the calculation unit of eyes' graylevel complexity and the calculation unit of coordinates surrounding face's rectangle box, respectively. After a whole image is inputted, the signal (face_ok) of indicating whether face exist, the upper-left corner coordinate (left_x,left_y) and the lower-right corner coordinate (right_x,right_y) surrounding face's rectangle box are obtained. After that, face_ok, (left_x,left_y), (right_x,right_y) and the image signals stored in SDRAM are sent to USB interface chip by USB communication unit, and then these signals is transmitted to the second module.

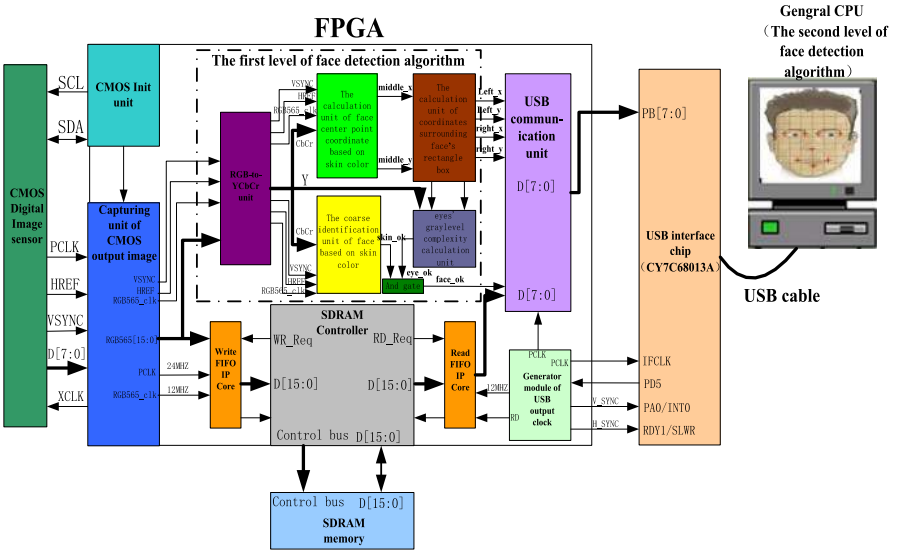

Fig. 2 The figure of system architecture

In the second module based on general purpose CPU (general purpose PC) shown in Fig. 2, the second layer of face detection algorithm combing PCA with SVM is implemented. What's more, the method, off-line learning and on-line detection, is used. Therein, off-line learning is based on Matlab and C language environment, and on-line detection is based on C language environment. Finally, the image are shown and corresponding face detection result is labeled.

In addition to the units mentioned above, in the two modules shown in Fig. 2, there are still other units. With respect to the first module, They are RGB to YCbCr unit, USB communication unit and SDRAM controller, respectively. For the second module, They are USB firmware program, driver program and application program, respectively. However, some units are not associated with face detection. So only the units related with face detection are discussed in the following sections.

3.1 RGB to YCbCr Unit

Skin color of face has obvious cluster characteristic. Along with the changes of race, age, and expression, the cluster characteristic is unchangeable. Because the cluster characteristic is from the color tone component of certain color spaces, the first step is to select color space. In our design, both color tone and luminance are needed to implement face detection based on skin and the calculation of eyes' graylevel complexity. So, the color space, YCbCr, is adopted.

In the first module shown in Fig. 2, the input image from CMOS digital image sensor is RGB565. To obtain YCbCr, a color space conversion unit, from RGB to YCbCr, is needed. The original conversion formula from RGB to YCbCr is a linear equations with float-point operations. Therefore, to implement the conversion formula

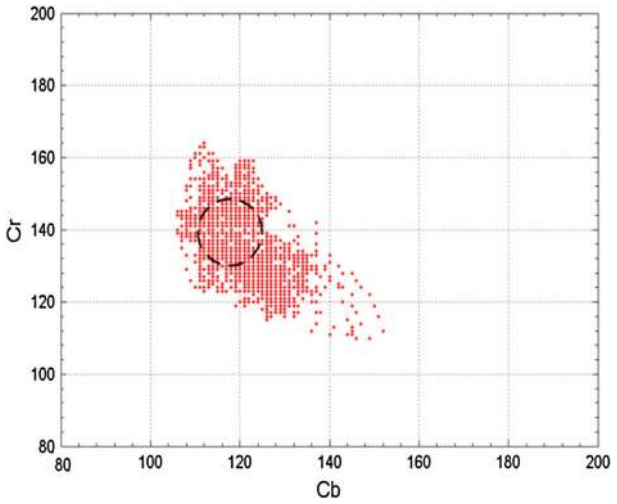

Fig. 3 The cluster characteristic of the sample skin model

based on FPGA, a simple transformation, from float-point to integer, with some losses in precision is used. This transformation can be given as in (1):

$$\begin{aligned} Y &= (66R + 129G + 25B + 4096) \gg 8 \\ C_{\mathrm{b}} &= (112B - 38R - 74G + 32768) \gg 8 \\ C_{\mathrm{r}} &= (112R - 94G - 18B + 32768) \gg 8 \end{aligned} \tag{1}$$

Where R, G and B are the components of RGB888 [a transformation from RGB565 to RGB888 is need before (1)], $\gg 8$ is a shift operation toward right 8 bits.

After obtaining YCbCr, the second step is to set up skin color model in the color space, YCbCr. Simple skin color model, single Gauss model, Gaussian mixture model and statistical histogram model often are used. In our design, due to image process stream-oriented and limited memory resource on-chip and off-chip, simple skin color model is adopted.

Figure 3 shows the simple skin color model constructed by us. It is a two-dimension system composed of the color tone component CbCr. Clearly, we can see that for skin color, CbCr gather in a small range. In our design, the clustering value of CbCr is selected as the part surrounding by dash line, and it is

$$\begin{cases} 108 \leq C_{\mathrm{b}} \leq 128 \\ 130 \leq C_{\mathrm{r}} \leq 150 \end{cases} \tag{2}$$

To verify whether Eq. (2) is valid, a input image shown in Fig. 4a is binarized using Eq. (2), and the binarization result is shown in Fig. 4b. From Fig. 4b, it is clear that skin domain and non-skin domain can be effectively segmented by Eq. (2).

Fig. 4 Binarizing the input image using the sample skin model

3.2 Coarse Identification Unit of Face Based on Skin Color

It has been known in the Sect. 2 that the second layer of algorithm is based on the results from the first layer of algorithms, such as face_ok, (left_x,left_y) and (right_x,right_y). However, face_ok is determined by the two signals shown in Fig. 2, skin_ok and eye_ok. Herein, Skin_ok indicates whether the skin domain equivalent to face in size exits and eye_ok indicates whether eyes in the skin domain exist. The joint of skin_ok and eye_ok can efficiently eliminate interferences from the domains similar skin color. Next, we will discuss about how to obtain the signal, skin_ok.

Skin domain and non-skin domain can be seen as binarization image after using Eq. (2), and in Sect. 2, it can be known that the resolution of face in a input image is about 200×200 pixels and the domain belonging to face is connected.Therefore, the labeling algorithm connected binarization domain can be used to count the number of pixels belonging to skin domain. When the number of pixels is greater than 35,000 (a threshold set based on that the resolution of face is about 200×200 pixels), skin_ok will output 1, indicating that face exist in the input image. The pseudocode counting the number of pixels belonging to the connected skin domain can be described as

$$
\begin{aligned}
&\text{if} \quad 108 \leq C_{\mathrm{b}}(x, y) \leq 128 \&\& 130 \leq C_{\mathrm{r}}(x, y) \leq 150 \\
&\quad C\text{ounter} <= C\text{ounter} + 1; \\
&\text{else} \\
&\quad C\text{ounter} <= C\text{ounter};
\end{aligned}
$$

where the range of $C_b(x, y)$ and $C_r(x, y)$ is from the Eq. (2), x and y are the row number and column number of current pixel in an image with the resolution 640×480, namely, $x \leq 480$ and $y \leq 640$. *Counter* is used for counting the number of pixels belonging to the connected skin domain.

After a whole frame image is inputted, if *Counter* is greater than 35,000, skin_ok will be equal to 1. Because in our design, pixel is inputted in the form of raster scanning and image process stream-oriented is used, the classic labeling algorithm of connected

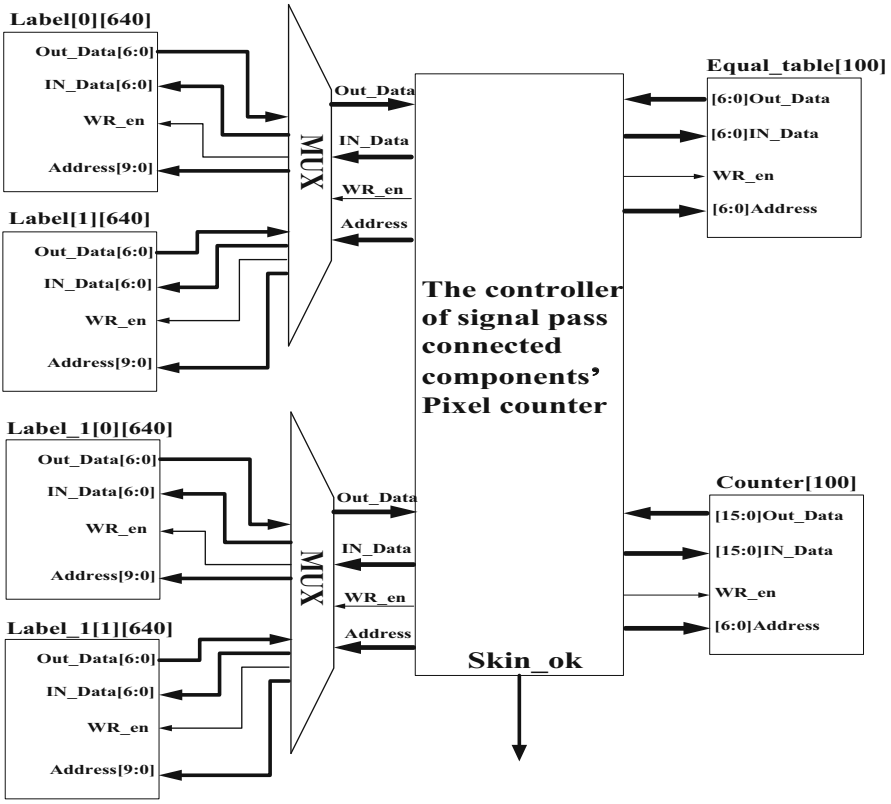

Fig. 5 The hardware block based on FPGA of the labeling algorithm of connected domain

binarization domain is not suitable for our design. In light of the algorithm named as single pass connected components analysis [14], an algorithm used to finish the process described in the upper pseudocode is implemented, and its hardware block diagram based on FPGA is shown in Fig. 5. For FPGA, the essence of this algorithm is a state machine used to read and write single port RAMs embedded in FPGA. Using this way, skin_ok can be obtained after single scanning image, and the cost of RAM embedded in FPGA is just about 20 Kb.

On the other hand, because pixel clock (the clock of CbCr) is 12 MHZ, to meet clock latency of single scanning, the clock of the block shown in Fig. 5 is selected as 144 MHZ.

3.3 The Calculation Unit of Face Center Point Coordinate Based on Skin Color

Traditionally, for the labeling algorithm of connected binarization domain, if the entire labeling table of connected domain is recorded, the center points of connected domains can be obtained by calculating the domains' the center of gravity. However, this will consume a lot of memories so that extra SRAM or SDRAM must be added into our

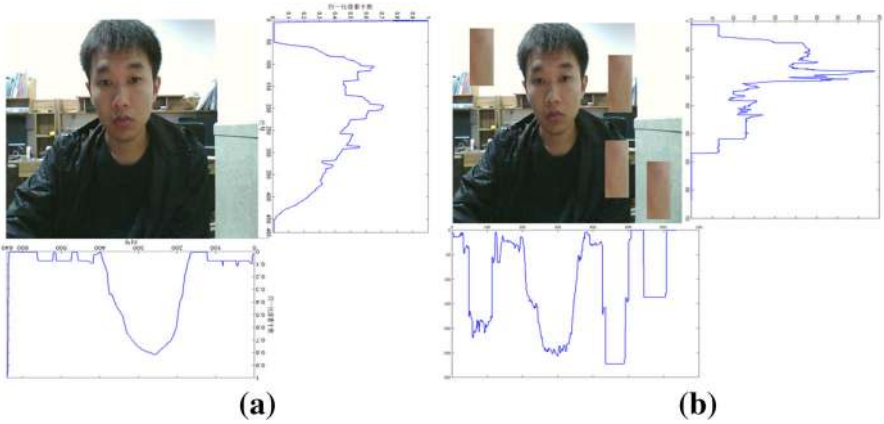

Fig. 6 The method of integral projection

system architecture shown in Fig. 2. On the other hand, the time requirement of single scanning image also can't be met.

Currently, we just consider that single front face exists in the input images. So, the method of integral projection can be used. Original integral projection is

$$H(x) = \sum_{y=1}^{640} f(x, y) \qquad 1 \le x \le 480 \tag{3}$$

$$V(y) = \sum_{x=1}^{480} f(x, y) \qquad 1 \le y \le 640 \tag{4}$$

where $f\ (x, y)$ can be seen as the pixel meeting the Eq. (2), namely the pixel of skin color. x is row number and its range is 1–480. y is column number and its range is 1–640. $H\ (x)$ indicates horizontal integral projection and $V\ (y)$ indicates vertical integral projection.

For an image, The result of original integral projection can be given in Fig. 6a. From the Fig. 6a, we can see that the location of the peak value of horizontal integral projection and vertical integral projection is the central location of face.

However, as shown in Fig. 6b, this original method is vulnerable to the interference of domains similar to skin color.

To eliminate the interference, based on the facts that there exists single front face in the input image, the resolution of the face is about 200×200 pixels and the pixels belonging to face are connected, a method combing connected pixels in one row with the constraint condition that face has the width 200 pixels is used. To describe this method, a simple example shown in Fig. 7 is used.

As shown in Fig. 7, Image [8] [16] denotes an image with 8 rows and 16 columns. The black dots are the pixels belonging to skin color, and the white dots are pixels indicating background. Assuming that the black circular domain located in the middle

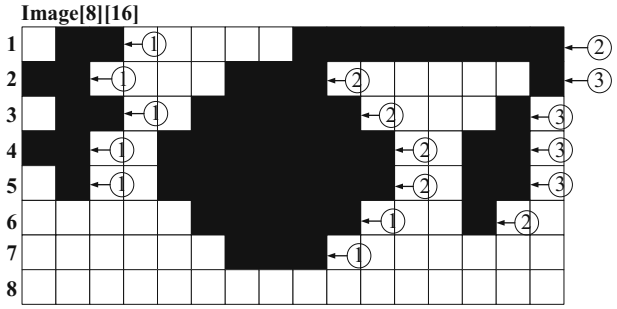

Fig. 7 A case of labeling connected row

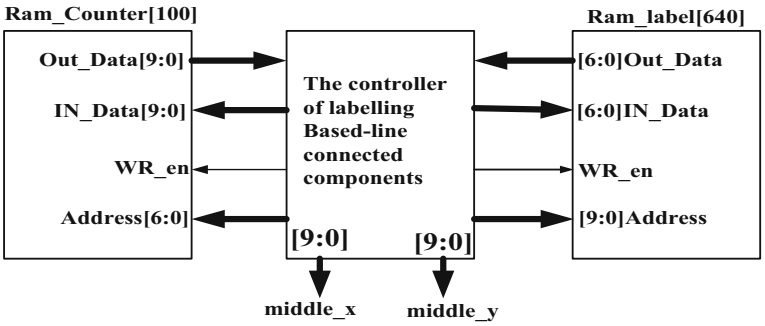

Fig. 8 The hardware block based on FPGA of the labeling algorithm of connected row

indicates face, and it has the width seven pixels. Our method is described in the following.

In the first row labeled as 1, there exists two connected rows labeled as ① and ②, and the horizontal integral projection of them are 2 and 8. So the biggest connected row is ②. using the same method, other rows' the biggest connected row can be obtained, they are 2—②, 3—②, 4—②, 5—②, 6—② and 7—①, respectively. After that, we can know that 1—② has the biggest horizontal integral projection among them, and its value is 8. However, in the upper assumption, the width of face is 7 pixels. So, the horizontal integral projection of 1—② is bigger than 7. So, instead of 1—②, 4—② is the biggest connected row. Namely, the central coordinate of face shown in the Fig. 7 is located in (4,8).

Similarly, this method can be used to our system by changing the constraint condition of the width of face from 7–200.

Figure 8 shows the hardware block diagram of this method based on FPGA. Its essence is also a state machine of reading and writing single port RAM. As shown in Fig. 8, Ram_Counter[100] is used to record the number of the connected rows in one row, and the number 100 indicates that the biggest number of the connected rows is 100. Each of address of Ram_Counter[100] corresponds to a connected row, and the data Out/In_Data corresponds to the value of the horizontal integral projection of the connected row. Ram_label[640] is used to record the connected labels of all pixels in one row, and the number 640 indicates that the number of pixel is 640 in one row.

Each of address of Ram_label[640] corresponds to a pixel, and the data Out/In_Data corresponds to connected labels of the pixels. On the other hand, because pixel clock (the clock of CbCr) is 12MHZ, to meet clock latency, the clock of the block shown in Fig. 8 is selected as 144MHZ.

3.4 The Calculation Unit of Eyes' Graylevel Complexity

Graylevel complexity is defined as the sum of gradient values of the pixels connected to each other in certain direction, and reflects the change degree of graylevel image in certain direction. For horizontal direction, its graylevel complexity can be expressed as in (5):

$$\Delta x = \sum_{y=1}^{M-1} |p(x, y+1) - p(x, y)| \qquad x \leq N, y \leq M \tag{5}$$

Where $p(x, y)$ is a gray pixel located in x row and y column. N and M indicate the number of row and column in a graylevel image.

In face inner, the row which eyes locate has the biggest horizontal graylevel complexity. As shown in Fig. 9, the row of the peak value of horizontal graylevel complexity is the row which eyes locate. Because for a face (in our design, resolution of face is 200×200), the distance between the row of eyes and the row of face center is fixed (in our design, the distance is about 50). After an entire image is inputted, the signal eye_ok can be obtained by checking whether the the distance between the row of eyes and the row of face center closes to 50. Next, an AND operation between eyes_ok and skin_ok is conducted to obtain the signal (face_ok) of indicating whether face exists.

Herein, it needs to be stressed that the color tone component CbCr and luminance component Y are obtained simultaneously, and eyes are located in the inner of the connected domain belonging to skin color. Therefore, when the central coordinate of face is calculated in unit of row, corresponding row's luminance component Y is stored. After all of pixels on one row are inputted, corresponding row's horizontal graylevel complexity is calculated using the period of horizontal blanking at a higher speed than pixel clock. In this way, when the center coordinate of face is obtained, the row which eyes locate can be acquired at the same time.

3.5 Other Units Based on FPGA Related with Face Detection

After center coordinate of face (middle_x,middle_y) is acquired, the upper-left corner coordinate (left_x,left_y) and the lower-right corner coordinate (right_x,right_y) surrounding face's rectangle box are obtained parallelly by the coordinates calculation unit of surrounding face's rectangle box shown in Fig 2. On the other hand, the USB communication unit is used to transmit image signals stored in SDRAM and the results ((left_x,left_y), (right_x,right_y) and face_ok) from the units of face detection based on FPGA to the USB2.0 interface chip shown in Fig. 2.

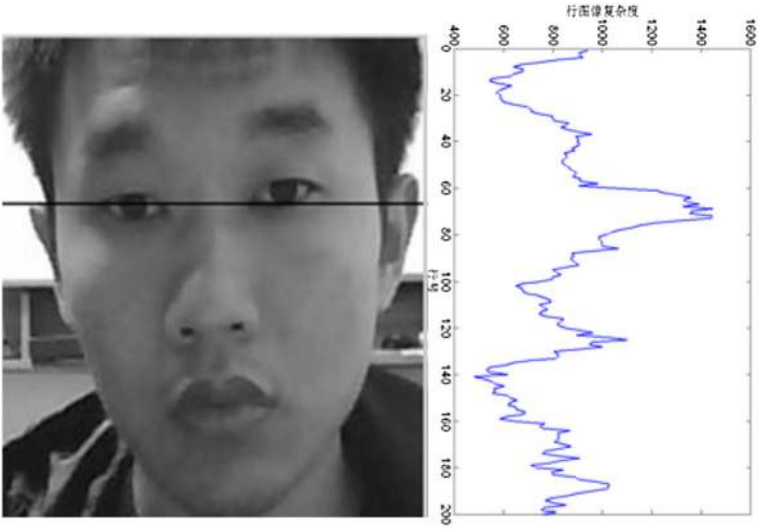

Fig. 9 Eyes' graylevel complexity

3.6 The Unit of Face Detection Algorithm Combining PCA with SVM

It is well-know that for the face detection based on SVM, a lot of time is spent in the process of training SVM. If this process is performed in detecting face, the real-time performance of system can not be met. So, the method, off-line learning and on-line detection, is used.

In our design, off-line learning is finished in the environment combing Matlab with C language program. First of all, our training sample images composed 1000 face images (positive sample) and 1000 non-face images (negative sample) are from the face database of MIT-CBCL, and these sample images are graylevel images with the resolution 19×19. Secondly, in Matlab, PCA is used to reduce dimension of the sample images from 361 to 20. In the process of reducing dimension using PCA, average face E(X), face subspace Eig_Face, training samples of SVM, T_SVM, and the labels of the training samples (+1 or -1), L_SVM, are obtained, and then saved into four textfiles. Thirdly, the training program of SVM based on C language reads the two textfiles, T_SVM and L_SVM, and then trains SVM, resulting in that the decision function of SVM (SVM_Model) is obtained, and is save into a textfile. Finally, In C language program, the second layer of real-time face detection algorithm based SVM is implemented by reading the three textfiles, E(X), Eig_Face, SVM_Model, and the results (image signal, face_ok, (left_x,left_y) and (right_x,right_y))from the first layer of face detection algorithm.

On the other hand, it must be noted that before implementing SVM in the second layer of face detection, the face image with resolution 200×200, which is from the first layer of face detection algorithm, is located by the two coordinates, (left_x,left_y) and (right_x,right_y), and then a normalization operation used to obtain the face image

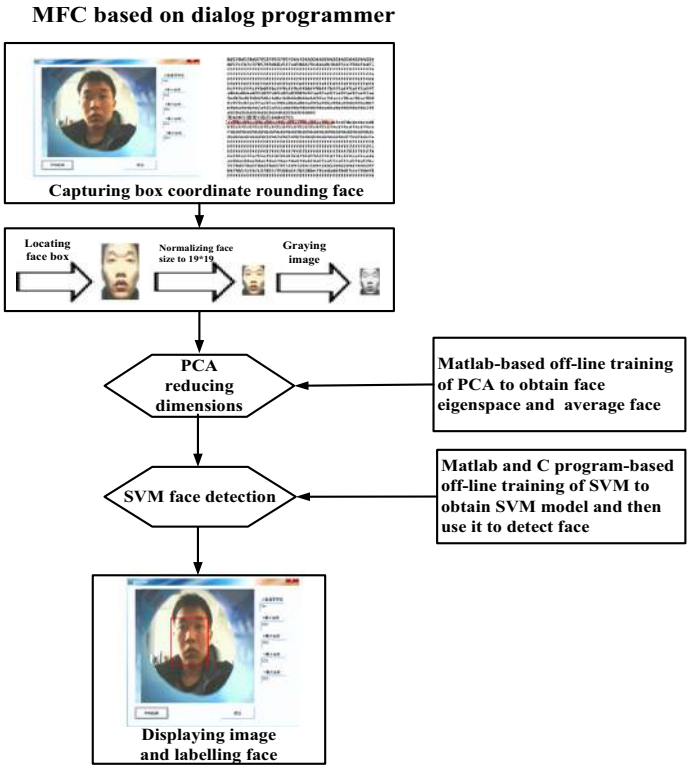

Fig. 10 The works of the second module

with the resolution 19 × 19 and a graying operation used to obtain graylevel image of face are implemented.

It is obvious that for the second layer of face detection algorithm, further detection result of face can be obtained after a normalization operation, a graying operation and multiply-accumulate loop operations (to implement PCA and SVM). This makes it possible for small terminal products and low-end embedded systems to real-timely detect face.

The works mentioned above of off-line learning and on-line detection are shown in Fig. 10.

4 The Synthesis Result of FPGA and the Testing Result of our Whole System

Verilog HDL hardware description language is used to implement the first module based on FPGA in the Altera FPGA chip, EP4CE10E22C8N, and A MFC dialog application is used to show the detection result of the second module in the general purpose PC.

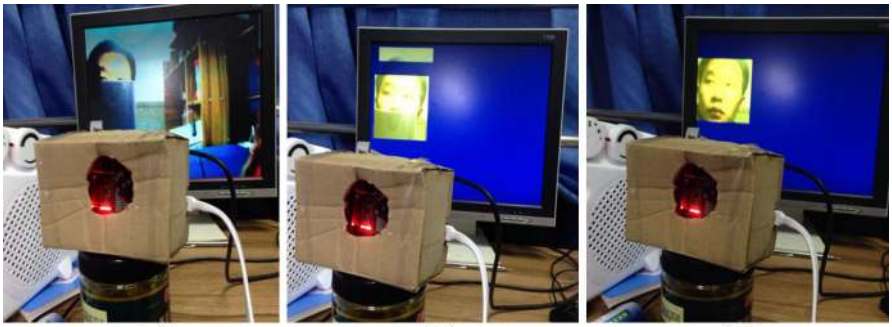

Fig. 11 The testing result of the first module based on FPGA

Table 1 The detection accuracy of system

The number of testing	Correct detection	False detection	Accuracy
1000	890	110	89 %

After synthesizing, placing and routing by using the Altera synthesis tool quartus II 13.0, the result that our first module just consumes eleven per cent of total hardware resource of FPGA can be obtained. On the other hand, to verify whether our first module can detect face real-timely and efficiently, the USB communication unit shown in Fig. 2 is replaced with VGA driver unit, and corresponding detection result is outputted to VGA display. The detection result can be given in Fig. 11. On the basis of Fig. 11, we can see that when face is sheltered, face and background can be displayed at the same time, but just face is displayed after removing the screen.

The performance of the whole system is tested in the second module. First of all, the detection accuracy of system is tested in the form of sampling single image, and corresponding result is shown in Table 1. Based on Table 1, we can see that the detection accuracy of the whole system is 89 %. Secondly, to verify whether the first module can improve the performance of the second module, the time of implementing one correct face detection is measured, and results are shown in Table 2. From Table 2, we can see that if the all face detection algorithms used in two modules is transplanted into the second module based on general purpose PC, the time of realizing one correct face detection is 0.093 s. In contrast to that, when the first module based on FPGA is added, the time of realizing one correct face detection decrease to 0.031 s. It is obvious that the first module based on FPGA can improve the performance of our system. Finally, the testing photo of our whole system is shown in Fig. 12, and the picture of real object of our system are shown in Fig. 13. From Fig. 12, we can see that although the arm belonging to connected skin color exists in the image, face still can be detected correctly.

Table 2 The comparison in performance

Testing case	The time of finishing face detection (s)
FPGA + PC	0.031
Just PC	0.093

(a) (b)

Fig. 12 The testing result of our whole system

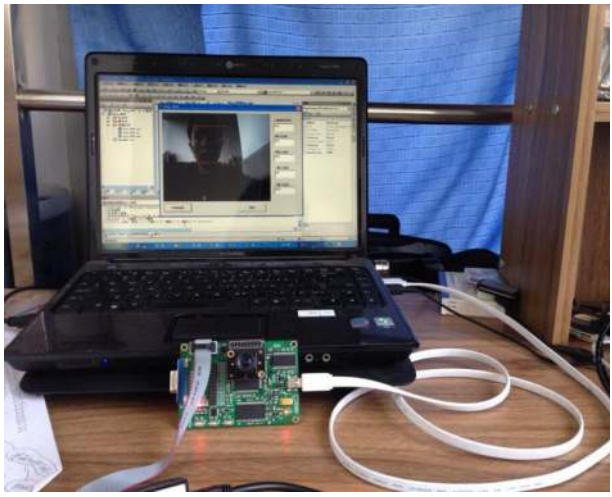

Fig. 13 The real object of our system

5 Conclusion

This paper proposes a novel system architecture of face detection, which is in possession of modular characteristic. What's more, the first module based on FPGA can be seen as a USB device with the characteristic, plug and play, making it possible to assist the target systems with USB interface, in implementing real-time face detec-

tion, especially the target systems with limited performance of processor and hardware resource. Testing results show that for the input image with the resolution 640×480, real-time face detection can be implemented, and detection accuracy is 89 %.

Acknowledgments Thanks to the support by National Natural Science Foundation of China (Nos. 70771030 and 70271047) and Project Science Foundation of Guangzhou University.

References

1. Wechsler H, Phillips JP, Bruce V, Soulie FF, Huang TS (eds) (1998) Face recognition: from theory to applications[M]. Springer, New York
2. Buenaposada JM, Muñoz E, Baumela L (2008) Recognising facial expressions in video sequences. Pattern Anal Appl 11(1):101–116
3. Nguyen D, Halupka D, Aarabi P et al (2006) Real-time face detection and lip feature extraction using field-programmable gate arrays. Syst Man Cybern Part B 36(4):902–912
4. Kim TK, Lee SU, Lee JH, et al. (2002) Integrated approach of multiple face detection for video surveillance[C]//pattern recognition, 2002. Proceedings of 16th international conference on IEEE, 2: 394–397
5. Hjelmås E, Low BK (2001) Face detection: a survey. Comput Vis Image Underst 83(3):236–274
6. Yang MH, Kriegman D, Ahuja N (2002) Detecting faces in images: a survey. Pattern Anal Mach Intell IEEE Trans 24(1):34–58
7. Theocharides T (2006) Embedded hardware face detection[D]. The Pennsylvania State University
8. McCready R (2000) Real-time face detection on a configurable hardware system. Springer, Berlin
9. Bigdeli A, Sim C, Biglari-Abhari M et al (2007) Face detection on embedded systems[M]// Embedded Software and Systems. Springer, Berlin pp 295–308
10. Acasandrei L, Barriga A (2012) FPGA implementation of an embedded face detection system based on LEON3[J]
11. Yim M, Shen WM, Salemi B et al (2007) Modular self-reconfigurable robot systems [grand challenges of robotics]. Robot Autom Mag IEEE 14(1):43–52
12. Underwood KD, Hemmert KS (2004) Closing the gap: CPU and FPGA trends in sustainable floating-point BLAS performance[C]//Field-Programmable Custom Computing Machines, 2004. FCCM 2004. 12th Annual IEEE Symposium on IEEE, 219–228
13. Rowley HA, Baluja S, Kanade T (1998) Neural network-based face detection. Pattern Anal Mach Intell IEEE Trans 20(1):23–38
14. Bailey DG, Johnston CT (2007) Single pass connected components analysis[C]//Proceedings of image and vision computing New Zealand, pp 282–287