Available online at www.sciencedirect.com

SCIENCE DIRECT®

ELSEVIER

Real-Time Imaging 10 (2004) 81–94

www.elsevier.com/locate/rti

Real-time face detection and tracking for mobile videoconferencing

Stavros Paschalakis*, Miroslaw Bober

Mitsubishi Electric ITE B.V. Visual Information Laboratory, The Surrey Research Park, 20 Frederick Sanger Road, Guildford, Surrey GU2 7YD, UK

Abstract

This paper addresses the issue of face detection and tracking in the context of a mobile videoconferencing application. While the integration of such technology into a mobile videophone is advantageous, allowing face stabilization, reduced bandwidth requirements and smaller display sizes, its deployment in such an environment may not be straightforward, since most face detection methods reported in the literature assume at least modest processing capabilities and memory and, usually, floating-point capabilities. The face detection and tracking method which is presented here achieves high performance, robustness to illumination variations and geometric changes, such as viewpoint and scale changes, and at the same time entails a significantly reduced computational complexity. Our method requires only integer operations and very small amounts of memory, of the order of a few hundred bytes, facilitating a real-time implementation on small microprocessors or custom hardware. In this context, this paper will also examine an FPGA implementation of the proposed algorithmic framework which, as will be seen, achieves extremely high frame processing rates at low clock speeds.
© 2004 Elsevier Ltd. All rights reserved.

1. Introduction

The main aim of videoconferencing is to enhance aural communication through the addition of visual information in the form or real-time or near-real-time video of the faces of the users. Third-generation video-enabled mobile handsets have the potential to change the limitations of the conventional phone, in both business and private use, but the visual communication technology has to be integrated seamlessly into the terminal, so that its use does not create additional and unnecessary complications for users. More specifically, such terminals are hand-held and usually non-stationary, so speakers have to ensure that the camera is pointing in the correct direction, which can be a distraction to a conversation and possibly negate the benefits of the added video. Another problem is picture instability caused by the relative motion between the hand-held camera and the speaker, which may not be successfully compensated for by the viewer. One other issue is the required bandwidth, which can be dramatically reduced through the encoding and transmission of only the part of each video frame which contains the object of interest, i.e. the face of a speaker, instead of the entire frames. In turn, this eliminates the need for unnecessarily large display units that accommodate the entire field of view of the cameras and also facilitates the appropriate scaling of the face images to the display dimensions, so that a complete face image is always presented to the viewers.

The deployment of real-time face detection and tracking technology in mobile videophones has the potential of addressing all the aforementioned issues. The problem of automatic face detection is by no means a new one but it is only in recent years, with the continuous increase in sophistication and decrease in cost of computer and video acquisition systems, that this difficult topic has been extensively addressed by researchers, resulting in the development of numerous and diverse face detection methodologies with widely varying characteristics, such as robustness to illumination variations, scale changes, viewpoint changes and computational complexity. With regards to computational complexity, a number of techniques have been reported in the literature which entail undemanding implementations, in the context of modern computer systems, but still usually assume the availability of modest processing power, memory, floating-point arithmetic capabilities, etc.

*Corresponding author. Tel.: +44-1483-885816; fax: +44-1483-579107.
E-mail address: stavros.paschalakis@vil.ite.mee.com (S. Paschalakis).

1077-2014/$ - see front matter © 2004 Elsevier Ltd. All rights reserved.
doi:10.1016/j.rti.2004.02.004

While such requirements may be reasonable in some applications, this will not generally be the case for mobile videoconferencing, considering the limitations of the general purpose processing hardware present in such power-limited devices, which will also have to be shared with other tasks such as video coding/decoding [1] and error management [2], along with the demand for an ever-decreasing physical size for the handsets. In this context, this paper presents a high-speed, high-performance and low-cost face detection and tracking method. The proposed approach is robust to illumination variations and geometric changes, such as scale and viewpoint changes, and entails a very low computational load. More specifically, the algorithm entails only integer operations, most of which are additions of a very low dynamic range, and very low memory requirements, in the order of only a few hundred bytes. These characteristics make the proposed method suitable for implementation on simple microprocessor architectures and also facilitate a custom hardware implementation. Hence, this paper will also examine an FPGA-based custom hardware implementation of the proposed algorithmic framework which, as will be seen, combines a low implementation cost with very high processing speeds at low clock speeds.

The rest of this paper is organized as follows. The following section will give a general overview of the face detection problem and very briefly examine the main approaches which have emerged towards addressing it. Our proposed face detection and tracking method will then be presented in detail. Then, the FPGA design and implementation of the proposed method will be considered, following which a discussion of the salient points of this work will conclude the paper.

2. Face detection methodologies

Historically, the problem of face detection cannot be described as a new one, since it has been an active area of study in the computer vision arena for more than 20 years. However, it is only in recent years, with the continuous increase in sophistication and decrease in cost of computer and video acquisition systems, that this difficult topic has been extensively addressed by researchers, resulting in the development of numerous and diverse face detection methodologies. The face detection problem may be generally defined as follows. Given an arbitrary digital image, the aim is to determine the location of all the faces (if any) in this image, regardless of the 3-dimensional position and orientation of a human head (provided that at least a partial face is still projected onto the image plane), presence or absence of structural components (such as beards, moustaches and glasses), facial expressions, partial occlusions of the face, and imaging conditions (such as changes in the scene illumination and shadows). Although, given the state of the art, a solution to the above problem is theoretically not unfeasible, in practice, the problem of face detection acquires a definition within each specific application, which usually implies a reduced set of design parameters and a more practical solution. As to what such application areas might be, examples include video conferencing, video coding, crowd monitoring, area surveillance, person recognition, identity verification, and automated lip-reading, to name but a few.

Although different researchers provide different categorizations for the various face detection methodologies proposed over the years [3,4], most methods may be broadly categorized as appearance-based or feature-based. Appearance-based approaches [5–21] treat face detection more as a pattern recognition problem and do not generally incorporate human knowledge regarding face appearance and facial characteristics, but rely on techniques from statistical analysis and machine learning in order to ascertain the characteristics of human faces. It is, therefore, not surprising that the set of appearance-based face detection methods which appear in the literature is actually based on well-established pattern recognition techniques. The methods which fall into this category employ techniques such as principal component analysis [5–10], factor analysis and linear discriminant analysis [11], neural networks [13–16], support vector machines [17,18] and hidden Markov models [19–21].

With feature-based methods faces are detected in images through the detection of facial features such as the eyes, the eyebrows, the nose and the mouth, or parts or combinations thereof [22–39]. Such detection methods usually rely on combinations of low-level image processing operations, such as edge detection and spatial filtering, used in conjunction with a priori heuristic rules regarding the structure of the human face. Within this category, a subdivision to three classes may be performed. These three categories are top-down, bottom-up and colour-based methods. With top-down methods, faces are detected in an image by first obtaining face candidates based on the known global appearance of a face and then verifying these candidates by searching for specific facial features, or combinations or parts thereof, within each one [22,23]. Conversely, the characteristic of bottom-up methods is that they initially search an image for facial feature candidates or, more generally, low-level features, and then attempt to group those candidates into constellations which represent faces [25–31].

Finally, the facial feature in question with colour-based methods [32–39] is primarily the colour of the human skin. The advantage of skin colour-based face detection methods is that, while computationally efficient, they are generally robust to scale and in-plane rotation transformations of the face, viewpoint changes, complex backgrounds, illumination changes and are suitable for the detection of multiple faces in an image. For this reason, colour-based techniques may be used on their own or in more sophisticated systems which also incorporate other feature-based or appearance-based techniques. Various studies [35,37,38] have demonstrated that the skin of different people and with different ethnicities, e.g. Caucasian, Asian and African, differs significantly more with respect to intensity rather than chrominance. Therefore, colour-based face detection methods usually transform the original 3-dimensional RGB colour space to a chrominance–luminance space and then employ only the 2-dimensional chrominance space for segmentation. Skin colours are detected in an image by employing a skin colour model, derived from training skin samples. This model usually takes the form of a Gaussian or mixture of Gaussian density functions which model the skin clusters in the chosen colour space, or is simply a histogram in the chosen colour representation system, whereby histogram projection is employed for skin detection. An even more straightforward alternative which appears in the literature [34] is to examine the skin colour histogram and derive thresholds for the colour components, which make up a skin "bounding box", although this approach may produce less satisfactory results.

3. Face detection and tracking algorithm

The proposed face detection and tracking method adopts the colour-based face detection philosophy, due to its combined high performance and computational efficiency potential. Fig. 1 shows that the proposed method may be decomposed into three main stages, namely subsampling, skin filtering and face detection/tracking.

3.1. Subsampling

The first step is the subsampling of the original digital video frame, whereby the image is divided into non-overlapping blocks of a fixed size and the pixels values within each block and each colour band are averaged to produce a single pixel in the subsampled image. The aim of this subsampling is twofold. First, it aims at reducing the amount of data for the subsequent processing, facilitating a faster implementation. Secondly, the subsampling step results in the generation of larger skin patches which, in turn, decrease the susceptibility of the method to the existence of facial features, such as a moustache, and structural features, such as glasses. Clearly, the subsampling factor must be carefully chosen in order to reach a balance between computational efficiency and detection performance. For example, our

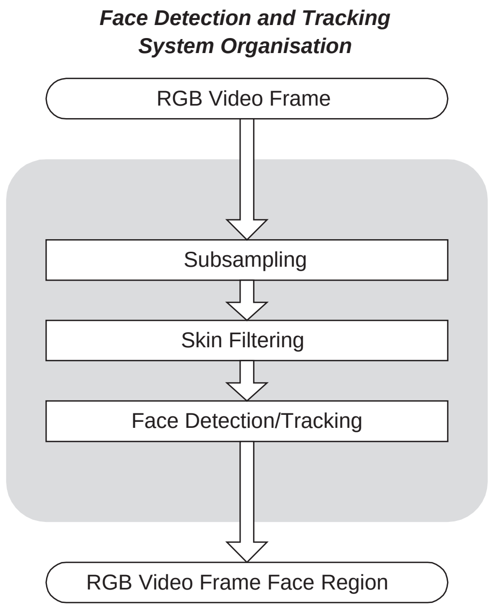


Fig. 1. Organization of the proposed face detection and tracking method.

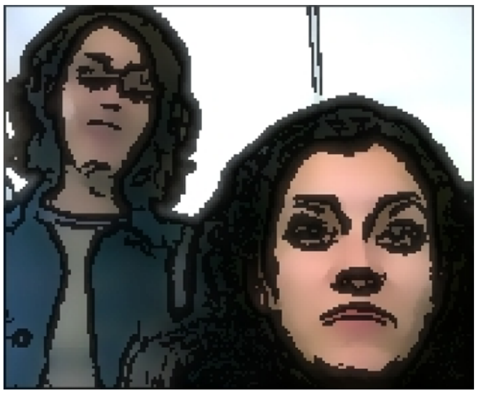


Fig. 2. Example QCIF image containing two faces. This is a typical image for a mobile videoconferencing setting.

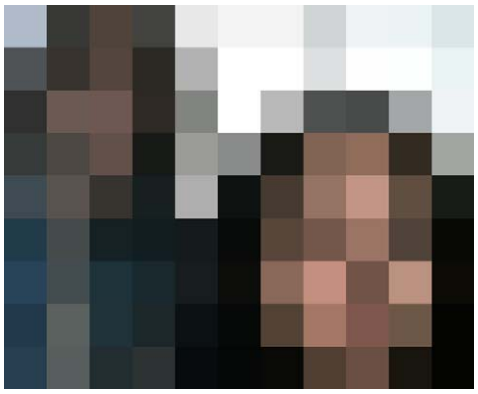


Fig. 3. Subsampled version of the image of Fig. 2 using a 16×16 subsampling mask.

current implementation assumes a video signal of QCIF resolution, i.e. 176×144 pixels, for which 16×16 pixel subsampling, resulting in a 11×9 pixel video frame, achieves this balance. As an example, Fig. 2 shows a QCIF image containing the faces of two people and Fig. 3 shows its subsampled version as described above. Note that all the face images have been "cartoonized" for the purposes of this paper in order to hide the identities of the subjects.

3.2. Skin filtering

The subsampling of the original video frame is followed by the conversion of the subsampled image from its original colour space to a second colour space for the purpose of colour-based skin detection. The current implementation assumes that the original video is in RGB format, most commonly provided by video cameras. Desirable characteristics of the second colour space, which is used for skin filtering, include some degree of insensitivity to illumination variations and the clustering of the skin colours of different people and with different ethnicities (e.g. Caucasian, Asian and African) in one or more relatively compact regions. To this end, a number of appropriate colour coordinate systems have been proposed in the literature, including the normalized rg chromaticity space [39], CIE LUV (discarding the Luminance component) [35], HSV and HSI (discarding the Value and Intensity components, respectively) [34,36], STV (discarding the Value component) [33], among others. The colour representation system employed here is a 2-dimensional adaptation of the 3-dimensional log-opponent colour space IR_gB_y which has been successfully used for the purpose of skin detection [40], termed LogRG for simplicity and defined as

$$\begin{aligned} L_1 &= L(\mathrm{G}) + L(\mathrm{RGB}_{\max}), \\ L_2 &= L(\mathrm{R}) - L(\mathrm{G}) + L(\mathrm{RGB}_{\max}) \end{aligned} \tag{1}$$

with $L(x) = \log_{10}(x+1)\ 10^6\ 2^{-15}$
and $\mathrm{RGB}_{\max}$= maximum grey value.

Assuming a 24-bit colour depth for the original RGB frame, the combined colour transformation and scaling formulae of (1) produce L_1 and L_2 values in the range [73 … 146] and [0 … 146], respectively. The function $L(x)$ may be efficiently implemented as a lookup operation into a 256 byte (256×8-bit) table. We have found that this colour representation system compares favourably to two colour spaces which are commonly used for skin detection, namely the 2-dimensional normalized rg and HS (of HSL) colour spaces.

More specifically, of the two main approaches towards colour-based skin detection, i.e. histogram projection and the creation of a Gaussian-based skin colour model, the former is adopted here. A recent study [37] has shown that histogram-based models are not only more computationally efficient than Gaussian-based models, but also offer slightly higher detection performance. With the histogram-based approach, a skin colour histogram is first created offline using manually segmented skin pixels. The bin values are then normalized to some range, e.g. the integer range [0 … 255]. During skin detection, an unknown pixel is projected to the skin colour histogram and the corresponding bin value is obtained, with a higher value indicating an increased likelihood that the unknown pixel is a skin pixel.

Thus, we created skin colour histograms in the LogRG, rg and HS spaces using a training database of $\sim 4.99 \times 10^6$ skin pixels, representing various Caucasian skin tones under unconstrained natural and artificial illumination conditions. Two different evaluations were then performed. First, an evaluation database of $\sim 2.19 \times 10^6$ skin pixels was used in conjunction with a non-skin database of $\sim 42.28 \times 10^6$ pixels to assess the performance of the models. This skin evaluation database represents various Caucasian skin tones under unconstrained natural and artificial illumination conditions, while the non-skin database was created from randomly selected images not containing humans. In this evaluation, both the LogRG and the rg skin colour models achieved an equal error rate (EER) of $\sim 6\%$, with the HS model achieving a higher EER of $\sim 8\%$. The minimum false rejection (FR) was found to be $\sim 0.13\%$ for LogRG, with a corresponding maximum false acceptance (FA) of $\sim 31.42\%$, $\sim 0.17\%$ for rg, with a corresponding maximum FA of $\sim 37.76\%$, and $\sim 0.02\%$ for HS, with a corresponding maximum FA of $\sim 36.87\%$. In general, we found the three models performing equally well, except in terms of the maximum FA, where the LogRG model performs significantly better. In the second evaluation, we addressed the issue of a more drastic deviation from the skin tones used for the creation of the skin colour models. Thus, a new evaluation database of $\sim 1.66 \times 10^6$ skin pixels was employed, which represents various non-Caucasian skin tones (i.e. African, Asian, Middle-Eastern, etc.) under unconstrained natural and artificial illumination conditions. This was used in conjunction with the same non-skin database described above. In this evaluation, the EER was found to be $\sim 14\%$ for the LogRG model, $\sim 16\%$ for the rg model and $\sim 15\%$ for the HS model. The minimum FR was found to be $\sim 5.03\%$ for LogRG, with a corresponding maximum FA of $\sim 31.42\%$, $\sim 6.16\%$ for rg, with a corresponding maximum FA of $\sim 37.76\%$, and $\sim 2.73\%$ for HS, with a corresponding maximum FA of $\sim 36.87\%$. As before, we found the three models performing comparably. This second evaluation also verifies the fact the models are robust not only to illumination variations but also to very different skin tones, arising from people of different ethnicities. Obviously, an additional advantage of the LogRG colour system, especially for a hardware implementation, is the simplicity of the calculations that it entails for a conversion from RGB (just three 8-bit integer additions and two table lookup operations), as well as the fact that only the R and G planes of the original video frame need subsampling, which also facilitates a faster and more economic implementation.

With our face detection and tracking system, each pixel value in the subsampled LogRG frame is projected to a LogRG skin colour histogram and the corresponding bin value is obtained, with a higher value indicating an increased likelihood that the unknown pixel is a skin pixel. A threshold is then used to give rise to a binary skin probability value, i.e. 0 for skin and 1 for non-skin. In terms of implementation, especially for a custom hardware implementation, two optimizations are possible. First, instead of using a skin colour histogram in conjunction with a threshold, a thresholded binary histogram may be used instead. This greatly reduces the storage requirements for the model. Secondly, a useful property of the LogRG skin colour model is that skin colours are concentrated in a relatively small area of the entire plane. Hence, only that area needs to be incorporated in the skin colour model, with values lying outside it being automatically assigned a skin colour probability of 0. In this implementation, the ranges of the colour coordinates L_1 and L_2 used in the skin model are [115 … 146] and [73 … 88], respectively, giving rise to a 32×16 window. This further reduces the storage requirements for the model. Consequently, the skin filtering process is reduced to a lookup operation into a 64 byte (512×1-bit) memory, created by vectorizing the aforementioned region of the complete colour plane. As an example, the result of this operation, referred to as a skin map, on the image of Fig. 3 may be seen in Fig. 4.

Despite the robustness of the skin colour model to illumination variations and the incorporation of various skin tones captured under varying illumination conditions, colour-based skin detection may produce non-optimal results under changing illumination conditions

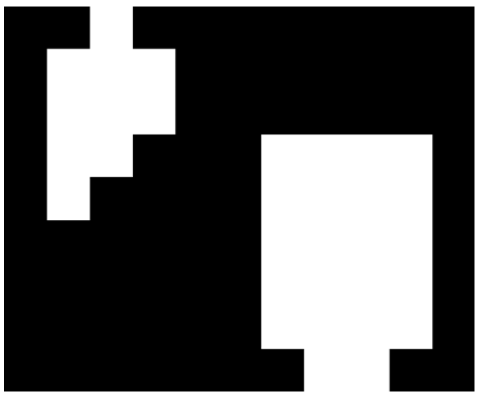


Fig. 4. Skin map for the subsampled image of Fig. 3.

when a static skin colour model is employed. This gives rise to the need for an adaptive model which may be calibrated to the skin tone of different people and to different lighting conditions. With the proposed system this calibration is user initiated and relies on LogRG converted pixels of the original frame instead of its subsampled version, and more specifically, on the pixels which are contained within a small predefined region of fixed size, e.g. 20×20 pixels. Obviously, the user of the system which incorporates the algorithm will be aware of the dimensions and location of this region and of the fact that it should contain a skin sample for successful calibration. An alternative to this approach is the automatic and continuous adaptation of the skin colour model based on the skin region identified in each frame, although this would still require a user initiated calibration at the beginning of the operation of the system. Although this method is currently under investigation, the known difficulty with such an approach is that the algorithm can "lock on" and calibrate from a non-skin but skin-colour-like region, or part thereof, resulting in a cumulative error and, finally, the complete instability of the skin colour model, whereby user intervention will again be required [39].

One difficulty with the proposed skin colour model adaptation approach is that the calibration area may often contain a number of pixels which do not correspond to skin and should not be used in the calibration of the model. Since manual segmentation is not as easy an option as for the offline skin colour model creation process, the adverse effects of this problem are alleviated by requiring that $R \geqslant G$ and $R \geqslant B$ for a pixel to be used in the calibration process, a heuristic rule that has been found to be generally applicable to different skin types and most common lighting conditions. More specifically, it was found that this simple rule applies to $\sim$99.86% of all the skin pixels used during the skin colour model creation and evaluation, i.e. $\sim 8.84 \times 10^6$ skin pixels representing Caucasian/Asian/African/Middle-Eastern/etc. skin tones under unconstrained natural and artificial illumination conditions. Thus, for the adaptation of the model, the calibration pixels which adhere to the aforementioned heuristic are transformed to the LogRG space as described above and incorporated into the skin colour model. A question that arises is how these values are incorporated into the existing model in view of the fact that this model is comprised of binarized values. In the general case, the adaptation of the pre-thresholded model requires the creation of an equivalent calibration model followed by the fusion of the two models, e.g. simple averaging, weighted averaging, simple averaging after sinking of the original model, so that the new data can have a significant impact, etc. With our binarized model such a process is not an option.

The solution which we have identified entails a form of compression of the pre-thresholded model. More specifically, all the colours of the original model may be separated into four different categories or bands, namely $C_0 \ldots C_3$, each of the four bands encoding (i) the 0/1 skin colour probability for the given colour, (ii) how this probability will change during an initial sinking process before the model is calibrated, and (iii) how this probability will again change based on the number of occurrences of the given colour within the calibration area during the model adaptation process. In more detail, C_0 represents LogRG colours which have a skin colour probability of 1 in the original model and will remain unchanged during the calibration. C_1 represents LogRG colours which have a probability of 1, which will be changed to 0 when the calibration step is initiated and then changed back to 1 if the given colour is encountered once during the calibration process. The logic for C_2 is the same as for C_1, but two encounters are required to update probabilities from 0 to 1. Finally, C_3 pixels have an original skin colour probability of 0 which will remain 0 when the calibration step is initiated and will be changed to 1 if the given colour is encountered three times during the calibration process. The storage of this information allows not only the efficient adaptation of the skin colour model, but also the recovery of the original model at any point, and the entire model requires only 320 bytes (512×5-bit) of storage. It should be noted that, although the original model may be calibrated any number of times, it is not possible to calibrate an already calibrated model. However, this is not a major drawback since we have found that this kind of repeated adaptation actually degraded the overall performance of the algorithm after this operation had been preformed a few times. Clearly, this compression of the original model is not unique, but has been chosen to strike a balance between performance and efficiency, in terms both of storage and of associated computations. One could chose different compression parameters, e.g. increase number of bands, to strike a different balance.

Finally, an interesting point one might raise regarding the deployment of this method in a mobile videophone handset is how the user controlled model adaptation process can actually be monitored by a user, especially while communicating with a second party, since the handset display will be showing the video transmitted by the other handset. A simple solution to this is to optionally superimpose a scaled down version, e.g. at 25% or 33%, of the tracked face of each handset on the same handset at one of the corners of the display, allowing users to simultaneously view the other person as well as the video that they are transmitting.

3.3. Face detection/tracking

The creation of the skin map is followed by a spatial filtering process for the purpose of noise reduction. With the current implementation, this filtering is a neighbourhood operation and entails resetting a skin pixel to a non-skin pixel if it has less than four skin pixels among its eight neighbours and setting a non-skin pixel to a skin pixel if all of its eight neighbours are skin pixels. The result of this filtering on the skin image of Fig. 4 may be seen in Fig. 5.

A connected component analysis algorithm is then applied to the spatially filtered skin map. The most common and effective algorithm of this type is commonly referred to as a "floodfill" operation which, in its most general form, entails a recursive implementation and is computationally demanding. The algorithm which is currently employed aims at computational simplicity, for the benefit of both a real-time software-based implementation as well as to facilitate a fast and efficient hardware implementation, and produces distinct skin regions by assigning different numerical tags to skin pixels. These tags are subsequently assigned to different groups in order to form skin regions. Briefly, the skin image is scanned left-to-right and top-to-bottom and each skin pixel is assigned a numerical tag $T_0 \ldots T_n$. The determination of this tag is based on the tags of four of its neighbours, namely the three pixels above and the one pixel immediately to the left. Because of the scanning method employed, examining the other neighbours would serve no purpose, since no tags will have been assigned to them at this point. If one or more of the aforementioned neighbours is a skin pixel, then the skin pixel under consideration adopts the tag of the lowest numerical value among the tags of its neighbours, e.g. if the choice is between T_0 and T_1, then T_0 is chosen for the pixel under consideration. If none of the aforementioned neighbours is a skin pixel, a new tag is assigned to the pixel in question, e.g. if T_0 and T_1 have been previously used, the pixel under consideration is assigned T_2.

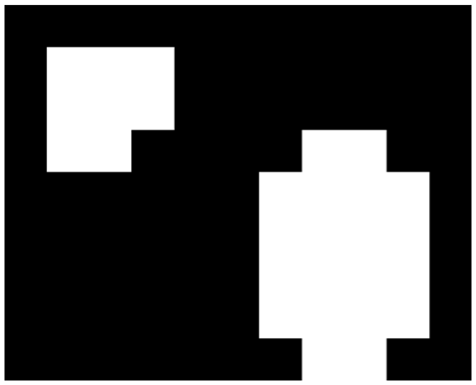


Fig. 5. The skin map of Fig. 4 after spatial filtering.

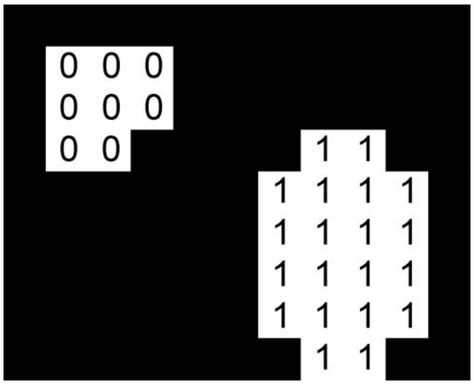


Fig. 6. The result of connected component analysis on the filtered skin map of Fig. 5. Two skin groups have been identified, namely G_0 and G_1.

The grouping algorithm also maintains a tag-group association array, which indicates to which of the skin groups $G_0 \dots G_n$ the pixels of a given tag correspond to. When a new tag is generated for a pixel with no skin neighbours, it automatically becomes associated with the skin group of the same numeric value, e.g. T_2 with G_2. Now assume, for example, that a pixel is assigned tag T_1, associated with skin group G_1, and that subsequent processing reveals that this pixel is actually neighbouring a T_0 pixel, associated with skin group G_0. Rather than changing the tags of all the T_1 pixels to T_0, the tag-group association array is updated to indicate that T_1 is also associated with skin group G_0. Then, when the processing of the skin image is complete, the tag-group association array reveals which skin pixels make up each group. For example, Fig. 6 shows the final output of the connected component analysis process on the spatially filtered skin image of Fig. 5.

This simple grouping algorithm has some limitations. Only a small number of tags can be created, especially for a custom hardware implementation. The current implementation employs eight tags ($T_0 \dots T_7$), one of which (T_7) is reserved for non-skin pixels. This number of tags was chosen with a 11×9 pixel skin image in mind, arising from the 16×16 pixel subsampling of the original QCIF video frame. Although it is theoretically possible for an 11×9 pixel skin image to require more than the effective seven tags for its skin pixels, extensive testing showed that, in practice, that situation does not easily arise. In fact, in over 100 h of operation of our prototype system at 30 frames/s, this situation never actually arose. In the unlikely event that this situation is encountered its effects will still not be detrimental since it will simply result in the abandonment of the processing of just one particular frame, whereby the position of the tracked face may be interpolated from previous frames. Furthermore, it should be noted that the computationally simple grouping algorithm which is currently employed is targeted to our face detection and tracking problem and is limited when compared to more complicated connected component analysis algorithms, i.e. it may occasionally be unsuccessful in connecting all the pixels of highly irregular shapes. Nevertheless, the skin maps of human faces do not generally give rise to highly irregular shapes, especially for image dimensions in the order of tens of pixels and after spatial filtering, while the real-time operation of the system does not allow such errors to become apparent. During the extensive evaluation of our prototype, as mentioned above, such connected component analysis problems did not become evident.

Having identified the disjoint skin regions of the skin map, the next step is the calculation of certain statistics of each skin group, which are then used by the face tracking process. These statistics are the zeroth order geometric moment m_{00}, i.e. the mass, and the first-order moments m_{10} and m_{01} [41], and are calculated as

$$m_{00} = \sum_x \sum_y f(x,y),$$

$$m_{10} = \sum_x \sum_y x \cdot f(x,y),$$

$$m_{01} = \sum_x \sum_y y \cdot f(x,y), \tag{2}$$

where $f(x,y)$ is the skin map

$$\text{and } f(x,y) = \begin{cases} 1 & \text{for skin pixels,} \\ 0 & \text{for non-skin pixels.} \end{cases}$$

Another optimization is that the calculation of these statistics need not be performed at such late a stage in the processing. More specifically, with the current implementation these statistics are actually calculated during the scanning of the skin image for the purpose of tag assignment and for the pixels of each separate tag rather than each separate group. Then, when the whole skin image has been processed, the information stored in the tag-group association array is used to calculate the statistics for the skin groups as sums of the statistics of the appropriate skin tags. Following the calculation of the statistics for each skin region, the centroid of each region is calculated as

$$\bar{x} = \frac{m_{10}}{m_{00}}, \quad \bar{y} = \frac{m_{01}}{m_{00}}. \tag{3}$$

The centroids of the skin regions for the image of Fig. 2 are shown in Fig. 7. The face tracking process is based on distance and mass measurements. Thus, the distance between the centroid of each skin region and the centroid of the face of the previous frame is calculated as the maximum axis distance d, given by

$$d = \max(\mathrm{d}x, \mathrm{d}y),$$

with $dx = |\bar{x} - \bar{x}_{prev}|, \ dy = |\bar{y} - \bar{y}_{prev}|,$

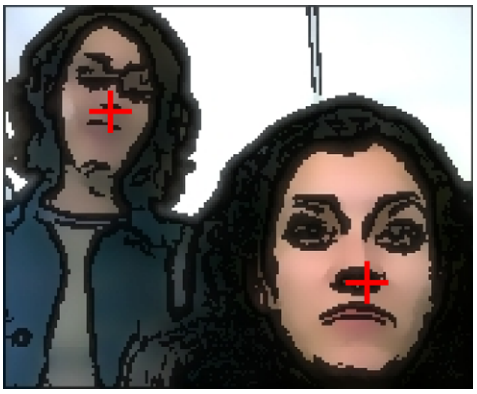


Fig. 7. The identified face centroids for the example image are indicated by the crosses.

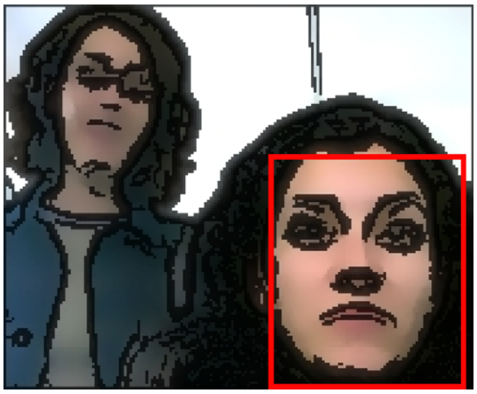


Fig. 8. The selected region for encoding and transmission is identified by the bounding rectangle.

where $(\bar{x}, \bar{y})$ is the centroid of a skin region of the current frame, (4)

and $(\bar{x}_{prev}, \bar{y}_{prev})$ is the centroid of the face of the previous frame.

The skin group which produces the smallest distance d is selected as the tracked face of the current frame, unless that skin region does not also have the largest mass among the skin regions for 10 consecutive frames. In that case, the selected face becomes the maximum mass face rather than the minimum distance face. This allows the system to always "lock on" to the most prominent skin region which, given the targeted mobile videophone application, is most likely to be the region of interest, while the hysteresis function employed for the switching from one region to another prevents flickering among skin regions of similar sizes. Also, another special situation is the absence of any skin pixels following the spatial filtering of the skin probability map and before grouping. In this event, the system assumes the last known face to be the current face. The same applies in the event that the processing of a frame needs to be abandoned due to the exhaustion of all the available skin tags as described earlier.

Finally, the coordinates of the centre of the tracking window are calculated. With the current implementation, these are not the same as the coordinates of the centroid of the selected face for each frame. The reason for this is that those coordinates are calculated on a 11×9 skin image and scaling them up to the dimensions of the original image will result in the display window moving in increments of 16 pixels, resulting in a poor visual effect. Instead, the display coordinates are calculated as the average of the centroid coordinates of the face region for the current frame and a number of previous frames, the total number of frames used in the averaging process currently being set to eight. This allows smooth face tracking, as well as smooth switching between faces. With regard to the size of the tracking window, this may be easily obtained from the area of the tracked face or it may be fixed to a pre-determined size. Keeping in mind our target mobile videophone application, and the associated display size limitations for the videophone handsets, the current system assumes a fixed tracking window of quarter QCIF resolution with portrait aspect ratio, i.e. 72×88 pixels. As an example, for the image of Fig. 2 and assuming the face on the right is selected as the tracked face, the region selected for encoding and transmission is identified by the bounding rectangle in Fig. 8. Fig. 9 shows how the transmitted image appears on the display of the handset of a second user, with the small image at the bottom right corner showing the identified face region of the second user, allowing both users to simultaneously view the other person as well as the video that they are themselves transmitting.

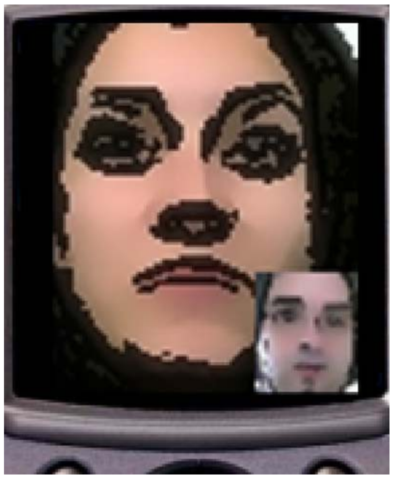


Fig. 9. An example of how the transmitted image appears on the display of the handset of a second user. The small image at the bottom right corner shows the identified face region of the second user, allowing both users to simultaneously view the other person as well as the video that they are themselves transmitting.

4. FPGA implementation

The face detection and tracking method presented in the previous section has been designed to achieve high performance at a low computational cost, relying only on simple integer arithmetic operations and table lookup operations, which facilitates its deployment on small microprocessors or custom hardware. This section will examine the custom hardware realization of the proposed algorithmic framework, and more specifically, an FPGA implementation, which has been carried out as a feasibility study towards a VLSI implementation. The FPGA family used is the ALTERA APEX20 K, although this is not restrictive, since the features that have been utilized are basic and common across most contemporary FPGA devices. This FPGA design can be viewed as comprising three main modules, corresponding to the three main algorithmic stages of the previous section, namely subsampling, skin filtering and face detection/tracking.

4.1. Subsampling

The circuit presented here has been designed and optimized for the processing of QCIF 176×144 pixel 24-bit RGB images using a 16×16 mask. Although the subsampling of the original frame is a neighbourhood rather than a pixel-based operation, the circuit has been designed so that it is able to process data directly from a camera without reliance on frame storage. The module assumes that pixels are delivered on a row-by-row basis, as is usually the case, e.g. left-to-right and top-to-bottom, and colour band values are delivered sequentially rather than in parallel, e.g. in an R-G-B order.

Fig. 10 shows that the module is effectively comprised of two accumulator units, namely ACC_1 and ACC_2.

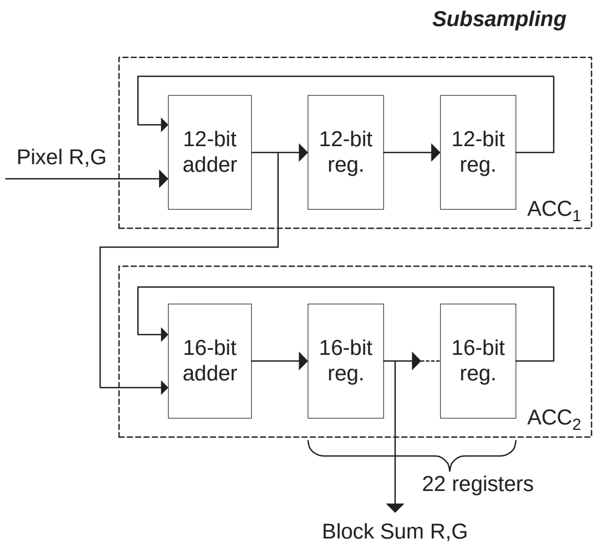


Fig. 10. Organization of the subsampling circuit.

ACC_1 accumulates the R and G pixel values which make up each row of each block of pixels. As seen earlier, only the R and G planes of the original frame require subsampling, since the B component is not used in the subsequent colour conversion and skin filtering process. Taking advantage of the fact that these pixel values are delivered sequentially and in a fixed order, ACC_1 does not contain two complete accumulators but is comprised of a single 12-bit full adder and a 2×12-bit shift register. The second accumulator unit ACC_2 accumulates the final sum for each row of each pixel block as produced by ACC_1 and therefore calculates the R and G pixel value sums for entire pixel blocks. Similarly to the previous accumulator unit, ACC_2 is made up of a 16-bit adder and a 22×16-bit shift register. The size of all the components has been chosen so that an overflow does not occur for the aforementioned image and mask characteristics. The upper byte of each final block sum produced by ACC_2 contains a required pixel value for the subsampled image. The processing time for each pixel byte is a single clock cycle and the module can process values continuously, regardless of transitions between rows or frames. The appropriate control of the register enable signals also allows the module to be paused at any point and for any number of clock cycles, allowing it to adapt to different camera timings and specifics. Overall, the control logic for this circuit is minimal. This architecture was found to be very effective for the image and subsampling mask characteristics mentioned earlier. Clearly, changing those characteristics may require modifications in order to achieve the most efficient implementation.

4.2. Skin filtering

The skin filtering circuit processes the subsampled image values as they are produced by the subsampling circuit and does not require the storage of the complete image. According to the previous algorithmic description, the tasks of this circuit are to transform the R and G values of the 11×9 subsampled image or 176×144 original image to the LogRG colour space and then use the new colour values to obtain skin colour probabilities or to adapt the original skin colour model to the current skin tone and lighting conditions, respectively. Consequently, the circuit has two distinct modes of operation, namely skin filtering and skin colour model calibration.

Before considering the implementation of the skin filtering module, it is useful to consider the implementation of the skin colour model itself. As seen earlier, the skin model requires 320 bytes (512×5-bit) of storage. More specifically, this is organized into three memory components. The first is a 512×1-bit RAM which holds the probability values for the current model. This will be referred to as “Skin Model Array 1” or “SMA_1”. There is also a 512×2-bit ROM which holds the category that each colour belongs to in the original model, as discussed earlier. This memory element will be referred to as “Skin Model Array 2” or “SMA_2”. Finally, a 512×2-bit RAM is also required to keep track of how the categories of the different colours are updated as the skin colour model calibration proceeds and, consequently, how the skin colour probabilities should be changed. This will be referred to as “Skin Model Array 3” or “SMA_3”.

Although both modes of operation rely on the same hardware, it is more convenient to first consider the skin filtering datapath, as shown in Fig. 11. The first step is the logarithmic transformation $L(x)$ of (1) of the R and G subsampled image values. This simply entails a lookup into a 256×8-bit ROM which outputs the values $L(x)$ as 8-bit integers. The processing time for each value is a single cycle, and the appropriate control circuitry allows zero or any number of cycles between the delivery of the two values of each RG pair. The remainder of the skin filtering process requires two cycles. In the first of these two cycles, Cycle 0, the logarithmically transformed R and G values are transformed to the LogRG colours L_1 and L_2 according to (1). Both values are also appropriately offset so that they can be used as lookup addresses to SMA_1 for the retrieval of a skin colour probability. In the final cycle of the skin filtering process, Cycle 1, the lower five bits of L_1 are appended to the lower 4 bits of L_2 to form a 9-bit address for an asynchronous read of the skin model memory SMA_1, which outputs a binarized skin colour probability, with 1 corresponding to a skin pixel and 0 to a non-skin pixel. An additional check is also performed to determine whether the complete L_1 and L_2 values are actually within the model range and, if not, the skin colour probability is cleared to 0. The circuit is pipelined and, consequently, there can be zero or any number of clock cycles between the delivery of any two RG pairs, regardless of row or frame transitions.

The calibration datapath of the skin filtering circuit can be seen in Fig. 12. The first step in the skin colour model adaptation process (not shown) is the combined retrieval and sinking of the original model probabilities, based on the colour category information stored in SMA_2, which are then stored in SMA_1. Also, the colour category information of SMA_2 is loaded into SMA_3. The combinatorial logic for these operations is trivial, as discussed in the algorithmic description section. Similar to the skin filtering process, the first step in the processing of a pixel during calibration is the logarithmic transformation of the R and G original pixel values, as described above for the subsampled values. Some additional checks are also performed to ensure that the pixel in question should be used in the calibration process, as shown in Fig. 12. The remainder of the calibration process requires two more clock cycles. The first of those cycles, Cycle 0, sees the calculation of the LogRG colours L_1 and L_2 in exactly the same fashion as

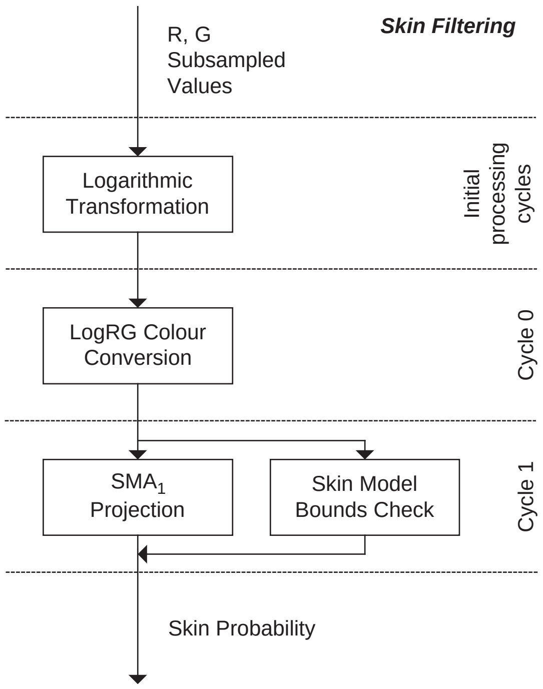


Fig. 11. The skin filtering datapath.

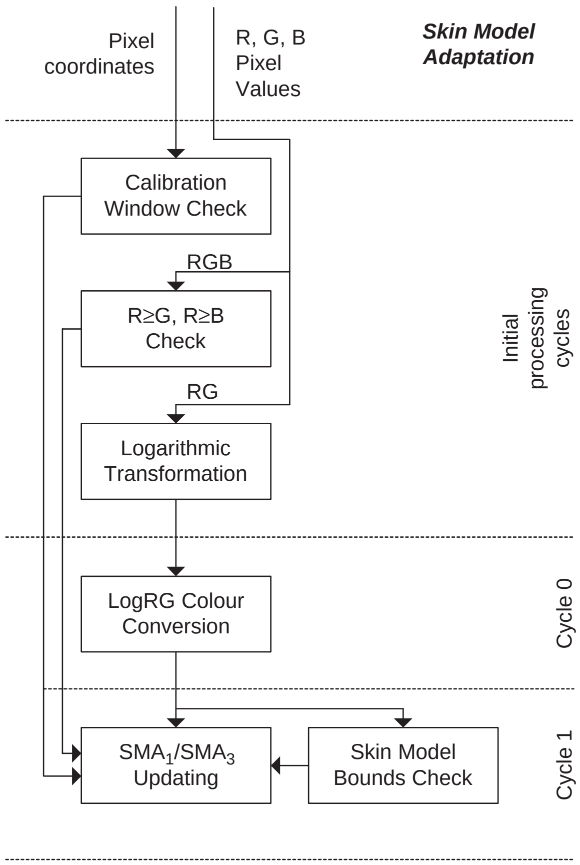


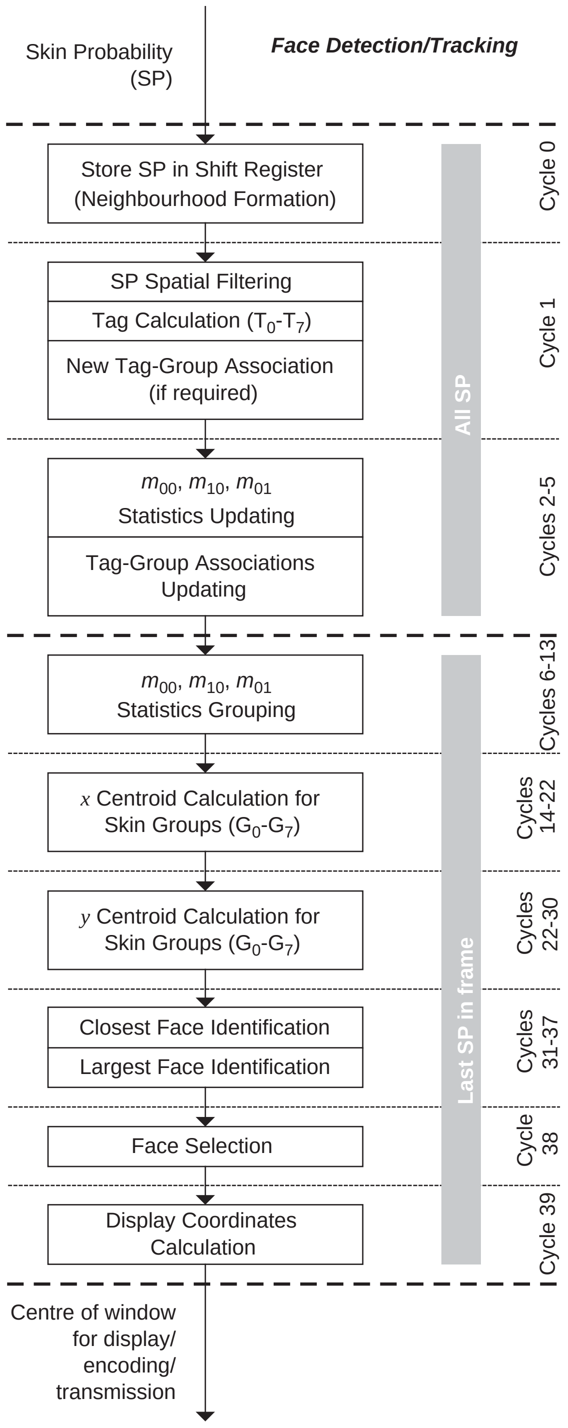
Fig. 12. The skin colour model adaptation datapath.

for the skin filtering procedure. In the next cycle, the L_1 and L_2 values are used to access SMA_1. Then, if it has been determined that the pixel should be used in the calibration process, the temporary colour category information of SMA_3 is updated and, if appropriate, the corresponding skin probability stored in SMA_1 is also updated. The retrieval of the original skin colour model, without sinking and subsequent calibration, is also straightforward with this organization. As before, the pipelining of this path allows zero or any number of clock cycles between the delivery of pixel bytes or RGB triplets, regardless of row or frame transitions.

4.3. Face detection/tracking

The tasks of the face detection/tracking circuit are to spatially filter the 11×9 binary skin map produced by the previous stage in the system, perform connected component analysis, calculate statistics for each skin region, identify the skin region in the current frame which corresponds to the face of the previous frame and calculate the centre of the face display window. As for the previous circuits, the face detection module processes the skin map values as they are produced by the skin filtering circuit and does not require the storage of the complete map.

Fig. 13 shows the organization of this module and the actions which are performed in each cycle of operation. In Cycle 0, the skin probability value produced by the skin filtering module is stored in a 25×1-bit shift register, which will be referred to as the "skin probability shift register". The module assumes that the skin map is delivered as a series of values and is not available as a complete image and, consequently, this register is required to store a continuously shifting part of that map in order to accommodate its spatial filtering, which is a neighbourhood operation. For the first 12 pixels of the first frame in a sequence of frames this storage is the only operation performed, since 13 skin values are required to complete the neighbourhood of the first one. In Cycle 1, the skin map pixel being processed is spatially filtered. This process relies on an adder tree to calculate the sum of the skin probabilities of the neighbourhood of the pixel being processed in a single cycle, which directly gives the number of neighbouring skin pixels, along with some logic to decide the skin or non-skin status of the pixel after filtering. The tag for the filtered pixel is also calculated during Cycle 1 and stored in a 13×3-bit shift register, which will be referred to as the "tag shift register". As discussed earlier, the determination of the tag for a skin pixel requires the examination of the tags of four of its neighbours, which is the reason that tags are stored in a shift register, i.e. so that the circuit maintains a continuously shifting sequence of the tag assignments in the image. If none of the neighbours is a skin pixel,



Skin Probability (SP)
Face Detection/Tracking
Store SP in Shift Register (Neighbourhood Formation)
SP Spatial Filtering
Tag Calculation (T_0-T_7)
New Tag-Group Association (if required)
m_{00}, m_{10}, m_{01} Statistics Updating
Tag-Group Associations Updating
m_{00}, m_{10}, m_{01} Statistics Grouping
x Centroid Calculation for Skin Groups (G_0-G_7)
y Centroid Calculation for Skin Groups (G_0-G_7)
Closest Face Identification
Largest Face Identification
Face Selection
Display Coordinates Calculation
Centre of window for display/ encoding/ transmission
All SP
Last SP in frame
Cycle 0
Cycle 1
Cycles 2-5
Cycles 6-13
Cycles 14-22
Cycles 22-30
Cycles 31-37
Cycle 38
Cycle 39

Fig. 13. Organization of the face detection/tracking module.

i.e. they are all assigned the non-skin tag T_7, a new tag is assigned to the skin pixel being processed. Otherwise, the skin pixel being processed takes the numerically lowest tag among the tags of the neighbourhood. The circuit also maintains an 8×3-bit register array, which will be referred to as the "tag-group association register array", which keeps a record of the skin group that each tag corresponds to. This array is initialized so that all eight tags correspond to group G_7, i.e. the non-skin group. When a new tag is assigned to a skin pixel, the appropriate tag-group association is also stored in this register array during Cycle 1, e.g. T_0 to G_0, T_1 to G_1 and so on.

In the following four cycles of operation, i.e. Cycles 2–5, the tag-group association array is updated for the tag of each of the four neighbourhood skin pixels of the previous cycle. The statistics m_{00}, m_{10} and m_{01} of (2) are also updated for each tag during Cycle 2. These values are stored in three register arrays, and more specifically, an 8×7-bit register array for m_{00} which will be referred to as the "m_{00} register array" and two 8×9-bit register arrays for m_{10} and m_{01}, respectively, which will be referred to as the "m_{10} register array" and the "m_{01} register array". The depth of the arrays has been chosen so that an overflow does not occur for the image size under consideration. As can be seen from (2), the calculation of m_{00}, m_{10} and m_{01} is trivial and effectively requires just one addition for each statistic and for each pixel processed.

The subsequent processing cycles are applicable only for the processing of the last pixel in a skin map. As can be seen from Fig. 13 the next step in the operation of the circuit is the grouping of the stored statistics over eight clock cycles, i.e. during Cycles 6–13. This simply requires the summation of the statistics of the tags which correspond to each group based on the information stored in the tag-group association register array. The implementation of this part of the circuit uses three adder trees and some simple additional logic for the calculation of the statistics of each of the eight groups in a single cycle, with the results stored back to the m_{00}, m_{10} and m_{01} register arrays.

The calculation of the x coordinate of the centroid of each skin group according to (3) is performed over nine cycles, i.e. during Cycles 14–22. A 9-bit $\times$ 7-bit two-stage pipelined integer divider is used for this, with the division for group G_0 spanning cycles 14 and 15, G_1 spanning cycles 15 and 16 and so on. The results are stored back into the m_{10} register array. The same process is performed during Cycles 22–30 for the calculation of the y coordinate of the centroid of each skin group, with the results stored back to the m_{01} register array. The next seven cycles, i.e. Cycles 31–37, are used to identify which of the seven skin regions represented by groups G_0–G_6 is spatially closest to the face of the previous frame as well as the largest among those skin regions. The identification of the closest face requires only some simple logic for the implementation of the distance equations of (4).

In Cycle 38, the face of the current frame is selected between the closest and the largest face based on the simple set of rules discussed in the algorithmic description section. If the circuit "run out" of skin tags during the processing of the frame or no skin pixels were found after filtering, the face of the previous frame is assumed to be the face of the current frame. Finally, the coordinates of the centre of the window for transmission and display are calculated during Cycle 39 as the average of the face centroids for the current and the previous seven frames, which are stored in two 8×4-bit shift registers, and are scaled to the dimensions of the original video frame. As discussed earlier, the module does not actually dictate what the dimensions of the tracking window should be, due to the fixed size of the mobile videophone device.

4.4. Circuit statistics

The entire circuit has been implemented almost exclusively using behavioural VHDL. The only exceptions are four memory modules, i.e. the 512×1-bit SMA_1 RAM, the 512×2-bit SMA_2 ROM, the 512×2-bit SMA_3 RAM and the 256×8-bit logarithmic transformation ROM, and the 9-bit $\times$ 7-bit integer divider, which have been implemented using VHDL parameterized library components. Table 1 shows the statistics for the implementation of this circuit on an ALTERA EP20K1000EBC652-1 device. These figures include 46 flip-flops for I/O synchronization. The total memory requirements of the system are less than 700 bytes. Assuming uninterrupted pixel byte delivery, a processing speed of $\sim$434 frames/s is obtained for a clock speed of $\sim$33 MHz. This is directly derived from the timing of the subsampling module, which is the bottleneck of the complete circuit, despite its pixel value per clock cycle processing time, i.e. the pixel delivery mechanism is the actual limitation of the system. It is, therefore, obvious that the circuit has the potential of achieving speeds far higher than real-time video processing requirements even at low clock speeds.

5. Discussion and conclusions

This paper considered the issue of face detection and tracking in the context of mobile videoconferencing. The advantages of deploying such a method in a larger mobile videoconferencing system are numerous and

Table 1
Design summary for the FPGA implementation of the proposed face detection and tracking system

Device	ALTERA EP20K1000EBC652-1
Total pins	35
Total logic elements	3173/38,400 (8.26%)
Total flip-flops	972
Total ESB bits	4608/327,680 (1.41%)
Total memory	697.5 bytes
CLK $f_{\max}$	33.13 MHz

include improved visual effect through face stabilization, reduction in the bandwidth requirements through the encoding and transmission of only the stabilized face rather than of the entire video frames, and the facilitation of smaller display sizes for the hand-held terminals which, in turn, facilitates smaller terminals. The advantage of the proposed face detection and tracking method is that, while robust to illumination variations and geometric changes such as scale and viewpoint changes, it entails a very low computational complexity. It comprises mainly integer additions and memory lookup operations, and minimal memory requirements, allowing a high-speed implementation on simple microprocessor architectures as well as a custom hardware implementation. The custom hardware realization of the proposed method was examined in terms of an FPGA-based implementation, which demonstrated its low cost in terms of logic and memory (less than 700 bytes) in conjunction with its frame processing capability at low clock speeds (over 400 frames/s at $\sim 33\,\text{MHz}$).

Although the frame processing capability of our custom hardware implementation far exceeds the requirements for a mobile videoconferencing application and, indeed, real-time video broadcasting processing requirements, it demonstrates the applicability of the algorithmic and hardware design philosophy to high-speed applications. Another possibility is the high-speed processing of multiple video frames by multiplexing multiple video sources into the system, through the replication of key components in the hardware system and the multiplexing of the majority of the existing hardware. Finally, based on the material presented here it is obvious that the proposed approach may also be applicable to other colour-based detection and tracking tasks.

Acknowledgements

The authors wish to acknowledge the contribution of Mr. James Cooper in the development of the face detection and tracking method.

References

[1] ITU-T Recommendation H.263 Version 2. Video coding for low bit rate communication, 1998.

[2] Yang Y, Zhu Q. Error control and concealment for video communications: a review, Proceedings of the IEEE, vol. 86(5), 1998. p. 974–97.

[3] Yang M, Kriegman DJ, Ahuja, N. Detecting faces in images: a survey. IEEE Transactions on Pattern Analysis and Machine Intelligence 2002;24(1): 34–58.

[4] Hjelmås E. Face detection: a survey. Computer Vision and Image Understanding. 2001;83(3):236–74.

[5] Turk M, Pentland A. Eigenfaces for recognition. Journal of Cognitive Neuroscience, 1991;3(1):71–86.

[6] Moghaddam B, Pentland A. Face recognition using view-based and modular eigenspaces. In: Automatic systems for the identification and inspection of humans, SPIE, vol. 2277, 1994.

[7] Pentland A, Moghaddam B, Starner T. View-based and modular eigenspaces for face recognition. In: Proceedings of IEEE International Conference on Computer Vision and Pattern Recognition CVPR'94, 1994. p. 84–91.

[8] Moghaddam B, Pentland A. Probabilistic visual learning for object detection. In: Proceedings of Fifth IEEE International Conference on Computer Vision ICCV'95, 1995. p. 786–93.

[9] Moghaddam B, Pentland A. Probabilistic visual learning for object representation. IEEE Transactions on Pattern Analysis and Machine Intelligence, 1997;19(7):696–710.

[10] Kervrann C, Davoine F, Pérez P, Forchheimer R, Labit C. Generalized likelihood ratio-based face detection and extraction of mouth features. Pattern Recognition Letters, 1997;18(9): 899–912.

[11] Yang MH, Ahuja N, Kriegman D. Face detection using a mixture of linear subspaces. In: Proceedings of Fourth IEEE International Conference on Automatic Face and Gesture Recognition FG'00, 2000. p. 70–6.

[12] Sung KK, Poggio T. Example-based learning for view-based human face detection. IEEE Transactions on Pattern Analysis and Machine Intelligence 1998;20(1):39–51.

[13] Rowley HA, Baluja S, Kanade T. Human face detection in visual scenes. In: Touretzky, TS, Mozer, MC, Hasselmo, ME. editors, Advances in neural information processing systems, MIT Press, vol. 8, 1996. p. 875–81.

[14] Rowley HA, Baluja S, Kanade T. Neural network-based face detection. In: Proceedings of IEEE International Conference on Computer Vision and Pattern Recognition CVPR'96, 1996. p. 203–8.

[15] Rowley HA, Baluja S, Kanade T. Neural network-based face detection. IEEE Transactions on Pattern Analysis and Machine Intelligence, 1998;20(1):23–38.

[16] Rowley HA, Baluja S, Kanade T. Rotation invariant neural network-based face detection. In: Proceedings of IEEE International Conference on Computer Vision and Pattern Recognition CVPR'98, 1998. p. 38–44.

[17] Osuna E, Freund R, Girosi F. Training support vector machines: an application to face detection. In: Proceedings of IEEE International Conference on Computer Vision and Pattern Recognition CVPR'97, 1997. p. 130–6.

[18] Li Y, Gong S, Sherrah J, Liddell H. Multi-view face detection using support vector machines and eigenspace modelling. In: Proceedings of International Conference on Knowledge-Based Intelligent Engineering Systems and Allied Technologies, 2000. p. 241–5.

[19] Samaria F, Young S. HMM-based architecture for face identification. Image and Vision Computing, 1994;12(8):537–43.

[20] Nefian AV, Hayes III, MH. Face detection and recognition using hidden Markov models. In: Proceedings of IEEE International Conference on Image Processing ICIP'98, vol. 1, 1998. p. 141–5.

[21] Nefian AV, Hayes III, MH. Hidden Markov models for face recognition. In: Proceedings of IEEE International Conference on Acoustics, Speech and Signal Processing ICASSP'98, 1998. p. 2721–4.

[22] Yang G, Huang TS. Human face detection in a complex background. Pattern Recognition 1994;27(1):53–63.

[23] Kotropoulos C, Pitas I. Rule-based face detection in frontal views. In: Proceedings of 22nd IEEE International Conference on Acoustics, Speech and Signal Processing ICASSP'97, vol. 4, 1997. p. 2537–40.

[24] Viola P, Jones MJ. Robust real-time object detection. Technical Report CRL 2001/01, Cambridge Research Laboratory, Compaq; 2001.

[25] Leung TK, Burl MC, Perona P. Finding faces in cluttered scenes using random labelled graph matching. In: Proceedings of Fifth IEEE International Conference on Computer Vision ICCV'95, 1995. p. 637–44.

[26] Burl MC, Leung TK, Perona P. Face localization via shape statistics. In: Proceedings of First IEEE International Conference on Automatic Face and Gesture Recognition FG'95, 1995. p. 154–9.

[27] Leung TK, Burl MC, Perona P. Probabilistic affine invariants for recognition. In: Proceedings of IEEE International Conference on Computer Vision and Pattern Recognition CVPR'98, 1998. p. 678–84.

[28] Yow KC, Cipolla R. A probabilistic framework for perceptual grouping of features for human face detection. In: Proceedings of Second IEEE International Conference of Face and Gesture Recognition FG'96, 1996. p. 16–21.

[29] Yow KC, Cipolla R. Feature-based human face detection. Image and Vision Computing 1997;15(9):713–35.

[30] Yow KC, Cipolla R. Enhancing human face detection using motion and active contours. In: Proceedings of Third Asian Conference on Computer Vision, 1998. p. 515–22.

[31] Han CC, Liao HYM, Yu KC, Chen LH. Fast face detection via morphology-based pre-processing. In: Proceedings of Ninth IEEE International Conference on Image Analysis and Processing ICIAP'98, 1998. p. 469–76.

[32] Terillon JC, Akamatsu S. Comparative performance of different chrominance spaces for color segmentation and detection of human faces in complex scene images. In: Proceedings of Vision Interface VI'99, 1999. p. 180–7.

[33] Terillon JC, David M, Akamatsu S. Automatic detection of human faces in natural scene images by use of a skin color model and of invariant moments. In: Proceedings of Third IEEE International Conference on Automatic Face and Gesture Recognition FG'98, 1998. p. 112–7.

[34] Sobottka K, Pitas I. Face localization and facial feature extraction based on shape and color information. In: Proceedings of IEEE International Conference on Image Processing ICIP'96, 1996. p. 483–6.

[35] Yang MH, Ahuja N. Detecting human faces in color images. In: Proceedings of IEEE International Conference on Image Processing ICIP'98, vol. 1, 1998. p. 127–30.

[36] Yoo TW, Oh IS. A fast algorithm for tracking human faces based on chromatic histograms. Pattern Recognition Letters, 1999;20(10):967–78.

[37] Jones MJ, Rehg JM. Statistical color models with application to skin detection. In: Proceedings of IEEE International Conference on Computer Vision and Pattern Recognition CVPR'99, 1999. p. 274–80.

[38] Yang J, Waibel A. A real-time face tracker. In: Proceedings of Third IEEE Workshop on Applications of Computer Vision, 1996. p. 142–7.

[39] Soriano M, Martinkauppi B, Huovinen S, Laaksonen M. Skin detection in video under changing illumination conditions. In: Proceedings of 15th International Conference on Pattern Recognition ICPR'2000, vol. 1, 2000. p. 839–42.

[40] Fleck M, Forsyth D, Bregler C. Finding naked people. In: Proceedings of European Conference on Computer Vision ECCV'96, vol. 2, 1996. p. 593–602.

[41] Prokop RJ, Reeves AP. A survey of moment-based techniques for unoccluded object representation and recognition. Computer Vision, Graphics and Image Processing: Graphical Models and Image Processing 1992;54(5):438–60.