P-FAD: Real-Time Face Detection Scheme on Embedded Smart Cameras

Qiang Wang, *Student Member, IEEE*, Jing Wu, *Member, IEEE*, Chengnian Long, *Member, IEEE*, and Bo Li, *Fellow, IEEE*

***Abstract*—Face detection on general embedded devices is fundamentally different from the conventional approach on personal computer or consumer digital camera due to the limited computation and power capacity. The resource-limited characteristic gives rise to new challenges for implementing a real-time video surveillance system with smart cameras. In this work, we present the design and implementation of Pyramid-like FAce Detection (P-FAD), a real-time face detection system constructed on general embedded devices. Motivated by the observation that the computation overhead increases proportionally to its pixel manipulation, P-FAD proposes a hierarchical approach to shift the complex computation to the promising regions. More specifically, P-FAD present a three-stage *coarse, shift, and refine* procedure, to construct a pyramid-like detection framework for reducing the computation overhead significantly. This framework also strikes a balance between the detection speed and accuracy. We have implemented P-FAD on notebook, Android phone and our embedded smart camera platform. An extensive system evaluation in terms of detailed experimental and simulation results is provided. Our empirical evaluation shows that P-FAD outperforms V-J detector calibrated color detector (VJ-CD) and color detector followed by a V-J detector (CD-VJ), the state of the art real-time face detection techniques by 4.7×–8.6× on notebook and by up to 8.2× on smart phone in terms of the detection speed.**

***Index Terms*—Embedded smart camera, face detection, pyramid-like scheme.**

I. Introduction

TECHNOLOGICAL developments in imaging, computation and embedded systems have made cameras no longer merely devices to record a particular scene in a 2D image, but also integrate an on-board processor, memory and communication interface on a single embedded device to make themselves smart. As a consequence, numerous applications have been proposed and investigated on embedded smart cameras, such as intelligent surveillance [3], human computer interface [8], person authentication [22], human activity analysis [24], and so on. Among these applications, which are usually involved with humans, face detection has played a fundamental and first-step role.

Face detection is, given an arbitrary image, to determine whether or not there are any faces in the image and, if present, return the image location and extent of each face [34]. Face detection has been one of the most studied topics in the computer vision literature [34], [35]. But most of them were designed for PC platform or implemented via dedicated hardware/coprocessor in some embedded devices such as consumer digital cameras [20]. Embedded smart cameras, such as camera nodes [4] in the visual sensor network, or the mobile devices (cell phones, tablets) with built-in cameras, usually adopt general-purpose processor for fulfilling its varying application task. General purposed embedded smart cameras are usually multi-task operating instead of single function. For example, cell phone is no longer a simple device to make a call, but a mobile computer on which we can listen music, browse web, download files simultaneously. Moreover, embedded smart cameras are resource-restricted (e.g., computation, memory, energy) comparing to the personal computer. Therefore, multi-task and restricted resource on general purpose processor without hardware acceleration pose great challenge in designing fast and efficient face detection algorithms.

In this paper, we investigate the real-time face detection on resource-limited embedded smart camera with general-purposed processor. Our work exploits on the observation that computation and storage overhead increase proportionally to its pixel manipulation in image processing. A natural way is to construct a hierarchical scheme: identifying face candidates with little computation through manipulating on full image and then eliciting the true faces from candidates with reliable algorithm. The key challenge in the hierarchical scheme is *how to construct a multi-layer architecture, in which the complex processing can be split from the pixel manipulation and guarantee detection accuracy simultaneously?*

We answer this problem by proposing Pyramid-like FAce Detection (P-FAD), which is inspired from the cascade structure that reserves more complex process on promising regions. P-FAD consists of five layers whose operating units decrease dramatically from top to down while the operations on every unit increase gradually. P-FAD is mainly using a three-stage

Manuscript received January 31, 2013; accepted March 17, 2013. Date of publication April 26, 2013; date of current version June 07, 2013. The research was supported in part by the Natural Science Foundation of China (NSFC) under Grant 61172064, 61104091, in part by the Program for New Century Excellent Talents in University under Grant NCET-11-0326, in part by the Specialized Research Fund for Doctoral Program of Higher Education under Grant 20100073120061, in part by the SJTU Science and Technology Innovation Funding under Grant AE0300006. The work of B. Li was supported in part by NSFC/RGC under Grant N-HKUST610/11, in part by HKUST under Grant RPC11EG29, SRFI11EG17-C and Grant SBI09/10.EG01-C, in part by Huawei Technologies Co. Ltd. under Grant HUAW18-15L0181011/PN, in part by Guangdong Bureau of Science and Technology under Grant GDST11EG06, and in part by China Cache Int. Corp. under Grant CCNT12EG01. This work was presented in part at the 6th ACM/IEEE International Conference on Distributed Smart Cameras, November, 2012. This paper was recommended by Guest Editor H. Aghajan.

Q. Wang, J. Wu, and C. Long are with the School of Electronic Information and Electrical Engineering, Shanghai Jiao Tong University, Shanghai 200240, China (e-mail: longcn@sjtu.edu.cn).

B. Li is with the Department of Computer Science and Engineering, The Hong Kong University of Science and Technology, Hong Kong (e-mail: bli@cse.ust.hk).

Color versions of one or more of the figures in this paper are available online at http://ieeexplore.ieee.org.

Digital Object Identifier 10.1109/JETCAS.2013.2256823

coarse, shift, and refine procedure to spilt the complex process. P-FAD first imposes a coarse operation on every pixel for skin detection. It is extremely efficient without losing the robustness to the changing environment. P-FAD then makes a shift among operating units: using scheme's layer 2–4 to shift the operations from pixel manipulation to the process of contour points, grouped regions and face candidates, respectively. Compared with the other kinds of hierarchical face detection scheme [5], [19], our shift part efficiently reduces the scale of process, results in reducing computation overhead significantly. Finally, P-FAD presents a modified Viola–Jones detector, to refine our final results.

We have used some self-shot scenarios as well as MOBIO database [15] to evaluate P-FAD separately and systematically. We also have implemented our scheme on notebook, Android cell phone and low-cost embedded smart camera. Experimental results demonstrate the P-FAD's resource-aware property: it just needs fewer than 1/30 run-time of the classical face detector or fewer than 1/7 run-time of the real-time optimized detector; on Android phone, P-FAD just consumes 1/40 energy of the Android API for face detection. Besides its resource-aware property, P-FAD still holds the acceptable detection accuracy comparing to the classical detectors, and even outperforms the real-time optimized detector in terms of the accuracy. Moreover, P-FAD is not customized/optimized for any hardware platform, so its resource-aware property can also be ported to other general-purposed smart camera platforms.

The remainder of this paper is organized as follows. In Section II, we review some related works and the difference between them with our work. Section III firstly states our five-layer pyramid-like scheme, and then we detail our three-stage strategy through three subsections. We have conducted extensive evaluations of P-FAD in Sections IV and V is our conclusion.

II. Related Work

In this section, the literatures of face detection are briefly reviewed, followed by a discussion of related works about resource-aware and real-time face detection methods.

Although there exist many ways to detect face [34], the most common used approaches can be simply classified as feature-based (e.g., skin color [13]) approach, classifier-based approach [35] and hybrid approach [5], [19]. Traditionally, researchers used the first two approaches separately and paid more attention on the detection accuracy while the detection speed has been omitted in some degree. For example, Rowley *et al.* [23] have presented a neural network-based upright frontal face detection system. It takes approximately 383 s to process a 320×240 image on a 200 MHz R4400 SGI Indigo 2. The faster version they have reported in their paper still needs 2–4 s. Hsu *et al.* [10] proposed a face detection method in color image that needs 23 s to process a 640×480 image on a 1.7 GHz CPU. The so-called most successful and fastest Viola–Jones [28] detector[1] can process 384×288 images at the speed of 15 frames per second (FPS) on a conventional desktop and 2 FPS on a low power 200 MIPS *Strong ARM* processors. Nevertheless, their image size is too small to be preferable (a 640×480 resolution is common used) and the 2 FPS on *Strong ARM* is sometimes unacceptable in a real-time application.

[1]In the rest of the paper, we will use "V-J detector" to represent "Viola–Jones detector."

To obtain the real-time performance in video streams, especially to overcome the restricted resource of embedded systems such as mobile device, several optimized face detectors appeared. In [6], [7], [9], and [26], V-J detector was implemented on the specify-designed hardware to achieve the high FPS. However, this kind of hardware optimization is not suitable for general-purposed smart cameras. On the other hand, researchers seek to combine the classifier-based approach (such as Viola–Jones detector) with the color-based approach to speed up their detector. [19] and [21] proposed a scheme to calibrate the light source of their color-based detector through a software-optimized V-J detector. Thanks to the high efficiency of color information, it results in a real-time (nearly 20 FPS) detection of face in the steady state. In [5] and [32], skin regions are obtained after skin-color detection, then a classifier-based detector are equipped to detect faces from given skin regions.

Our P-FAD is also a hybrid scheme: both use the skin color information and the modified V-J detector to detect a face. The major differences of our work are as follows.

- Pyramid-like cascade structure is proposed to build the layers of hierarchy scheme. Noting that the related works on hierarchy scheme operate on full image in all layers [19], [32]. Our proposed scheme reserves the complex process on the promising regions, which results in a huge reduction of computation overhead.
- A shift framework consists of AIMD-based contour dection, dynamic group and region merge&filter, is proposed to shift the operation units. This framework generates the promising regions (face candidates for V-J detector) from the result of skin detection. Traditional method usually makes the similar shift through some image process such as morphological filter followed by connect component [19], [25]. This kind of image process consistently operate on a binary image. On the contrast, our shift framework just deal with contours and limited regions, that makes it extremely fast and takes even less run-time than a typically image subtraction process.
- The improved/modified skin-color detector and V-J detector are presented to adapt to our scheme. Many researchers has used Gaussian mixture model (GMM) for the skin color [13]. In order to adapt this model under the dynamic environment, a reliable ground truth should be obtained to train the model online [16]. In our adaptive skin color detector, this reliable ground truth is obtained from the former output of V-J detector. We also simplify the implementation of skin detector to improve the efficiency. Moreover, we construct the time consumption model of V-J detector's cascade structure, then modify it to a proper implementation in our P-FAD.

III. Pyramid-Like Face Detection Scheme

In this section, we first introduce the hierarchical framework for face detection on embedded smart camera briefly. Then we focus on tackling the challenging issues in constructing the hierarchical scheme.

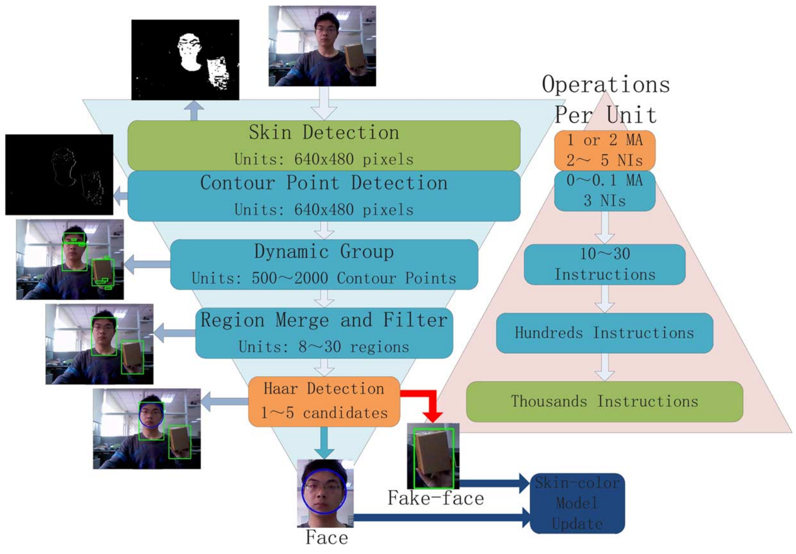


Fig. 1. Pyramid-like architecture. MA is the abbreviation of memory access operation, NI means normal instruction. As we know, memory access usually costs much more machine cycles than normal instruction.

P-FAD is a hierarchical detection scheme, as shown in Fig. 1. More specifically, P-FAD consists of five layers: skin detection, contour point detection, dynamic group, region merge & filter and modified V-J detector. P-FAD first detects the skin by a relatively coarse detection method; then through contour point detection, dynamic group, region merge & filter, P-FAD shifts operating unit from pixels to contour points, regions and face candidates, respectively; the finally results are refined by the modified V-J detector. We argue that the hierarchical detection scheme is tailored to implementing real-time detection scheme with low computation and storage overhead, in which operating units decrease dramatically from top to down while the operations on each unit are increasing. It could make pixel manipulation as few as possible to make a significant reduction in run-time and guarantee the detection accuracy through further complex process. Thus, P-FAD has an inverted pyramid-like appearance on the scales of every layer's operate units while the process's complexity is increasing with a pyramid-like shape. To achieve low overhead and high detection accuracy, the following critical issues should be answered in P-FAD: How to derive the efficiency detection regions with operating on full images as few as possible? How to achieve a robust detector with high accuracy by considering the changing environment such as illumination and different individuals? In the following, we detail the design of five layers in P-FAD.

A. Coarse Process: Skin Detection

Skin detection is the first layer with pixel manipulation in P-FAD. Because the pixel-level manipulation accounts for the most processing time in image process, a crucial issue in skin detection is to consider the process complexity. To reduce processing time significantly, the basic design principle is to present a relatively coarse but highly time-saving skin detection.

In P-FAD, skin detection is based on skin-color information because skin-color provides computationally effective yet, robust information against rotations, scaling and partial occlusions [13]. We model the skin color in CbCr subset of YCbCr color space. CbCr subset can eliminate the luminance effect and provide nearly best performance among different color spaces [27]. To classify a pixel as a skin-pixel or none-skin-pixel, we choose the widely used *Gaussian mixture models (GMM)*. It has relatively simplified parameters without losing accuracy, to represent the skin-color distribution through its probability distribution function (PDF) in CbCr subspace, defined as

$$p(x(t)) = \sum_{i=1}^{N} \omega_{i,t}\eta_i(x(t), \mu_{i,t}, \Sigma_{i,t}) \tag{1}$$

$$\eta_i(x(t), \mu_{i,t}, \Sigma_{i,t}) = \frac{e^{-\frac{1}{2}(X_t-\mu_{i,t})^T \Sigma_{i,t}^{-1}(X_t-\mu_{i,t})}}{2\pi|\Sigma_{i,t}|^{\frac{1}{2}}} \tag{2}$$

where t denotes the frame index, $x(t)$ is a two-dimension color vector in CbCr subspace, $\eta_i(x(t), \mu_{i,t}, \Sigma_{i,t})$ is the ith *single Gaussian model (SGM)* component contributing to mixed model with a weight ω_i. The SGM is a elliptical (two-dimension) Gaussian joint probability distribution function, determined by its mean vector $\mu_{i,t}$ and the covariance matrix $\Sigma_{i,t}$. At last, the pixel with the color vector $x(t)$ can be judged as a skin-color pixel or not through comparing the $p(x(t))$ with a predefined threshold.

The main difficulties to implement the GMM in P-FAD are the following.

- The fixed Gaussians' parameters $(\mu_{i,t}, \Sigma_{i,t})$ obtained by offline training from a large face data-set is not robust to the changing environment (illumination, different individuals), which is shown in Fig. 2. Moreover, some correction and compensation methods [10] are unfeasible on embedded platform for its computation overhead.
- The computation overhead is high in (1) on every pixels for judging a pixel as a skin color pixel or not.

To solve the above problems, we propose an adaptive GMM skin color detection algorithm with online learning and simplify judging criteria. The pseudo-code is given in Algorithm 1. First, the sample sets $(S_{\text{skin}}, S_{\text{fake}})$ derived from the final output of P-FAD is exploit to train the adaptive GMM online. The training

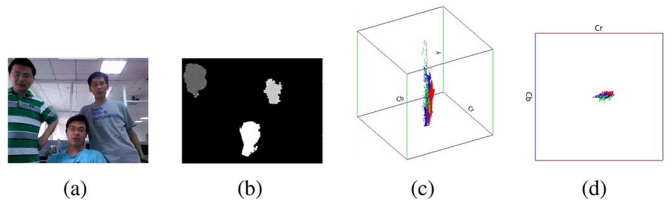


Fig. 2. Different persons' skin color cluster in color space: in (c) and (d), green, blue, and red dots represent the right, middle and right person respectively. Notice that although the whole skin color are clustered in a small region in CbCr subspace, different person's cluster is varying. Especially, green dots are divided into two clusters in CbCr subspace because of the illumination influence on left person: the left part of his face is brighter than the right part, which results in the higher Y values in YCbCr space. (a) original picture, (b) face mask for three persons, (c) skin color in YCbCr space, (d) skin color in CbCr subspace.

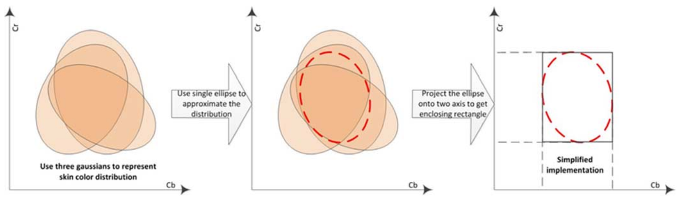

Cr
Use three gaussians to represent skin color distribution
Cb
Use single ellipse to approximate the distribution
Cr
Cb
Project the ellipse onto two axis to get enclosing rectangle
Cr
Simplified implementation
Cb

Fig. 3. Simplified implementation of GMM for skin detection.

speed $\alpha(t)$ has different sign symbol for the two sets S_{skin} and S_{fake}, which denotes the learning and forgetting the two sets' current distribution information respectively (lines 5–7). Then the parameters $(\mu_{i,t}, \Sigma_{i,t}, \omega_{i,t})$ of each SGM are updated based on current learning parameter $\alpha(t)$ (lines 9–15). Note the threshold $\sigma = 2.5|\Sigma_{i,t-1}|$ means the confidence interval with the 95% probability to confirm the fact that current skin color distribution belong to the given SGM (line 10). Moreover, we should add/remove the SGM to/from GMM if there is no SGM can approximately represent the current skin-color distribution (lines 17–22).

To the end, we judge a pixel as a skin color pixel or not based on our simplified rectangle judgment. The basic idea of simplification is to approximate the equal-value boundary of $p(x(t))$ by a single ellipse and project this ellipse onto two axis to get its minimum enclosing rectangle. This simplification is illustrated in Fig. 3. First, the mean values of GMM's parameter $\mu_{i,t}$ and $\Sigma_{i,t}$, denoting by $\tilde{\mu}(t)$ and $\tilde{\Sigma}(t)$, are to approximate ellipse's position and shape. It can be computed according to the following (3). Then, we decompose the covariance matrix $\tilde{\Sigma}(t)$ of ellipse to get the ellipse's rotation angle θ_C and the length of axis $\sqrt{(\sigma/d_{11})}$, $\sqrt{(\sigma/d_{22})}$. The ellipse's minimum enclosing rectangle's width and height, denote as $W_{cr}(t)$ and $W_{cb}(t)$ respectively, can be deprived by (4). Obviously, the center position of rectangle is the same with approximate ellipse's.

$$\tilde{\mu}(t) = \frac{1}{N}\sum_{i=1}^{N}\mu_{i,t}, \tilde{\Sigma}(t) = \frac{1}{N}\sum_{i=1}^{N}\Sigma_{i,t} = C^T DC \tag{3}$$

$$W_{cr}(t) = 2\sqrt{\frac{\sigma}{d_{11}}}\cos(\theta_C), W_{cb}(t) = 2\sqrt{\frac{\sigma}{d_{22}}}\cos(\theta_C). \tag{4}$$

Herein D is a diagonal matrix and d_{11}, d_{22} represent the elements of matrix D. Based on threshold $\sigma = 2.5|\Sigma_{i,t-1}|$, we get the length of approximate ellipse's axis: $\sqrt{(\sigma/d_{11})}$, $\sqrt{(\sigma/d_{22})}$.

Algorithm 1: Adaptive GMM algorithm

```
1: $S_{skin}$, $S_{fake}$: skin and fake-skin color pixels' set
2: $\mu(t)$: mean vector of $S_{skin}$ or $S_{fake}$
3: $\Sigma(t)$: covariance matrix of $S_{skin}$ or $S_{fake}$
4: The learning speed is $\alpha(t)$
5: **for** $S_{skin}$ and $S_{fake}$ **do**
6:     Calculate:
       $\mu(t) = \frac{1}{n}\sum_{k=1}^{n} x_k(t)$
       $\Sigma(t) = \frac{1}{n-1}\sum_{k=1}^{n}(x_k(t) - \mu_(t))(x_k(t) - \mu_(t))^T$
       Where $x_k(t) \in S_{skin}$ or $x_k(t) \in S_{fake}$
7:     $\alpha(t) = \begin{cases} |\alpha(t)| & S_{skin} \\ -|\alpha(t)| & S_{fake} \end{cases}$
8:     *//update the parameters of every SGM*
9:     **for** every SGM $\eta_i(x(t), \mu_{i,t}, \Sigma_{i,t})$ in GMM **do**
10:        **if** $(\mu(t) - \mu_{i,t-1})^T \mu(t) - \mu_{i,t-1}) \leq 2.5|\Sigma_{i,t-1}|$
           **then**
11:            $\mu_{i,t} = (1 - \alpha(t))\mu_{i,t-1} + \alpha(t)\mu(t)$
               $\Sigma_{i,t} = (1 - \alpha(t))\Sigma_{i,t-1} + \alpha(t)\Sigma(t)$
               $\omega_{i,t} = (1 - \alpha(t))\omega_{i,t-1} + \alpha(t)$
12:        **else**
13:            $\mu_{i,t} = \mu_{i,t-1}$, $\Sigma_{i,t} = \Sigma_{i,t-1}$
               $\omega_{i,t} = (1 - \alpha(t))\omega_{i,t-1}$
14:        **end if**
15:    **end for**
16:    *//add/remove a SGM to/from GMM*
17:    **if** There is no SGM satisfied in $S_{skin}$
       $(\mu(t) - \mu_{i,t-1})^T(\mu(t) - \mu_{i,t-1}) \leq 2.5|\Sigma_{i,t-1}|$ **then**
18:        **if** The total number of SGM in GMM reached
           Maximum **then**
19:            remove a SGM from GMM whose covariance
               matrix is largest
20:        **end if**
21:        add a new SGM whose mean vector is $\mu(t)$ and
           covariance matrix is maximum
22:    **end if**
23: **end for**
24: *//rectangle realization*
25: according to Eq. (3)-(4), calculate the rectangle's
    parameters: $\tilde{\mu}(t), W_{cr}(t), W_{cb}(t)$
26: classify the pixel as skin or not through judging wether
    the pixel fall in the rectangle
```

The rotation angle of ellipse can be easily computed from orthogonal matrix C.

B. *Shift Process: Contour Points Detection, Dynamic Group and Region Merge & Filter*

After a coarse yet highly efficient skin detection, the next question in P-FAD is how to shift the operations from pixel manipulation to face candidates confirmation. Usually and intuitively, after we get a binary image (which usually called face mask or foreground mask) from skin detection, many binary image process will follow to form a so-called region of interest (ROI), such as mask correction (dilation, erosion, open/close operation), component labeling, region group and so on [13]. The ROIs could be the face candidates. Notice that all these operations are pixel-level and computation overhead increases proportionally to its pixel manipulation in image processing.

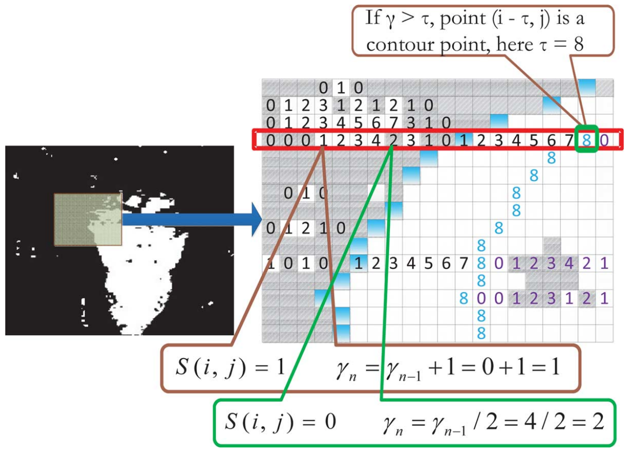

If γ > τ, point (i - τ, j) is a contour point, here τ = 8
$S(i, j) = 1$ $\gamma_n = \gamma_{n-1} + 1 = 0 + 1 = 1$
$S(i, j) = 0$ $\gamma_n = \gamma_{n-1} / 2 = 4 / 2 = 2$

Fig. 4. The value of pixels contour determine index.

To tackle this problem, we propose a three-layer architecture, which locates in P-FAD's layer 2–4, to efficiently shift the operations units from pixel to contour points, regions and face candidates, sequentially. The operations on each unit increase from several normal instructions (NIs), tens NIs to hundreds NIs. This three-level shift strategy results in a minimum pixel-level operation so as to bring huge computation saving.

1) AIMD Based Contour Points Detection: As we know, the contour of a target is the dividing line of foreground and background. When an image is scanned pixel by pixel, the point in the contour is the starting point of an adequately connected component. So we can build an index and compare it to a threshold τ to determine at which pixel the condition of adequately connected has been met. Usually, the value of τ is corresponding to the resolution of the image and the size of the target (judged by the relations between the face size and camera's configuration). Moreover, a weighted accumulation γ, is defined as the pixels contour determining index. At pixel location (i, j), γ is calculated as

$$\gamma_n = S(i,j)(\gamma_{n-1}+1) + (1 - S(i,j))(\gamma_{n-1}/2) \qquad (5)$$

where $S(i,j)$ yield to 0 or 1 for every pixel after the skin detection in P-FAD's first layer. The subscript of γ means the accumulation process in scanning.

This index is formed by the inspiration of "additive increase, multiplicative decrease" (AIMD) method in TCP congestion control [11]. Then, we can calculate every pixel's contour determining index pixel by pixel while scanning the binary image. The order of scanning is from left to right, and from top to bottom. For example, Fig. 4 illustrates a binary image, where the contour points of the target are painted blue. We select a section of the image and zoom it to pixel level. Every pixel point's contour determining index can be calculated one by one using (5). The current pixel's index is depending on the last scanned one's Additive increasing (plus one) or Multiplicative Decrease (divided by 2). Finally, we set the value of τ to judge whether or not the current scanned pixel is belong to the contour. Through this method, we can get the face' contour points while effectively discard the noisy pixels which are introduced by illumination changes.

Algorithm 2: AIMD-based contour detecting algorithm

```
1: The pixel's position is $i, j$
2: The threshold of determining the contour is $\tau$
3: The contour's direction $C_{dir} = \{LEFT, RIGHT\}$
4: The points of left contour are saved in the array $CL$
5: The points of right contour are saved in the array $CR$
6: for $i \le FrameHeight$ do
7:     $\gamma = 0$
8:     for $j \le FrameWidth$ do
9:         if $B(i,j) == 1$ then
10:            if $C_{dir} == LEFT$ then
11:                $\gamma++$
12:            end if
13:            if $C_{dir} == RIGHT$ then
14:                $\gamma = \gamma/2$
15:            end if
16:        end if
17:        if $S(i,j) = 0$ then
18:            if $C_{dir} == RIGHT$ then
19:                $\gamma++$
20:            end if
21:            if $C_{dir} == LEFT$ then
22:                $\gamma = \gamma/2$
23:            end if
24:        end if
25:        if $\gamma > \tau$ then
26:            if $C_{dir} == LEFT$ then
27:                save current $(i,j)$ to $CL$
28:                $C_{dir} = RIGHT$
29:            end if
30:            if $C_{dir} == RIGHT$ then
31:                save current $(i,j)$ to $CR$
32:                $C_{dir} = LEFT$
33:            end if
34:        end if
35:    end for
36: end for
37: The contour points have been saved in $CL$ and $CR$
```

The details of the AIMD-based contour detection method are explained by the pseudo-code provided in Algorithm 2. Because our scan direction is from left to right in one line, there muse be two kinds of contour points: the contour point divide the foreground in its right-side and leave the background in its left-side, as well as the reversed situation. So we notate the first situation as left contour points and the other one are right contour points. C_{dir} is denoted as the type of contour we are detecting. CL and CR are the storage array of left contour and right contour.

2) Dynamic Group Method for Region Formation: After the contour detection, the contour points are stored by its spatially scanned location in the image. For multiple targets, we must classify their contour points into different groups according to their corresponding targets.

For all the contour points, the points that can form a closed area must belong to a target's contour, otherwise they are discarded as noise. Based on this feature, we can create the group dynamically during the scanning of all contour points, and get

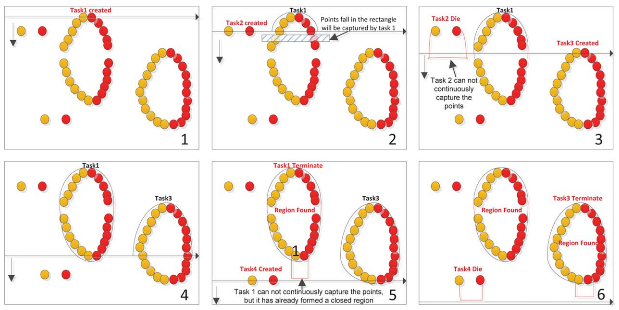

Task1 created
1
Task2 created
Task1
Points fall in the rectangle will be captured by task 1
2
Task2 Die
Task1
Task3 Created
Task 2 can not continuously capture the points
3
Task1
Task3
4
Task1 Terminate
Region Found
Task3
Task4 Created
Task 1 can not continuously capture the points, but it has already formed a closed region
5
Region Found
Task3 Terminate
Region Found
Task4 Die
6

Fig. 5. Procedure of dynamic group.

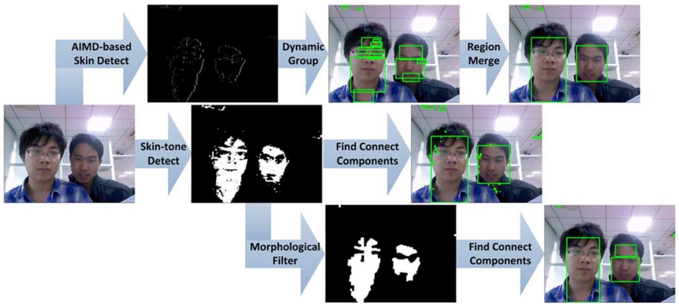


Fig. 6. Flow chart of P-FAD's shift procedure and traditional way to produce face candidates.

all the groups after the scanning is over. Herein, we propose Dynamic Group method to carry through the classification. During the scanning process.

- **Group creating** means that if the point is not captured by any existing groups, a new group will be created.
- **Group terminate** means the points in a given Group have formed a closed area.
- **Group dead** means the points in the group have not formed a closed area and the group has not captured new points during a period of time. For example, if the group can't capture new point for five times continuously, it will die.

Fig. 5 illustrates the above-mentioned phases. When the scanning is over, the proposed Dynamic Group method will form many contour points' groups. Each of them corresponds to a detected target.

3) Region Merge and Filter: At last, we present region merge & filter to produce face candidates. Specifically, we merge the adjacent small regions to a complete face candidates, which is usually split for the interference of eyebrows, glasses and so on. We also filter out the nonface regions through some priori knowledge, such as height-width ratio ranging from 1.1 to 1.5 in our scheme.

Fig. 6 shows the flow chat of AIMD based contour point detection, dynamic group and region merge & filter procedure. It also shows the conventional way to form a face candidates [32], which is usually based on the process of binary image, such as component finding [25], morphological filter, and so on. The detailed comparison can be seen in Section IV-A2.

C. Refine Process: Modified V-J Detector

The above four layers in P-FAD may produce several face candidates in a frame. To verify the final output, we choose V-J detector [28] as the final layer. In a typical configuration, a VGA image would produce nearly 881484 sub-windows [1] to be classified as face or not in V-J detector. However, the sub-windows can be reduced to hundreds by exploiting the above four layers in P-FAD. Thus, the computation overhead of Viola–Jones' detector in P-FAD is reduced significantly. Moreover, we argue that the fully cascade structure in V-J detector is not almost required because P-FAD can present an early rejection in above four layers. It is time-consuming when a sub-window goes through the whole cascade. Suppose there are n stages in a cascade, we define $\vec{t} = [t(1), \ldots t(x), \ldots t(n)]^T$, in which $t(x)$ denotes the time consumption if the start stage is x. So the cascade's time consumption of different start stage can be modeled as follows:

$$\vec{t}_{\text{total}} = N_{\text{reject}}\vec{t}_{\text{reject}} + N_{\text{accept}}\vec{t}_{\text{accept}} \tag{6}$$

where N_{reject} and N_{accept} are the number of rejected and accepted sub-windows in the whole cascade structure respectively. $\vec{t}_{\text{reject}}$ and $\vec{t}_{\text{accept}}$ are the expectation time consumption to reject and accept a sub-window respectively. They can be derived as following:

$$\vec{t}_{\text{reject}} = k\Theta\{\mathbf{P}_{n\times n}\mathbf{A}^T_{n\times n}\}, \vec{t}_{\text{accept}} = k[a_{1n}, \ldots a_{nn}]^T \tag{7}$$

$$p_{ij} = \begin{cases} P(j)\prod_{k=i}^{j-1}(1-P(k)), & i \le j \\ 0, & \text{otherwise} \end{cases} \tag{8}$$

$$a_{ij} = \begin{cases} \sum_{k=i}^{j} F(k), & i \le j \\ 0, & \text{otherwise} \end{cases} \tag{9}$$

where p_{ij} in Matrix $\mathbf{P}$ denotes the probability of the sub-window starting from the ith stage would be rejected in stage j, a_{ij} in matrix $\mathbf{A}$ is the sum of features from stage i to j. Assume that the time consumption is proportional (k times) to the number of processed features. The $\Theta(\bullet)$ maps the matrix to a vector whose elements are the elements in matrix's dialog. Thus, the detector's time consumption is determined by two set of arguments: $\vec{F} = [F(1), \ldots F(x), \ldots F(n)]^T, F(x)$ is the number of features in stage x; and $\vec{P} = [P(1) \ldots P(x) \ldots P(n)]^T, P(x)$ is the probability to reject the non-face sub-windows in stage x. From the OpenCV 2.3 baseline face detector, we can get $\vec{F}$, the cascade detector's feature quantity distribution in each stage. Details can be seen in [17]. We also assume $P(x)$ is linear increasing from 50% to 99% which is obedient to the cascade structure [28]. We simulate some practically scanning conditions using above formula and also get an implementation on a 2.2-GHz notebook. Simulation and experimental results in Fig. 7 show that the time cost function is convex for the start stage. Then the minimum value of start stage can be only obtained at the first and last stage.

In P-FAD, according to Fig. 7, the optimal start stage is depend on the ratio $\alpha = N_{\text{reject}}/N_{\text{accept}}$. We define the overhead rate $\gamma = (t_{\text{total}}(1) - t_{\text{total}}(n))/(t_{\text{total}}(1) + t_{\text{total}}(n))$ as the effect of choosing first stage to start, while negative value means the choice could save time. Fig. 8 shows that when the α is extremely large, which meets the traditional Viola–Jones detector situation, the γ is near to -1 indicates choosing first

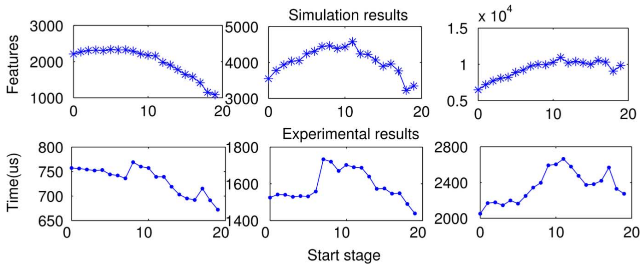


Fig. 7. Viola–Jones detector's time cost of different start stage in three scenarios. First row is the simulation result and second row is the experimental test. In the first column there are eight rejected sub-windows and two accepted sub-windows in the scan procedure, second column the number is (28, 3) and (91, 5) in the third column.

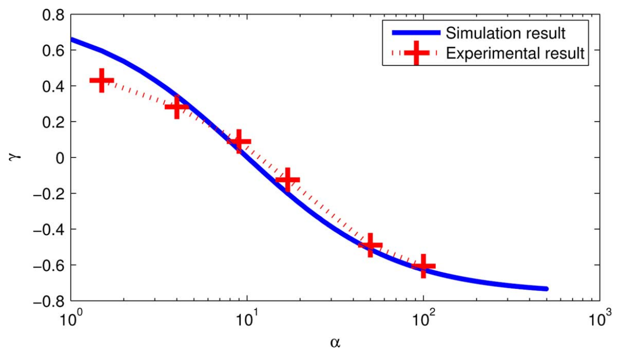


Fig. 8. The overhead of time consumption when choose the first stage to start. The overhead ratio γ is means the time saving or overhead between two start stage choice comparing to the sum of two choice's consumption.

stage to start is undoubtedly saving most processing time. However, when α is smaller than 12, the γ will be positive. Thus, in P-FAD, when the number of nonface sub-windows is few, choosing the last stage is optimal for time-consuming.

Based on the above discussion, we may implement a modified Viola–Jones detector in P-FAD according to the online estimation of α. Note that the total number of sub-windows is determined by its scan strategy before the classifying, so we only need to estimate the N_{accept}. In our implementation, we just assume N_{accept} approximately equals to the number of face candidates, which is reasonable when checking the statistical data in Table IV. Obviously, further work could be done to improve the estimation's accuracy.

D. *The Algorithmic Complexity of P-FAD*

In this section, we briefly conclude our scheme's computation complexity. First of all, we must note that the scheme's whole computation complexity is mainly determined by the P-FAD's first and second layer while last three layer can be omitted for their extremely fewer operating units which are usually uncorrelated with the image size. Specifically, Layers 1 and 2 are pixel manipulation and they are completed simultaneously in a single image scan which could reduce the repeat access to memory. Suppose the image size is N, layer 1 needs N or $2N$ memory access to get the CbCr value, $N \sim 4N$ comparison instructions to run our simplified rectangle judgment and N instructions to link the second layer. Layer 2 needs at most N/τ (usually $\tau = 10$) memory access to store the contour points and $3N$ normal instructions [29]. As a result, our pixel manipulation totally needs $N \sim 2.1N$ memory access as well as $5N \sim 8N$ normal instruction. Secondly, the time consumption in Layers 3 and 4 is extremely low for its dynamic properties and relatively much fewer operating units, see Table II. At last, the time consumption of the modified V-J detector in P-FAD is reduced significantly compared to its original version, it is usually proportional to the number of face candidates instead of being influenced by image size N.

For the original V-J detector, it firstly need to slide search window across the original image with a step Δ, that will produce N/Δ sub-windows to be classified. Notice that the size of face is not invariant, so it also scale down the image with a factor s and then search the image again [28]. As a result, the totally sub-windows of the V-J detector is $\sum_{k=0}^{\log_{1/s} N} N/\Delta$, which is nearly equal to $C * NlgN$ when N is large (C is a constant).

As a summary, the computation overhead in P-FAD is $O(N)$, which is similar with the simple image process functions, and much lower than the original Viola–Jones detector with computation overhead $O(NlgN)$.

IV. Performance Evaluation

In this section, the proposed P-FAD is firstly separately evaluated on a self-record video, which aims to illustrate the performance of P-FAD's each layer. Then the systematic evaluations and the comparison with other schemes are conducted on a MOBIO database [15], in which our method's high efficiency as well as acceptable accuracy are both demonstrated. Separated evaluation and systematic tests are both conducted on a DELL notebook with a 2.2-GHz Pentium Dual-Core CPU, 2 GB memory and runs on Windows 7 operating system. At last, we implement our P-FAD on an Android phone and a low-cost embedded smart camera, to prove the computation efficiency of P-FAD, which is required by resource-limited situation.

A. *Separated Evaluation*

The separated evaluation of P-FAD's each layer is on a 640×480 (VGA resolution) video sequence, which is directly obtained from the notebook's camera without any postprocess. This video sequence totally contains 1546 frames under the changing illumination as well as the varying pose of four person. We choose 100 frames from this sequence which contain 251 frontal or near frontal faces totally. Part of frames are listed in Fig. 9.

1) Evaluation of the Skin Detection: First, we evaluate the adaptive GMM algorithm (locate in P-FAD's first layer) on this video sequence. The skin-tone detection's probability of detection (PD) and probability of false alarm (FA) with two fixed rectangle model and a nonlinear color transformation model [10] are shown in Fig. 10. The six selected frames correspond to the picture 2, 3, 5, 8, 10, and 16, respectively, in Fig. 9. It can be seen that last three frames are darker than the first three frames because a man stood by the windows, and different persons in different frames have various pose.

From Fig. 10, fixed model 1 may lose its PD when the environment get darker; fixed model 2 may produce too much FA in a brighter condition although it gets the similar performance with adaptive model in the last frame. We can conclude that our

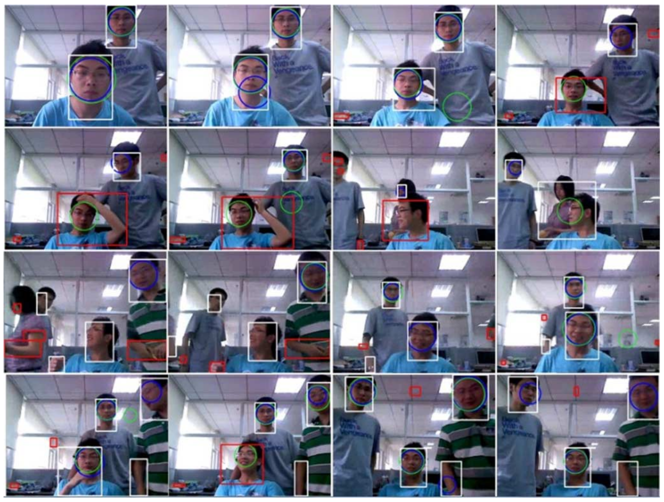


Fig. 9. Part of frames from the test video sequence and their detection results. Rectangles: the output of Layer 4, where red ones are rejected by region filter and white ones are face candidates to our modified Haar detector; Circles: final results, where blue ones are P-FAD's results and green ones are the OpenCV detector's results.

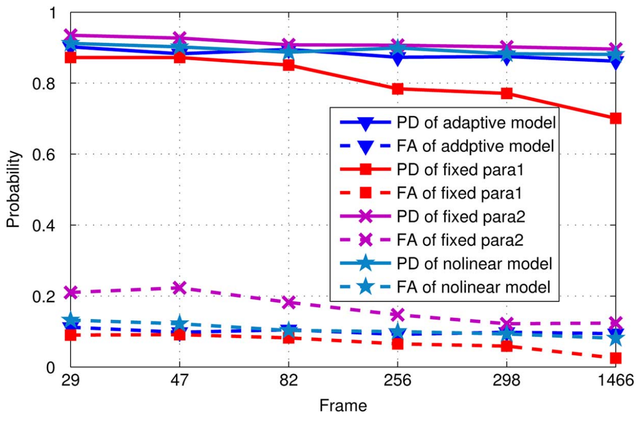


Fig. 10. Skin-tone detection result in a video sequence: Fixed parameter 1 is "$77 \leq Cb \leq 127, 133 \leq Cr \leq 173$," fixed parameter 2 is "$82 \leq Cb \leq 134, 127 \leq Cr \leq 170$."

adaptive method and the nonlinear transformation model are robust to different environment.

Table I list the run-time of different methods for skin detection. It shows that the proposed adaptive method can achieve fast detection speed and high detection accuracy simultaneously. The point here need to be mentioned is that our method actually needn't to build the so-called mask or binary image because the output of skin detection is directly processed by the following layer. It results in even fewer memory access than the simplest fixed model. As the memory access operations dominate the run-time for a general-purpose processor, it makes our method is faster than that of fixed model. Further, nonlinear transformation involves so many float point arithmetics that it could not meet the real-time requirement despite its good performance.

2) Evaluation of the Shift Process: In Fig. 6, we gave the flow chart of the our three-layers shift process as well as the component finding approach [25]. Table II shows their time consumption. The implementation of component finding approach

TABLE I
AVERAGE TIME CONSUMPTION OF SKIN DETECTION IN DIFFERENT METHOD

Method	Adaptive GMM	Fixed para	Nonlinear transorm
Time(us)	2025	2574	170756

TABLE II
TIME CONSUMPTION OF SHIFT PROCESS

	None	One face	Two faces	Three faces
Proposed	1005us	1056us	1125us	1191us
Condition I	4086us	4159us	4357us	4320us
Condition II	24550us	24123us	24685us	24539us

TABLE III
TIME CONSUMPTION OF P-FAD'S LAYER 2–4

Layer	contour detection	dynamic group	region merge & filter
Time(us)	996 - 1131	6 - 69	3 - 26

TABLE IV
DETECTION RESULTS OF A VIDEO SEQUENCE (IMAGE SIZE 640 × 480) ON A 2.2 GHz NOTEBOOK. FP: FALSE POSITIVE, TP: TRUE POSITIVE, DR: DETECTION RATE

	V-J [17]	P-FAD (without layer5)	P-FAD
Faces	251		
Detected	254	295	233
TP	227	218	214
FP	27	77	19
DR(%)	90.4	86.9	85.3
Subwindows	1246469		93
Time(ms)	426	3.32	7.23

is obtained from OpenCV 2.3 [17]. The mask size of morphological filter is 3×3. The number of iterations is three to obtain a good filter performance. The results show our shift procedure make a wonderful time saving comparing to the component finding method with (Condition I)&without (Condition II) morphological filtering. Moreover, Table III shows that the time complexity of integrated shift procedure is adaptive to the number of faces. That is because our method just deals with target's contour points and regions instead of operating on full image.

Next, we evaluate the accuracy of our shift process, which is measured by the overlap degree of the output rectangles (face candidate) and the ground truth. To illustrate the shift part of P-FAD and component finding approach achieve similar accuracy, we choose the results of Condition I as the ground truth. If we choose the 80% overlapping as a threshold, there are more than 95% face candidates in Condition I could be detected by our method, and the histogram of overlapping area is shown in Fig. 11.

3) Comparison With the Original V-J Detector: Since we have used the modified V-J detector in the last layer, we compare our scheme's detection accuracy and detection speed with it. In our comparison, V-J detector is implemented through OpenCV 2.3 library [17]. Table IV presents the details of comparison result. P-FAD gets the DR (Detect Rate) of 85.3% comparing to the V-J detector's 90.4%, but P-FAD's number of FP (False Positive) is 19, just 70.4% of Viola–Jones detector's. Therefore, P-FAD has the similar performance compared with V-J detector in term of accuracy. Furthermore, we can spilt our

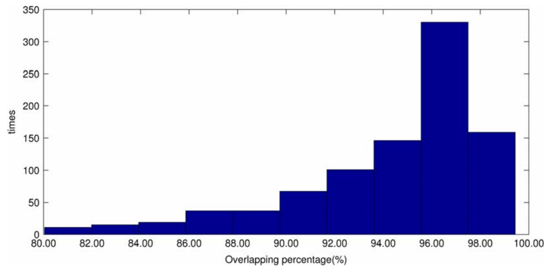


Fig. 11. Histogram plot of the overlapping area between the results of our shift process with and component finding method with morphological filter.

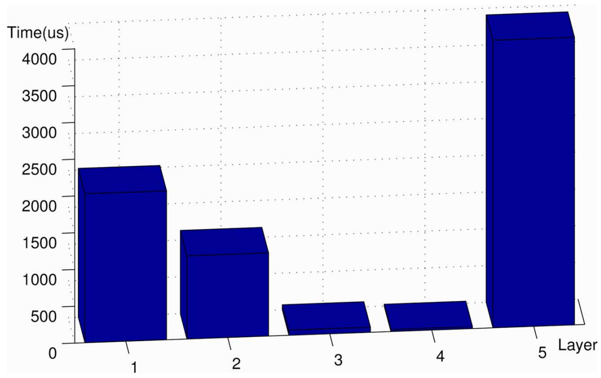


Fig. 12. Time consumption for each layer of P-FAD.

scheme to two part: color-based detection (from layer 1–4) and modified Haar detector (layer 5) for the convenience of comparison. We can find the first part have a relatively acceptable DR comparing to the V-J detector, but it also brings a large FP. Through part II, although the DR cannot be improved, the FP of P-FAD is decreased dramatically. As a result, the combine of these two parts makes our whole scheme has a comparable accuracy with V-J detector. Moreover, our P-FAD just consumes 7.23 ms to process a frame, which speed up the original V-J detector by more than 50 times.

At last, we list out the time consumption of P-FAD's five layers, see Fig. 12.

B. Systematic Evaluation

1) Goals, Methodologies and Metrics: The goals of our systematic evaluation is to demonstrate whether P-FAD

- could be highly efficient to meet the real-time requirement on resource-limited embedded smart cameras;
- guarantee the acceptable performance (higher detection rate, fewer false alarms and exact face position) comparing to the other methods;
- could be robust to the environment such as illumination change, different person and varying pose.

Besides P-FAD, we also implement four other methods for comparison. They are V-J detector [28], nonlinear transformed color detector (NT-CD) [10], V-J detector calibrated color detector (VJ-CD) [19], and color detector followed by a V-J detector (CD-VJ) [32].

V-J detector [28] is a classifier-based method (use Haar feature) and to be regarded as the most successful detector in the last decade. NT-CD is a pure color-based detector, in which a nonlinear transformation of color vector in YCbCr color space is proposed. As P-FAD has both used color information as well as the Haar-feature of face, we should compare our method with them. Furthermore, VJ-CD incorporates a V-J detector to impose the on-the-fly light source calibration for a color-based detector. CD-VJ refines the results of color-based detector through the post process of a V-J detector. Both of them, like P-FAD, are hybrid method that properly combine the color-based detector and V-J detector. This combination results in a considerable computation reduction while still hold the acceptable accuracy. We will show our P-FAD could achieve superior performance than VJ-CD and CD-VJ in terms of detection speed.

We evaluate and compare the above mentioned methods on the MOBIO database [15]. MOBIO database is a multi-modal database that was designed for face detection in mobile/tablet environment, which is the developing trend of smart cameras. MOBIO was captured at six different sites in five different country through mobile device (Nokia N93i mobile phone and a standard 2008 MacBook laptop), where Switzerland (IDIAP) and Czech Republic (BUT) are two sites which are used to evaluation the system. We have chosen the BUT sites, in which 32 subjects (15 females and 17 males) are included and there are 192 unique audio-video samples for each of the 32 participants. Besides, every video also provides one frame, that means 6144 images totally to evaluate face detector. These images are taken from different persons under highly changing situations, such as pose, illumination, expression and the vibration of mobile devices.

In MOBIO, the manual locations of eyes in images are provided. We can use it to assess the performance of face detection through Jesorsky measure [12], which are usually used to assess the performance of face/eye localization. From the detected face bounding box (rectangle), the position of eyes can be often inferred. We compare it with the annotate eyes location to evaluate face detector. The Jesorsky measure ϵ_J, defines how much both eye location vary from the ground truth relative to the distance between the eyes. Details about the definition of ϵ_J and the choice of threshold to judge a success detection can be found in [15]. Furthermore, we can derive DR and FA based on Jesorsky measure to evaluate detector.

2) Evaluation on Videos: To show our method's highly efficiency, we systematic evaluate our method as well as four other methods on videos which are taken from the MOBIO database. Specifically, the videos are $f401\_01\_f12\_i0\_0 \cdot mp4$, $f433\_10\_p05\_i0\_0 \cdot mp4$, $m424\_11\_f07\_i0\_0 \cdot mp4$ and $m429\_04\_f12\_i0\_0.mp4$ from BUT site in MOBIO. There are four different people in videos under changing environment. The persons are asked to answer some short questions, which leads to the continually changing of pose. The mobile device for recording the video, is vibrating to simulate for the real situation.

Table V shows the detection accuracy and run-time of different methods. Although V-J and NT-CD achieve a higher detection accuracy than hybrid method (VJ-CD, CD-VJ, and P-FAD), they sacrifice the computational ability which makes them unfeasible in resource-limited embedded smart camera. P-FAD provides a detection speed gain of 4.7× over CD-VJ

TABLE V
PERFORMANCE COMPARISON OF PROPOSED P-FAD WITH V-J, NT-CD, VJ-CD, AND CD-VJ ON VIDEOS (IMAGE SIZE 640 × 480)

	Face detection rate (%)	False acceptance rate (%)	Time consumption (ms)
Viola-Jones detector (V-J)	88.7	5.3	426.0
Nonlinear Transformed Color Detector (NT-CD)	86.3	20.1	170.8
V-J detector followed by Color Detector (VJ-CD)	85.3	14.6	62.5
Color Detector followed by V-J detector (CD-VJ)	82.4	3.7	34.3
P-FAD	85.2	4.4	7.23

TABLE VI
ACCURACY COMPARISON OF PROPOSED P-FAD WITH V-J AND NT-CD METHODS ON 384 IMAGES, WHICH ARE TAKEN FROM THE BUT SITE OF MOBIO DTATBASE

	P-FAD	Viola-Jones detector(V-J)	Nonlinear Transformed Color Detector(NT-CD)
Face detection rate(%)($\epsilon_J < 0.25$)	87.5	89.2	88.3
False acceptance rate(%)($\epsilon_J > 0.25$)	4.3	6.4	23.1
Average Jesorsky measure ϵ_J($\epsilon_J < 0.25$)	0.0834	0.0823	0.102

and 8.6× over VJ-CD with similar detection accuracy. The main reasons are as follows.

- Compared to VJ-CD treating V-J detector and color detector equally on the full image, P-FAD's attentional cascade structure reserves the complex process on the promising region.
- Compared to CD-VJ repeatedly using binary image process to spilt the complex process from pixel manipulation, P-FAD provides an extremely low cost shift.

3) Evaluation on Images: The above evaluation on videos demonstrates the real-time property of P-FAD. In this images-based evaluation part, we focus on precisely evaluating its detection accuracy. Precise evaluation is based on the Jesorsky measure to assess the error between annotated eye location and referred eye location from detected bounding box. This error is normalized by the distance between left and right eyes. As we know, face detection is usually followed by other facial analysis techniques, such as face alignment, eye localization, face recognition, and authentication, in which eye location is usually needed. So the Jesorsky metric is more precise measure than the overlap-based measure [15].

Before using the Jesorsky measure to evaluate the position accuracy of our face detector, we should train a transformed matrix T as follows:

$$\underbrace{\begin{bmatrix} x_l \\ y_l \\ x_r \\ y_r \end{bmatrix}}_{\text{eye location}} = T * \underbrace{\begin{bmatrix} x_b \\ y_b \\ w_b \\ h_b \end{bmatrix}}_{\text{bounding box}} . \tag{10}$$

Matrix T is used to estimate the eye location from detected bounding box. According to (10), the bounding boxes can be transformed into eye location. Herein, we only consider the linear relationship between the bounding box's parameters and the eye location, which results in a 4 × 4 transform matrix. We have picked 32 images from UNIS site of MOBIO database, to estimate the entries of T via least squares fit. Result is given as following:

$$T = \begin{bmatrix} 0.983 & 0.025 & 0.395 & 0 \\ 0.023 & 0.907 & 0 & 0.536 \\ 0.995 & 0.077 & 0.880 & 0 \\ 0.043 & 0.910 & 0 & 0.5264 \end{bmatrix} . \tag{11}$$

In this part, we just evaluate P-FAD with V-J detector and NT-CD to focus on the accuracy performance. The test images are picked from BUT site of MOBIO. We pick 12 images for every subject with varying situations, so that 384 images are picked from 32 subjects totally. Table VI shows the comparison of their performance. The calculation of detection rate as well as false acceptance rate are based on the Jesorsky measure ϵ_J: one detection result is registered as positive when $\epsilon_J < 0.25$. This threshold means that if the distance between left and right eyes is 10 pixels, the maximum error of referred eye location should be less than 2.5 pixel to be a successful detection. Fig. 13 lists a part of detection results of P-FAD. Fig. 14 plots the details of three methods' detection accuracy. We can see our P-FAD's error distribution is comparable with V-J detector, but much better than the NT-CD. The average error of P-FAD is just 81.8% of NT-CD's.

C. Implementation on Smart Phones

The emerging mobile technologies (phones, tablets, etc.) and its related applications have changed people's life. We have implemented P-FAD on a HUAWEI U8800 smart phone. It is with a 5 Mega-pixel camera at the rear, 800 MHz Qualcomm Snapdragon MSM7230 processor, 512 MB RAM and runs a Android 2.2 operating system.

First, we compare the detection speed of different face detection method. Besides the aforementioned five methods, the Android API of face detection is evaluated too. As the development language of Android apps is Java, which is usually far from efficient in spite of its cross-platform property. We actually implement P-FAD and the other algorithms in C++ (which is more efficient than Java) and then use Java native interface (JNI) method to integrate algorithms into our Android test application.

Generally, an applicable and complete implementation of face detection on smart phone includes three major stages: obtain the image, process the image, and display the results. Table VII shows the time consumption of these three stages on U8800 in different resolution. In stage I, we have two ways to obtain the image: directly grab a image from camera, or load image data from files such as bmp and jpeg. Moreover, To facilitate the further process, sometimes we need to transform the color space in stage I. In stage II, we give different algorithms'

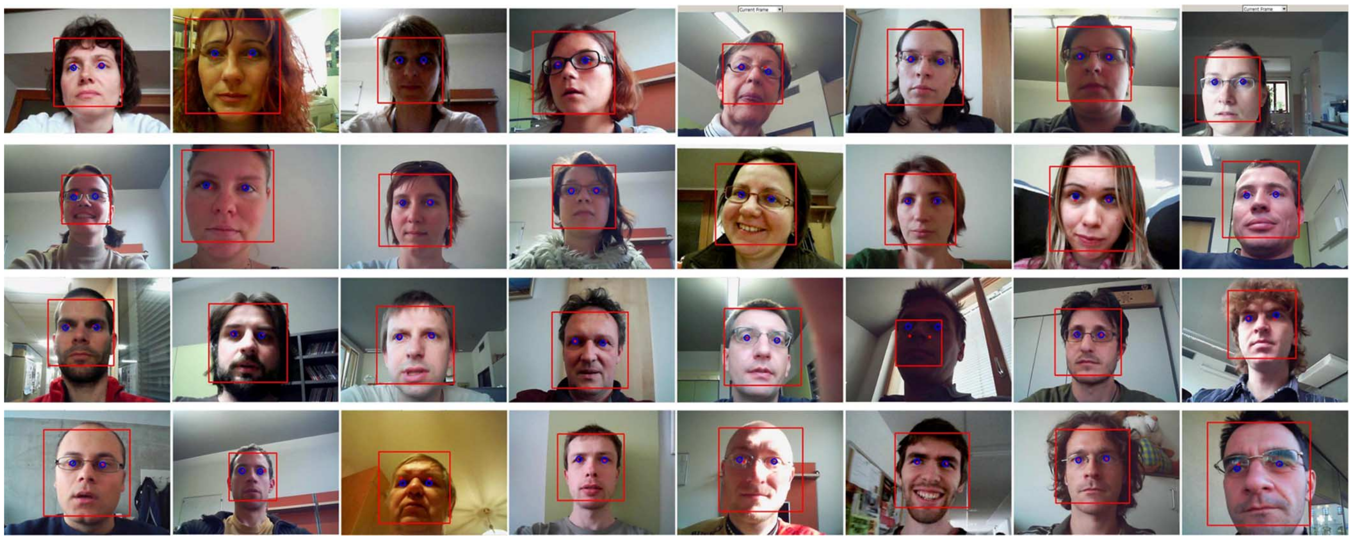


Fig. 13. Some detection results of P-FAD on images. In which red rectangle are the bounding box of face which is detected by P-FAD, red point is the referred eye location through (10), the center of blue circle is the ground truth provided by database.

TABLE VII
TIME CONSUMPTION (MS) OF IMPLEMENTING FACE DETECTION ON HUAWEI U8800 WITH DIFFERENT IMAGE RESOLUTION

Image size	**320x240** (QVGA)	**640x480** (VGA)	**800x480** (WVGA)
Stage 1: Grab/Load and transform (in color space) a image			
Grab/Load a image	4.2/16.4	15.7/46.3	20.4/103.6
color space transformation	9.2	31.7	41.7
Stage 2: Different methods to process a image (face detection)			
P-FAD	4.5	18.8	27.4
Color Detector followed by V-J detector (CD-VJ)	22.4	83.7	134.3
V-J detector followed by Color Detector (VJ-CD)	40.3	154.6	262.5
Nonlinear Transformed Color Detector (NT-CD)	106.3	520.1	670.8
Viola-Jones detector (V-J)	330.7	1327.3	1779.8
Android API for face detection	207.2	963.2	1424.5
Stage 3: Display the process result on the screen of the smart phone			
Display the result on the screen	8.3	55.2	67.2

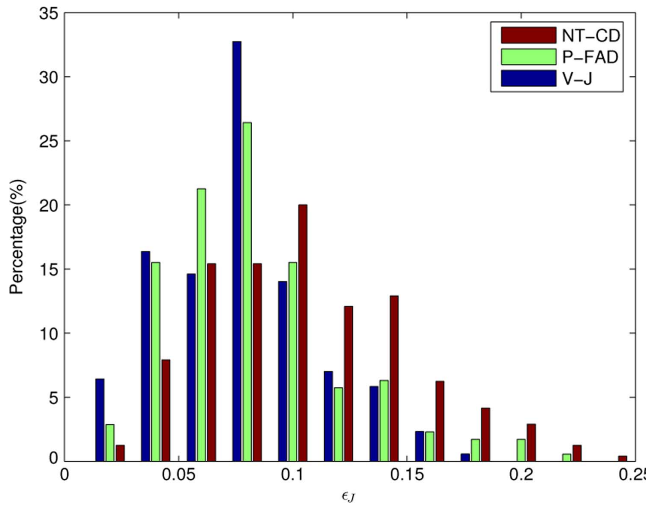


Fig. 14. Histogram of face detection's error, which is measured by the ϵ_J. Notice that we have use the threshold 0.25 (less than) to confirm the positive detection, so the entries of more than 0.25 are omitted, herein only to precisely illustrate the accuracy in these positive detections.

run-time for comparison in Table VII. It shows that P-FAD achieves the best performance among these algorithms. Stage III is the time consumption to display the process results, which could be omitted when the application does not need to display the image (such as background program). Table VII shows that the face detection procedure dominates the time consumption of an application by using V-J detector, Android API and NT-CD. The real-time approaches, VJ-CD and CD-VJ, can speed up the detection process. But it is still the running bottleneck of an application. P-FAD outperforms VJ-CD and CD-VJ up to 8.2× on a VGA image, which fully eliminates the detection bottleneck problem. Practically, built-in camera outputs image data in YCbCr color space directly. Therefore, a face detection application with P-FAD can process a VGA image in 34.5 ms (28.9 FPS) under the background mode. Even in the worst case $(\text{load image} + \text{color transform} + \text{process} + \text{display})$, P-FAD based system could still achieve the 7.1FPS for VGA size and 26 FPS for QVGA size, while V-J detector and Android API can only get the 0.7/2.7FPS and 0.9/4.1FPS, respectively.

Considering the limited battery power that smart phone has, we have also used a software tool PowerTutor [18], to monitor the energy usage of different methods on smart phone. Fig. 15 plots the energy consumption of different method to process a image in different resolution. P-FAD consumes nearly 1/50 energy of V-J detector, and 1/40 energy of Android API.

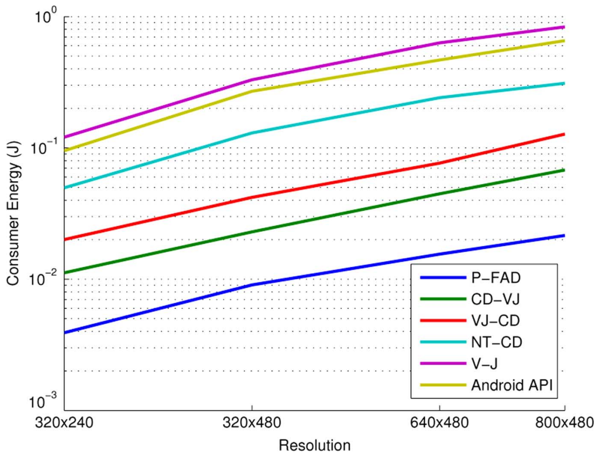


Fig. 15. Energy consumption (per frame) of face detection on U8800.

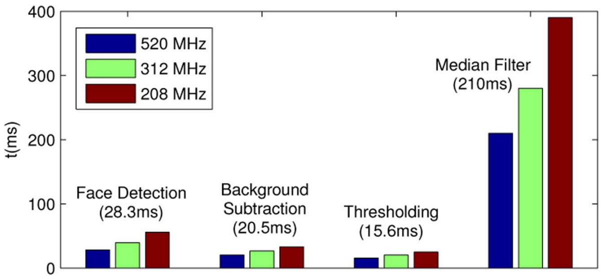


Fig. 16. Time consumption on embedded smart cameras.

D. Implementation on Embedded Smart Camera

At last, we test our P-FAD scheme on a low-cost embedded smart camera to show our method's resource-aware property. The embedded smart camera platform consists of a Intel Xscale microprocessor PXA270 and a image sensor OV9655, which is based on the CITRIC architecture [4]. Fig. 16 shows the limited capability of our embedded platform by giving the run-time of basic image processing function in three frequencies. P-FAD requires only 28.3 ms to process a VGA image, which is almost the same as a typically background subtraction operation. As we have discussed in Section III-D, the reason is that the background substraction needs at least three memory access for every pixel. While our pixel-level operation, the first and second layer in pyramid-like scheme, only needs 1 to 2.1 operations on memory. Though further process are complex, the dramatically decreased operating units could save run-time.

V. Conclusion

We have presented P-FAD, a hierarchical face detection framework for reducing the computing and storage overhead on embedded smart cameras. Our goal was to reduce the pixel manipulation without compromising the detection performance. We met this goal by devising a three-stage procedure, called as coarse, shift, and refine process. It shifts the operating unit from pixel to contour points, regions and face candidates and reserves complex processing for the given promising units. Our experimental results exhibit its resource-aware property that P-FAD could process a VGA image in just 7.23 ms on a notebook, 18.8 ms on a smart phone and 28.3 ms on a light-weight embedded smart camera platform.

References

[1] L. Acasandreni and A. Barriga, "Accelerating Viola-Jones face detection for embedded and SoC environments," in *Proc. 5th ACM/IEEE Int. Conf. Distrib. Smart Cameras*, Aug. 2011, pp. 1–6.

[2] M. Bramberger, J. Brunner, B. Rinner, and H. Schwabach, "Real-time video analysis on an embedded smart camera for traffic surveillance," in *Proc. Real-Time Embed. Technol. Appl. Symp.*, May 2004, pp. 174–178.

[3] M. Bramberger, A. Doblander, B. Rinner, and H. Schwabach, "Distributed embedded smart cameras for surveillance applications," *Computer*, vol. 39, no. 2, pp. 68–75, Feb. 2006.

[4] P. Chen, P. Ahammad, and C. Boyer, "CITRIC: A low-bandwidth wireless camera network platform," in *Proc. 2nd ACM/IEEE Int. Conf. Distrib. Smart Cameras*, Aug. 2008, pp. 1–10.

[5] H. Y. Chen, C. L. Huang, and C. M. Fu, "Hybrid-boost learning for multi-pose face detection and facial expression recognition," *Pattern Recognit.*, vol. 41, pp. 1173–1185, 2008.

[6] J. Cho, S. Mirzaei, and R. Kastner, "FPGA-based face detection system using Haar classifers," in *Proc. ACM/SIGDA Int. Symp. Filed Program. Gate Arrays*, 2009, pp. 103–112.

[7] C. Gao and S. L. Lu, "Novel FPGA based Haar classifier face detection algorithm acceleration," in *Proc. Int. Conf. Field Program. Logic Appl.*, Sep. 2008, pp. 373–378.

[8] Y. C. Ham and Y. Shi, "Developing a smart camera for gesture recognition in HCI applications," in *Proc. Int. Symp. Consum. Electron. Symp.*, May 2009, pp. 994–998.

[9] D. Hefenbrock, J. Oberg, N. T. N. Thanh, R. Kastner, and S. B. Baden, "Accelerating Viola-Jones face detection to FPGA-level using GPUs," in *Proc. IEEE Annu. Int. Symp. Field-Program. Custom Comput. Mach.*, 2010, pp. 11–18.

[10] R. L. Hsu, A.-M. Mohamed, and A. K. Jain, "Face detection in color images," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 24, no. 5, pp. 696–706, May 2002.

[11] V. Jacobson, "Congestion avoidance and control," in *Proc. SIGCOMM Conf.*, May 1988, vol. 18, no. 4, pp. 314–329.

[12] O. Jesorsky, K. J. Kirchberg, and R. Frischholz, "Robust face detection using the Hausdorff distance," in *Proc. Int. Conf. Audio- Video-Based Biometric Person Authent.*, U.K., 2001, pp. 65–90.

[13] P. Kakumanu, S. Makrogiannis, and N. Bourbakis, "A survey of skin-color modeling and detection methods," *Pattern Recognit.*, vol. 40, pp. 1106–1122, 2007.

[14] R. Kleihorst, M. Reuvers, B. Krose, and H. Broers, "A smart camera for face recognition," in *Proc. Int. Conf. Image Process. Conf.*, 2004, pp. 2849–2852.

[15] C. McCool, S. Marcel, A. Hadid, M. Pietikainen, P. Matejka, J. Cernocky, N. Poh, J. Kittler, A. Larcher, C. Levy, D. Matrouf, J. F. Bonastre, P. Tresadern, and T. Cootes, "Bi-modal person recognition on a mobile phone: Using mobile phone data," in *Proc. 2012 IEEE Int. Conf. Multimedia Expo Workshops*, Jul. 2012, pp. 635–640.

[16] S. McKenna, S. Gong, and Y. Raja, "Modeling facial colour and identity with Gaussian mixtures," *Pattern Recognit.*, vol. 31, no. 12, pp. 1883–1892, Dec. 1998.

[17] OpenCV [Online]. Available: http://www.opencv.org.cn/opencvdoc/2.3.2/html/index.html

[18] PowerTutor [Online]. Available: http://ziyang.eecs.umich.edu/projects/powertutor/

[19] M. Rahman, N. Kehtarnavaz, and J. Ren, "A hybrid face detection approach for real-time deployment on mobile devices," in *Proc. 16th IEEE Int. Conf. Image Process.*, Nov. 2009, pp. 3233–3236.

[20] L. Ray and H. Nicponski, "Face detecting camera and method," U.S. Patent 6 940 545, Sep. 2005.

[21] J. Ren, N. Kehtarnavaz, and L. Estevez, "Real-time optimization of Viola-Jones face detection on mobile platforms," in *Proc. IEEE Int. Conf. Acoust., Speech, Signal Process.*, Oct. 2009, pp. 1–4.

[22] J. Ren, X. Jiang, and J. Yuan, "A complete and fully automated face verification system on mobile devices," *Pattern Recognit.*, vol. 46, no. 1, pp. 45–56, Jan. 2013.

[23] H. A. Rowley, S. Baluja, and T. Kanade, "Neural network-based face detection," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 20, no. 1, pp. 23–38, 1998.

[24] G. Sebastian, X. Xie, W. Philips, C. W. Chen, and H. Aghajan, "A best view selection in meetings through attention analysis using a multi-camera network," in *Proc. 6th Int. Conf. Distrib. Smart Cameras*, 2012, pp. 1–6.

[25] K. Suzuki, I. Horiba, and N. Sugie, "Fast connected-component labeling based on sequential local operations in the course of forward raster scan followed by backward raster scan," in *Proc. 15th Int. Conf. Pattern Recognit. Conf.*, Aug. 2000.

[26] T. Teocharides, N. Vijaykrishnam, and M. J. Irwin, "A parallel architecture for hardware face detection," in *Proc. IEEE Comput. Soc. Annu. Symp. Emerg. VLSI Technol. Archit.*, 2006.

[27] J. C. Terrillon, M. N. Shirazi, H. Fukamachi, and S. Akamatsu, "Comparative performance of different skin chrominance models and chrominance spaces for the automatic detection of human faces in color images," in *Proc. 4th IEEE Int. Conf. Automat. Face Gesture Recognit.*, 2000, pp. 54–61.

[28] P. Viola and M. Jones, "Robust real-time face detection," *Int. J. Comput. Vis.*, vol. 57, no. 2, pp. 137–154, 2004.

[29] Q. Wang, P. Zhou, J. Wu, and C. Long, "RaFFD: Resource-aware fast foreground detection in embedded smart cameras," in *Proc. IEEE GLOBECOM Conf.*, Dec. 2012, pp. 1–6.

[30] Q. Wang, J. Wu, C. Long, and B. Li, "P-FAD: Real-time face detection scheme on embedded smart cameras," in *Proc. ACM/IEEE Int. Conf. Distrib. Smart Cameras*, Nov. 2012, pp. 1–6.

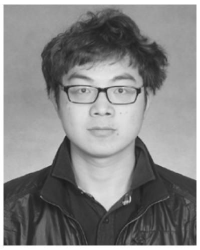
[31] T. Winkler and B. Rinner, "TrustCAM: Security and privacy-protection for an embedded smart camera based on trusted computing," in *Proc. IEEE Int. Adv. Video Signal-Based Surveill. Conf.*, Aug. 2010, pp. 593–600.

[32] Y. W. Wu and X. Y. Ai, "Face detection in color images using AdaBoost algorithm based on skin color information," in *Proc. 2008 Int. Workshop Knowl. Discov. Data Mining*, 2008, pp. 339–342.

[33] C. Wu and H. Aghajan, "User-centric environment discovery with camera networks in smart homes," *IEEE Trans. Syst., Man, Cybern. Part A*, vol. 41, no. 2, pp. 375–383, Mar. 2011.

[34] M. H. Yang, D. J. Kriegman, and N. Ahuja, "Detecting faces in images: A survey," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 24, no. 1, pp. 34–58, Jan. 2002.

[35] C. Zhang and Z. Y. Zhang, "A survey of recent advances in face detection," *Microsoft Res.*, Jun. 2010.



Qiang Wang (S'12) received the B.S. degree in control science and engineering from Wuhan University of Science and Technology, Wuhan, China, in 2011. He is currently working toward the M.S. degree in the school of Electronic Information and Electrical Engineering, Shanghai Jiao Tong University, Shanghai, China.

His current research interests include computer vision, resource-awared image process and their applications for smart camera network.

Jing Wu (S'05–M'08) received the B.Sc. degree from Nanchang University, Nanchang, China, the M.Sc. degree from Yanshan University, Qinhuangdao, China, and the Ph.D. degree from University of Alberta, Edmonton, AB, Canada, in 2000, 2002, and 2008, respectively, all in electrical engineering.

From 2008 to 2011, she was a Project Engineer with Tyco Thermal Controls-Tracer Division, Edmonton, AB, Canada. Since 2011, she has been with Shanghai Jiao Tong University, Shanghai, China, and is currently an Associate Professor. Her current research interests include network-based control systems, smart grid and their applications to industrial problems.

Dr. Wu is a registered Professional Engineer in Alberta, Canada. She was a recipient of the Dissertation Fellowship for 2007 and the iCore Graduate Student Scholarship for 2005/2006, at the University of Alberta.

Chengnian Long (M'07) received the B.S., M.S., and Ph.D. degrees from Yanshan University, Hebei, China, in 1999, 2001, and 2004, respectively, all in control theory and engineering.

He is presently a Professor in the School of Electronic Information and Electrical Engineering, Shanghai Jiao Tong University, Shanghai, China. He joined the Shanghai Jiao Tong University in January 2009. Before that, he was at the Department of Electrical and Computer Engineering, University of Alberta since January 2007, where he was awarded Killam postdoctoral fellowship. He visited Department of Computer Science and Engineering, Hongkong University of Science and Technology, in 2006. His current research interests include wireless networks and their applications to smart grid, smart camera networks.

Bo Li (S89–M92–SM99–F'11) received the B.Eng. degree in computer science from Tsinghua University, Beijing, China, and the Ph.D. degree in the electrical and computer engineering from the University of Massachusetts, Amherst, MA, USA.

He is a Professor in the Department of Computer Science and Engineering, Hong Kong University of Science and Technology. He holds a Cheung Kong Chair Professor in Shanghai Jiao Tong University, China, and he is the Chief Technical Advisor for China Cache Corporation, the largest CDN operator in China (NASDAQ CCIH). He was with IBM Networking System, Research Triangle Park, NC, USA (1993–1996). He was an Adjunct Researcher at Microsoft Research Asia (1999–2007) and at Microsoft Advanced Technology Center (2007–2008). His current research interests include large-scale content distribution, data center networking, cloud computing, and device-to-device communications. He made pioneering contributions in the Internet video broadcast with a system, Coolstreaming (the keyword had over 2 000 000 entries on Google), which was credited as the world first large-scale peer-to-peer live video streaming system. The work first appeared in IEEE INFOCOM (2005) has not only been widely cited, but also spearheaded a momentum in peer-to-peer streaming industry with no fewer than a dozen successful companies adopting the same mesh-based pull streaming technique to deliver live media content to hundreds of millions of users in the world.

Dr. Li received the prestigious State Natural Science Award from China and five Best Paper Awards from IEEE. He was a Distinguished Lecturer in IEEE Communications Society (2006–2007). He has been an Editor or a Guest Editor for over a dozen IEEE journals and magazines. He was the Co-TPC Chair for IEEE INFOCOM (2004).