Machine Vision and Applications (2015) 26:233–249
DOI 10.1007/s00138-014-0655-9

ORIGINAL PAPER

Real-time automatic multilevel color video thresholding using a novel class-variance criterion

Chi-Yi Tsai · Tsung-Yen Liu

Received: 2 March 2014 / Revised: 12 October 2014 / Accepted: 8 December 2014 / Published online: 1 January 2015

Abstract Color image segmentation is a crucial preliminary task in robotic vision systems. This paper presents a novel automatic multilevel color thresholding algorithm to address this task efficiently. The proposed algorithm consists of a learning process and a multi-threshold searching process. The learning process learns the color distribution of an input video sequence in HSV color space, and the multi-threshold searching process automatically determines the optimal multiple thresholds to segment all colors-of-interest in the video based on a novel class-variance criterion. For the learning process, a simple and efficient color-distribution learning algorithm operating with a color-pixel extraction method is proposed to learn a color distribution model of all colors-of-interest in the video images, which simplifies the search for optimal thresholds for the colors-of-interest through a conventional multilevel thresholding method. For the multi-threshold searching process, a nonparametric multilevel color thresholding algorithm with an extended within-class variance criterion is proposed to automatically find the optimal upper bound and lower bound threshold values of each color channel. Experimental results validate the performance and computational efficiency of the proposed method by comparing with three existing methods, both visually and quantitatively.

Keywords Multi-object segmentation · Nonparametric multilevel color thresholding · Extended within-class variance · Automatic multi-threshold searching

C.-Y. Tsai (✉) · T.-Y. Liu
Department of Electrical Engineering, Tamkang University, 151 Ying-zhuan Road, Tamsui Dist., New Taipei City 25137, Taiwan, ROC
e-mail: chiyi_tsai@mail.tku.edu.tw

T.-Y. Liu
e-mail: 600460066@s00.tku.edu.tw

1 Introduction

Image analysis and pattern recognition are essential and important components in numerous robotic vision systems. To improve the quality of analysis and recognition results, image segmentation plays a critical pre-processing role to divide the input image into several homogeneous regions according to statistical characteristics of image features such as gray level, edge, texture and color, etc. Segmentation of natural images is typically a challenging task as it may obtain multiple segmentation results with various numbers of segments or model dimensions, which is called the inherently ambiguous problem [1]. Various gray-level image segmentation techniques have been proposed to address this problem; however, they usually cannot be applied directly on color images (i.e., at least one color representation model is required) [2]. Because color is an crucial feature in numerous robotic vision applications, color image segmentation has gained considerable attention and has become an active research field in recent years.

According to [2], most color-image segmentation techniques are developed by combining gray-level image segmentation methods with various color representation models, such as RGB, YUV, and HSV (or HSI), etc. This study used an HSV model because it is more intuitive to human vision and can separate color information of an image from its luminance information [2,3]. For gray-level image segmentation, histogram thresholding is one of the most widely used approaches because of its simplicity, robustness, and accuracy [3]. Conventional image thresholding techniques can be categorized into two types: parametric and nonparametric approaches [4]. Parametric approaches require estimation of the probability density function of the gray-level distribution of each object, which is typically a nonlinear optimization problem and may lead to an inefficient algo-

rithm with high computational complexity. By contrast, nonparametric approaches determine the optimal thresholds by directly optimizing certain cost functions [5–8]. For example, Otsu [5] proposed an exhaustive search method to determine the optimal thresholds by maximizing a between-class variance criterion. However, this method requires a large amount of time for multilevel thresholding because it evaluates all possible solutions. To speed up the maximization process of the Otsu's method, Liao et al. [6] proposed a fast Otsu method that uses a look-up table acceleration approach to maximize a modified between-class variance instead of the original one, efficiently reducing the computational complexity of the exhaustive search process. Next, Huang et al. [7] extended the fast Otsu method by combining it with a two-stage process, termed as the two-stage multi-threshold Otsu (TSMO) method. This method considerably reduces the number of evaluation operations required in the exhaustive search process; however, it may yield a locally optimal solution. Recently, Gao et al. [8] proposed an improved quantum-behaved particle swarm algorithm to increase the convergence rate of the Otsu's method. These nonparametric image thresholding approaches can be applied to each component of RGB color space or the hue component of HSV color space to obtain a coarse color segmentation result, which requires an additional color-region merging process to produce the final result [9,10].

Because image segmentation can be considered as a multiclass classification problem, a number of researchers use data clustering techniques, such as self-organizing map (SOM) network or fuzzy c-means (FCM) algorithm, to address this problem [11–14]. For instance, Wu et al. [11] proposed an SOM-based automatic multilevel thresholding algorithm for color segmentation and human hand localization. An SOM transduction algorithm was proposed to learn a non-stationary color distribution model in HSI color space to overcome the dynamic lighting issue. The authors in [12] proposed an automatic multilevel thresholding algorithm for color image quantization. A growing time adaptive SOM network was proposed to learn a 3-dimensional RGB histogram, and a peak-finding algorithm was then used to automatically determine the optimal thresholds. In [13], the authors proposed a coarse-to-fine image segmentation algorithm based on FCM and image thresholding techniques. This approach first uses an automatic histogram thresholding technique to coarsely segment the image, and the FCM technique is subsequently used in the fine segmentation process to assign unclassified pixels to the closest class. Yu et al. [14] proposed an adaptive unsupervised scheme for color image segmentation. This method uses the FCM technique and ant colony optimization to adaptively detect the cluster centroid distribution and centroid number, which are critical initialization problems in image segmentation. Although both neural-network-based and fuzzy-clustering-based approaches are usually effective in color image segmentation, they require a supervised training process or a cluster-number decision process to obtain optimal segmentation performance, thereby increasing the complexity and restricting the applicability of image segmentation in practice.

Recently, the authors in [15] combined the current TSMO method with data fusion techniques to achieve unsupervised color segmentation. The advantage of this method is that it can improve the information quality and obtain the optimal segmentation result by combining several segmentations of the same image together based on the Dempster–Shafer evidence theory [16]. However, the calculations required for this method are complex and require a long processing time in the data fusion process, resulting in a computationally inefficient solution to color image segmentation.

This study addressed color image segmentation used in a real-time robotic vision system. This is a typical color video segmentation problem, and current methods usually apply a conventional bi-level image thresholding technique to each color channel [17] or to the SI plane [18] of video images. However, these methods may not work when thresholding multiple color objects-of-interest in a video sequence. A previous work [19] proposed a multilevel color thresholding algorithm to segment multiple color-homogeneous regions in a still color image. In this study, the algorithm was extended to color video segmentation. The proposed algorithm consists of a learning process and a multi-threshold searching process. For the learning process, an efficient color-distribution learning algorithm operating with a color-pixel extraction method is proposed to learn a color distribution model of all colors-of-interest in an input video sequence. This helps to search the optimal thresholds for all colors-of-interest using a conventional multilevel thresholding method. For the multi-threshold searching process, a nonparametric multilevel color thresholding algorithm based on a novel class-variance criterion is proposed to automatically find the optimal upper bound and lower bound threshold values of the hue, saturation and brightness components for each color object. The proposed method was compared visually and quantitatively with three current methods, and the experimental results verified the performance and computational efficiency of the proposed method.

The remainder of this paper is organized as follows. Section 2 presents the problem addressed in this study; Sect. 3 introduces the processing steps of the proposed multilevel color thresholding algorithm; Sect. 4 presents the proposed multi-threshold searching algorithm based on a novel class-variance criterion in detail; Sect. 5 presents the experimental results used to evaluate the performance of the proposed multilevel color thresholding approach; and lastly, Sect. 6 provides a conclusion.

2 Problem statement

Image segmentation is an essential task in various robot vision applications. Figure 1 shows the system framework of a conventional robot visual servo control system, in which a robot manipulator attempts to grasp a color object and place it into a target location (i.e., a box of the same color). This type of system usually consists of several brick works including image segmentation, object classification, pose estimation, and visual servoing, etc. To obtain pose information of an object-of-interest from video images, the first task is to accurately extract all color objects from the background. This purpose can be simply achieved using a multilevel color thresholding method, which distinguishes pixels belonging to the foreground and background based on the color distribution of a specific object-of-interest.

Let $H_{\text{in}} \in [0, 360]$, $S_{\text{in}} \in [0, 1]$, and $V_{\text{in}} \in [0, 1]$ denote the input color values of each pixel in HSV color space. Suppose that the color distribution in HSV space of each object-of-interest is continuous with a bounded range for each color channel. Then, for N object classes, $N \geq 1$, it is possible to find $3N$ optimal threshold pairs $(\hat{h}_{\text{l}}^{(n)}, \hat{h}_{\text{u}}^{(n)})$, $(\hat{s}_{\text{l}}^{(n)}, \hat{s}_{\text{u}}^{(n)})$, and $(\hat{v}_{\text{l}}^{(n)}, \hat{v}_{\text{u}}^{(n)})$, for $n = 1 \sim N$, to threshold an input HSV color image into N labels such that

$$L_N(x, y) = \begin{cases} n, & \hat{h}_{\text{l}}^{(n)} \leq H_{\text{in}}(x, y) \leq \hat{h}_{\text{u}}^{(n)}, \\ & \hat{s}_{\text{l}}^{(n)} \leq S_{\text{in}}(x, y) \leq \hat{s}_{\text{u}}^{(n)}, \\ & \hat{v}_{\text{l}}^{(n)} \leq V_{\text{in}}(x, y) \leq \hat{v}_{\text{u}}^{(n)}, \\ 0, & \text{Otherwise}, \end{cases} \tag{1}$$

where $L_N(x, y)$ is the output-labeled image separating the N-object and background pixels with minimal error. Expression (1) indicates that, if the hue, saturation, and brightness values of an input color pixel are within the range of nth lower bound to nth upper bound threshold values, the corresponding output pixel would be assigned to the object class labeled n; otherwise, it is assigned to a null class. Expression (1) is an extension of the typical multilevel thresholding problem in HSV color space (see Remark 1). However, [20] indicated that multilevel thresholding is generally less reliable as it is difficult to establish multiple thresholds effectively isolating each region-of-interest. Therefore, searching for optimal multiple thresholds usually requires a supervised adjustment (i.e., manual adjusting of the thresholds to optimize thresholding) to improve the reliability of multilevel thresholding. This problem highlights the importance of developing a reliable and unsupervised multi-threshold searching algorithm to assist in the search for optimal thresholds.

A previous study [19] proposed a histogram-based multi-threshold searching algorithm to manage this problem, but it was restricted to a still color image. Actually, a robot equipped with an on-board vision system collects information of objects-of-interest via video signals rather than just an image. In other words, the main challenge of multilevel color thresholding in a robotic vision system is the manner in which to determine the optimal threshold values of each color channel for segmenting multiple color objects in a video sequence. Therefore, this paper proposes a novel automatic multi-threshold searching algorithm based on a novel performance criterion to efficiently achieve this objective.

Remark 1 Let L denote the maximal value of the input pixel. Given a multimodal image histogram containing N dominant modes, typical multilevel thresholding techniques aim to find

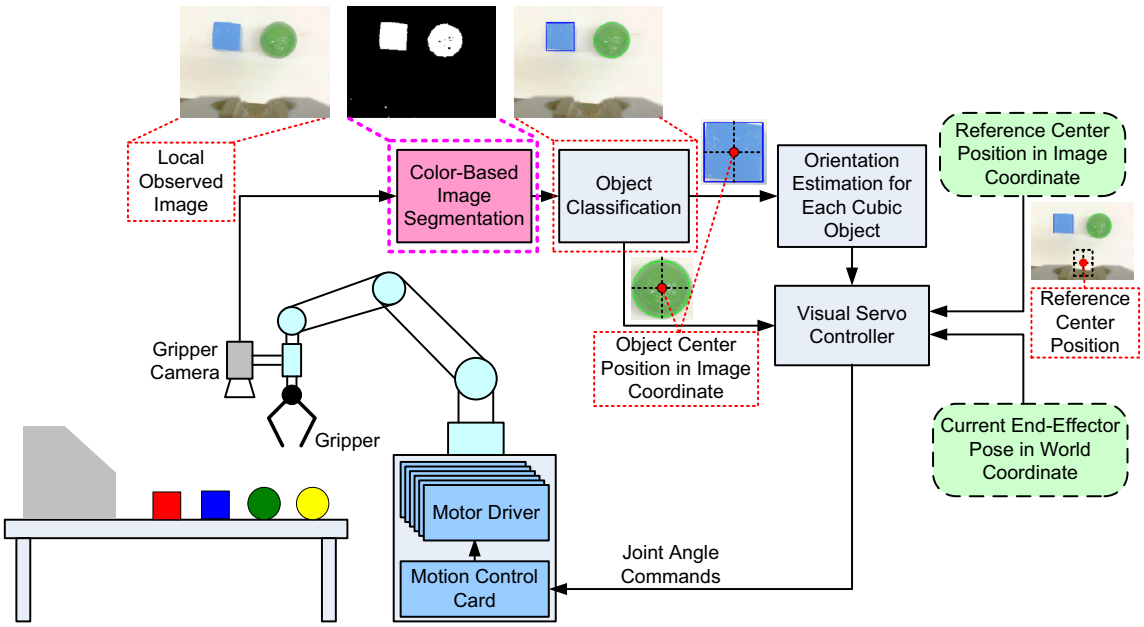

Fig. 1 System framework of a conventional robot visual servo control system

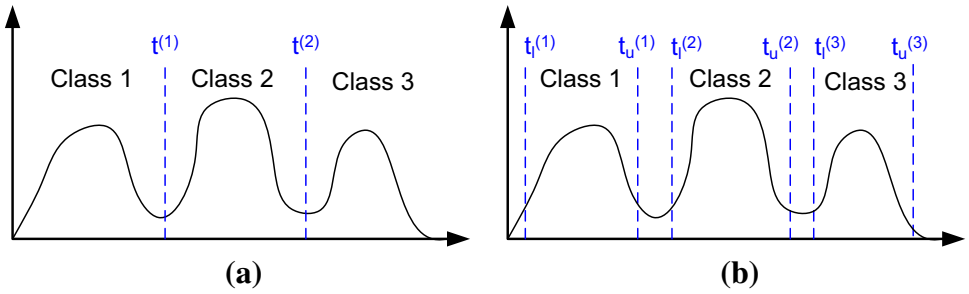

Fig. 2 Two types of the multilevel thresholding problem: **a** the typical problem and **b** the extended problem addressed in this study

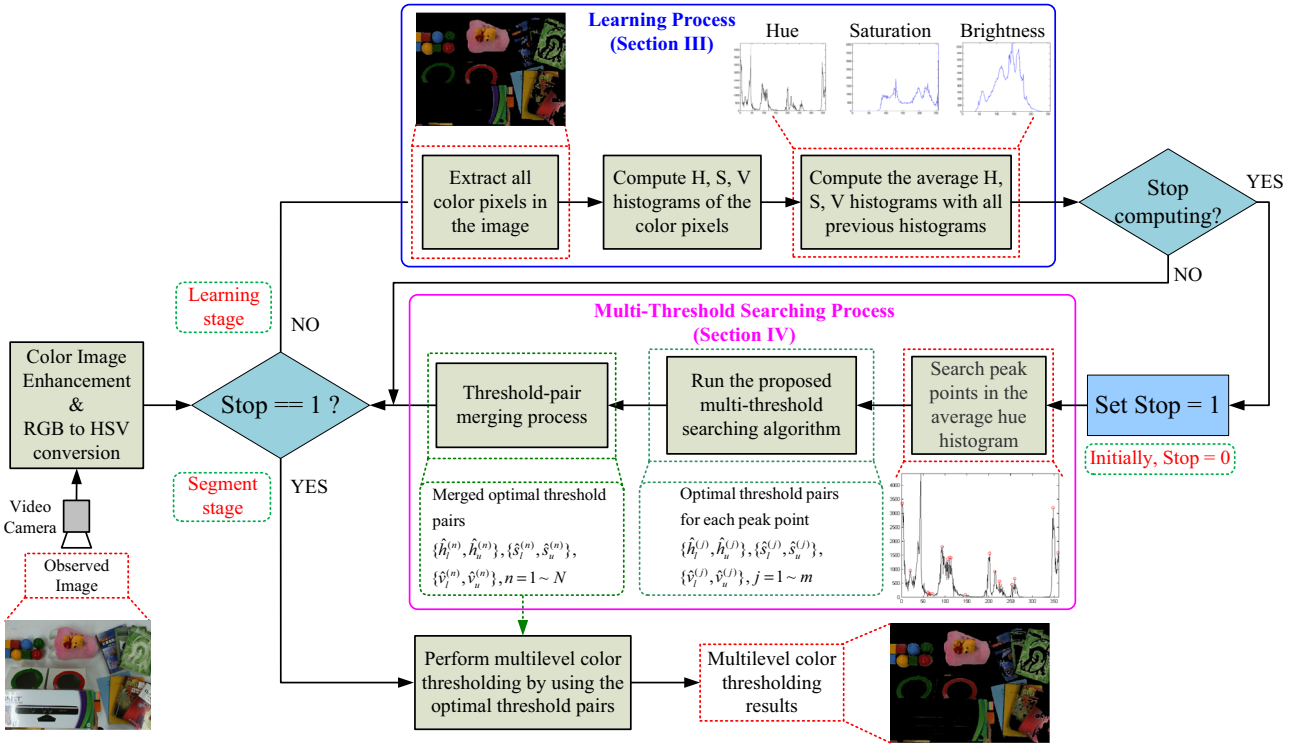

Fig. 3 Flowchart of the proposed multilevel color thresholding algorithm

$N-1$ thresholds $0 < t^{(1)} < t^{(2)} < \cdots < t^{(N-1)} < L$ to separate these modes into N classes, as shown in Fig. 2a. By contrast, the problem considered here is extended to find $2N$ thresholds $t_{\mathrm{l}}^{(1)} < t_{\mathrm{u}}^{(1)} < t_{\mathrm{l}}^{(2)} < \cdots < t_{\mathrm{u}}^{(N)}$ to achieve the same objective (Fig. 2b). This approach may provide better thresholding results compared to the original one, but increasing the complexity of the multi-threshold searching process. Moreover, when we set $t_{\mathrm{l}}^{(1)} = 0, t_{\mathrm{l}}^{(2)} = t_{\mathrm{u}}^{(1)} + 1, \ldots, t_{\mathrm{l}}^{(N)} = t_{\mathrm{u}}^{(N-1)} + 1$, and $t_{\mathrm{u}}^{(N)} = L$, the problem reverts to the typical one as $t_{\mathrm{u}}^{(n)} = t^{(n)}$ for $n = 1 \sim N-1$, and thus it can be considered as an extension of the typical multilevel thresholding problem.

Remark 2 Nowadays, RGB-D cameras (i.e., Microsoft Kinect) quickly become popular in the area of robotic vision as they can provide color and depth information of a scene simultaneously. A few research works already applied the RGB-D camera to study planar segmentation of rigid objects [21,22] or semantic segmentation of indoor scenes [23,24], and the visual servo system shown in Fig. 1 also can use the RGB-D camera to improve the robustness of object detection by segmenting depth images. Although depth image segmentation helps to understand geometrical shape of the object, it is unable to detect color information of the object without assistance of color image segmentation. Therefore, the proposed color thresholding method can work with a depth image segmentation method to realize a robust object detection algorithm when the system using a RGB-D camera as the sensing device.

3 The proposed algorithm

Figure 3 shows the flowchart of the proposed multilevel color thresholding algorithm, which consists of a learning process and a multi-threshold searching process. The former aims to

learn the color distribution of an input video sequence in HSV color space, and the latter searches for the optimal multiple thresholds to segment all color objects in the video based on a novel performance criterion. The design of the learning process is introduced in this section, and the proposed multi-threshold searching algorithm is presented in the next section.

3.1 Color-pixel extraction

As shown in Fig. 3, the first step in the learning process is to extract all color pixels from the input color image represented in HSV color space. To achieve this objective, a published color-extraction algorithm [19] was used to distinguish the color and non-color regions of the image efficiently. Because the hue value of pixels having low brightness or low saturation is numerically unstable [2], the color extraction algorithm first highlights all low-saturation or low-brightness pixels of the image such that

$$\alpha(x, y) = \frac{V_{\text{in}}(x, y)}{S_{\text{in}}(x, y) + \varepsilon} + \frac{1}{S_{\text{in}}(x, y)V_{\text{in}}(x, y) + \varepsilon}, \tag{2}$$

where ε is a small non-zero positive value to avoid dividing by zero. In Expression (2), the first term highlights the low-saturation pixels and the second term the low-brightness pixels of the image. Because Expression (2) results into a high-dynamic range image, a dynamic range compression process is used to produce a ratio-map image, denoted by $R(x, y)$, with respect to $\alpha(x, y)$ such that

$$R(x, y) = 255 \left[\frac{\alpha(x, y)}{1 + \alpha(x, y)} \right]^{\beta}, \tag{3}$$

where $R(x, y) \in [0,255]$ is a gray-scale image highlighting all non-color pixels in the original image, and β is a non-zero positive parameter satisfying $\beta \geq 1$ to control the contrast of the ratio-map image. Finally, the color and non-color object classes can be distinguished easily using a bi-level thresholding method (i.e., the conventional Otsu's bi-level thresholding method [5,6]) to threshold the ratio-map image obtained from (2) and (3).

Figure 4 shows an example of the color-pixel extraction process. Figure 4a and b show the original image and its cor-

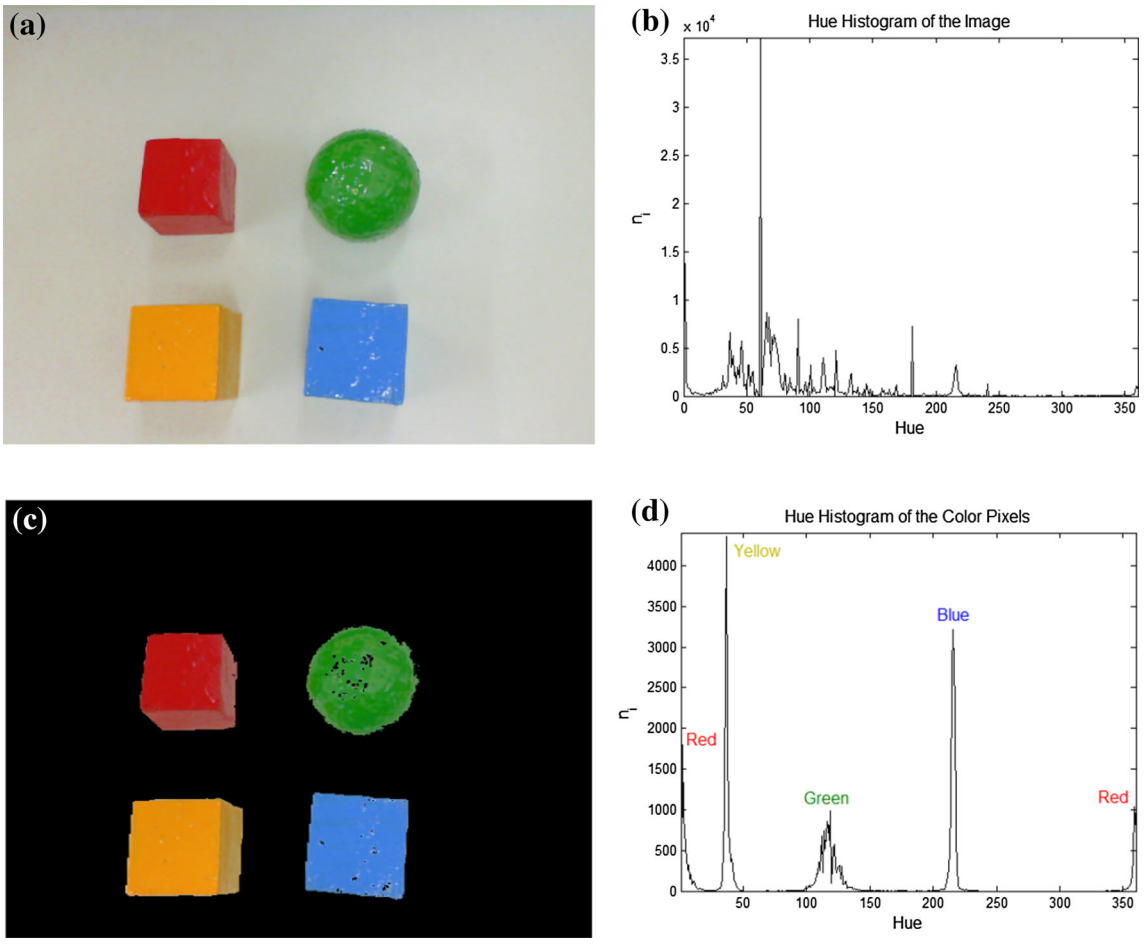

Fig. 4 Extraction of all color pixels in a color image: **a** original image and **b** its corresponding hue histogram; **c** color pixels of the original image and **d** the corresponding hue histogram of the color pixels

responding hue histogram, respectively. From Fig. 4b, one can see that it is difficult to distinguish color and non-color object classes in the image. Figure 4c shows color pixels of the original image extracted by the color extraction algorithm, and Fig. 4d illustrates the corresponding hue histogram of the color pixels. A visual comparison of Fig. 4b with 4d shows that Fig. 4d contains only hue information of the color pixels, which benefits the following multi-threshold searching process to determine the optimal color thresholds.

3.2 Estimating a color distribution model from the input video images

Suppose that all colors-of-interest (which can belong to static or moving objects) in the input video sequence are kept the same during the color thresholding process. Then it is possible to estimate a color distribution model of the video from the hue, saturation, and brightness histograms of the color information extracted from the video images by the color extraction algorithm. Let $\mathbf{H}_{\text{hist}}^{(k)} \in \Re^{1\times 361}$, $\mathbf{S}_{\text{hist}}^{(k)} \in \Re^{1\times 256}$, and $\mathbf{V}_{\text{hist}}^{(k)} \in \Re^{1\times 256}$ denote the hue, saturation, and brightness histograms, respectively, of the color information extracted from the kth video image. A simple method to estimate the color distribution model of the video is using the average of the extracted color histograms (hue, saturation, and brightness) in a subset of the input video images. That is, with a positive constant $M \geq 1$, the first M video images are used as the training data to compute the average color histograms, denoted by $\bar{\boldsymbol{H}}_{\text{hist}} \in \Re^{1\times 361}$, $\bar{\boldsymbol{S}}_{\text{hist}} \in \Re^{1\times 256}$, and $\bar{\boldsymbol{V}}_{\text{hist}} \in \Re^{1\times 256}$, such that

$$\begin{aligned}
\bar{\boldsymbol{H}}_{\text{hist}}(i) &= \frac{1}{M}\sum_{k=1}^{M}\mathbf{H}_{\text{hist}}^{(k)}(i),\\
\bar{\boldsymbol{S}}_{\text{hist}}(j) &= \frac{1}{M}\sum_{k=1}^{M}\mathbf{S}_{\text{hist}}^{(k)}(j),\\
\bar{\boldsymbol{V}}_{\text{hist}}(j) &= \frac{1}{M}\sum_{k=1}^{M}\mathbf{V}_{\text{hist}}^{(k)}(j),
\end{aligned} \tag{4}$$

where i and j are two integer variables ranging from 0 to 360 and from 0 to 255, respectively. Subsequently, the optimal threshold pairs can be found according to the three histograms $\bar{\boldsymbol{H}}_{\text{hist}}$, $\bar{\boldsymbol{S}}_{\text{hist}}$, and $\bar{\boldsymbol{V}}_{\text{hist}}$ for thresholding colors-of-interest in the video because they represent the expected probability of occurrence of each color-of-interest in the input video images.

Note that, to stop the learning process, we first count the number of input video images. When the value of the counter is larger than the constant M, the computation of the average color histograms is terminated and a stop flag is set. When the multi-threshold searching process is finished, the program switches to the segment stage and begins to threshold the following video images in real time.

4 Search for multiple thresholds with a novel within-class variance criterion

This section presents the proposed multi-threshold searching algorithm and its acceleration design based on a novel performance criterion. As shown in Fig. 2, the multi-threshold searching process consists of three steps. First, a peak-point detection algorithm is applied on the average hue histogram $\bar{\boldsymbol{H}}_{\text{hist}}$ to find the position of peak points in the histogram. Second, the proposed multi-threshold searching algorithm is used to determine lower bound and upper bound thresholds for each peak point in $\bar{\boldsymbol{H}}_{\text{hist}}$. Finally, a threshold-pair merging method [19] is used to find the optimal lower bound and upper bound thresholds of the hue channel, which can threshold all colors-of-interest in the video.

4.1 Peak-point detection in the average hue histogram

To detect the peak points of the average hue histogram, a low-pass filtering operation is first performed on the average hue histogram because the detection of peak points in a histogram is very sensitive to noise (Fig. 5a). Let $\bar{\boldsymbol{H}} \in \Re^{1\times 361}$ denote the smoothed average hue histogram. Then, the peak points in the histogram $\bar{\boldsymbol{H}}$ can be detected such that

$$\begin{aligned}
\mathbf{P} = \{i \in \aleph | &\bar{\boldsymbol{H}}(i) - \bar{\boldsymbol{H}}(i-2) > 0, \bar{\boldsymbol{H}}(i) - \bar{\boldsymbol{H}}(i-1) > 0,\\
&\bar{\boldsymbol{H}}(i) - \bar{\boldsymbol{H}}(i+1) > 0, \bar{\boldsymbol{H}}(i) - \bar{\boldsymbol{H}}(i+2) > 0,\\
&1 \leq i \leq 360\}, \text{ with } \bar{\boldsymbol{H}}(x)\big|_{\substack{x<0\\ x>360}} = 0,
\end{aligned} \tag{5}$$

where $\mathbf{P} = \{h_{\text{p}}^{(1)}, h_{\text{p}}^{(2)}, \ldots, h_{\text{p}}^{(m)}\}$ with $h_{\text{p}}^{(1)} < h_{\text{p}}^{(2)} < \cdots < h_{\text{p}}^{(m)}$ is the peak-point set recording the position of each peak point in the histogram, m is the number of detected peak points, and i is an integer index ranging from 1 to 360. Because an object class in the histogram usually has at least one peak point, the value of m usually satisfies $m \geq N$ in the case of N object classes. For example, Fig. 5b shows the smoothed average hue histogram and the corresponding peak-point detection result obtained by the proposed method. The average hue histogram has four classes ($N = 4$), but the proposed method detected six peak points ($m = 6$) in this case. Note that, the detected peak-point set $\mathbf{P}$ can be used as the initial positions in the next multi-threshold searching operation to assist in determining the optimal lower bound and upper bound thresholds for each object class.

Remark 3 Figure 5b shows a potential problem that a class mode may contain multiple peak points, each of them having its own lower bound and upper bound thresholds. In general,

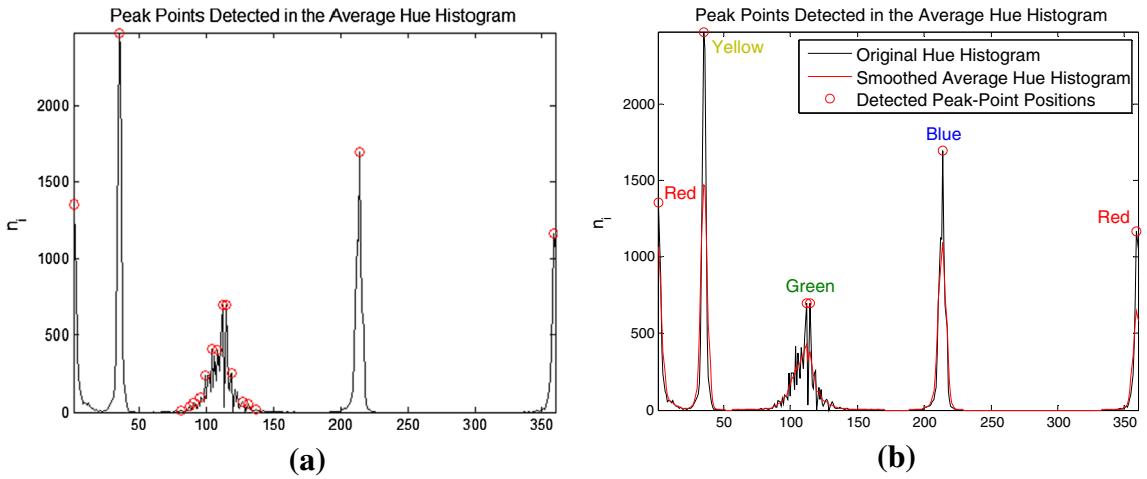

Fig. 5 Detection of peak points in the average hue histogram **a** without the low-pass filtering operation and **b** with the low-pass filtering operation. In this case, the number of classes is $N = 4$, and the number of detected peak points is $m = 6$

the thresholding ranges bounded by these thresholds are adjacent to each other. For example, the two peak points of the green-color class shown in Fig. 5b are detected at $h_{\mathrm{p}}^{(1)} = 112$ and $h_{\mathrm{p}}^{(2)} = 115$ using Expression (5). Then, the optimal lower bound and upper bound thresholds of each peak point can be found as $(\hat{t}_{\mathrm{l}}^{(1)}, \hat{t}_{\mathrm{u}}^{(1)}) = (67,113)$ and $(\hat{t}_{\mathrm{l}}^{(2)}, \hat{t}_{\mathrm{u}}^{(2)}) = (114,149)$ using the proposed multi-threshold searching method. Obviously, the two thresholds $\hat{t}_{\mathrm{u}}^{(1)}$ and $\hat{t}_{\mathrm{l}}^{(2)}$ are adjacent to each other and can be seen as two redundant thresholds since they can be removed to form a new optimal threshold pair $(\hat{t}_{\mathrm{l}}^{(1)}, \hat{t}_{\mathrm{u}}^{(2)}) = (67,149)$, producing the same thresholding range for the green-color class. We thus term this process as threshold-pair merging process, which removes all redundant thresholds and maintains the same thresholding ranges. Refer to [19] for more technical details of the threshold-pair merging algorithm.

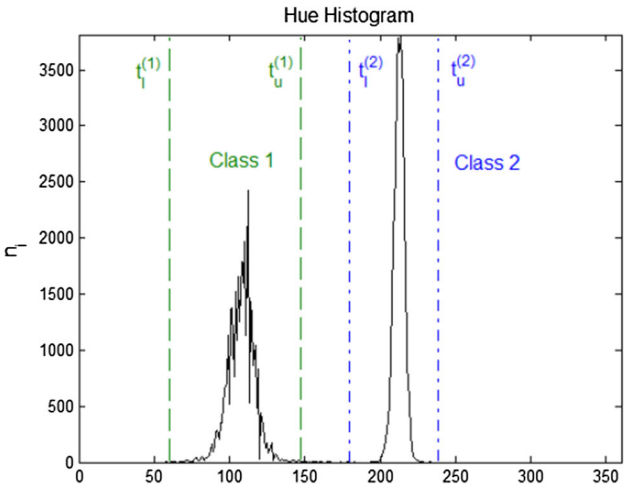

Fig. 6 Thresholding of two classes ($N = 2$) using two threshold pairs $(t_{\mathrm{l}}^{(1)}, t_{\mathrm{u}}^{(1)})$ and $(t_{\mathrm{l}}^{(2)}, t_{\mathrm{u}}^{(2)})$

4.2 Evaluation of multilevel thresholding using an extended within-class variance criterion

To search the optimal thresholds, it is crucial to define a performance criterion to quantitatively evaluate the thresholding result with respect to multiple threshold pairs. This subsection presents a novel performance criterion to achieve this objective. Consider the case of $N = 2$ (thresholding of two classes) as shown in Fig. 6. In this case, the objective was to find two threshold pairs $(t_{\mathrm{l}}^{(1)}, t_{\mathrm{u}}^{(1)})$ and $(t_{\mathrm{l}}^{(2)}, t_{\mathrm{u}}^{(2)})$ for thresholding Class 1 and Class 2, respectively. The proposed method is inspired from the Otsu's method [5] but works with a different class-variance criterion. Let N_{p} denote the number of total pixels in the image, L the maximum gray-level value of all pixels, and n_i the number of pixels with gray level i. Given the pixel-value probability $p_i = n_i/N_{\mathrm{p}}$ for $i = 1 \sim L$. In the case of bi-level thresholding, Otsu defined a within-class variance criterion with respect to a threshold value t such that [5]

$$\sigma_w^2(t) = \omega_1(t)\sigma_1^2(t) + \omega_2(t)\sigma_2^2(t), \tag{6}$$

where $\omega_1(t) = \sum_{i=1}^{t} p_i$ and $\omega_2(t) = \sum_{i=t+1}^{L} p_i$ are the probabilities of class occurrence; $\sigma_1^2(t) = \sum_{i=1}^{t}[i-\mu_1(t)]^2 p_i/\omega_1(t)$ and $\sigma_2^2(t) = \sum_{i=t+1}^{L}[i-\mu_2(t)]^2 p_i/\omega_2(t)$ are the class variance, where $\mu_1(t) = \sum_{i=1}^{t} i p_i/\omega_1(t)$ and $\mu_2(t) = \sum_{i=t+1}^{L} i p_i/\omega_2(t)$ are the class mean levels. However, the within-class variance Criterion (6) cannot be used in our case because it considers only a single threshold value for bi-level thresholding. To address this problem, Criterion (6) was modified into a multi-threshold form such that

$$\bar{\sigma}_w^2(t_1^{(1)}, t_u^{(1)}, t_1^{(2)}, t_u^{(2)}) = f_c(t_1^{(1)}, t_u^{(1)}) \sum_{i=t_1^{(1)}}^{t_u^{(1)}} [i - \bar{\mu}_1(t_1^{(1)}, t_u^{(1)})]^2 p_i + f_c(t_1^{(2)}, t_u^{(2)}) \sum_{i=t_1^{(2)}}^{t_u^{(2)}} [i - \bar{\mu}_2(t_1^{(2)}, t_u^{(2)})]^2 p_i, \tag{7}$$

where $\bar{\mu}_1(t_1^{(1)}, t_u^{(1)}) = \sum_{i=t_1^{(1)}}^{t_u^{(1)}} i p_i / \bar{\omega}_1(t_1^{(1)}, t_u^{(1)})$ and $\bar{\mu}_2(t_1^{(2)}, t_u^{(2)}) = \sum_{i=t_1^{(2)}}^{t_u^{(2)}} i p_i / \bar{\omega}_2(t_1^{(2)}, t_u^{(2)})$ are the within-class mean levels; $\bar{\omega}_1(t_1^{(1)}, t_u^{(1)}) = \sum_{i=t_1^{(1)}}^{t_u^{(1)}} p_i$ and $\bar{\omega}_2(t_1^{(2)}, t_u^{(2)}) = \sum_{i=t_1^{(2)}}^{t_u^{(2)}} p_i$ are the probabilities of within-class occurrence; and $f_c(t_l, t_u)$ is a class-concentration factor defined as

$$f_c(t_l, t_u) = \begin{cases} 1, & \text{for } \frac{N_{nz}(t_l, t_u)}{t_u - t_l + 1} \geq \gamma \\ 0, & \text{otherwise,} \end{cases}$$

where $N_{nz}(t_l, t_u)$ denotes the number of non-zero bins from t_lth to t_uth bins of the histogram, and $\gamma \leq 1$ is a non-negative parameter describing the desired concentration rate of a class within the range of $[t_l, t_u]$. Note that, Criterion (7) is regarded as the extended within-class variance criterion because Expressions (6) and (7) are equivalent when $t_1^{(1)} = 1, t_u^{(1)} = t, t_1^{(2)} = t + 1, t_u^{(2)} = L$, and $\gamma = 0$. Consequently, the optimal thresholds $\{\hat{t}_1^{(1)}, \hat{t}_u^{(1)}, \hat{t}_1^{(2)}, \hat{t}_u^{(2)}\}$ are obtained by maximizing Criterion (7) such that

$$\{\hat{t}_1^{(1)}, \hat{t}_u^{(1)}, \hat{t}_1^{(2)}, \hat{t}_u^{(2)}\} = \underset{1 \leq t_1^{(1)} < t_u^{(1)} < t_1^{(2)} < t_u^{(2)} \leq L}{\arg\max} \bar{\sigma}_w^2(t_1^{(1)}, t_u^{(1)}, t_1^{(2)}, t_u^{(2)}), \tag{8}$$

where the optimal thresholds $\{\hat{t}_1^{(1)}, \hat{t}_u^{(1)}, \hat{t}_1^{(2)}, \hat{t}_u^{(2)}\}$ can be found using a simple, but inefficient, exhaustive search method.

In the case of $N > 2$ (multilevel thresholding), Criterion (7) can be easily extended with respect to m threshold pairs $\{(t_1^{(1)}, t_u^{(1)}), \ldots, (t_1^{(m)}, t_u^{(m)})\}$ satisfying $t_1^{(1)} < t_u^{(1)} < \cdots < t_1^{(m)} < t_u^{(m)}$ such that

$$\bar{\sigma}_w^2(t_1^{(1)}, t_u^{(1)}, \ldots, t_1^{(m)}, t_u^{(m)}) = \sum_{j=1}^{m} \left\{ f_c(t_1^{(j)}, t_u^{(j)}) \times \sum_{i=t_1^{(j)}}^{t_u^{(j)}} [i - \bar{\mu}_j(t_1^{(j)}, t_u^{(j)})]^2 p_i \right\}, \tag{9}$$

where $m \geq N$ was explained previously, and $\bar{\mu}_j(t_1^{(j)}, t_u^{(j)})$ is the jth within-class mean level defined in (7). The optimal threshold pairs $\{(\hat{t}_1^{(1)}, \hat{t}_u^{(1)}), \ldots, (\hat{t}_1^{(m)}, \hat{t}_u^{(m)})\}$ are determined by maximizing Criterion (9) so that

$$\{\hat{t}_1^{(1)}, \hat{t}_u^{(1)}, \ldots, \hat{t}_1^{(m)}, \hat{t}_u^{(m)}\} = \underset{1 \leq t_1^{(1)} < t_u^{(1)} < \cdots < t_1^{(m)} < t_u^{(m)} \leq L}{\arg\max} \bar{\sigma}_w^2(t_1^{(1)}, t_u^{(1)}, \ldots, t_1^{(m)}, t_u^{(m)}). \tag{10}$$

Expression (10) is a general multi-threshold searching method, but the computational cost becomes expensive as the number of thresholds increases. To reduce the computational load and find an optimal solution, a vital observation is that the position of each peak point is located within the range between the corresponding lower bound and upper bound thresholds. More specifically, let $t_p^{(1)} < t_p^{(2)} < \cdots < t_p^{(m)}$ denote the position of m peak points in the histogram, then we have the conditions $t_1^{(j)} < t_p^{(j)} < t_u^{(j)}$ for $j = 1 \sim m$. Next, let $\bar{\sigma}_{wj}^2(t_1^{(j)}, t_u^{(j)}) = \sum_{i=t_1^{(j)}}^{t_u^{(j)}} [i - \bar{\mu}_j(t_1^{(j)}, t_u^{(j)})]^2 p_i$ denote the within-class variance of jth class; Expression (9) can be rewritten as

$$\bar{\sigma}_w^2(t_1^{(1)}, t_u^{(1)}, \ldots, t_1^{(m)}, t_u^{(m)}) = \sum_{j=1}^{m} f_c(t_1^{(j)}, t_u^{(j)}) \bar{\sigma}_{wj}^2(t_1^{(j)}, t_u^{(j)}). \tag{11}$$

These observations imply that $\bar{\sigma}_w^2(t_1^{(1)}, t_u^{(1)}, \ldots, t_1^{(m)}, t_u^{(m)})$ is maximized by independently maximizing each term in the sum. Therefore, the proposed multi-threshold searching algorithm divides the optimization problem (10) into several optimization sub-problems and resolves each sub-problem successively such that

$$\{\hat{t}_1^{(j)}, \hat{t}_u^{(j)}\} = \underset{t_u^{(j-1)} < t_1^{(j)} < t_p^{(j)} < t_u^{(j)} < t_1^{(j+1)}}{\arg\max} \left\{ f_c(t_1^{(j)}, t_u^{(j)}) \times \bar{\sigma}_{wj}^2(t_1^{(j)}, t_u^{(j)}) \right\} \quad \text{for } j = 1 \sim m, \tag{12}$$

where $t_u^{(0)} = 0$ and $t_1^{(m+1)} = L + 1$. By doing so, the number of computations of the multi-threshold searching process can be reduced dramatically as the dimension of each sub-problem is reduced to two dimensions. Note that, the initial values of the lower bound and upper bound thresholds are set according to the detected peak-point set $\mathbf{P}$ such that $(t_1^{(j)}, t_u^{(j)}) = (h_p^{(j)} - 1, h_p^{(j)} + 1)$, for $j = 1 \sim m$, to guarantee that the optimum upper bound threshold for jth peak point does not take a value which is greater than the lower bound threshold for the $(j + 1)$th peak point.

The proposed multi-threshold searching algorithm can be applied for multilevel thresholding of gray and color images (see Remark 4). For the multilevel color-thresholding problem considered in this study, the proposed algorithm can be applied directly to the hue histogram of the color image without modification. Therefore, the proposed algo-

rithm was applied to the average hue histogram (as mentioned at the beginning of this section) to determine the optimal hue thresholds for an input video sequence. Finally, the optimal hue thresholds were merged using a threshold-pair merging process (see Remark 3). This helps to increase the computational efficiency of the following color segmentation process as the number of thresholding operations is minimized, and produces the same thresholding result.

Remark 4 As mentioned in Remark 1 and Criterion (7), the proposed algorithm can be applied to N-class multilevel gray-image thresholding ($N \geq 2$) when $\gamma = 0, t_{\mathrm{l}}^{(1)} = 0, t_{\mathrm{l}}^{(n)} = t_{\mathrm{u}}^{(n-1)} + 1$, and $t_{\mathrm{u}}^{(N)} = L$ for $n = 2 \sim N$. The optimal thresholds $\{\hat{t}_{\mathrm{u}}^{(1)}, \ldots, \hat{t}_{\mathrm{u}}^{(N-1)}\}$ are subsequently determined by minimizing Criterion (9) with these conditions such that

$$\{\hat{t}_{\mathrm{u}}^{(1)}, \ldots, \hat{t}_{\mathrm{u}}^{(N-1)}\} = \underset{0<t_{\mathrm{u}}^{(1)}<\cdots<t_{\mathrm{u}}^{(N-1)}<L}{\arg\min} \bar{\sigma}_w^2(0, t_{\mathrm{u}}^{(1)}, t_{\mathrm{u}}^{(1)} + 1, \ldots, t_{\mathrm{u}}^{(N-1)}, t_{\mathrm{u}}^{(N-1)} + 1, L)\Big|_{\gamma=0}, \tag{13}$$

which is equivalent to the traditional multilevel Otsu's method because maximizing the between-class variance is equal to minimizing the within-class variance [5].

Remark 5 According to Expressions (10) and (12), both proposed algorithms frequently evaluate the extended within-class variance of the jth class $f_{\mathrm{c}}(t_{\mathrm{l}}^{(j)}, t_{\mathrm{u}}^{(j)})\bar{\sigma}_{wj}^2(t_{\mathrm{l}}^{(j)}, t_{\mathrm{u}}^{(j)})$ during the multi-threshold searching process. An efficient approach to reduce the computational cost is to pre-compute an (L-1)-by-(L-1) two-dimensional lookup table (2D-LUT) to record the value of $f_{\mathrm{c}}(t_{\mathrm{l}}, t_{\mathrm{u}})\bar{\sigma}_{wj}^2(t_{\mathrm{l}}, t_{\mathrm{u}})$ for all threshold combinations satisfying $t_{\mathrm{l}} < t_{\mathrm{u}}$ with $t_{\mathrm{l}} \in [1, L-1]$ and $t_{\mathrm{u}} \in [2, L]$. Thus, the computation of within-class variance evaluation is reduced to a 2D-LUT indexing operation only, drastically speeding up the searching process of both proposed algorithms.

4.3 Determination of the saturation and brightness thresholds

To determine the saturation and brightness thresholds, the cumulative distribution function (CDF) of the average saturation and brightness histograms were used to find the lower bound thresholds $\hat{s}_{\mathrm{l}}$ and $\hat{v}_{\mathrm{l}}$, respectively. Figure 7 shows the CDF of a histogram with level value $i = 1 \sim L$. Given a small nonzero positive parameter δ, a lower bound threshold $t_{\mathrm{l}} \in [1, L)$ is defined as the smallest integer satisfying the following condition

$$\sum_{i=1}^{t_{\mathrm{l}}} p_i \geq \delta, \tag{14}$$

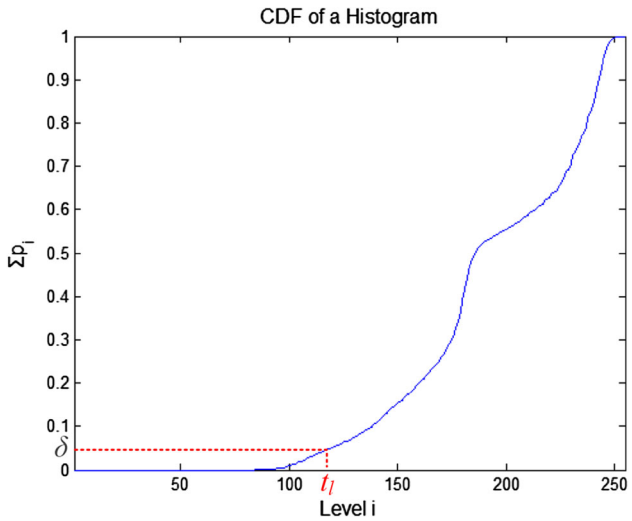

Fig. 7 CDF of a histogram with level value $i = 1 \sim L$

where p_i is, again, the probability of level value i. Based on this concept, the lower bound thresholds of saturation and brightness, $\hat{s}_{\mathrm{l}} \in [0, 1)$ and $\hat{v}_{\mathrm{l}} \in [0, 1)$, are calculated by evaluating a constrained L1-distance criterion related to the CDF of the average saturation and brightness histograms, respectively, such that

$$\hat{s}_{\mathrm{l}} = \frac{1}{L}\left\{\underset{t_{\mathrm{l}}\geq 1}{\arg\min}\left|\sum_{i=1}^{t_{\mathrm{l}}} \bar{p}_{i/L}^{s} - \delta\right|_{\sum_{i=1}^{t_{\mathrm{l}}} \bar{p}_{i/L}^{s} - \delta \geq 0}\right\}$$

$$\text{and } \hat{v}_{\mathrm{l}} = \frac{1}{L}\left\{\underset{t_{\mathrm{l}}\geq 1}{\arg\min}\left|\sum_{i=1}^{t_{\mathrm{l}}} \bar{p}_{i/L}^{v} - \delta\right|_{\sum_{i=1}^{t_{\mathrm{l}}} \bar{p}_{i/L}^{v} - \delta \geq 0}\right\}, \tag{15}$$

where $|x|_{\Omega(x)}$ denotes the L1-distance of x subject to a constraint set $\Omega(x)$; $\bar{p}_{i/L}^{s}$ and $\bar{p}_{i/L}^{v}$ denote the probability of level value i/L of the average saturation and brightness histograms, respectively. Finally, the lower bound and upper bound thresholds of saturation and brightness channels for all object classes were set to $(\hat{s}_{\mathrm{l}}^{(n)}, \hat{s}_{\mathrm{u}}^{(n)}) = (\hat{s}_{\mathrm{l}}, 1)$ and $(\hat{v}_{\mathrm{l}}^{(n)}, \hat{v}_{\mathrm{u}}^{(n)}) = (\hat{v}_{\mathrm{l}}, 1)$, respectively, with $n = 1 \sim N$ in our design.

5 Experimental results

To evaluate the performance of the proposed algorithm, it was compared with three current methods: the two-stage multi-threshold Otsu (TSMO) method [7], the cooperative quantum-behaved particle swarm optimization (CQPSO) method [8], and the histogram-based multi-threshold searching (HMTS) method [19]. Table 1 tabulates the parameter settings of the proposed method and compared methods used in

Table 1 Parameter settings of the competing and proposed methods

Method	Symbol	Values	Description
TSMO method [7]	Class	5	Class number
	Group	32	Group number
CQPSO method [8]	Num	2,000	Particle number
	Dim	4	Dimension of each particle
	Iter	500	Maximum iteration number
HMTS method [19]	ε	0.00390625	Small non-zero positive value in (2)
	β	1	Contrast control parameter in (3)
	λ	0.3	Scale factor of threshold initialization
Proposed method	ε	0.00390625	Small non-zero positive value in (2)
	β	1	Contrast control parameter in (3)
	γ	0.98	Desired concentration rate in (7)
	δ	0.0002	Nonzero positive parameter in (15)

the experiments. The following experiments contained three parts. First of all, we evaluated that Expression (12) produces the same result as Expression (10) but requires less computational time to find the optimal solution. Second, the performance between the proposed method and the TSMO, CQPSO methods was compared using still color images. Finally, the extension of the proposed method to color video segmentation was compared with the HMTS method. The real-time performance of the proposed method was evaluated in the final test.

5.1 Performance comparison between Expressions (12) and (10)

As stated in Sect. 4.2, Expression (10) is a general form for searching the optimal thresholds; however the computational cost becomes expensive as the number of thresholds increases. By contrast, Expression (12) provides a computationally efficient solution to speed up the search process because it reduces the search space dimension from $2m$ to 2. To evaluate the performance of Expression (12), two test patterns were used: one is two-class (Fig. 8a) and the other one is three-class (Fig. 8b). Figure 8c and d show the gray-level histogram of Fig. 8a and b, respectively. Expressions (10) and (12) were subsequently applied to both histograms to find the optimal thresholds for each pattern. These two expressions were implemented in C++ running on the same platform to provide a fair comparison. Table 2 records the value of the optimal thresholds and the corresponding processing times obtained using Expressions (10) and (12), including the cost of 2D-LUT building (see Remark 5). It is clear from Table 2 that Expressions (10) and (12) obtained the same optimal thresholds for both histograms; however, Expression (10) required a long processing time: 850.6 ms for the two-class histogram and 43 min for the three-class histogram. By contrast, Expression (12) was computationally more efficient because it required approximately 90 ms for the two-class histogram and 92 ms for the three-class histogram, greatly speeding up the multi-threshold searching process. These experimental results validate the performance of Expression (12); thus, it was used as the default multi-threshold searching algorithm for the proposed method in the following experiments.

5.2 Multilevel color thresholding for color image segmentation

In the second part of the experiments, two test images, as shown in Figs. 9a and 10a, were used to evaluate the performance of the proposed method. The TSMO and CQPSO methods were used for comparison with the proposed method. The mean square error (MSE) metric was used to quantify the performance of the proposed method. The MSE metric was defined as

$$\mathrm{MSE}(\mathbf{T}_\mathrm{c}, \mathbf{T}_\mathrm{u}) = \frac{1}{UV} \sum_{1 \le y \le U} \sum_{1 \le x \le V} \|\mathbf{T}_\mathrm{c}(x, y) - \mathbf{T}_\mathrm{u}(x, y)\|^2, \tag{16}$$

where U and V are the total row and column number of the image, respectively, $\mathbf{T}_\mathrm{c}$ is the color segmentation result obtained from the color-pixel extraction algorithm presented in Sect. 3.1, and $\mathbf{T}_\mathrm{u}$ is the corresponding result using an unsupervised multilevel thresholding method. Note that, the color-pixel image obtained by the color-pixel extraction algorithm was used as the ground truth in the experiments because each compared method used its hue, saturation, and brightness histograms to determine the optimal thresholds for thresholding the detected color information. After the learning process, these optimal thresholds can be used to segment the color content within the scene using a computationally efficient color thresholding method instead of the complicated color-pixel extraction algorithm to achieve real-time performance (see Sect. 5.3).

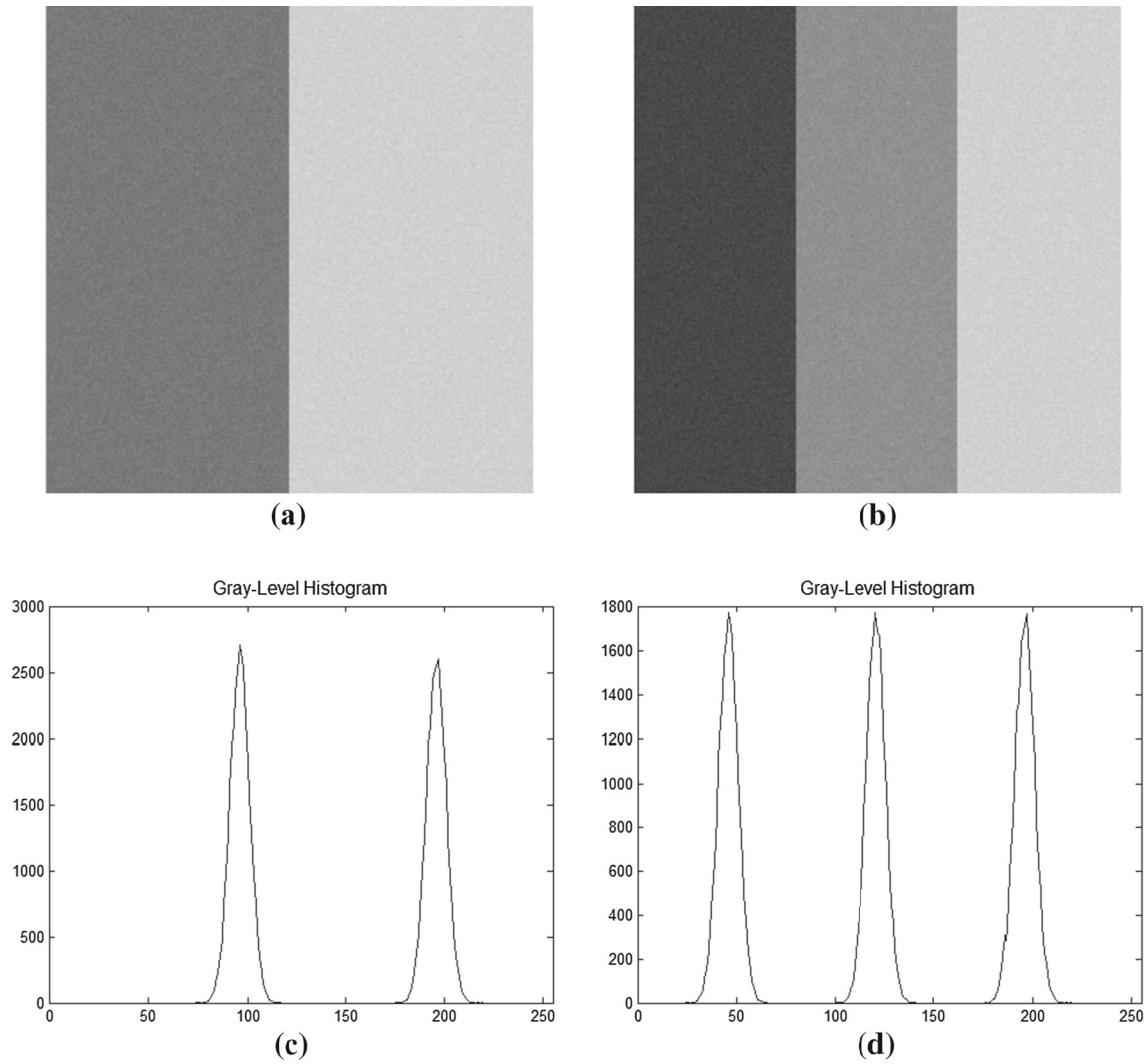

Fig. 8 Test patterns used in the performance comparison between Expressions (12) and (10): **a** two-class test pattern, **b** three-class test pattern, **c** gray-level histogram of (**a**), and **d** gray-level histogram of (**b**)

Table 2 Optimal thresholds obtained by Expressions (10) and (12) with their processing times (in ms)

Method	Figure	n	$\hat{t}_{\mathrm{l}}^{(n)}$	$\hat{t}_{\mathrm{u}}^{(n)}$	Processing time (ms)
Expression (10)	Fig. 8c	1	78	116	850.6173
		2	176	215	
	Fig. 8d	1	28	65	2,606,552.5241
		2	101	141	
		3	179	215	
Expression (12)	Fig. 8c	1	78	116	90.2299
		2	176	215	
	Fig. 8d	1	28	65	92.3880
		2	101	141	
		3	179	215	

Figure 9b shows the color segmentation result obtained by extracting all color pixels in the input color image using the color-pixel extraction algorithm. Figure 9c and d present the multiple thresholds of the hue channel and the multilevel thresholding result obtained from the TSMO method, respectively, and Fig. 9e and f the corresponding results obtained from the CQPSO method. These four figures show that the TSMO and CQPSO methods cannot provide satisfactory segmentation results because the multiple thresholds cannot threshold the two color classes in the hue histogram effectively. This is also shown in Table 3, which records the value of multiple thresholds obtained by each competing method and its corresponding MSE measures. Table 3 shows that the MSE metric between Fig. 9b and d is 1,485.9000 and between Fig. 9b and f is 2,469.2000. This implies a low similarity

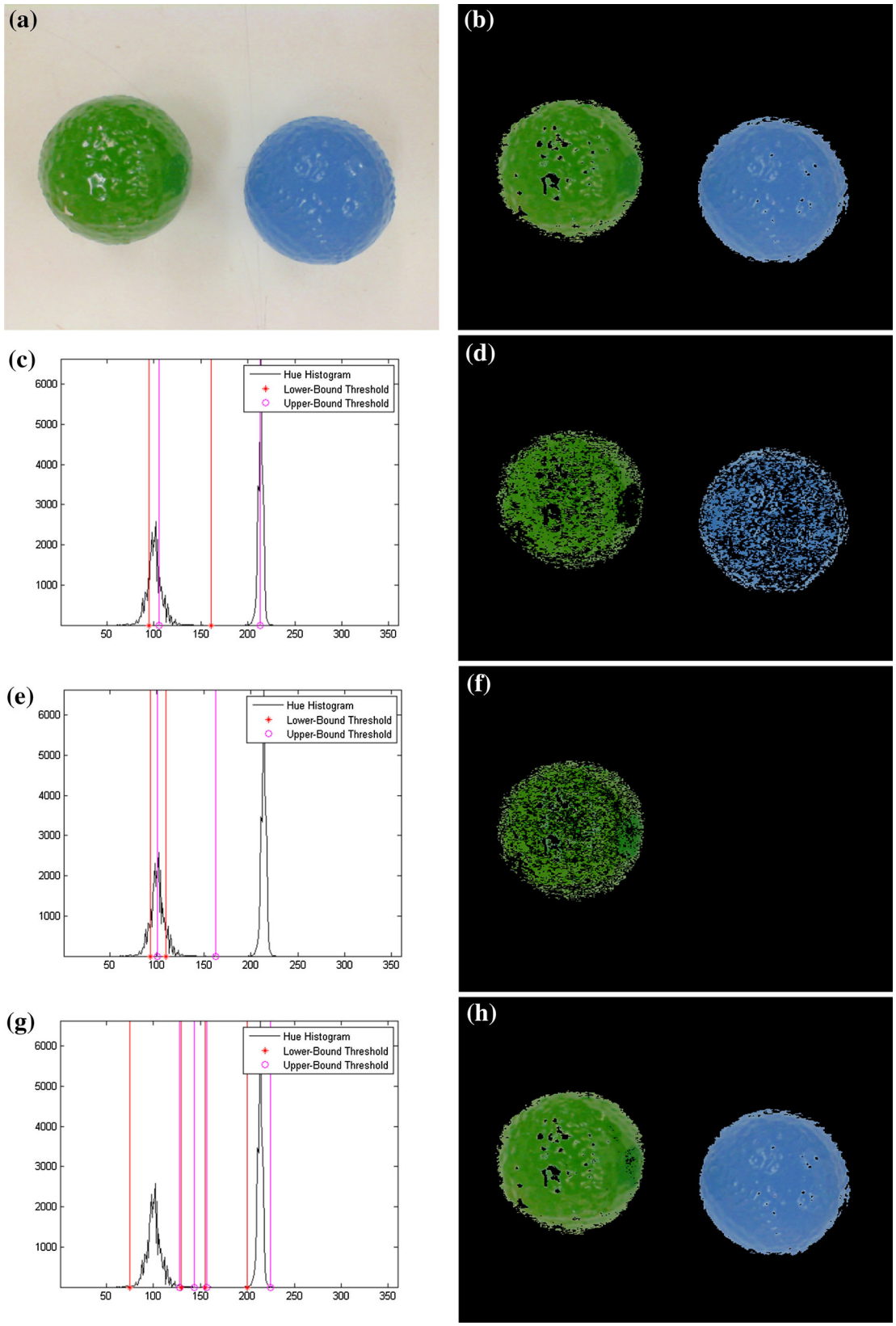

Fig. 9 Experimental results of multilevel color thresholding process: **a** original image, **b** color-pixel image obtained by the color-pixel extraction algorithm; color thresholding **c** by the TSMO method and **d** its thresholding result, **e** by the CQPSO method and **f** its thresholding result, **g** by the proposed method and **h** its thresholding result

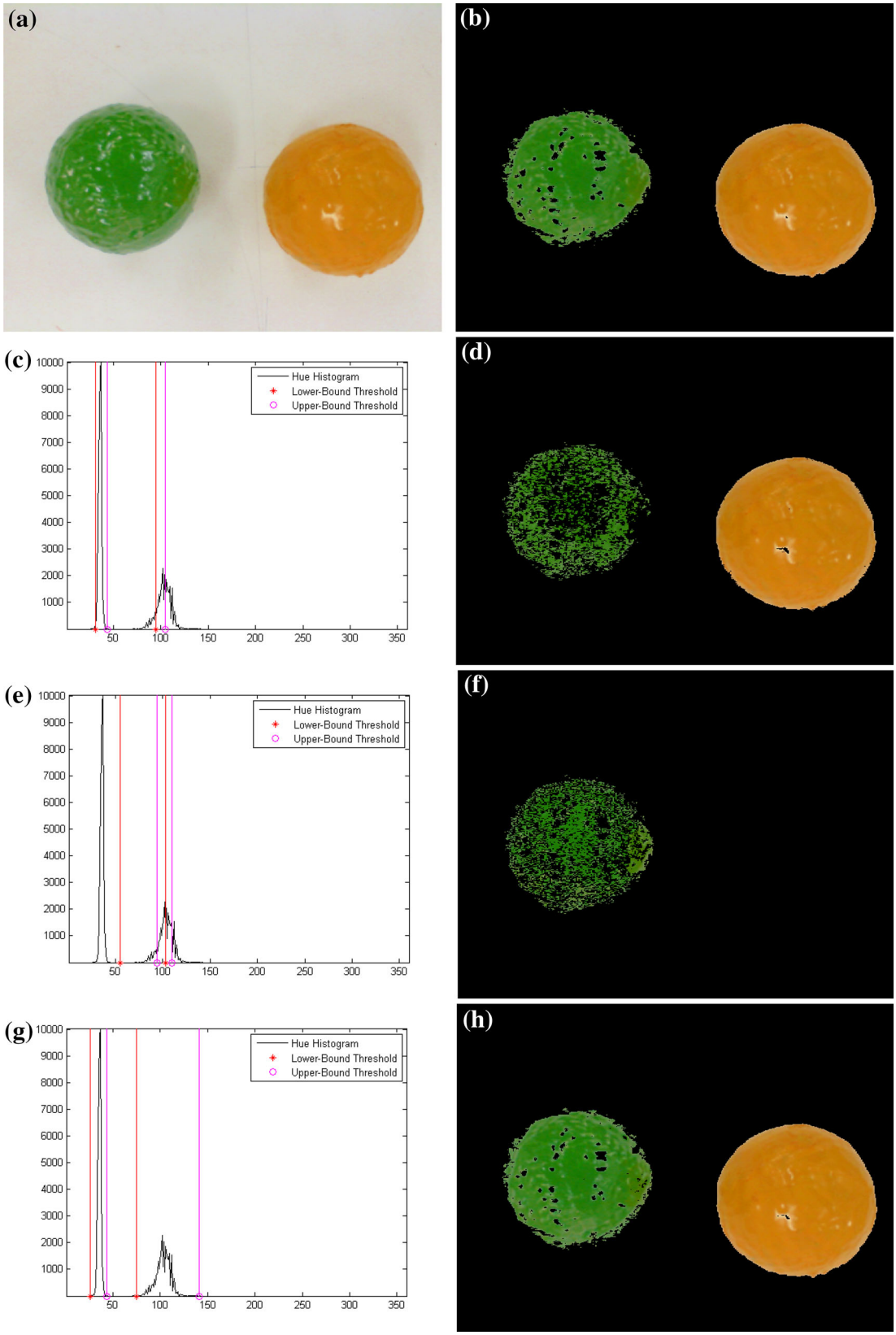

Fig. 10 Experimental results of multilevel color thresholding process: **a** original image, **b** color-pixel image obtained by the color-pixel extraction algorithm; color thresholding **c** by the TSMO method and **d** its thresholding result, **e** by the CQPSO method and **f** its thresholding result, **g** by the proposed method and **h** our thresholding result

Table 3 Optimal thresholds obtained by each method and the MSE measures

Method	Figure	n	$\hat{h}_{\mathrm{l}}^{(n)}$	$\hat{h}_{\mathrm{u}}^{(n)}$	$\hat{s}_{\mathrm{l}}^{(n)}$	$\hat{s}_{\mathrm{u}}^{(n)}$	$\hat{v}_{\mathrm{l}}^{(n)}$	$\hat{v}_{\mathrm{u}}^{(n)}$	MSE
TSMO method [7]	Fig. 9d	1	95	106	0.3203	1.0	0.4297	1.0	1,485.9000
		2	161	213					
	Fig. 10d	1	31	44	0.3984	1.0	0.4492	1.0	434.5369
		2	95	105					
CQPSO method [8]	Fig. 9f	1	93	101	0.3203	1.0	0.4297	1.0	2,469.2000
		2	110	163					
	Fig. 10f	1	55	94	0.3984	1.0	0.4492	1.0	2,960.6000
		2	103	110					
Proposed method	Fig. 9h	1	75	128	0.3203	1.0	0.4297	1.0	63.3679
		2	130	144					
		3	155	157					
		4	200	226					
	Fig. 10h	1	26	44	0.3984	1.0	0.4492	1.0	77.8700
		2	75	142					

Table 4 Optimal thresholds obtained by each method and the average MSE measures of color video segmentation testing

Method	n	$\hat{h}_{\mathrm{l}}^{(n)}$	$\hat{h}_{\mathrm{u}}^{(n)}$	$\hat{s}_{\mathrm{l}}^{(n)}$	$\hat{s}_{\mathrm{u}}^{(n)}$	$\hat{v}_{\mathrm{l}}^{(n)}$	$\hat{v}_{\mathrm{u}}^{(n)}$	Average MSE
HMTS method [19]	1	1	57	0.2422	1.0	0.1250	1.0	482.0440
	2	70	124					
	3	176	239					
	4	321	360					
Proposed method	1	1	141	0.2656	1.0	0.1719	1.0	251.7890
	2	189	234					
	3	333	360					

between the results produced by the color-pixel extraction algorithm and the compared methods. By contrast, Fig. 9g illustrates the multiple thresholds of the hue channel obtained from the proposed method. From Fig. 9g, the multiple thresholds can effectively threshold the two color classes in the hue histogram (including those that are not continuously distributed), producing a low MSE metric of 63.3679, as shown in Table 3. Figure 9h shows the multilevel thresholding result obtained from the proposed method. A visual comparison of Fig. 9b and h observes that the proposed method produces a similar color segmentation result to the color-pixel extraction algorithm. These results explain why the proposed method produces a better color thresholding result than the compared methods. Similar results also can be observed for the case shown in Fig. 10a–h. Therefore, these experiments verified the performance of the proposed method. More experimental results are available online [25].

5.3 Multilevel color thresholding for real-time color video segmentation

In the final experiment, the proposed method was applied to a video clip containing 2,050 video images (or frames) to evaluate the performance of multilevel color thresholding in real-time video segmentation applications. In this experiment, the first 650 images of the video were used as the training data to calculate the average color histograms ($\bar{\boldsymbol{H}}_{\mathrm{hist}}$, $\bar{\boldsymbol{S}}_{\mathrm{hist}}$, and $\boldsymbol{V}_{\mathrm{hist}}$) defined in (4). Next, the proposed and current HMTS methods were applied to these histograms to find the optimal threshold values for each color channel. Finally, these two groups of threshold values were used to segment the remaining video images (1,400 images) and compute the average MSE measure between the color-thresholding result and the color-pixel image obtained from the color-pixel extraction algorithm. Table 4 tabulates the value of multiple thresholds obtained by each method and the corresponding average MSE measures. One can see from Table 4 that the proposed method provides more accurate threshold values with a smaller average MSE measure than the HMTS method. Figure 11 explains why the proposed method outperforms the HMTS method. Figure 11a and b shows one of the video images and its color pixels extracted by the color-pixel extraction algorithm, respectively. The multilevel color-thresholding results obtained from the competing and proposed methods are shown in Fig. 11c and d, respectively. Visually comparing Fig. 11c and d finds that the pro-

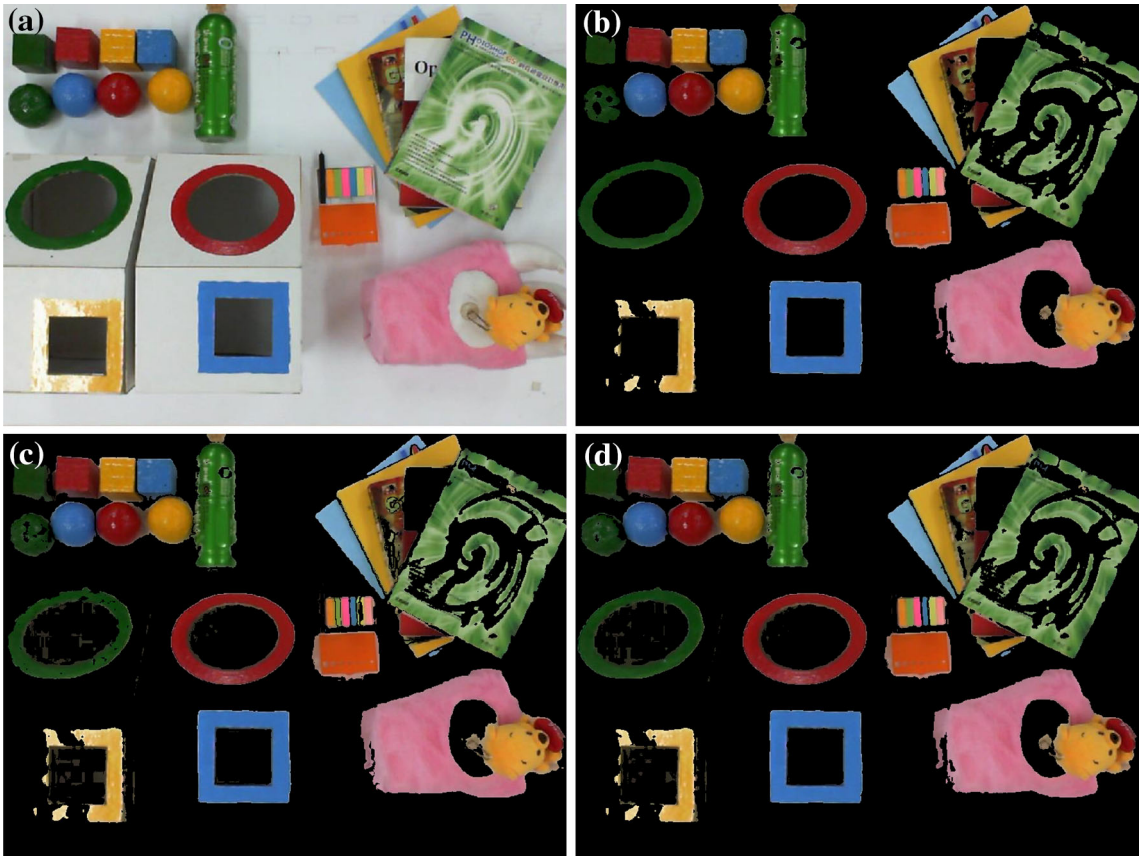

Fig. 11 Experimental results of multilevel color thresholding for color video segmentation: **a** one of the video images, **b** color-pixel image obtained from the color-pixel extraction algorithm, color thresholding result obtained from **c** the HMTS method and **d** the proposed method

Table 5 Average processing time per frame (in milliseconds) of the proposed method for color video segmentation with a video format of 640 × 480 RGB24

Stage	Process	Processing time (ms)	Total processing time (ms)
Learning stage	RGB-to-HSV conversion	3.1670	47.1683
	Color-pixel extraction	37.2165	
	Averaging color histograms	6.7848	
Multi-threshold searching[a]	Building look-up table	275.9530	276.1974
	Optimization and merging	0.2444	
Segmentation stage	RGB-to-HSV conversion	3.7427	5.9497
	Multilevel color thresholding	2.2070	

[a] Perform only once

posed method produced a similar color segmentation result to Fig. 11b, leading to a smaller MSE measure than that of the HMTS method. Therefore, these experimental results verified the color-thresholding performance of the proposed method.

For the evaluation of real-time performance, the proposed automatic multi-threshold searching algorithm was implemented in C++ running on a Windows XP machine with 3.3 GHz Intel Core 4 processor and 4 GB of memory. Table 5 records average processing time of each stage required for the proposed method to process a standard VGA (640 × 480 RGB24 format) video stream. As shown in Table 5, the processing time of the proposed method in the learning stage is 47.1683 ms per frame in average. The processing time of the proposed multi-threshold searching method is approximately 276.1974 ms; however, it performs only once for an input video stream. After the learning stage, the proposed method switches to the segmentation stage to perform a multilevel color-thresholding operation on the incoming video images. In this stage, the average processing time is reduced to 5.9497 ms per frame, including RGB-to-HSV color space conversion and multilevel color thresholding process. This achieves about 168 frames per second in real-time color video segmentation applications. The video clips from this experiment are available online [25].

6 Conclusions and future work

This paper presents a novel and efficient class-variance-based multi-threshold searching algorithm to achieve automatic multilevel color thresholding for segmentation of multiple colors-of-interest in color video images. The proposed method comprises a color-distribution learning algorithm combined with a color-pixel extraction algorithm to learn a color distribution model (three average color histograms) of the colors-of-interest in an input video sequence. This helps to simplify the search of optimal threshold values for thresholding the colors-of-interest in the video sequence using a conventional multilevel thresholding method. To search the optimal threshold values, a novel nonparametric multilevel color-thresholding algorithm operating with a novel class-variance criterion was developed. The proposed method automatically and efficiently finds the optimal upper bound and lower bound threshold values of hue, saturation and brightness channels for a given color distribution model of the colors-of-interest. The TSMO, CQPSO, and HMTS methods were compared visually and quantitatively with the proposed method, and the experimental results verified the performance of the proposed method.

It is worth noting that the proposed algorithm can be applied to both color image and color video thresholding applications. Moreover, the proposed algorithm can achieve real-time performance. These advantages greatly increase the practicality of the proposed method and make it suitable for integrating with a real-time robotic vision system. The extension to color texture segmentation will be addressed in a future study.

Acknowledgments This work was supported by the National Science Council of Taiwan, ROC under Grant NSC 102-2221-E-032-050.

References

1. Yang, A.Y., Wright, J., Ma, Y., Sastry, S.S.: Unsupervised segmentation of natural images via lossy data compression. Comput. Vis. Image Underst. **110**(2), 212–225 (2008)
2. Cheng, H.D., Jiang, X.H., Sun, Y., Wang, J.: Color image segmentation: advances and prospects. Pattern Recognit. **34**(12), 2259–2281 (2001)
3. Kin, W.S., Park, R.H.: Color image palette construction based on the HSI color system for minimizing the reconstruction error. In: IEEE International Conference on Image Processing, Lausanne, Switzerland, pp. 1041–1044 (1996)
4. Abutaleb, A.S.: Automatic thresholding of gray-level pictures using two-dimensional entropy. Comput. Vis. Graph. Image Process. **47**(1), 22–32 (1989)
5. Otsu, N.: A threshold selection method from gray-level histograms. IEEE Trans. Syst. Man Cybern. **9**(1), 62–66 (1979)
6. Liao, P.-S., Chen, T.-S., Chung, P.-C.: A fast algorithm for multilevel thresholding. J. Inf. Sci. Eng. **17**(5), 713–727 (2001)
7. Huang, D.-Y., Wang, C.-H.: Optimal multi-level thresholding using a two-stage Otsu optimization approach. Pattern Recognit. Lett. **30**(3), 275–284 (2009)
8. Gao, H., Xu, W., Sun, J., Tang, Y.: Multilevel thresholding for image segmentation through an improved quantum-behaved particle swarm algorithm. IEEE Trans. Instrum. Meas. **59**(4), 934–945 (2010)
9. Cheng, H.D., Sun, Y.: A hierarchical approach to color image segmentation using homogeneity. IEEE Trans. Image Process. **9**(12), 2071–2082 (2000)
10. Cheng, H.D., Jiang, X.H., Wang, J.: Color image segmentation based on homogram thresholding and region merging. Pattern Recognit. **35**(2), 373–393 (2002)
11. Wu, Y., Liu, Q., Huang, T.S.: An adaptive self-organizing color segmentation algorithm with application to robust real-time human hand localization. In: Proceedings of the Asian Conference on Computer Vision, Taiwan, pp. 1106–1111 (2000)
12. Haghighatdoost, V., Safabakhsh, R.: Automatic multilevel color image thresholding by the growing time adaptive self organizing map. In: 2nd IEEE International Conference on Information and Communication Technologies, Damascus, Syria, pp. 1768–1772 (2006)
13. Chaabane, S.B., Sayadi, M., Fnaiech, F., Brassart, E.: Color image segmentation using automatic thresholding and the fuzzy C-means techniques. In: 14th IEEE Mediterranean Electrotechnical Conference, Ajaccio, France, pp. 857–861 (2008)
14. Yu, Z., Au, O.C., Zou, R., Yu, W., Tian, J.: An adaptive unsupervised approach toward pixel clustering and color image segmentation. Pattern Recognit. **43**(5), 1889–1906 (2010)
15. Harrabi, R., Braiek, E.B.: Color image segmentation using multilevel thresholding approach and data fusion techniques: application in the breast cancer cells images. EURASIP J. Image Video Process. **2012**(11), 1–11 (2012)
16. Ben Chaabane, S., Fnaiech, F., Sayadi, M., Brassart, E.: Estimation of the mass function in the Dempster–Shafer's evidence theory using automatic thresholding for color image segmentation. In: 2nd International Conference on Signals, Circuits and Systems, Nabeul, Tunisia, pp. 1–5 (2008)
17. Du, Y., Chang, C.-I., Thouin, P.D.: Unsupervised approach to color video thresholding. Opt. Eng. **43**(2), 282–289 (2004)
18. Rotaru, C., Graf, T., Zhang, J.: Color image segmentation in HSI space for automotive applications. J. Real Time Image Process. **3**(4), 311–322 (2008)
19. Tsai, C.-Y., Liu, T.-Y., Chen, W.-C.: A novel histogram-based multi-threshold searching algorithm for multilevel color thresholding. Int. J. Adv. Robot. Syst. **9**(223), 1–13 (2012)
20. Gonzalez, R.C., Woods, R.E.: Digital Image Processing, 2nd edn. Prentice-Hall, New Jersey (2002)
21. Enjarini, B., Gräser, A.: Planar segmentation from depth images using gradient of depth feature. In: IEEE/RSJ International Conference on Intelligent Robots and Systems, Vilamoura, Portugal, pp. 4668–4674 (2012)
22. Erdogan, C., Paluri, M., Dellaert, F.: Planar segmentation of RGBD images using fast linear fitting and Markov Chain Monte Carlo. In: 9th Conference on Computer and Robot Vision, Toronto, Canada, pp. 32–39 (2012)
23. Ren, X., Bo, L., Fox, D.: RGB-(D) scene labeling: features and algorithms. In: IEEE Conference on Computer Vision and Pattern Recognition, Providence, USA, pp. 2759–2766 (2012)
24. Gupta, S., Arbeláez, P., Malik, J.: Perceptual organization and recognition of indoor scenes from RGB-D images. In: IEEE Conference on Computer Vision and Pattern Recognition, Portland, USA, pp. 564–571 (2013)
25. The experimental results website: [Online]. http://www.ee.tku.edu.tw/~RVLab/_Experiments/MultiThres/MultiThres_Results.htm

Chi-Yi Tsai received his B.S. and M.S. degree in electrical engineering from National Yunlin University of Science and Technology, Yunlin, Taiwan, in 2000 and 2002, respectively, and Ph.D. degree in electrical and control engineering from National Chiao Tung University, Hsinchu, Taiwan in 2008. In 2010, he joined the Department of Electrical Engineering, TamKang University, New Taipei City, Taiwan, where he is currently an Associate Professor. His research interests include image processing, color image enhancement processing, visual tracking control for mobile robots, visual servoing, and computer vision.

Tsung-Yen Liu received his B.S. degree in electrical engineering from Chunghua University, Hsinchu, Taiwan, in 2011 and M.S. degree in electrical engineering from Tamkang University, New Taipei City, Taiwan in 2013. He is currently a deputy engineer in the Flat Knitting Department of the R&D division, Pai-Lung Machinery Mill Corporation. His research interests include image processing and computer vision.