Video Decolorization Using Visual Proximity Coherence Optimization

Yizhang Tao, Yiyi Shen, Bin Sheng, Ping Li, and Rynson W. H. Lau

***Abstract*—Video decolorization is to filter out the color information while preserving the perceivable content in the video as much and correct as possible. Existing methods mainly apply image decolorization strategies on videos, which may be slow and produce incoherent results. In this paper, we propose a video decolorization framework that considers frame coherence and saves decolorization time by referring to the decolorized frames. It has three main contributions. First, we define decolorization proximity to measure the similarity of adjacent frames. Second, we propose three decolorization strategies for frames with low, medium, and high proximities, to preserve the quality of these three types of frames. Third, we propose a novel decolorization Gaussian mixture model to classify the frames and assign appropriate decolorization strategies to them based on their decolorization proximity. To evaluate our results, we measure them from three aspects: 1) qualitative; 2) quantitative; and 3) user study. We apply color contrast preserving ratio and C2G-SSIM to evaluate the quality of single frame decolorization. We propose a novel temporal coherence degree metric to evaluate the temporal coherence of the decolorized video. Compared with current methods, the proposed approach shows all around better performance in time efficiency, temporal coherence, and quality preservation.**

***Index Terms*—Decolorization Gaussian mixture model (DC-GMM) classifier, decolorization proximity, result reutilization, temporal coherence, video decolorization.**

I. Introduction

VIDEO decolorization is an important video editing tool with different applications. By filtering out the chroma information, grayscale videos can better demonstrate the object texture. It also makes it easier to detect object outlines, an important research area in machine learning. Further, grayscale videos require less memory to store by preserving only the most important information. However, most existing works mainly focus on image decolorization and reuse the same strategy for video frames. They neglect the nuance between an image and a video, which may lead to luminance incoherence, also called visual flickering, as shown in Fig. 1. As video frames are decolorized separately, current methods also show high redundancy in processing adjacent frames when their contents are similar. Such high computational cost and visual flickering problems are drawing more and more attention.

As video frames are a sequence of consecutive frames sharing similar contents, rather than a group of unrelated images, we propose in this paper a video decolorization framework that considers frame coherence. It has two main steps. First, it computes the decolorization proximity, measuring the similarity

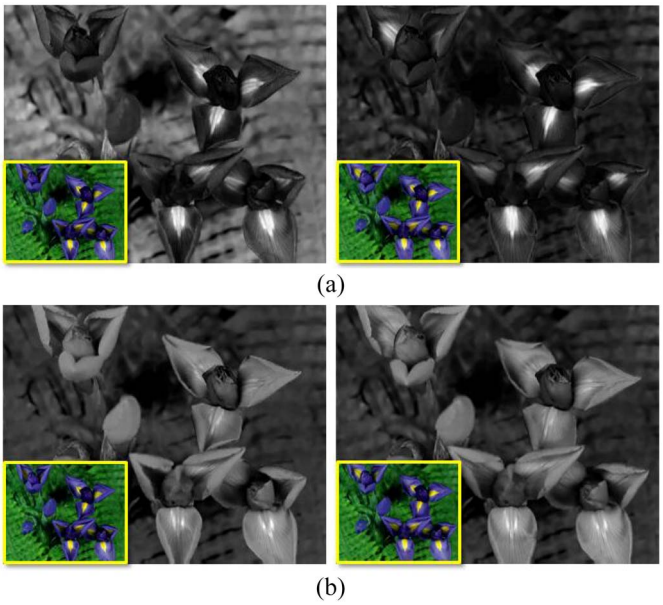


Fig. 1. Results of decolorizing two consecutive video frames, where the corresponding input color frames are shown on the lower left corners. (a) Results from [1], which simply applies an image decolorization method on video frames. (b) Results from our approach, which considers frame coherence. Note that the luminance values of the two input frames are not changed. While the results from [1] have very different luminance values, our approach can preserve the luminance consistence well.

Manuscript received August 25, 2016; revised January 4, 2017; accepted April 9, 2017. Date of publication May 2, 2017; date of current version April 13, 2018. This work was supported in part by the National Natural Science Foundation of China under Grant 61572316 and Grant 61671290, in part by the National High-Tech Research and Development Program of China (863 Program) under Grant 2015AA015904, in part by the Key Program for International S&T Cooperation Project under Grant 2016YFE0129500 of China, in part by the Science and Technology Commission of Shanghai Municipality Program under Grant 13511505000, in part by the Interdisciplinary Program of Shanghai Jiao Tong University under Grant 14JCY10, in part by the Natural Science Foundation of Jiangsu Province under Grant BK20140404, in part by the Research Grants Council of Hong Kong under Grant 28200215, and in part by the SRG Grant from City University of Hong Kong under Grant 7004415. This paper was recommended by Associate Editor Y. Yuan. *(Corresponding author: Bin Sheng.)*

Y. Tao, Y. Shen, and B. Sheng are with the Department of Computer Science and Engineering, Shanghai Jiao Tong University, Shanghai 200240, China (e-mail: shengbin@sjtu.edu.cn).

P. Li is with the Department of Mathematics and Information Technology, Education University of Hong Kong, Hong Kong (e-mail: pli@eduhk.hk).

R. W. H. Lau is with the Department of Computer Science, City University of Hong Kong, Hong Kong (e-mail: rynson.lau@cityu.edu.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/TCYB.2017.2695655

of adjacent frames. Second, the decolorization Gaussian mixture model (DC-GMM) takes the proximity values as features and learns to select a suitable decolorization strategy for each frame. We propose three different decolorization strategies: 1) low-proximity; 2) median-proximity; and 3) high-proximity decolorization (HPD) strategies, which have increasing computational costs and decolorization qualities. Our DC-GMM would try to select the right strategy for each input video frame in order to optimize between overall computation time and the output quality of the decolorized frames.

To evaluate our framework, we conduct quantitative and qualitative experiments as well as user study on diverse color videos. We analyze the computational complexity of each decolorization step. We apply color contrast preserving ratio (CCPR) and C2G-SSIM to measure the quality of the decolorization on individual frames. We also propose a novel temporal coherence degree (TCD) metric to measure the temporal coherence of the output video frames. Conclusively, our video decolorization framework has the following advantages.

1) *Time Efficiency:* By considering contents similarity of adjacent frames with high-proximity, our decolorization approach can effectively reduce the computation time by reusing previous grayscale frames.
2) *Quality Preservation:* We modify the salience-preserving decolorization algorithm [2] to be more suitable for decolorizing video frames in preserving chroma details.
3) *Temporal Coherence:* It makes sure the perceptual continuity across frames and avoids visual flickering in the grayscale videos.

The rest of this paper is organized as follows. Section II reviews related works on proximity evaluation, decolorization, and temporal coherence. Section III introduces the proposed decolorization framework, including proximity computation, the DC-GMM, and the three decolorization strategies. Section IV evaluates the proposed method on computational complexity, quantitative, and qualitative analyses, together with other decolorization methods. Finally, Section V briefly concludes this paper.

II. Related Work

A. Proximity Between Frames

The first step of our framework is to compute the decolorization proximity, i.e., the similarity degree between frames. Various similarity evaluation methods have been proposed and applied to different fields, such as target objects tracking and image retrieval. Compared with target objects tracking [3], its application on content-based image retrieval may be more relevant to our problem.

KL-divergence between Gaussian mixtures [4] has high computational complexity. Histogram matching [5], [6] is more efficient, but its robustness to rotation may have negative effects on our computed proximity scores. Beecks *et al.* [7] introduced the signature quadratic form distance to measure the distance between two Gaussian mixture models (GMMs) of high-dimensional feature descriptors. However, it fails to determine similarity strictly. Gooch *et al.* [8] used a strict way to define the color-difference of two arbitrary pixels of an image, by combining the luminance and chrominance difference. However, it has a high complexity.

B. Decolorization Methods

A lot of efforts have been made to improve the performance of image decolorization methods. Most color-to-gray algorithms could be categorized into local and global approaches.

Local methods typically divide an image into several regions and focus on the relationships among the pixels in one region. Neumann *et al.* [9] proposed a gradient-based algorithm to obtain the resultant image under Coloroid space. Smith *et al.* [10] incorporated the Helmholtz–Kohlrausch color appearance effects and adjusted the contrast by Laplacian pyramid. Other researches [11]–[14] have similar ideas to properly preserve contrast details locally. However, local methods may lose global integrality.

Global methods regard an image as a whole and try to establish a one-to-one mapping between each pixel and its corresponding grayscale value. Gooch *et al.* [8] attempted to minimize the chrominance and luminance differences between the color image and the grayscale one, but the algorithm is time-consuming. Ancuti *et al.* [15]–[17] proposed a new spatial distribution that discriminates the illuminated areas and color features better. However, it may not perform well on images with abundant hues and probably lead to contrast loss. Kim *et al.* [18] proposed a fast color-to-gray conversion method to preserve feature discriminability and color ordering well. However, it fails to explicitly consider features from global contrast. Liu *et al.* [19] made a good combination between RGB2GRAY and the parametric two-order model, but did not consider temporal coherence between adjacent images. Song *et al.* [1], [20] considered multiscale contrast preservation in both spatial and range domain, and it is efficient. Lu *et al.* [21], [22] proposed an algorithm based on a new quantitative metric. Recently, Du *et al.* [23] proposed to consider both local and global information in the decolorization process. Nevertheless, all these methods do not discuss the difference between image and video decolorization, which inspires us to develop a novel method unique to video decolorization.

C. Video Temporal Coherence

Maintaining temporal coherence is crucial to video editing, which has been studied in mobile applications [24], multimedia [25], and general usage [26]. Tao *et al.* [27] showed the wide applications and the significant impact of temporal coherence on human perception in decolorization. Ogata *et al.* [28] proposed a novel video editing model, which regards a video as a sequence of 0.5 or 1 second-long small boxes. Since the editing rule for each box is independent, video editing is accurate and systematic but the relationship between frames may be omitted. To improve frame coherence, a lot of researches have been conducted. Cuevas and Garcia [29] carried out a real-time high-quality shot detection strategy, analyzing abrupt transitions and gradual transition. Fast pixel-based analysis for abrupt transition

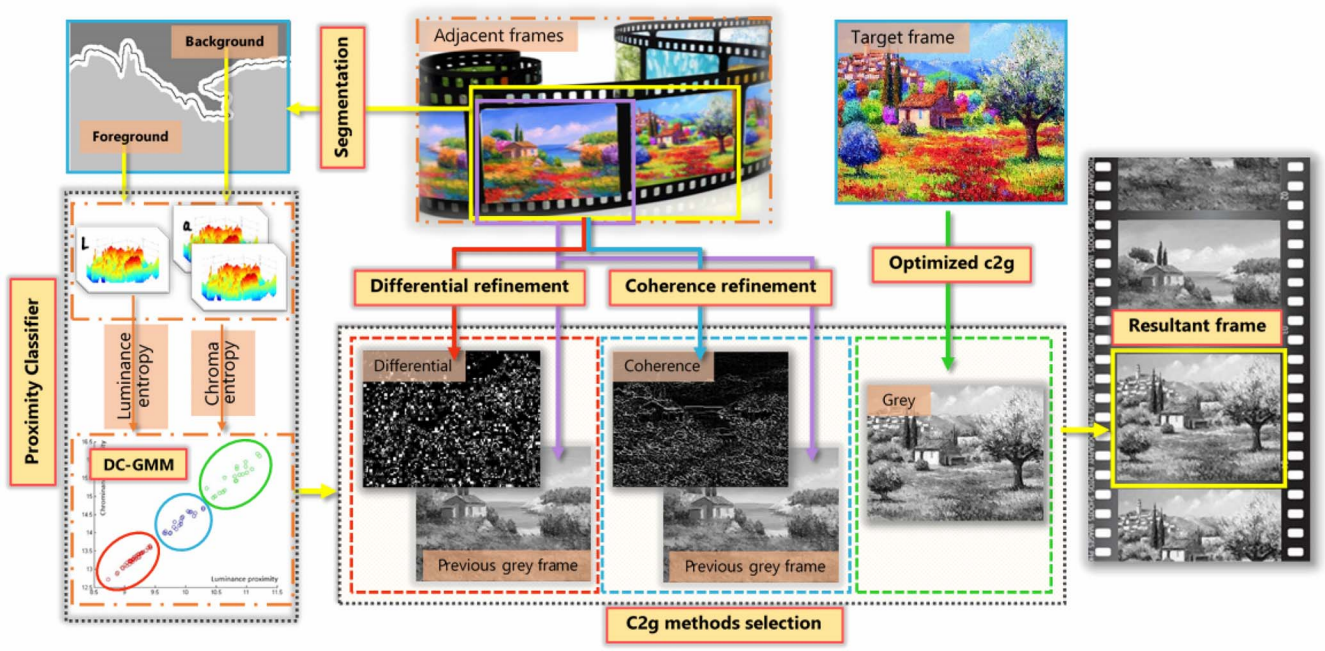


Fig. 2. Overview of our video decolorization framework. We first compute the decolorization proximity for each frame. We then use the DC-GMM classifier to select a specific decolorization strategy, and finally decolorize the frame into grayscale using the selected strategy.

can greatly reduce redundant time. Wang *et al.* [30] focused on spatial consistency and temporal coherence problems in complex video editing tasks, in order to make the transition between objects and background much smoother. Besides, methods based on Markov random field (MRF), an extension of Markov chain, are applied in many approaches. Chen and Tang [31] utilized the spatio-temporal MRF to estimate likelihood from the probabilistic motion field, which gives us a meticulous insight of the model. Other similar methods involving MRF [32], [33] also proposed effective solutions under different scenarios. However, we are not aware of any video decolorization works that emphasize on temporal coherence. In contrast, in this paper, all three decolorization strategies attempt to preserve temporal coherence.

III. Proposed Method

A. Method Overview

The proposed approach computes the decolorization proximity values of the foreground and background contents in the laboratory color space. These proximity values are then input to the DC-GMM as a video feature to help select the appropriate decolorization strategy from three strategies for each frame. Fig. 2 shows our proximity-based video decolorization framework, using temporal coherence optimization. After proximity calculation, each frame is classified by DC-GMM into three types: 1) high-proximity; 2) median-proximity; or 3) low-proximity. If the decolorization proximity is classified as high, the frame will be processed by the HPD strategy, which can greatly reduce computation time by reusing the previous decolorized frame via optical flow. If the decolorization proximity is low, the frame is regarded as a totally new frame and be decolorized by the low-proximity decolorization (LPD) strategy, an optimized salience C2G method with temporal coherence to avoid visual flickering. Otherwise, the frame is decolorized by the median-proximity decolorization (MPD) strategy, which may reuse previous decolorized results. To maximize both decolorization quality and efficiency, we penalize continuous selection of the same strategy over many consecutive frames.

For the sake of clarity, we introduce the main symbols that we use in the subsequent paragraphs as follows.

- I_i: The ith color frame.
- G_i: The decolorization result of the ith color frame.
- δ_i: The decolorization proximity value, computed between the $(i-1)$th and ith color frames. (See Section III-B).
- D_i: The differential refinement of the ith grayscale frame. (See Section III-C).
- C_i: The coherence refinement of the ith grayscale frame. (See Section III-D).

B. Decolorization Proximity

As pointed out in Section II-A, there are a lot of works for estimating the similarity of two images. However, most consider rotation or partial occlusion. Although there are some strict methods, e.g., Gooch *et al.* [8], they are typically computationally expensive.

1) Proximity Computation: Although the Euclidean distance is a strict distance function for determining the similarity of two frames, it is computationally expensive for high resolution images. Hence, we modify it to differentiate one image from another on all pixels under the laboratory color space to obtain the joint differential histogram, taking the whole spatial information into account. We then compute the absolute values of the pixel differences.

Algorithm 1 Proximity Computation

Require: input consecutive frames I_i and I_{i-1};
Ensure: I_i and I_{i-1} are in the *Lab* color space;
$X = I_i - I_{i-1}$;
1: $temp = histogram\ of\ (X)$; $\delta = zeros$;
2: **for** $j = 1 : length(temp)$ **do**
3: **if** $temp(j)! = 0$ **then** $\delta_i = \delta_i - temp(j) * log(temp(j))$;
4: **end if**
5: **end for**
6: $[\delta_i^L, \delta_i^a, \delta_i^b] = \delta_i$ under L, a, b channels;
7: $\delta_i^c = \sqrt{((\delta_i^a)^2 + (\delta_i^b)^2)/2}$;
8: $decolor\_strategy = DC - GMM\ classifier\ (\delta_i^L, \delta_i^c)$;
9: **return** *decolor_strategy*;

In addition, as inspired by mutual information [34], which is intricately linked to image entropy [35], we apply it in our computation to describe the average quantity of information in the differential histogram. Compared with the Euclidean distance, entropy is less affected uniform luminance changes. Although Wang *et al.* [36] also proposed a similarity measurement to separate the luminance change from structural information, it is computationally expensive. As a result, entropy can represent the real difference between frames more accurately and efficiently.

Hence, our decolorization proximity δ_i between frames $i-1$ and frame i is computed as

$$\delta_i = H(|I_i - I_{i-1}|) \tag{1}$$

where $H(\cdot)$ is the image entropy function defined as: $H(X) = -\sum_{j=0}^{255} P_X(j) \times \log P_X(j)$. $P_X(j)$ is the probability distribution of pixels with value j in image X. X is also in the Laboratory space, computed as $X = |I_i - I_{i-1}|$.

We compute δ_i in each of the L, a, and b channels to obtain proximity values δ_i^L, δ_i^a, and δ_i^b. These values will be considered as the input features of the DC-GMM classifier. Our proximity computation is summarized in Algorithm 1.

2) DC-GMM Classifier: Since different types of videos may have different amounts of change across frames (e.g., romantic versus kung fu videos), it is difficult to set fixed thresholds for selecting the three types of decolorization strategies. We apply machine learning techniques to address such problem. However, there are no images databases labeled with strict proximity values. Hence, we need to use an unsupervised learning technique. The GMM is known to be an effective unsupervised classifier, and have been applied to many applications. For example, Nguyen and Wu [37] proposed a new nonsymmetric mixture model for image segmentation, and Li *et al.* [38] modeled the visual affection with GMMs. Inspired by these successful GMM-based methods, we propose a novel classifier DC-GMM for our framework. DC-GMM can classify adjacent frame pairs into three types of proximity based on the video intrinsic properties.

When designing DC-GMM, we have the following concerns. First, we define a new function of data likelihood with the penalty term. Compared with applying the three decolorization strategies uniformly, continuously using the HPD will cause video quality loss while continuously using LPD will lead to high computational cost. Thus, we add a penalty term to the likelihood to avoid DC-GMM from assigning data points to one cluster continuously. For a suitable penalty term, we expect it to allow a small number of successive data points to be assigned to one cluster, but impose penalty as this number reaches a certain threshold. In addition, the scope of the penalty function should have the same magnitude order as the Gaussian distribution probability. As such, we define the penalty as a sigmoid function. The probability function of DC-GMM is then

$$P(u \in c_j) = \frac{\exp\{-\frac{1}{2}(u-\mu_j)^T \Sigma_j^{-1}(u-\mu_j)\}}{\sqrt{(2\pi)|\Sigma_j|}} - \frac{\lambda}{1+\exp\left\{-\xi(m_j-5)\right\}}, \quad m_j > 5 \tag{2}$$

where u is the data point, and c is the cluster. μ_j and Σ_j are the mean and covariance of c_j, respectively. ξ is the unit penalty, and m_j is the number of data points being successively assigned to c_j. λ is the penalty weight. When λ is large enough, we would inhibit more than five successive data points being assigned to one cluster.

Second, we separate the target frame into foreground and background. Since the foreground moving objects usually draw more attentions than the background, we separate them from the frame such that we can put more emphasis on them. We follow the method proposed by Ahn and Kim [39] to use a probability model to extract objects from a video sequence, even with a nonstationary background. This method is suitable for our problem, as it is efficient and considers temporal coherence in the segmentation. We then compute the decolorization proximity values under the laboratory color space for the foreground and the background separately. After that, we compute the quadratic mean of the a and b channels, which contribute to chrominance information. Finally, we consider the luminance (L channel) and the chrominance proximity values (δ_i^L, δ_i^c) as two features that depict each data point.

Third, we adopt a strict assignment approach. For the proximity features of the foreground pixels, if the probability of a data point belonging to one cluster is larger than the sum of those of the other two clusters, we are confident to assign the data point to this cluster. Otherwise, we need to additionally check to see if we should assign it to this cluster. We then look at the features for the background. If these features produce the same result, then we are confident to assign it to this cluster. However, if the result contradicts with that of the foreground, we assign it to the cluster with a lower proximity.

The DC-GMM classifier is summarized in Algorithm 2. It divides the data points into three clusters, as shown in Fig. 3. The cluster located nearest to the original point is considered as high-proximity, since their chrominance and luminance entropies are the lowest, indicating that the two frames are very similar. In contrast, the cluster located farthest from the original point is regarded as low-proximity, while the middle cluster is regarded as median-proximity. In the following sections, we introduce the three decolorization strategies for these three clusters, the HPD strategy, the MPD strategy, and the LPD strategy.

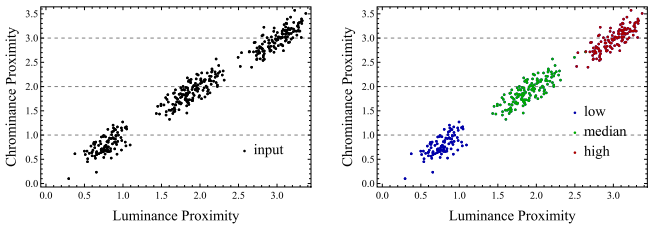


Fig. 3. DC-GMM classifier. (a) Distribution of the raw data, whose features are luminance proximity and chrominance proximity. (b) DC-GMM classifier is designed to divide the data into three clusters. Red, green, and blue color are used to represent different clusters, in which the red dots, with higher lumina and chroma proximity, are considered as high-proximity and will be processed by HPD, green dots as medium-proximity by MPD, and blue dots as low-proximity by LPD.

Algorithm 2 DC-GMM Classifier

Require: unlabeled data set u; cluster number $k = 3$;
Initial: data point u_i is assigned to cluster c_i by k-means++;
ξ is the unit penalty of successively classifying data points into a same cluster;
for $j = 1{:}k$ **do**
  $\mu_j = mean(\{u_i | c_i == j\})$;
  $\Sigma_j = cov(\{u_i | c_i == j\})$;
  $\pi_j = len(\{u_i | c_i == j\})/n$;
end for
while *True* **do**
  for $i = 1{:}n$ **do**
    $m_j =$ *the number of data points successively assigned to cluster* c_j;
    if $m_j \leq 5$ **then**
      $c_i = \arg\max_{j=1}^{k} \{\frac{\exp\{-\frac{1}{2}(u_i-\mu_j)^T \Sigma_j^{-1}(u_i-\mu_j)\}}{\sqrt{(2\pi)|\Sigma_j|}}\}$;
    else
      $c_i = \arg\max_{j=1}^{k} \{\frac{\exp\{-\frac{1}{2}(u_i-\mu_j)^T \Sigma_j^{-1}(u_i-\mu_j)\}}{\sqrt{(2\pi)|\Sigma_j|}} - \frac{\lambda}{1+\exp\{-\xi(m_j-5)\}}\}$;
    end if
  end for
  for $j = 1{:}k$ **do**
    $\mu_j = mean(\{u_i | c_i == j\})$;
    $\Sigma_j = cov(\{u_i | c_i == j\})$;
    $\pi_j = len(\{u_i | c_i == j\})/n$;
  end for
  $L = 0$;
  for $i = 1{:}k$ **do**
    $L += \log \pi_i \times \frac{\exp\{-\frac{1}{2}(u_i-\mu_j)^T \Sigma_i^{-1}(u_i-\mu_j)\}}{\sqrt{(2\pi)|\Sigma_j|}}$;
  end for
  if L reaches convergence **then**
    break;
  end if
end while
return $c[n]$;

C. High-Proximity Decolorization Strategy

In this strategy, we assume that frame I_i is very similar to frame I_{i-1}. Hence, we can reuse the grayscale output G_{i-1} of I_{i-1}, by adding a *differential refinement frame* D_i to it. This can greatly reduce the decolorization time.

Fig. 4. Optical flow algorithm tracks the pixel movements across frames. (A darker color represents a larger pixel movement.) In these two frames, a darker color appears around the hand, as the man waves his hand. We use this optical flow information to compute the differential refinement frame. (a) Frame i. (b) Optical flow. (c) Frame $i+1$.

To compute D_i, we consider the interframe motion computed from optical flow. Optical flow attempts to match the pixels between two frames to determine pixel movements, as shown in Fig. 4. The differential refinement D_i is computed based on these matched pixels as follows:

$$D_i(x) = \phi \sum_{k\in\{a,b\}} \left(I_i^k(x+v) - I_{(i-1)}^k(x)\right) + (1-\phi)\left(I_i^L(x) - I_{(i-1)}^L(x)\right) \quad (3)$$

where v is the movement vector of pixel x, and ϕ is a user set parameter to determine the weight of luminance and chroma information in the grayscale frame. Fig. 5 shows one result.

D. Median-Proximity Decolorization Strategy

Compared with HPD, the two input color frames I_i and I_{i-1} for MPD have a slightly larger difference. For the sake of efficiency, while we may reuse the previous grayscale result G_{i-1}, it may lead to visual inconsistency. In order to smoothen the contents difference between G_i and G_{i-1}, we define a coherence refinement frame C_i. The final result G_i is then a weighted sum of G_{i-1} and C_i.

The key to compute C_i is to establish a good measurement for interframe motion coherence. We define C_i using the forward flow from frame I_{i-1} to frame I_i as

$$M_i = \|I_i(x) - I_{i-1}(x-v)\|_2 \quad (4)$$

where $\|*\|_2$ stands for L_2-norm of a vector.

Inspired by the MRF model [33], assuming that each frame only depends on its previous one, we define each node as one pixel in the frame and each edge as the probability of optical flow. By maximizing this transmission possibility from the grayscale frame G_{i-1} to the current frame I_i, we can obtain the most probable grayscale result G_i for I_i considering coherence information. Fig. 6 shows one result. The probability of C_i is defined as

$$P(C_i) \propto \prod_{x\in I_i} \exp\left\{-(C_i(x) - G_{i-1}(x))/\sigma_p^2\right\} \times \exp\left\{-\frac{1}{M_i(x)}(C_i(x) - G_{i-1}(x-u))/\sigma_t^2\right\} \quad (5)$$

where the first term is the expectation of the coherence refinement, while the second term is the optical flow between the two adjacent frames. σ_p and σ_t are noise variances in frame transition. u is the movement vector of pixel x.

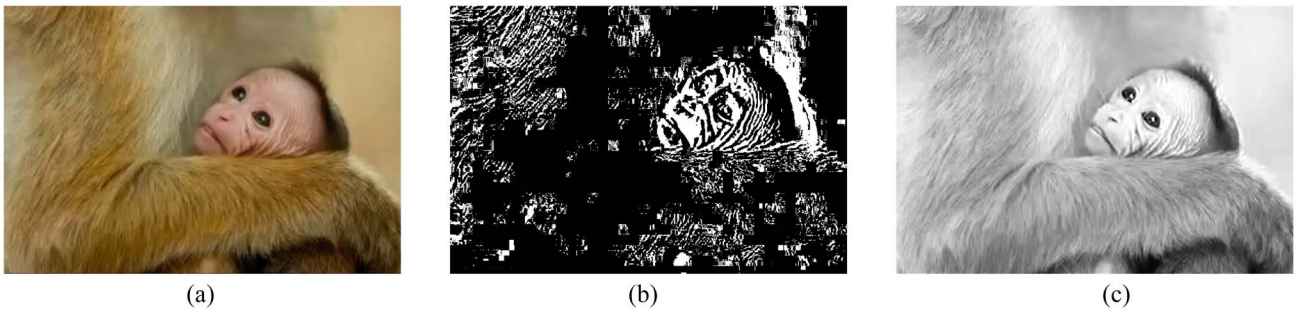


Fig. 5. HPD strategy in action. (a) Input color video frame. (b) Differential refinement frame, which measures the difference between the current frame and the previous frame. HPD combines (b) with the previous grayscale frame to obtain the (c) output frame.

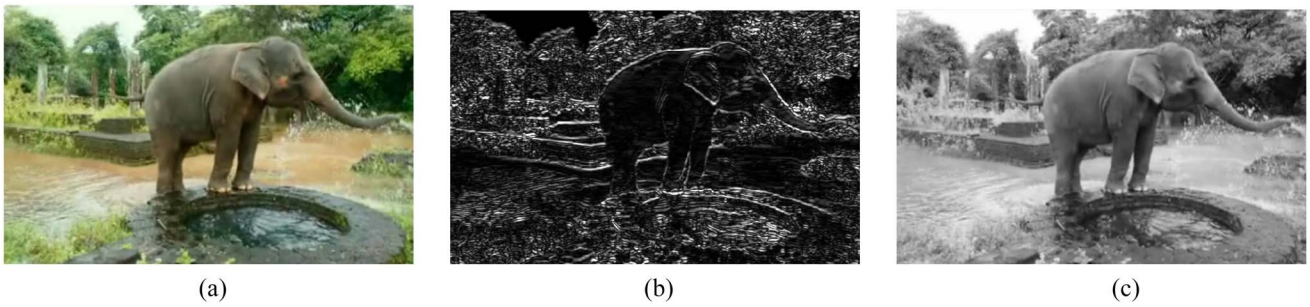


Fig. 6. MPD strategy in action. (a) Input color video frame. (b) Coherence refinement frame. It is used to help smoothen the output video. Compared with the differential refinement frame shown in Fig. 5(b), it contains more temporal details. MPD combines (b) with previous grayscale frame to obtain the (c) output frame.

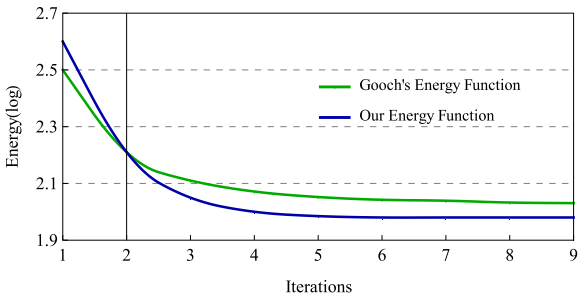


Fig. 7. Convergence of the energy function. The green curve is the energy function formulated by Gooch *et al.* [8], while the blue curve is the energy function optimized by our method with an additional time term. We observe that our optimized energy function achieves convergence with fewer iterations. In addition, ours achieves a lower energy, meaning that our decolorized videos are much closer to the input videos.

We use gradient ascent to find the maximum of $P(C_i)$. Given an appropriate step length α and initial value of C_i, which is set as I_i, we can iteratively update C_i, i.e., $C_i = C_i + \alpha \cdot (\partial / \partial C_i(x))P(C_i)$, until $P(C_i)$ reaches convergence. This process can be terminated at around ten iterations in our experiments.

E. Low-Proximity Decolorization Strategy

Since we assume that the contents of the two input frames are not very similar, we regard frame I_i as a completely new frame. Inspired by Gooch *et al.* [8], we propose an optimized salience decolorization algorithm here as our LPD, which is more suitable for video frames. Fig. 8 compares our approach with [8]. Our algorithm preserves the contrast

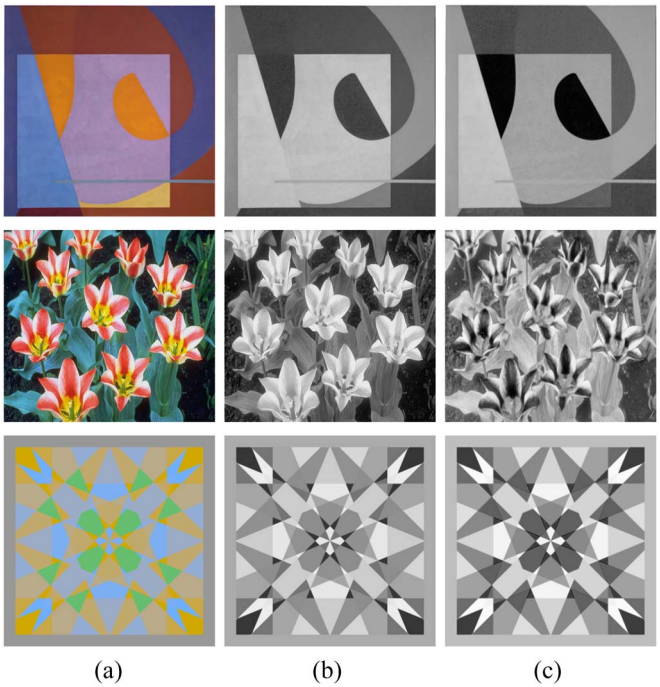


Fig. 8. LPD strategy in action. (a) Input color frames. (b) Results produced by Gooch *et al.* [8]. We can see that some of the colors are converted to the same grayscale value. (c) Results produced by our LPD. Compared with the results by Gooch *et al.* [8], our grayscale frames preserve better contrast details. In other words, different colors are represented discriminably in our output frames.

of both neighboring pixels in one frame and pixels in the adjacent frames. Coherence refinement is again considered for smoothening between the two adjacent frames.

TABLE I
ENERGY COMPARISON

	Song [1]	Kim [18]	Lu [22]	Liu [19]	Ours
Action	2.872	3.017	2.741	2.916	2.478
Cartoon	2.693	2.818	2.601	2.744	2.314
Scenery	2.445	2.562	2.419	2.476	2.085

TABLE II
TIME OF EACH STEP

	Time (s/frame)		
Proximity computation	0.03		
DC-GMM classifier	0.09		
HPD, MPD, and LPD	1.00	3.51	4.69

TABLE III
PROPORTION OF THREE METHODS

		HPD	MPD	LPD
Action	Time (s/frame)	1.056	3.517	5.074
	Proportion (%)	32.81	40.18	27.01
Cartoon	Time (s/frame)	1.004	3.521	4.687
	Proportion (%)	35.64	39.92	24.44
Scenery	Time (s/frame)	0.993	2.938	4.126
	Proportion (%)	40.07	40.39	19.54

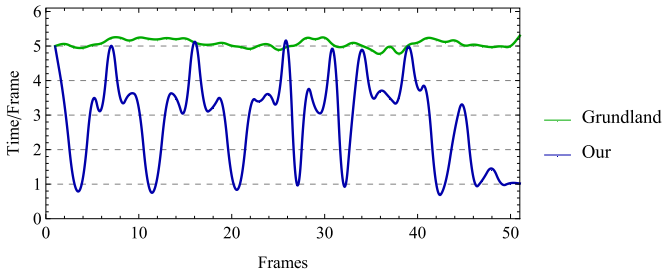


Fig. 9. Line charts for the decolorization time. The green curve shows the decolorization time per frame by Song *et al.* [1], while the blue curve shows the decolorization time per frame by our method, through applying the high, median, and low proximity decolorization strategies. Benefitted from the high efficiency of the high and median decolorization strategies, our framework requires much less time compared with Song *et al.* [1].

Unlike images, videos have a time order. Hence, we add a time term to the energy function so that the grayscale frames are able to contain both spatial and time information, resulting in a smoother and more consistent grayscale video. Fig. 7 shows the convergence of the energy function. From Table I, we can see that our approach is able to achieve a lower energy compared with other methods, which means that our output grayscale videos are perceptively much closer to the input color videos. In addition, scenery videos can be better decolorized using our approach, as their contents are relatively static across frames. Action movies have higher convergent energy because of the relatively higher dynamic contents. The optimized energy function is as follows:

$$E_s = \sum_{(x,y)\in I} ((G_i(x) - G_i(y)) - \eta_i(x,y))^2 \quad (6)$$

$$E_t = \sum_{x\in I} ((G_i(x) - G_{i-1}(x)) - (I_i(x) - I_{i-1}(x)))^2 \quad (7)$$

$$E = (1-\beta)E_s + \beta E_t \quad (8)$$

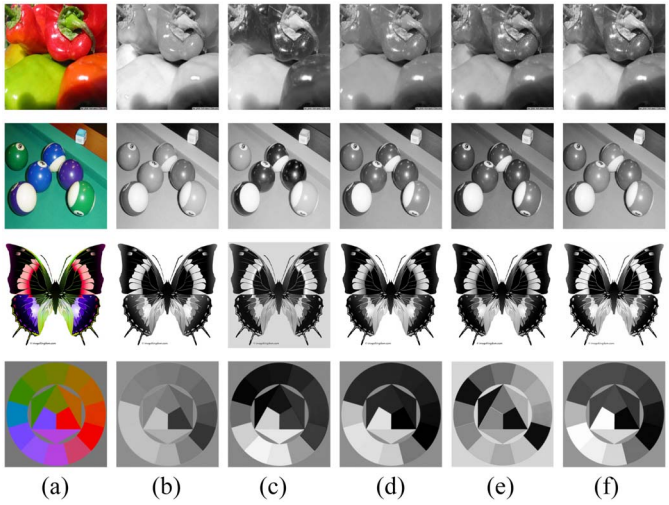


Fig. 10. (a) Images from the Cadik standard dataset decolorized by (b) Song *et al.* [1], (c) Kim *et al.* [18], (d) Lu *et al.* [22], and (e) Liu *et al.* [19]. Comparatively, (f) our results show best performance in terms of contrast preservation and discrimination for different colors in the origin images.

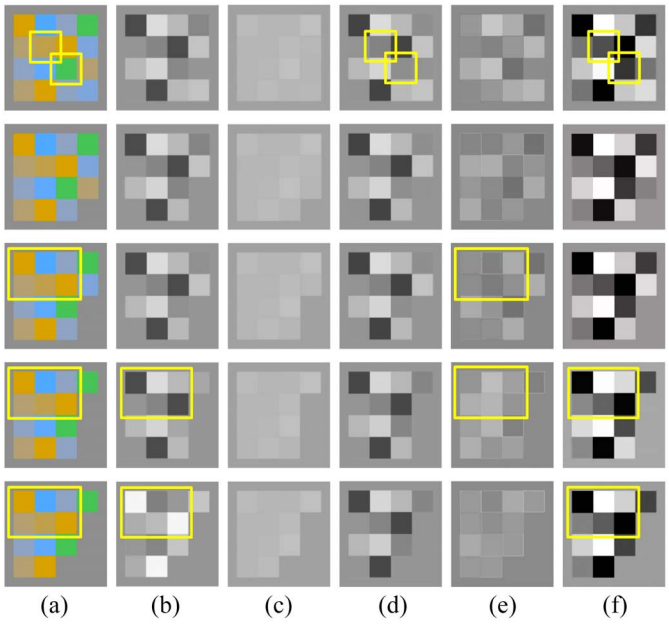


Fig. 11. (a) Input. Comparison between the results proposed by (b) Song *et al.* [1], (c) Kim *et al.* [18], (d) Lu *et al.* [22], (e) Liu *et al.* [19], and (f) our decolorization. In Song's results [1], the same color in different frames is wrongly converted into different grayscale values, resulting in color inconsistence. Kim's results [18] and the last frame in Liu's results [19] both show poor performance in maintaining chroma and luminance contrasts. Finally, some frames in Lu's results [22] and Liu's results [19] partially lost contrast details, converting different colors into less discriminable or even the same grayscale value. Comparatively, our results shows the best performance both in contrast preservation (i.e., different colors can be discriminated clearly) and temporal coherence (i.e., the same color is shown consistently through different frames in our decolorized outputs).

where $\eta_i(x, y)$ is the difference between two arbitrary pixels x, y in I_i. E_s is the energy of I_i, while E_t is the time energy between the input frames I_i and I_{i-1}. The final energy E is by combining E_s and E_t with weight β.

To minimize the energy function, the conjugate gradient method (CGM) is applied, which is regarded as the best

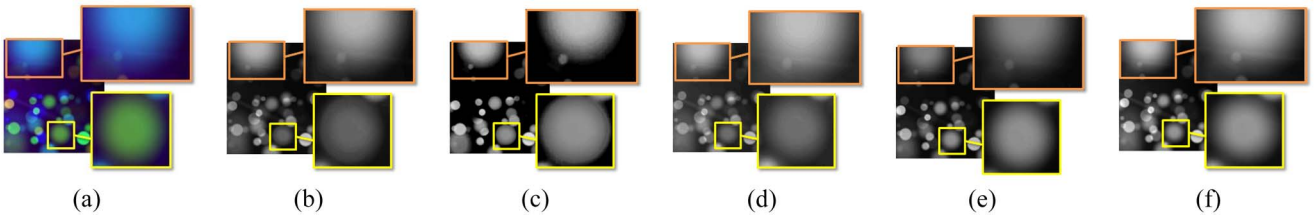


Fig. 12. (a) Input frame. Comparison between the edges of objects in magnified areas. The luminance transition of circle edges needs to be preserved when it is converted into grayscale. The luminance difference between the circle and the background also needs to be preserved as well. (b) In Song's result [1], the green circle is decolorized too dark. (c) In Kim's result [18], the size of blue circle is smaller than the input one, and the boundary of the green circle is too concrete. (d) Lu's result [22] shows a larger blue circle and low luminance for the green circle. (e) Finally, in Liu's result [19], the blue circle is smaller and darker and the tiny blue facula neighboring the green circle is hard to discern. Through the comparison, (f) our result shows most similar margin, size, and luminance details compared to the original input video frame, meaning that our video decolorization preserves the detailed features better.

Fig. 13. Object details preservation. There are two monkeys in the (a) input frame. However, in the result produced by (b) Song *et al.* [1], the edges of the monkey are blur, especially for the monkey on the left. It is hard to discriminate the monkey from the background. (c) In our result, the edges in the highlighted area is preserved very well as the edges of the monkey exist in the original color frame.

and most time efficient iterative algorithm to solve unconstrained optimization problems, such as energy minimization. Thus, it can be used to find the optimal grayscale value convergent to the initial color difference $\eta_i(x, y)$. However, CGM greatly relies on the initial values and may fall into a local minimum. We have two refinements to address these limitations.

1) To start with a better initial value, we add weighted RGB information of the color image into the initial grayscale value, instead of simply using the luminance channel. The weight is set to average at first, and will be updated during iterations.
2) To avoid falling into a local minimum, we define an evaluation for the iteration results. In view of the probability theory, the energy function infers that the grayscale difference $G_i(x) - G_i(y)$ between pixel pair (x, y) [represented by $g(x, y)$] follows a Gaussian distribution with mean signed distance scalar $\eta_i(x, y)$ in the perceptually uniform laboratory color space. σ determines the degree of distribution concentration

$$G\Big(\eta(x, y), \sigma^2\Big) \propto \exp\left\{-\frac{|g(x, y) - \eta(x, y)|^2}{2\sigma^2}\right\}. \quad (9)$$

We repeat the CGM with different initial weights until the value of (9) is large enough. Hence, the iterative algorithm will produce a comparatively better result than the local minimum.

IV. Experimental Results

We have conducted the experiments on MATLAB, using the Cadik standard image dataset and test videos downloaded by ourselves. The rest of this section is organized as follows. Section IV-A evaluates our method in cutting down redundant decolorization time based on computational complexity analysis. Section IV-B demonstrates qualitative performance of our method compared with the results in other papers. On the other aspects, Section IV-C uses quantitative metrics to evaluate the detail preservation and temporal coherence of our method and other methods. Finally, Section IV-D presents the user study conducted on decolorization performance to compare our method with existing methods, in terms of preference and accuracy.

A. Computation Time

First, we discuss the computation complexity about each step and show it in Table II. Let n be the number of pixels in one frame. The running time of the proximity calculation is $O(n)$, which on average costs less than 0.1 s. The DC-GMM classifier applies EM algorithm to converge and on average costs less than 0.1 s. Among the three decolorization strategies, HPD method is the most efficient, whose running time is $O(n)$. MPD method iterates by gradient ascent method and LPD method iterates by CGM to converge. HPD, MPD, and LPD on average take 1, 3, and 5 s, respectively. The bottleneck to this problem is the overall time of the decolorization algorithm. If the algorithm chooses MPD or LPD, a large amount of time will be consumed in the iteration though we have done optimization to it. On the contrary, the HPD can save a lot of time through bypassing the most time-consuming part. That is why we designed DC-GMM to choose the proper decolorization strategies. Accelerating the proposed decolorization strategies using GPUs would be our future work.

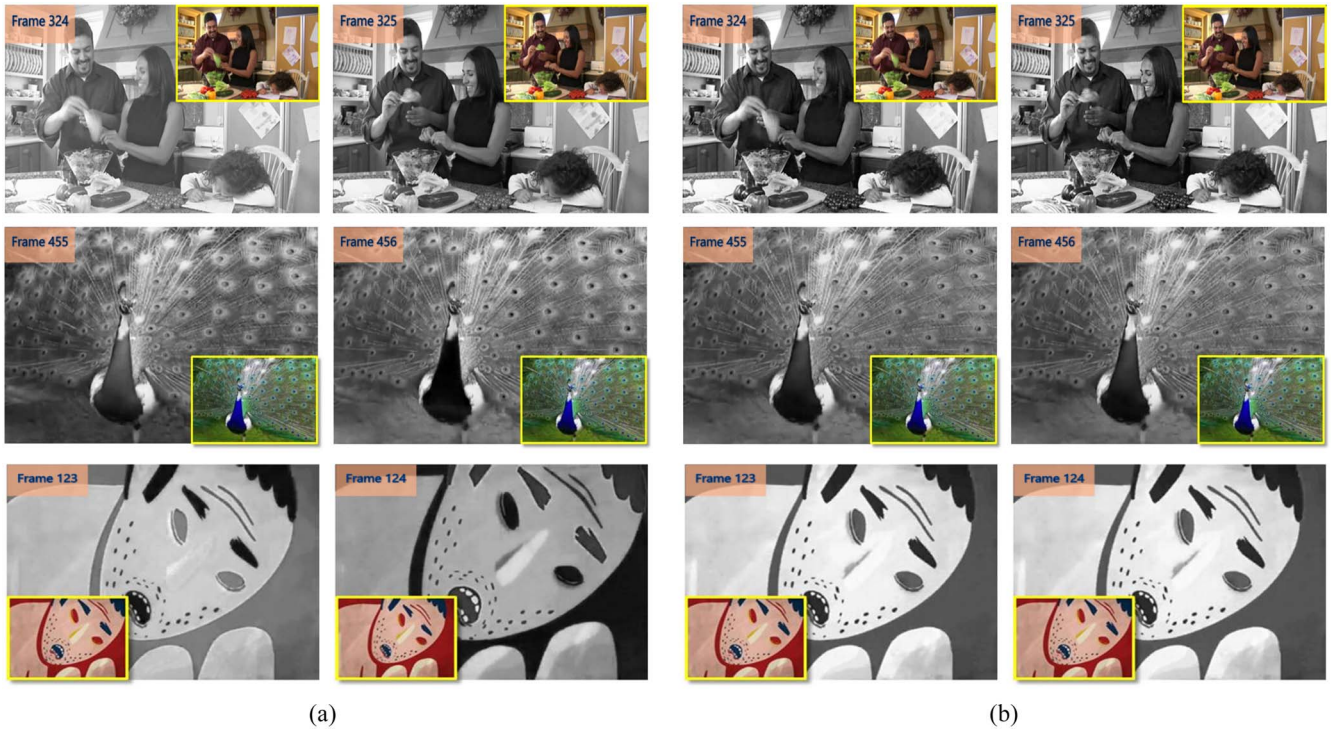


Fig. 14. Coherence preservation. These are three different videos. For each video, we convert two adjacent color frames into grayscale. The results produced by (a) Song *et al.* [1] displayed visual flickering effects, which means that the original adjacent color frames are almost the same while the decolorized frames show obvious differences, especially in global and local luminance values. Compared with Song *et al.* [1], (b) our results preserve luminance coherence better.

TABLE IV
CCPR RESULTS OF ACTION AND SCENERY VIDEOS BY DIFFERENT DECOLORIZATION METHODS

γ	Action					Scenery				
1	Song [1]	Kim [18]	Lu [22]	Liu [19]	Ours	Song [1]	Kim [18]	Lu [22]	Liu [19]	Ours
2	0.73	0.71	0.72	0.71	0.75	0.77	0.74	0.75	0.75	0.80
3	0.66	0.63	0.66	0.65	69	0.73	0.70	0.72	0.71	0.74
4	0.63	0.61	0.62	0.62	0.66	0.69	0.67	0.70	0.68	0.71
5	0.60	0.58	0.60	0.59	0.63	0.67	0.66	0.69	0.65	0.69
6	0.58	0.56	0.57	0.56	0.61	0.65	0.64	0.66	0.64	0.66
7	0.56	0.54	0.55	0.55	0.60	0.64	0.62	0.64	0.63	0.66
8	0.54	0.52	0.54	0.53	0.58	0.63	0.60	0.62	0.61	0.65
9	0.53	0.51	0.52	0.52	0.56	0.62	0.60	0.61	0.60	0.64
10	0.52	0.50	0.51	0.51	0.55	0.61	0.59	0.61	0.59	0.63
11	0.51	0.48	0.50	0.49	0.55	0.60	0.58	0.60	0.59	0.63
12	0.50	0.47	0.49	0.48	0.54	0.59	0.56	0.59	0.58	0.62
13	0.49	0.46	0.48	0.47	0.52	0.58	0.55	0.58	0.57	0.62
14	0.48	0.44	0.47	0.46	0.53	0.57	0.54	0.58	0.55	0.61
15	0.47	0.44	0.46	0.45	0.51	0.56	0.53	0.58	0.54	0.60

Second, we apply our algorithm to various videos, including action movies, cartoons, and scenery, Table III records time cost and proportion of the three decolorization strategies, HPD, MPD, and LPD, on average of different kinds of videos. One frame in action movies requires 3.12 s to decolorize on average by our method, more time efficient than 5.07 s processed by current frame-by-frame decolorization methods. Scenery videos have more static and less changeable contents than action movies and cartoons. Thus, the proportion of frames decolored by HPD and MPD is much larger, and each frame only costs 2.39 s on average by our method. In the conclusion, our framework can save about 40% redundant time in video decolorization.

To give a clearer demonstration, we record the decolorization time of a video in a line-chart (shown in Fig. 9). The line-chart also shows our algorithm's effective use of the three decolorization strategies to balance the time efficiency and quality.

B. Qualitative Evaluation

To have a direct demonstration of our well-preserved video quality, we make comparisons between results of our approach and other existing methods under the Cadik standard dataset (Fig. 10) and various types of videos. Whatever the contents of videos are, scenery, figures, or cartoons, our framework

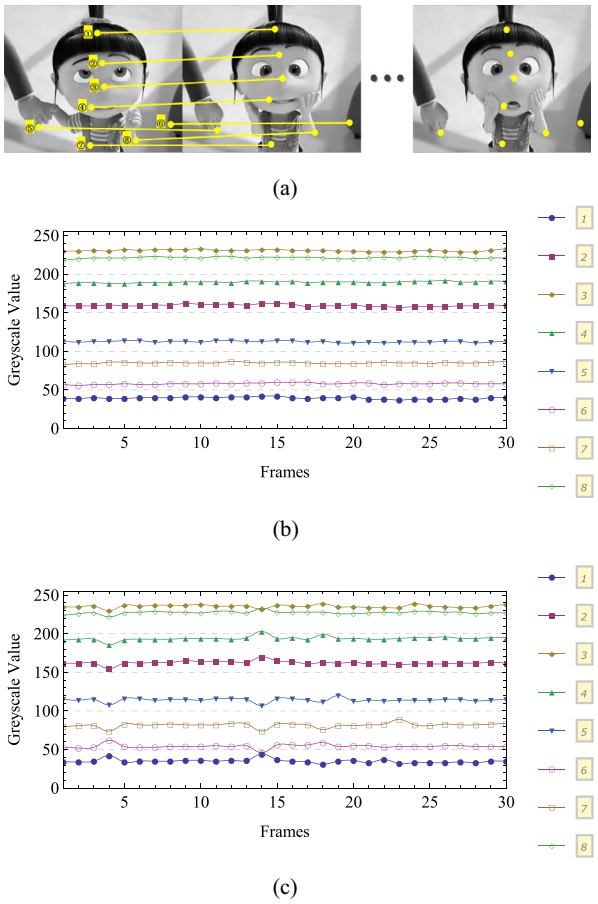


Fig. 15. Temporal coherence evaluation. (a) Eight key points numbered from 1 to 8 obtained by SIFT. Grayscale values of these key points along 30 frames decolorized by (b) our method and (c) results by Liu *et al.* [19]. Since the color videos are coherent between frames, preserving the coherence in the decolorized videos is very necessary for good visual experience. Comparatively our fitted curves are much smoother, which means that we better preserve the temporal coherence and consistency of the same pixels, even for the contents of the videos. The singular points in the curves of Liu *et al.* [19] show that the same contents may be decolorized to different ones as some pixels have inconsistent grayscale values.

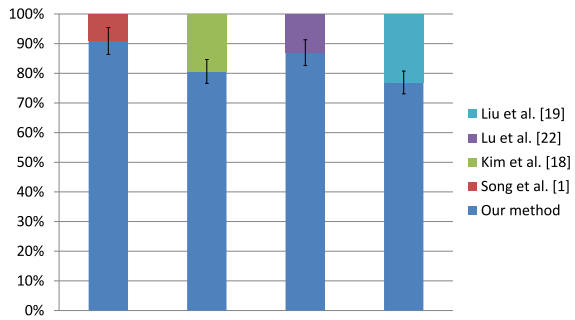


Fig. 16. Results of the preference experiment. Each time, the participants are asked to choose one result video from two grayscale videos, one from our approach and the other from one of the other four methods [1], [18], [19], and [22]. The error bars are at 95% confidential level.

has impressive performance in details restoration, clear objects margin, and temporal coherence.

In a wide comparison shown in Fig. 11, we decolorize a video with several squares of different colors, which will

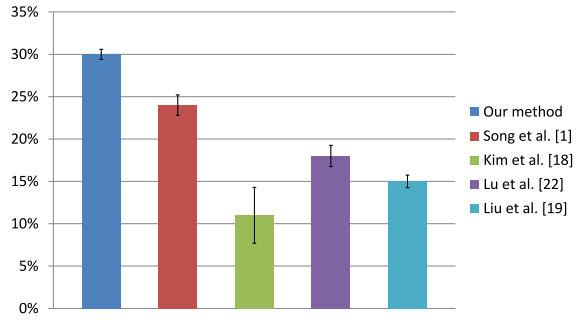


Fig. 17. Results of the accuracy experiment. Each time, participants are asked to choose one grayscale video that represents the original color video best, from five candidate grayscale videos, one from our approach and others from the four methods [1], [18], [19], and [22]. The error bars are at 95% confidential level.

disappear one by one. With the disappearance of squares, the color and luminance will change and thus cause problems during decolorization. The results of Song *et al.* [1] lose the contrast details and the last frame even has wrong chrominance and luminance relationship. The results of Kim *et al.* [18] completely lose the contrast details, and the initial color and luminance are difficult to distinguish. The results of Lu *et al.* [22] also have some trouble in retaining contrast details. The last frame of Liu *et al.* [19] showed very poor performance in maintaining chroma and luminance contrasts. In additional, their results seem to demonstrate visually flickering defects due to lack considerations for temporal coherence. Our approach comparatively avoids the above problems and has better perceptive visual quality.

To show our edges preservation, we give out a magnified detailed comparison in Fig. 12, where Song *et al.* [1] and Lu *et al.* [22] failed to preserve the contrasts between the center and border of the facula. Kim *et al.* [18] lost the gradual fuzzy near the facular edges. Compared with their methods, our approach can better preserve the details in the decolorized videos. When objects move fast, grayscale frames without smooth transition will seem fuzzy and dim. As shown in Fig. 13, our results preserve distinct edges of the monkeys, while Song *et al.* [1] is blurring their outline.

In addition, our framework takes temporal coherence into consideration to avoid visual flickering effects. In Fig. 14, with the contents luminance unchanged in the input color frames, our approach better maintains the luminance coherence. However, the results of Song *et al.* [1] show serious visual artifacts because of decoloring the videos frame by frame, which also appears in many related methods.

Furthermore, to prove the robustness of our approach, we have tested various videos for comparison, and our results perform better all around in the cases tested. An instance of movie "The Avengers" is shown in Figs. 18–20, in which we compare our results with Kim *et al.* [18], Lu *et al.* [22], and Song *et al.* [1]. Compared with Kim *et al.* [18] (shown in Fig. 18), our results show better coherence between adjacent frames and preserve discriminative contrast of neighboring colors better. Besides, Lu *et al.* [22] (Fig. 19) showed poor performance in discriminating the red color from the

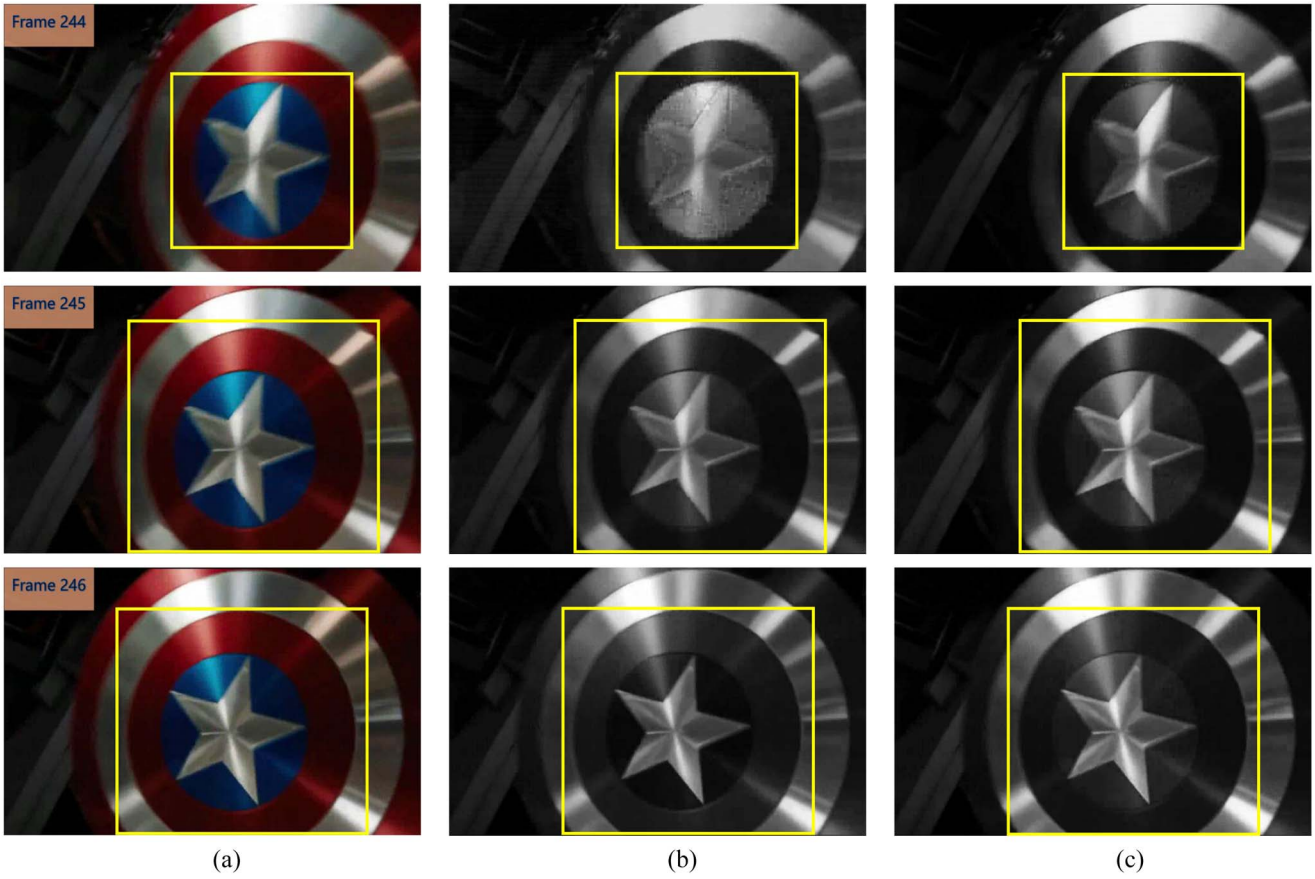


Fig. 18. Comparison with the methods proposed by Kim *et al.* [18]. (a) First column shows the input color frames. (b) Second column shows the results produced by Kim *et al.* [18], which lose color contrast and cannot discriminate the silver and blue colors in the original color frames. In addition, although the same color should be kept constant across frames, the grayscale values for red and blue are visually interchanged, which is a temporal chroma inconsistence problem. (c) Third column shows the results by our method, where the consistence of colors is well preserved.

neighboring black color. Song *et al.* [1] (Fig. 20) worked not very well when the content is dark, and there are messy stripes in the background and anamorphosis in the character's face. Compared with these three methods [1], [18], and [22], our decolorization avoids these problems and performs better.

C. Quantitative Evaluation

1) Color Contrast Preserving Ratio: Though we focus on video decolorization, we still need to deal with images, which are the basic components of a video. To quantitatively evaluate the performance of each frame decolorization, we apply the CCPR [40] via computing the percentage of distinct pixels in input color frames, which remain distinctive after decolorization. CCPR is defined as follows:

$$\text{CCPR} = \frac{\#\{(x,y)|(x,y) \in \Omega, |G(x) - G(y)| > \gamma\}}{||\Omega||} \tag{10}$$

where (x, y) is a pair of random pixels in the frame, and $|G(x) - G(y)|$ is the grayscale difference between this pixel pair. Ω is the set containing all neighboring pixel pairs, while $||\Omega||$ is the total number of pixel pairs in Ω. γ is a threshold indicating whether the color/grayscale difference is smaller enough. $\#\{\cdot\}$ refers to the number of pixel pairs with grayscale difference higher than γ.

We have tested two types of videos, action and scenery, and collected the frames in color videos of both kinds. We vary γ from 1 to 15 in the experiments and record the average CCPR values for each video genre in Table IV. It can be clearly seen from Table IV that our method has the highest CCPR values under most conditions, which means we can better preserve the color contrast of input color video frames. Also, all proposed methods do better in scenery videos since the contents of scenery videos are usually simple and consist of big color lumps.

2) C2G-SSIM: To make a more all around and objective single frame decolorization evaluation, we apply C2G-SSIM [41] to automatically predict the perceived luminance, contrast and structure performance of C2G converted frames.

We also test on the two kinds of videos, action and scenery, and collect 100 frames in color videos of both video types. We record the average C2G-SSIM values for each video genre in Table V. It can be clearly seen from Table V that our approach has the highest C2G-SSIM in both video genres, which means our decolorization can better preserve the luminance, contrast and structure of input color video frames. Also, all the tested methods perform better in scenery videos since the contents of scenery videos are comparatively static.

3) Temporal Coherence: Video decolorization, being different from single image decolorization, considers the

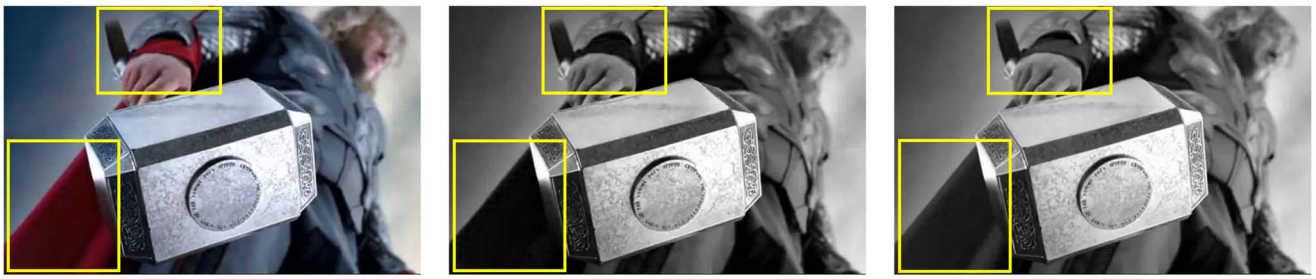


Fig. 19. Comparison with Lu *et al.* [22]. Note that in the marked regions, the red color should be discriminated from the neighboring black color. However, in the result produced [22], it converts red and neighboring black into the same grayscale value, which makes it difficult to distinguish the two colors. (a) Input frame. (b) Lu's result [22]. (c) Our result.

Fig. 20. Comparison with Song *et al.* [1]. (a) First column shows the input color frames. (b) In the results produced by Song *et al.* [1], we find anamorphosis in the character's face and messy stripes in the background. (c) In our results, such defects do not exist.

TABLE V
C2G-SSIM RESULTS OF DIFFERENT DECOLORIZATION METHODS

	Song [1]	Kim [18]	Lu [22]	Liu [19]	Ours
Action	0.9503	0.9377	0.9495	0.9477	0.9560
Scenery	0.9928	0.9800	0.9870	0.9806	0.9950

consistency and coherence between adjacent frames. In other words, the grayscale values of one same object, even one same pixel, should change smoothly along with the frames so that the decolorized videos will look coherent with no visual flickering effect.

To quantitatively evaluate the temporal coherence among frames, we apply SIFT to obtain the key points of the first frame and track these key points along the time line of the videos. We then record all the key points' values in the video decolorized by our method and other methods, and then fit them to the curves. Comparisons under action and scenery videos are shown in Fig. 15. To make it clear, we only mark eight top key points in the graph and record the grayscale values of key points in 30 successive frames. It is obvious that our curves are smooth, indicating our resulting videos are perceptually temporal coherent. Comparatively, the curves of Liu *et al.* [19] have several abrupt salient points and pit points. That means, their method loses the coherence of same pixels between frames and will cause the same object to look different and inconsistent in the decolorized videos.

We also define a function to evaluate the TCD statistically by measuring how smooth the grayscale value curves are

$$\mathrm{TCD} = \sum_{i=1}^{|Kp|} w_i \sum_{j=1}^{n} \left| f_i''(j) \right| \tag{11}$$

where, Kp is the set of key points found by SIFT, and $|Kp|$ is the number of elements in it. $f_i(\cdot)$ is the function of grayscale value curve of key point i obtained by proposed methods and j represents the jth frame in all n frames. w_i weights the importance of specific key points, which depends on its stableness measured by Harris corner detector using SIFT.

The second order derivatives of grayscale value curves show the smooth degree of the curves on specific points. We add them up with corresponding weight and get the TCD values to evaluate temporal coherence preservation performance of

TABLE VI
TCD COMPARISON

	Song [1]	Kim [18]	Lu [22]	Liu [19]	Ours
Action	0.631	1	0.734	0.896	0.137
Scenery	0.703	1	0.785	0.936	0.372

our method in comparison with other methods. The results are shown in Table VI (with all values normalized to 0–1), and our method has the smallest TCD value, pointing out that our decolorized videos preserve the temporal coherence best. In addition, because each method (HPD, MPD, and LPD) is proposed to preserve the coherence between the current frame and the previous one, the temporal coherence is still preserved even though the three methods are applied alternatively according to the decolorization proximity computed.

D. User Study

As there are no specific standard metrics to quantitatively measure the quality of the videos decolorized using different methods, we have conducted a user study to evaluate our method with others. We randomly selected a total of ten videos, covering animation, action movie, documentary, etc. Three hundred participants were invited to watch the grayscale videos decolorized by our approach and by other existing methods [1], [18], [19] and [22], in random order.

The user study consists of two parts, namely, the preference and the accuracy experiments. In the preference experiment, the participants were asked to watch two grayscale videos, one produced by our approach and another by a random method from [1], [18], [19], and [22]. They were then asked to choose the one that they prefer most. In the accuracy experiment, the participants were shown with six videos, including the input color video and five grayscale videos produced by all methods [1], [18], [19], [22], and our method. The participants were then asked to choose one video out of the five grayscale videos that they consider representing the color video best. Results of the user study are shown in Figs. 16 and 17. We can see that our proposed method performs the best in both the two experiments. The error bars show that our method consistently outperforms the other methods.

V. CONCLUSION

This paper proposes a method to address the video decolorization problem. It is based on computing the proximity of two consecutive frames, which is then used to select a suitable decolorization strategy to decolorize the input video frame. Specifically, each input video frame is classified to one of the three different clusters by the DC-GMM classifier, according to the luminance and chrominance proximity between this frame and its previous frame. We propose three decolorization strategies to decolorize the three clusters of frames. First, we propose the time efficient HPD strategy to decolorize the color frames assigned to the high-proximity cluster, which have very similar contents with their previous frames. Second, we propose the LPD strategy to decolorize the color frames assigned to the low-proximity cluster, which have dissimilar contents with their previous frames. Although LPD can better preserve the details in the color frames, it has a high computationally cost. Finally the color frames assigned to the remaining cluster are processed by the MPD strategy. All three proposed strategies consider temporal coherence between frames as they decolorize the input video frames.

To evaluate the proposed method, we have compared it with other methods on different types of frames and videos. We have also conducted quantitative evaluations, CCPR, C2G-SSIM, and TCD, to statistically measure the decolorization quality and ability to preserve temporal coherence of our method. Finally, we have conducted a user study that include the preference and accuracy experiments, to comprehensively evaluate our method. To sum up, our approach produces the best decolorization performance, in terms of computation efficiency, decolorization quality, and preservation of temporal coherence. As a future work, we would be interested in accelerating the proposed decolorization framework via GPUs.

REFERENCES

[1] Y. Song, L. Bao, and Q. Yang, "Real-time video decolorization using bilateral filtering," in *Proc. WACV*, Steamboat Springs, CO, USA, 2014, pp. 159–166.
[2] D. Ravipati, P. Karreddi, and A. Patlola, "Real-time gesture recognition and robot control through blob tracking," in *Proc. SCEECS*, Bhopal, India, 2014, pp. 1–5.
[3] S. He, Q. Yang, R. W. H. Lau, J. Wang, and M.-H. Yang, "Visual tracking via locality sensitive histograms," in *Proc. CVPR*, Portland, OR, USA, 2013, pp. 2427–2434.
[4] J. Goldberger, S. Gordon, and H. Greenspan, "An efficient image similarity measure based on approximations of KL-divergence between two Gaussian mixtures," in *Proc. ICCV*, 2003, pp. 487–493.
[5] D. Russakoff, C. Tomasi, T. Rohlfing, and C. Maurer, Jr., "Image similarity using mutual information of regions," in *Proc. ECCV*, 2004, pp. 596–607.
[6] F. Huang, X. Qu, H. J. Kim, and J. Huang, "Local pixel patterns," *Comput. Vis. Media*, vol. 1, no. 2, pp. 157–170, 2015.
[7] C. Beecks, A. M. Ivanescu, S. Kirchhoff, and T. Seidl, "Modeling image similarity by Gaussian mixture models and the signature quadratic form distance," in *Proc. ICCV*, Barcelona, Spain, 2011, pp. 1754–1761.
[8] A. A. Gooch, S. C. Olsen, J. Tumblin, and B. Gooch, "Color2Gray: Salience-preserving color removal," *ACM Trans. Graph.*, vol. 24, no. 3, pp. 634–639, 2005.
[9] L. Neumann, M. Čadík, and A. Nemcsics, "An efficient perception-based adaptive color to gray transformation," in *Proc. CAE*, 2007, pp. 73–80.
[10] K. Smith, P.-E. Landes, J. Thollot, and K. Myszkowski, "Apparent greyscale: A simple and fast conversion to perceptually accurate images and video," *Comput. Graph. Forum*, vol. 27, no. 2, pp. 193–200, 2008.
[11] W. Zhu, R. Hu, and L. Liu, "Grey conversion via perceived-contrast," *Vis. Comput.*, vol. 30, no. 3, pp. 299–309, 2014.
[12] Y. Zhao and Z. Tamimi, "Spectral image decolorization," in *Proc. ISVC*, Las Vegas, NV, USA, 2010, pp. 747–756.
[13] A. Y.-S. Chia, K. Yaegashi, and S. Masuko, "Image decolorization by maximally preserving color contrast," in *Proc. ICPRAM*, Angers, France, 2014, pp. 453–459.
[14] Z. Xu, "Optimization-based image decolorization," in *Proc. HKUST UROP*, 2012, pp. 1–12.
[15] C. Ancuti, C. Ancuti, and P. Bekaert, "Enhancing by saliency-guided decolorization," in *Proc. CVPR*, Colorado Springs, CO, USA, 2011, pp. 257–264.
[16] C. O. Ancuti, C. Ancuti, C. Hermans, and P. Bekaert, "Image and video decolorization by fusion," in *Proc. ACCV*, Queenstown, New Zealand, 2010, pp. 79–92.
[17] C. O. Ancuti, C. Ancuti, C. Hermans, and P. Bekaert, "Fusion-based image and video decolorization," in *Proc. SIGGRAPH ASIA Sketches*, Seoul, South Korea, 2010, Art. no. 44.
[18] Y. Kim, C. Jang, J. Demouth, and S. Lee, "Robust color-to-gray via nonlinear global mapping," *ACM Trans. Graph.*, vol. 28, no. 5, pp. 1–4, 2009.

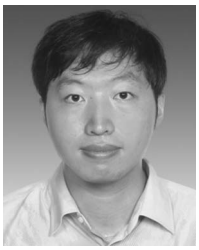
[19] Q. Liu, P. Liu, Y. Wang, and H. Leung, "Semi-parametric decolorization with Laplacian-based perceptual quality metric," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 27, no. 99, pp. 1–12, Apr. 2017.
[20] Y. Song, L. Bao, X. Xu, and Q. Yang, "Decolorization: Is rgb2gray() out?" in *Proc. Siggraph Asia Tech. Briefs*, Hong Kong, 2013, Art. no. 15.
[21] C. Lu, L. Xu, and J. Jia, "Contrast preserving decolorization," in *Proc. ICCP*, Seattle, WA, USA, 2012, pp. 1–7.
[22] C. Lu, L. Xu, and J. Jia, "Real-time contrast preserving decolorization," in *Proc. SIGGRAPH Asia Tech. Briefs*, Singapore, 2012, pp. 1–4.
[23] H. Du, S. He, B. Sheng, L. Ma, and R. W. H. Lau, "Saliency-guided color-to-gray conversion using region-based optimization," *IEEE Trans. Image Process.*, vol. 24, no. 1, pp. 434–443, Jan. 2015.
[24] A. Hourunranta, A. Islam, and F. Chebil, "Video and audio editing for mobile applications," in *Proc. ICME*, Toronto, ON, Canada, 2006, pp. 1305–1308.
[25] C. L. Madhwacharyula, M. S. Kankanhalli, and P. Millhem, "Content based editing of semantic video metadata," in *Proc. ICME*, Taipei, Taiwan, 2004, pp. 33–36.
[26] T. Kimura, K. Sumiya, and H. Tanaka, "A video editing support system using users' gazes," in *Proc. PACRIM*, Victoria, BC, Canada, 2005, pp. 149–152.
[27] Y. Tao, Y. Shen, B. Sheng, P. Li, and E. Wu, "Temporal coherent video decolorization using proximity optimization," in *Proc. CGI*, Heraklion, Greece, 2016, pp. 41–44.
[28] R. Ogata, Y. Nakamura, and Y. Ohta, "Computational video editing model based on optimization with constraint-satisfaction," in *Proc. ICICS-PCM*, Singapore, 2003, pp. 688–693.
[29] C. Cuevas and N. Garcia, "Temporal segmentation tool for high-quality real-time video editing software," *IEEE Trans. Consum. Electron.*, vol. 58, no. 3, pp. 917–925, Aug. 2012.
[30] H. Wang, N. Xu, R. Raskar, and N. Ahuja, "Videoshop: A new framework for spatio-temporal video editing in gradient domain," in *Proc. CVPR*, San Diego, CA, USA, 2005, p. 1201.
[31] J. Chen and C.-K. Tang, "Spatio-temporal Markov random field for video denoising," in *Proc. CVPR*, Minneapolis, MN, USA, 2007, pp. 1–8.
[32] S.-Y. Lin and J.-Y. Shi, "A Markov random field model-based approach to natural image matting," *J. Comput. Sci. Technol.*, vol. 22, no. 1, pp. 161–167, 2007.
[33] J. Zhang *et al.*, "Video dehazing with spatial and temporal coherence," *Vis. Comput.*, vol. 27, no. 6, pp. 749–757, 2011.
[34] P. Viola and W. M. Wells, III, "Alignment by maximization of mutual information," *Int. J. Comput. Vis.*, vol. 24, no. 2, pp. 137–154, 1997.
[35] M.-H. Lim and P. C. Yuen, "Entropy measurement for biometric verification systems," *IEEE Trans. Cybern.*, vol. 46, no. 5, pp. 1065–1077, May 2016.
[36] Z. Wang, A. C. Bovik, H. R. Sheikh, and E. P. Simoncelli, "Image quality assessment: From error visibility to structural similarity," *IEEE Trans. Image Process.*, vol. 13, no. 4, pp. 600–612, Apr. 2004.
[37] T. M. Nguyen and Q. M. J. Wu, "A nonsymmetric mixture model for unsupervised image segmentation," *IEEE Trans. Cybern.*, vol. 43, no. 2, pp. 751–765, Apr. 2013.
[38] T. Li *et al.*, "Data-driven affective filtering for images and videos," *IEEE Trans. Cybern.*, vol. 45, no. 10, pp. 2336–2349, Oct. 2015.
[39] J.-K. Ahn and C.-S. Kim, "Real-time segmentation of objects from video sequences with non-stationary backgrounds using spatio-temporal coherence," in *Proc. ICIP*, San Diego, CA, USA, 2008, pp. 1544–1547.
[40] C. Lu, L. Xu, and J. Jia, "Contrast preserving decolorization with perception-based quality metrics," *Int. J. Comput. Vis.*, vol. 110, no. 2, pp. 222–239, 2014.
[41] K. Ma, T. Zhao, K. Zeng, and Z. Wang, "Objective quality assessment for color-to-gray image conversion," *IEEE Trans. Image Process.*, vol. 24, no. 12, pp. 4673–4685, Dec. 2015.

Yizhang Tao is currently pursuing the graduation degree with the Department of Computer Science and Electronic Engineering, Shanghai Jiao Tong University, Shanghai, China.

He is with the Visual Media and Data Management Laboratory, Department of Computer Science and Engineering, Shanghai Jiao Tong University. His current research interests include machine learning, virtual reality, and computer vision.

Yiyi Shen is currently pursuing the graduation degree with the Department of Computer Science and Electronic Engineering, Shanghai Jiao Tong University, Shanghai, China.

She is a member of the Visual Media and Data Management Laboratory, Department of Computer Science and Engineering, Shanghai Jiao Tong University. Her current research interests include data mining, deep learning, and image processing and analysis.



Bin Sheng received the B.A. degree in English and B.E. degree in computer science from the Huazhong University of Science and Technology, Wuhan, China, in 2004, the M.S. degree in software engineering from the University of Macau, Macau, China, in 2007, and the Ph.D. degree in computer science from the Chinese University of Hong Kong, Hong Kong, in 2011.

He is currently an Associate Professor with the Department of Computer Science and Engineering, Shanghai Jiao Tong University, Shanghai, China. His current research interests include virtual reality and computer graphics.

Ping Li received the Ph.D. degree from the Chinese University of Hong Kong, Hong Kong.

He is currently a Lecturer with the Education University of Hong Kong, Hong Kong. His current research interests include image/video stylization, learning analytics, big data visualization, computer graphics, and creative media.

Rynson W. H. Lau received the Ph.D. degree from the University of Cambridge, Cambridge, U.K.

He was with the Faculty of Durham University, Durham, U.K. and is currently with the City University of Hong Kong, Hong Kong. His current research interests include computer graphics and computer vision.

Dr. Lau serves on the editorial board of *Computer Animation and Virtual Worlds*. He has served as the Guest Editor of a number of journal special issues, including the ACM TRANSACTIONS ON INTERNET TECHNOLOGY, the IEEE TRANSACTIONS ON MULTIMEDIA, the IEEE TRANSACTIONS ON VISUALIZATION AND COMPUTER GRAPHICS, and the IEEE COMPUTER GRAPHICS & APPLICATIONS. He has also served in the committee of a number of conferences, including the Program Co-Chair of ACM VRST 2004, ACM MTDL 2009, and IEEE U-Media 2010, and the Conference Co-Chair of CASA 2005, ACM VRST 2005, ACM MDI 2009, and ACM VRST 2014.