Spatiotemporal Colorization of Video Using 3D Steerable Pyramids

Somdyuti Paul, Saumik Bhattacharya, and Sumana Gupta, *Member, IEEE*

***Abstract*— We propose a new technique for video colorization based on spatiotemporal color propagation in the 3D video volume, utilizing the dominant orientation response obtained from the steerable pyramid decomposition of the video. The volumetric color diffusion from the sources that are marked by scribbles occurs across spatiotemporally smooth regions, and the prevention of leakage is facilitated by the spatiotemporal discontinuities in the output of steerable filters, representing the object boundaries and motion boundaries. Unlike most existing methods, our approach dispenses with the need of motion vectors for interframe color transfer and provides a general framework for image and video colorization. The experimental results establish the effectiveness of the proposed approach in colorizing videos having different types of motion and visual content even in the presence of occlusion, in terms of accuracy and computational requirements.**

***Index Terms*— 3D steerable pyramid, dominant orientation, occlusion handling, prioritization, spatiotemporal color propagation.**

I. Introduction

COLORIZATION refers to the process of assigning meaningful colors to the input monochrome image or video sequence to produce a fully colored image or video as the output. This process is of utmost importance in commercial digital media editing since the addition of colors to a monochrome image or video greatly improves its perceptual quality. The term *colorization* was coined by Wilson Markle in 1970 to refer to the computerized process of adding colors to black and white movies. Since then, image colorization has become one of the most widely explored areas of research in the field of image processing. However, perhaps due to its complexity and greater processing costs, video colorization has received considerably less research attention so far and most approaches handle video colorization by falling back on image colorization techniques, i.e., by colorizing each keyframe individually, followed by color transfer to other frames, using motion vectors.

The primary objective of colorization is to generate a 3D data consisting of the intensity information and two color channels from the given single dimensional data representing the pixel intensities. This transformation is not unique and thereby colorization is, in general, an ill-posed problem. The earliest methods of colorization were based on luminance keying, which maps pixel intensities to color values using a lookup table [1]. Such an approach obviously does not provide the flexibility of mapping the same intensity value to different colors and also fails for images with complex textures as it ignores the texture information. Several techniques were developed that utilize a reference color image to transfer appropriate colors to a similar grayscale image by matching intensity as well as texture information [2]–[6]. However, such approaches are restrictive as they depend on the availability of a reference color image having similar characteristics with the given input image, do not allow the user to control the assigned colors, and are often suitable for a specific class of images such as cartoons [5], [6]. Further, such reference image-based techniques often rely on a reliable segmentation of an image, which is difficult, especially for images with complex textures [3]. Hence, accuracy of color transfer is severely affected by segmentation errors. All these factors led to the emergence of scribble-based colorization techniques that allow precise control over the color to be assigned to each region by marking the input image with color scribbles [7]. The colors from the scribbles are algorithmically propagated to the noncolored pixels. A breakthrough in the field of colorization was provided by the global optimization-based colorization approach developed in [8], wherein the l_2 norm of the difference between the color of each pixel and the weighted sum of the colors of its neighbors is minimized in order to assign color values to each pixel. Subsequently, other optimization-based colorization frameworks like a mixed l_0/l_1 norm minimization [9] and a hybrid scheme [10] combining the advantages of scribble-based and example-based approaches were developed to improve the performance of the method of [8] using fewer scribbles. These optimization-based techniques rely on the premise that neighboring pixels with similar intensity should have similar color. This assumption, however, fails for textured images. The colorization technique of [11] derives the color of a pixel by blending the scribble colors as a weighted sum, in which the weight assigned to a color is computed using a minimum geodesic distance of the given pixel from that color. A prioritized source propagation-based colorization approach was introduced in [12], in which scribble colors were propagated using a weighted sum approach based on a priority measure derived from the accuracies and intensity values of neighbors of a pixel. Since object boundaries or edge information is not explicitly taken into account in all the scribble-based techniques discussed so far, color leakage across weak boundaries is inevitable. In [13], the technique

Manuscript received September 8, 2015; revised December 3, 2015 and January 12, 2016; accepted March 1, 2016. Date of publication March 8, 2016; date of current version August 2, 2017. This paper was recommended by Associate Editor C. Zhang.

The authors are with the Department of Electrical Engineering, IIT Kanpur, Kanpur 208016, India (e-mail: somdyutip@gmail.com; saumik@iitk.ac.in; sumana@iitk.ac.in).

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/TCSVT.2016.2539539

of [12] was further developed to handle this problem by introducing a Gaussian pyramid of gradient images for priority computation. In this approach, larger regions with smoother gradients are colorized first at the top level of the pyramid, and regions near object boundaries are colorized later at lower levels in the decreasing order of priority at each level.

Although many of the image colorization algorithms can be readily extended to video through temporal color transfer with the aid of motion vectors [13]–[15], very few techniques have been developed that treat video colorization as a spatiotemporal color propagation problem. Most of the conventional techniques also ignore the problem of color transfer to occluded areas of a video sequence, which cannot be accomplished by unidirectional color transfer between frames. The optimization-based approach of [8] is also applied for video colorization by performing the optimization over the 3D video volume. The method, however, suffers from the inaccuracies of optical flow-based motion vector estimation, which is employed to determine the temporal neighborhood of a pixel, causing color leakage across boundaries of moving objects between frames, although such effects can be mitigated by employing more advanced optical flow techniques that have been proposed recently for handling large motions and occlusion [16], [17]. In [11], the approach is extended to video colorization by generalizing the computation of the geodesic distances in three dimensions, with the inclusion of the time axis. In both these techniques, occluded areas are erroneously colored due to the unidirectional color flow. Moreover, color bleeding persists due to lack of edge information as already discussed. A video colorization method over the 3D volume based on optimization in the rotation-aware Gabor feature space was proposed in [18], which utilizes the Gabor filtering-based framework of [19] to transform each pixel of the video sequence to a higher dimensional feature space, followed by adaptive clustering over the feature space to generate connected subgraphs. Gabor flow is used to establish temporal pixel correspondences. Nevertheless, the feature space clustering and optimization steps are computationally intensive due to the high dimensionality of the Gabor feature space data.

In this paper, we propose a spatiotemporal steerable pyramid decomposition of the video sequence, which enables us to properly accentuate the spatiotemporal edges oriented at any arbitrary angle. On the contrary, the limitation of the Gaussian pyramid of orthogonal gradients lies in the fact that it is capable of strongly highlighting only horizontal and vertical edges, but fails to significantly enhance weak angular edges. We adopt the prioritized source propagation approach of [13] to reliably propagate colors of the scribbles to the other parts of the video. Since the priorities in our work are based on the steerable pyramid obtained from the video volume, the possibility of color leakage across weak edges is effectively eliminated. Linear filters oriented in space-time were applied to extract motion information in the form of oriented spatiotemporal energy in [20], which made steerable filters a useful tool for motion analysis. Subsequently, spatiotemporal steerable pyramids were employed in several important applications such as visual tracking [21] and action recognition [22], [23]. In the spatial domain, steerable pyramid serves as an efficient texture descriptor as explored in [24]. Thus, spatiotemporal steerable pyramid can be exploited to yield a robust descriptor of motion as well as texture, which makes it potentially useful for video colorization. Our approach takes both motion as well as texture into account through a novel usage of the spatiotemporal edges of the pyramid, which allows accurate colors to be propagated within the video volume, thereby avoiding the inaccuracies inherent in motion vector estimation and the additional computational burden of color refinement. In effect, colors are allowed to diffuse outward in a 3D volume from the source pixels to similar regions present in the entire volume. Color diffusion to dissimilar regions in the 3D space is prevented by the barriers provided by spatiotemporal edges in the 3D pyramid. Our approach also effectively handles occlusion by forward and backward color propagation within the video volume. The principal contributions made in this paper are as follows:

1) a technique for spatiotemporal color propagation over the entire video volume;
2) temporal color transfer among video frames without relying on motion vector estimation;
3) a strategy for efficient occlusion handling using bidirectional color propagation;
4) a generalized framework that supports both image and video colorization.

These points will be further elaborated in the remaining sections. The organization of the rest of this paper is as follows. Section II provides a brief overview of our colorization framework. Section III formally describes the proposed colorization algorithm. Section IV presents the experimental results obtained by applying the proposed approach for the colorization of several videos and images. Section V describes our segmentation and recolorization process. Finally, the conclusion is drawn in Section VI.

II. Overview of Methodology

The video colorization approach developed in this paper propagates color spatiotemporally over the entire video volume, exploiting the motion cues and intensity dissimilarity cues obtained from the steerable pyramid decomposition of the video. In contrast to the existing video colorization techniques, this approach eliminates the necessity of sequentially coloring the video frames by transferring colors from the keyframes with the help of motion vectors. Motion vector estimation is often error prone due to the global intensity fluctuations among different frames of the same video shot. Thus, by dispensing with the usage of motion vectors, our technique not only avoids the color artifacts arising due to inaccuracies of motion vectors, but also alleviates the additional computational cost of motion vector estimation, making it fast and robust to cross frame color artifacts.

Faithful colorization of the occluded areas in a video sequence is an intriguing challenge that has received little attention so far. In conventional video colorization techniques that unidirectionally propagate colors from a keyframe to the subsequent frames, the occluded areas incur colorization errors since they do not have a correspondence with the keyframe

Fig. 1. Block diagram showing steps of the colorization approach.

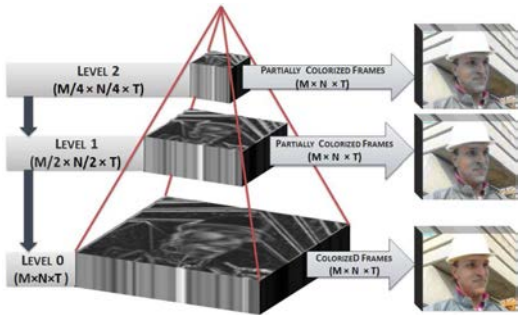

Fig. 2. Three-level pyramid illustrating level-by-level color propagation for a video volume of size $M \times N \times T$.

from which they receive the color from. In the proposed technique, each frame of the video sequence receives its color simultaneously from two keyframes: one that lies ahead of it in time, and is referred to as the forward keyframe, and the other that lies behind it in time, and is referred to as the backward keyframe. Regions of a given frame that were covered in the backward keyframe will be uncovered in the forward keyframe, and vice versa. Thus, every pixel in any intermediate frame will have a correspondence with at least one of the keyframes, and hence all occluded regions of the intermediate frames can be faithfully colorized.

The processing stages of the proposed technique are illustrated with the help of Fig. 1. Color scribbles are marked by the user on the selected keyframes. A steerable pyramid is then constructed using the outputs of a bank of steerable filters, steered to a specified number of orientations. At each level of the pyramid, the dominant orientation response is considered at each pixel in order to perfectly capture all the spatiotemporal edges having different angular orientations. At each pyramid level except the bottom, pixels are assigned priorities according to its dominant orientation response and accuracies of spatiotemporal neighbors. Starting with the topmost level of the pyramid, colorization proceeds in the decreasing order of priorities. In this process, color propagation takes place to large spatiotemporally smooth regions first at the top level of the pyramid. Regions at or near spatiotemporal edges are colorized later while processing the subsequent pyramid levels. Fig. 2 depicts this pyramidal colorization scheme. The approach suppresses color leakages across weak spatiotemporal edges having all possible angular orientations, which are inherent in any video sequence. Since object boundaries as well as motion boundaries manifest themselves as spatiotemporal edges, intra-frame leakage across object boundaries as well as inter-frame leakage due to motion is effectively suppressed.

III. Proposed Algorithm

The input to the algorithm is a shot of a grayscale video sequence $\mathbf{V}$ of size $M \times N \times T$, where each frame is of size $M \times N$ and T is the number of frames in the shot. First of all, keyframe selection is performed as described in Appendix B. The user scribbles the desired colors on a few keyframes, which form a small subset of the total frames. The first and last frames of $\mathbf{V}$ are always taken as keyframes to ensure bidirectional color propagation to every frame. There can be any number of keyframes between the first and last frames depending on the video content. This algorithm uses the YIQ color space, although YUV or YCbCr color spaces can also be used [11]. However, other color spaces like HSV (hue saturation value) give rise to visible color distortion due to its highly nonlinear relationship to the RGB (Red Green Blue) space [25], [26]. Let $\mathbf{v}$ be a pixel at location (x, y, t) in the video clip, where x and y are the spatial coordinates and t is the frame number. The luminance of pixel $\mathbf{v}$ is denoted by $Y(\mathbf{v})$ and its color information is given by the vector $\mathbf{C}(\mathbf{v}) = [I(\mathbf{v})\ \ Q(\mathbf{v})]$. The color accuracy of a pixel is defined as

$$a(\mathbf{v}) = \begin{cases} 1, & \text{if } I(\mathbf{v}) \neq 0 \text{ or } Q(\mathbf{v}) \neq 0 \\ 0, & \text{otherwise.} \end{cases} \tag{1}$$

Thus, initially only the pixels that correspond to the user's scribbles have accuracy 1, and all other pixels have accuracy 0. As soon as a pixel is colorized, its accuracy is updated to 1. Such pixels having accuracy 1 are regarded as color sources for colorization of the remaining pixels at later stages of the colorization process.

A. Construction of Steerable Filters

The steerable pyramid decomposition of the video is performed in order to identify the spatiotemporal smoothness of regions within the video volume, which are used to assign priorities in the subsequent stages. An architecture for construction of steerable filters in 2D was put forth in [27]. Extending this architecture to three dimensions, the detailed construction of 3D separable steerable filters is presented in [28], in which it is assumed that the functions to be steered have an axis of rotational symmetry, which point along the direction cosines α, β, and γ on being rotated by a transformation $\mathbf{\Omega}$, and can be written in the form

$$f^{\mathbf{\Omega}}(x, y, t) = W(r) P_J(x') \tag{2}$$

where $r = (x^2 + y^2 + t^2)^{1/2}$, $W(r)$ is a spherically symmetric windowing function, $x' = \alpha x + \beta y + \gamma t$, and $P_J(x')$ is a Jth degree polynomial in x'.

In order to steer $f^{\mathbf{\Omega}}(x, y, t)$ along any arbitrary direction given by the general transformation $\mathbf{\Omega}$, a set of basis functions $f^{\mathbf{\Omega}_j}(x, y, t)$ having direction cosines α_j, β_j, and γ_j have to be linearly combined according to the steering equation given by

$$f^{\mathbf{\Omega}}(x, y, t) = \sum_{j=1}^{K} k_j(\alpha, \beta, \gamma)\, f^{\mathbf{\Omega}_j}(x, y, t) \tag{3}$$

where $k_j(\alpha, \beta, \gamma)$ are the orientation-dependent steering coefficients. Equation (3) holds if and only if [27], [28]

Fig. 3. Basis filters of G_2 and their responses. Top row: a temporal slice of each of the six basis filter at $t = 6$. Bottom row: the corresponding temporal slice of the 3D outputs for ten frames of the *Foreman* sequence.

$K \geq (J+1)(J+2)/2$, and $k_j(\alpha, \beta, \gamma)$ satisfies

$$\begin{pmatrix} \alpha^J \\ \alpha^{J-1}\beta \\ \alpha^{J-1}\gamma \\ \alpha^{J-2}\beta^2 \\ \vdots \\ \gamma^J \end{pmatrix} = \begin{pmatrix} \alpha_1^J & \alpha_2^J & \cdots & \alpha_K^J \\ \alpha_1^{J-1}\beta & \alpha_2^{J-1}\beta & \cdots & \alpha_K^{J-1}\beta \\ \alpha_1^{J-1}\gamma & \alpha_2^{J-1}\gamma & \cdots & \alpha_K^{J-1}\gamma \\ \alpha_1^{J-2}\beta^2 & \alpha_2^{J-2}\beta^2 & \cdots & \alpha_K^{J-2}\beta^2 \\ \vdots & \vdots & \vdots & \vdots \\ \gamma_1^J & \gamma_2^J & \cdots & \gamma_K^J \end{pmatrix} \times \begin{pmatrix} k_1(\alpha, \beta, \gamma) \\ k_2(\alpha, \beta, \gamma) \\ k_3(\alpha, \beta, \gamma) \\ \vdots \\ k_K(\alpha, \beta, \gamma) \end{pmatrix}. \quad (4)$$

In this paper, we consider the second derivative of Gaussian $G_2(x, y, t)$ and its Hilbert transform $H_2(x, y, t)$ to form a quadrature pair of steerable functions. The 3D Gaussian-like function is given by

$$G(x, y, t) = e^{-(x^2+y^2+t^2)}. \quad (5)$$

The second derivative of Gaussian with respect to x and its Hilbert transform are given by [28]

$$\begin{aligned} G_2(x, y, t) &= (4x^2 - 2)e^{-(x^2+y^2+t^2)} \\ H_2(x, y, t) &= (-2.254x + x^3)e^{-(x^2+y^2+t^2)}. \end{aligned} \quad (6)$$

It can be easily established that the functions in (6) are of the form of (2), i.e., they can be expressed as a Jth degree polynomial times a spherically symmetric windowing function. According to (4), for G_2, $J = 2$, so six basis functions are required, whereas for H_2, $J = 3$, and hence ten basis functions are required. The procedure for obtaining the basis functions and the corresponding steering coefficients is elaborated in [28]. The orientation of the filters is sampled by representing the direction cosines of the axis of symmetry in terms of spherical coordinates $(\theta, \phi, \rho = 1)$ as

$$\begin{aligned} \alpha &= \cos(\theta)\sin(\phi) \\ \beta &= \sin(\theta)\sin(\phi) \\ \gamma &= \cos(\phi). \end{aligned} \quad (7)$$

Thus, by changing the values of θ and ϕ, the filter can be steered to any desired orientation. The steerable filters used in this work are based on Tables IV–IX. The basis filters for both inphase and quadrature phase and their outputs are shown in Figs. 3 and 4.

B. Dominant Orientation Response Using Steerable Pyramid

At the bottom level of the pyramid, for each orientation, the video volume is convolved with each of the basis filters and their outputs are linearly combined using the steering coefficients for that particular orientation. Thus, for each orientation, a complex 3D filter response is obtained due to the quadrature pair of steerable filters, where G_2 forms the real part and H_2 forms the imaginary part of the filter response. In order to build the subsequent levels of the pyramid, the input video volume is passed through a 3D low-pass filter and downsampled by a factor of 2 both vertically and horizontally. However, there is no downsampling along the temporal dimension. This is because the sampling density along the temporal dimension is normally much less than the spatial sampling density (usually, the number of frames in a video shot is less than the vertical and horizontal dimensions of each frame) due to which temporally adjacent pixels are relatively less correlated compared with spatially adjacent pixels.

The response of the steerable filters for each orientation is strong for those spatiotemporal edges which have that particular orientation. Thus, if the orientations are sampled at sufficiently close intervals, all edges will have strong response along some orientation. The orientation along which the filter response is maximum for any spatiotemporal edge is called the dominant orientation for that edge. Thus, in order to get a measure of the spatiotemporal smoothness, we consider the filter response at the dominant orientation for each pixel in the video. Let the filter response be taken along a total of O orientations. Then the complex output of the filter at the bottom level $l = 0$ of the pyramid at pixel $\mathbf{v}_0$ is given by

$$X(\mathbf{v}_0) = [X_1(\mathbf{v}_0)\ X_2(\mathbf{v}_0) \cdots X_i(\mathbf{v}_0) \cdots X_O(\mathbf{v}_0)] \quad (8)$$

in which each X_i represents the complex filter output at the ith orientation and is of size $M \times N \times T$. Then, the level 0 of the pyramid at $\mathbf{v}_0$ is constructed as follows:

$$P_0(\mathbf{v}_0) = \max_i\ [|X_1(\mathbf{v}_0)|\ |X_2(\mathbf{v}_0)| \cdots |X_i(\mathbf{v}_0)| \cdots |X_O(\mathbf{v}_0)|] \quad (9)$$

where $|X_i(\mathbf{v}_0)|$ stands for the absolute value of the complex filter response at the ith orientation at pixel $\mathbf{v}_0$. Thus, P_0 will be of size $M \times N \times T$. To construct an L-level pyramid, the video input to the adjacent lower level is passed through a 3D low-pass filter and spatially downsampled by a factor of 2. It is then used as the input to the quadrature pair of steerable filters, steered to the O different orientations, and the same procedure of finding the response at the dominant orientation for each pixel of the reduced video volume is repeated. So, P_1 the level 1 of the pyramid will have size $M/2 \times N/2 \times T$, and for each higher level up to $L-1$, the spatial dimensions of the level will be reduced by a factor of 2 compared with the previous level. Thereby, the size of P_l, the pyramid at level l is $M/2^l \times N/2^l \times T$, and pixel $\mathbf{v}_l(x, y, t)$ in P_l relates to pixel $\mathbf{v}_{l-1} = (2x, 2y, t)$ in P_{l-1} due to the spatial subsampling. Fig. 5 summarizes the pyramid construction. It should be noted that some pixels are likely to have multiple orientations, such as those at corners or junctions, where edges having two or more different orientations meet (e.g., the center of a cross);

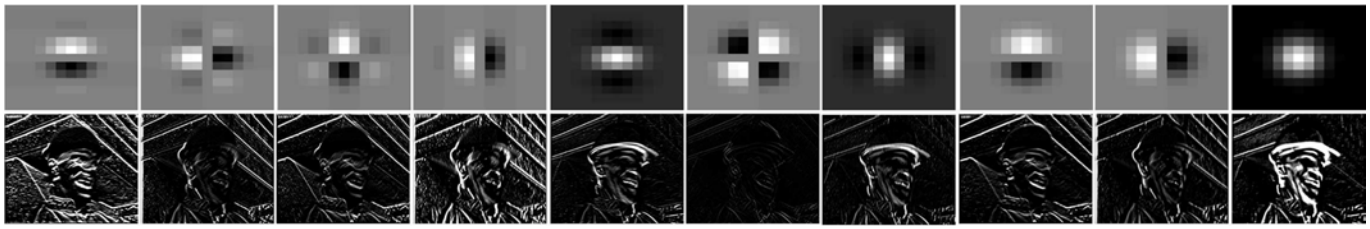

Fig. 4. Basis filters of H_2 and their responses. Top row: a temporal slice of each of the ten 3D basis filters at $t = 6$. Bottom row: the corresponding temporal slice of the 3D outputs for ten frames of the *Foreman* sequence.

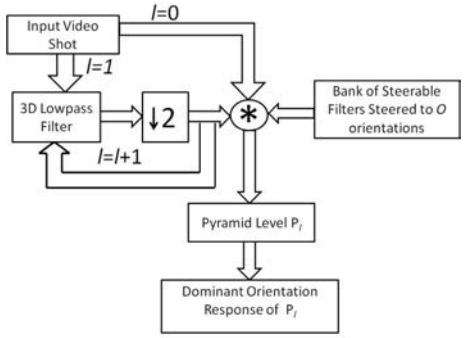

Fig. 5. Block schematic of steerable pyramid decomposition.

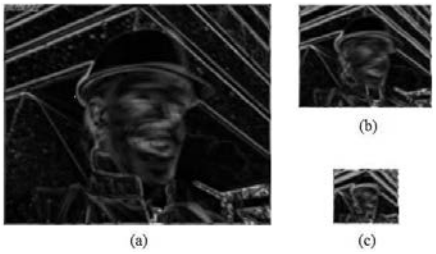

Fig. 6. Temporal slice of a three-level pyramid at $t = 6$. (a) $l = 0$. (b) $l = 1$. (c) $l = 2$.

hence, at these pixels, steerable filtering would give rise to a maximum at more than one orientation. For such pixels, we simply take the filter response at the first maximum as the dominant orientation response, although the response at any maxima can be chosen, as the value of the orientation itself is immaterial for our purpose. For the analysis of multiple orientations, a filter having higher angular resolution such as the fourth derivative of Gaussian would be a more appropriate choice. Since our interest lies in merely obtaining a strong filter response at all points at which there is a spatiotemporal discontinuity, using G_2-H_2 quadrature pair is sufficient. Fig. 6 shows the temporal slices of a three-level 3D pyramid.

To further enhance the weak spatiotemporal edges relative to the stronger ones, we perform histogram equalization of each temporal slice of the normalized dominant orientation responses at each level [1]. The significance of this step is evident from Fig. 7. It can be seen that histogram equalization substantially enhances even the faintest angular edges detected by the steerable filters and thus effectively prevents color leakage.

To illustrate the significance of using spatiotemporal 3D pyramids in capturing weak angular edges, we compare its

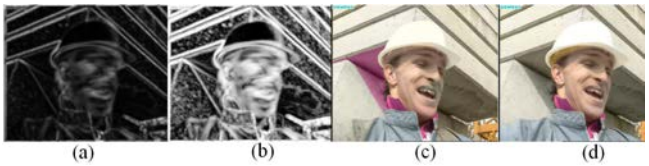

Fig. 7. Effect of histogram equalization. (a) Temporal slice of a three-level pyramid at $t = 6$. (b) Histogram equalization of (a). (c) Colorized frame of the *Foreman* sequence without histogram equalization. (d) Corresponding colorized frame with histogram equalization.

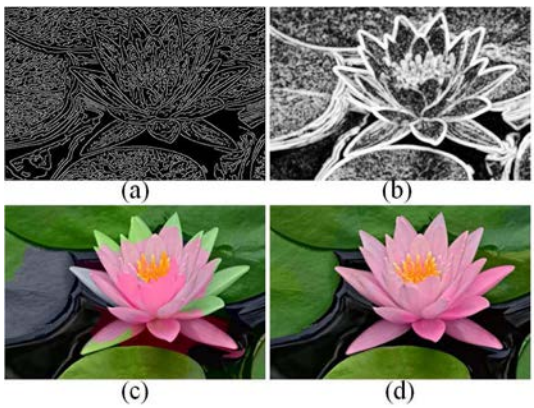

Fig. 8. Comparison of colorization performance obtained using Canny edge detector and dominant orientation response of steerable pyramid. (a) Canny edge response with threshold 0.02. (b) Dominant orientation response at level 0. (c) Colorization with result of (a). (d) Colorization with result of (b).

colorization performance with that obtained using the Canny edge detector in Fig. 8. It is apparent from Fig. 8 that even with a dense edge map obtained with a very small value of threshold, there is still substantial color leakage due to broken edges. On the other hand, the result obtained using dominant orientation response of steerable filters is free from leakage.

C. Prioritization

Each nonsource pixel in the video volume is assigned a priority at each level of the pyramid. Color propagation from source pixels to nonsource pixels takes place in the decreasing order of priorities. At the pyramid level l, $P_l(\mathbf{v}_l)$ represents the smoothness of the spatiotemporal region $\mathbf{R}_l(\mathbf{v}_l)$ given by

$$\mathbf{R}_l(\mathbf{v}_l) = \begin{cases} \Psi((x,y,t), 1\times 1\times 1) & \text{if } l = 0 \\ \Psi((2^l x, 2^l y, t), (2^l+1)\times(2^l+1)\times 3) & \text{if } l \geq 1. \end{cases} \tag{10}$$

Here, $\Psi((x,y,t), m \times n \times \tau)$ is the $m \times n \times \tau$ spatiotemporal neighborhood centered around the pixel (x,y,t) of

the video. The augmented region $\mathbf{S}_l(\mathbf{v}_l)$, which includes the region $\mathbf{R}_l(\mathbf{v}_l)$, is given by

$$\mathbf{S}_l(\mathbf{v}_l) = \begin{cases} \Psi((x,y,t), 3\times 3\times 3) & \text{if } l = 0 \\ \Psi((2^l x, 2^l y, t), (2^l+3)\times(2^l+3)\times 3) & \text{if } l \geq 1. \end{cases} \tag{11}$$

It is to be noted that unlike the spatial dimensions, the size of the regions $\mathbf{R}_l(\mathbf{v}_l)$ as well as $\mathbf{S}_l(\mathbf{v}_l)$ does not increase in the temporal dimension as the level number l increases since the pyramid construction involves no subsampling along the temporal dimension. Following the same reasoning, both the temporal neighborhoods $\mathbf{R}_l(\mathbf{v}_l)$ and $\mathbf{S}_l(\mathbf{v}_l)$ of pixel $\mathbf{v}_l(x,y,t)$ are simply the three frame neighborhoods spanning the previous frame at $t-1$, the current frame at t, and the next frame at $t+1$. The computation of priorities is done over the augmented region $\mathbf{S}_l(\mathbf{v}_l)$. We follow the approach of [13] to define the priorities for each pixel of the 3D video volume as

$$\mu_l(\mathbf{v}_l) = \begin{cases} \sum\limits_{\mathbf{u}\in \mathbf{S}_0(\mathbf{v}_0)} a(\mathbf{u}) w_{\mathbf{v}_0,\mathbf{u}} & \text{if } l = 0 \\ \sum\limits_{\mathbf{u}\in \mathbf{S}_l(\mathbf{v}_l)} a(\mathbf{u}) e^{-\zeta P_l(\mathbf{v}_l)} & \text{if } l \geq 1. \end{cases} \tag{12}$$

At level 0, intuitively, a higher priority should be assigned to a pixel if its neighbors have higher accuracy and similar intensity values. In this regard, $w_{\mathbf{v}_0,\mathbf{u}}$ forms a measure of intensity similarity of pixel $\mathbf{v}_0$ with its neighbor $\mathbf{u}$ and is given by

$$w_{\mathbf{v}_0,\mathbf{u}} = e^{-\zeta |Y(\mathbf{v}_0) - Y(\mathbf{u})|}. \tag{13}$$

The accuracy value of each neighbor is weighted by this intensity similarity measure for the corresponding neighbor and summed over the spatiotemporal augmented neighborhood $\mathbf{S}_0(\mathbf{v}_0)$ to give the pixel priorities for level 0. Here, the parameter ζ controls the exponential weights in (12) and (13).

For level $l \geq 0$, pixel $\mathbf{v}_l$ of the video volume is assigned a higher priority if the augmented region $\mathbf{S}_l$ contains more source pixels and $P_l(\mathbf{v}_l)$, the magnitude of the lth level of the pyramid at $\mathbf{v}_l$ is smaller, which implies greater spatiotemporal smoothness of the region $\mathbf{R}_l(\mathbf{v}_l)$.

D. Spatiotemporal Color Propagation

The colorization process proceeds from the top level toward the bottom level of the 3D pyramid. In this approach, the largest and spatiotemporally smoothest regions are colorized first at the topmost level of the pyramid, followed by the comparatively less smooth regions at the subsequent levels in the decreasing order of priorities at each level. At the bottom level, pixels that are at or very close to spatiotemporal edges or object boundaries are colorized. Initially, the pixel having the highest priority $\mu_{L-1}(\mathbf{v}_{L-1})$ at the topmost level $L-1$ of the pyramid is chosen. This corresponds to the region $\mathbf{R}_{L-1}(\mathbf{v}_{L-1})$ as described in Section III-C. Therefore, for $l = L-1$, each nonsource pixel $\mathbf{u} \in \mathbf{R}_l(\mathbf{v}_l)$ is assigned color according to the colors of the source pixels present in the augmented region $\mathbf{S}_l(\mathbf{v}_l)$ as follows:

$$\mathbf{C}(\mathbf{u}) = \frac{1}{\sigma} \sum_{\mathbf{z}\in \mathbf{S}_l(\mathbf{v}_l)} a(\mathbf{z}) w_{\mathbf{u},\mathbf{z}} \mathbf{C}(\mathbf{z}). \tag{14}$$

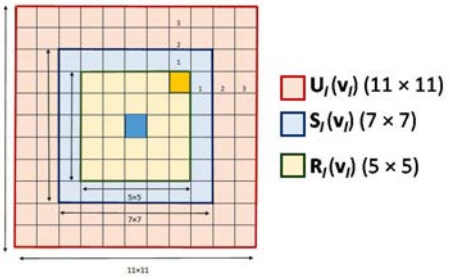

Fig. 9. Colored boxes showing the spatial dimensions of the window $\mathbf{U}_l(\mathbf{v}_l)$ relative to the sizes of regions $\mathbf{R}_l(\mathbf{v}_l)$ and $\mathbf{S}_l(\mathbf{v}_l)$ for $l = 2$.

Here, $\sigma = \sum_{\mathbf{z}\in \mathbf{S}_l(\mathbf{v}_l)} a(\mathbf{z}) w_{\mathbf{u},\mathbf{z}}$ is a normalizing factor, and $w_{\mathbf{u},\mathbf{z}}$ is defined as in (13). As soon as pixel $\mathbf{u}$ is colorized, its accuracy $a(\mathbf{u})$ is updated to 1, and its priority is set to 0, i.e., $\mathbf{u}$ now becomes a source pixel. After colorizing all nonsource pixels in $\mathbf{R}_l(\mathbf{v}_l)$, the priorities of all those pixels in the vicinity whose priorities get affected by the colorization of pixels in the window $\mathbf{R}_l(\mathbf{v}_l)$ are updated. At the higher level of the pyramid, the region over which priority is updated corresponds to the spatiotemporal patch of the video defined as

$$\mathbf{U}_l(\mathbf{v}_l) = \begin{cases} \Psi((x,y,t), 3\times 3\times 3) & \text{if } l = 0 \\ \Psi\left((2^l x, 2^l y, t), \left(2\left(\left\lfloor \frac{2^l+1}{2} \right\rfloor + \left\lfloor \frac{2^l+3}{2} \right\rfloor\right)+1\right)\right. & \\ \left. \times\left(2\left(\left\lfloor \frac{2^l+1}{2} \right\rfloor + \left\lfloor \frac{2^l+3}{2} \right\rfloor\right)+1\right)\times 3\right) & \\ & \text{if } l \geq 1. \end{cases} \tag{15}$$

The size of the update window $\mathbf{U}_l(\mathbf{v}_l)$ is dependent on the sizes of the regions $\mathbf{R}_l(\mathbf{v}_l)$ and $\mathbf{S}_l(\mathbf{v}_l)$ as illustrated by Fig. 9, which pictorially depicts the spatial sizes of all the three windows for $l = 2$. The priorities of the farthest pixels that are affected by the colorization of region $\mathbf{R}_l(\mathbf{v}_l)$ are due to four corner pixels of $\mathbf{R}_l(\mathbf{v}_l)$. The corner pixel marked by yellow is situated at a distance of half the size of the region $\mathbf{R}_l(\mathbf{v}_l)$ from the central pixel shown in blue, and it can affect the priorities of pixels within a maximum distance of half the size of the priority computation window $\mathbf{S}_l(\mathbf{v}_l)$ both horizontally and vertically. The priority update then takes place according to (12) for all pixels $\mathbf{u}$, which map to the region $\mathbf{U}_l(\mathbf{v}_l)$. Then, the pixel having the next highest priority at level $l = L-1$ is chosen, and the corresponding region is colorized. This iterative process continues until the priority becomes less than a threshold δ. Then, we move on to the next level $l = L-2$ and repeat the above process of colorization in decreasing order of priority. In this way, we move down the levels of the pyramid, gradually colorizing regions of lesser and lesser spatiotemporal smoothness toward the lower levels. At level $l = 0$, colorization continues until all the pixels in the entire video volume are colorized. It is to be noted that for $t = 1$ or $t = T$, i.e., at the temporal boundaries of a video shot, the temporal dimensions of the windows $\mathbf{R}_l(\mathbf{v}_l)$, $\mathbf{S}_l(\mathbf{v}_l)$, and $\mathbf{U}_l(\mathbf{v}_l)$ have to be taken as 2, encompassing the first frame or the Tth frame and its immediate neighboring frame.

This process automatically suppresses color leakage across object boundaries within a frame as well as motion boundaries

Fig. 10. Colorization of the *Foreman* sequence. (a) Scribbled backward keyframe. (b) and (c) Intermediate grayscale frames. (d) Scribbled forward keyframe. (e)–(h) Left to right: colorization results of [8], [11], [13], and the proposed approach. (i)–(l) Top to bottom: enlarged parts of the 121th frame shown by the boxes outlined in the corresponding colors for [8], [11], [13], and the proposed approach.

between frames, since these correspond to spatiotemporal edges, which are colorized last, at the finest level. Moreover, the use of dominant orientation responses of steerable filters effectively suppresses the problem of leakage across very faint edges. Further, color propagation from source pixels present in both the forward and backward keyframes effectively counters the occlusion problem as discussed earlier.

E. Image Colorization

We consider image colorization as a special case of video colorization. Specifically, when the number of frames $T = 1$, video colorization reduces to an image colorization problem. Thus, an image can be treated as a video having a single frame. Now, size of $\mathbf{V}$ becomes $M \times N \times 1$, and the location of pixel $\mathbf{v}$ is given by $(x, y, 1)$. The user scribbles the desired colors on the image. The image is subjected to steerable filtering using a temporal slice of the 3D steerable filters. In this way, 3D filters are used for image filtering to keep the framework consistent for both video and image colorization and to avoid the design of additional 2D filters for images. The dominant orientation response of the steerable filter at each level now represents the spatial smoothness instead of spatiotemporal smoothness, and thereafter, pyramids are constructed in an identical fashion to that of the video pyramids. The colorization process also proceeds identically. Again, the spatiotemporal neighborhoods $\mathbf{R}_l(\mathbf{v}_l)$ and $\mathbf{S}_l(\mathbf{v}_l)$ defined by (10) and (11), respectively, reduce to spatial neighborhoods, and colorization takes place in the decreasing order of priorities at each level. Using (14), the color of a pixel $\mathbf{u}$ is obtained from the color of the source pixels present in its spatial neighborhood $\mathbf{S}_l(\mathbf{v}_l)$.

IV. Experimental Results

In order to experimentally evaluate the performance of our algorithm, we use 13 tap basis filters to construct the 3D steerable filters. The complex filters are designed in accordance with Tables IV–IX, which provide the tap coefficients of the 1D filters that are used to construct the separable quadrature pair of basis filters, along with the steering coefficient for each basis. The dominant orientation response is found from the output of the steerable filters steered to 90 different orientations by uniformly varying θ and ϕ in the 3D space. These many orientations are found to be sufficient to adequately capture the motion and texture information manifested by spatiotemporal edges for most of the video sequences encountered in practice.

For our experiments, we have taken $L = 3$, i.e., the dominant orientation response is used to build a three-level pyramid. The parameter ζ is experimentally set to 50 in our work, and the threshold δ is set to 0.1.

A. Video Colorization

We present the performance of the proposed algorithm for different types of motion such as translational, zoom, and affine motion, even in the presence of occlusion.[1] Fig. 10 compares the colorization performance of the proposed approach with that of [8], [11], and [13] for the *Foreman* sequence having affine motion in the form of rotation.

[1]Results of video colorization and recolorization are available at https://goo.gl/57dGhm.

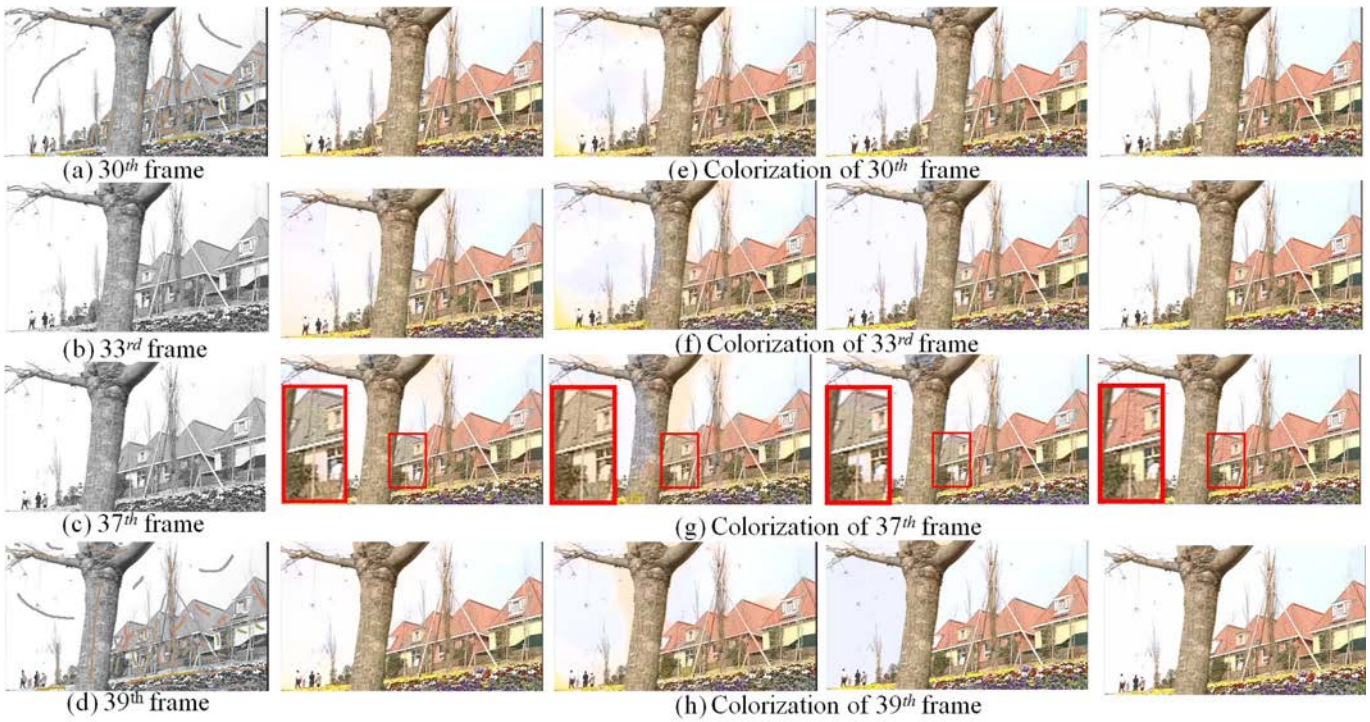

Fig. 11. Colorization of the *Flower* sequence. (a) Scribbled backward keyframe. (b) and (c) Intermediate grayscale frames. (d) Scribbled forward keyframes. (e)–(h) From left to right: colorization results of [8], [11], [13], and the proposed approach. Note that the areas of the 37th frame, which were occluded by the tree in the backward keyframe, as shown by the boxes outlined in red, are transferred correct colors from the forward keyframe in our approach.

Optical flow or block matching-based motion vector estimation techniques fail for affine motion. The areas encompassed by the boxes outlined in four different colors in the colorized 121th frame are enlarged, which clearly illustrates the error incurred by the other methods in the form of color bleeding, which is absent in the colorization result of the proposed technique. The effective occlusion handling strategy of the proposed algorithm is illustrated by the colorization result of the *Flower* sequence in Fig. 11. Our method accurately colorizes the areas in the intermediate frames, which were occluded by the tree in the backward keyframe through color transfer from the forward keyframe in which these areas get uncovered and vice versa. On the other hand, the comparison with the methods of [8], [11], and [13] reveals that the unidirectional color transfer causes occluded regions to incur significant errors in colorization in all these methods.

Fig. 12 illustrates the colorization performance of our algorithm for the *Waterfall* sequence depicting zoom motion compared with the methods of [8], [11], and [13]. Conventional methods of motion vector estimation such as block matching or optical flow fail for zoom motion, and thereby color transfer between frames using motion vectors gives erroneous output. The proposed approach, however, faithfully colorizes the zoomed regions since it only relies on the spatiotemporal discontinuities in the steerable filter outputs produced by zoom motion. The videos showing the colorization results achieved by the proposed technique in comparison to the methods of [8], [11], and [13][1] evince the temporal consistency of the proposed method since the colorized results produced by the other techniques reveal the presence of greater temporal flickering, especially perceptible in the colorization of *Flower* and *Waterfall* sequences.

Fig. 13 illustrates the colorization results for the *Skiing* sequence obtained using our algorithm. It can be readily observed that accurate colors are transferred to the skiing person without employing motion vectors. Figs. 14 and 15 present the colorization results of two more video sequences that illustrate the fact that videos having distinctly different visual content, ranging from natural scenes such as the *Cheetah* sequence to artificial ones such as the *Cartoon* sequence can be effectively colorized using the proposed approach.

B. Image Colorization

To evaluate the performance of our colorization algorithm, we compare the results obtained using the proposed method of image colorization with that of the methods of [8], [11], and [13] with the help of Fig. 16. Fig. 16(b) demonstrates that due to the usage of fewer scribbles, the method of [8] results in color fading, while those of [11] as well as [13] produce color leakage in the flower petal in the colorization of the *Lotus* image. Also, for the *Balloons* image, the method of [8] produces color fading near the boundaries and further causes considerable degradation in the visual quality of the colored image, whereas significant amount of color bleeding across object boundaries in the case of the method of [11] and [13] is evident from Fig. 16(d). In both the cases, the images colorized by our method are, however, free from leakage and color fading artifacts. This reveals that while the other methods require a large number of user scribbles to obtain visually acceptable results, our method can produce

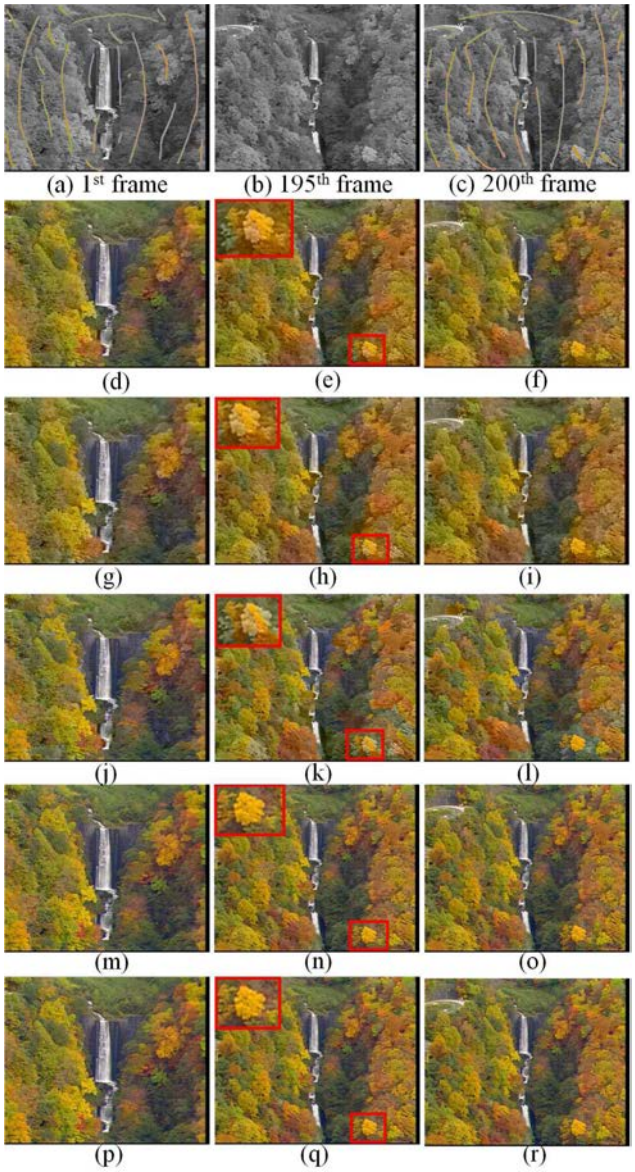

Fig. 12. Colorization of the *Waterfall* sequence. (a) Scribbled backward keyframe. (b) Intermediate grayscale frame. (c) Scribbled forward keyframe. Colorization results of the corresponding frames. (d)–(f) [8], (g)–(i) [11], (j)–(l) [13], and (m)–(o) the proposed approach. (p)–(r) Corresponding ground truth frames. Note that the color quality of the output produced by our method is closest to that of the original compared to the techniques of [8], [11], and [13]. To illustrate this, a small region of the 195th frame is enlarged as shown by the boxes outlined in red.

Fig. 13. Colorization result of the *Skiing* sequence. (a) Scribbled backward keyframe. (b)–(e) Color propagation result of intermediate frames. (f) Scribbled forward keyframe.

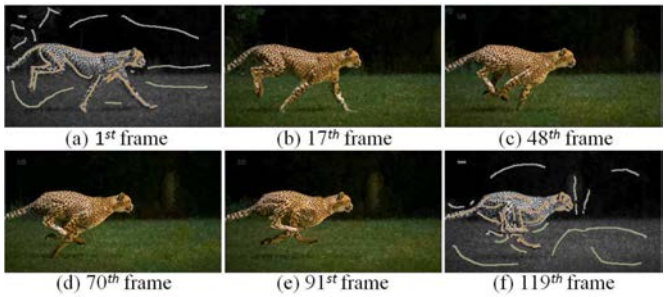

Fig. 14. Colorization result of the *Cheetah* sequence. (a) Scribbled backward keyframe. (b)–(e) Color propagation result of intermediate frames. (f) Scribbled forward keyframe.

appreciable colorization performance with relatively few user scribbles in addition to handling the problem of leakage across weak edges.

C. Analysis

In order to analyze the effect of the number of orientations on algorithm performance, we consider the energy of the dominant orientation response as a measure of the performance of the steerable filter for a particular number of orientations. Since the proposed algorithm requires that spatiotemporal discontinuities are enhanced as strongly as possible to prevent color flow to dissimilar regions, a larger value of the dominant orientation energy implies greater success of the steerable filter in enhancing the spatiotemporal discontinuities. Fig. 17(b) shows the plot of the normalized dominant orientation energies of the steerable pyramid at level 0. For a natural sequence like *Foreman* as well as the synthetic *Circle* image in Fig. 17(a) having many different orientations, the normalized energy increases sharply as the number of orientations initially increases. However, from the plot, it is clear that the energy remains almost constant as the number of orientations further increases. This justifies our choice of 90 orientations as increasing the number of orientations beyond 90 is seen to have little effect on the dominant orientation energies and hence on colorization performance.

The performance of the algorithm is highly influenced by the choice of the parameter ζ, which controls the priority as well as the color update terms in (12)–(14). The colorization results obtained using four different values of ζ are shown in Fig. 18, where the obvious difference in colorization performance elucidates that very large values of ζ result in color contour artifacts, whereas smoother colorization results are obtained on reducing the value of ζ. Using a very small value of ζ, however, lessens the effect of intensity dissimilarity of a pixel with its neighbors and thereby mixes colors of the source pixels present in the spatiotemporal augmented neighborhood with more uniform weights irrespective of their intensity values, which often leads to erroneous results. Fig. 18 establishes the fact that the chosen value of ζ provides a good balance between these two effects and thereby gives good colorization performance.

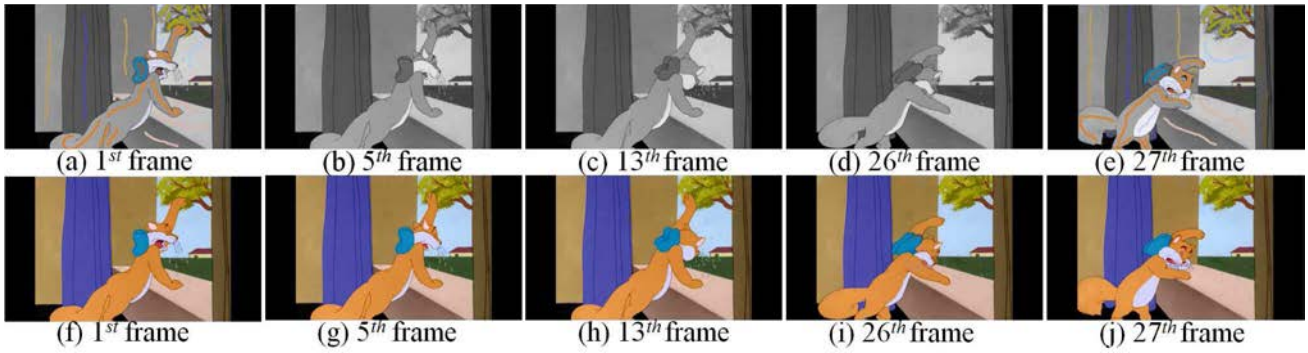

Fig. 15. Colorization result of a black and white *Cartoon* sequence. (a) Scribbled backward keyframe. (b)–(d) Grayscale intermediate frames. (e) Scribbled forward keyframe. (f) Colorized backward keyframe. (g)–(i) Color propagation results of intermediate frames. (j) Colorized forward keyframe.

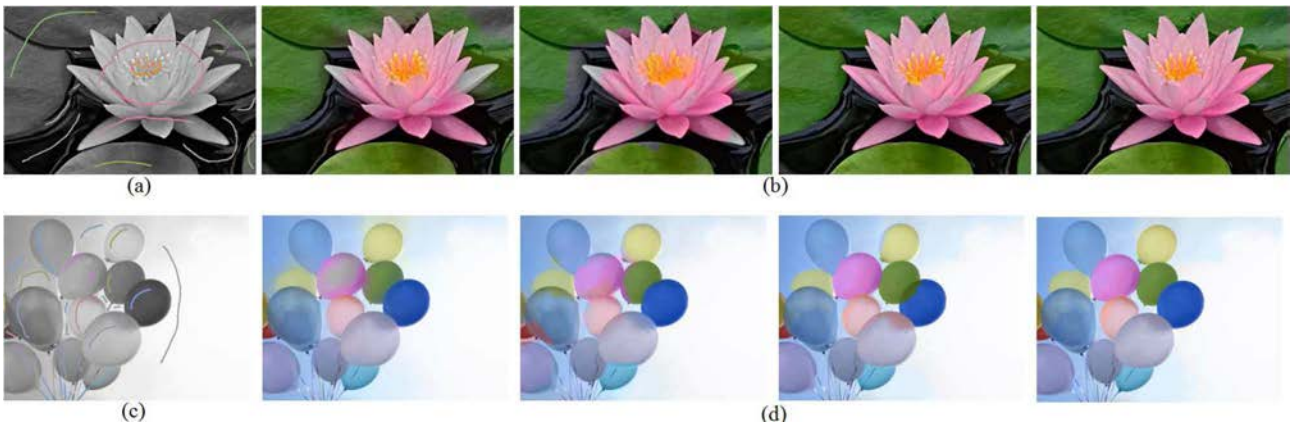

Fig. 16. Colorization of *Lotus* and *Balloons* images. (a) and (c) Scribbled input images. (b) and (d) Left to right: colorization results of [8], [11], [13], and the proposed approach.

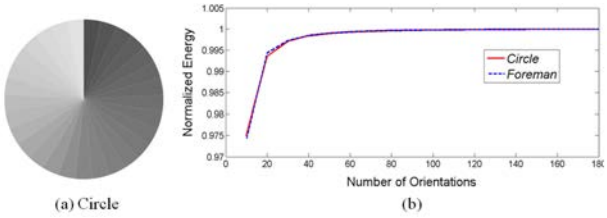

Fig. 17. Effect of number of orientations. (a) Synthetic image having many orientations. (b) Plot showing normalized energy versus number of orientations for *Foreman* (energy per frame) and the image of (a).

The proposed technique is robust to the number and placement of scribbles. This is established by Fig. 19, where we observe that there is no significant deterioration of the colorization result on reducing the number of scribbles as well as altering their positions. In contrast, other colorization techniques such as [8], [11] and [13] perform quite poorly while using the reduced set of initial scribbles as shown by Fig. 10.

D. Performance

Although many colorization algorithms use peak signal to noise ratio (PSNR) as a quantitative measure of colorization performance, it is worth mentioning that PSNR cannot, however, be taken as an absolute measure of the reliability of the colorization process. We believe that colorization quality can, in fact, be best judged by visual assessment of the colorized results. Nevertheless, in the absence of any other suitable quantitative measures, the usage of PSNR does

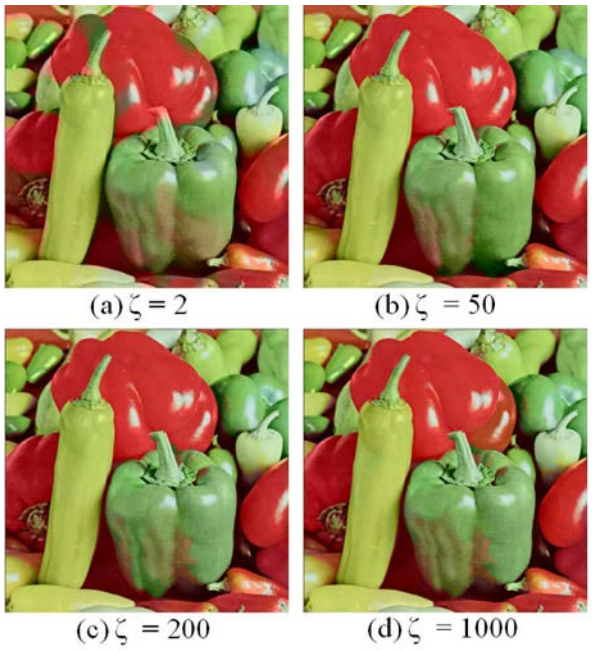

Fig. 18. Colorization result of the *Pepper* image for different values of ζ.

Fig. 19. Effect of varying the number and placement of scribbles. (a) Scribbled keyframe. (b) Same keyframe with reduced scribbles. (c) Intermediate grayscale frame. (d) Colorization result using the scribbles of (a). (e) Colorization result using reduced scribbles of (b).

TABLE I
CPSNR VALUES OF TEST VIDEO SEQUENCES

Sequence	No. of Frames	CPSNR (dB)			
		Levin *et al.* [8]	Yatziv and Sapiro [11]	Hyun *et al.* [13]	Proposed algorithm
Foreman	151	32.2883	28.5604	31.6237	34.1147
Flower	60	25.3654	23.1224	24. 3348	25.0103
Cheetah	119	34.8552	32.7720	34.2452	35.8022
Skiing	100	32.0321	28.4036	31.7252	36.3087

TABLE II
CPSNR VALUES OF TEST IMAGES

Image	CPSNR (dB)			
	Levin *et al.* [8]	Yatziv and Sapiro [11]	Hyun *et al.* [13]	Proposed algorithm
Lotus	26.4750	24.6437	27.0759	29.3498
Balloons	29.2231	26.1201	28.1689	32.0314
Pepper	25.4895	19.6194	25.8390	26.7341

TABLE III
PER FRAME COLORIZATION TIME OF TEST VIDEOS

Sequence	Resolution	Time-per frame (s)			
		Levin *et al.* [8]	Yatziv and Sapiro [11]	Hyun *et al.* [13]	Proposed algorithm
Skiing	360 × 640	64.29	179.21	115.55	46.36
Waterfall	288 × 352	27.00	51.93	37.49	15.13
Flower	200 × 352	16.99	63.07	26.60	14.34
Foreman	288 × 352	24.91	87.23	47.06	15.91
Cheetah	216 × 384	20.57	59.14	34.62	10.16
Cartoon	216 × 384	19.19	61.62	34.59	7.98

provide a relative measure of performance for comparison of different colorization techniques. Tables I and II compare the performance of the proposed algorithm with that of the methods of [8], [11], and [13] for the test videos and images, respectively, using the color PSNR (CPSNR). CPSNR is evaluated by considering the average of the mean square error of the R, G, and B color channels for calculating the PSNR in order to account for the distortion in the color channels with respect to the ground truth [10]. Tables I and II demonstrate that the proposed algorithm also performs better in terms of CPSNR in addition to producing better visual quality of colorized outputs.

We have implemented our algorithm using MATLAB 2014 on an Intel Core i7-2600 3.40-GHz PC with 8-GB RAM. The frame resolution of the video sequences used for our experiments and the per-frame colorization time of the methods of [8], [11], and [13] and the proposed technique is specified in Table III. This shows that our algorithm is considerably faster compared with the other three methods and takes less than a minute to colorize each frame for most videos with standard frame resolution.

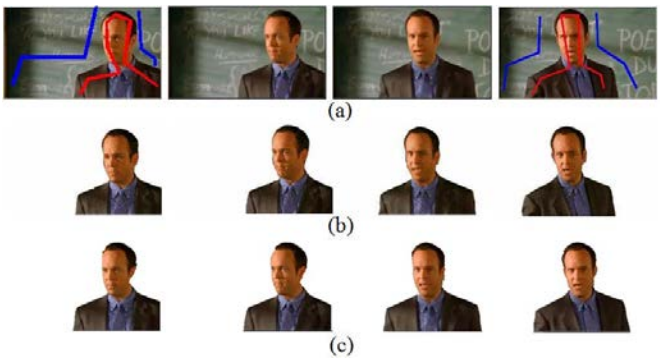

Fig. 20. Segmentation of a video sequence. (a) Original frames along with scribbled keyframes. (b) Segmentation results of [29]. (c) Segmentation results obtained by the proposed approach.

V. SEGMENTATION AND RECOLORIZATION

An image or a video sequence can be selectively recolorized using the framework developed for colorization. In order to achieve this effect, the image or video keyframes are remarked with scribbles using only two colors, namely,

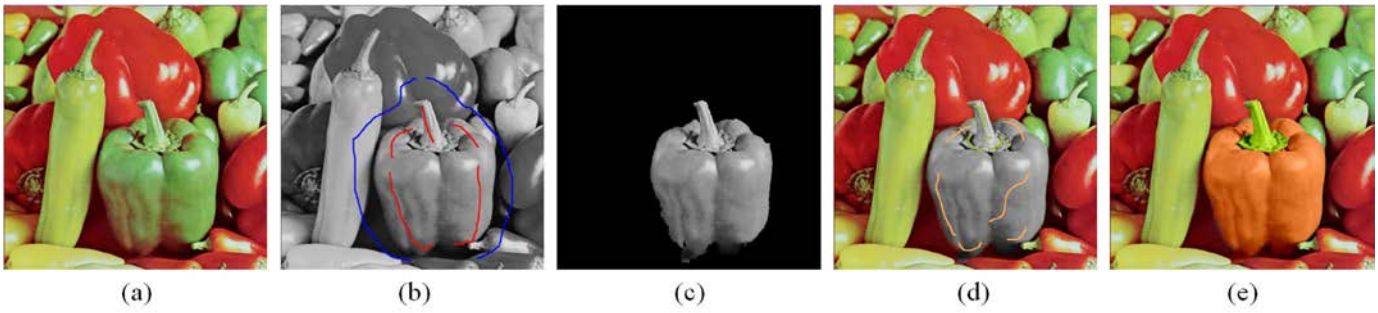

Fig. 21. Recolorization of the *Pepper* image. (a) Original image. (b) Scribbled image showing the area to be recolorized with red scribbles. (c) Mask for recolorization. (d) Remarked image. (e) Recolorized image.

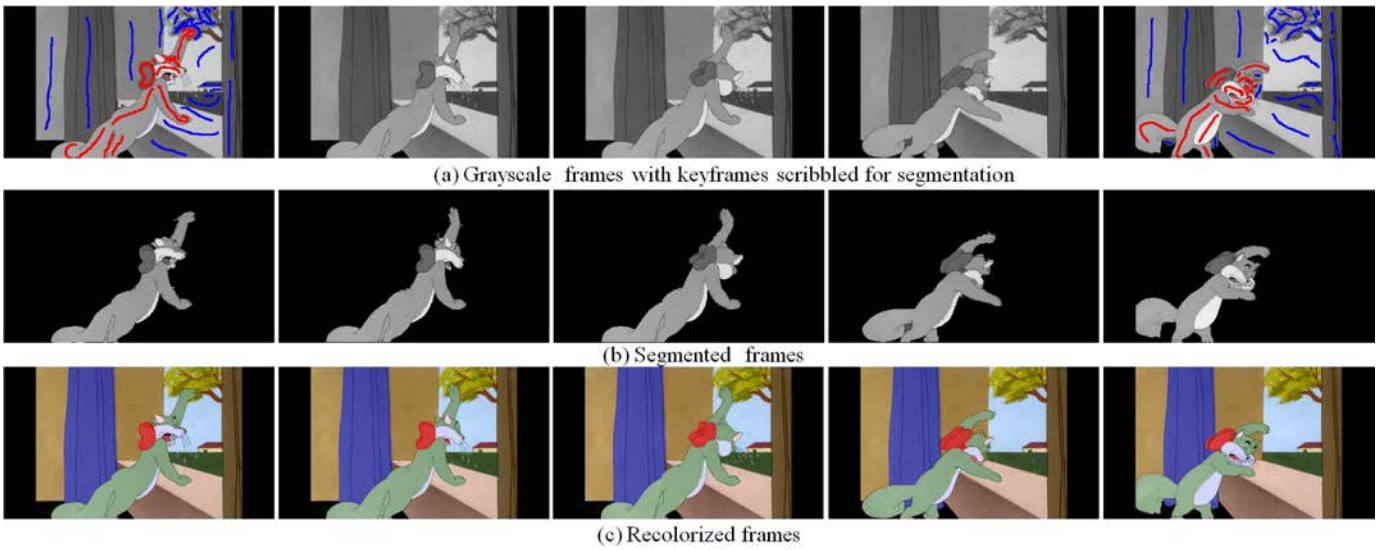

Fig. 22. Recolorization of the colorized *Cartoon* sequence from Fig. 15. (a) Grayscale frames with keyframes scribbled for segmentation. (b) Segmented frames. (c) Recolorized frames.

the red and blue primaries. For image recolorization, the object of interest is marked with red scribbles, and the other parts of the image are marked with blue scribbles. For video recolorization, the grayscale versions of the keyframes of the video sequence are scribbled in the same manner. Color is propagated from the scribbles to all other parts of the image or video sequence using the colorization framework developed in Section III. After color is completely propagated, the area of interest will be red, whereas the remaining part of the image or video sequence will be blue. Now, by extracting the red channel and marking the red pixels with value 1 and all other pixels with value 0, we easily obtain the binary image or video masks for recolorization.

The mask image or video can be effectively used to segment out the object of interest from the image or video sequence under consideration. This essentially boils down to a technique yielding similar results to image and video matting techniques such as [29]–[31]. However, in contrary to the matting techniques that mostly employ color statistics in some form to estimate the foreground, the proposed approach is able to achieve a similar performance using just a grayscale version of the image or video as illustrated by Fig. 20, which compares our segmentation performance with that of the method of [29]. As in the other matting techniques, the matte extracted using our approach is also useful for other diverse applications such as object tracking over a group of frames, foreground extraction, background modifications, and removal of unwanted objects, in addition to recolorization. To recolorize the area of interest, new colors are marked by the user on the segmented object of interest in the image as in Fig. 21(d), and image or video colorization is performed in the usual manner. At the end, the segmented and recolored object of interest is inserted back into the original colored image or video using the binary mask, thus completing our recolorization process. The recolorization result for an image and a video sequence are shown in Figs. 21(e) and 22(c), respectively.[1]

VI. Conclusion

In this paper, we have developed a video colorization algorithm that is capable of colorizing the entire video volume together, without requiring the use of motion vectors between frames. The absence of motion vector computation as well as the spatiotemporal color propagation in 3D space makes our algorithm faster compared with the techniques that employ motion vectors for temporal color transfer. Further, the

TABLE IV

SEPARABLE BASIS FUNCTIONS OF THE SECOND DERIVATIVE OF GAUSSIAN G_2 AND THE CORRESPONDING INTERPOLATION FUNCTIONS

Basis Function	Interpolation Function ($k(\alpha,\beta,\gamma)$)
$G_{2a} = C(2x^2-1)e^{-(x^2+y^2+t^2)}$	α^2
$G_{2b} = C(2xy)e^{-(x^2+y^2+t^2)}$	$2\alpha\beta$
$G_{2c} = C(2y^2-1)e^{-(x^2+y^2+t^2)}$	β^2
$G_{2d} = C(2xt)e^{-(x^2+y^2+t^2)}$	$2\alpha\gamma$
$G_{2e} = C(2yt)e^{-(x^2+y^2+t^2)}$	$2\beta\gamma$
$G_{2f} = C(2t^2-1)e^{-(x^2+y^2+t^2)}$	γ^2

Here, $C = \frac{2}{\sqrt{3}}\left(\frac{2}{\pi}\right)^{3/4}$ is a normalization constant.

TABLE V

THIRTEEN TAP FILTERS FOR THE $x-y-t$ SEPARABLE BASIS SET OF G_2

	1D Function	Tap# 0	1	2	3	4	5	6
g_1	$C(2r^2-1)e^{-r^2}$	-0.8230	-0.3205	0.3028	0.3036	0.1055	0.0183	0.0017
g_2	e^{-r^2}	1.0000	0.7788	0.3679	0.1054	0.0184	0.0019	0.0001
g_3	$2Cre^{-r^2}$	0	0.6409	0.6055	0.2602	0.0603	0.0079	0.0006
g_4	re^{-r^2}	0	0.3894	0.3679	0.1581	0.0366	0.0048	0.0004

generalized nature of our algorithm makes it possible to colorize images using the same framework. As the experimental results demonstrate, the proposed algorithm can effectively handle the colorization of videos having different kinds of motion and is also robust to the presence of occlusion in a video sequence. Also, the comparison of our colorization results with those of [8], [11], and [13] reveals that our method can overcome the color blurring and leakage problems encountered in those methods, in addition to providing greater temporal coherence. Further, the comparison reveals that the proposed technique is also robust to variation in the number and placement of scribbles as it suffices to indicate the color of a uniform grayscale region with a single scribble.

As part of our future work, we plan to develop a method to adaptively choose the orientations of the 3D steerable pyramid based on the video content. This is expected to improve the performance of our algorithm as a predefined number of orientations might not be suitable for all videos, and using a high value of the number of orientations results in unnecessary computation for many videos with simpler visual content. Also, rather than randomly assigning scribbles to the keyframes according to the user's discretion, we seek to algorithmically decide the number of scribbles as well their placement depending on the visual content of the video. Besides optimizing colorization performance, this would also reduce the manual effort of the user in assigning scribbles.

APPENDIX A
SEPARABLE BASIS FUNCTIONS OF STEERABLE FILTERS

The mathematical formulations used in Tables IV–IX are adapted from [28]. Tables V and VIII are derived from Tables IV and VII, respectively, using a sample spacing of 0.5.

The 3D basis filters for G_2 and H_2 can be constructed using the 1D filters with the help of Tables IV–IX. The basis filters are subsequently used to steer G_2 and H_2 to any direction specified by direction cosines $\Theta = (\alpha, \beta, \gamma)$ according to (3).

TABLE VI

CONSTRUCTION OF THE BASIS FILTERS FOR G_2, SHOWING THE 1D FILTER TO BE EMPLOYED ALONG x, y, AND t DIMENSIONS

G_2 Basis Filter	Filter in x	Filter in y	Filter in t
G_{2a}	g_1	g_2	g_2
G_{2b}	g_3	g_4	g_2
G_{2c}	g_2	g_1	g_2
G_{2d}	g_3	g_2	g_4
G_{2e}	g_2	g_3	g_4
G_{2f}	g_2	g_2	g_1

TABLE VII

SEPARABLE BASIS FUNCTIONS FOR THE POLYNOMIAL FIT TO THE HILBERT TRANSFORM OF THE SECOND DERIVATIVE OF GAUSSIAN H_2 AND THE CORRESPONDING INTERPOLATION FUNCTIONS

Basis Function	Interpolation Function
$H_{2a} = C(x^3-2.254x)e^{-(x^2+y^2+t^2)}$	α^3
$H_{2b} = Cy(x^2-0.751333)e^{-(x^2+y^2+t^2)}$	$3\alpha^2\beta$
$H_{2c} = Cx(y^2-0.751333)e^{-(x^2+y^2+t^2)}$	$3\alpha\beta^2$
$H_{2d} = C(y^3-2.254y)e^{-(x^2+y^2+t^2)}$	β^3
$H_{2e} = Ct(x^2-0.751333)e^{-(x^2+y^2+t^2)}$	$3\alpha^2\gamma$
$H_{2f} = C(xyt)e^{-(x^2+y^2+t^2)}$	$6\alpha\beta\gamma$
$H_{2g} = Ct(y^2-0.751333)e^{-(x^2+y^2+t^2)}$	$3\beta^2\gamma$
$H_{2h} = Cx(t^2-0.751333)e^{-(x^2+y^2+t^2)}$	$3\alpha\gamma^2$
$H_{2i} = Cy(t^2-0.751333)e^{-(x^2+y^2+t^2)}$	$3\beta\gamma^2$
$H_{2j} = C(t^3-2.254t)e^{-(x^2+y^2+t^2)}$	γ^3

Here, $C = 0.877776$ is a normalization constant.

TABLE VIII

THIRTEEN TAP FILTERS FOR THE $x-y-t$ SEPARABLE BASIS SET OF H_2

	1D Function	Tap# 0	1	2	3	4	5	6
h_1	$C(r^3-2.254r)e^{-r^2}$	0	-0.6850	-0.4049	-0.0006	0.0561	0.0169	0.0022
h_2	$C(r^2-0.751333)e^{-r^2}$	-0.6595	-0.3427	0.0803	0.1386	0.0522	0.0093	0.0009
h_3	e^{-r^2}	1	0.7788	0.3679	0.1054	0.0183	0.0019	0.0001
h_4	Cre^{-r^2}	0	0.3418	0.3229	0.1388	0.0321	0.0042	0.0003
h_5	re^{-r^2}	0	0.3894	0.3679	0.1581	0.0366	0.0048	0.0004

TABLE IX

CONSTRUCTION OF THE BASIS FILTERS FOR H_2, SHOWING THE 1D FILTER TO BE EMPLOYED ALONG x, y, AND t DIMENSIONS

G_2 Basis Filter	Filter in x	Filter in y	Filter in t
H_{2a}	h_1	h_3	h_3
H_{2b}	h_2	h_5	h_3
H_{2c}	h_5	h_2	h_3
H_{2d}	h_3	h_1	h_3
H_{2e}	h_2	h_3	h_5
H_{2f}	h_4	h_5	h_5
H_{2g}	h_3	h_2	h_5
H_{2h}	h_5	h_3	h_2
H_{2i}	h_3	h_5	h_2
H_{2j}	h_3	h_3	h_1

APPENDIX B
KEYFRAME SELECTION

Keyframes are selected by adaptively thresholding a dissimilarity measure derived from the change in orientations of a steerable filter response over a group of frames. The dissimilarity measure of the tth frame with respect to its

Algorithm 1 Keyframe Selection in a Video Shot

Input: Video shot
Output: Keyframe numbers
Initialisation: $K_b = 1, K_f = T$, Sliding window W

$i_b = K_b + 1$, $i_f = K_f - 1$
Compute $B_b(k)$ where $k \in \{K_b, \cdots, K_b + (W-1)\}$
Compute $B_f(k)$ where $k \in \{K_f, \cdots, K_f - (W-1)\}$
τ_b : computed over K_b to $K_b + (W-1)$
τ_f : computed over K_f to $K_f - (W-1)$
while $i_f > i_b$ **do**
  if $(B_b(i_b) > \tau_b)$ **then**
    $K_b = i_b$
    print K_b
    update $B_b(k)$ for $k \in \{K_b, \cdots, K_b + (W-1)\}$
    recompute τ_b
  end if
  if $(B_f(i_f) > \tau_f)$ **then**
    $K_f = i_f$
    print K_f
    update $B_f(k)$ for $k \in \{K_f, \cdots, K_f + (W-1)\}$
    recompute τ_f
  end if
  $i_b = i_b + 1$
  $i_f = i_f - 1$
end while

immediate backward keyframe K_b and immediate forward keyframe K_f is given by

$$B_b(t) = \frac{1}{MN} \sum_{\mathbf{p}} |\Delta_{K_b}(\mathbf{p}) - \Delta_t(\mathbf{p})|/360$$

$$B_f(t) = \frac{1}{MN} \sum_{\mathbf{p}} |\Delta_{K_f}(\mathbf{p}) - \Delta_t(\mathbf{p})|/360$$

where $\Delta_t(\mathbf{p})$ is the dominant orientation response at pixel $\mathbf{p}$ obtained on steerable filtering of the tth frame using a temporal slice of the 3D steerable filters. The keyframe selection is then performed as per Algorithm 1. Here, τ_b and τ_f are the adaptive thresholds given by

$$\tau_b = \frac{\max\{B_b(t), \ldots, B_b(t+(W-1))\}}{c \times \frac{1}{W} \sum_{i=0}^{W-1} B_b(t+i)}$$

$$\tau_f = \frac{\max\{B_f(t), \ldots, B_f(t-(W-1))\}}{c \times \frac{1}{W} \sum_{i=0}^{W-1} B_f(t-i)}.$$

For the purpose of keyframe selection, the filter is steered to 90 different orientations by keeping the value of ϕ fixed at 0°. The constant c is set to 10 and the size of the sliding window W is taken to be 10 frames.

REFERENCES

[1] R. C. Gonzalez and R. E. Woods, *Digital Image Processing*, 2nd ed., Boston, MA, USA: Addison-Wesley, 2009.

[2] T. Welsh, M. Ashikhmin, and K. Mueller, "Transferring color to greyscale images," *ACM Trans. Graph.*, vol. 21, no. 3, pp. 277–280, Jul. 2002.

[3] R. Irony, D. Cohen-Or, and D. Lischinski, "Colorization by example," in *Proc. 16th Eurograph. Conf. Rendering Techn.*, 2005, pp. 201–210.

[4] A. Bugeau, V.-T. Ta, and N. Papadakis, "Variational exemplar-based image colorization," *IEEE Trans. Image Process.*, vol. 23, no. 1, pp. 298–307, Jan. 2014.

[5] D. Sýkora, J. Buriánek, and J. Žára, "Unsupervised colorization of black-and-white cartoons," in *Proc. 3rd Int. Symp. Non-Photorealistic Animation Rendering*, 2004, pp. 121–127.

[6] Z. Zhen, G. Yan, and M. Lizhuang, "An automatic image and video colorization algorithm based on pattern continuity," in *Proc. Int. Conf. Audio, Lang. Image Process.*, Jul. 2012, pp. 531–536.

[7] T. Horiuchi and S. Hirano, "Colorization algorithm for grayscale image by propagating seed pixels," in *Proc. IEEE Int. Conf. Image Process.*, vol. 1. Sep. 2003, pp. I-457–I-460.

[8] A. Levin, D. Lischinski, and Y. Weiss, "Colorization using optimization," *ACM Trans. Graph.*, vol. 23, no. 3, pp. 689–694, Aug. 2004.

[9] K. Uruma, K. Konishi, T. Takahashi, and T. Furukawa, "Image colorization based on the mixed L_0/L_1 norm minimization," in *Proc. 19th IEEE Int. Conf. Image Process.*, Sep./Oct. 2012, pp. 2113–2116.

[10] J. Pang, O. C. Au, Y. Yamashita, Y. Ling, Y. Guo, and J. Zeng, "Self-similarity-based image colorization," in *Proc. IEEE Int. Conf. Image Process.*, Oct. 2014, pp. 4687–4691.

[11] L. Yatziv and G. Sapiro, "Fast image and video colorization using chrominance blending," *IEEE Trans. Image Process.*, vol. 15, no. 5, pp. 1120–1129, May 2006.

[12] J.-H. Heu, D.-Y. Hyun, C.-S. Kim, and S.-U. Lee, "Image and video colorization based on prioritized source propagation," in *Proc. IEEE Int. Conf. Image Process.*, Nov. 2009, pp. 465–468.

[13] D.-Y. Hyun, J.-H. Heu, C.-S. Kim, and S.-U. Lee, "Prioritized image and video colorization based on Gaussian pyramid of gradient images," *J. Electron. Imag.*, vol. 21, no. 2, p. 023027, Jun. 2012.

[14] V. G. Jacob and S. Gupta, "Colorization of grayscale images and videos using a semiautomatic approach," in *Proc. 16th IEEE Int. Conf. Image Process.*, Nov. 2009, pp. 1653–1656.

[15] S. Teng, Y. Shen, Z. Zhao, L. Li, and M. Cao, "An interactive framework for video colorization," in *Proc. 7th Int. Conf. Image Graph.*, Jul. 2013, pp. 89–94.

[16] J. Revaud, P. Weinzaepfel, Z. Harchaoui, and C. Schmid, "EpicFlow: Edge-preserving interpolation of correspondences for optical flow," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, Jun. 2015, pp. 1164–1172.

[17] R. Kennedy and C. J. Taylor, "Hierarchically-constrained optical flow," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, Jun. 2015, pp. 3340–3348.

[18] B. Sheng, H. Sun, M. Magnor, and P. Li, "Video colorization using parallel optimization in feature space," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 24, no. 3, pp. 407–417, Mar. 2013.

[19] B. Sheng, H. Sun, S. Chen, X. Liu, and E. Wu, "Colorization using the rotation-invariant feature space," *IEEE Comput. Graph. Appl.*, vol. 31, no. 2, pp. 24–35, Mar./Apr. 2011.

[20] E. H. Adelson and J. R. Bergen, "Spatiotemporal energy models for the perception of motion," *J. Opt. Soc. Amer. A*, vol. 2, no. 2, pp. 284–299, 1985.

[21] K. J. Cannons and R. P. Wildes, "The applicability of spatiotemporal oriented energy features to region tracking," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 36, no. 4, pp. 784–796, Apr. 2014.

[22] K. Rapantzikos, Y. Avrithis, and S. Kollias, "Dense saliency-based spatiotemporal feature points for action recognition," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, Jun. 2009, pp. 1454–1461.

[23] X. Zhen and L. Shao, "Spatio-temporal steerable pyramid for human action recognition," in *Proc. 10th IEEE Int. Conf. Workshops Autom. Face Gesture Recognit.*, Apr. 2013, pp. 1–6.

[24] J. A. Montoya-Zegarra, N. J. Leite, and R. da S. Torres, "Rotation-invariant and scale-invariant steerable pyramid decomposition for texture image retrieval," in *Proc. 20th Brazilian Symp. Comput. Graph. Image Process.*, Oct. 2007, pp. 121–128.

[25] K. Plataniotis and A. N. Venetsanopoulos, "Color spaces," in *Color Image Processing and Applications*. Berlin, Germany: Springer, 2000, pp. 16–32.

[26] F. Pierre, J. Aujol, A. Bugeau, and V. Ta, "Luminance-hue specification in the RGB space," in *Proc. Scale Space Variat. Methods Comput. Vis.*, 2015, pp. 413–424.

[27] W. T. Freeman and E. H. Adelson, "The design and use of steerable filters," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 13, no. 9, pp. 891–906, Sep. 1991.

[28] K. G. Derpanis and J. M. Gryn, "Three-dimensional nth derivative of Gaussian separable steerable filters," in *Proc. IEEE Int. Conf. Image Process.*, vol. 3. Sep. 2005, pp. III-553–III-556.

[29] X. Bai and G. Sapiro, "A geodesic framework for fast interactive image and video segmentation and matting," in *Proc. IEEE 11th Int. Conf. Comput. Vis.*, Oct. 2007, pp. 1–8.
[30] J. Fan, X. Shen, and Y. Wu, "Scribble tracker: A matting-based approach for robust tracking," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 34, no. 8, pp. 1633–1644, Aug. 2012.
[31] M. Gong, Y. Qian, and L. Cheng, "Integrated foreground segmentation and boundary matting for live videos," *IEEE Trans. Image Process.*, vol. 24, no. 4, pp. 1356–1370, Apr. 2015.

Somdyuti Paul received the B.Tech. degree in electronics and communication engineering from West Bengal University of Technology, Kolkata, India, in 2012 and the M.Tech. degree in electrical engineering from IIT Kanpur, Kanpur, India, in 2015.

Her research interests include image processing, computer vision, and machine learning.

Saumik Bhattacharya received the B.Tech. degree in electronics and communication engineering from West Bengal University of Technology, Kolkata, India, in 2011. He is currently working toward the Ph.D. degree with IIT Kanpur, Kanpur, India.

His research interests include computational photography, computer vision, and image processing.

Sumana Gupta (M'91) received the B.E. and M.E. degrees in electrical communication engineering from Indian Institute of Science, Bangalore, India, and the Ph.D. degree in electrical engineering from the Imperial College of Science and Technology, London, U.K.

She is a Professor with the Department of Electrical Engineering, IIT Kanpur, Kanpur, India. Her research interests include video compression, video restoration, and image analysis.