Video Colorization Based on Optical Flow and Edge-Oriented Color Propagation

Mayu Otani* and Hirohisa Hioki**

*Nara Institute of Schience and Technology, 8916-5 Takayama, Ikoma, Nara 630-0192, Japan;
**Kyoto University, Yoshidanihonmatsu cho, Sakyo ku, Kyoto 606-8501, Japan

ABSTRACT

We propose a novel video colorization method based on sparse optical flow and edge-oriented color propagation. Colorization is a process of adding color to monochrome images or videos. In our video colorization method, it is assumed that key frames are appropriately selected out of a grayscale video stream and are properly colorized in advance. Once key frames are colorized, our method colorizes all the remaining grayscale frames automatically. It is also possible to colorize key frames semi-automatically by our method. For colorizing a grayscale frame between a pair of colorized key frames, sparse optical flow is computed first. The optical flow consists of reliable motion vectors around strong feature points. The colors around feature points of key frames are then copied to the grayscale frame according to the estimated motion vectors. Those colors are then propagated to the rest of the grayscale frame. Colors are blended appropriately during the propagation process. A pair of accuracy and priority measures is introduced to control how the color propagation proceeds. To successfully propagate colors, it is important not to wrongly spread colors across edges. For this purpose, a set of neighboring pixels is adaptively selected not to include edge-like areas and thus not to spread colors across edges. To evaluate effectiveness of our method, image colorization and video colorization were performed. Experimental results show that our method can colorize images and videos better than previous methods when there are edges. We also show that the proposed method enables us to easily modify colors in colored video streams.

Keywords: colorization, video colorization, color propagation

1. INTRODUCTION

Colorization is a process of adding color to monochrome images or videos. Typical applications of colorization includes restoring colors of old films and providing special color effects for photos. Since a common video stream consists of a huge number of frames, video colorization is a time consuming process. To reduce the cost of the colorization process, an automatic colorization method is required.

Several methods have been proposed for both image and video colorization. One of main approaches is *"example based method."* For example, Irony et al. proposed a colorization method which refers a given color image and automatically adds "micro color strokes" to a grayscale image and propagates the colors.[1] This method is suitable for tasks of colorizing complicated texture.

Another approach is *"user involved method."* In colorization methods of this type, for colorizing a frame, a user is supposed to give color strokes appropriately which are used to estimate colors for the rest of the image. Although user involved methods are *semi-automatic* in the strict sense because they require the user's assist at the initial step, but we simply call them *automatic* method in the sense that they can automate most processes of colorization. In this paper, we employ this approach because it is superior to the others in respect of making

E-mail: *otani.mayu.ob9@is.naist.jp, **hioki@i.h.kyoto-u.ac.jp

Mayu Otani ; Hirohisa Hioki; Video colorization based on optical flow and edge-oriented color propagation . Proc. SPIE 9020, Computational Imaging XII, 902002 (March 7, 2014); doi:10.1117/12.2037496.

Copyright 2014 Society of Photo-Optical Instrumentation Engineers. One print or electronic copy may be made for personal use only. Systematic reproduction and distribution, duplication of any material in this paper for a fee or for commercial purposes, or modification of the content of the paper are prohibited.

http://dx.doi.org/10.1117/12.2037496

color images reflect the user's intention. Levin et al. proposed an interactive colorization system based on the assumption that neighboring pixels of similar intensities have similar colors.[2] Under this assumption, they developed a cost function to be minimized for colorizing a grayscale image when initial color strokes are given. In addition, they cover video colorization by extending the definition of neighborhood based on dense optical flow. The colorized frames obtained by this method, however, can show wrongly spread colors over different regions because edges between regions are not apparently considered.

To solve this problem, J. H. Heu et al. proposed a method where color propagation in an image is controlled by a pair of accuracy and priority measures defined for each pixel.[3] Let us call their method as JHH method in the rest of this paper. The accuracy measure of a pixel represents the reliability of the color provided for it. The priority measure specifies an order among pixels to be colorized. The priority value of a pixel is defined by a function of the pixel and its four neighbors. In the color propagation process of JHH method, the grayscale pixels are colorized from the one with the highest priority value. The color of a pixel is derived by blending the colors of its four neighboring pixels with considering their accuracy values. The accuracy and priority values of pixels are updated as the color propagation process proceeds. For video colorization, JHH method copies colors based on dense optical flow. The accuracy values of pixels are computed by estimating the reliability of the derived motion vectors. The colors of pixels are then updated through the color propagation process.

Even with this method, however, we still suffer from the problem of inappropriate color propagation, since the method always propagates colors using four neighbors of each pixel irrespective of edges. At the same time, wrong estimation of motion vectors can also damage the result. A more precise color transferring and propagating method is hence required to achieve video colorization with a high quality.

In this paper, we propose a new colorization method based on sparse optical flow and edge-oriented color propagation. In our method, it is assumed that key frames are selected out appropriately from a grayscale video stream and are properly colorized in advance. Once key frames are colorized, our method colorizes all the remaining grayscale frames automatically. More specifically, we use a pair of colorized key frames for colorizing an in-between grayscale frame. We can also colorize key frames semi-automatically with our color propagation method when initial color strokes are provided appropriately.

Our method consists of two stages. In the first stage, a grayscale frame between a pair of key frames is partially colorized by transferring colors around feature points in the key frames. By computing sparse but reliable optical flow for strong feature points, we can transfer only reliable colors to the grayscale frame and thus reduce the influence of wrong motion estimation. Colors transferred by a motion vector to a target frame are accompanied with an accuracy value which represents how reliable the motion vector is. Accuracy values are used for estimation of the colors of remaining grayscale pixels. As a result of the first stage process, the grayscale frames are thus partially colorized. We propagate the copied colors around feature points to remaining grayscale pixels in the second stage. For propagating colors, we use priority and accuracy measures like the JHH method. As is already stated, in the JHH method, four neighboring pixels are always used for computing priority values and propagating colors, which causes colors are wrongly propagated across edges. In our edge-oriented color propagation method, instead of four pixels neighbors, we introduce nine different sets of neighboring pixels and select one of them adaptively with considering edges for computing priority values and propagating colors. This propagation method enables us to avoid spreading inappropriate colors across edges. The accuracy and priority values are updated during the color propagation process.

We have carried out experiments of still image colorization and video colorization. The still image experimental results show our propagation algorithm works well when there are edges. Video experimental results demonstrate that our method can colorize videos with good quality and thus our method is useful. We also show that our method enables us to easily modify colors in colored video streams.

2. COLORIZATION ALGORITHM

Our method mainly aims at colorizing grayscale video streams where key frames are appropriately selected and are separately colorized in advance. As we mentioned above, our method consists of two stages: color transfer from colorized key frames and color propagation in each grayscale frame. In the first stage, we transfer colors from a pair of key frames to an in-between grayscale frame. We copy colors around strong feature points to a grayscale frame using sparse optical flow. In the second stage, we propagate the copied colors to grayscale pixels remaining in the target frame based on accuracy and priority measures. In order to avoid spreading colors across edges during color propagation, we introduce nine different sets of neighboring pixels and select one of them adaptively.

For representing color values, we employ YUV color space. In this paper, the intensity component Y and color components U and V of a pixel p in a frame k are denoted as:

$$Y_k(p) \tag{1}$$

$$C_k(p) = (U_k(p), V_k(p)). \tag{2}$$

In this section, we explain the first stage of our method in Section 2.1 including the way of transferring colors by sparse optical flow and the definition of the accuracy value. We then describe the details of the second stage of the edge-oriented color propagation method in Section 2.2. We first introduce the nine sets of neighboring pixels and describe how to use them in our method. Next, the definitions of the priority value and a blending function for color propagation are given. We also explain how to update accuracy and priority values and how we complete the color propagation process.

2.1 Color Transfer to Grayscale Frame

In the first stage, we copy colors from a pair of key frames to a target grayscale frame according to sparse optical flow. Let k_1 and k_2 be colorized key frames and g be a grayscale frame between them. Optical flow is computed once between the first (front) key frame k_1 and the target frame g. Another optical flow is computed between the second (back) key frame k_2 and the same target frame g. Both optical flows are used for copying colors. Copying color from both front and back key frames helps us to better colorize a grayscale frame when new objects are suddenly appeared. If we copy colors only from the first key frame k_1, we cannot provide color components with objects that are not appeared in the frame k_1. In contrast, if we copy colors from front and back key frames, it would increase chance to provide appropriate colors for such objects. To copy color components to correct regions, we have to derive reliable optical flow. For this purpose, we employ pyramidal implementation of the Lucas-Kanade feature tracker.[4]

For computing optical flow, first, we extract corner-like feature points from a pair of key frames. Motion vectors of the corner-like points are then easily estimated using the Lucas-Kanade feature tracker. We can thus derive optical flow which is sparse but consists of reliable motion vectors, which prevents us from copying colors wrongly. According to the obtained optical flow, we copy colors around feature points to the grayscale frame g. Let p_i be a feature point in key frame $k_j (j = 1, 2)$. Colors of pixels around p_i are copied to the frame g as follows:

$$C_g(s + v_{p_i}) \leftarrow C_{k_j}(s), s \in \mathcal{B}_{p_i}, p_i \in k_j \tag{3}$$

where v_{p_i} is the motion vector of p_i and $\mathcal{B}_{p_i}$ is a block centered at p_i. The size of $\mathcal{B}_{p_i}$ is given by a parameter. We now define the accuracy values for the pixel at $s + v_{p_i}$ of the frame g as

$$a(s + v_{p_i}) = \begin{cases} exp(-|Y_{k_j}(s) - Y_g(s + v_{p_i})|) & SAD(p_i + v_{p_i}, p_i) \geq T, \\ 1 & otherwise, \end{cases} \tag{4}$$

$$SAD(p_i + v_{p_i}, p_i) = \sum_{q \in \mathcal{B}_{p_i}} |Y_g(q + v_{p_i}) - Y_{k_j}(q)| \tag{5}$$

where T is a threshold value. Note that $0 < a(p) \leq 1$ for each pixel p. Accuracy is high when the color of a pixel is copied with a reliable motion vector.

According to the above definitions, it can happen that more than one color components of pixels in key frames are to be copied to a certain pixel in the target frame. Suppose that we transfer color components from color pixels $s_i (s_i \in \mathcal{B}_{p_i}, i = 1, \ldots, n)$ to the same target pixel t. In this case, we define the color and accuracy value of the pixel t by taking a weighted sum as:

$$a(t) \leftarrow \frac{1}{n}\sum_{i=1}^{n} a(s_i + v_{p_i}) \tag{6}$$

$$C_g(t) \leftarrow \frac{\sum_{i=1}^{n} C(s_i)a(s_i + v_{p_i})}{na(t)}. \tag{7}$$

Note that we keep the intensity value $Y_g(p)$ intact for each pixel p in the target frame g. When the first stage of our method is finished, we obtain a partially colorized frame as shown in Figure 1.

(a) Front key frame (b) Partially colorized frame (c) Back key frame

Figure 1. An example of color transfer from a pair of key frames

2.2 Color Propagation

In the second stage, we propagate the colors to the rest of the frame. We assume that a pixel has a similar color with its neighboring pixels unless there are edges. When propagating colors to a grayscale pixel from neighboring pixels, it is important not to spread colors across edges. For this purpose, we adaptively select a set of neighboring pixels without edges. This adaptive selection of neighborhood is inspired by an edge preserving smoothing algorithm.[5]

Since the pixels often reveal significantly different intensity values across an edge, it is likely that a set of neighboring pixels includes an edge when the variance of intensity values is large. With this consideration, we compute the intensity variance values for different sets of neighboring pixels and select the set that has the minimum intensity variance for propagating colors. Figure 2 shows the nine sets of neighboring pixels used in our method. The priority value of a pixel is defined based on the intensity differences and the accuracy values

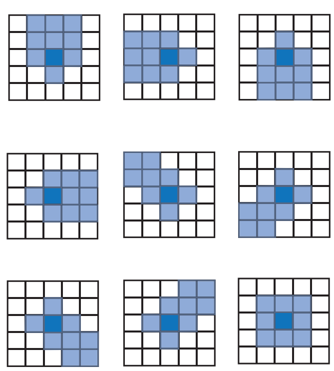


Figure 2. Nine sets of neighboring pixels for the pixel at the center

of pixels in the selected set of neighboring pixels $\mathcal{N}_p$ as:

$$\pi(p) = \sum_{q \in \mathcal{N}_p} a(q) exp(-|Y(q) - Y(p)|). \tag{8}$$

The priority value of a pixel p becomes high when $\mathcal{N}_p$ consists of pixels that have similar intensity values and high accuracy values.

Because it is expected that we can more confidently propagate the color to a pixel with a higher priority value than to ones with lower priority values, in the color propagation process, we search for a pixel with the highest priority value and propagate color to it first. The color of grayscale pixel is derived by blending the colors of pixels in the N_p according to the following equation:

$$C(p) \leftarrow \sum_{q \in \mathcal{N}_p} C(q) \frac{a(q) exp(-|Y(q) - Y(p)|)}{\pi(p)}. \tag{9}$$

After a pixel p is colorized, we update its accuracy as $a(p) = 1$ and recalculate the priority values of the pixels in the area where the accuracy of the pixel p can affect. We then search for a pixel with the highest priority value again and propagate color to it from its neighboring pixels. This color propagation process, i.e., searching a pixel with the highest priority, propagating colors and updating accuracy and priority values, is repeated until the accuracy values of all the pixels become 1. When the color propagation process is finished, we obtain a fully colorized frame.

3. EXPERIMENTS

In this section, colorization results by our method are compared with those by previous colorization methods. We performed still image colorization and video colorization. The peak signal to noise ratio (PSNR) was employed to evaluate the resulting images.

3.1 Still Image Colorization

We evaluated the performance of our propagation method through still image colorization. We compared our method with the JHH method[3] and Levin et al.'s method.[2] The color image shown in Figure 3(a) was converted into grayscale and color strokes were added as shown in Figure 3(b). This image was provided for Levin et al.'s method, JHH method, and ours. The results of colorization are shown in Figure 3(c), (d), and (e) respectively.

(a) original image (b) input image (c) Levin et al.'s method (32.66dB) (d) JHH method (31.66dB) (e) Our method (32.76dB)

Figure 3. Still image colorization results (1)

In Figure 3 and Figure 4, we can find that the sleeve region and fingers are wrongly colorized by Levin et al.'s method and JHH method, while our method successfully restores correct colors. Figure 5 shows another example of still image colorization. The given color strokes are so limited as shown in Figure 5(b). From Figure 5(c), it

(a) Levin et al.'s method

(b) JHH method

(c) Our method

Figure 4. Close-ups of results shown in Figure 3

(a) original image

(b) input image

(c) Levin et al.'s method (26.33dB)

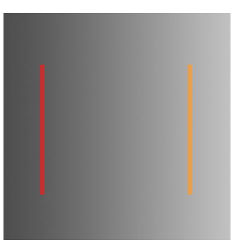
(d) JHH method (32.80dB)

(e) Our method (33.39dB)

Figure 5. Still image colorization results (2)

seems that more initial strokes are required for Levin et al.'s method to achieve an acceptable quality. As we see in Figure 5(d), the image generated by JHH method is well colorized roughly, but we can still see inappropriate colors spreading in the right bottom region. Our method gives a satisfactory image with this small amount of strokes as shown in Figure 5(e).

As these results show, our method outperformed the previous methods. It is, however, not always the case. Because our method progressively propagates colors based on neighborhood, it is not suitable for restoring gradual changes of colors as shown in Figure 6. This result indicates that colorization algorithm using optimization is more eligible for this kind of images.



(d)Input image

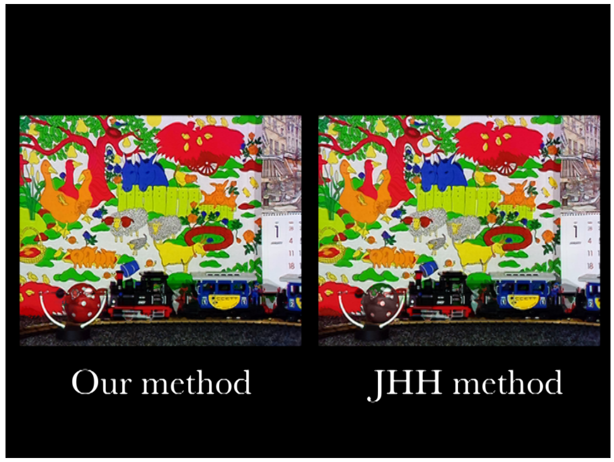
(b)Levin et al.'s method

(c)Our method

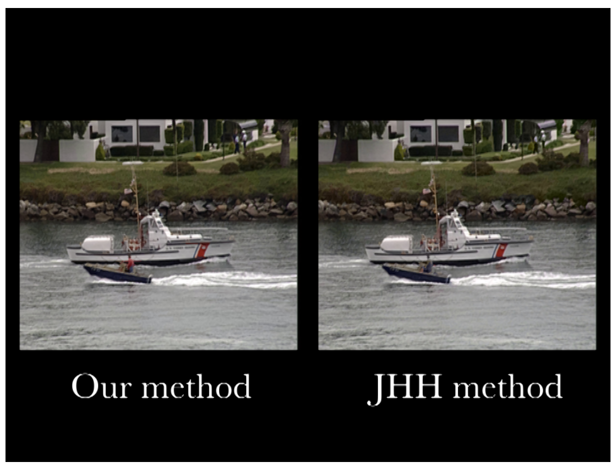
Figure 6. Colorized gradation images by Levin et al.'s method[2] and ours

3.2 Video Colorization

We performed video colorization by our method and JHH method. In this experiment, in order to evaluate how video colorization works among frames, test video streams were prepared by converting color video streams into grayscale frames with color key frames remaining at every five frames. We colorized test video streams and computed PSNR values. The mean of PSNR values of the colorized frames by our method slightly exceeded



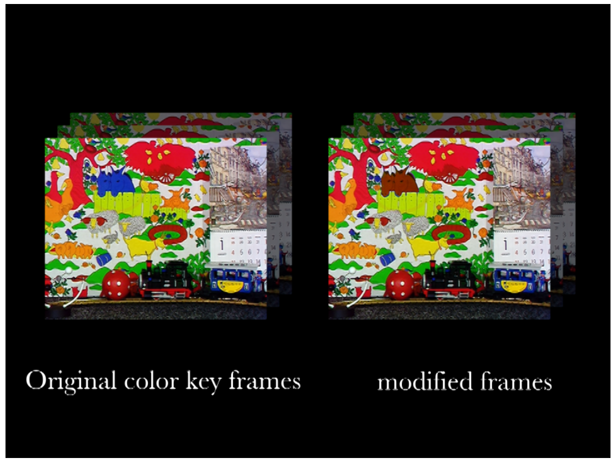
Video 1. A video colorization result (1)
http://dx.doi.org/10.1117/12.2037496.1



Video 2. A video colorization result (2)
http://dx.doi.org/10.1117/12.2037496.2

JHH method's one for the video stream "mobile" (Video 1): the value of our method was 32.51dB and that of JHH method's was 32.04dB. For another video stream "coast guard" (Video 2), however, the value of ours was 45.32dB and that of JHH method's was 46.59dB. Although our method did not show significant advantage in PSNR, we found that our method more successfully colorized objects those were occluded by other objects or cut into frames from outside. An example is shown in Video 1. The ball in front of the locomotive was successfully colorized by our method, while not by JHH method.

We also modified colors of a region in the "mobile" video stream. By just modifying colors in key frames, we could colorize all the frames with modified colors. The color of horses in Video 3 was modified from blue to brown.



Video 3. A result of video color modification
http://dx.doi.org/10.1117/12.2037496.3

4. SUMMARY

We have presented a novel video colorization method based on sparse optical flow and edge-oriented color propagation in this paper. Given a pair of color key frames, our method automatically colorizes the in-between grayscale frames. Colors around feature points of key frames are transferred to a grayscale frame by computing reliable motion vectors as sparse optical flow. Those colors are then propagated to the rest of the grayscale frame. On propagating colors, we select adaptively one of nine sets of neighboring pixels not to spread colors across edges. Experimental results show that our method can better colorize images and videos than previous methods when there are edges. We also show that our method can easily modify colors of color video streams.

Future directions of this study include adaptive selection of key frames to further automate our video colorization process. Colorization of video streams consisting of line art frames is another interesting issue, which would be quite useful for creating animation films.

ACKNOWLEDGEMENTS

The images shown in Figure 3(a) and 5(a) are produced respectively by Einharch, licensed under a Creative Commons BY-NC-SA : `http://www.flickr.com/photos/einharch/5100682571/` and by T.Kiya, licensed under a Creative Commons BY-SA : `http://www.flickr.com/photos/cq-biker/5742946788/`. These images were modified by experiments in our study as shown in this paper. The original video streams shown in Video 1, Video 2 and Video 3 are provided at a web site 'Video Trace Library.[6]'

REFERENCES

[1] R. Irony, D. Cohen-Or, and D. Lischinski, "Colorization by example," in *Proceedings of the Sixteenth Eurographics Conference on Rendering Techniques*, *EGSR'05*, pp. 201–210, Eurographics Association, (Aire-la-Ville, Switzerland, Switzerland), 2005.

[2] A. Levin, D. Lischinski, and Y. Weiss, "Colorization using optimization," *ACM Trans. Graph.* **23**, pp. 689–694, Aug. 2004.

[3] J.-H. Heu, D.-Y. Hyun, C.-S. Kim, and S.-U. Lee, "Image and video colorization based on prioritized source propagation," in *Proceedings of the 16th IEEE International Conference on Image Processing*, *ICIP'09*, pp. 465–468, IEEE Press, (Piscataway, NJ, USA), 2009.

[4] J. yves Bouguet, "Pyramidal implementation of the lucas kanade feature tracker," *Intel Corporation, Microprocessor Research Labs* , 2000.

[5] M. Nagao and T. Matsuyama, "Edge preserving smoothing," *Computer Graphics and Image Processing* **9**(4), pp. 394 – 407, 1979.

[6] P. Seeling and M. Reisslein, "Video transport evaluation with H.264 video traces," *IEEE Communications Surveys and Tutorials, in print* **14**(4), pp. 1142–1165, 2012. Traces available at `trace.eas.asu.edu`.