A Practical Monochrome Video Colorization Framework for Broadcast Program Production

Rei Endo, Yoshihiko Kawai, and Takahiro Mchizuki

***Abstract*—Techniques of using convolutional neural networks (CNNs) to colorize monochrome still images have been widely researched. However, the results of automatic colorization are often different from the user's intentions and historical fact. A lot of color correction work still needs to be done in order to produce a colorized video. This is a major problem in situations such as broadcasting production where footage must be appropriately colorized in accordance with historical fact. In this article, we propose a practical video colorization framework that can easily reflect the user's intentions. The proposed framework uses a combination of two CNNs—a user-guided still-image-colorization CNN and a color-propagation CNN—that allows the correction work to be performed efficiently. The user-guided still-image-colorization CNN produces key frames by colorizing several monochrome frames from the target video on the basis of user-specified colors and color-boundary information. The color-propagation CNN automatically colorizes the entire video on the basis of the key frames, while suppressing discontinuous changes in color between frames. A quantitative evaluation showed that it is possible to produce color video reflecting the user's intention with less effort than with earlier methods.**

***Index Terms*—Colorization, convolutional neural network (CNN), generative adversarial network (GAN).**

I. Introduction

VIDEO from old monochrome film not only has strong artistic appeal in its own right, but also contains many important historical facts and lessons. However, it tends to look very old-fashioned to viewers. To convey the world of the past to viewers in a more engaging way, TV programs often colorize monochrome video [1], [2]. Outside of TV program production, there are many other situations where colorization of monochrome video is required. For example, it can be used as a means of artistic expression, as a way of recreating old memories [3], and for remastering old images for commercial purposes.

In most cases, the colorization of monochrome video has required experts to colorize each individual frame manually. This is a very expensive and time-consuming process. As a result, colorization has only been practical in projects with very large budgets. In recent years, efforts have been made to reduce costs by using computers to automate the colorization process. When using automatic colorization technology for TV programs and movies, an important requirement is that users should have some way of specifying their intentions regarding the colors to be used. A function that allows specific objects to be assigned specific colors is indispensable when the correct color is based on historical fact, or when the color to be used has already been decided upon during the production of a program. Our aim is to devise colorization technology that meets this requirement and produces broadcast-quality results.

There have been many reports on accurate still-image colorization techniques [4], [5], [6], [7], [8], [9]. However, the colorization results obtained by these techniques are often different from the user's intention and historical fact. In some of the earlier technologies, this issue is addressed by introducing a mechanism whereby the user can control the output of the convolutional neural network (CNN) [10] by using user-guided information (colorization hints) [11], [12]. However, for long videos, it is very costly and time-consuming to prepare suitable hints for every frame. The amount of hint information needed to colorize videos can be reduced by using a technique called video propagation [13], [14], [15]. Using this technique, color information assigned to one frame can be propagated to other frames. In the following, a frame to which information has been added in advance is called a "key frame", and a frame to which this information is to be propagated is called a "target frame". However, even using this technique, it is difficult to colorize long videos because if there are differences in the colorings of different key frames, color discontinuities may occur in places where the key frames are switched.

In this article, we propose a practical video colorization framework that can easily reflect the user's intentions. Our aim is to realize a technique that can be used to colorize entire video sequences with appropriate colors chosen on the basis of historical fact and other sources, so they can be used in broadcast programs and other productions. The basic concept is that a CNN is used to automatically colorize the video, and then the user corrects only those video frames that were colored differently from his/her intentions. By using a combination of two CNNs—a user-guided still-image-colorization CNN and a color-propagation CNN—the correction work can be performed efficiently. The user-guided still-image-colorization CNN produces key frames by colorizing several monochrome frames from the target video in accordance with user-specified colors and color-boundary information. The color-propagation CNN automatically colorizes the entire video on the basis of the key frames, while suppressing discontinuous changes

Manuscript received December 19, 2019; revised May 22, 2020; accepted June 7, 2020. Date of publication October 20, 2020; date of current version March 4, 2021. *(Corresponding author: Rei Endo.)*

The authors are with the Science and Technology Research Laboratories, NHK (Japan Broadcasting Corporation), Setagaya 157-8510, Japan (e-mail: endou.r-mm@nhk.or.jp).

Digital Object Identifier 10.1109/TBC.2020.3028343

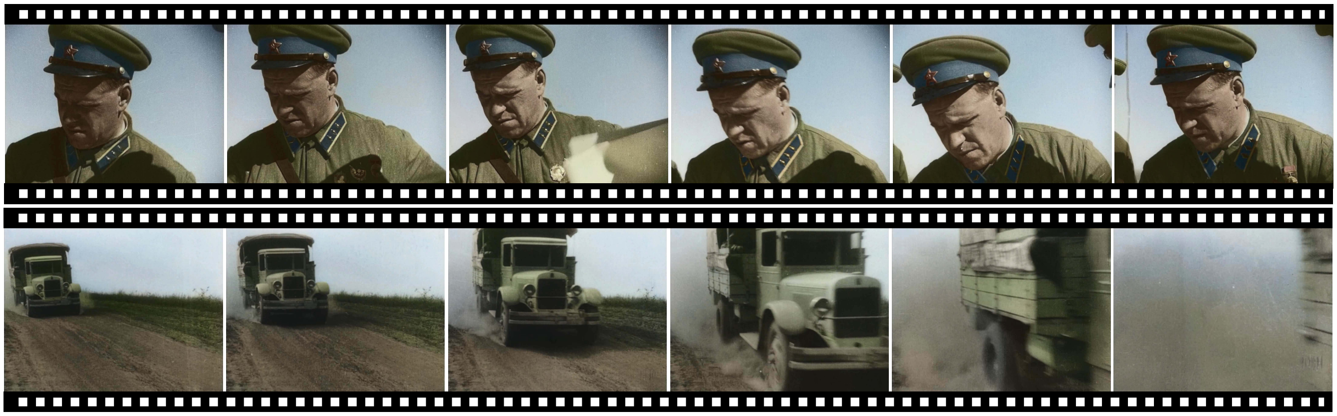

Fig. 1. A colorized film made for an actual TV broadcast production using the proposed framework.

in color between frames. The results of qualitative evaluations show that our method reduces the workload of colorizing videos while appropriately reflecting the user's intentions. In particular, when our framework was used in the production of actual broadcast programs, we found that it could colorize video in a substantially shorter time compared with manual colorization. Figure 1 shows some examples of colorized images produced with the framework for use in broadcast programs.

The main contributions of this article are as follows: (1) identifying the practical issues with current video colorization technology and devising a practical framework to solve these issues; (2) providing the still-image-colorization CNN with the positions of color boundaries so that the user's intentions can be reflected more flexibly in the CNN output; (3) providing a color-propagation CNN with the ability to use multiple key frames so that the user is freed from the laborious task of preparing key frames with strictly controlled color shading; (4) reporting on an application of our framework to the production of actual broadcast programs.

This article is organized as follows. In Section II, we present an overview of the conventional techniques. In Section III, we describe our framework. In Section IV, we describe the two types of CNN used in our framework. In Section V, we discuss the experimental results. Finally, in Section VI, we present our conclusions.

II. Related Work

Still-image-colorization techniques can be broadly divided into three types: example-based colorization [16], [17], [18], [19], data-driven automatic colorization [16], [20], [21], [22], [23], and user-guided edit propagation [11], [12], [24], [25], [26], [27], [28], [29].

Example-based methods colorize by selecting a color from an example color image provided by the user. Welsh *et al.* used the similarity of image patches to select colors from an example color image [16]. Pierre *et al.* realized colorization with spatial consistency by selecting colors that minimize the total variation of the colors of the entire image [17], [18]. Varga and Sziranyi proposed an example-based method that uses a CNN [19]. Their CNN learns from a lot of monochrome-image and example-color-image pairs in advance. It can accurately colorize images that include objects with complex shapes. The example-based methods require users to provide an example color image that is very similar to the monochrome image to be colorized. Hence, they cannot be used in situations where users cannot prepare an example color image.

Data-driven methods use an image database to obtain a mapping from grayscale to color images. In particular, several very precise techniques using CNNs and large-scale training data sets have recently been reported [4], [5], [6], [7], [8], [9]. In data-driven methods, colors are determined on the basis of a large number of color images used in the pretraining. As a result, example images are not required for colorization. Iizuka *et al.* achieved vivid colorization through parallel learning of global and local image features [6]. Larsson *et al.* proposed a framework for *a posteriori* hue control to match a specific histogram following data-driven colorization [7]. Isola *et al.* used a framework called a conditional generalized adversarial network (GAN) that uses competition between two adversarial CNNs to achieve precise transformations of images with regard to various attributes including color [9]. The colors produced by these data-driven methods are stored in the database used for learning. Consequently, it is not always possible for the CNN to apply suitable colors to objects such as cars and clothing that vary widely in coloration. Also, although some methods allow the color of the whole image to be changed, it is not possible to change only the color of a specific object in the image. When colorized images are used in a broadcast program, objects whose correct color is known as a matter of historical fact must be colorized with that correct color. Therefore, a way that can easily change the color of individual objects is required in broadcast program production.

User-guided methods work by propagating manually provided colorization hints (some of the colors and strokes) to the surrounding areas. In the past, color propagation was based on low level similarities such as changes in luminance [18], [24], [25], [30], [31]. Pierre *et al.* proposed a method to combine example color images with colorization hints [18].

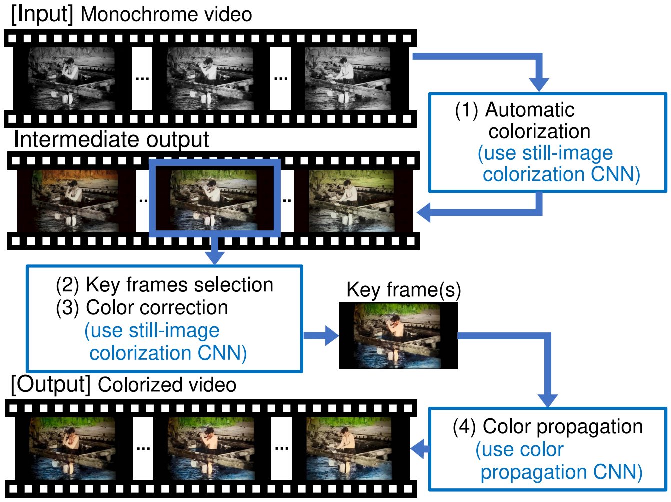
[Input] Monochrome video
(1) Automatic colorization (use still-image colorization CNN)
Intermediate output
(2) Key frames selection
(3) Color correction
(use still-image colorization CNN)
Key frame(s)
[Output] Colorized video
(4) Color propagation (use color propagation CNN)

Fig. 2. Colorization procedure of our framework. Steps (1) and (3) are performed by the still-image-colorization CNN, while step (4) is performed by the color-propagation CNN. Step (2) is performed manually.

Casaca *et al.*'s method divides the object into small areas by using Laplacian coordinates segmentation and colorizes each area with a color specified by the user [31]. This method does not require an example color image. However, since an object with a complicated shape is divided into a number of small areas, colorizing images containing many such objects requires a lot of manual editing. The advent of CNNs has made it possible to perform colorization with simpler operations for images containing objects with complicated shapes. Chang *et al.* proposed a method that controls the colors output by using a color palette [11]. Zhang *et al.* implemented a localized control based on manual instructions by training a CNN on emulated user instructions [12]. These methods are extremely innovative in that they make it easy for people to perform colorization according to their intentions without having to acquire expert image-editing skills. However, it is difficult for a CNN to distinguish color boundaries in regions of monochrome images where the luminance difference is small. There is a possibility that this problem will be solved in the future by using an affinity learning module, as in the study of Wang *et al.* [32]. However, the current methods require many coloring hints in order to achieve the intended colorization in regions of the monochrome images where the luminance difference is small.

Video propagation techniques operate on pairs of frames in a video sequence, whereby information assigned to one frame is propagated to other frames [13], [14], [15]. These techniques are based on the fact that consecutive frames are often very similar, allowing propagation to be performed with high precision when the correlation between frames has been recognized. There is a clearly a strong relationship between the color information and monochrome images. Therefore, these techniques can be used to propagate colors from colorized key frames to other non-colorized frames. Liu *et al.* achieved high-precision propagation at a low computational cost by estimating the transformation parameters for propagation instead of directly propagating the color information [15]. Although these techniques are able to uniformly colorize whole video clips in a short time, they all assume that multiple color key frames are available.

Problems arise when attempting to precisely colorize long video sequences. To properly colorize videos of a certain length, it is necessary to prepare multiple key frames for two reasons. One is that the color propagation accuracy decreases as the distance between the key frame and the target frame increases. The other is that it is not possible to specify the colors of transient objects that do not appear in the key frames. In Liu *et al.*'s method, the color to be applied to the target frame is determined from only one key frame. Therefore, unless color consistency is maintained between key frames, color discontinuities will occur at places where the key frames are switched. Even if the set of key frames used for colorization is prepared using a still-image colorization technique in a user-guided system, it is still very difficult to create accurate enough colorization hints for a large number of frames so that the color tones do not change between key frames.

III. Proposed Framework

Here, we propose a practical video colorization framework that can easily reflect the user's intentions. The basic concept is that a CNN is used to automatically colorize the video, and then the user corrects only those video frames colored differently from his/her intentions. This semi-automatic colorization can reduce the amount of work users have to do without degrading the colorization quality. In particular, the color correction work can be done efficiently by using the user-guided still-image-colorization CNN and color-propagation CNN. All the user has to do is to correct the colors of a few key frames by using the user-guided still-image-colorization CNN. After that, the color-propagation CNN automatically corrects the colors of other frames by referring to the key frames.

Figure 2 shows the colorization procedure of our system. The colorization process consists of four steps: (1) In the fully automatic colorization step, the still-image-colorization CNN colorizes each frame extracted from the monochrome video to obtain a set of intermediate output frames. In this case, the CNN is not provided with any user hints about how the colorization should be performed. (2) In the key frame selection step, the user selects key frames from the set of intermediate output frames. (3) In the color correction step, if color corrections based on historical fact or other information are required, the user creates instructions corresponding to these corrections. The colors to be included in the user instructions are determined by experts, such as historians, who are familiar with the objects (e.g., buildings and automobiles) and cultural aspects (e.g., colors of clothes that were in fashion in the past) shown in the monochrome video. The experts decide the color to be corrected by referring to historical records and knowledge of the dyes and paints available at that time. It is common practice to have such experts supervise the colorization of TV programs and movies. They check the intermediate output frames and make sure the color of each object matches the historical facts. Then, they create a color correction table that describes the objects and the colors they should have. The user creates instructions by referring to the color correction table. The

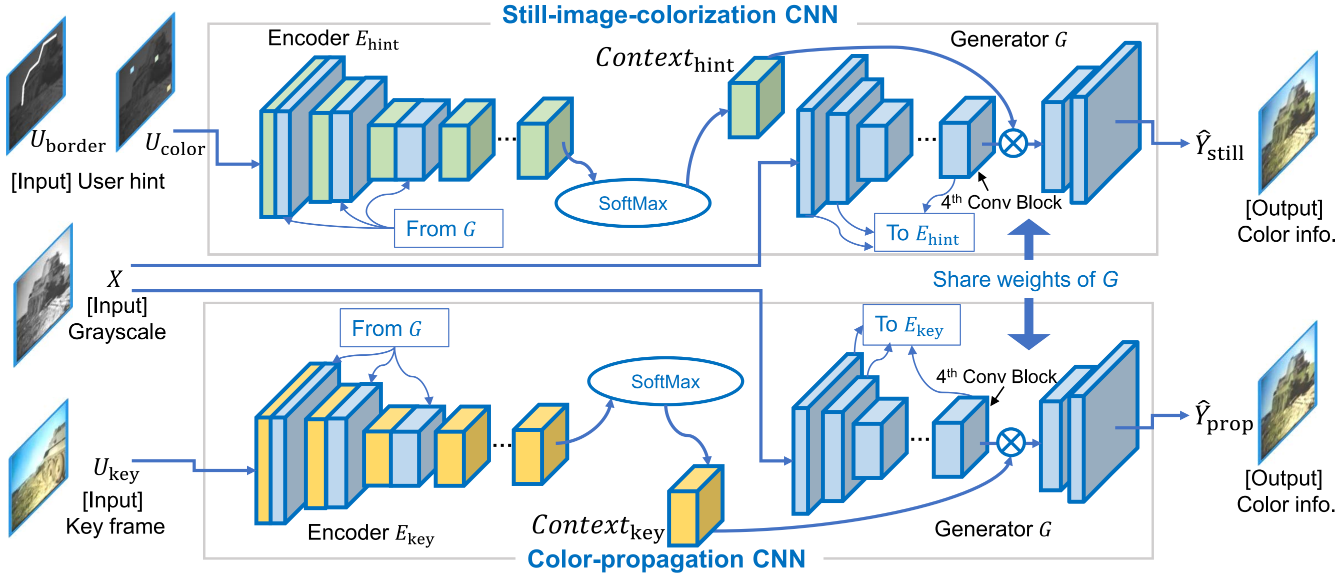

Fig. 3. Network structure of the still-image-colorization CNN and color-propagation CNN. The CNNs consist of a generator G that generates color information $\hat{Y}$, and encoders E_{hint} and E_{key} that control the output of G. The difference between these two CNNs is that one only uses E_{hint}, while the other only uses E_{key}. (G is exactly the same in both CNNs.) G is a color estimation network that outputs color information $\hat{Y}$ from an input grayscale image X. E_{hint} and E_{key} are networks that extract context information for use in correcting the output of G based on hints provided by the user. The color propagation CNN shown in the figure corresponds to the case where only one key frame is provided as input. When there are multiple input key frames, the operation is slightly different (See Fig. 4).

expert could also create the user instructions directly. Using these instructions, the still-image-colorization CNN colorizes the key-frames selected on the key frame selection step. As well, a CNN cannot fully understand user intentions. If the CNN cannot properly colorize in accordance with the user's intentions, the user makes manual color corrections. (4) In the color propagation step, the colors of the key frames specified by the user are propagated to the other uncolorized frames by the color-propagation CNN. The final result is a colorized video with consistent color tones between consecutive frames that appropriately reflects the user's intentions.

IV. Still-Image-Colorization CNN and Color-Propagation CNN

This section describes the network structures and learning methods of the two CNNs used in the proposed framework. Figure 3 shows the structure of the CNNs. The CNNs are extensions of Zhang *et al.*'s method [12], but the variety of user instructions that can be given to them are different. Zhang *et al.*'s method only takes instructions that specify colors, whereas our method also takes instructions describing the boundaries between regions of different colors and specifications of key frames.

The still-image-colorization CNN colorizes video by treating each constituent frame as an individual still image. The input consists of a grayscale image $X \in \mathbb{R}^{1\times H\times W}$ and two sets of user hints $U_{\text{color}} \in \mathbb{R}^{3\times H\times W}$ and $U_{\text{border}} \in \mathbb{R}^{1\times H\times W}$. The output consists of color information$\hat{Y}_{\text{still}} \in \mathbb{R}^{2\times H\times W}$. Here, H and W represent the height and width of the input images, respectively. U_{color} are hints about the colors of areas of the image, and U_{border} are hints about the positions of the color boundaries. For U_{color}, we used the approach proposed by Zhang *et al.* [12]. Each pixel of the first channel corresponding to the first dimension of U_{color} takes a value of 1 if a color is specified at this location, or -1 otherwise. The pixel values of the second and third channels provide the color information (i.e., a and b channels of the Lab color space). U_{border} is additional information that we propose to use, consisting of a matrix indicating whether or not each pixel is a color boundary. A pixel value is 1 if the user wants a color boundary at this location, or -1 otherwise. With U_{color} alone, it is difficult to colorize a region of constant luminance with multiple colors, because there are no clues in the monochrome image to determine the color boundaries. By using U_{border}, the user can teach the positions of the color boundaries to the CNN so that such regions can be colorized properly. The fully automatic colorization step (Step (1) in Figure 2) does not use user hints, so all pixels of U_{color} and U_{border} take -1.

The color-propagation CNN propagates the colors from the colorized key frames to the entire video. The input consists of a grayscale image X and a set of key frames $U_{\text{key}} \in \mathbb{R}^{R\times 3\times H\times W}$. The output consists of color information $\hat{Y}_{\text{prop}} \in \mathbb{R}^{2\times H\times W}$. R represents the number of input key frames. Unlike the conventional method [15] that only uses a single key frame, the color-propagation CNN can take any number of key frames as input. This makes it possible for it to process colorization discontinuities that may occur between key frames. Note that the operation is slightly different when there are multiple input key frames (See Fig. 4).

A. Network Architecture

As shown in Fig. 3, the two CNNs of the proposed framework consist of a generator G that generates color information and encoders E_{hint} and E_{key} for controlling the output of G.

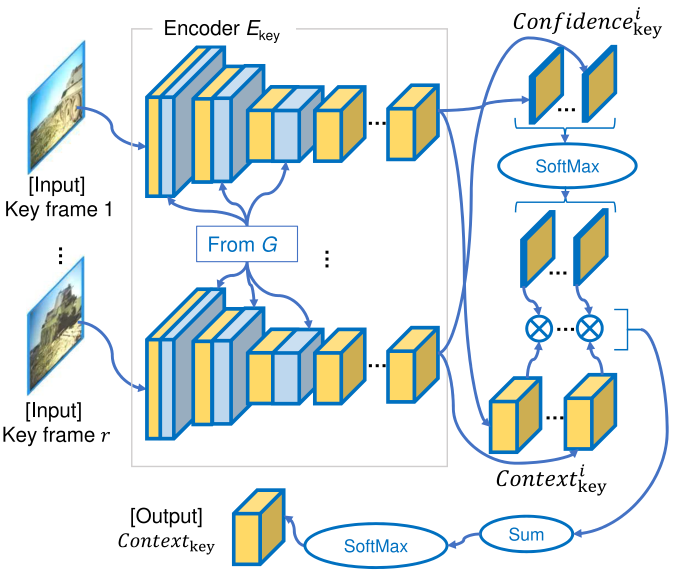

Fig. 4. Key frame encoder of the color-propagation CNN with multiple input key frames. E_{key} outputs $Context^{i}_{key}$ and $Confidence^{i}_{key}$ for each frame i. The $Context^{i}_{key}$s are merged by using $Confidence^{i}_{key}$ to obtain the output $Context_{key}$.

The structure of each CNN is outlined below. Details of the structures and hyperparameters can be found in Appendix A.

The generator G takes a grayscale image X as input and outputs color information $\hat{Y}$. The generater has the same structure as Zhang *et al.*'s method [12], consisting of 3 downsampling blocks, 4 convolution blocks, and 3 upsampling blocks. Each downsampling block consists of two or three convolution layers followed by a normalization layer and a nearest-neighbor downsampling layer. Each convolution block consists of three convolution layers followed by a normalization layer. Each upsampling block consists of a transposed convolution layer followed by one or two convolution layers and a normalization layer. The downsampling blocks and upsampling blocks are connected by three shortcut connections (See [12] or Appendix A for the shortcut connections that are omitted in Fig. 3). Finally, a convolution layer is used as a projection layer for outputting color information. When using the context matrix $Context \in \mathbb{R}^{512\times\frac{H}{8}\times\frac{W}{8}}$ produced by E_{hint} and E_{key}, the output $O_{G,conv4} \in \mathbb{R}^{512\times\frac{H}{8}\times\frac{W}{8}}$ of the fourth convolution block is input to the first upsampling layers after calculating the element-wise product.

The user hint encoder E_{hint} takes the intermediate layer outputs of G, the color specification information U_{color}, and the color boundary information U_{border} as input and outputs a context matrix $Context_{\text{hint}}$. E_{hint} consists of 3 downsampling blocks, 4 convolution blocks, and one projection layer for $Context_{\text{hint}}$. The intermediate outputs of G (the outputs of the layers before the downsampling layers and the output of the final convolution block) are concatenated with the inputs of E_{hint} (the inputs of the all downsampling blocks and the input of the first convolution block). The output of the projection layer $O_{E_{\text{hint}},proj} \in \mathbb{R}^{512\times\frac{H}{8}\times\frac{W}{8}}$ is processed by a softmax function in the channel direction (first dimension) to give $Context_{\text{hint}}$.

The key frame encoder E_{key} takes the intermediate layer outputs of G and the key frame U_{key} as input and outputs a context matrix $Context_{\text{key}}$. E_{key} consists of 3 downsampling blocks, 4 convolution blocks, and two projection layers for $Context_{\text{key}}$ and $Confidence_{\text{key}}$. The output of the projection layer is processed by a softmax function in the channel direction (first dimension) to give $Context_{\text{key}}$. The color-propagation CNN supports multiple key frame inputs. Figure 4 shows the procedure when there are multiple key frames. Here, E_{key} outputs $Confidence^{i}_{\text{key}} \in \mathbb{R}^{1\times\frac{H}{8}\times\frac{W}{8}}$ at the same time as $Context^{i}_{\text{key}}$ for each key frame i. $Confidence^{\text{i}}_{\text{key}}$ indicates the extent to which each pixel of the $Context^{i}_{\text{key}}$ plays a useful role in the colorization of grayscale image X. In order to use all of the $Context^{\text{i}}_{\text{key}}$, the quantity of data must be compressed to the same size as a single context matrix. In our framework, the data is compressed by combining $Context^{\text{i}}_{\text{key}}$ with weightings provided by $Confidence^{\text{i}}_{\text{key}}$. A softmax function is applied to the first dimension of the concatenated $Confidence^{\text{i}}_{\text{key}}$s to obtain the weight matrix $Weight \in \mathbb{R}^{R\times\frac{H}{8}\times\frac{W}{8}}$. Each matrix obtained by splitting $Weight$ in the first dimension is the weight map for each $Context^{\text{i}}_{\text{key}}$. Then, using the resulting weight matrix, all of the $Context^{\text{i}}_{\text{key}}$s are summed to obtain the final $Context_{\text{key}}$ value for multiple input key frames.

B. Optimization

In our framework, G is pretrained by using the cross-entropy as a loss function, as in Zhang *et al.*'s method [12], to learn the color distribution of the training data correctly. In the next step, the shared generator G and the encoders E_{hint}, E_{key} are trained simultaneously. During the training, the CNN outputs $\hat{Y}_{\text{still}}$ and $\hat{Y}_{\text{prop}}$ are always generated at the same time. The losses for optimization are calculated on the basis of the same function $\mathcal{L}$ for $\hat{Y}_{\text{still}}$ and $\hat{Y}_{\text{prop}}$. $\mathcal{L}$ is defined by

$$\mathcal{L} = \lambda_1\mathcal{L}_{\text{MSE}} + \lambda_2\mathcal{L}_{\text{GAN}} + \lambda_3\mathcal{L}_{\text{FM}}, \tag{1}$$

where $\mathcal{L}_{\text{MSE}}$ is the mean square error (MSE) loss, $\mathcal{L}_{\text{GAN}}$ and $\mathcal{L}_{\text{FM}}$ are the adversarial losses proposed in [33], and λ_i is the weighting of each loss function. We used $\lambda_1 = 1$, $\lambda_2 = 0$, and $\lambda_3 = 0$ for $\hat{Y}_{\text{still}}$, and used $\lambda_1 = 1$, $\lambda_2 = 0.0004$, and $\lambda_3 = 0.004$ for $\hat{Y}_{\text{prop}}$. The learning rate scheduler was ADADELTA [34].

C. Automatic Generation of User Hints

The training of the CNNs requires a dataset that includes monochrome images, ground-truth color information, user hints, and key frames. The datasets used in this study were automatically generated from color videos. First, a sequence of frames is extracted from a color video and converted to grayscale to produce a large number of color/grayscale image pairs. Next, U_{color} and U_{border} are automatically generated from the resulting color frame images, and U_{key} is automatically generated from the color frame sequence. U_{color} is generated by randomly sampling patches from the color images, as in Zhang *et al.*'s method [12]. U_{border} is also randomly generated by sampling random patches from the edge images of the color images. The edge image is created using

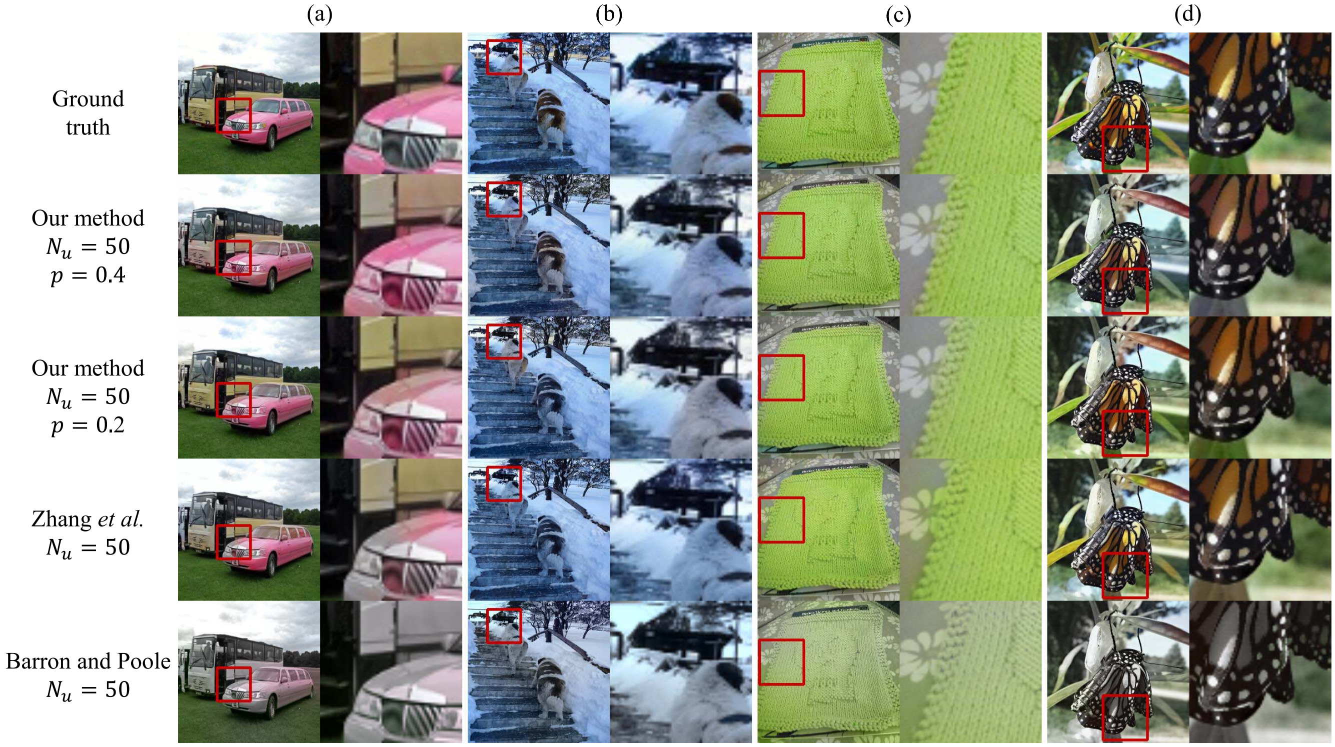
(a)
(b)
(c)
(d)
Ground truth
Our method $N_u = 50$ $p = 0.4$
Our method $N_u = 50$ $p = 0.2$
Zhang *et al.* $N_u = 50$
Barron and Poole $N_u = 50$

Fig. 5. Results from still-image-colorization methods [12], [30]. N_u is the number of user-specified points. p is the proportion of points that indicate color boundaries.

the Canny edge detection method [35]. First, the color image is projected to the Lab color space and the a and b channels are extracted. Then, edges are detected from each channel by using the Canny method. The edge image is the logical sum of the detection results. In the experiments reported below, the parameters of the Canny method are $m \times 0.66$ for minVal and $m \times 1.33$ for maxVal, where m is the median pixel value. U_{key} is produced by extracting sets of frames at random from each cut of a video.

V. Experimental Results

We used the ILSVRC2012 [36] and ACT datasets [37] for our evaluation. Since ILSVRC2012 is an image data set, it could not be used directly for training the color-propagation CNN. We therefore generated key frames by geometrically transforming the color images in the same way as in Liu *et al.* [15], by scaling, rotation, and translation. The ACT dataset consists of short video clips for the recognition of actions, and it was used directly for training the color-propagation CNN.

The evaluation criterion is the average value of the peak signal-to-noise ratio (PSNR) between the correct color image and output color image. Care must be taken when using pixel-level criteria to evaluate the accuracy of colorization [12]. For objects that have the same shape but often have different colors, the error between the outputting color and the ground truth may become excessively large, even if the CNN estimates a realistic color. For example, when red clothes are colored blue, this is not necessarily a mistake. However, our aim here is to facilitate colorization according to the user's intentions. Therefore, PSNR is a valid criterion when the user hints used in the evaluation are generated from the ground truth.

A. Still-Image Colorization

Table I shows how the PSNR changes when varying the number of user-specified points. The number of user-specified points corresponds to the number of mouse clicks performed by the user in manual operation. The number of points was calculated by assuming each color specification requires one point to specify one location, and each color boundary specification requires two points to designate one location (i.e., one line). We used graph-based image segmentation [38] and the DouglasPeucker algorithm [39] to approximate boundary lines with points. p is the proportion of points that indicate color boundaries. There is a difference between the color boundary hints generated by this graph-based emulation method and the color boundary hints that humans actually give. However, we chose this method as a means to emulate user-created color boundary hints in our implemented system. In our implementation, the user-created color boundary hints are a set of straight lines. The color boundary hints do not have to be straight lines, but it is difficult for users to draw free-form curves as intended. Therefore, we chose to use line-based color boundary hints. The user only needs to specify the start and end points to create a hint with the line connecting the points as the color boundary. We believe that our graph-based emulation method can generate boundary hints that are relatively close to those of our implementation.

As shown in table I, our method obtains a high PSNR when $p = 0.2$. However, the PSNR decreases when $p = 0.4$. This is because the color boundary designations do not include the

TABLE I
ACCURACY OF THE STILL-IMAGE COLORIZATION (PSNR). p IS THE PROPORTION OF POINTS THAT INDICATE COLOR BOUNDARIES. N_u IS THE NUMBER OF USER-SPECIFIED POINTS

Method	$N_u = 25$	$N_u = 50$	$N_u = 100$
Our($p = 0.2$)	**29.68**	**30.72**	**31.57**
Our($p = 0.4$)	29.13	30.05	30.93
Zhang *et al.* [12]	29.60	30.42	31.13
Barron and Poole[31]	25.89	26.92	28.96

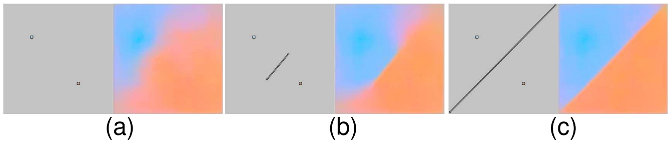

Fig. 6. Effects of using U_{border} to specify color boundaries. In each image, the positions of user hints U_{color} and U_{border} are shown on the left, and the CNN output is shown on the right. (a) Results obtained from a CNN provided with a uniformly gray input image and two points, one blue and one red. (b)(c) Results obtained when a single straight line color boundary is added by specifying two points in addition to the ones specified in (a).

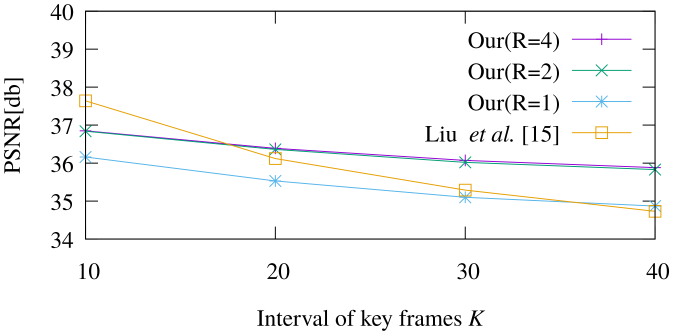

Fig. 7. Accuracy of the color propagation (PSNR). K is the interval of key frame extraction. R is the number of key frames input to our CNN.

color information itself, so the CNN is unable to select the correct color without sufficient color designations. Therefore, we believe that the user's intentions can be more accurately reflected with the same amount of effort by using a small number of color-boundary hints in combination with color hints.

Figure 5 shows the colorization results of our still-image-colorization CNN, Zhang *et al.*'s method [12], and Barron and Poole's method [30]. Compared with the other methods, our method tends to reduce color blur near the boundaries. In particular, in the images on the right side of column (c), there is a clear color boundary between the green fabric and the gray background. Also, in the images on the right side of column (a), our method colorizes the entire vehicle more uniformly than the other methods do. This suggests that the color-boundary hints help to set the color boundaries at the proper position. In the image on the left side of column (a), the color of the bus is mixed with the pink color of the car in our method when $p = 0.4$. This is probably because p is too high and the number of color hints is reduced. The color-boundary hints do not include the correct color values.

Figure 6 shows the behavior resulting from different color-boundary hints. Specifically, this figure shows the results of specifying the colors at two points in a single gray image, with a single color boundary between them. As the figure shows, when a color boundary is specified, the CNN only creates a clear color boundary along this line segment. This sort of behavior is difficult to achieve with color hints alone. Therefore, it can be said that the proposed method provides the user with greater flexibility to indicate the appearance of the intended colorization.

Our CNN learns only the narrow areas with strong color differences as color boundaries. Therefore, there is always a strong color difference at the position of the boundary indicated by the user. On the other hand, in actual color images, there is a region where the color changes gradually, like a gradation. In our method, users cannot explicitly give a hint specifying a gradual change. In such cases, it is necessary to give many color hints. This problem will be tackled in future work.

B. Color Propagation

We evaluated the accuracy of the color-propagation CNN by using the ACT datasets [37]. For comparison, we used Liu *et al.*'s method [15]. In each video of the evaluation data, we extracted a color frame every K frames for use as the key frames, as in [15]. When multiple key frames were input to our CNN, they were used in order of closeness to the target frame. The evaluation index was the average PSNR between the correct color frame and the output color frame.

Figure 7 shows the PSNR values obtained with key frame extraction intervals of $K = 10$, 20, 30 and 40. Our method achieved higher PSNRs, except for $K = 10$ when the number of key frames input to the CNN was $R \geq 2$. In particular, for $K = 40$, the improvement was approximately 3.3% relative to Liu *et al.*'s method when the number of key frames input to the CNN was $R = 4$. In both methods, the PSNR tends to decrease as K increases, but this decrease is smaller in the proposed method than in Liu *et al.*'s method. Liu *et al.*'s method does not directly estimate color images; instead, it calculates a conversion formula between two frames. However, it is thought that the precision is smaller when K is fairly large because of the difficulty of estimating a correct conversion formula due to complex deformations and movements of objects between the key frame and the target frame. On the other hand, the proposed method performs color propagation by modifying G, which is a fully automatic colorization CNN. That is, even if the color propagation of E_{key} does not function well, it is still possible that G may produce a colorization close to the ground truth. This is why the proposed method achieves a higher PSNR than Liu *et al.*'s method even when $K = 40$, $R = 1$ (i.e., when it is given the same amount of information). As a result, the proposed method appears to be robust against complex deformations and movements of objects.

On the other hand, Liu *et al.*'s method is more accurate when $K = 10$. When K is small, the gaps between the key frames and target frames are very small. In most cases, this means the conversion formula between two frames is also simple, so Liu *et al.*'s method can produce more accurate estimates. If key frames can be prepared every ten frames,

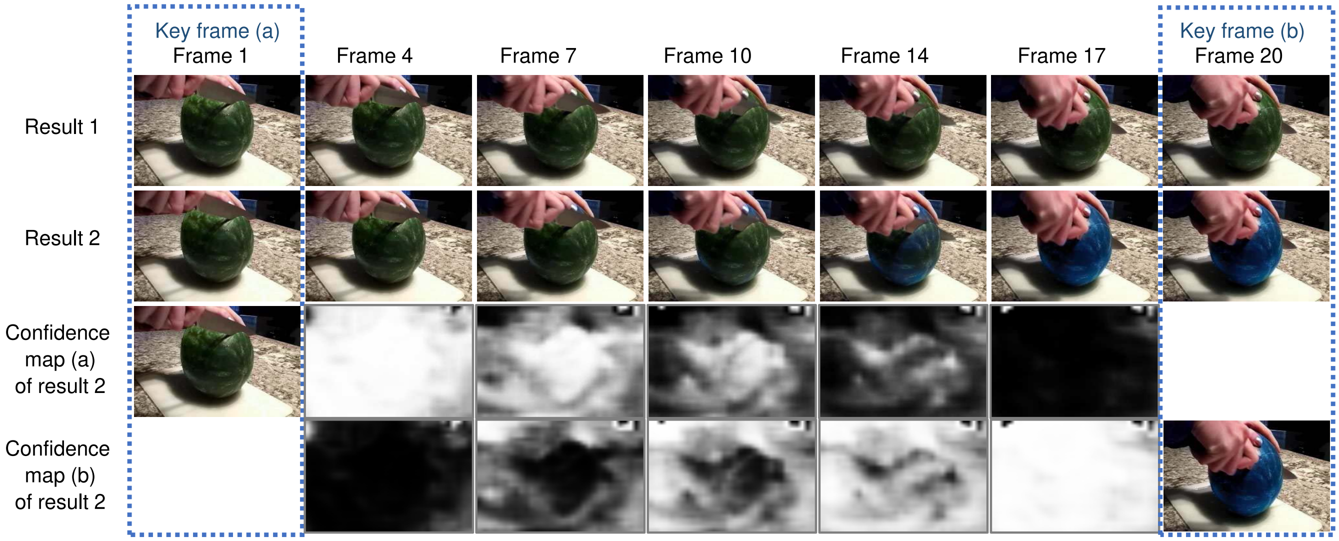

Fig. 8. Effects of key frame color on the colorization of videos. The first and second rows show the results obtained after colorizing the first and last frames with the proposed still-image-colorization CNN and then using these as key frames for color propagation to the other frames. In the second row, the color of the watermelon was intentionally changed between the key frames. The third and fourth rows show $\mathit{Confidence}^{i}_{\mathrm{key}}$, which is the output of the key frame encoder E_{key}. The white area is the area where the value of $\mathit{Confidence}^{i}_{\mathrm{key}}$ is high, which means that the CNN is focused on that area. Most of the colors of frame 4 are propagated from key frame (a) while those of frame 17 are propagated from key frame (b).

Liu *et al.*'s method works very well. In practice, however, it takes a lot of time to prepare so many key frames. Also, in Liu *et al.*'s method where it is only possible to refer to a single key frame, color discontinuities are liable to occur when switching from one key frame to another. To prevent color discontinuities, the key frames have to be prepared in order to ensure they are consistently colorized; this is a very difficult task.

In Figure 7, the difference in accuracy between $R = 2$ and $R = 4$ is smaller than that between $R = 1$ and $R = 2$. This indicates that the CNN may not be able to utilize the information of the third and subsequent key frames so well. Table II shows the average $\mathit{Confidence}^{i}_{key}$ values of key frames sorted by distance when $R = 4$. The distance means the number of other frames between the target frame and each key frame in the video. The average $\mathit{Confidence}^{i}_{key}$ values indicate which key frame the network prioritized for colorization. Our CNN attaches importance to closer key frames. This may be because the closer the key frame is to the target frame, the more similar the image is to the target frame. Our CNN depends on the closest pair of key frames for about 75% of the information used for colorization. This means that the information of two neighboring frames is very important for estimating the motion of an object. As well, the CNN does not use distant key frames so well. Therefore, we can conclude that simply increasing the number of key frames does not significantly increase the accuracy.

On the other hand, the proposed method can smooth out the changes in colorization by absorbing the changes in color between key frames. Fig. 8 shows example outputs of our method. The second row shows an example when a key frame having a very unusual color is used. The first frame, where the watermelon is colored green, is used as key frame (a), and the final frame, where the watermelon is colored blue, is used as key frame (b). The information supplied to the CNN consists entirely of images; it includes no information on the distance between the colorization target frame and the key frames in the video. Nonetheless, the watermelon becomes increasingly blue in the frames that are closer to key frame (b). This evidence supports the conclusion that E_{key} can output suitable values for *Confidence* on the basis of the correlation between the target frame and key frame. From the above, it can be said that—unlike Liu *et al.*'s method—it is not necessary to maintain strict color consistency between key frames when using our method. Therefore, we believe that our method makes it easier to colorize videos in a short time.

TABLE II
AVERAGE $\mathit{Confidence}^{i}_{key}$ VALUES OF KEY FRAMES WHEN $R = 4$. FIRST, THE KEY FRAMES WERE SORTED IN ORDER OF DECREASING DISTANCE FROM THE TARGET FRAME IN THE VIDEO; THEN, THE AVERAGE VALUES WERE CALCULATED

Closest	Second closest	Third closest	Farthest
0.54	0.22	0.14	0.11

C. Video Colorization Framework

We evaluated the time required for users to colorize video in our framework. We asked a technical engineer employed by a broadcasting station to colorize monochrome videos (30 frames each). We compared our method with Zhang *et al.*'s method [12], Iizuka *et al.*'s method [6] and a completely manual method. Two color videos were selected from the ACT dataset [37] as colorization targets. For the color of each object included in the videos, a color-correspondence table was created on the basis of the ground truth in advance. The two color videos were converted to monochrome videos; then, the engineer colorized each of the monochrome videos. First, the engineer colorized the video by using the completely

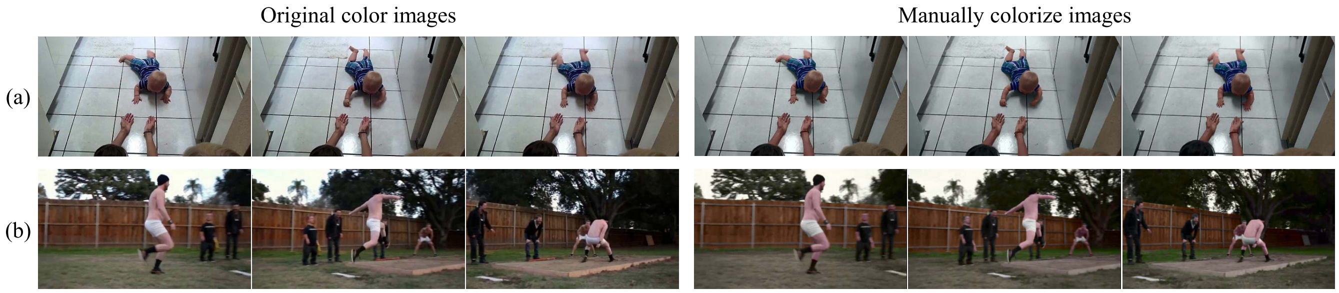

Fig. 9. Frames of video used to evaluate the time spent on colorization work. The left side shows the original images, and the right side shows the manually colorized images. The original videos are included in the ACT dataset [37]. The evaluation used only the first 30 frames of the original videos.

TABLE III
TIME REQUIRED TO COLORIZE THE ENTIRE 30-FRAME IMAGE AND THE FIRST FRAME. THE TIME IS THE TOTAL TIME OF CNN-BASED COLORIZATION AND MANUAL CORRECTION

	(a)		(b)	
Method	1st frame	30 frames	1st frame	30 frames
Ours	12m1s	17m	10m0s	1h9m
Zhang *et al.* [12]	13m54s	6h25m	10m13s	5h20m
Iizuka *et al.* [6]	15m50s	8h28m	16m1s	9h18m
Manual	18m42s	9h22m	22m0s	12h23m

manual method while referring to the color-correspondence table. Figure 9 shows parts of the manually colorized videos. In the figure, (a) is a video in which the subject and the camera remain mostly still, while (b) is a video in which the subject and the camera move quickly. The compared methods are not video colorization frameworks. Therefore, colorization procedures when using our method and compared methods were based on their properties, as follows:

- In our method, the first frame was colorized by the still-image-colorization CNN in accordance with the engineer's instructions. Fine areas that could not be completely colored by the CNN were manually corrected. Next, the subsequent frames were colorized by the color-propagation CNN using the first frame as the key frame. When the color-propagation CNN could not colorize a frame correctly, the engineer manually corrected it and added it as a new key frame.
- In Zhang *et al.*'s method, all frames were colorized by the CNN in accordance with the engineer's instructions. After that, fine areas that could not be completely colored by the CNN were manually corrected.
- In Iizuka *et al.*'s method, all frames were fully automatically colorized and then manually corrected.

Table III shows the work time for each colorization framework. The work time for first frame colorization in our framework was less than or equal to that of Zhang *et al.*'s method. In the colorization of the first frame, our framework required an average of 4m14s for the CNN colorization with user instructions and 6m47s for the subsequent manual correction. On the other hand, Zhang *et al.*'s method required 3m33s for the CNN colorization with user instructions and 8m31s for the subsequent manual correction. We think that our framework increases the time taken to create user instructions because it allows the color boundaries to be specified. On the other hand, the time for manual correction was reduced, and the total work time was also slightly reduced. In the colorization of the entire 30 frames, our framework did not require the engineer to correct many of the frames manually because the color-propagation CNN colorized the frames well. With regard to the total work times of (a) and (b), our framework reduced the work time by a factor of approximately 8 compared with Zhang *et al.*'s method. In addition, the work time was only about 1/15th that of the manual colorization.

Now, let us describe the limitations of our framework and future issues. From table III , the colorization of video (b) took more than twice as long as the colorization of video (a). This is because (b) includes fast-moving objects, and it was necessary to create many key frames. Two key frames were created for video (a), whereas seven key frames were created for video (b). In video (b), a new object appears approximately every 5 frames, so it was necessary to colorize the frame containing the new object and create a key frame each time. Thus, the effect of our framework is strongly influenced by the amount of motion of the objects in the video. In particular, we will have to devise a way to properly and automatically colorize objects that do not exist in the key frame but exist in the target frame.

We implemented our framework on a GTX1070 GPU; it takes less than 20 milliseconds to colorize one frame (the same as other colorization methods [6,11]). If an ideal fully automatic colorization CNN with no errors could be realized, 30 frames could be colorized in 600 milliseconds or less. That is, almost all of the work time measured in the experiment was time spent in manual operation, and most of it was color correction work. This indicates that there is still a lot of room for reducing the work time by improving the colorization accuracy.

D. Practical Use of Our Colorization Framework

We put the proposed framework to practical use in broadcast program production [40], where it was used to colorize monochrome video footage dating back to around 1939. Note that the CNN used in the actual system differs from the one described in Section III with regard to the following points:

- We used a local enhancer CNN as proposed in the *pix2pixHD* framework [33] to expand the output to 2K resolution.
- For the learning data, we used about 18 months of TV programs broadcast during 2017.

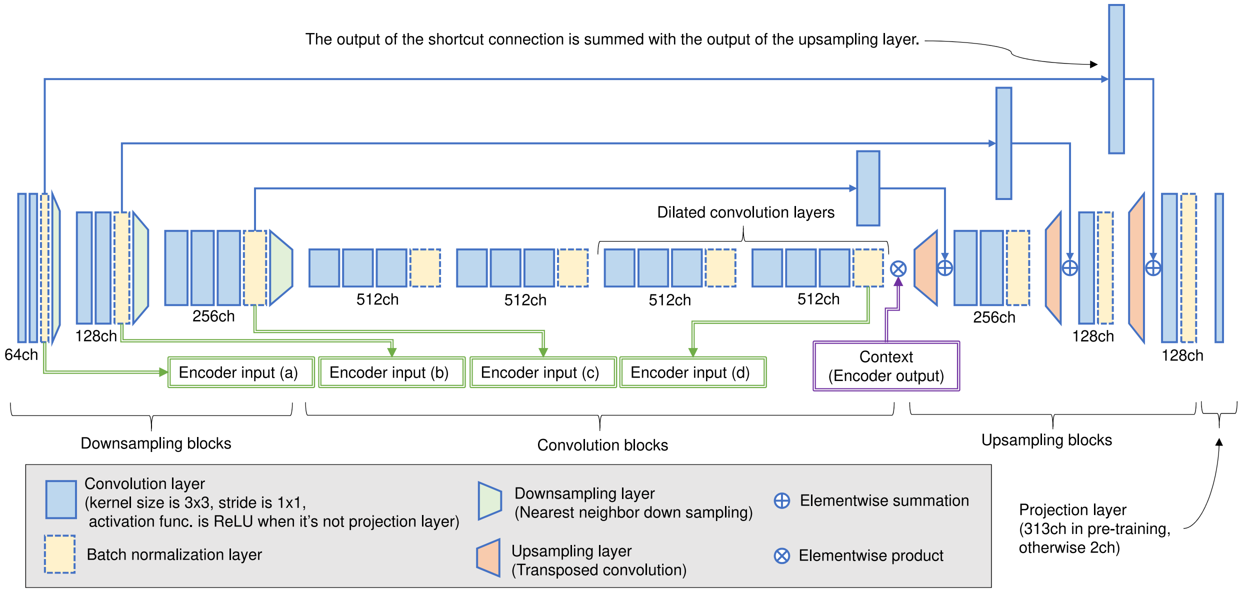

Fig. 10. Details of the generator G.

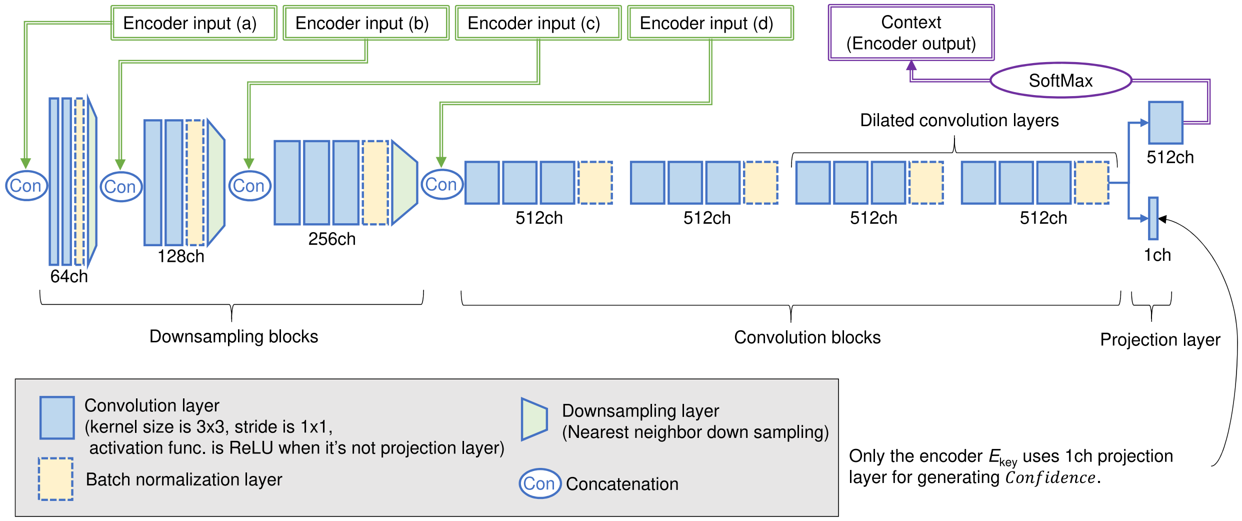

Fig. 11. Details of the encoders E_{hint} and E_{key}.

- To match the learning data to the condition of the film to be colorized, Gaussian random noise was added with 50% probability to the monochrome images input to the network. This change was essential to bridge the gap between the pristine modern broadcast video and the noisy historical film footage.

Figure 1 shows some examples of images colorized using the proposed framework for use in TV broadcasts. The colorization workload required for a broadcast program of approximately one hour was about 36 person-days. This is remarkable when we consider that previous colorization work of a similar scale required approximately 2, 250 person-days of effort. Although this is not a strict comparison because the target videos for colorization were not the same, the video resolutions and frame rates were the same, and the videos were similar in content. Thus, our framework may indeed be able to reduce workload to about 1/62 of that of manual colorization. It should be pointed out as well that the workload figures include time spent doing work other than in this framework, including colorization-related investigations and the like. The program production team said that our framework enabled the experts to verify the color correction results in real time, and that this greatly reduced working time. They said that in past productions, a lot of time was taken up by the experts pointing out color errors and the subsequent manual color corrections, a process that had to be repeated over many days. On the other hand, with our framework, errors pointed out by experts can be immediately reflected in the colorization result, and it can immediately be reconfirmed whether the adjusted color is correct. Therefore, the color verification work can be completed in one meeting. This may have led to a significant reduction in work time.

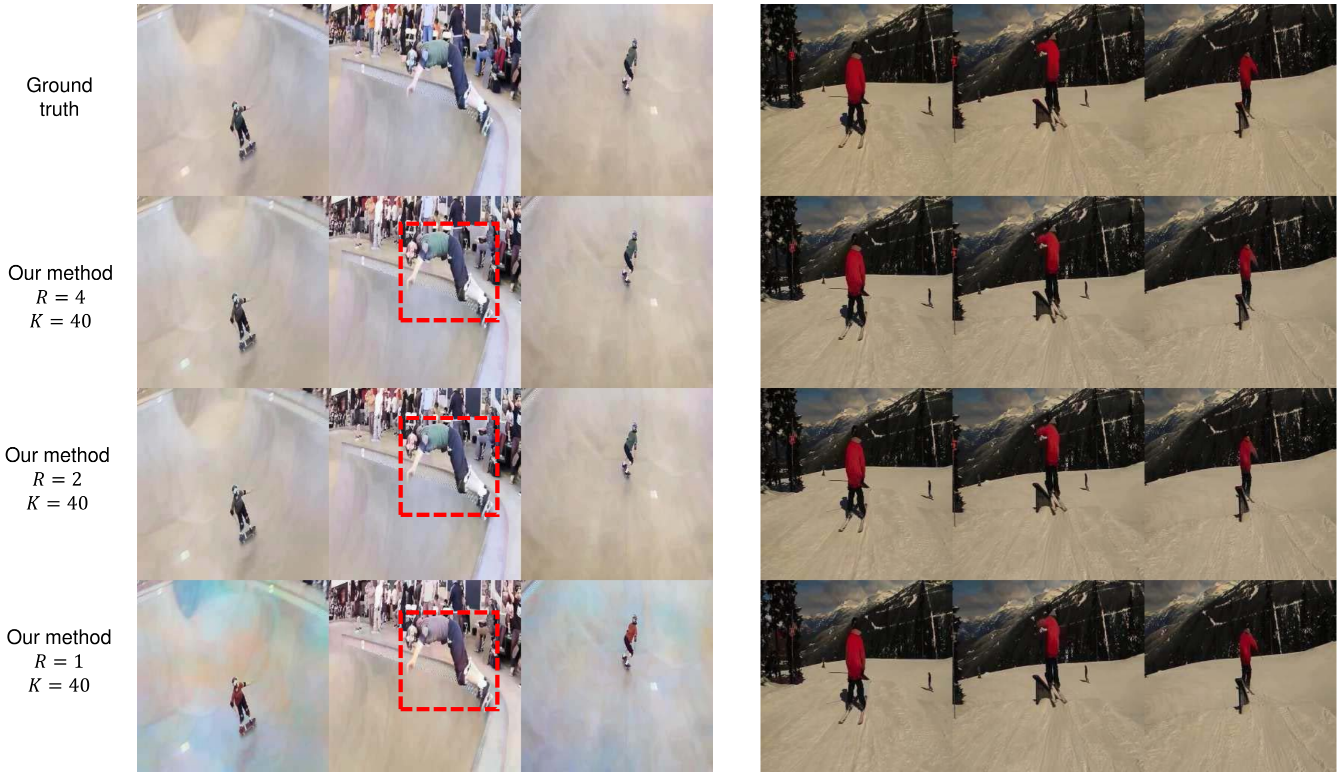
Ground truth
Our method $R = 4$ $K = 40$
Our method $R = 2$ $K = 40$
Our method $R = 1$ $K = 40$

Fig. 12. Results of our color-propagation CNN with the interval of key frame extraction K set to 40. R is the number of key frames input to the CNN. The accuracy of color propagation decreases in the case of a cut with large interframe difference (the left half of the figure). Even in such cases, the accuracy is improved by increasing R in our framework.

On the other hand, it became clear that the still-image-colorization CNN did not work properly for objects that appeared small in the old film images. Small objects in verification images, which are pristine modern images, were colorized relatively correctly. Therefore, one reason could have been that the still-image-colorization CNN was trained on pristine modern color images rather than noisy historical footage. This meant that we had to use manual colorization for regions that could not be properly colorized by the still-image-colorization CNN, such as the collar badges in the upper row of figure 1. On the other hand, the color-propagation CNN worked well even when the key frames included parts that had been colorized manually. The resolution of the issue affecting the still-image-colorization CNN is worthy of further study.

VI. Conclusion

We proposed a colorization framework for monochrome videos in which it is possible to perform colorization reflecting the user's intentions. We have made it easy for the user to perform correction work in a short space of time through our newly proposed combination of two CNNs: a still-image-colorization CNN and a color-propagation CNN. We confirmed that the incorporation of a new color-boundary hinting method into the still-image-colorization CNN resulted in an improvement in the PSNRs of the colorized images in comparison with those of previous methods. Furthermore, by making it possible for the user to designate color boundaries, this framework offers the user more choices of how to issue hints to the CNN and thereby allows colorization to be performed more flexibly according to the user's intentions. Our color-propagation CNN colorizes entire motion image sequences in accordance with key frames colorized by the still-image-colorization CNN. With a structure that takes multiple key frames as input, we confirmed that an improvement of PSNR of approximately 3.3% can be achieved when $K = 40$ and $R = 4$. In addition, the user can easily create color videos with a low level of color discontinuity without having to maintain strict color shade consistency between key frames. In an experiment on colorizing two 30-frame videos, our framework reduced the colorization work time to about $1/8$th that of the conventional method. We used the proposed framework in actual broadcast program production and showed the possibility of colorizing video in a significantly shorter time compared with manual colorization.

Appendix A
Network Architecture Details

This section describes the details of the networks (the generator G, the encoders E_{hint} and E_{key}) described in Section IV.

A. Generator G

Fig. 10 shows the details of the generator G. The generator consists of 3 downsampling blocks, 4 convolution blocks, and 3 upsampling blocks. Each downsampling block consists of two or three convolution layers followed by a batch normalization layer and a nearest-neighbor downsampling layer. Each convolution block consists of three convolution layers followed by a batch normalization layer. Each upsampling block consists of a transposed convolution layer followed by one or two convolution layers and a batch normalization layer. The downsampling blocks and upsampling blocks are connected by three shortcut connections with one convolution layer. Finally, there is a convolution layer that acts as a projection layer for the output.

The kernel size is 3 × 3, and the stride is 1 × 1 for all of the convolutional layers. The convolution layers, except the projection layer, use ReLU as an activation function. The projection layer does not use an activation function. The dilation parameter in each convolution layer for the second and third convolution blocks is 2 and the parameters in the other layers are each 1. As shown the figure, the number of channels in the convolution layer for the first downsampling block is 64, doubling with each reduction in spatial resolution. The number of channels in the projection layer is 2 except during pre-training, and the projection layer outputs color information. The projection layer for pre-training outputs a color distribution, and the number of channels depends on the colors in the training data. In this study, the color was quantized into 400 classes by dividing both the a-channel and b-channel in the Lab color space into 20, and of these, we used 313 color classes, as in Zhang *et al.* [12]. Classes that were not used had no, or few, instances in the training data.

B. Encoders E_{hint} and E_{key}

Fig. 11 shows the details of the encoders E_{hint} and E_{key}. The encoders consist of 3 downsampling blocks, 4 convolution blocks, and a projection layer for *Context*. Only E_{key} has an additional projection layer for *Confidence*. Except for the projection layer, the parameters of the layers are the same as the downsampling blocks and convolution blocks in *G*. The number of channels in the projection layer for *Context* is 512. The number of channels in the projection layer for *Confidence* is 1.

Appendix B
Additional Colorization Results

Fig. 12 shows the colorization results of the experiment described in Section V-B.

References

[1] (2014). *The Mind of Evil DVD—Babelcolour*. Accessed: May 17, 2019. [Online]. Available: http://babelcolour.com/dvd-work/mind-of-evil/

[2] C. Krauthammer. (1987). *Essay: Casablanca in Color? I'm Shocked, Shocked!* Accessed: May 17, 2019. [Online]. Available: http://content.time.com/time/magazine/article/0,9171,963207,00.html

[3] (2018). *Ai Photo Colorization Brings Memories of Prewar Okinawa*. Accessed: May 17, 2019. [Online]. Available: https://withnews.jp/article/f0180123004qq000000000000000G00110701qq000016643A

[4] Z. Cheng, Q. Yang, and B. Sheng, "Deep colorization," in *Proc. Int. Conf. Comput. Vis. (ICCV)*, vol. 14, 2015, pp. 415–423.

[5] A. Deshpande, J. Rock, and D. Forsyth, "Learning large-scale automatic image colorization," in *Proc. Int. Conf. Comput. Vis. (ICCV)*, vol. 14, 2015, pp. 567–575.

[6] S. Iizuka, E. Simo-Serra, and H. Ishikawa, "Let there be color! Joint end-to-end learning of global and local image priors for automatic image colorization with simultaneous classification," *ACM Trans. Graph.*, vol. 35, no. 4, pp. 1–11, 2016.

[7] G. Larsson, M. Maire, and G. Shakhnarovich, "Learning representations for automatic colorization," in *Proc. Eur. Conf. Comput. Vis. (ECCV)*, 2016, pp. 577–593.

[8] R. Zhang, P. Isola, and A. A. Efros, "Colorful image colorization," in *Proc. Eur. Conf. Comput. Vis. (ECCV)*, 2016, pp. 649–666.

[9] P. Isola, J.-Y. Zhu, T. Zhou, and A. A. Efros, "Image-to-image translation with conditional adversarial networks," in *Proc. Conf. Comput. Vis. Pattern Recognit. (CVPR)*, vol. 1, 2017, pp. 5967–5976.

[10] Y. LeCun, L. Bottou, Y. Bengio, and P. Haffner, "Gradient-based learning applied to document recognition," *Proc. IEEE*, vol. 86, no. 11, pp. 2278–2324, May 1998.

[11] H. Chang, O. Fried, Y. Liu, S. DiVerdi, and A. Finkelstein, "Palette-based photo recoloring," *ACM Trans. Graph.*, vol. 34, no. 4, pp. 1–11, 2015.

[12] R. Zhang *et al.*, "Real-time user-guided image colorization with learned deep priors," *ACM Trans. Graph.*, vol. 36, no. 4, pp. 1–11, 2017.

[13] V. Jampani, M. Kiefel, and P. V. Gehler, "Learning sparse high dimensional filters: Image filtering, dense crfs and bilateral neural networks," in *Proc. Conf. Comput. Vis. Pattern Recognit. (CVPR)*, 2016, pp. 4452–4461.

[14] V. Jampani, R. Gadde, and P. V. Gehler, "Video propagation networks," in *Proc. Conf. Comput. Vis. Pattern Recognit. (CVPR)*, 2017, pp. 3154–3164.

[15] S. Liu *et al.*, "Switchable temporal propagation network," in *Proc. Eur. Conf. Comput. Vis. (ECCV)*, 2018, pp. 89–104.

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

[17] F. Pierre, J.-F. Aujol, A. Bugeau, N. Papadakis, and V.-T. Ta, "Exemplar-based colorization in RGB color space," in *Proc. IEEE Int. Conf. Image Process. (ICIP)*, 2014, pp. 625–629.

[18] F. Pierre, J.-F. Aujol, A. Bugeau, N. Papadakis, and V.-T. Ta, "Luminance-chrominance model for image colorization," *Soc. Ind. Appl. Math.*, vol. 8, no. 1, pp. 536–563, 2015.

[19] D. Varga and T. Sziranyi, "Twin deep convolutional neural network for example-based image colorization," in *Proc. Int. Conf. Comput. Anal. Images Patterns (CAIP)*, 2017, pp. 184–195.

[20] R. Ironi, D. Cohen-Or, and D. Lischinski, "Colorization by example," in *Proc. Eurograph. Symp. Rendering Techn.*, 2005, pp. 201–210.

[21] Y. Liu, M. F. Cohen, M. Uyttendaele, and S. Rusinkiewicz, "AutoStyle: Automatic style transfer from image collections to users' images," *Comput. Graph. Forum*, vol. 33, no. 4, pp. 21–31, 2014.

[22] A. Y. S. Chia *et al.*, "Semantic colorization with Internet images," *ACM Trans. Graph.*, vol. 30, no. 6, pp. 1–8, 2011.

[23] E. Reinhard, M. Ashikhmin, B. Gooch, and P. Shirley, "Color transfer between images," *IEEE Comput. Graph. Appl.*, vol. 21, no. 5, pp. 34–41, Oct. 2001.

[24] Y. Qu, T.-T. Wong, and P.-A. Heng, "Manga colorization," *ACM Trans. Graph.*, vol. 25, pp. 1214–1220, Jun. 2006.

[25] Q. Luan, F. Wen, D. Cohen-Or, L. Liang, Y.-Q. Xu, and H.-Y. Shum, "Natural image colorization," in *Proc. Eurograph. Conf. Render. Techn. (EGSR)*, 2007, pp. 309–320.

[26] Y. Huang, Y. Tung, J. Chen, S. Wang, and J. Wu, "An adaptive edge detection based colorization algorithm and its applications," in *Proc. ACM Int. Conf. Multimedia*, 2005, pp. 351–354.

[27] X. Chen, D. Zou, Q. Zhao, and P. Tan, "Manifold preserving edit propagation," *ACM Trans. Graph.*, vol. 31, no. 6, pp. 1–7, 2012.

[28] L. Xu, Q. Yan, and J. Jia, "A sparse control model for image and video editing," *ACM Trans. Graph.*, vol. 32, no. 6, pp. 1–10, 2013.

[29] Y. Endo, S. Iizuka, Y. Kanamori, and J. Mitani, "DeepProp: Extracting deep features from a single image for edit propagation," *Comput. Graph. Forum*, vol. 35, no. 2, pp. 189–201, 2016.

[30] J. T. Barron and B. Poole, "The fast bilateral solver," in *Proc. Eur. Conf. Comput. Vis. (ECCV)*, 2016, pp. 617–632.

[31] W. Casaca, M. Colnago, and L. G. Nonato, "Interactive image colorization using Laplacian coordinates," in *Proc. Comput. Anal. Images Patterns (CAIP)*, 2015, pp. 675–686.

[32] Y. Wang, Y. Niu, P. Duan, J. Lin, and Y. Zheng, "Deep propagation based image matting," in *Proc. 27th Int. Joint Conf. Artif. Intell. (IJCAI)*, 2018, pp. 999–1006.

[33] T.-C. Wang, M.-Y. Liu, J.-Y. Zhu, A. Tao, J. Kautz, and B. Catanzaro, "High-resolution image synthesis and semantic manipulation with conditional GANs," in *Proc. Conf. Comput. Vis. Pattern Recognit. (CVPR)*, 2018, pp. 8798–8807.

[34] M. D. Zeiler, "AdaDelta: An adaptive learning rate method," 2012. [Online]. Available: arxiv.abs/1212.5701.

[35] J. F. Canny, “A computational approach to edge-detection,” *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. PAMI-8, no. 6, pp. 679–698, Nov. 1986.

[36] O. Russakovsky *et al.*, “ImageNet large scale visual recognition challenge,” *Int. J. Comput. Vis.*, vol. 115, no. 3, pp. 211–252, 2015.

[37] X. Wang, A. Farhadi, and A. Gupta, “Actions ~ transformations,” in *Proc. Conf. Comput. Vis. Pattern Recognit. (CVPR)*, 2016, pp. 2658–2667.

[38] P. F. Felzenszwalb and D. P. Huttenlocher, “Efficient graph-based image segmentation,” *Int. J. Comput. Vis.*, vol. 59, no. 2, pp. 167–181, 2004.

[39] M. Visvalingam and J. D. Whyatt, “The Douglas–Peucker algorithm for line simplification: Re-evaluation through visualization,” *Comput. Graph. Forum*, vol. 9, no. 3, pp. 213–228, 1990.

[40] *NHK special—The Nomonhan Incident NHK (Japan Broadcasting Coopration) Broadcasts on August 15 in Japan*, Japan Broadcast. Cooprat., Tokyo, Japan, 2018.

Rei Endo received the B.E., M.E., and Ph.D. degrees in engineering from Keio University, Japan, in 2008, 2010, and 2013, respectively. He is a Research Engineer with the Science and Technology Research Laboratories, NHK (Japan Broadcasting Corporation). His research interest includes the development of computer vision and broadcasting services. He is a Member of IPSJ and ITE.

Yoshihiko Kawai received the B.E., M.E., and Ph.D. degrees in engineering from Osaka University, Japan, in 1999, 2001, and 2010, respectively. He is a Senior Manager with the Science and Technology Research Laboratories, NHK (Japan Broadcasting Corporation). His research interest includes image recognition. He is a Member of ITE.

Takahiro Mochizuki received the B.E., M.E., and Ph.D. degrees in engineering from the Tokyo Institute of Technology in 1994, 1996, and 2010, respectively. He is a Senior Research Engineer with the Science and Technology Research Laboratories, NHK (Japan Broadcasting Corporation). His research interests include image processing, image retrieval, and video summarization. He is a Member of IEICE and ITE.