Received March 3, 2019, accepted March 17, 2019, date of publication March 25, 2019, date of current version April 5, 2019.

Digital Object Identifier 10.1109/ACCESS.2019.2906472

Recursive Conditional Generative Adversarial Networks for Video Transformation

SAN KIM AND DOUG YOUNG SUH, (Member, IEEE)
Department of Electrical Engineering, Kyung Hee University, Seoul, South Korea

Corresponding author: Doug Young Suh (suh@khu.ac.kr)

This research was supported by the Korea Electric Power Corporation (Grant number: R18XA02).

ABSTRACT Conditional generative adversarial networks (cGANs) are used in various transformation applications, such as super-resolution, colorization, image denoising, and image inpainting. So far, cGANs have been applied to the transformation of still images, but their use could be extended to the transformation of video contents, which has a much larger market. This paper considers problems with the cGAN-based transformation of video contents. The major problem is flickering caused by the discontinuity between adjacent image frames. Several postprocessing algorithms have been proposed to reduce that effect after transformation. We propose a recursive cGAN in which the previous output frame is used as an input in addition to the current input frame to reduce the flickering effect without losing the objective quality of each image. Compared with previous postprocessing algorithms, our approach performed better in terms of various evaluation metrics for video contents.

INDEX TERMS Image-to-image transformation, generative adversarial network, reducing flicker, video transformation.

I. INTRODUCTION

Image-to-image transformation is the transformation of an input image into a desired output image. Many applications have already been developed including sketch2photo [11], which converts a sketch image to a real image, and image quilting [3], which synthesizes an image with a desired shape and texture. More examples are colorization [4], image denoising [1], [6], image inpainting [2], and super-resolution [5].

In recent years, convolutional neural networks (CNNs) have been used to perform various image-to-image transformation tasks [7]–[10]. CNN learns to minimize the loss function, which is the goal of evaluating the quality of the results. The learning process is automated, but it requires a lot of manual effort to design an effective loss function. However, if we simply set CNN's loss function to minimize the Euclidean distance between the predicted pixels and the ground truth pixels, it would tend to cause blurry results. This is because the Euclidean distance is minimized by averaging all valid outputs. Setting up a loss function that forces CNN to do what we really want is a difficult problem and generally requires specialized knowledge.

Therefore, in GANs [15], the loss function is automatically learned (instead of setting a specific loss function) so that predicted pixels and ground truth are not distinguished. GANs simultaneously perform faking and identifying in the learning process so that the generated distribution becomes equal to the true distribution. GANs consist of a generative network and a discriminative network. In GANs, the discriminative network learns to distinguish the generated image from the ground truth, while at the same time the generative network learns the true distribution by faking the discriminative network. Therefore, GANs have been widely used for creating images [38]–[40].

GANs learn only the true distribution of data, but conditional GANs (cGANs [16]) can learn the conditional distribution of data. A generative network of cGANs uses the noise and condition data together as input, whereas a generative network of GANs uses only noise. Thus, we can use cGANs for image transformation. A paired image transformation program called pix2pix [12] has already been developed for applications such as those shown in Fig. 1. This makes it possible to create various applications without specialized knowledge of the application what we want to make. Using this cGAN we can also make generative models that are affected by the output of previous frames.

Pix2pix requires a paired input-target data set. Therefore, it is difficult to use in applications that lack paired

The associate editor coordinating the review of this manuscript and approving it for publication was Mostafa Rahimi Azghadi.

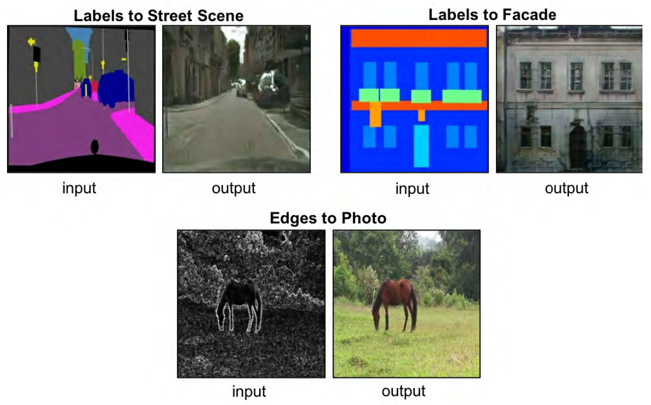

FIGURE 1. Image-to-image transformation tasks. Many tasks in computer vision and image processing can be regarded as image-to-image transformations, such as Sobel filters, sharpness, and denoising. If we use such image-to-image transformation to transform image sequences, the quality of the output sequence is reduced by flicker because the image-to-image transformation does not account for the relationship between frames. Therefore, we proposed a model that can reduce flicker in image sequence transformation between paired image sequences.

input-target data sets. For example, if we want to use pix2pix to convert horses into zebras, we need pictures of horses and zebras in the same pose, same size, and same background. Preparing such a data set would be very difficult. To overcome that drawback, Cycle GAN [14] and Disco GAN [17] were developed. Star GAN [18] was then developed to reduce the number of generative networks and discriminative networks in CycleGAN. CycleGAN is for cases in which the shape of the input image and the target image are similar, such as a horse-to-zebra conversion. DiscoGAN is for different types of input and target images, such as a chair-to-bag conversion. A super-resolution GAN (SRGAN [13]) was developed to transform a low-quality image into a high-quality image. Various other image-to-image transformation applications using GANs and cGANs have also been developed.

Any of those image transformation models could be extended to the transformation of a sequence of images, that is, video. However, using them in that way causes severe flicker because a network of GANs treats each image in a sequence as new data. Because GANs do not use that data when learning, they generate noise during transformation.

The problem is that this noise appears arbitrarily around each output image even if the input changes only slightly. As a result, applying a GAN-based image-to-image transformation to image sequences produces flicker, even when generating video directly using a GAN [20].

The flicker is not greatly reduced by allowing the previous input frame to affect the current output frame, not even by using the previous input frame as the conditional parameter, because the previous input frame is irrelevant to the noise generated in the current frame. However, we can effectively use the output of the previous frame to reduce the noise when generating the current frame.

In this paper, we present a novel model to drastically reduce flicker by using the output of the previous frame. This model can be applied to both image sequence transformation and image-to-image transformation. The contributions of this paper are as follows.

1) We propose a method to add and modify layers in a generative network to reduce flicker.
2) We propose a loss function and a learning method for training the modified generative network.
3) When there is a reference image that is flicker-free, we propose and use metrics to measure flicker. We show that the proposed model has little effect on the image quality and greatly reduces the flicker in an image sequence transformation.

The rest of the paper is organized as follows. First, we briefly introduce related work in Section II. Section III explains the proposed model, and Section IV describes the data set used to train the proposed model. We experiment with the proposed model in Section V and discuss the results of experiment in Section VI. We present the overall results of this paper in Section VII.

II. RELATED WORK

This section briefly introduces generative adversarial networks (GANs), conditional GANs (cGANs), and pix2pix, an image-to-image transformation application using cGANs. We also introduce the flicker phenomenon and propose metrics to measure flicker.

A. GENERATIVE ADVERSARIAL NETWORKS

GANs are a technology that has been attracting attention for the past four years and has been applied to many areas. In GANs, a generative network learns the distribution of data, and a discriminative network determines whether a corresponding sample bas been generated from a generative network or a target sample. The two networks have opposing goals for one value function, and they are learned simultaneously for their own purposes.

The goal of the generative network is to learn the distribution of data. First, we define input noise variables $P_z(z)$. Let P_{data} be the distribution of the data to be learned, and let x be the data variable. Let P_g be the generated distribution that is learned by the generative network. Next, let $G(z)$ be the mapping from noise z to the data space. Let G be a function of a generative network and D be a function of a discriminative network with a scalar output value. $D(x)$ is the probability that x comes from the data rather than the generative network. We train these two networks simultaneously to satisfy Eq. (1).

$$\min_G \max_D V(G, D) = \mathbb{E}_{x\sim P_{data}(x)}[\log D(x)] + \mathbb{E}_{z\sim P_z(z)}[\log 1 - D(G(z))]. \quad (1)$$

Eq. (1) maximizes the value function V for D, and causes G to minimize $[\log 1 - D(G(z))]$, the second term of V. In the discriminative network, the parameter is updated to ensure that $D(G(z))$ is 0 and $D(x)$ is 1. That is, D is learned to distinguish whether a sample is from x or P_g, and G is learned so that D cannot distinguish whether the sample comes from

x or P_g. This model is thus called generative adversarial networks because D and G learn in ways hostile to the same loss function and thereby determine the distribution of data.

A cGANs adds a condition y to a conventional GAN, and its objective is Eq. (2).

$$\min_G \max_D V(G, D) = \mathbb{E}_{x \sim P_{data}(x)}[\log D(x|y)] + \mathbb{E}_{z \sim P_z(z)}[\log 1 - D(G(z|y))]. \quad (2)$$

In other words, the GANs update the parameters of the generative network so that P_g is approximately equal to the probability density of the data, and cGANs update the parameters of the generative network so that the conditional probability density and P_g become approximately equal. Here, both GANs and cGANs learn a distribution by deceiving the discriminative network. This means that GANs approximate P_g to P_{data} by maximizing the likelihood of P_g and P_{data}. However, not all the samples can be included in the data space. Therefore, the system maximizes the likelihood of P_g and P_{data} using only some samples(i.e., $KL(P_g|P_{data})$ cannot be 0. Any time an input datum is included in the data space but not used for learning, random noise is generated in the output, which causes flicker in the image sequence. Of course, long-term learning can reduce the value of $KL(P_g|P_{data})$, but that runs the risk of overfitting to the samples used in learning.

B. PAIRED IMAGE TRANSFORMATION WITH cGANs

Pix2pix [12], which offers image-to-image transformation with cGANs, uses an encoder-decoder network and the U-Net architecture for the generative network. U-Net is an encoder-decoder network with skip connections added from the encoder stack to the decoder stack. Pix2pix uses a patch-GAN, also called a Markovian discriminative network, as the discriminative network to increase the influence of the input features missed by the encoder. In other words, patchGAN can distinguish whether a sample image is fake or not by using patch-sized parts of the sample image. Maximizing the patch size will cause the discriminative network to use the entire sample image and is called imageGAN. Minimizing the patch size allows the discriminative network to use only the pixels included in the sample image and is called pixelGAN. The larger the patch size, the better the quality of the result, but only to a certain patch size. Because the optimal patch size depends on parameters such as the size of the input image and the object size in the input image, imageGAN can be used for its convenient implementation. For an image-to-image transformation from X to Y, let $P_{data}(x)$ be the distribution of x, $P_{data}(y)$ be the distribution of y, and $P_z(z)$ be the prior probability distribution of noise. The loss function of the cGAN is shown in Eq. (3).

$$\mathcal{L}_{cGAN}(G, D) = \mathbb{E}_{x \sim P_{data}(x), y \sim P_{data}(y)}[\log D(x, y)] + \mathbb{E}_{x \sim P_{data}(x), z \sim P_z(z)}[\log 1 - D(x, G(x, z))]. \quad (3)$$

Unlike in GANs, x is additionally input to D and G. Thus, cGANs give x as a condition to generate and discriminate y. That is, G learns the mapping from X to Y with a conditional discriminative network. Moreover, a previous study [29] showed that it is more beneficial to use traditional losses, such as L1 and L2 distances, together with the cGAN loss and that L1 distances are less blurring than L2 distances. Thus, the L1 distance loss (as in Eq. (4)) is used with the cGAN loss function.

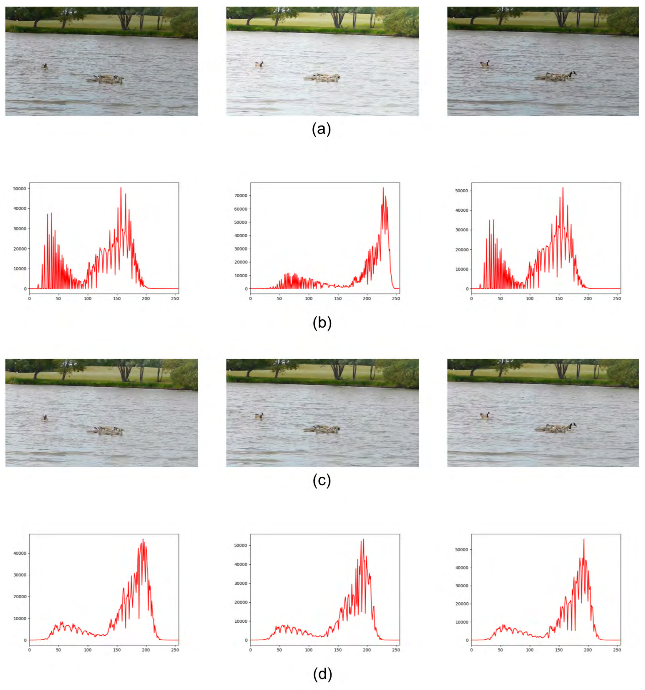

FIGURE 2. (a) An image sequence with global flicker. (b) Corresponding luminance histograms. (c) The sequence after global flicker correction using midway equalization [28]. (d) Luminance histograms of the result of midway equalization.

$$\mathcal{L}_{L1}(G) = \mathbb{E}_{x \sim P_{data}(x), y \sim P_{data}(y), z \sim P_z(z)}[\|y - G(x, z)\|_1]. \quad (4)$$

Therefore, the final objective function of transformation applications from X to Y using cGANs is shown in Eq. (5)

$$G^* = \arg \min_G \max_D \mathcal{L}_{cGAN}(G, D) + \lambda \mathcal{L}_{L1}(G) \quad (5)$$

λ is the weight value of L1 loss.

C. FLICKER

Generally, flicker occurs when the light exposure is not constant during image recording. That is, it occurs when there is a flicker in the lighting where the picture is taken, and a flicker occurs when the picture capture period does not match the period of the lighting flicker cycle. In this case, flicker usually happens globally, so it can be solved using a simple postprocessing method such as histogram equalization. For example, as shown in Fig. 2, the flicker caused by a global intensity change can be greatly reduced through simple histogram equalization and histogram transformation. Fig. 2 (a) and (b) show an image sequence with global flicker and the luminance histogram of each frame of the corresponding image sequence, respectively. Fig. 2 (c) is an image sequence

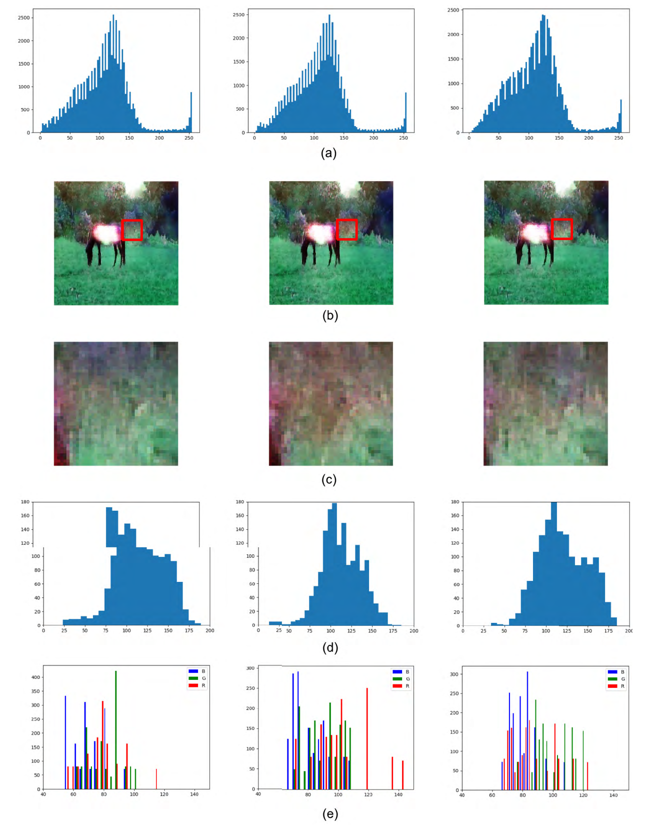

FIGURE 3. Local flicker generated by image-to-image transformation with cGANs. (a) Global luminance histogram of each frames. (b) Location of region of interest (ROI). (c) Magnified images of ROI. (d) Luminance histogram of ROI. (e) RGB histograms of ROI.

with the global flicker removed by means of midway image equalization [28]. Fig. 2 (d) shows the luminance histogram of Fig. 2 (c).

However, it is difficult to reduce local flicker using those methods. That is, it is difficult to reduce the flicker using conventional methods when the illuminant is scattered around the image when the image is taken, or when local flicker occurs due to the changes in the position of the sun or clouds in a time-lapse. Therefore, the flicker should be reduced by considering the motion of the object. The flicker in a GAN occurs locally, as shown in Fig. 3 (b), (c), and (d), and it has little relation to the motion of an object. Also, the shape and position of the flicker are random, and global luminance is almost the same, as shown in Fig. 3 (a) and (b). In addition, the flicker that occurs in GANs differs from typical local flicker. As shown in Fig. 3 (e), the flicker of GANs differs not only from luminance; the color itself is completely different, making it is difficult to reduce using the conventional flicker reduction method.

D. METRICS FOR FLICKER

In general, local flickers are difficult to measure in image sequences because it is difficult to tell whether the change in luminance is caused by object movement or local flicker, as shown in Fig. 4. In the case of Fig. 4 (a), the change

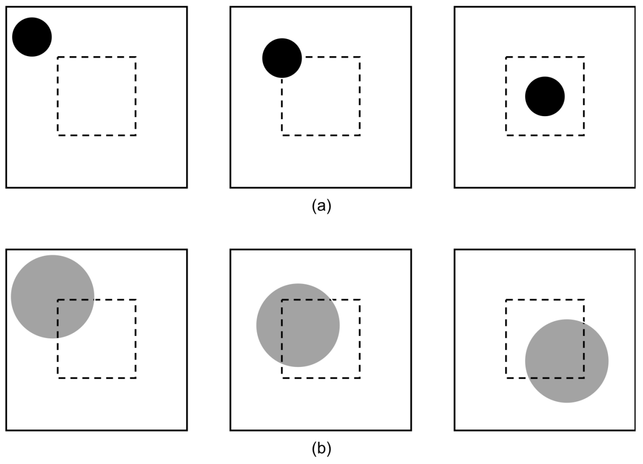

FIGURE 4. Because local flicker is difficult to distinguish from object movement, it is difficult to measure without a flicker-free sequence. (a) The luminance change in the dashed-line box is caused by the motion of the object. (b) The luminance change in the dashed-line box is caused by the local flicker.

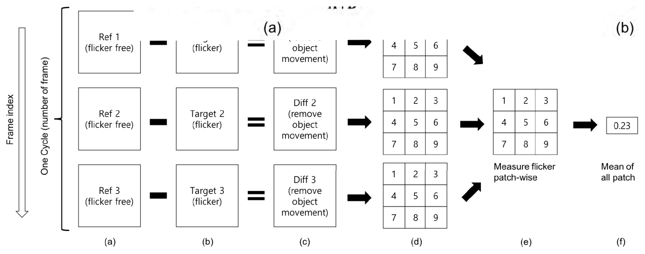

FIGURE 5. Method to measure local flicker. (a) Flicker-free sequence (reference sequence). (b) Target sequence that has local flicker. (c) Sequence subtracted from flicker-free sequence and target sequence to illuminate object movement. (d) Grid-partitioning as patch size. (e) Measure of flicker patch-wise over one cycle time. (f) Evaluation of final flicker as the mean of all grids (all patches).

in luminance in the dashed-line rectangle is caused by the movement of the object, and in the case of Fig. 4 (b), it is caused by local flicker. But conventional flicker metrics [19], [27] cannot tell the difference between the two. Because of this, some studies [21], [27] use luminance or mean intensity over time to show flicker reduction. However, most papers [22], [23], [27] indicate that flicker reduction is achieved by showing the resulting image sequence without quantifying the metric.

However, if we can eliminate the luminance change caused by motion, we can show flicker reduction numerically. We propose a metric that measures local flicker, though it requires a reference image sequence that is flicker free. As shown in Fig. 5, one cycle, which in this paper is 6 frames, is the duration for measuring the flicker. First, we consider a reference image sequence that is flicker free, as shown in Fig. 5 (a), and a target image sequence that has flicker, as shown in Fig. 5 (b). To remove the motion of the object from the target image sequence, we created a new image sequence (Fig. 5 (c)) by subtracting the target image sequence from the reference image sequence. In the image sequence with the object motion removed, we divide each frame into

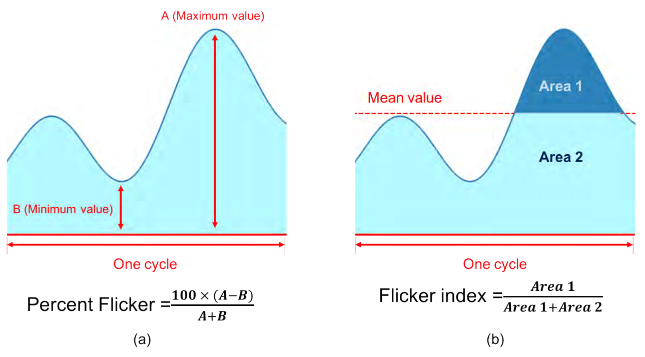
A (Maximum value)
B (Minimum value)
One cycle
Percent Flicker = $\frac{100 \times (A-B)}{A+B}$
(a)
Mean value
Area 1
Area 2
One cycle
Flicker index = $\frac{Area\ 1}{Area\ 1+Area\ 2}$
(b)

FIGURE 6. **Flicker metrics: (a) percent flicker, (b) flicker index.**

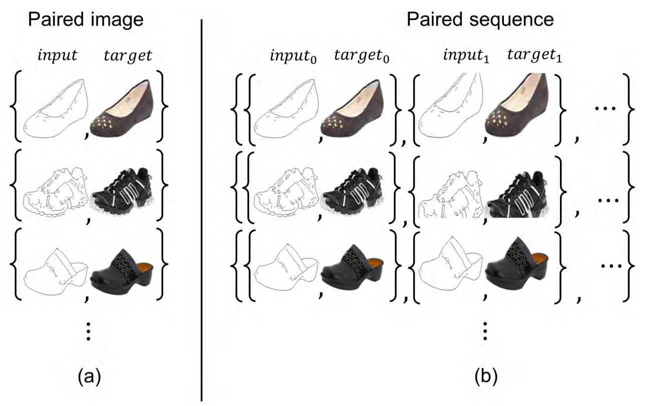

FIGURE 7. **(a) Paired image set. (b) Paired image sequence set.**

spatial patches as shown in Fig. 5 (d) and take the mean value for each patch. Then, we measure flicker using a conventional flicker measurement metric (such as percent flicker) patch-wise in the sequence as shown in Fig. 5 (e). As the patch size increases, the absolute value of the measured flicker decreases. Therefore, we must compare the measured values using the same patch sizes. If the patch size is 1×1, it is calculated pixel-wise, and has the largest value of the measured flicker. We take the mean of each flicker measurement calculated patch-wise as the final flicker value (Fig. 5 (f)). The flicker metrics used are as follows.

1) PERCENT FLICKER [19], [27]

Percent flicker is a metric that computes the flicker using the maximum and minimum values of luminance for a given cycle, as shown in Fig. 6 (a). We divide the difference between the maximum and minimum values by the sum of the maximum and minimum values, then multiply by 100. Less flicker is reflected as a lower percent flicker.

2) FLICKER INDEX [19], [27]

The flicker index uses area to measure flicker. First, we measure the average luminance for a given cycle. Then, we measure area 2, which is smaller than the average, and area 1, which is larger than the average, as shown in Fig. 6(b). Finally, we divide area 1 by the sum of area 1 and area 2. That is, the area with a value larger than the average is divided by the total area. Smaller values indicate less flicker. Unlike the percent flicker, the flicker index is not sensitive to the luminance that changes dramatically over a very short duration because it uses an area to measure flicker. Thus, the flicker index reduces the effect of temporary and dramatic brightness changes, which are difficult for human beings to perceive, on measured values.

E. METRICS FOR OUTPUT QUALITY

Even if an image-transformation method shrinks flicker, it does not represent an improvement if the output quality decreases. In general, images generated using GANs do not have reference images. Therefore, mean opinion scores (MOS) and inception scores are typically used to measure quality. In the case of MOS, most people use Amazon Mechanical Turk (AMT) to measure the quality of a generated image by selecting an arbitrary number of evaluation groups. However, those results are very subjective and difficult to compare in absolute numbers.

The inception score [24], [25] evaluates the image quality from GANs using Google's inception image classification model. For example, two images created from two different models trained to produce the image of a horse are inserted into a pre-trained inception model. The inception model estimates how close the image is to a horse as a value of probability, with a higher probability held to indicate higher GAN performance.

The inception score method is used quite often because it has the advantage of consistency and a certain degree of objectivity. However, there is no reason to use Google's inception model for evaluation, and models could be optimized to produce a high score from a pre-trained inception model regardless of the actual quality of the image. Also, classes that are not in the famous data set, such as CIFAR10 or the IMAGE-NET dataset, need to be added to (trained in) the classification model directly by gathering datasets from those classes, destoying objectivity.

If a reference image exists, full reference (FR) metrics such as VIF and SSIM are most objective way to measure quality. Fortunately, we can measure the quality using the FR metrics because the dataset we used in our experiments has reference images.

III. PROPOSED RECURSIVE CONDITIONAL GAN

In this section, we propose a cGAN model to reduce flicker in an image sequence. We use paired sequence sets as training datasets. As shown in Fig. 7, a paired image set consists of one input-target image pair per sample. A paired image sequence set, on the other hand, consists of a paired sequence, which consists of several input-target image pairs per sample. Each sample sequence has an order for its image pairs. Therefore, when learning a model using a paired image sequence set, the model should be trained in the correct sequence order.

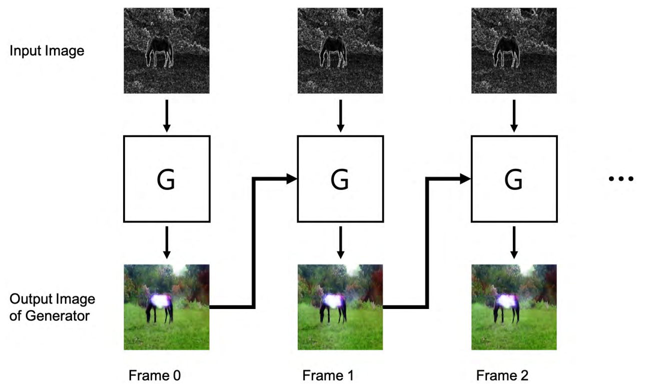

FIGURE 8. Training step for generator in one sample sequence.

In paired-image transformation, cGANs provide a generative model to learn the mapping from an observed image x and random noise z to produce an output y. In image-sequence transformation, the output of the previous frame is added to the generative model as the conditional parameter. Thus, in image-sequence transformation, cGANs form a generative model that learns the mapping from $G(x_{i-1})$, x_i and random noise z to produce the output y_i. $G(x_{i-1})$ is the output of the previous frame, x_i is the current input frame, and y_i is the current target frame.

A. OBJECTIVE

We must set the loss so that the output of the previous frame affects the output of the current frame (Fig. 8). Let us assume that we want to make a transformation from X (input data) to Y (target data). Let $P_{data}(x)$ be the data distribution of x, and $P_{data}(y)$ be the data distribution of y. The objective of the cGANs for the image sequence transformation is Eq. (6). We omit random noise z, which is input into the generator, in Eq. (6), (7), (8), and (9) for simplicity of the expression. Therefore, the true expression of $G(x_i, G(x_{i-1}))$ is $G(x_i, G(x_{i-1}), z)$.

$$\mathcal{L}_{cGAN}(G, D) = \begin{cases} \mathbb{E}_{x_i,y_i}[\log D(x_i, y_i)] \\ \quad + \mathbb{E}_{x_i}[\log 1 - D(x_i, G(x_i))] & \text{if } i = 0. \\ \mathbb{E}_{x_i,y_i}[\log D(x_i, y_i)] \\ \quad + \mathbb{E}_{x_i}[\log 1 - D(x_i, G(x_i, G(x_{i-1})))] & \text{o/w.} \end{cases} \tag{6}$$

In x_i and y_i, the subscript i means the index of the frame in one sample sequence, and the index starts from zero. Compared with Eq. (3), which is a conditional GANs loss in Pix2pix, $G(x_{i-1})$, which is the cGANs output of the previous frame, is added to the generative network as an input. The role of the discriminative network does not change. The output of the previous frame is not entered as a conditional parameter in the discriminative network because the output of the previous frame does not help to determine whether the output of the current frame is a real image or a fake image. In the first frame of the sample sequence, $i = 0$, there is no previous frame, so the output of the previous frame is not input to the generative network. Otherwise, the output of the previous frame is input into the generative network.

Previous studies have found that it is beneficial to use GAN loss and L1 loss together, so L1 loss is also used as expressed in Eq. (7). As in Eq. (6), when the frame index is 0, the output of the previous frame is not input into the generative network. The output of the previous frame is input into the generative network in the remaining cases.

$$\mathcal{L}_{L1}(G) = \begin{cases} \mathbb{E}_{x_i,y_i}[\|y_i - G(x_i)\|_1] & \text{if } i = 0. \\ \mathbb{E}_{x_i,x_{i-1},y_i}[\|y_i - G(x_i, G(x_{i-1}))\|_1] & \text{o/w.} \end{cases} \tag{7}$$

Finally, we add a new loss (Eq. (8)) to make the output of the previous frame affect the output of the current frame. This loss is the L1 distance between the output of the current frame and the output of the previous frame. If the frame index is 0, there is no previous frame, so this loss becomes 0.

$$\mathcal{L}_{bwt}(G) = \begin{cases} 0 & \text{if } i = 0. \\ \mathbb{E}_{x_i,x_{i-1}}[\|G(x_i, G(x_{i-1})) - G(x_{i-1})\|_1] & \text{o/w.} \end{cases} \tag{8}$$

Therefore, our final objective is Eq. (9).

$$G^* = \arg\min_G \max_D \mathcal{L}_{cGAN}(G, D) + \lambda_1 \mathcal{L}_{L1}(G) + \lambda_2 \mathcal{L}_{bwt}(G). \tag{9}$$

The generative network learns the mapping from X to Y using Eq. (6), and it learns the low-frequency domain of the data space using Eq. (7). The generative network learns the relation between the output of the previous frame and the output of the current frame. In Eq. (9), λ_1 and λ_2 are the weight of each loss.

$$\mathcal{G}_{bwt}(G, j) = \begin{cases} 0 \quad \text{if } i - j < 0. \\ \mathbb{E}_{x_i,x_{i-j}}[\|G(x_i, G(x_{i-j})) - G(x_{i-j})\|_1] & \text{o/w.} \end{cases} \tag{10}$$

$$\mathcal{L}_{bwt}(G) = \frac{1}{N}\sum_{j=1}^{N} \mathcal{G}_{bwt}(G, j) \tag{11}$$

The objective function described so far uses only one previous output frame to reduce flicker. However, we can use multiple previous output frames. When multiple output frames are used as input to the network, only $\mathcal{L}_{bwt}$ is changed. Let the index of the current frame be i. $\mathcal{G}_{bwt}(G, j)$, the L1 norm between the current output frame and the jth previous output frame, is Eq. (10). Because the index of the frame starts from 0, if $i - j$ is smaller than 0, the value of $\mathcal{G}_{bwt}(G, j)$ becomes 0. If N previous output frames are used as conditional parameters, $\mathcal{L}_{bwt}(G)$ becomes Eq. (11).

B. NETWORK ARCHITECTURES

1) GENERATIVE NETWORK G

Because the generative network of the existing GAN focuses on the image, information about the output of the previous

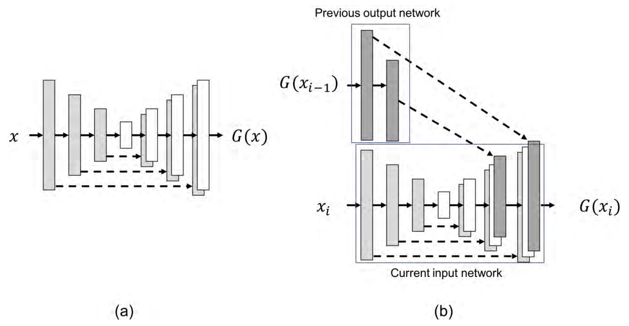

FIGURE 9. Architecture for a generative network. (a) U-Net. (b) Architecture for the generative network of the proposed model.

frame is not input into the generative network. Therefore, we need to modify the generative network to be able to generate the current output that is influenced by the output of the previous frame. This modification policy is to creates a shallow network that can enter information from the previous frame and connect to the output side of the existing network, as shown in Fig. 9 (b).

For example, if the generative network is U-Net [34], as shown in Fig. 9 (a), we create the previous output network, which uses the output of the previous frame as input (as in the upper part of Fig. 9 (b)). Then, we concatenate that network to the decoder part of the current input network, which is the lower part of Fig. 9(b). In Fig. 9, a dashed line indicates a skipped connection. As a result, the previous output network is the network in which $G(x_i)$ is influenced by $G(x_{i-1})$, and the current input network is the existing cGAN-based image-to-image transformation structure.

In this paper, we used a generative network with a previous output network added to U-Net [34], as shown in Fig. 9 (b). We used leaky ReLU, ReLU, convolution, transpose convolution, and batch normalization in a generative network. Table 1 shows the structure of the generative network used in this paper when only one frame is used as a conditional parameter. In Table 1, DeConv indicates a transpose convolution.

When multiple previous frames are used as a conditional parameter, we increase the number of previous output networks, as shown in Fig. 10. Then, we concatenate the previous output networks to the decoder part of the current input network. Fig. 10 (a) shows a network architecture when 2 previous frames are used as input for the generative network, and Fig. 10 (b) shows a network architecture when 3 previous frames are used. In Table 1, if 2 previous frames are used as input for the generative network, the generative network has 2 previous output networks. The generative network has 3 previous output networks if 3 previous output frames are used as input for the network. Layer 1 of each previous output network concatenates to Layer 15, and Layer 2 of each previous output network concatenates to Layer 16 in Table 1.

2) DISCRIMINATIVE NETWORK D

The discriminative network we use does not differ from the discriminative network in pix2pix. In other words, it is a discriminative network that uses the CNN structure, which is mainly used in classification applications. Convolution, leaky ReLU, and batch normalization are used for the discriminative network. The channel size of the output layer is 1 because

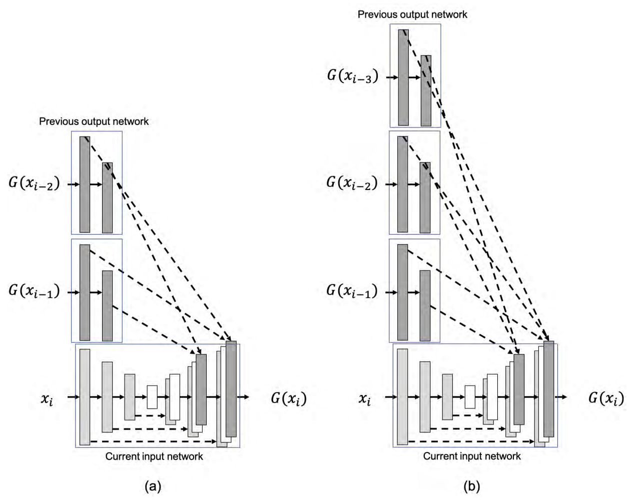

FIGURE 10. Network architecture for the generative network when the multiple previous frames are input to the network. (a) A generative network structure in which two previous frames are used as conditional parameters. (b) A generative network structure in which three previous frames are used as conditional parameters.

TABLE 1. Network architectures of the generative network.

Generative network			
Current input network		previous output network	
Input: current input frame		Input: output of previous frame	
[Layer 1]	Conv.(4,4,64), stride=2; LReLU;	[Layer 1 prev.]	Conv. (4,4,64), stride=2; LReLU;
[Layer 2]	Conv.(4,4,128), stride=2; Batch norm; LReLU;	[Layer 2 prev.]	Conv.(4,4,128),stride=2; Batch norm; LReLU;
[Layer 3]	Conv.(4,4,256), stride=2; Batch norm; LReLU;		
[Layer 4]	Conv.(4,4,512), stride=2; Batch norm; LReLU;		
[Layer 5]	Conv.(4,4,512), stride=2; Batch norm; LReLU;		
[Layer 6]	Conv.(4,4,512), stride=2; Batch norm; LReLU;		
[Layer 7]	Conv.(4,4,512), stride=2; Batch norm; LReLU;		
[Layer 8]	Conv.(4,4,512), stride=2; Batch norm; LReLU;		
[Layer 9]	DeConv.(4,4,512),stride=2; Batch norm;		
	Concatenate Layer (Layer 9, Layer 7); ReLU		
[Layer 10]	DeConv.(4,4,512),stride=2; Batch norm;		
	Concatenate Layer (Layer 10, Layer 6); ReLU		
[Layer 11]	DeConv.(4,4,512),stride=2; Batch norm;		
	Concatenate Layer (Layer 11, Layer 5); ReLU		
[Layer 12]	DeConv.(4,4,512),stride=2; Batch norm;		
	Concatenate Layer (Layer 12, Layer 4); ReLU		
[Layer 13]	DeConv.(4,4,256),stride=2; Batch norm;		
	Concatenate Layer (Layer 13, Layer 3); ReLU		
[Layer 14]	DeConv.(4,4,128),stride=2; Batch norm;		
	Concatenate Layer (Layer 14, Layer 2, Layer 2 prev.); ReLU		
[Layer 15]	DeConv.(4,4,64),stride=2; Batch norm;		
	Concatenate Layer (Layer 15, Layer 1, Layer 1 prev.); ReLU		
[Layer 16]	DeConv.(4,4,3),stride=2; Batch norm; Tanh		
Output: transformed image of current frame			

TABLE 2. Network architectures of the discriminative network.

Discriminative network		
Input 1: input image		Input 2: target image or output image of generative network
Input: current input frame		Input: output of previous frame
Input layer	Concatenate inputs (input1, input2)	
[Layer 1]	Conv. (4, 4, 64), stride=2; LReLU;	
[Layer 2]	Conv. (4, 4, 64), stride=2; LReLU;	
[Layer 3]	Conv. (4, 4, 64), stride=2; LReLU;	
[Layer 4]	Conv. (4, 4, 64), stride=2; LReLU;	
[Layer 5]	Conv. (4, 4, 64), stride=2; LReLU;	
[Layer 6]	Conv. (4, 4, 64), stride=2; LReLU;	
Output: transformed image of current frame		

it determines whether the input is real or fake. Also, because the output value is binary(real or fake), we use a sigmoid instead of softmax for the loss calculation. Table 2 shows the structure of the discriminative network used in this paper.

C. TRAINING

The training method is as follows. When learning the first frame of a sample sequence, there is no output from a previous frame. That is, in the case of the first frame of the sequence shown in Fig. 8, only the current input image is input into the generative network. Therefore, in the first frame, the input is a zero tensor in $G(x_{i-1})$ of Fig. 9 (b). After the first frame, there is an output of a previous frame, so the output from the previous frame is input together with the current input image. When multiple previous output frames are used as conditional parameters and some previous output frames do not exist, enter 0 tensor instead of the previous output frames. Because the loss function and structure of the discriminative network are not changed, the discriminative network is trained in the same way as in existing cGAN-based image-to-image transformation applications.

IV. DATA SET

In this paper, we will use a paired image sequence set to train the proposed model.

A. PAIRED IMAGE SEQUENCE SET FROM VIDEO CLIPS

We extracted the consecutive frames from many videos and made target sequences. A large number of video clips is required for generalized learning of the desired application. Therefore, for this paper, we obtained many horse video clips from the Pexels video site [35] and made many sequences. This method is useful for applications (such as colorization and super-resolution) for which it is relatively easy to obtain video clips for training. The edge sequences (input sequences) were obtained by using the Sobel filter on the target sequences. We made the paired image sequence set shown in Fig. 7 by combining the input and output sequences, and we used it to train the proposed model.

B. PAIRED IMAGE SEQUENCE SET FROM AN IMAGE

It would be nice to be able to get a lot of video clips as explained in Section IV-A, but actually obtaining video clips much harder than collecting images. For example, if we want to transform a real video into a Monet-like style, we need to get a Monet video clip, which is almost impossible. We tried to do a paired image sequence transformation, but collecting paired video clips is much more difficult than collecting paired images.

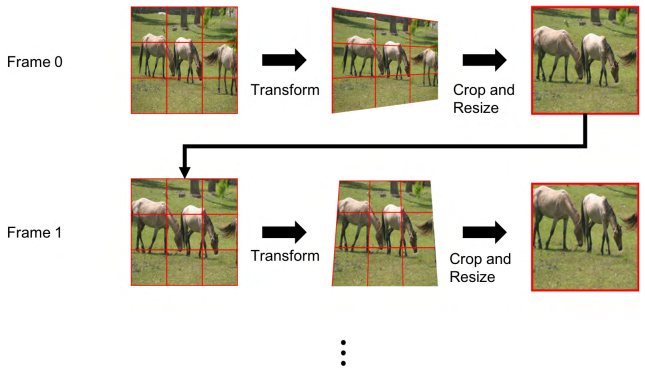

FIGURE 11. Method for making a sequence from a single image.

Therefore, we made a new frame by performing image transformations such as warping and affine transforms, as shown in Fig. 11. We repeatedly applied the above method to create the desired number of frames. Unlike the actual image sequences, we cannot obtain new information from this sequence, but we can obtain the relationship between adjacent frames in the sequence. Therefore, we can train cGANs to reduce flicker using a paired image sequence set made using this method. We also used the cityscapes dataset [32] and CMP facades [33] in this way to great paired image sequence sets for our experiments.

V. EXPERIMENTS

A. EXPERIMENTAL SET UP

To ensure fair comparisons, we applied the same settings to pix2pix. The length of the image sequence was unified to 5, and all experiments were done using Nvidia GTX 1080 GPUs and Tensorflow [36]. We used Adam Optimizer [37] as a solver and taught it with a learning rate of 0.0002 and a first momentum of 0.5. The batch size was set to 25.

We used a dataset created from a video clip that we collected directly from Pexels videos [35]. The collected data were extracted with a Sobel filter to create a paired image sequence set. In addition, we used an existing image dataset by transforming it into a sequence using the method described in Section IV. The image sequence transformation tasks and datasets used in the experiments are as follows.

- Semantic labels ⇒ photos from the cityscapes [32] and facades datasets [33].
- Edge ⇒ photo from the horse dataset created by us; the original video clip was from Pexels videos [35], and a Sobel filter was used to extract the edges.

Because the size of each dataset was very different, the number of training iterations for each dataset also differed. However, the training epochs for different tasks

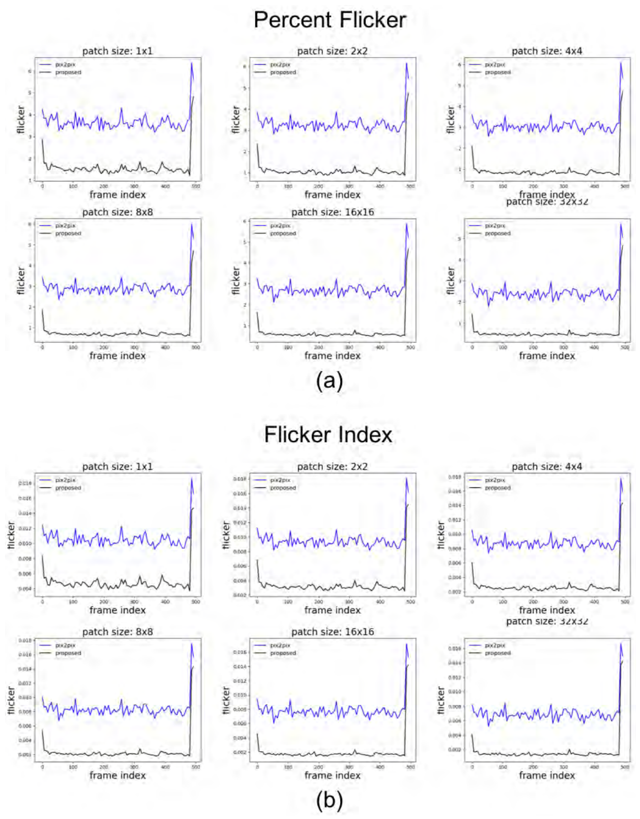

FIGURE 12. Flicker measurement was performed by changing the patch size and using the flicker measurement method proposed in Section II-D: (a) percent flicker and (b) flicker index.

and datasets were all 80. Therefore, the number of training iterations was larger than 80K.

B. EVALUATION METRICS

We evaluated metrics for de-flicker and image quality. Fig. 12 shows the flicker of pix2pix and the proposed model. We measured the flicker using the method proposed in Section II-D. The percent flicker (PF) and the flicker index (FI) were used as the flicker metrics and measured using different patch size. The absolute value of flicker decreased as the patch size increased, but the relative tendency was almost unchanged. Therefore, we evaluated the flicker using the proposed flicker method with a one cycle duration of 250 ms (6 frames in 24 fps) and a patch size of 1x1. We also presented changes in luminance in certain areas and changes in mean luminance to promote an intuitive understanding of flicker.

We used Visual Information Fidelity (VIF [31]) and the Structural Similarity Metric (SSIM [30]) as quality metrics.

VI. DISCUSSION

In this section, we will evaluate the proposed model against the existing paired image transformation application pix2pix. We evaluated the output sequence of both models. We also compared two popular de-flicker algorithms against the proposed model by applying them to the output sequence

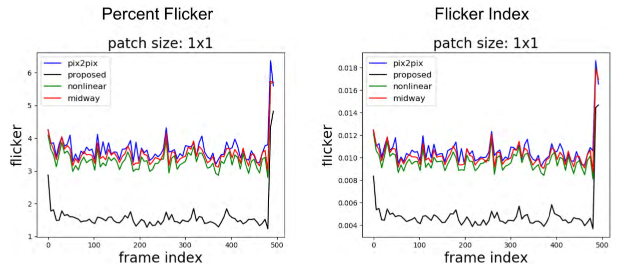

FIGURE 13. Evaluated flicker of Sobel edge-to-photo task with horse dataset: (a) percent flicker, (b) flicker index.

from pix2pix. The output sequence from the proposed model had less flicker than the output sequence of pix2pix, which was not significantly reduced by the existing flicker reduction algorithms.

A. SEQUENCE SET

As explained in Section IV-B, there is a large difference between the amount of information that can be obtained from a video clip and the amount of information that can be obtained from a sequence created from a single image. Therefore, we analyzed these two cases separately. First, we analyzed the results from the horse dataset in which each method performed the edge-to-photo conversion task using video clips. We set the λ_1 in Eq. (9) to 50 and λ_2 to 30 when training the proposed model. The λ in Eq. (5) was 80 when training pix2pix. The test sequence used for evaluation was 500 frames, 24 fps. The test sequence consisted of a paired input-target sequence.

Figure 13 shows that the proposed model produced fewer flickers than pix2pix by both the PF and FI metrics. We also measured the flicker of the output sequence using nonlinear flicker compensation [26] and midway equalization [28]. The algorithm in [26] is presented as ‘nonlinear’, and it estimates motion and reduces flicker. The midway algorithm is another de-flicker algorithm. We applied these two algorithms to the output sequence from pix2pix and measured the flicker. With the midway equalization, the flicker was reduced compared to pix2pix alone, and the nonlinear algorithm showed less flicker than the midway equalization. However, both those flicker reduction effects were less effective than the proposed method.

To gain a better understanding of the effect of flicker reduction, we spatially divided the image sequence into 4×4 patches, as shown in Fig. 14 (a), and then we looked at the luminance changes in the upper left area of the output sequence. Pix2pix, midway equalization, and the nonlinear flicker compensation show that the change in luminance is very rapid. On the other hand, the luminance change in the proposed method is relatively gentle. This tendency is observable in all regions of the image sequence divided into 4×4 patches. Figure 14 (b) shows the mean value of the luminance change. Because the motion of the object in the test sequence is not large and the fps of the image is 24, it can be seen that the change in luminance is greatly influenced by the flicker.

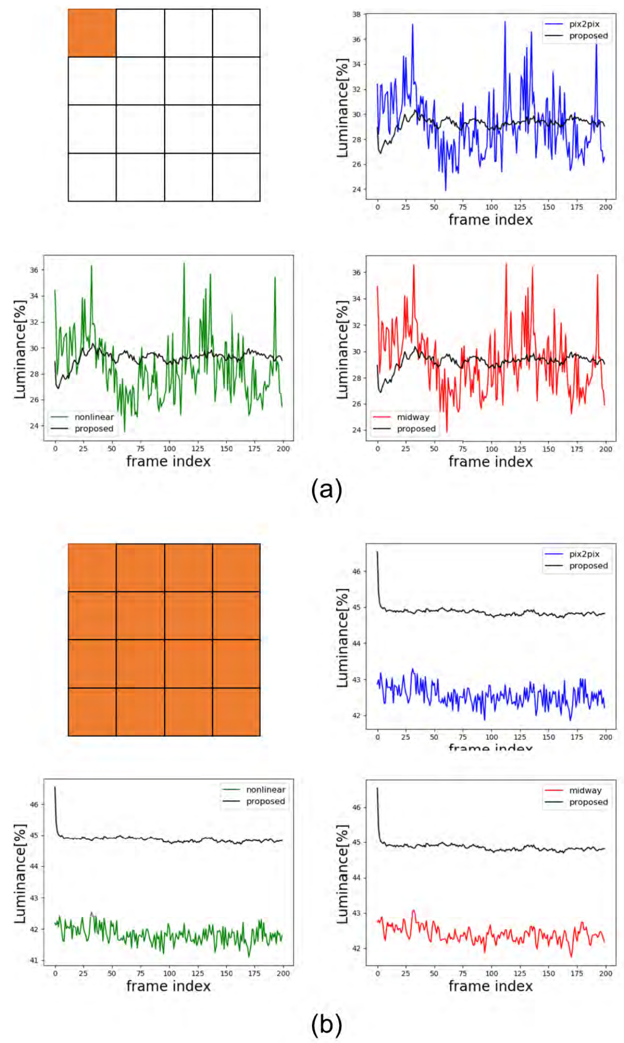

FIGURE 14. (a) Change in luminance in the top left corner of the 4 × 4 grid. (b) Mean change in luminance.

Table 3 shows the average VIF and SSIM for 500 frames. The proposed model has a lower SSIM but higher VIF than pix2pix. A higher VIF indicates that the quality of the image is better from a person's perspective. The SSIM decreased slightly for several reasons, one of which is that the edge is slightly blurred by Eq. (8). However, the difference in SSIM is almost zero. In other words, the image quality from the proposed model showed a small improvement compared to pix2pix.

Using midway equalization on pix2pix produced little difference in the quality of the image or flicker. Because the midway equalization is more effective in reducing global flicker than local flicker, applying it to this pix2pix output sequence (which has lots of local flickers) produces little effect on either flicker reduction or image quality. The nonlinear flicker compensation algorithm showed a slight numerical improvement in VIF and SSIM performance compared to pix2pix, but the improvement was too small to be visible to users.

TABLE 3. Image quality of each model and method.

	VIF	SSIM
pix2pix	$6.90e^{-2}$	$2.95e^{-1}$
proposed model	$\mathbf{7.18e^{-2}}$	$2.92e^{-1}$
midway	$6.89e^{-2}$	$2.95e^{-1}$
nonlinear	$6.90e^{-2}$	$\mathbf{2.97e^{-1}}$

Figure 15 shows the luminance difference in each output sequence. In the target sequence, almost no change in luminance is visible between adjacent frames. In contrast, the pix2pix output sequence has a partial change in luminance between frames, and its shape and position also differ. The output sequences from the nonlinear flicker compensation algorithm and midway equalization do not differ much from the pix2pix output sequence. That is, the postprocessing did not substantially reduce the flicker caused by GAN noise. On the other hand, the output sequence of the proposed model shows relatively little change in luminance.

In addition, we confirmed that the body of the horse is saturated to white in the pix2pix output sequence. The proposed method reduced that saturation through its correlation with the output of the previous frame. However, the output sequence from the proposed model has some blurring caused by using the output of the proposed model does not differ significantly from the existing model when training using a video clip, but the flicker is greatly reduced.

B. IMAGE SET

We tested the semantic label-to-photo task using the facades and cityscapes datasets. In the proposed model, the weight parameters $\lambda_1 = 50$, $\lambda_2 = 30$ were used for the facades dataset, and $\lambda_1 = 50$, $\lambda_2 = 50$ were used for the cityscapes dataset. In the pix2pix model, the weight parameter λ in Eq. (5) was fixed at 80 for both tasks. The facades dataset was tested using 4 test sequences with 40 frames. Test sequences 1 and 2 are extracted from real video clips, and test sequences 3 and 4 were made using the method described in Section IV-B with images not used for learning. The cityscapes dataset was tested using a test sequence with 7,000 frames. The average luminance changes in each output sequence from the proposed model and pix2pix are shown in Fig. 16. The flicker of the proposed model is smaller than that in pix2pix in both experiments.

Table 4 shows that the proposed model produced a smaller value than pix2pix for both PF and FI, regardless of the patch size for all the test sequences from both datasets. The mean luminance change values for both the cityscapes and facades datasets in Fig. 16 show that the proposed model has less fluctuation than pix2pix. From Table 4 and Fig. 16, it is reasonable to conclude that the proposed model shows less flicker than pix2pix.

Table 5 shows that the pix2pix results have higher VIF and SSIM values than the results from the proposed model, but the difference is so small that it is almost meaningless.

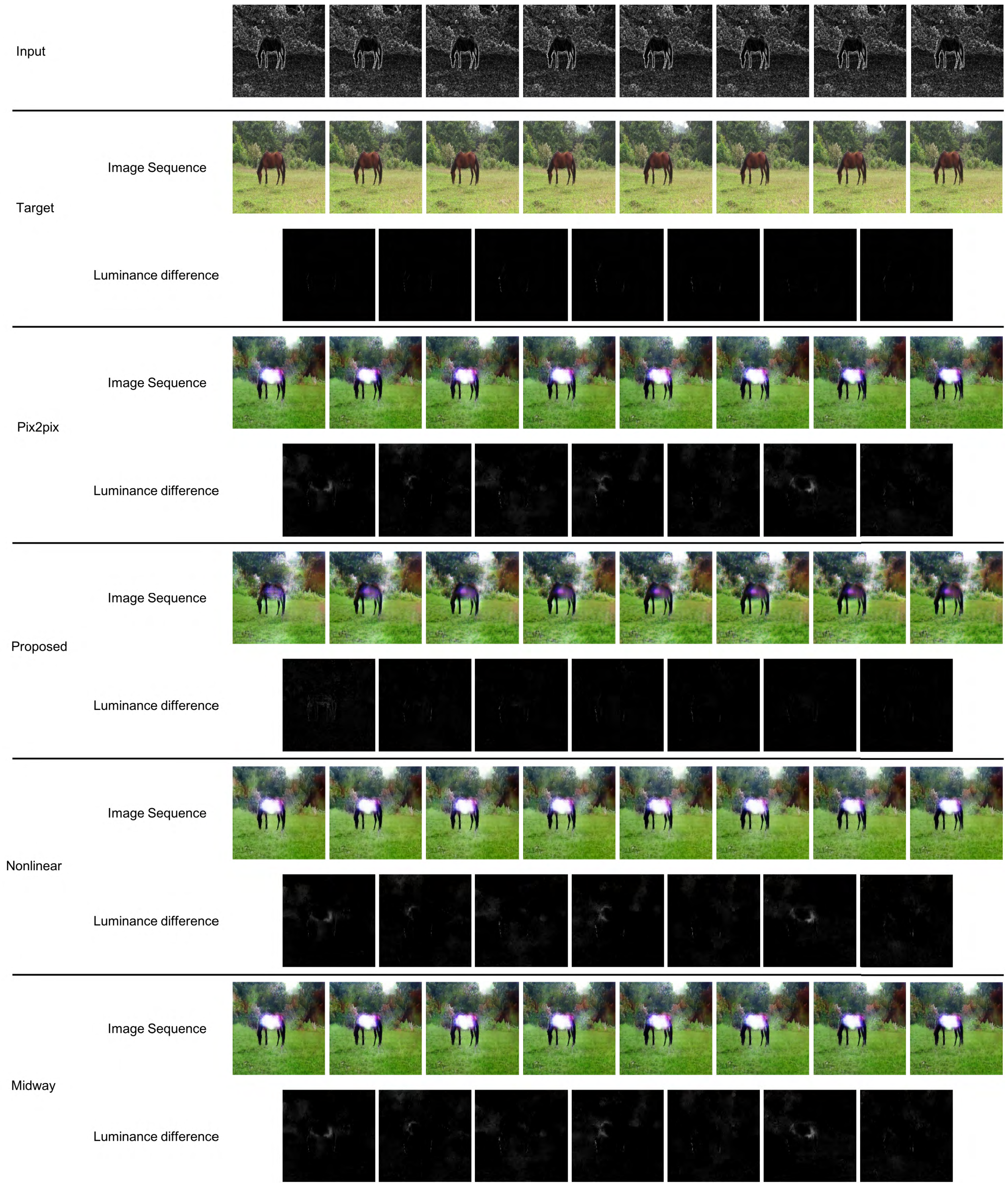

FIGURE 15. Luminance difference between the adjacent frames in each model and method. The luminance difference with the proposed model is much smaller than with pix2pix, nonlinear flicker compensation, and midway equalization.

C. IMAGE QUALITY AND FLICKER ACCORDING TO THE RATIO OF λ_1 AND λ_2

Training using a sequence that is made from an image is very different from using a video clip. For example, if we create an image sequence by zooming and then cropping, the model learns only how to expand the image. Therefore, if the zoom rate of the test sequence differs from the zoom rate of the image sequence set used for training, an after-image effect could be generated. The same is true for using warping and affine transformations to create a sequence

TABLE 4. Measured flicker of semantic labels-to-photo task with the cityscapes and facades datasets.

Cityscapes						
	Percent flicker					
patch size	1 × 1	2 × 2	4 × 4	8 × 8	16 × 16	32 × 32
pix2pix	10.46	8.99	7.53	6.10	4.85	3.84
proposed	**7.79**	**6.27**	**4.81**	**3.41**	**2.26**	**1.46**
	Flicker index					
pix2pix	$3.142e^{-2}$	$2.684e^{-2}$	$2.238e^{-2}$	$1.808e^{-2}$	$1.428e^{-2}$	$1.128e^{-2}$
proposed	$\mathbf{2.44e^{-2}}$	$\mathbf{1.95e^{-2}}$	$\mathbf{1.49e^{-2}}$	$\mathbf{1.05e^{-2}}$	$\mathbf{6.95e^{-3}}$	$\mathbf{4.46e^{-3}}$
Facades – Test sequence 1						
	Percent flicker					
patch size	1 × 1	2 × 2	4 × 4	8 × 8	16 × 16	32 × 32
pix2pix	15.71	14.82	13.13	10.45	7.45	4.76
proposed	**14.57**	**13.77**	**12.36**	**9.86**	**6.99**	**3.71**
	Flicker index					
pix2pix	$4.63e^{-2}$	$4.38e^{-2}$	$3.89e^{-2}$	$3.13e^{-2}$	$2.23e^{-2}$	$1.41e^{-2}$
proposed	$\mathbf{4.43e^{-2}}$	$\mathbf{4.20e^{-2}}$	$\mathbf{3.78e^{-2}}$	$\mathbf{3.07e^{-2}}$	$\mathbf{2.21e^{-2}}$	$\mathbf{1.17e^{-2}}$
Facades – Test sequence 2						
	Percent flicker					
patch size	1 × 1	2 × 2	4 × 4	8 × 8	16 × 16	32 × 32
pix2pix	9.67	8.54	7.04	5.36	4.08	3.12
proposed	**8.13**	**6.84**	**5.26**	**3.45**	**2.17**	**1.20**
	Flicker index					
pix2pix	$2.88e^{-2}$	$2.56e^{-2}$	$2.09e^{-2}$	$1.56e^{-2}$	$1.19e^{-2}$	$8.94e^{-3}$
proposed	$\mathbf{2.49e^{-2}}$	$\mathbf{2.14e^{-2}}$	$\mathbf{1.63e^{-2}}$	$\mathbf{1.06e^{-2}}$	$\mathbf{6.65e^{-3}}$	$\mathbf{3.67e^{-3}}$
Facades – Test sequence 3						
	Percent flicker					
patch size	1 × 1	2 × 2	4 × 4	8 × 8	16 × 16	32 × 32
pix2pix	14.16	13.34	11.55	8.43	6.18	4.43
proposed	**12.47**	**11.66**	**9.89**	**6.91**	**4.72**	**2.95**
	Flicker index					
pix2pix	$4.20e^{-2}$	$3.96e^{-2}$	$3.42e^{-2}$	$2.48e^{-2}$	$1.79e^{-2}$	$1.27e^{-2}$
proposed	$\mathbf{3.77e^{-2}}$	$\mathbf{3.53e^{-2}}$	$\mathbf{3.00e^{-2}}$	$\mathbf{2.10e^{-2}}$	$\mathbf{1.41e^{-2}}$	$\mathbf{8.81e^{-3}}$
Facades – Test sequence 4						
	Percent flicker					
patch size	1 × 1	2 × 2	4 × 4	8 × 8	16 × 16	32 × 32
pix2pix	8.11	7.27	5.84	4.41	3.58	2.83
proposed	**6.68**	**5.72**	**4.15**	**2.54**	**1.60**	$\mathbf{8.53e^{-1}}$
	Flicker index					
pix2pix	$2.38e^{-2}$	$2.13e^{-2}$	$1.70e^{-2}$	$1.28e^{-2}$	$1.03e^{-2}$	$8.19e^{-3}$
proposed	$\mathbf{2.01e^{-2}}$	$\mathbf{1.72e^{-2}}$	$\mathbf{1.24e^{-2}}$	$\mathbf{7.59e^{-3}}$	$\mathbf{4.78e^{-3}}$	$\mathbf{2.49e^{-3}}$

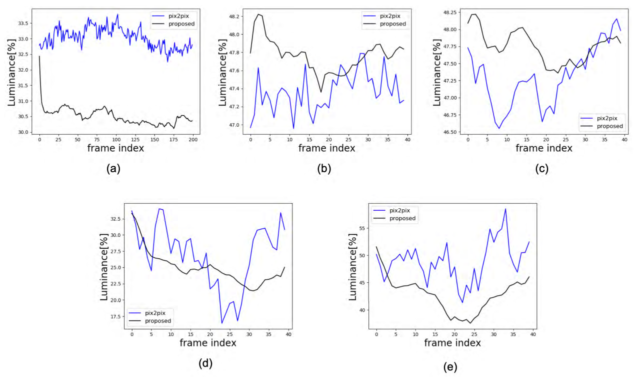

FIGURE 16. Change in mean luminance of the output sequence in semantic labels-to-photos task: (a) cityscapes dataset and (b – e) facades dataset (test sequences 1 – 4).

TABLE 5. Quality of Pix2pix and proposed model.

Cityscapes		
	VIF	SSIM
pix2pix	$\mathbf{2.96e^{-2}}$	$2.12e^{-1}$
proposed model	$2.63e^{-2}$	$\mathbf{2.12e^{-1}}$
Facades – Test sequence 1		
	VIF	SSIM
pix2pix	$\mathbf{4.43e^{-2}}$	$\mathbf{2.20e^{-1}}$
proposed model	$3.14e^{-2}$	$1.84e^{-1}$
Facades – Test sequence 2		
	VIF	SSIM
pix2pix	$\mathbf{3.38e^{-2}}$	$\mathbf{1.85e^{-1}}$
proposed model	$3.27e^{-2}$	$1.81e^{-1}$
Facades – Test sequence 3		
	VIF	SSIM
pix2pix	$\mathbf{5.28e^{-2}}$	$\mathbf{2.33e^{-1}}$
proposed model	$3.85e^{-2}$	$2.01e^{-1}$
Facades – Test sequence 4		
	VIF	SSIM
pix2pix	$\mathbf{4.90e^{-2}}$	$\mathbf{2.12e^{-1}}$
proposed model	$3.88e^{-2}$	$1.94e^{-1}$

from an image. Therefore, it is necessary to make training sequences that use various transformations to reduce such after-image effects. However, no matter how many transformations we use, the model cannot obtain new information from such a sequence.

In contrast, the actual video presents new information between frames. For example, when a person turns his head from left to right, the right face is new information. When a model is trained with a video clip, the model learns that the effect on the previous output is reduced for any part of the image in which new information appears in the input. When a model learns from using a sequence created from an image, the influence of the previous frame is maintained because the model cannot obtain new information from the input.

Therefore, it is important to select an appropriate λ_2 in Eq. (9). If λ_2 is set too high, the flicker might increase. To confirm the effect of the ratio of λ_2 and λ_1 on the output sequence, we fixed λ_1 in the facades dataset to 50 and measured the flicker and quality as we changed λ_2. We used the 4 test sequences described in section VI-B.

Due to the nature of the edge-to-photo task, the absolute difference from the original image is not the most important quality parameter. So, the proposed model is also helpful in improving the quality of the image. Image quality tended to decrease as λ_2 increased. However, the decreasing tendency was greater in test sequences 1 and 2 than in test sequences 3 and 4. That is, for a test sequence of patterns different from the learned sequence, the image quality deteriorated faster as λ_2 became larger. On the other hand, flicker decreased as λ_2 increased in both the PF and FI metrics. Fig. 17 shows that the luminance difference between frames decreased as λ_2 increased. That is, as the value of λ_2 increases, the flicker decreases, but the image quality deteriorates. Therefore, if we use a sequence created from an image for training, selecting an appropriate λ_2 is important for good results.

Table 7 shows the number of float operations and the number of parameters for each model. After training the model, we used only the generator; therefore, we compared the number of parameters and float operations in the generator only. For the number of parameters, the proposed model increased about 0.5% compared to pix2pix and it increased about 19% in the number of float operations.

TABLE 6. Flicker and quality of proposed model and Pix2pix with different λ_2 for the proposed model.

Test sequence 1				
	PF 1×1	FI 1×1	VIF	SSIM
pix2pix	15.71	$4.63e^{-2}$	$4.43e^{-2}$	$2.20e^{-1}$
proposed ($\lambda_2 = 5$)	14.65	$\mathbf{4.29e^{-2}}$	$\mathbf{6.74e^{-2}}$	$\mathbf{2.66e^{-1}}$
proposed ($\lambda_2 = 10$)	15.26	$4.29e^{-2}$	$6.74e^{-2}$	$2.25e^{-1}$
proposed ($\lambda_2 = 20$)	14.67	$4.39e^{-2}$	$5.14e^{-2}$	$2.29e^{-1}$
proposed ($\lambda_2 = 30$)	**14.57**	$4.43e^{-2}$	$3.13e^{-2}$	$2.29e^{-1}$
Test sequence 2				
	PF 1×1	FI 1×1	VIF	SSIM
pix2pix	9.67	$2.88e^{-2}$	$5.28e^{-2}$	$2.33e^{-1}$
proposed ($\lambda_2 = 5$)	9.17	$2.74e^{-2}$	$\mathbf{6.98e^{-2}}$	$\mathbf{2.67e^{-1}}$
proposed ($\lambda_2 = 10$)	8.81	$2.67e^{-2}$	$5.47e^{-2}$	$2.43e^{-1}$
proposed ($\lambda_2 = 20$)	8.46	$2.57e^{-2}$	$6.64e^{-2}$	$2.52e^{-1}$
proposed ($\lambda_2 = 30$)	**8.13**	$\mathbf{2.49e^{-2}}$	$3.85e^{-2}$	$2.01e^{-1}$
Test sequence 3				
	PF 1×1	FI 1×1	VIF	SSIM
pix2pix	14.16	$4.20e^{-2}$	$3.38e^{-2}$	$1.84e^{-1}$
proposed ($\lambda_2 = 5$)	13.76	$4.09e^{-2}$	$\mathbf{5.79e^{-2}}$	$\mathbf{2.45e^{-1}}$
proposed ($\lambda_2 = 10$)	13.27	$3.98e^{-2}$	$4.73e^{-2}$	$2.23e^{-1}$
proposed ($\lambda_2 = 20$)	13.37	$4.03e^{-2}$	$4.52e^{-2}$	$2.05e^{-1}$
proposed ($\lambda_2 = 30$)	**12.47**	$\mathbf{3.77e^{-2}}$	$3.27e^{-2}$	$1.81e^{-2}$
Test sequence 4				
	PF 1×1	FI 1×1	VIF	SSIM
pix2pix	8.11	$2.12e^{-2}$	$4.89e^{-2}$	$2.12e^{-1}$
proposed ($\lambda_2 = 5$)	7.88	$2.06e^{-2}$	$6.02e^{-2}$	$2.29e^{-1}$
proposed ($\lambda_2 = 10$)	7.35	$1.93e^{-2}$	$5.13e^{-2}$	$2.21e^{-1}$
proposed ($\lambda_2 = 20$)	7.18	$1.87e^{-2}$	$\mathbf{6.27e^{-2}}$	$\mathbf{2.31e^{-1}}$
proposed ($\lambda_2 = 30$)	**6.68**	$\mathbf{1.72e^{-2}}$	$3.87e^{-2}$	$1.94e^{-1}$

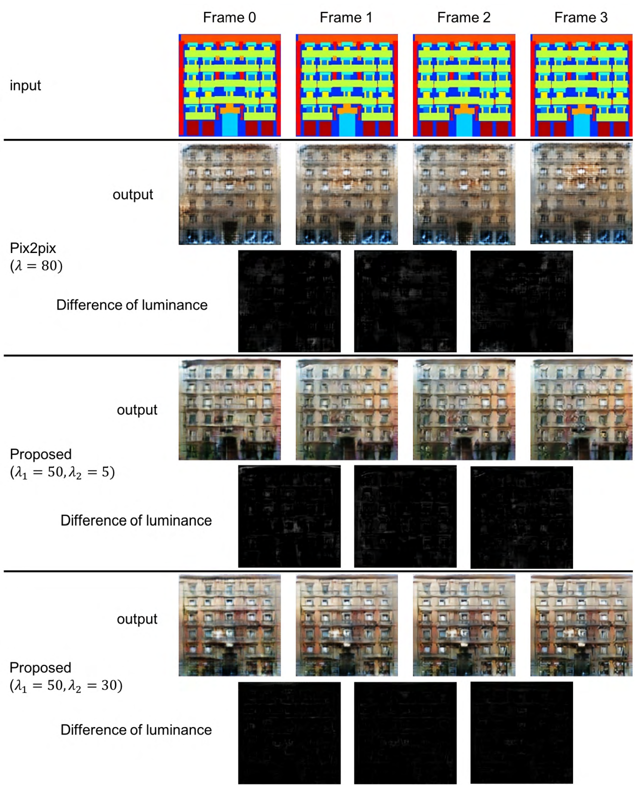

FIGURE 17. Images resulting from ratio changes in λ_1 and λ_2. The proposed model reduced flicker compared with pix2pix, and the image became clearer as the ratio of λ_2 became higher.

TABLE 7. Number of parameters and float operations.

Part of each model	Parameters (10^6)
G – pix2pix	**54.42**
G – proposed	**54.69**
D – pix2pix	2.77
D – proposed	2.77
total – pix2pix	57.19
total – proposed	57.46

Part of each model	Float operations (10^9)
G forward – pix2pix	**12.21**
G forward – proposed	**14.56**
G backward – pix2pix	24.20
G backward – proposed	28.91
D forward – pix2pix	6.40
D forward – proposed	6.40
D backward – pix2pix	14.66
D backward – proposed	14.66

D. INFLUENCE OF THE NUMBER OF PREVIOUS OUTPUT FRAMES

Fig. 18 presents the results from evaluating the 4 test sequences in the facades dataset described in Section VI-B. As the number of previous output frames input to the generative network increased, flicker was greatly reduced regardless of the test sequence. In test sequence 2, the VIF quality metric tended to deteriorate as the number of previous output frames input to the generative network increased. The remaining test sequences were degraded when the number of inputs to the generative network increased from 1 to 2, but when the number of inputs increased from 2 to 3, the quality improved again. Particularly in the case of test sequence 4, using three previous output frames as a conditional parameter produced higher VIF than using a single previous output. In most cases, using one various output frame as the input showed the highest quality value.

For SSIM, test sequence 1 showed the highest value when three previous output frames were used as conditional parameters. With test sequence 4, using three frames as conditional parameters produced better results than using 1 or 2 frames, but none of those results was higher than the results from pix2pix. For the remaining test sequences, the results from using three frames were worse than those from using frame, but the difference was not significant.

Overall, the flicker decreased significantly as the number of frames used as conditional parameters increased. Using three frames as conditional parameters showed a tendency

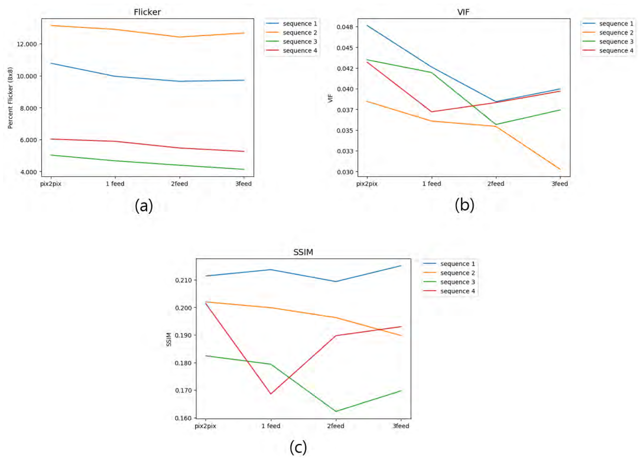

FIGURE 18. **The flicker and quality measurement results according to the number of frames used as a condition parameter. (a) Flicker. (b) VIF. (c) SSIM.**

to deteriorate the quality compared with using one frame, but those differences were tiny. Using two frames as conditional parameters showed the lowest quality values in all test sequences. Because the two previous output frames are almost the same, the generative network cannot learn the context between them. However, when three frames were used as conditional parameters, the generative network learned the context between the output frames, and the performance improved again.

Thus, as the number of previous frames used as conditional parameters increased, flicker decreased and quality deteriorated and then increased again. Therefore, when using several previous frames as conditional parameters, using more than three frames as conditional parameters can minimize quality deterioration. However, as the number of frames used as conditional parameters increases, the sequence length used for training must also increase, and collecting data in long sequences is more difficult than collecting short sequences. However, if a collection of long-sequence datasets is available, using three or more previous output frames as conditional parameters produces good overall performance.

VII. CONCLUSION

In this paper, we proposed a method to reduce flicker while maintaining image quality in image sequence transformation using a cGAN. The proposed method dramatically reduced flicker and maintained an image quality almost the same as existing image-to-image transformation with cGANs. However, the edge of the output sequence from the proposed model is blurred compared to pix2pix. Therefore, we could apply postprocessing to improve the sharpness. When the model was trained using a video clip, flicker was greatly reduced without reducing image quality. However, when a sequence made from an image was used to train the proposed model, an after-image effect was generated when the input sequence had a new pattern not used in training. To solve that problem, it is possible (for some parts of an image) to add a synthesizer that combines the current input frame with the previous output frame selectively according to whether a part of the image contains new information. Also, our model can be applied to image transformation with a more complicated structure such as that used in Cycle GAN, Star GAN, and Disco GAN.

REFERENCES

[1] M. Elad and M. Aharon, "Image denoising via sparse and redundant representations over learned dictionaries," *IEEE Trans. Image Process.*, vol. 15, no. 12, pp. 3736–3745, Dec. 2006. Accessed: Sep. 4, 2018. [Online]. Available: https://ieeexplore.ieee.org/document/4011956/. doi: 10.1109/TIP.2006.881969.

[2] M. Bertalmio, G. Sapiro, V. Caselles, and C. Ballester, "Image inpainting," in *Proc. 27th Annu. Conf. Comput. Graph. Interact. Techn. (SIGGRAPH)*, Hoboken, NJ, USA, 2000, pp. 417–424. Accessed: Sep. 4, 2018. [Online]. Available: https://dl.acm.org/citation.cfm?id=344972

[3] A. A. Efros and W. T. Freeman, "Image quilting for texture synthesis and transfer," in *Proc. 28th Annu. Conf. Comput. Graph. Interact. Techn. (SIGGRAPH)*, New York, NY, USA, 2001. pp. 341–346. Accessed: Sep. 4, 2018. [Online]. Available: https://dl.acm.org/citation.cfm?id=383296

[4] Q. Luan, F. Wen, D. Cohen-Or, L. Liang, Y.-Q. Xu, and H.-Y. Shum, "Natural image colorization," in *Proc. 18th Eurograph. Conf. Rendering Techn.*, Aire-la-Ville, Switzerland, 2007, pp. 309–320. Accessed: Sep. 4, 2018. [Online]. Available: https://www.cs.tau.ac.il/dcor/articles/2007/natural-image.pdf

[5] K. Nasrollahi and T. B. Moeslund, "Super-resolution: A comprehensive survey," *March. Vis. Appl.*, vol. 25, no. 6, pp. 1423–1468, Aug. 2014. Accessed: Sep. 4, 2018. [Online]. Available: https://dl.acm.org/citation.cfm?id=2647819. doi: 10.1007/s00138-014-0623-4.

[6] A. Buades, B. Coll, and J.-M. Morel, "A non-local algorithm for image denoising," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, San Diego, CA, USA, Jun. 2005, pp. 60–65. Accessed: Sep. 4, 2018. [Online]. Available: https://dl.acm.org/citation.cfm?id=344972

[7] R. Zhang, P. Isola, and A. A. Efros. "Colorful image colorization," presented at the Eur. Conf. Comput. Vision 2016 (ECCV). Accessed: Sep. 4, 2018. [Online]. Available: https://arxiv.org/abs/1603.08511

[8] S. Xie and Z. Tu, "Holistically-nested edge detection," in *Proc. IEEE Int. Conf. Comput. Vis. (ICCV),* Santiago, Chile, Dec. 2015, pp. 1395–1403. Accessed: Sep. 4, 2018. [Online]. Available: https://ieeexplore.ieee.org/document/7410521/

[9] D. Eigen and R. Fergus, "Predicting depth, surface normals and semantic labels with a common multi-scale convolutional architecture," in *Proc. IEEE Int. Conf. Comput. Vis. (ICCV),* Santiago, Chile, Dec. 2015, pp. 2650–2658. Accessed: Sep. 4, 2018. [Online]. Available: https://ieeexplore.ieee.org/document/7410661/

[10] J. Long, E. Shelhamer, and T. Darrell, "Fully convolutional networks for semantic segmentation," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR),* Boston, MA, USA, Jun. 2015, pp. 3431–3440. Accessed: Sep. 4, 2018. Available: https://ieeexplore.ieee.org/document/7298965/

[11] T. Chen, M.-M. Cheng, P. Tan, A. Shamir, and S.-M. Hu, "Sketch2Photo: Internet image montage," in *ACM Trans. Graph.*, vols. 5–28, Dec. 2009, Art. no. 124. Accessed: Sep. 4, 2018. doi: 10.1145/1618452.1618470.

[12] P. Isola, J.-Y. Zhu, T. Zhou, and A. A. Efros, "Image-to-image translation with conditional adversarial networks," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR),* Honolulu, HI, USA, Jul. 2017, pp. 5967–5976. Accessed: Sep. 4, 2018. [Online]. Available: https://ieeexplore.ieee.org/document/8100115/

[13] C. Ledig *et al.*, "Photo-realistic single image super-resolution using a generative adversarial network," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Honolulu, HI, USA, Jul. 2017, pp. 105–114. Accessed: Sep. 4, 2018. [Online]. Available: https://ieeexplore.ieee.org/document/8099502/

[14] J.-Y. Zhu, T. Park, P. Isola, and A. A. Efros, "Unpaired image-to-image translation using cycle-consistent adversarial networks," in *Proc. IEEE Int. Conf. Comput. Vis. (ICCV)*, Venice, Italy, Oct. 2017, pp. 2242–2251. Accessed: Sep. 4, 2018. [Online]. Available: https://ieeexplore.ieee.org/document/8237506/

[15] I. Goodfellow *et al.*, "Generative adversarial nets," in *Proc. 27th Int. Conf. Neural Inf. Process. Syst. (NIPS)*, 2014, pp. 2672–2680. Accessed: Sep. 4, 2018. [Online]. Available: https://papers.nips.cc/paper/5423-generative-adversarial-nets
[16] M. Mirza and S. Osindero. "Conditional generative adversarial nets." Accessed: Mar. 23, 2019. [Online]. Available: https://arxiv.or/abs/1411.1784
[17] T. Kim, M. Cha, H. Kim, J. Lee, and J. Kim. (2017). "Learning to discover cross-domain relations with generative adversarial networks." Accessed: Sep. 4, 2018. [Online]. Available: https://arxiv.org/abs/1703.05192
[18] Y. Choi, M. Choi, M. Kim, J.-W. Ha, S. Kim, and J. Choo. (2017). "StarGAN: Unified generative adversarial networks for multi-domain image-to-image translation." Accessed: Sep. 4, 2018. [Online]. Available: https://arxiv.org/abs/1711.09020
[19] M. E. Poplawski, N. M. Miller, "Flicker in solid-state lighting: measurement techniques, and proposed reporting and application criteria," presented at the Proc. CIE Midterm and Centenary Conf., Paris, France, Apr. 2013.
[20] C. Vondrick, H. Pirsiavash, and A. Torralba, "Generating videos with scene dynamics," in *Proc. Int. Conf. Neural Inf. Process. Syst. (NIPS)*, 2016, pp. 613–621, Accessed: Sep. 4, 2018. [Online]. Available: http://www.cs.columbia.edu/ vondrick/tinyvideo/paper.pdf
[21] N. Hara, A. Ichigaya, M. Kurozumi, Y. Nishida, and Y. Ohtsuka, "Flicker reduction in MPEG-2 video by post-processing," *IEEE Trans. Consum. Electron.*, vol. 51, no. 1, pp. 210–217, Feb. 2005. Accessed: Sep. 4, 2018. [Online]. Available: https://ieeexplore.ieee.org/document/1405722/. doi: 10.1109/TCE.2005.1405722.
[22] J. Delon, "Movie and video scale-time equalization application to flicker reduction," in *IEEE Trans. Image Process.*, vol. 15, no. 1, pp. 241–248, Jan. 2006. Accessed: Sep. 4, 2018. [Online]. Available: https://ieeexplore.ieee.org/document/1556641/. doi: 10.1109/TIP.2005.860328.
[23] J. Delon and A. Desolneux, "Stabilization of flicker-like effects in image sequences through local contrast correction," *SIAM J. Imag. Sci.*, vol. 3, no. 4, pp. 703–704, Oct. 2010. Accessed: Sep. 4, 2018. [Online]. Available: https://epubs.siam.org/doi/10.1137/090766371. doi: 10.1137/090766371.
[24] S. Barratt and R. Sharma. (Sep. 4, 2018). "A note on the inception score." [Online]. Available: https://arxiv.org/abs/1801.01973
[25] A. Borji. (Sep. 4, 2018). "Pros and cons of GAN evaluation measures." [Online]. Available: https://arxiv.org/abs/1802.03446
[26] G. Forbin and T. Vlachos, "Nonlinear flicker compensation for archived film sequences using motion-compensated graylevel tracing," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 18, no. 6, pp. 803–816, Jun. 2008. Accessed: Sep. 4, 2018. [Online]. Available: https://ieeexplore.ieee.org/abstract/document/4455568/. doi: 10.1109/TCSVT.2008.919114.
[27] *IEEE Recommended Practices for Modulating Current in High-Brightness LEDs for Mitigating Health Risks to Viewers*, IEEE Standard 1789-2015, 2015.
[28] J. Delon, "Midway image equalization," *J. Math. Imag. Vis.*, vol. 21, no. 2, pp. 119–134, 2004.
[29] X. Wang and A. Gupta, "Generative image modeling using style and structure adversarial networks," in *Proc. ECCV*, 2016, pp. 318–335.
[30] Z. Wang, A. C. Bovik, H. R. Sheikh, and E. P. Simoncelli, "Image quality assessment: From error visibility to structural similarity," *IEEE Trans. Image Process.*, vol. 13, no. 4, pp. 600–612, Apr. 2004.
[31] H. R. Sheikh and A. C. Bovik, "Image information and visual quality," *IEEE Trans. Image Process.*, vol. 15, no. 2, pp. 430–444, Feb. 2006.
[32] M. Cordts *et al.*, "The cityscapes dataset for semantic urban scene understanding," in *Proc. CVPR*, Jun. 2016, pp. 3213–3223.
[33] R. Tylecek and R. Šára, "Spatial pattern templates for recognition of objects with regular structure," in *Proc. GCPR*, Saarbrücken, Germany, 2013, pp. 364–374.
[34] O. Ronneberger, P. Fischer, and T. Brox, "U-Net: Convolutional networks for biomedical image segmentation," in *Proc. MICCAI*. Cham, Switzerland: Springer, 2015, pp. 234–241.
[35] *Pexels Videos*. Accessed: Sep. 4, 2018. [Online]. Available: https://videos.pexels.com/search/horse
[36] M. Abadi *et al.*, "Tensorflow: Large-scale machine learning on heterogeneous distributed systems," in *Proc. 12th USENIX Symp. Oper. Syst. Design Implement. (OSDI)*, Savannah, GA, USA, Nov. 2016, pp. 1–19.
[37] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," in *Proc. Int. Conf. Learn. Represent. (ICLR)*, 2014, pp. 1–15.
[38] T. Karras, T. Aila, S. Laine, and J. Lehtinen, "Progressive growing of GANs for improved Quality, Stability, and Variation," in *Proc. Int. Conf. Learn. Represent. (ICLR)*, 2018, pp. 1–16.
[39] J. Zhao, M. Mathieu, and Y. LeCun, "Energy-based generative adversarial network," in *Proc. Int. Conf. Learn. Represent. (ICLR)*, 2017, pp. 1–17.
[40] A. Radford, L. Metz, and S. Chintala, "Unsupervised representation learning with deep convolutional generative adversarial networks," presented at the Int. Conf. Learn. Represent. (ICLR), 2016.

SAN KIM received the B.S. degree in electronic engineering from Kyung Hee University, in 2017, where he is currently pursuing the M.S. degree in electronics engineering with the College of Electronics and Information. He has been a Korean Delegate for the ISO/IEC MPEG Forum, since 2017.

DOUG YOUNG SUH received the B.Sc. degree from the Department of Nuclear Engineering, Seoul University, South Korea, in 1980, and the M.Sc. and Ph.D. degrees in electrical engineering from the Georgia Institute of Technology, Atlanta, GA, USA, in 1986 and 1990, respectively. In 1990, he joined the Korea Academy of Industry and Technology and conducted research on HDTV, until 1992. Since 1992, he has been a Professor with the College of Electronics and Information Engineering, Kyung Hee University, South Korea. His research interests include networked video and video game. He has been a Korean Delegate for ISO/IEC MPEG, since 1996.