Self-Play Reinforcement Learning for Fast Image Retargeting

Nobukatsu Kajiura
n_kajiura@hal.t.u-tokyo.ac.jp
The University of Tokyo
Tokyo, Japan

Satoshi Kosugi
kosugi@hal.t.u-tokyo.ac.jp
The University of Tokyo
Tokyo, Japan

Xueting Wang
xt_wang@hal.t.u-tokyo.ac.jp
The University of Tokyo
Tokyo, Japan

Toshihiko Yamasaki
yamasaki@hal.t.u-tokyo.ac.jp
The University of Tokyo
Tokyo, Japan

ABSTRACT
In this study, we address image retargeting, which is a task that adjusts input images to arbitrary sizes. In one of the best-performing methods called MULTIOP, multiple retargeting operators were combined and retargeted images at each stage were generated to find the optimal sequence of operators that minimized the distance between original and retargeted images. The limitation of this method is in its tremendous processing time, which severely prohibits its practical use. Therefore, the purpose of this study is to find the optimal combination of operators within a reasonable processing time; we propose a method of predicting the optimal operator for each step using a reinforcement learning agent. The technical contributions of this study are as follows. Firstly, we propose a reward based on self-play, which will be insensitive to the large variance in the content-dependent distance measured in MULTIOP. Secondly, we propose to dynamically change the loss weight for each action to prevent the algorithm from falling into a local optimum and from choosing only the most frequently used operator in its training. Our experiments showed that we achieved multi-operator image retargeting with less processing time by three orders of magnitude and the same quality as the original multi-operator-based method, which was the best-performing algorithm in retargeting tasks.

CCS CONCEPTS
• Computing methodologies → Reinforcement learning: Image processing; Neural networks.

KEYWORDS
image retargeting; multi-operator; reinforcement learning; deep learning; self-play

Figure 1: Comparison of how to search for the optimal retargeted image by multi-operator image retargeting methods. For simplicity, the figure shows a three-step case with two kinds of operators (scaling and seam carving). MULTIOP [22] combines multiple retargeting operators and generates retargeted images at each stage. Whereas, in our method, the optimal retargeting operator is predicted for each step by a reinforcement learning agent.

1 INTRODUCTION
Image retargeting, which is a task of adjusting input images into arbitrary sizes, has been actively studied owing to the diversity in display devices and the versatility in the media sources of images. In image retargeting, it is important to generate natural results, while retaining important objects/regions. Nevertheless, it is difficult to achieve this with simple operations, such as cropping or uniform scaling. Although some content-aware retargeting methods have been proposed [1, 4, 7, 9, 13, 16, 21, 30, 31], using a single retargeting operator will not succeed in all cases or for all sizes. In this study, we apply a multi-operator image retargeting by utilizing several retargeting operators and appropriately combining them to obtain better results that will be tuned for each image.

In a previous method using multi-operators, Rubinstein et al. [22] proposed an image-to-image distance measure called Bi-Directional Warping (BDW), and the optimal combination of retargeting operators was searched for using dynamic programming. Although they achieved a better performance than other retargeting approaches, their approach had a critical limitation: a huge computational time. This is because retargeted images using multiple operators had to
be generated at each stage (Figure 1(a)). In contrast, we propose a high-speed multi-operator image retargeting by predicting the optimal retargeting operator step by step. We achieve this by using a reinforcement learning agent instead of generating multiple images to search for retargeting operator combinations (Figure 1(b)). By improving the efficiency of searching for the appropriate retargeting operators via reinforcement learning, fewer images are generated and the computational time is drastically reduced.

The purpose of the agent is to find retargeting operators that can minimize the distance between the original image and the retargeted image as much as possible. When applying reinforcement learning to this search, we find two problems and propose the following solutions for each. Firstly, as the dynamic range of distance (the BDW score) varies greatly depending on the image content, the distance cannot be directly used as a reward for training. Besides, we also find that predicting the BDW score via neural networks is extremely difficult. To solve this problem, we propose a self-play-based reward. By making an agent play against its copy and calculating the reward based on the victory or defeat, the agent can be trained based not on the absolute BDW score but the relative score between them. Secondly, simply using victory or defeat as a reward to the agent often leads to overfitting in which only one or two actions are selected. The retargeted images are then often worse than the results of MULTIOP [22]. This is because the chance of victory increases by just picking a relatively strong action. In order to solve this problem, we propose to dynamically change the loss weight of each action. By changing the weight of the loss according to the frequency of the action selected, the selection probabilities of the relatively strong action and the relatively less frequently used action are evaluated equally. As a result, the agent can correctly learn the optimal action.

Our experiments show that our method is faster by three orders of magnitude than MULTIOP [22]. Furthermore, we also show that our method can achieve the same image quality as MULTIOP [22] and that our image quality is better than those of the other state-of-the-art approaches.

Our main contributions are summarized as follows:

- We propose a reinforcement-learning-based method that can achieve ultra-fast multi-operator image retargeting.
- We show that a self-play-based reward can be insensitive to the large variance in the distance measure.
- We propose to dynamically change the loss weight of each action so that multiple operators could be evaluated and selected in balance to avoid overfitting.
- Experiments show that our method achieves multi-operator image retargeting faster by three orders of magnitude, and achieves the same image quality as MULTIOP [22] according to the user study.

2 RELATED WORK

2.1 Image Retargeting

For image retargeting, single-operator-based methods including hand-crafted techniques, deep-learning-based methods, and multi-operator-based methods are introduced. The key factor in image retargeting is how to suppress the loss and distortion of content.

A typical method for image retargeting is seam carving [1]. This method finds the optimal seam according to the image energy map via dynamic programming, and continuously removes seams to change the image size. Rubinstein et al. [21] introduced a new energy map and a graph-cut approach and achieved temporal-consistency-aware video retargeting. Han et al. [9] found multiple seams simultaneously with region smoothness and seam shape prior to using a 3-D graph-theoretic approach. Frankovich and Wong [4] introduced the absolute-energy cost function, which penalized seam candidates that crossed areas of local extrema.

Retargeting approaches based on warping were also reported. Liu and Gleich [16] used a non-linear fisheye-view warp that emphasized important regions while shrinking others. Gal et al. [7] designed a warping technique to preserve user-specified features by constraining their deformation to be a similarity transformation. Wolf et al. [31] introduced a non-homogenous mapping of video frames and retargeted videos by warping. Wang et al. [30] proposed scale-and-stretch warping, which distributed the distortion in all spatial directions, thus utilizing the available homogeneous regions to suppress the overall distortion. Krähenbühl et al. [13] proposed streaming video, which is a non-uniform and pixel-accurate warp to the target resolution.

Recently, deep-learning-based methods have been proposed. Cho et al. [3] proposed a weakly- and self-supervised learning model (WSSDCNN) that learns the shift map of each pixel in the input and output images. Tan et al. [26] proposed an unsupervised learning model, Cycle-IR, which learned the forward and reverse mapping of input and output images. Lee et al. [14] used object detection and object tracking with a deep neural network to enable consistent video retargeting. The deep-learning-based methods have a great advantage because their inference time can be very short. However, they are still inferior to hand-crafted algorithms because of the distortions in the resultant images.

Rubinstein et al. [22] claimed that using multiple retargeting operators for resizing images is often better than using a single one and proposed a method called MULTIOP, which searched for a suitable retargeting method for each image by combining multiple retargeting methods step by step. They included cropping, scaling, and seam carving into their retargeting operators, generated multiple retargeted images using dynamic programming, and searched for the optimal combination of retargeting operators for the original image. To evaluate the distance between original and retargeted images, they defined a new image similarity measure, called BDW. In a user study using the RetargetMe dataset [20], MULTIOP achieved the highest rating together with streaming video [13]. Because the combination of multiple retargeting operators achieved high performance, various improvements have been proposed. Zhang et al. [34] kept the quality of visually important objects by stretching original images in both vertical and horizontal directions and then applying seam carving and scaling. Song et al. [25] proposed deep-learning-based multi-operator image retargeting by learning the proportion of cropping, scaling, and seam carving operators. Compared to these two methods, we do not fix the order of retargeting operators, so we can explore a wider space of operator combinations. Zhou et al. [35] first applied reinforcement learning to find the optimal combination of multiple operators and used a semantic and aesthetic reward. However, we find that directly...
applying reinforcement learning could not use the BDW score as a reward. To solve this problem, we propose a self-play reinforcement learning architecture. The proposal can still exploit the advantage of BDW while aesthetic reward can be employed as well.

2.2 Reinforcement Learning for Image Processing

In recent years, reinforcement learning has been applied to image processing applications. Cao et al. [2] proposed a super-resolution method for facial images by letting the agent choose the local region to be enhanced. Park et al. [19] used Deep Q-Network [18] for color enhancement by iteratively choosing the image manipulation action. Hu et al. [10] proposed a photo retouching method for RAW images by choosing the image manipulation filter. Yu et al. [32] proposed an image restoration method by selecting a toolbox from a toolbox. Furuta et al. [5, 6] proposed a fully convolutional network that allowed agents to perform pixel-wise manipulations from a toolbox. Kosugi and Yamasaki [12] used reinforced adversarial learning for photo enhancement utilizing unpaired training data. Li et al. [15] proposed Aesthetics Aware Reinforcement Learning (A2-RL), which improved the aesthetic quality of images via image cropping. The agent iteratively chose the region of the cropping window to maximize the aesthetic score of the cropped image.

3 METHOD

The purpose of this study is to perform content-aware image retargeting. In MULTIOP [22], multiple retargeted images were generated at each step, and the optimal sequence of operations was decided using dynamic programming. However, the computational time was a big issue. We propose a method for predicting the optimal retargeting operator step by step using a reinforcement learning agent. While the conventional method generates multiple images with pruning, the proposed method generates retargeted images using the shortest path (Figure 1).

We show the overview of our method in Figure 2. We formulate image retargeting as a sequential decision-making process. The agent interacts with the environment and chooses an action to optimize the target. When we denote the original image as $I^\text{orig}$ and the current step number as $i$, the agent first receives the current state $s_{ij}$, which contains $I^\text{orig}_{ij}$ and the current retargeted image $I_{ij}$. Note that the retargeted image in the first step is the same as the original image (i.e., $I_{ij} = I^\text{orig}_{ij}$). Then, the agent samples the action $a_{ij}$ from the action space according to the probability distribution of the learned policy. Based on the selected action $a_{ij}$, the current retargeted image $I_{ij}$ is updated using a retargeting function $F$, that is, $I_{(i+1)j} = F(I_{ij}, a_{ij})$. This new image is used to make the new state $s_{(i+1)j}$, and the agent repeats the action sampling based on $s_{(i+1)j}$. This sequential decision-making process is repeated $\max_i$ times where $\max_i$ is used as the final retargeting result. As the proposed method does not need to generate multiple images by dynamic programming, the sequential process is faster than MULTIOP [22]. According to the evaluation of $I_{\max, j}$, the

agent receives a reward $R_{\max, j}$ at the end of the episode, and based on $R_{\max, j}$, the reward for the action of the agent at each step is defined as $R_{ij} = R_{\max, j} \times y^{\max-1}$, where $y$ is the discount factor.

As a reinforcement learning algorithm, we use the asynchronous advantage actor-critic (A3C) [17], which consists of two networks. The first one is a value network $V(s_{ij}; \theta_v)$, which estimates the value of the current state. The loss function to optimize the network parameter $\theta_v$ is defined so that $V(s_{ij}; \theta_v)$ can predict the reward, $L^v_{ij} = (R_{ij} - V(s_{ij}; \theta_v))^2 / 2.$

The second network is a policy network $\pi(a_{ij}|s_{ij}; \theta_p)$, which outputs the probability of each action. The network parameter $\theta_p$ is optimized to minimize the following loss function,

$$L^\pi_{ij} = -\log \pi(a_{ij}|s_{ij}; \theta_p)(R_{ij} - V(s_{ij}; \theta_v)) - \beta H(\pi(s_{ij}; \theta_p)),$$

where $H$ is a function that calculates entropy, which encourages the agent to explore and prevents convergence to a local optimum. By minimizing $L^\pi_{ij}$, the policy network $\pi(a_{ij}|s_{ij}; \theta_p)$ is trained to maximize the expected reward.

In the following sections, we describe in detail the state and action spaces, the reward, and the training loss of our framework.

3.1 State and Action Spaces

In A3C, the agent determines the action according to the policy output calculated from the current state at each step. The state contains the observation from the environment. In a sequential decision-making process, the current state can be represented as $s_{ij} = \{o_{ij}, o_{i1}, \cdots, o_{(i-1)j}, o_{ij}\}$, where $o_{ij}$ is the current observation of the agent. The historical experience is usually important for future decision-making because a human’s decision-making process considers not only the current observation but also historical experience. To memorize the historical observations, we use an LSTM unit following the A2-RL model [15]. In our model, the current observation $o_{ij}$ consists of the original image $I^\text{orig}_{ij}$, the current retargeted image $I_{ij}$, and a one-hot vector $u$ representing the number of steps to the end of the episode.

As for the action space, we define the agent’s actions as selecting a retargeting operator and applying that operator to the image. In our model, left cropping (CRL), right cropping (CRR), scaling (SCL), and seam carving (SC) [21] are used as retargeting operators. We let $a_{ij}$ take the value of $\{0, 1, 2, 3\}$ and associate each value to each operator. Note that these retargeting operators are chosen to match those in MULTIOP [22], and it is easy to add or delete a retargeting operator as needed. All these actions adjust the image width by 2.5% of the original image size. The observation and action space are illustrated in Figure 2 for an intuitive representation.

3.2 Self-Play-based Reward

In our reinforcement learning framework, the agent receives a reward according to the evaluation of the retargeted image. In a previous retargeting method, MULTIOP [22], they defined a new image similarity measure named BDW and used this measure to evaluate the distance between the original image and the retargeted image. In this study, we also provided the agent with a reward based on BDW. Note that other evaluation functions such as aesthetic score [29] can be used as a reward. The simplest reward for the
The architecture of the self-play reinforcement learning model is illustrated in Figure 2. The agent receives two states, $s_{ij}$ and $s'_{ij}$, and samples two actions $a_{ij}$ and $a'_{ij}$ from the action space according to the probability distribution of the policy output $\pi(a_{ij}|s_{ij}; \theta_{\pi})$ and $\pi(a'_{ij}|s'_{ij}; \theta_{\pi})$. Subsequently, the agent executes the sampled actions to update the current retargeted images $I_{ij}$ and $I'_{ij}$ to the new retargeted images $I_{(i+1)j}$ and $I'_{(i+1)j}$, respectively. The observation of the state and the selection of the action are repeated, and at the end of an episode, the agent receives a reward based on the victory or defeat of the BDW score. The self-play-based reward $R_{ij}^{max}$ is formulated as:

$$R_{ij}^{max} = \begin{cases} +1 & \text{if } \text{BDW}(I_{ij}^{orig}, I_{ij}^{max}) < \text{BDW}(I'_{ij}^{orig}, I'_{ij}^{max}) \\ -1 & \text{otherwise} \end{cases}$$

(3)

By using the self-play-based reward, the agent considers only the relative BDW score and can deal with the large variance in the BDW score.

### 3.3 Frequency-Aware Weighted Loss

When using a self-play-based reward, we find that the agent often leads to a local optimum in which only a few actions are selected and the retargeted images are often worse than the results of MUL-TIOP [22]. This is because the chances of victory are increased by just picking a relatively strong action. To solve this problem, we propose to dynamically change the loss weight of each action. Hence, the policy output of the relatively strong action and the relatively weak action are evaluated in balance; we change the loss weight according to the number of times the action is selected. In every episode, we count how many times each action is selected in the case of winning and losing. We define the counted results as four-dimensional vectors $f_{win}^{ij}$ and $f_{lose}^{ij}$. For example, if $a_{ij}$ takes the value of 1 three times, and the agent wins, $f_{win}^{ij}$ increases by three. We count these numbers over multiple images $I_j$ and we...
treat these multiple images as a mini-batch \( \mathcal{B} \). When the network parameters are updated, the loss weight \( w_{ij} \) is calculated as:

\[
w_{ij} = \begin{cases} 
\min(f^{\text{win}})/f^{\text{win}}_{aij} & \text{(if } R_{\text{max},ij} = +1) \\
\min(f^{\text{lose}})/f^{\text{lose}}_{aij} & \text{(if } R_{\text{max},ij} = -1) 
\end{cases}
\]

and gradients with reference to \( \theta_\pi \) and \( \theta_\alpha \) are accumulated as:

\[
d\theta_\pi = \sum_{j=0}^{\lfloor |\mathcal{B}| \rfloor - 1} \sum_{i=0}^{l_{\text{max}} - 1} \nabla_{\theta_\pi} w_{ij} L_i \pi_i \\
d\theta_\alpha = \sum_{j=0}^{\lfloor |\mathcal{B}| \rfloor - 1} \sum_{i=0}^{l_{\text{max}} - 1} \nabla_{\theta_\alpha} w_{ij} L_i \pi_i,
\]

where \( |\mathcal{B}| \) denotes the batch size of the mini-batch \( \mathcal{B} \). Utilizing this frequency-aware weighted loss, it is possible to avoid falling into a local optimum, in which the relatively strong action is always selected. We provide the whole training procedure of our reinforcement learning model in Algorithm 1.

4 EXPERIMENTS

4.1 Experimental Settings

Dataset. To train and test our model, we used the MIRFLICKR-1M dataset [11], which was composed of one million images downloaded from Flickr using the Creative Commons license. From this dataset, we extracted 3,000 and 100 landscape-oriented images without loss of generality for training and testing, respectively. As an additional test dataset, we used the RetargetMe dataset [20], which was the benchmark for image retargeting and contained 80 images. We selected 68 landscape-oriented images for testing.

Implementation. We used landscape-oriented images as inputs and retargeted them to shorten their width. During training, \( l_{\text{max}} \) was randomly sampled from \( \{1, ..., 20\} \) at the start of the episode. As the retargeted image was 2.5\% shorter than the original image in each step, the final retarget image size was 97.5\% to 50\% of the original size. During the test process, by specifying \( l_{\text{max}} \), retargeted images with the desired size were obtained.

During the observation, the original image was resized so that the width was 40 pixels; it was then padded with zero values so that the height was also 40 pixels. Furthermore, the retargeted image was resized so that the height was the same as the resized original image; it was then padded with zero values so that the height was 40 pixels. The vector \( u \) represents the number of steps to the end of the episode, was set to 20 dimensions. The elements of \( u \) from the first to the \((20 - l_{\text{max}})\)-th were initialized to zero; the other elements were initialized as one. At the end of the \( i \)-th step, the \((20 - l_{\text{max}} + i)\)-th element was changed to zero.

Our model started with a 6-layer convolution block, which received the merged original and retargeted image and outputted a 25,600-dimensional vector. The output was concatenated with the vector \( u \) and was the input for three fully-connected layers and an LSTM layer. It outputted a 1,024-dimensional vector into the value network and policy network. The value network consisted of one fully-connected layer and outputted the estimated value of the current state. Likewise, the policy network consisted of one fully-connected layer but outputted a 4-dimensional vector, where each element corresponded to the probabilities for taking each action.

We optimized our model using the RMSProp [27] algorithm with a learning rate of \( 7 \times 10^{-4} \); the other parameters were set to their default values. We trained for 10,000 episodes and used the model after the final episode for the test; it took 35 hours to complete the training. The mini-batch size \( |\mathcal{B}| \), the discount

\[\text{Algorithm 1: Training procedure of our self-play RL model}\]

\begin{itemize}
  \item[1] \( i_{\text{max}} \) is sampled from \( \{1, ..., 20\} \);
  \item[2] \( f^{\text{win}} = [0, 0, 0, 0], f^{\text{lose}} = [0, 0, 0, 0] \);
  \item[3] for \( j = 0 \) to \( |\mathcal{B}| - 1 \) do \\
    \item[4] \( f_{ij}^{\text{prev}} = \mathcal{B}_j \);
    \item[5] \( l_{ij} = f_{ij}^{\text{prev}} \), \( l'_{ij} = f_{ij}^{\text{prev}} \);
    \item[6] \( u = \{0, ..., 0, 1, ..., 1\} \);
    \item[7] \( 20 - i_{\text{max}} \) \( i_{\text{max}} \) do \\
      \item[8] Make the state \( s_{ij} \) from \( f_{ij}^{\text{prev}}, l_{ij}, \) and \( u \);
      \item[9] Make the state \( s'_{ij} \) from \( l_{ij}, l'_{ij}, \) and \( u \);
      \item[10] Sample \( a_{ij} \) and \( a'_{ij} \) according to \( \pi(a_{ij}|s_{ij}; \theta_\pi) \) and \( \pi(a'_{ij}|s'_{ij}; \theta_\pi) \);
      \item[11] \( l_{(i+1)j} = F(l_{ij}, a_{ij}) \), \( l'_{(i+1)j} = F(l'_{ij}, a'_{ij}) \);
      \item[12] \( u_{20 - i_{\text{max}} + i} = 0 \);
    \item[13] \end{do本案}
    \item[14] \( R_{\text{max},ij} = \begin{cases} +1 & \text{(if } \text{BDW}(l'_{ij}, l_{max,ij}) < \text{BDW}(l_{ij}, l'_{max,ij}) \} \\
      -1 & \text{(otherwise)}
      \end{cases} \)
    \item[15] for \( i = i_{\text{max}} - 1 \) to 0 do \\
      \item[16] if \( R_{\text{max},ij} = +1 \) then \\
        \item[17] \( f_{ij + 1}^{\text{win}} = 1 \);
      \item[18] else \\
        \item[19] \( f_{ij + 1}^{\text{lose}} = 1 \);
      \item[20] end \\
    \item[21] end \\
    \item[22] \( d\theta_\pi = 0 \), \( d\theta_\alpha = 0 \);
    \item[23] for \( j = 0 \) to \( |\mathcal{B}| - 1 \) do \\
      \item[24] for \( i = i_{\text{max}} - 1 \) to 0 do \\
        \item[25] \( w_{ij} = \begin{cases} \min(f^{\text{win}})/f^{\text{win}}_{aij} & \text{(if } R_{\text{max},ij} = +1) \\
          \min(f^{\text{lose}})/f^{\text{lose}}_{aij} & \text{(if } R_{\text{max},ij} = -1) \}
        \end{cases} \)
        \item[26] \( R_{ij} = \gamma R_{(i+1)j} \);
        \item[27] \( d\theta_\pi = d\theta_\pi + \nabla\theta_\pi L_i \pi_i \);
        \item[28] \( d\theta_\alpha = d\theta_\alpha + \nabla\theta_\alpha L_i \pi_i \);
      \item[29] end \\
    \item[30] end \\
    \item[31] Update \( \theta_\pi \) with \( d\theta_\pi \) and \( \theta_\alpha \) with \( d\theta_\alpha \);
\end{itemize}
factor $\gamma$, and the weight of the entropy loss $\beta$ were set to 16, 0.99, and 0.01, respectively.

### 4.2 Qualitative Evaluation

We show the retargeting results of our method and some previous methods in Figure 4 and Figure 5, where the retargeted size is 50% of the original size. The previous methods: scaling, GAIC [33], seam carving [21], MULTIOP [22], WSSDCNN [3], and Cycle-IR [26] were compared. The retargeting operators of scaling and seam carving were the same as the operators used in our method. GAIC is a method used to find the optimal cropping window by taking into account the contents of the image. MULTIOP uses multiple operators and searches for the optimal combination using dynamic programming; this increases the computational time. WSSDCNN and Cycle-IR are deep-learning-based methods. As shown in Figure 4 and Figure 5, the retargeting results by scaling, seam carving, WSSDCNN, and Cycle-IR retained the information of the original image; nevertheless, the structure was distorted. GAIC generated natural results; however, the information of the original image was lost. Compared to these methods, MULTIOP preserved the natural structure of the original image and its important information. Our method could also naturally retarget images while keeping important information; our results are very similar to the results of MULTIOP. The BDW scores also show that the results from MULTIOP, and our method is almost the same and better than the other methods. These results show that our method achieved a multi-operator image retargeting that has the same performance as MULTIOP. Figure 6 shows the results where the retargeted size was 75% of the original size and how the actions were sequentially applied to the images. It is shown that the appropriate combination of the retargeting operators for each image was selected by the agent and that the multi-operator image retargeting was achieved through reinforcement learning.

To analyze our method, we conducted ablation experiments. We used a self-play-based reward to deal with the large variance in the BDW scores. To verify the efficacy of the self-play-based reward,
We evaluated our method through a user study. As described in 4.1, we used 100 images from the MIRFLICKR-1M dataset [11] and 68 landscape-oriented images were retargeted to 50% of the original width and 29 landscape-oriented images were retargeted to 75% of the original width. Regarding the RetargetMe dataset, 39 landscape-oriented images were retargeted to both 75% and 50% of the original width. Among these, the RetargetMe dataset was retargeted to both 75% and 50% of the original width. For this reason, we compared theirs with our results only under this condition. In a user study, 50 crowd workers via the Amazon Mechanical Turk were asked to compare two images retargeted by our method and one of the previous methods; they were then instructed to select the better image. All images were arranged randomly to avoid bias. Table 1 shows the average vote rate. Our method obtained a higher vote rate than scaling, GAIC [33], seam carving [21], WSSDCNN [3], and Cycle-IR [26]; this shows that our method is subjectively superior to these methods. Furthermore, there was almost no difference in the average vote rate between the MULTIOP [22] and our method, showing that the performance of our method was subjectively similar to that of MULTIOP.

### 4.4 Time Efficiency

The benefit of the proposed method is that we can retarget images much faster than MULTIOP [22] while maintaining the image quality by using reinforcement learning. To show the time efficiency of our method, we compared the computational time of our method to that of MULTIOP. We used the RetargetMe dataset [20] for the evaluation; all input images were resized to $640 \times 480$ px. All processes were done on the same machine, which had an Intel® Xeon® Gold 6136 (3.00 GHz) CPU. The average computational time of our model and MULTIOP [22] are shown in Table 2. As shown in this table, our method achieved a multi-operator image retargeting that was much faster than MULTIOP [22] while maintaining the image quality. Compared to MULTIOP, which generated multiple images and evaluated them with BDW, the proposed method predicted the appropriate operators step by step, which resulted in a faster image retargeting. The time and space complexities of the MULTIOP [22] were $O(i^3)$. As this is a polynomial in the number of steps $i$, but an exponential in the number of operators $n$, the calculation time of MULTIOP increased greatly as the number of steps gets bigger. In comparison, the computational time of our method did not change drastically, even when the retarget ratio increased. This trend became more pronounced as the number of operators increased.

![Figure 5: Qualitative comparison with other deep-learning-based methods where the retargeted size was 50% of the original size: (a) original image, (b) WSSDCNN [3], (c) Cycle-IR [26], (d) Our method. The number below each image represents the BDW score. (@yanivba, Wolfgang Staudt)](image)

**Figure 5:** Qualitative comparison with other deep-learning-based methods where the retargeted size was 50% of the original size: (a) original image, (b) WSSDCNN [3], (c) Cycle-IR [26], (d) Our method. The number below each image represents the BDW score. (@yanivba, Wolfgang Staudt)

![Image](image)

![Image](image)

![Image](image)

![Image](image)

![Image](image)

![Image](image)

**Table 1: Average vote rate [%].**

| Method         | RetargetMe MIRFLICKR-1M | RetargetMe MIRFLICKR-1M |
|----------------|-------------------------|-------------------------|
| SCL / Ours     | 43.8/56.2               | 42.1/57.9               |
| GAIC [33] / Ours | 42.2/57.8             | 45.7/54.3               |
| SC [21] / Ours | 40.9/59.1              | 46.6/53.4               |
| MULTIOP [22] / Ours | 48.3/51.7       | 50.2/49.8               |
| WSSDCNN [3] / Ours | 41.5/58.5           | 45.2/54.8               |
| Cycle-IR [26] / Ours | -                     | -                       |

| Method  | 75%  | 50%  |
|---------|------|------|
| Ours    | 46.9/53.1 | - |

**Table 2: Average computational time of $640 \times 480$ images.**

| Method         | 75%  | 50%  |
|----------------|------|------|
| MULTIOP [22]   | 3400 s | 32000 s |
| Ours           | 5.0 s  | 9.9 s  |
4.5 Reward Option

In the above sections, the BDW scores are used as the reward, and our model searches for a retargeted image that minimizes the BDW score. In addition to the BDW score, other reward functions can be integrated into our method very easily. In this section, we show the experiment where the reward is replaced with another evaluation function, the aesthetic score proposed by Wang et al. [29]. In this setting, we search for a retargeted image that maximized the aesthetic score. When the aesthetic function is denoted as AES, the reward Eq. (3) is replaced as follows,

\[
R_{i_{\text{max}}} = \begin{cases} 
+1 & \text{(if } \text{AES}(I_{i_{\text{max}}}) > \text{AES}(I'_{i_{\text{max}}}) \text{)} \\
-1 & \text{(otherwise)}
\end{cases}
\]  

(8)

Figure 7 shows the results of our model based on the aesthetic score and other methods. For comparison, we conduct experiments where MULTIOP [22] optimized the aesthetic score instead of the BDW score (Figure 7(e)). Although our retargeting results (Fig 7(f)) did not exactly match results by MULTIOP [22], our method obtained higher aesthetic scores than the other methods previously described. These results show that the efficacy of our method does not depend on the type of evaluation function.

5 CONCLUSIONS

In this study, we addressed image retargeting, a task where we adjust input images into arbitrary sizes. Despite the previous multi-operator method having a high performance, the method required a huge computational time for generating multiple retargeted images to find the best combination of retargeting operators. Therefore, we proposed a reinforcement-learning-based method to achieve fast multi-operator image retargeting by predicting the optimal retargeting operator step by step. To deal with issues of a large variance in the evaluation value, and a local optimum where only the relatively strong action is selected, we proposed a self-play-based reward and a frequency-aware weighted loss. These two contributions enabled us to achieve a fast and effective multi-operator image retargeting via reinforcement learning. Experimental results showed that our method achieved multi-operator image retargeting that was faster by three orders of magnitude and had the same performance as the state-of-the-art method.

ACKNOWLEDGMENTS

A part of this research was supported by JSPS KAKENHI Grant Number 18H03339, 19K20289.
