Computer Modeling in Engineering & Sciences

Tech Science Press

DOI: 10.32604/cmes.2022.018201

ARTICLE

Image Translation Method for Game Character Sprite Drawing

Jong-In Choi[1], Soo-Kyun Kim[2] and Shin-Jin Kang[3,*]

[1]Department of Digital Media, Seoul Women's University, Seoul, 01797, Korea

[2]Computer Engineering, Jeju National University, Jeju, 63243, Korea

[3]School of Games, Hongik University, Sejong, 30016, Korea

[*]Corresponding Author: Shin-Jin Kang. Email: directx@hongik.ac.kr

Received: 06 July 2021 Accepted: 04 November 2021

ABSTRACT

Two-dimensional (2D) character animation is one of the most important visual elements on which users' interest is focused in the game field. However, 2D character animation works in the game field are mostly performed manually in two dimensions, thus generating high production costs. This study proposes a generative adversarial network based production tool that can easily and quickly generate the sprite images of 2D characters. First, we proposed a methodology to create a synthetic dataset for training using images from the real world in the game resource production field where machine learning datasets are insufficient. In addition, we have enabled effective sprite generation while minimizing user input in the process of using the tool. To this end, we proposed a mixed input method with a small number of segmentations and skeletal bone paintings. The proposed image-to-image translation network effectively generated sprite images from the user input images using the skeletal loss. We conducted an experiment regarding the number of images required and showed that 2D sprite resources can be generated even with a small number of segmentation inputs and one skeletal bone drawing.

KEYWORDS

Sprite generation; body segmentation; pose estimation; generative adversarial network; deep learning

1 Introduction

With the recent development of generative adversarial network (GAN)-based image-to-image translation technology, various artificial intelligence (AI)-assisted two-dimensional (2D) image tools have been suggested. Attempts to easily create intended high-level image data through a simple interface are being continuously made in the sketch [1], carton [2], and caricature [3] fields. In line with this trend, many attempts have been made to introduce GAN-based methods in the game sprite field as well. This study aims to improve the efficiency of the 2D game sprite production process by applying the GAN-based image-to-image translation technology to the game sprite production field. The 2D sprite production in games require many trials and errors. Due to the nature of game development, 2D characters or object designs are changed frequently, and the image size compression and shape change works occur frequently. In the traditional 2D

sprite production process, these works were performed manually, and this generated high costs. If sprites that have the texture and shape intended by users can be produced quickly in this process, the game production cost can be greatly reduced.

We have created a tool that allows users to produce sprites just by selecting a few colors and drawing simple bone structures. To that end, we have created a large synthetic dataset for training on the game engine, and trained the image-to-image network using this dataset. The trained network generates game sprites from simple drawings created by the user.

The methodology that we propose requires user's actions such as coloring each part of the sprite shape and drawing the core skeleton with a single line. Then from this, the system automatically generates sprite images, which are continuous animations of 2D images used in the games. The final output can be used as a free production output during 2D sprite production. We have applied the proposed AI-assisted sprite tool with various input combinations, and verified the possibility of using it as a preprocessing tool for sprite production in game development. In addition, the proposed method can be used for animation prototyping, quick viewing of results, or searching characters with user-defined poses.

2 Previous Works

Image generation and transformation using GANs is a technology for generating new images that did not exist before by receiving various noises or to transform input images or videos to images or videos having a different shape or information. First introduced in a paper titled "Generative Adversarial Nets" [4], GANs, unlike the traditional methods, uses two artificial neural networks that trains and uses one artificial neural network. One is a generative neural network that generates images, and the other is a discriminative neural network that discriminates whether the image generated by the generative neural network is real or fake. GANs are used in various research and application fields including image generation [5], image inpainting [6], text generation [7], medical image processing [8–15], semantic segmentation [16–19], image coloring [20,21], image-to-image translation [22], and art generation [23]. Moreover, GANs are widely used in face synthesis and face editing such as face age [24–26] and gender translation [27].

However, it has been pointed out that the training GANs is unstable. To solve this problem, Google announced deep convolutional generative adversarial networks (DCGANs) [28]. They improved the stability and performance of training by replacing every layer that cannot be differentiated—such as max pooling in expanding noises in the generative neural network structure of GAN—with convolutions that can be differentiated. This method showed much higher performance than the GANs that have been attempted before and demonstrated that the images generated by the generative neural network can be controlled. After DCGAN, Google announced the boundary equilibrium generative adversarial networks (BEGANs) [29] which boosts the performance of GANs by focusing on the distribution of the loss function itself when transforming images, rather than focusing on the image data distribution of loss functions for training the GAN. They showed that the difference in the degree of training progression between the generative and discriminative neural networks greatly affects the performance of images generated by the generative neural network. They generated 128×128 images for the first time, and showed the best image performance of GAN at the time. Pix2Pix [30] suggested a GAN that receives an image as input and translates it to a different type of image beyond the traditional GAN that generates images with a distribution of a training dataset by translating noise inputs to an image. The U-net structure was used for the generative neural network that generates an image, and the discriminative neural network discriminated images by receiving a pair of generative images of

contents different from the input image. It showed that image translation can be done naturally by using the L1 loss, which discriminates between the generated image in pixel units, and the loss used in GAN training together.

The initial GAN models had two problems. The first problem is that the resolution of the generated image was only 128 × 128. The second problem is that for images generated from noise, it was impossible to predict what the image looks until the generation is finished. Recent research on GANs has attempted to solve these two problems. ProGAN [31] suggested the progressive growing method, which generates high-resolution images by learning the method of generating images step by step from ultra-low resolutions (4 × 4) to generate high-resolution face images. In the past, BEGAN [29], EBGAN [32], and DCGAN [28] could generate face images of 128 × 128 resolution, but the resolution of recently generated face images has dramatically increased to 1024 × 1024. SinGAN [33] used a different generation model in each step by applying ProGAN [31]. They obtained the effect of removing artifacts by adding noise when generating an image in each step. Then they removed the weight and successfully generated high-resolution images by increasing the image size by 4/3 times in each step.

Pix2PixHD [34] is a representative image translation study that generates images using edge information or a segmentation map. Unlike the traditional Pix2Pix [30], they additionally used instance information. The existing segmentation map had the problem that when two identical objects are overlapped, their differences could not be indicated. Hence, they input a boundary map in addition to a segmentation map. SPADE [35] claimed that the layout normalization layer was wrong in the segmentation map. They obtained better results through a new layout normalization layer than other algorithms by improving on the problem that when large parts in the segmentation map have the same label, the texture information is lost.

Recent image translation studies have generated images using layouts. LostGAN [36] researched on the generation of images from layouts. The research of generating images from layouts need to predict the shape first because the shape of objects is not given. They used a new normalization layer called ISLA Norm to predict shapes and used it for image generation. Layout2image [37] attempted to identify the relationships between objects by applying the long short-term memory(LSTM) network to layouts.

With the development of the GAN algorithm, studies to apply it to super resolution have increased. Unlike the traditional methods, SRGAN [38] proposed a perceptual loss unlike the existing methods to restore texture information. Perceptual loss does not simply compare the image pixel values after super resolution, but compares them in a feature space extracted by a deep network. As a result, the fine texture information could be restored naturally when they are seen by people. This obtained better visual results than Dong et al. [39] who only used a deep network in the past.

Deepfill [40] is a method researched to perform image inpainting using GANs. Image inpainting refers to a task to restore missing parts in images. They have recently released Deepfillv2 [41] by upgrading the image inpainting result using gated convolution. In addition, a study that performed image inpainting by reflecting user's intention more clearly has been published. SC-FEGAN [42], which restores images in accordance with a sketch or color, or edge-connect [43], and which uses edge information has been published as well.

StyleGAN [44] is a technology that has been developed from ProGAN [31]. It has the advantage that the hair color, age, gender, etc. of people can be analyzed and freely changed by the network. They used the normalization layer, which was influenced by AdaIN [45], to give style

information such as the gender, hair color, age group of people to the network. However, strange patterns appeared in the output when images were generated. To improve this problem, they released StyleGAN2 [46] which improved the existing problems in image generation. StarGAN [47] enables image translation for only the part desired by the user using one model. This technology can be used to change only one feature, such as the hair color or skin color of people. Recently, StarGAN v2 [48] has been proposed, which experimented using animal face photographs as well as human faces.

Recently, a number of studies have been conducted on image synthesis and scene re-composition. For controlling objects from generated images, a model needs to perform three-dimensional (3D) inference, which is a non-trivial task for models that operate in 2D. To address this problem, GIRAFFE [49] incorporates a compositional 3D scene representation into the model to enable independent control of the positions, orientations, and perspectives of an object. With IMAGINE [50], diverse and high-quality images can be generated from a single image without having to train the generator. One of the main functions is to apply constraints on the shape and position of an object during the synthesis process and give users intuitive control over the generation process. This enables the generation of non-repeating, high-quality images. With GAN, another challenge in terms of control was the control of the color of the images. Inspired by a color transfer method based on class histograms, researchers at the Technical University of Munich and Google developed HistoGAN [51]. It leveraged the architecture of StyleGAN2 [46] by modifying the last two blocks of StyleGAN [44], thus incorporating the histogram features. In this way, by incorporating a target color histogram as part of the input to HistoGAN, the tones of the image can be controlled. The high computational cost of state-of-the-art networks such as StyleGAN2 has been an obstacle to the instantaneous and interactive generation of images for users. Today, a full generator takes approximately 3 s to render an image, which is too slow for a majority of computation purposes. To address this problem, AnycostGAN [52] was developed for interactive natural image editing. AnycostGAN enables user control to generate images in lower channel and resolution configurations in exchange for obtaining faster results at a lower computational cost. These controls allow users to generate visually similar images, potentially at five times faster speeds.

Our study is along the same lines of this research on image generation using GANs. We have generated a large synthetic dataset from the off-the-shelf model, and a game engine from the motion recognition field, and used it in machine learning. In addition, we applied skeletal loss to the traditional segmentation network to more effectively generate 2D characters.

3 Dataset

It is difficult to obtain a game sprite dataset for training from the sprites in games that have been released before because the sprite datasets in the existing games are small, have different styles for each game, and the amount of data is insufficient for machine learning. Therefore, we need a methodology for generating a large amount of data that can be used for game sprite generation. To address this problem, we have introduced real images in training in the game sprite field, because real images have large amounts of data, and various segmentation techniques are advanced. If a segmentation network trained with real images is used in the game sprite generation field, a large amount of training data can be stably obtained.

To generate the data images required in sprite generation, BodyPix [53] and OpenPose [54] were used. BodyPix shows stable step-by-step segmentation performance. Moreover, stored video

and image files can be analyzed as well as real-time images. We rendered body parts segmented from stored images using these techniques. Measurements can be made for up to 24 body parts.

OpenPose is a technique for extracting the poses of people from images. This technique can also extract poses from video and image files as real-time images. We have rendered the poses of people as long oval images by analyzing stored images. It is possible to specify the bone colors for each part, but we have indicated the bones of poses in one red color because the information on the body parts already exists in BodyPix.

Fig. 1 shows the data images used for result generation in this study. (a) shows the image of the original character. (b) shows the rendered image of the character segmented in 24 parts. (c) shows the rendered image of the character segmented in 10 parts. (d) shows the rendered image of the character segmented in three parts. (e) shows the image rendered in a single color without segmentation. The red lines in (b) to (e) indicate the skeletons corresponding to the character's pose. Tables 1–3 list the names of body parts segmented in 24, 10, and 3 segments. (L), (R), (F), and (B) denote left, right, front, and back, respectively.

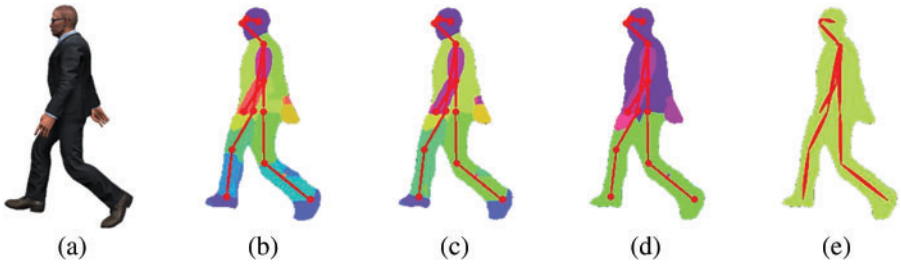

Figure 1: Data for generating sprite images for each segmentation resolution

Table 1: Body parts segmented in 24 parts

Face (L)	Face (R)	Torso (F)	Torso (B)
Hand (L)	Hand (R)	Foot (L)	Foot (R)
Upper Arm (L/F)	Upper Arm (L/B)	Upper Arm (R/F)	Upper Arm (R/B)
Lower Arm (L/F)	Lower Arm (L/B)	Lower Arm (R/F)	Lower Arm (R/B)
Upper Leg (L/F)	Upper Leg (L/B)	Upper Arm (R/F)	Upper Arm (R/B)
Lower Leg (L/F)	Lower Leg (L/B)	Lower Leg (R/F)	Lower Leg (R/B)

Table 2: Body parts segmented in 10 parts

Face	Hand (L)	Foot (L)	Arm (L)	Leg (L)
Torso	Hand (R)	Foot (R)	Arm (R)	Leg (R)

Table 3: Body parts segmented in 3 parts

Face	Upper Body	Lower Body

For our experiment, we generated data using 1000 continuous sprite images of a moving animation. Of the 1000 images, 500 images constitute walking animation, and the other 500

images constitute running animation. For each segmentation resolution, we used 8000 images as experimental data by applying a left-right reversal, and 90° rotations for data augmentation. Fig. 2 shows the images generated by data augmentation. (a) is the original image, (b) is the image rotated by 90°, (c) is the image rotated by 180°, and (d) is the image rotated by 270°.

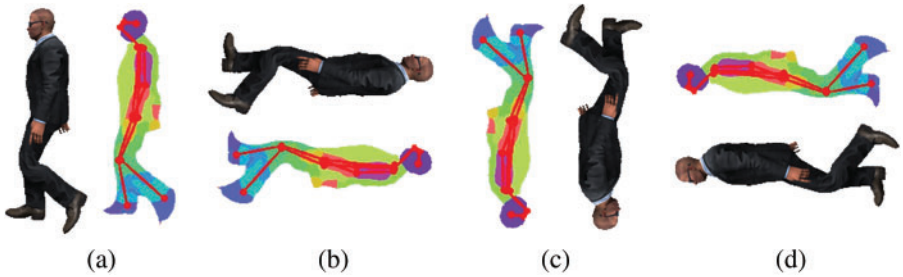

Figure 2: Data for generating sprite images for each segmentation resolution

4 Network

This study aims to generate high-resolution images—such as sprites—from simple images such as silhouettes. To achieve this aim through learning, simple image and sprite image pair data are required. To collect such labelled pair data in large amounts, an extremely high manual cost is required, but we could obtain them through a real human image segmentation network as described in Chapter 3.

Our goal is to provide a tool that can generate sprites by inputting a minimal number of images by user. To achieve this goal, the user inputs rules but the complexity must be low. We tried to find the optimal input data format while changing the number of segmentations and methodology.

We allowed users to distinguish the main parts of the character by color so as to generate high-resolution textures from simple images. It can be expected that stable transformations can be achieved to some degree because sufficient data were obtained through the inverse transformation process of the existing segmentation network. However, the problem is whether the network has capacity to respond to various poses input by user. The real action data are mainly composed of simple actions such as walking and running learned from image data. However, sprites in games have exaggerated poses and forms that are difficult to see in the real world. In particular, games have various combat-oriented actions and the network might be able to interpret and restore them.

For this, we used the UGATIT network model [55] that showed good performance in image-to-image translation as the baseline network. The UGATIT network features an auxiliary classifier and AdaIN; these two features of the UGATIT network have a significant advantage in shape modification, compared with existing image-to-image models. This study aims to develop a network that specializes in recognition of characters' pose.

Through existing research on user interfaces, we could confirm the stable image-to-image translation performance of this network [56]. It was generally 2D to 2D translation. Although game sprite images are 2D images, the head, neck, arm, and legs have depth values depending on the camera angle, and have priorities in rendering. If only the reverse segmentation transformation of simple images is performed, the main body parts will have difficulty in interpreting the depth values, thus generating unclear images. This problem can be solved if users provide

depth information in the tool, but this makes it difficult to configure an appropriate interface for inputting the depth value. To solve this problem, we have added a loss value for the difference between the input image and the skeleton image generated by the image-to-image network. We estimated the skeleton value of the generated image by adding the skeleton network to learning. Based on this, it was enabled to predict the body structure just by inputting simple lines.

Let $x_s \in X_s$ and $x_t \in X_t$ represent samples from the source and target domains, respectively. Furthermore, let $G_1(x_s)$ and $G_2(x_t)$ represent the translated source and target domains, respectively. Our proposed model consists of two generators ($G_1(x_s)$ and $G_2(x_t)$), two discriminators ($D_1(G_1(x_s))$ and $D_2(G_2(x_t))$), and two feature extractors $F_1(x_s, x_t)$. $G_1(x_s)$ creates an image that fits the target style based on the GAN framework, and $G_2(x_t)$ is used for cycle consistency. The discriminators D_1 and D_2 distinguish between real and fake translated images. The feature extractor F_1 provides a loss values to the CycleGAN framework to facilitate shape transformation. The final loss function of our model can be written as the loss of L_{total}.

$$\underset{G_1,G_2}{argmin}\ \underset{D_1,D_2}{max}\ L_{total}\big(G_1, G_2, D_1, D_2, F_1\big) \tag{1}$$

L_{total} comprises five loss terms: L_{lsgan}, L_{cycle}, $L_{identity}$, L_{cam}, and $L_{skeleton}$. The adversarial loss L_{lsgan} is employed to match the distribution of the translated images to the target image distribution. The cycle loss L_{cycle} is applied for a cycle consistency constraint to the generator. The identity loss $L_{identity}$ is used to ensure that the color distributions of the input and output images are similar. These three losses are calculated using G_1, G_2, D_1, and D_2 with the traditional GAN framework. These terms are described in detail in [55,57]. L_{cam} uses information from the auxiliary classifiers to determine the differences between two domains [58].

The additional feature loss $L_{skeleton}$ is the difference in 256-dimensional cosine similarity between the input and generated images. This value shows how similar the input and generated images are in terms of basic figure shape. Therefore, when creating a segmentation image, it induces the creation of similar images in terms of shape appearance as much as possible.

$L_{skeleton}$ loss can be applied differently according to the visual characteristics of each game by adjusting weight value α.

$$\begin{aligned} L_{total} =& L_{lsgan}(G_1, G_2, D_1, D_2) + L_{cycle}(G_1, G_2, D_1, D_2) \\ &+ L_{identity}(G1, G_2, D_1, D_2) + L_{cam}(G_1, G_2, D_1, D_2) \\ &+ L_{geometry}(F_1, G_1, G_2) \end{aligned} \tag{2}$$

$$L_{geometry}(F_1) = \alpha\, cosine\text{-}similarity(F_1, G_1, G_2) \tag{3}$$

5 Experiment

We tried to verify the usefulness of the proposed methodology by changing the number of segmentations and the number of skeleton bones. To that end, we defined sprites in 16 directions. This is the number of directions mainly used in general 2D games. For optimization, we set the maximum number of iterations to 200, the learning rate as 0.001, and its decay rate as 20% per five iterations. We used the SGD optimizer for training with the batch size = 8. The NVIDIA RTX Titan GPU was used for training. For one training, we constructed a dataset of 10,000 data of the 256×256 size. The ratio of Training:Val:Test data was set to 8:1:1. Under this configuration, the user input level required for sprite production was examined through various experiments.

Fig. 3 shows the result of training one segmentation with a dataset for which skeleton information was not used. The top image is the ground truth, the middle is an image generated by the network, and the bottom is an input image indicated in a single color. This experiment was performed to examine the stability of image generation from the segmentation of the proposed network. Experiments were performed while changing the network structure only with the same dataset and hyper parameter environment. It can be seen that the Pix2Pix technique cannot restore the image normally, but the proposed method has the ability to produce stable images in general.

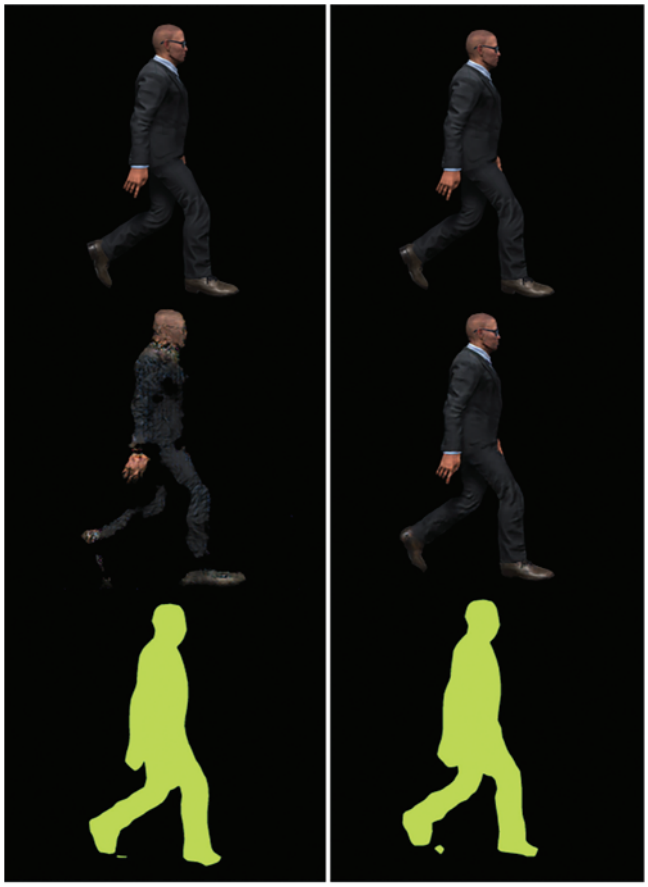

Figure 3: Result of image generation from a single segmentation image (a: Pix2Pix result, b: result of limited network, top: ground truth, middle: generated image, bottom: input energy)

Fig. 4 shows the result of the network trained by adding skeleton images to one segmentation. The skeletal loss was used during training. Overall, the accuracy of the image resolution improved slightly, and the depth values were more clearly discriminated. In the first and fourth images, even if the segmentation information was missing in the legs, the legs were still generated faintly. This result shows that adding the proposed skeletal loss has a complementary property in restoring the image from segmentation.

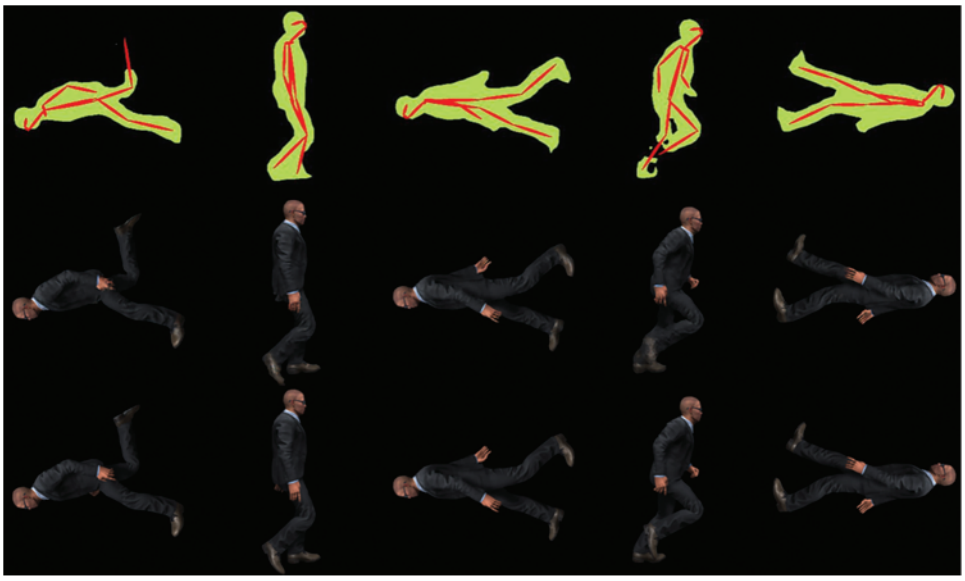

Figure 4: Result of images obtained by applying single segmentation + skeletal loss

Fig. 5 shows the result of adding skeletal loss to a three-segmentation map. It can be seen that the omitted parts in the leg are restored more accurately than in the second experiment. Furthermore, the 5th result image shows the generation of textures only in the part connected with the skeleton ignoring the incorrectly input segmentation value. This result shows that compared to single segmentation, multiple-segmentation input data is more helpful for the network in understanding the context.

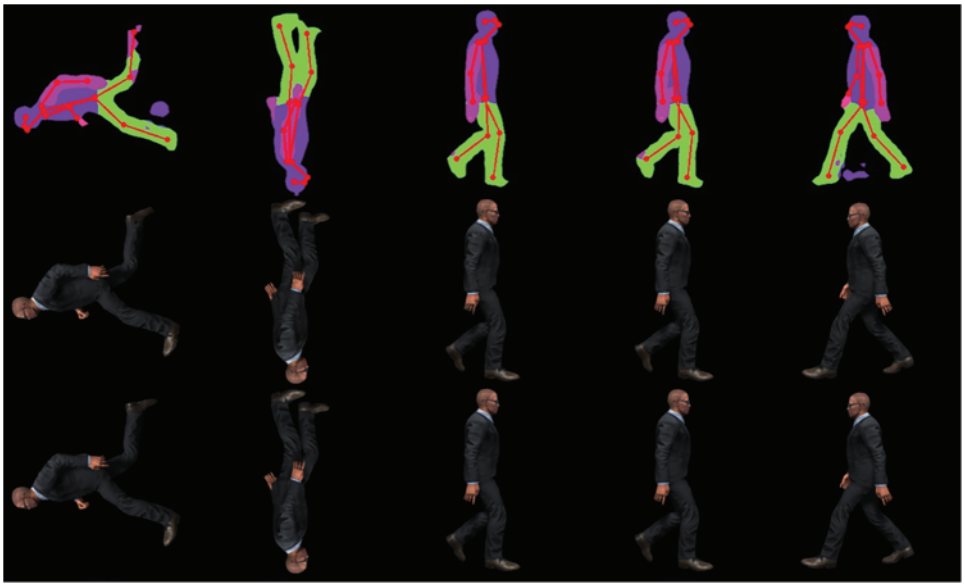

Figure 5: Result of images obtained by applying three segmentations + skeletal loss

Fig. 6 shows the result of training with a dataset that used 24 segmentations only without the skeleton information. The result shows the corresponding sprites are generated well even without skeleton data. This suggests that the structural information is understood by the segmentation colors. This confirms that if the user inputs many patches, accurate sprites can be generated as expected. However, one disadvantage is that the user's cognitive complexity increases because the number of colors to be input increases.

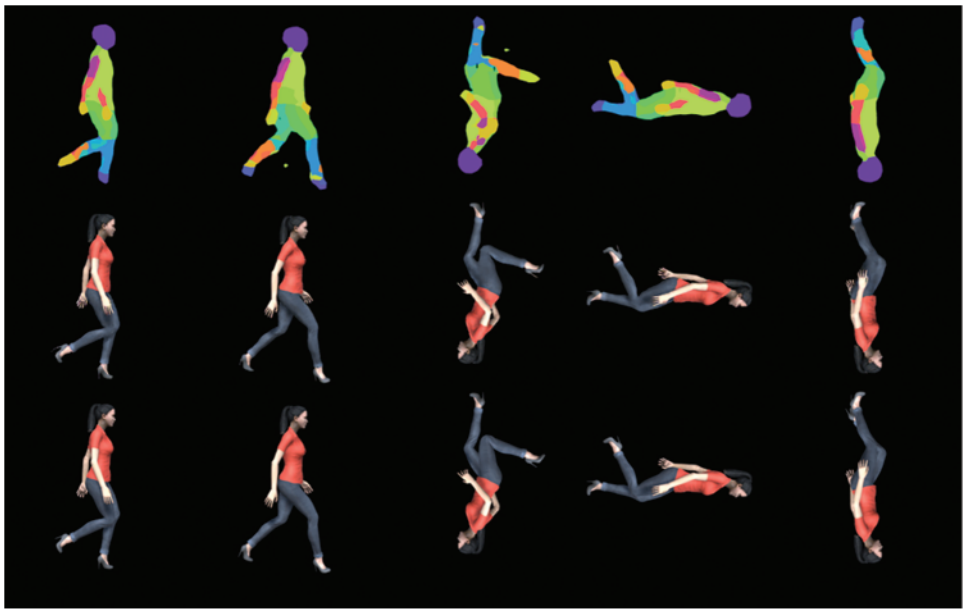

Figure 6: Result of images obtained by applying 24 segmentations only

Fig. 7 shows the resulting images when an image drawn randomly by the user is input to the network trained with single segmentation, three segmentations + skeleton data, 24 segmentations, and 24 segmentations + skeleton. It can be seen that the more segmentation maps and skeleton information are provided, the more accurate is the image generated. Since the input motions are not in the range of 16 directions, the network experiences many difficulties in generating such motions. When the resulting images of the network trained with 24 segmentations and the network trained with three segmentations + skeleton data are compared, the figure shape of the images generated by the network trained with three segmentations + skeleton data is more similar to the ground truth. This shows that simply increasing the number of segmentations does not improve the restoration result. The bottom images in Fig. 6 are the result of pixelization of the generated image using a pixel art converter [47]. The blurred result in the real image disappears by the pixelization, and the pixelized result shows a similar level to the result of a dot-based production technique mainly used in games. This shows that our result can be more stable when it has an art style in which a character with a real body shape is pixelized.

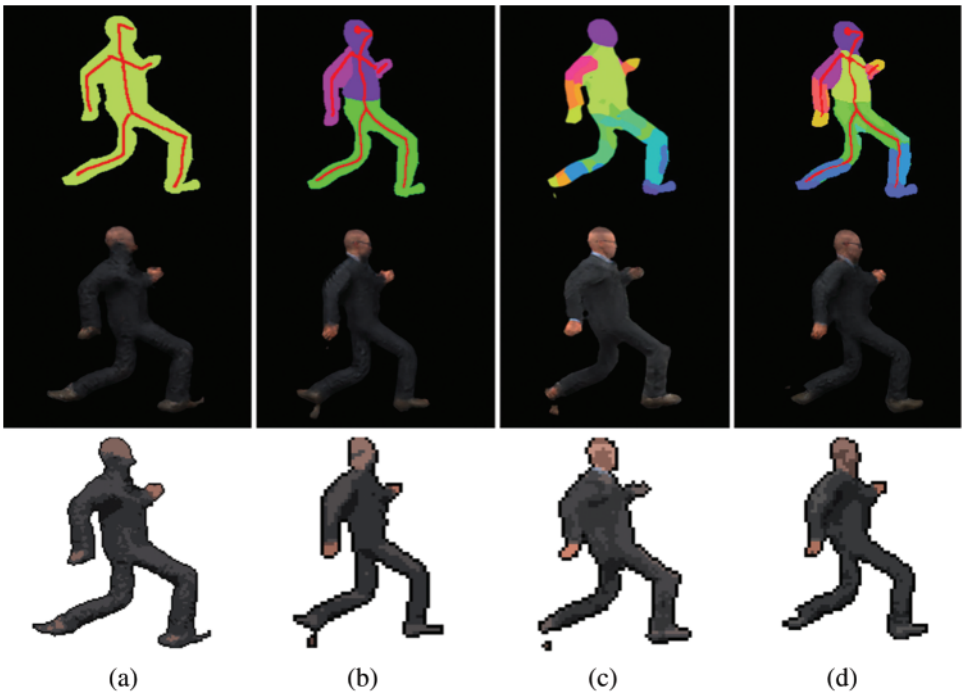

Figure 7: Result of images generated by segmentation map and skeleton data manually input by user (a) single segmentation + skeleton data, (b) three segmentations + skeleton data, (c) 24 segmentations, and (d) 24 segmentations + skeleton

Table 4 shows the general segmentation performance of the network. The pixel accuracy, mean accuracy, mean IU, and Frequency-Weighted IU were measured for the test set in four experiments. In general, more accurate results were generated when the number of segmentations was larger and the skeletal loss was added. In numerical values, the result of the 24 segmentations + skeletal loss shows a higher performance than the result of 24 segmentations only. However, the result of the three segmentations + skeletal loss shows a similar level of performance—although a little lower. If the performance is similar, it can be much easier and more intuitive to draw one skeleton rather than coloring many segments in terms of user's convenience. This experiment shows that skeletal loss can replace the input of many segmentations. The number of segmentations should be adjusted in accordance with the required level of output.

Table 4: Comparison of network performance by segmentation level and application of skeletal loss

	1 segmentation	3 segmentations + skeleton	24 segmentations	24 segmentations + skeleton
Pixel accuracy	0.49	0.59	0.69	0.72
Mean accuracy	0.44	0.63	0.66	0.71
Mean IU	0.42	0.66	0.62	0.65
Frequency Weighted IU	0.39	0.51	0.59	0.61

Fig. 8 shows the results when the user randomly omitted colors and skeletons. It can be seen that the proposed network can generate a result similar to the final output by inferring the corresponding result even if the user omitted some parts of the image by mistake. This effect improves user convenience because the final output can be produced similarly even if the segmentations and skeleton image input by the user are not accurate.

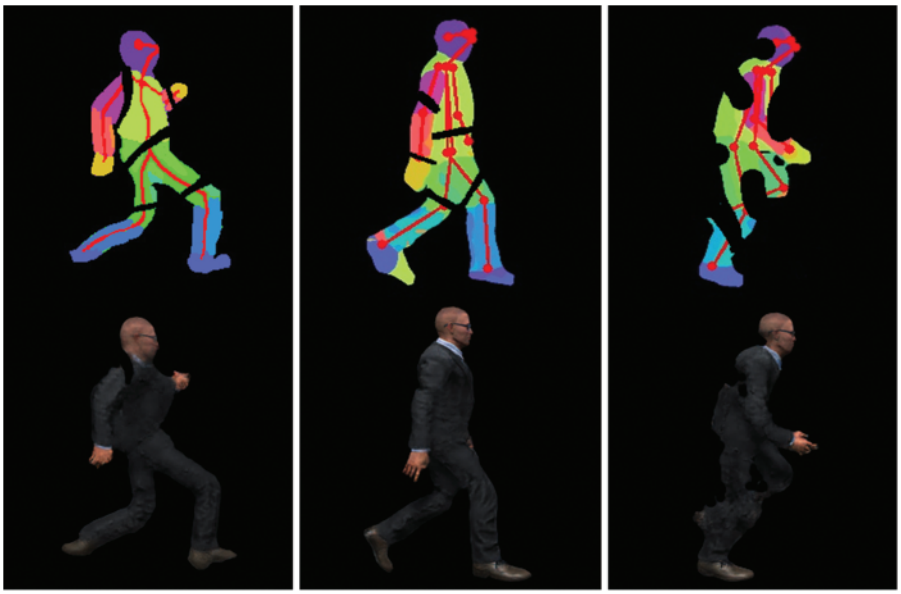

Figure 8: Experiment with 24 segmentation + 1 skeleton where colors and bones are omitted

Through the above experiments, we attempted to verify whether the proposed network can be actually used in the sprite production process in the game industry. In terms of user convenience, the proposed methodology gives a low burden for users because they only have to provide a small number of image patches and draw a single skeleton on the user interface. For the three-segmentation level, the user only needs to discriminate between the body, head, arms, and legs and only needs to add a simple line drawing above the segmentation image to create a skeleton. The generated image output could be generated stably for images within the specified 16 directions, but the poses randomly drawn by the user showed a low resolution in the generated images. This means that higher level data are required to interpret poses. However, even if it is a random pose image generated by our network, the final image can be generated just by adding a brush-up process by the artist based on the image. Therefore, the proposed input interface and network are expected to be introduced in the preprocessing stage for 2D sprite art production.

6 Conclusions

This study introduced a method for generating game sprites from simple image drawings using the supervised learning technique. For sprite generation, the user only needs to color the main parts with colors that can be perceived by the user in the trained network, and draw the bones of the character with a single color. Then the sprites of the shape drawn by the user are generated approximately. Our methodology generated a synthetic dataset from real photographs to increase the size of the insufficient game sprite data, and based on this, skeletal loss was used for the image-to-image translation network to generate the target sprite image. Through a

few experiments, we verified that the proposed network works stably even if segmentation colors were omitted, or if a mistake in skeleton input was made by the user. Although it may be difficult to obtain a pairing dataset that can be used for teacher learning, our method can perform the intended training with a dataset of a similar domain. The proposed methodology shows that the resource production productivity can be enhanced using the image-to-image translation technique in the game 2D sprite generation field that requires much manual work. In the future, we will conduct further research to upgrade the network structure that can respond to various multi-domains, and to improve the capacity to respond to a variety of shapes.

Despite its many advantages, the proposed method has some limitations. For example, when the training was performed only for walking motions but images of a squatting or jumping pose are generated, the generated images will appear very strange. In addition, the image data used in this study have a short interval between the animation frames for training (0.03 s), but if the interval between frames is increased to approximately 0.5 s, the pose between frames disappears, which may lead to lowered quality of the generated images. Furthermore, for user-defined skeletons, if the bone length is not properly adjusted, the quality of the generated images may be degraded.

Funding Statement: This work was supported by the National Research Foundation of Korea (NRF) grant funded by the Korean Government (MSIT) (Nos. NRF-2019R1A2C1002525, NRF-2020R1G1A1100125) and a research grant from Seoul Women's University (2020-0181).

Conflicts of Interest: The authors declare that they have no conflicts of interest to report regarding the present study.

References

1. Ghosh, A., Zhang, R., Dokania, P. K., Wang, O., Efros, A. A. et al. (2019). Interactive sketch & fill: Multiclass sketch-to-image translation. *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pp. 1171–1180. Seoul, Korea.
2. Xie, M., Li, C., Liu, X., Wong, T. T. (2020). Manga filling style conversion with screentone variational autoencoder. *ACM Transactions on Graphics, 39(6),* 1–15. DOI 10.1145/3414685.3417873.
3. Zheng, Z., Wang, C., Yu, Z., Wang, N., Zheng, H. et al. (2019). Unpaired photo-to-caricature translation on faces in the wild. *Neurocomputing, 355,* 71–81. DOI 10.1016/j.neucom.2019.04.032.
4. Goodfellow, I., Pouget-Abadie, J., Mirza, M., Xu, B., Warde-Farley, D. et al. (2014). Generative adversarial nets. *Advances in Neural Information Processing Systems, 27*. DOI 10.5555/2969033.2969125.
5. Joo, D., Kim, D., Kim, J. (2018). Generating a Fusion Image: One's Identity and Another's Shape. *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 1635–1643. Salt Lake City, UT, USA. DOI 10.1109/CVPR.2018.00176.
6. Iizuka, S., Simo-Serra, E., Ishikawa, H. (2017). Globally and locally consistent image completion. *ACM Transactions on Graphics, 36(4),* 1–14. DOI 10.1145/3072959.3073659.
7. Nie, W., Narodytska, N., Patel, A. (2018). Relgan: Relational generative adversarial networks for text generation. *International Conference on Learning Representations*, Vancouver, Canada.
8. Majurski, M., Manescu, P., Padi, S., Schaub, N., Hotaling, N. et al. (2019). Cell image segmentation using generative adversarial networks, transfer learning, and augmentations. *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops*, California, USA.
9. Zhang, R., Pfister, T., Li, J. (2019). Harmonic unpaired image-to-image translation. *International Conference on Learning Representations (Poster)*, New Orleans, Louisiana, USA.

10. Baumgartner, C. F., Koch, L. M., Tezcan, K. C., Ang, J. X., Konukoglu, E. (2018). Visual feature attribution using wasserstein gans. *Proceedings of 2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition*, vol. 1, pp. 8309–8319. Salt Lake City, UT, USA. DOI 10.1109/CVPR.2018.00867.
11. Kwon, G., Han, C., Kim, D. S. (2019). Generation of 3D brain mri using auto-encoding generative adversarial networks. *International Conference on Medical Image Computing and Computer-Assisted Intervention*, pp. 118–126. Shenzhen, China.
12. Ying, X., Guo, H., Ma, K., Wu, J., Weng, Z. et al. (2019). X2CT-GAN: Reconstructing CT from biplanar X-rays with generative adversarial networks. *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 10619–10628. California, USA.
13. Yang, Q., Yan, P., Zhang, Y., Yu, H., Shi, Y. et al. (2018). Low-dose ct image denoising using a generative adversarial network with wasserstein distance and perceptual loss. *IEEE Transactions on Medical Imaging, 37(6),* 1348–1357. DOI 10.1109/TMI.2018.2827462.
14. Yang, G., Yu, S., Dong, H., Slabaugh, G., Dragotti, P. L. et al. (2017). Dagan: Deep de-aliasing generative adversarial networks for fast compressed sensing mri reconstruction. *IEEE Transactions on Medical Imaging, 37(6),* 1310–1321. DOI 10.1109/TMI.42.
15. Xue, Y., Xu, T., Zhang, H., Long, L. R., Huang, X. (2018). Segan: Adversarial network with multi-scale l 1 loss for medical image segmentation. *Neuroinformatics, 16(3),* 383–392. DOI 10.1007/s12021-018-9377-x.
16. Zhu, X., Zhang, X., Zhang, X. Y., Xue, Z., Wang, L. (2019). A novel framework for semantic segmentation with generative adversarial network. *Journal of Visual Communication and Image Representation, 58,* 532–543. DOI 10.1016/j.jvcir.2018.11.020.
17. Souly, N., Spampinato, C., Shah, M. (2017). Semi supervised semantic segmentation using generative adversarial network. *Proceedings of the IEEE International Conference on Computer Vision*, pp. 5688–5696, Venice, Italy.
18. Luc, P., Couprie, C., Chintala, S., Verbeek, J. (2016). Semantic segmentation using adversarial networks. *NIPS Workshop on Adversarial Training (Poster)*. Barcelona, Spain.
19. Zhu, W., Xiang, X., Tran, T. D., Hager, G. D., Xie, X. (2018). Adversarial deep structured nets for mass segmentation from mammograms. *IEEE 15th International Symposium on Biomedical Imaging*, pp. 847–850. Wasington DC, USA.
20. Nazeri, K., Ng, E., Ebrahimi, M. (2018). Image colorization using generative adversarial networks. *International Conference on Articulated Motion and Deformable Objects*, pp. 85–94. Palma de Mallorca, Spain.
21. Liu, Y., Qin, Z., Luo, Z., Wang, H. (2017). Auto-painter: Cartoon image generation from sketch by using conditional generative adversarial networks. *Neurocomputing*, *311*, 78–87.
22. Azadi, S., Fisher, M., Kim, V. G., Wang, Z., Shechtman, E. et al. (2018). Multi-content gan for few-shot font style transfer. *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 7564–7573. Salt Lake City, USA.
23. Elgammal, A., Liu, B., Elhoseiny, M., Mazzone, M. (2017). Can: Creative adversarial networks, generating "art" by learning about styles and deviating from style norms. *International Conference on Computational Creativity*, pp. 96–103. Atlanta, GA.
24. Yang, H., Huang, D., Wang, Y., Jain, A. K. (2018). Learning face age progression: A pyramid architecture of gans. *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 31–39. Salt Lake City, USA.
25. Yang, H., Huang, D., Wang, Y., Jain, A. K. (2019). Learning continuous face age progression: A pyramid of gans. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, *43*, 499–515. DOI 10.1109/TPAMI.2019.2930985.
26. Zhang, Z., Song, Y., Qi, H. (2017). Age progression/regression by conditional adversarial autoencoder. *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 5810–5818. Hawaii, USA.
27. Wang, Y., Gonzalez-Garcia, A., van de Weijer, J., Herranz, L. (2019). Sdit: Scalable and diverse cross-domain image translation. *Proceedings of the 27th ACM International Conference on Multimedia*, pp. 1267–1276. Nice, France.

28. Radford, A., Metz, L., Chintala, S. (2015). Unsupervised representation learning with deep convolutional generative adversarial networks. *International Conference on Learning Representations (Poster)*, San Juan, Puerto Rico.
29. Berthelot, D., Schumm, T., Metz, L. (2017). Began: Boundary equilibrium generative adversarial networks. arXiv preprint arXiv:1703.10717.
30. Isola, P., Zhu, J. Y., Zhou, T., Efros, A. A. (2017). Image-to-image translation with conditional adversarial networks. *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 1125–1134. Honolulu, Hawaii.
31. Karras, T., Aila, T., Laine, S., Lehtinen, J. (2017). Progressive growing of gans for improved quality, stability, and variation. *International Conference on Learning Representations (Conference Track)*, Vancouver, BC, Canada.
32. Zhao, J., Mathieu, M., LeCun, Y. (2016). Energy-based generative adversarial network. *International Conference on Learning Representations (Conference Track)*, Toulon, France.
33. Shaham, T. R., Dekel, T., Michaeli, T. (2019). Singan: Learning a generative model from a single natural image. *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pp. 4570–4580. Seoul, Korea.
34. Wang, T. C., Liu, M. Y., Zhu, J. Y., Tao, A., Kautz, J. et al. (2018). High-resolution image synthesis and semantic manipulation with conditional gans. *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 8798–8807. Salt Lake City, UT, USA.
35. Park, T., Liu, M. Y., Wang, T. C., Zhu, J. Y. (2019). Semantic image synthesis with spatially-adaptive normalization. *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 2337–2346. Long Beach, CA, USA.
36. Sun, W., Wu, T. (2019). Image synthesis from reconfigurable layout and style. *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pp. 10531–10540. Seoul, Korea.
37. Zhao, B., Meng, L., Yin, W., Sigal, L. (2019). Image generation from layout. *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 8584–8593. Long Beach, CA, USA.
38. Ledig, C., Theis, L., Huszár, F., Caballero, J., Cunningham, A. et al. (2017). Photo-realistic single image super-resolution using a generative adversarial network. *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 4681–4690. Honolulu, HI, USA.
39. Dong, C., Loy, C. C., He, K., Tang, X. (2015). Image super-resolution using deep convolutional networks. *IEEE Transactions on Pattern Analysis and Machine Intelligence, 38(2),* 295–307. DOI 10.1109/TPAMI.2015.2439281.
40. Yu, J., Lin, Z., Yang, J., Shen, X., Lu, X. et al. (2018). Generative image inpainting with contextual attention. *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 5505–5514. Salt Lake City, Utah.
41. Yu, J., Lin, Z., Yang, J., Shen, X., Lu, X. et al. (2019). Free-form image inpainting with gated convolution. *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pp. 4471–4480. Seoul, Korea.
42. Jo, Y., Park, J. (2019). Sc-fegan: Face editing generative adversarial network with user's sketch and color. *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pp. 1745–1753. Seoul, Korea.
43. Nazeri, K., Ng, E., Joseph, T., Qureshi, F., Ebrahimi, M. (2019). Edgeconnect: Structure guided image inpainting using edge prediction. *Proceedings of the IEEE/CVF International Conference on Computer Vision Workshops*, Seoul, Korea.
44. Karras, T., Laine, S., Aila, T. (2019). A style-based generator architecture for generative adversarial networks. *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 4401–4410. Long Beach, CA, USA.
45. Huang, X., Belongie, S. (2017). Arbitrary style transfer in real-time with adaptive instance normalization. *Proceedings of the IEEE International Conference on Computer Vision*, pp. 1501–1510. Venice, Italy.
46. Karras, T., Laine, S., Aittala, M., Hellsten, J., Lehtinen, J. et al. (2020). Analyzing and improving the image quality of stylegan. *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 8110–8119. Seattle, WA, USA.

47. Choi, Y., Choi, M., Kim, M., Ha, J. W., Kim, S. et al. (2018). Stargan: Unified generative adversarial networks for multi-domain image-to-image translation. *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 8789–8797. Salt Lake City, Utah.
48. Choi, Y., Uh, Y., Yoo, J., Ha, J. W. (2020). Stargan v2: Diverse image synthesis for multiple domains. *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 8188–8197. Seattle, WA, USA.
49. Niemeyer, M., Geiger, A. (2021). Giraffe: Representing scenes as compositional generative neural feature fields. *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 11453–11464. Nashville, TN, USA.
50. Wang, P., Li, Y., Singh, K. K., Lu, J., Vasconcelos, N. (2021). Imagine: Image synthesis by image-guided model inversion. *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 3681–3690. Nashville, TN, USA.
51. Afifi, M., Brubaker, M. A., Brown, M. S. (2021). Histogan: Controlling colors of gan-generated and real images via color histograms. *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 7941–7950. Nashville, TN, USA.
52. Lin, J., Zhang, R., Ganz, F., Han, S., Zhu, J. Y. (2021). Anycost gans for interactive image synthesis and editing. *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 14986–14996. Nashville, TN, USA.
53. Zhu, T., Oved, D. (2019). Bodypix. https://github.com/tensorflow/tfjs-models/tree/master/body-pix#person-body-part-segmentation.
54. Cao, Z., Hidalgo, G., Simon, T., Wei, S. E., Sheikh, Y. (2019). Openpose: Realtime multi-person 2d pose estimation using part affinity fields. *IEEE Transactions on Pattern Analysis and Machine Intelligence, 43(1),* 172–186. DOI 10.1109/TPAMI.34.
55. Kim, J., Kim, M., Kang, H., Lee, K. (2019). U-Gat-it: Unsupervised generative attentional networks with adaptive layer-instance normalization for image-to-image translation. *International Conference on Learning Representations (Conference Track)*, Addis Ababa, Ethiopia.
56. Kang, S., Choi, J. -i. (2020). Instance segmentation method of user interface component of games. *Applied Sciences, 10(18),* 6502. DOI 10.3390/app10186502.
57. Zhu, J. Y., Park, T., Isola, P., Efros, A. A. (2017). Unpaired image-to-image translation using cycle-consistent adversarial networks. *Proceedings of the IEEE International Conference on Computer Vision*, pp. 2223–2232. Venice, Italy.
58. Selvaraju, R. R., Cogswell, M., Das, A., Vedantam, R., Parikh, D. et al. (2017). Grad-CAM: Visual explanations from deep networks via gradient-based localization. *Proceedings of the IEEE International Conference on Computer Vision*, pp. 618–626. Venice, Italy.