Generate 2D Full-body Anime Video From Single Anime Image And Real Human Video

Zhenwei Tan
Wuhan University, China

Abstract

We address the problem of motion transfer from real human to 2D anime characters. Given a video sequence of a human performing an action(e.g. dancing), and a single image of an anime character, we aim to generate a new video sequence of that anime character performing the same action. We adopt keypoint-based pose figure as intermediate representation and propose a three-stage pipeline to synthesize the output anime video step by step. In addition, we generate a new dataset specially used for motion transfer between real human and anime character.

1 Introduction

The traditional pipeline of 2D anime production contains laborious work of drawing keyframes of the anime according to the 'layout', a sketch indicating what the anime character is doing in every frame. For amateur anime lovers who want to produce their own anime shorts, it takes great effort to master extra skills and techniques even if they have designed their own anime character in a painting. Fortunately, fast developing generative models like GAN(Goodfellow et al., 2014), VAE(Pu et al., 2016) make it possible to generate plausible images and videos automatically from the given condition. Besides, synthesis and pose transfer works are hot in recent years and provide lots of plug-in modules. These all contributes to generate anime shorts without any knowledge of anime production, which give birth to this work.

In this work, we propose an approach to generate an anime video sequence automatically given a real human video sequence and an image of anime character. Figure 1 gives an example of our work. Left $Reference$ is the given anime character image. The bottom of right, which is marked as $RealHuman$, is the given real human video sequence. The upper of right, which is marked as $Anime$, is the generated anime video sequence. The real human video performs the motion that we wish to transfer to target anime character image. We namely making target anime character do what the real human do in the video. We separately train different phases of transferring and combined them together as a video-to-video translation in testing, enabling practitioners to use it as reference and amateurs to make anime on their own easily.

Copyright © 2019, Association for the Advancement of Artificial Intelligence (www.aaai.org). All rights reserved.

An image-to-image mapping should be discovered by the whole framework in order to transfer motion from real human to anime video frame by frame. Under this consideration, pairs of real human and anime frame which performs the same motion should be essential if we want to apply supervised learning. However, unfortunately, there are no ready-made datasets for learning this corresponding translation. What's more, we even cannot find available ready-made datasets for well-placed full-body anime character. The widely-used anime datasets is either for anime face(Jin et al., 2017a) or messy anime artworks with extremely complex perspectives and composition(Branwen, 2015), which makes the situation even worse. In order to push the framework focusing on the motion transfer, instead of image understanding, and prevent the training process from crashing because of the dirty data, we construct our own real-to-anime motion transfer dataset via 3D anime software MikuMikuDance. Given a real human video and an anime character, we construct a pipeline to automatically generate the corresponding anime video.

It has proved that multi-stage training achieves better performance and also interpretability than one-step training(Pan et al., 2017; Zhang et al., 2017a; Wang and Gupta, 2016; Yamamoto et al., 2018; Ohnishi et al., 2018) in image generation. Keypoint-based pose, which can record the motion signatures and abstract the identity of original input, is widely accepted as intermediate medium for image-to-image motion transfer. The pose estimation for real human is quite mature and there are many off-the-shelf pose detectors such as OpenPose(Cao et al., 2018). Therefore, we directly use them to extract pose figures from input real human video. To gap the difference of body size and body ratio between the real human in video and anime character in image, we also need anime character's pose figure for reference. However, pose estimation for real human and anime characters are not interlinked, which means we cannot directly apply off-the-shelf pose detectors on anime character. We only find one work(Khungurn and Chou, 2016) special for anime pose estimation but not open source and hence pre-train the state-of-the-art pose estimation work(Osokin, 2019) for anime character using augmented anime pose dataset(dragonmeteor, 2015)

Our contributions

- We construct a new dataset that pairs full-body real human

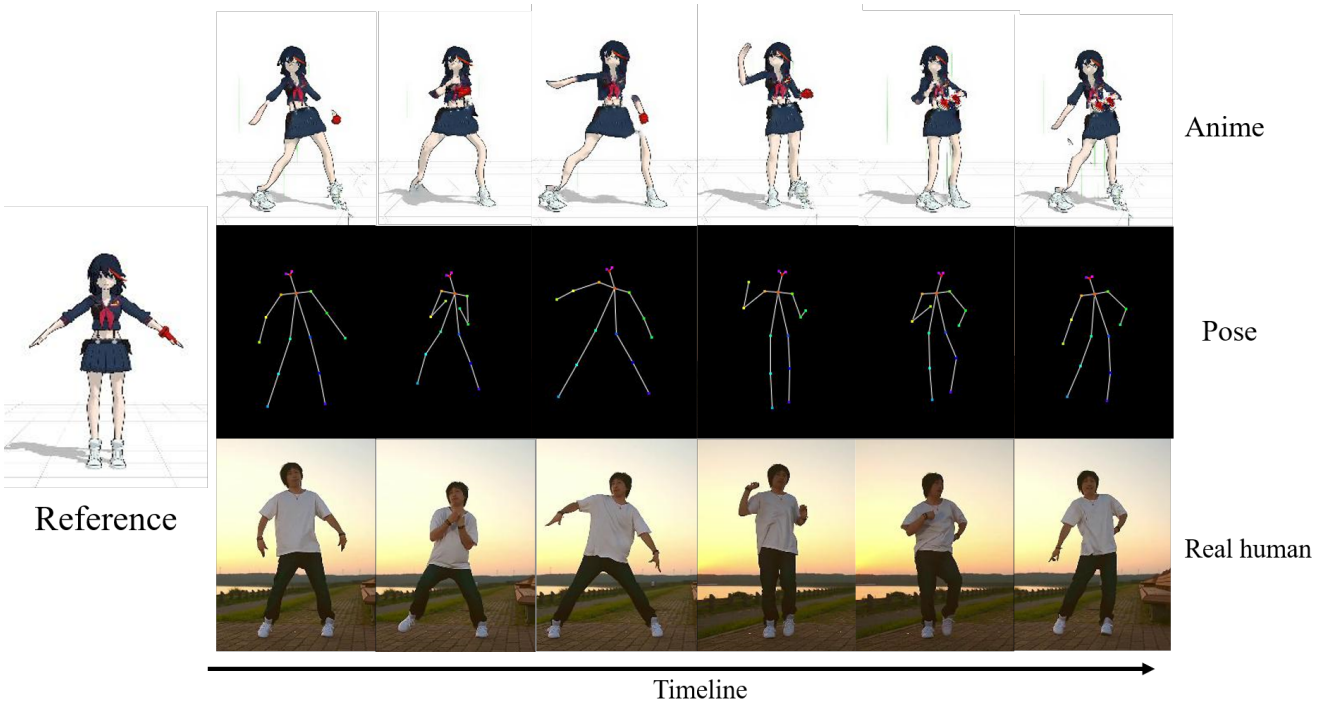


Figure 1: An example of our result. Left: $Reference$ represents our reference anime image. Right: $Realhuman$ represents the real human video sequence we input, $Pose$ represents the normalized keypoint-based pose, which is our intermediate result, and $Anime$ represents the anime video sequence we output

and anime character performing the same motion.

- We propose an approach about how to get an end-to-end model that transfers motion from real human video to the anime character in the 2D dimension.
- We dig out suitable state-of-the-art pose estimation and pose transfer work and retrain them using our anime dataset.

2 Related Work

2.1 Pose transfer for real human

Motion transfer focuses on creating a plausible image given the source image and arbitrary target pose. Because of the excellent performance shown by the generative adversarial network and its numerous variations for image translation(Odena, Olah, and Shlens, 2017; Zhu et al., 2017; Isola et al., 2017; Karras et al., 2017; Karras, Laine, and Aila, 2019), recent works apply deep learning on motion transfer and show impressive performance. Instead of relying on the multi-camera situation(Xu et al., 2011), these deep learning models can be trained on pairs of simple original images and target images and achieve plausible results(Ma et al., 2017, 2018; Qian et al., 2018; Balakrishnan et al., 2018; Zhu et al., 2019).

(Ma et al., 2017) construct a two-stage framework, which first combines the condition image and target pose into a coarse image of the person and then refines the blurry result in an adversarial way. (Balakrishnan et al., 2018) segments the complex task into small subtasks. It first does source image segmentation, dividing the original input's body into different part layers. Then it calculates the spatial transformation between source pose and target pose. Finally, a foreground and background synthesis applied parallelly to generate the final output. (Qian et al., 2018) propose a solution based on generating pose-normalized images for the scalability and generalizability issues. (Zhu et al., 2019) utilizes the attention mechanism and progressive style pipeline to address the challenging task of pose transfer. Very recently, (Chan et al., 2019) presents a simple video-based method for motion transfer between different people. It trains the model on a specific person with an encoder-decoder-like framework. The well-trained model can then generate videos of this specific person through poses extracted from other people's motion videos. All these works mentioned above are real human pose transfer, where the condition image is the same material as the output image, while in this work, we try to do a motion transfer with arbitrary anime character and real human pairs.

2.2 Pose estimation

To extract pose information from input and output image, state-of-the-art pose estimation techniques are widely used in motion transfer. For 2D pose estimation, (Wei et al., 2016) first introduces the keypoint heatmap as the representation for pose information. (Newell, Yang, and Deng, 2016) proposes the idea of learning keypoint heatmap with differ-

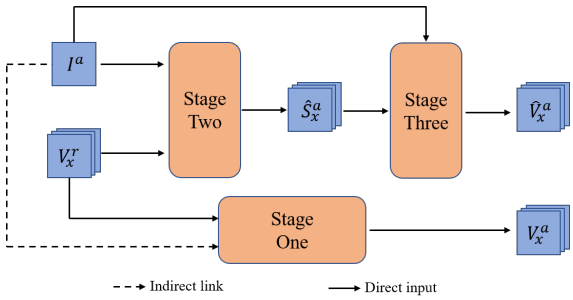


Figure 2: Overall pipeline of approach

ent scale receptive fields. The latest work GCCPM(Osokin, 2019) improves original convolutional pose machine architecture and achieves higher accuracy with less training data. Because of the friendly interface and off-the-shelf model, OpenPose(Cao et al., 2018) is plugged in many motion transfer works for real human pose estimation. In this work, we directly use the interface offered by OpenPose for real human pose estimation and pre-train GCCPM for anime pose estimation.

2.3 Anime generation

Anime production involves a laborious process of drawing keyframes and anime generative tasks aim to relieve the burden of anime practitioner or offer an easy approach for amateurs to create anime. As a subset of artwork generation, previous works mainly focused on anime face generation (tdrussell, 2016; Jin et al., 2017b) and style transfer(Zhang et al., 2017b, 2018). They either generate part of the anime or depend on a strong prerequisite such as finely-drawn sketch. By adding structure condition, (Hamada et al., 2018) improves progressiveGAN(Karras et al., 2017) to generate full-body anime character with arbitrary pose keypoints. However, their model is still trained for a specific anime character and thus the diversity of their results is limited. (Dvorožnák et al., 2018; Poursaeed et al., 2020) utilize a layered style-aware puppet to generate anime. They express pose transfer as a deformation of a layered 2.5D template mesh, and try to predict mesh deformations matching the template to a target image. The templates are all artificially drawn by artists, which means their work is not reproducible without dataset. In this work, we propose an approach that can generate motion video of various anime characters with one-time training and the dataset is all collected from the Internet.

3 Approach

In the following sections, we first introduce the problem formulation and notation for motion transfer between human and anime character and then propose a three-stage strategy that exploits structure and content information separately from real human video and anime image and finally generate the target anime video.

3.1 Definition of the problem

In this work, given an arbitrary real human video V_x^r with specific motion x, and a single 2D anime image I^a, we aim to synthesis a video sequence V_x^a in which anime character a performing motion x. Therefore, we can clearly formulate the objective of our project as following:

$$F(V_x^r, I^a) = \hat{V}_x^a \tag{1}$$

where F$(\cdot)$ is an abstraction function of our entire system. $\hat{V}_x^a$ denotes the predicted anime video. We tend to minimalize the difference between $\hat{V}_x^a$ and its ground truth anime video V_x^a.

During the training phase, we can have full access to (V_x^r, I^a, V_x^a) pairs, including intermediate parameters, and apply preprocess and post-process to the training data. Finally, we get the total well-trained models by optimizing the parameters by minimalizing the training loss. While during the testing phase, only (V_x^r, I^a) are available and the result $\hat{V}_x^a$ should be generated end-to-end.

However, ready-made (V_x^r, I^a, V_x^a) pairs, which serve as ground truth in this work, can be collected neither online nor from the real world. In detail, we don't have anime video performing the same motion x as the real videos we collected and it is impossible to manually pair videos from the network. We will address this problem in Stage One. Following other synthesis tasks introduce in 2.1, we also use the intermediate feature to bridge the input and output. We will introduce our feature extraction and normalization method in Stage Two. With the help of the end-to-end generative network, we combine the features prepared to synthesis the final result, which will be shown in Stage Three. The total pipeline of our approach is shown in fig3

3.2 Stage One: automatic data generation

As introduced in 3.1, we must first construct (V_x^r, I^a, V_x^a) pairs before developing the inference model but we cannot collect it directly from the real world. To fix this common dataset problem in synthetic tasks, the previous work either use extra information to generate artificial ground truth(Poursaeed et al., 2020) or control output quality of different aspects according to different original input (Pan et al., 2017; Yamamoto et al., 2018). In this work, instead of collecting 2D anime image I^a, we turn to the collection of 3D anime model M^a. We utilize the 3D anime model M^a to generate 2D anime video V_x^a of the collected real human counterpart V_x^r. According to previous 2D anime generation works(Khungurn and Chou, 2016; Poursaeed et al., 2020), extracting low-dimension information from its high-dimension version proves to be a good dataset construction strategy. By dimension reduction, 2D image we need in the training phase of Eq(1) can be easily obtained with a fixed camera Cam_0 without any loss.

$$DimReduction(M^a, Cam_0) = I^a, \tag{2}$$

Extra 3D information offered by 3D anime model M^a enables us to arbitrarily change its pose, which can be used to pair it with any real human pose. With 3D modeling software that takes in 3D anime model M^a and motion file representing motion x over time, anime video ground truth V_x^a can be generated. We choose 3D modeling software MikuMikuDance(denoted as MMD) for data generation because

following OpenMMD(Luo, 2016) pipeline, we can directly convert real human video V_x^r to '.vmd' file(video motion document)Vmd_x^a of anime model M^a instead of manually configuring motion files.

$$OpenMMD(V_x^r, M^a) = Vmd_x^a, \tag{3}$$

$$MMD(Vmd_x^a, M^a) = V_x^a \tag{4}$$

where $OpenMMD()$ represents the integrated automatic processing generating video motion file, $MMD()$ represents the automatic processing of MMD manipulated by script, VMD_x^a represents the '.vmd' file which contains modified motion information according to V_x^a and M^a.

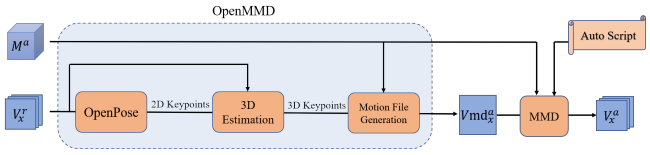


Figure 3: Pipeline of Stage One: automatic data generation

In detail, for $OpenMMD()$, the processing pipeline can be divided to four steps. 1) Use OpenPose API to do pose estimation over real human video V_x^r frame by frame and saved as keypoints JSON files. For smoothing intermediate results as a video, we additionally apply a Savitzky-Golay filter to output keypoints of all frames. 2) Combine all the keypoints JSON files to a continuous sequence with strong baselines for 3D human pose estimation, proposed in (Martinez et al., 2017). 3) Estimation of depth for objects, backgrounds and the moving person in the video using FCRN(Laina et al., 2016) 4) Combine the formatted results to 'vmd' file VMD_x^a, which can be directly fed to MMD for generating anime videos, originally proposed by (Tome, Russell, and Agapito, 2017). In the original open source project OpenMMD, the aforementioned four steps are loosely combined and executed separately. In this work, we integrate them and operate end-to-end inference.

For $MMD()$ processing, it is impossible to manually set parameters, load 3D model and motion file and operate the software to generate videos one by one. Therefore, we write a automation script based on winAppDriver, a service supporting Selenium-like UI test automation on Windows applications. Given the anime model M^a saved in file and motion file Vmd_x^a, tedious data generation operations can be done automatically.

So far, we generate the (V_x^r, V_x^a) pairs for training phase automatically and all these processing can be easily completed on personal laptop. Stage One can be formulated as the following equation:

$$Stage1(V_x^r, M^a) = MMD(OpenMMD(V_x^r, M^a), M^a) \tag{5}$$

$$= V_x^a \tag{6}$$

3.3 Stage Two: structure feature generation and normalization

Structure feature Generation As introduced in 2.1, the intermediate feature is frequently used in synthesis tasks for the segmenting the task to different stages, which makes synthesis more reasonable and effective. In this work, we actually want to extract structure features from real human video V_x^r and texture feature from anime image I^a and then combine these features together. The structure feature represents the motion x for each video frame and the texture feature represents the specific anime character a.

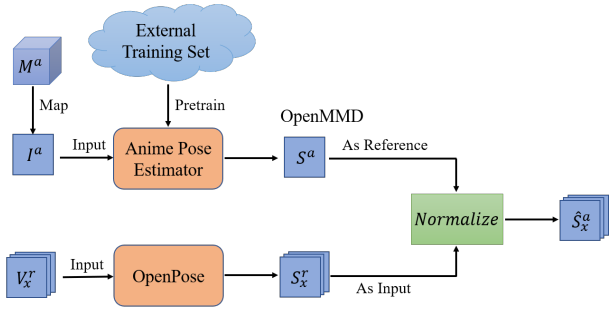


Figure 4: Pipeline of Stage Two: structure feature generation and normalization

We use widely-accepted keypoint-based pose to represent structure feature. Here structure feature S_x^a denotes keypoints maps of anime character a performing motion x, and S_x^r denotes keypoints maps of real person performing the same motion x. In order to extract structure feature from original input video V_x^r and V_x^a, we leverage two pose estimation model. Because pose estimation for single real human has achieve great performance, we simply adopt OpenPose interface to estimate real human pose of frames extracted from video V_x^r, as previous works did.

$$P_r(V_x^r) = S_x^r \tag{7}$$

where $P_r()$ denotes the OpenPose estimation inference function of our pre-trained model with the same smoothing operation in Stage One

For anime pose estimation, we find that it performs badly if we directly apply or fine-tune on real human pose estimation. The feature extracted from real human and anime is quite different. Therefore, we pre-train state-of-the-art pose estimation model(Osokin, 2019) through external anime pose training set(dragonmeteor, 2015). We choose this pose estimation model because it shows good performance training on small datasets and our anime pose training samples are extremely insufficient even with data augmentation. The pre-trained anime pose estimation model will be used in the part of normalization.

Structure Normalization Even in real human videos, there will be different camera positions and shooting angles and different limb proportions of people. These differences can only be enlarged due to the different identities of anime and real human video. Besides, different anime characters can vary greatly in body proportion. Therefore, when transferring motion between anime and real human, it is essential to apply structure normalization on structure information S_x^r so that it appear in accordance with the target anime's body, namely minimizing the difference between S_x^r and S_x^a

In testing phase, we have no access to the ground truth V_x^a and thus have no idea about the ground truth S_x^a. Therefore, we must make full use of the given 2D anime image I^a to analyze the anime character's body proportion. Following (Balakrishnan et al., 2018; Chan et al., 2019), we apply mathematics transformation based on keypoints maps of real human and anime character, which normally implies its body proportion. We apply our anime pose estimation model on 2D anime image I^a:

$$P_a(I^a) = S^a, \tag{8}$$

where $P_a()$ denotes our pre-trained anime pose estimation model, S^a denotes structure feature of anime character a with unknown pose.

There will be two situations that we can perfectly normalize S_x^a based on the real human frame from V_x^r or externally offered and the 2D anime Image I^a. 1)They both properly show their joints distance of 3D space, e.g. they all open their arms and face the camera. 2) They perform the same pose. Actually we can easily satisfy either of them in real testing because we control the source of S_x^r and I^a. We sequentially scale the shoulders, hips, arms/legs/head in accordance with the proportional relationship between real human and anime character. Noted that 'sequentially' means the scale center is their parent joint and we choose the neck keypoint as the origin center. The normalization processing can be formulated as:

$$N(S_x^r, P_a(I^a)) = \hat{S}_x^a, \tag{9}$$

where $N()$ abstracts the normalization processing. $\hat{S}_x^a$ denotes our predicted structure feature of anime character a performing x after normalization and we try to minimize the difference between $\hat{S}_x^a$ and S_x^a.

In stage Two, input the ground truth pair (V_x^r, V_x^a), we generate the predicted structure feature $\hat{S}_x^a$ of the target anime video V_x^a. Stage Two can be formulated as the following equation:

$$Stage2(V_x^r, I^a) = N(P_r(V_x^r), P_a(I^a))) = \hat{S}_x^a \tag{10}$$

3.4 Stage Three: anime video generation by pose transfer

Because of performance of GANs applied in generation tasks prove outstanding in recent works, especially synthesis and style transfer tasks, our video synthesis method use a modular generative neural network presented in (Zhu et al., 2019) designed for transferring the pose of a given person to a target pose. This model comprises a sequence of Pose-Attentional Transfer Blocks that each transfers certain regions it attends to and is trained using a progressive strategy that proves working well on high-resolution image generation tasks. The texture feature from I^a can be extracted implicitly and combined with the normalized structure feature $\hat{S}_x^a$ from Stage Two.

It is ideal to train on V_x^a using corresponding structure feature S_x^a generated from anime pose estimation. However, the performance of anime pose estimation is unstable due to the insufficient dataset, which causes strong noise if we directly use pose estimation results of V_x^a served as the training data. This may cause the training phase crash. As a compromise, we intend to set normalized S_x^r, namely $\hat{S}_x^a$ from Stage Two, as S_x^a, according to M^a, in the training phase of Stage Three since they perform the same motion x. Because we have initial coordinates of joints for each M^a, it is easy to obtain the correct limb proportion. We also manually select the frame in the real human video that can show the limb proportion and calculate it. It is not time-consuming but achieves acceptable results which can serve as the training set.

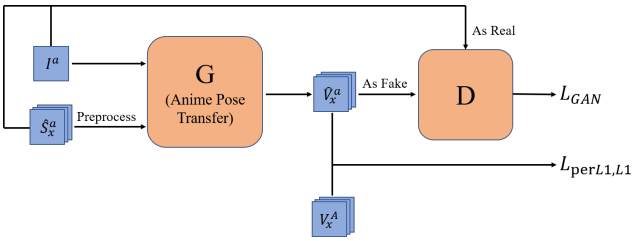


Figure 5: Pipeline of Stage Three: anime video generation by pose transfer

Besides, We added an additional part to the regularization of Stage Three. We find that the head of anime generated by MMD is sometimes in the wrong direction. To fix this defect, we use our anime pose estimation results on V_x^a as an auxiliary. Although anime pose estimation for the whole keypoints map is unstable, the detection of the neck and head is always right. Based on this fact, we rotate the head and eye keypoints of S_x^r to align with the direction of those in S_x^a. After all rotation and scaling finished, we align and crop the V_x^r and $\hat{S}_x^a$ to increase the character's share of the entire screen and thus reduce the difficulty of training.

In Stage Three, input structure feature $\hat{S}_x^a$ and single anime image I^a, we generate the predicted anime video $\hat{V}_x^a$ frame by frame, which is our overall target. Stage Three can be formulated as the following equation:

$$Stage3(\hat{S}_x^a, I^a) = \hat{V}_x^a \tag{11}$$

3.5 Loss Functions

Throughout the entire pipeline, the only parts that need to be trained are anime pose estimation and pose transfer. Next we discuss the loss function of these two parts.

Anime Pose Estimation Except using special anime pose dataset, we just follow the original training method of (Osokin, 2019). We also preprocess the dataset to keypoint heatmaps and for its regression the following loss function are used:

$$L_1 = \frac{1}{N}\sum_{n=1}^{N}\sum_{x,y}(G_n(x,y) - P_n(x,y))^2 \tag{12}$$

where N is a number of keypoint heatmaps and Gn(x, y), Pn(x, y) represent the ground truth and predicted n-th heatmap value at (x, y) pixel location. Such loss func-

tion is computed for each stage. To measure accuracy PCKh(Andriluka et al., 2014) metric is used.

Anime Pose Transfer In original GAN setup, the generator network G plays a mini-max game against discriminator D. The generator try its best to synthesize plausible images to deceive the discriminator D to judge it as 'real' images(ground truth) by learning differences from ground truth data. On the other hand, he discriminator D must discern between "real" (ground truth) images and "fake" images produced by the generator. The two networks are trained in turn and make each other better. Through this adversarial training, the results produced by the generator will be realistic and detailed. In this work, during train phase, the input are two frames (F_c^a, F_i^a) sampled from the same anime video V_x^a and F_i^a's corresponding keypoints maps $\hat{S}_i^a$ generated in Stage Two. In this way, we train a model that can synthesis anime image given a reference anime image and target pose keypoints map. We formulate our adversarial loss as:

$$L_{GAN} = \mathbb{E}_{\hat{S_i^a} \in \hat{S_x^a}, (F_c^a, F_i^a) \in V_x^a} \tag{13}$$

$$log[(D_A(F_c^a, F_i^a) \cdot D_S(S_i^a, F_i^a)] \tag{14}$$

$$+ \mathbb{E}_{\hat{S_i^a} \in \hat{S_x^a}, F_c^a \in V_x^a, \hat{F_i^a} \in \hat{V_x^a}} \tag{15}$$

$$log[((1 - D_A(F_c^a, \hat{F_i^a})) \cdot (1 - D_S(S_i^a, \hat{F_i^a}))] \tag{16}$$

where D_A denotes the discriminator responsible for texture feature consistence, D_S denotes the discriminator responsible for structure feature consistence, $\hat{S_i^a}, F_c^a, F_i^a, \hat{F_i^a}$ denotes the anime pose, anime condition frame, real anime frame, and fake anime frame. These frames belong to the real anime video distribution or fake anime video distribution.

We also combine the perceptual L1 loss and pixel-wise L1 loss of $(\hat{F_i^a}, F_i^a)$ to reduce distortions and gain pixel-level supervision. Therefore, the total loss function can be following:

$$L_2 = L_{GAN} + L_{perL1} + L_{L1} \tag{17}$$

4 Experiments

4.1 Dataset

For pose estimation, we use off-the-shelf anime pose dataset from GitHub(dragonmeteor, 2015). However, this dataset only contains $2,000$ labeled examples, which is far from enough. Therefore, we use data augmentation to increase the available training samples to $11,790$.

For pose transfer, we collect 20 high-resolution real human dancing videos from video websites. We require these videos to have no camera shake. The average duration of videos is over three minutes with a frame rate of 30. We select and clip these origin videos into 201 HR clips. Besides, we collect 61 3D anime character models in MMD from 'DeviantArt' and 'bowlroll.net'. We extracted the initial coordinates of each joint and saved them in the NumPy file. These real human video clips and anime character models can be paired and generate a lot of data. To control the amount and reduce redundancy, when generating the pairs in Stage one, we sample one every six frames and randomly skip some anime characters for certain video clips. After stage one, we extract $79,829$ frames from (V_x^r, V_x^a) pairs. By arranging and combining these frames as the source and target frame, for each anime character, we get about $400,000$ samples that can be directly used in pose transfer, which in format of (original frame, target frame, original keypoint map, target keypoint map).

4.2 Setting

For pose estimation, we basically follow the original setting of (Osokin, 2019). We initialize the parameters of shallow layers by pre-trained mobileNets(Howard et al., 2017) to ease the training and reset the keypoints number to fit our training data. We use Adam as optimizer and set $initiallr = 4e-5, weight_decay = 0.0005, batch_size = 32$. The training eventually overfits due to the small dataset, so we truncate the training in epoch 47.

For pose transfer, we modify from the official training setting for dataset DeepFashion(Liu et al., 2016) because its picture composition is similar to our training samples. We use a cropped image size of $(192, 256)$ and optimizer Adam with $initiallr = 0.0002$. We try to train the model using different datasets and train each with at least 350 epochs because the original work trains 700 epochs . It takes about 11 minutes for each epoch with $batch_size = 8$ on one Nvidia P100.

4.3 Feasibility Study

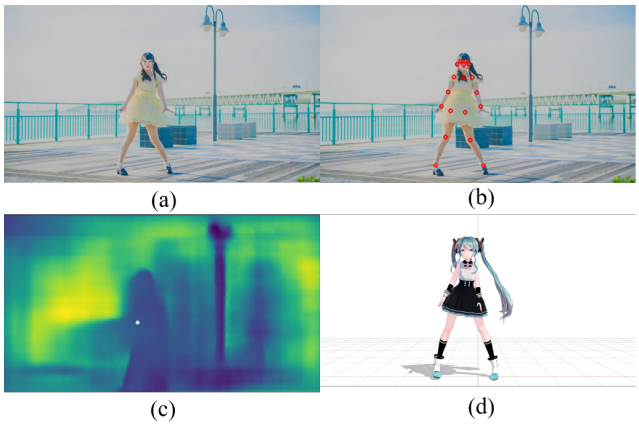


Figure 6: Generate the ground truth V_x^a from input V_x^r.Here (a) represents one frame from V_x^a, (b) represents the estimation results of keypoint-based pose, (c) represents the results of depth detection, (d) represents the generated ground truth V_x^r from MMD.

Figure 1 shows the overall result in a certain frame. In this section, we sequentially verify the feasibility with visualized results of the three stages in the entire pipeline. For Stage three, we only give part of the model verification because of the limited time and computing resources.

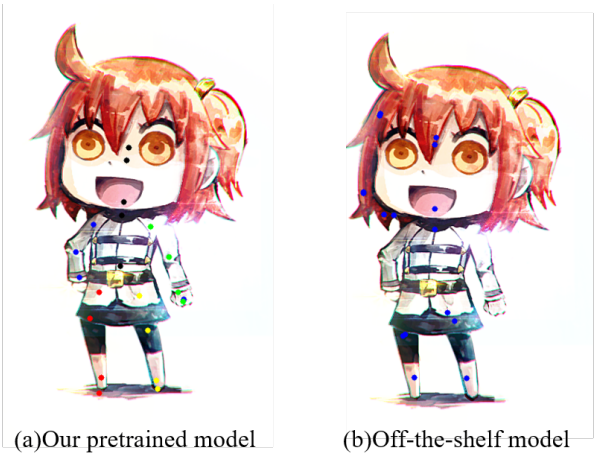
Stage One In stage One, we construct a end-to-end pipeline that generate (V_x^r, V_x^a) pairs as ground truth, uti-

lizing 3D information from M^a collected. Figure 6 shows the generation pipeline step by step.

We can see that even if the artificial ground truth is not perfect, it successfully transfers the motion x performed by a real human to the anime character, in high-resolution and without deformation, which is enough to serve as the ground truth in this work.

Stage Two In Stage Two, we generate the keypoint-based pose as structure feature and apply normalization on structure feature extracted from real human video in accordance with the given anime character. Here we demonstrate the performance of our anime pose estimation model and normalization process.

By comparing with results of directly applying real human pose estimation on anime characters, we prove the necessity of training anime pose estimation model from scratch, as shown in Figure 7. Obviously, our anime pose estimation model performs better than off-the-shelf real human pose estimation model on anime dataset.



(a)Our pretrained model (b)Off-the-shelf model

Figure 7: Comparison of real human and anime pose estimation results on anime characters. Dots on figure visualizes the result of pose estimation

We also try to fine-tune on pre-trained real human pose estimation model. Table 1 shows the losses of anime pose estimation training on the same dataset with different training settings. It demonstrates that the low-level feature of real people and anime characters cannot be shared and thus we can only train anime pose estimation from scratch.

Normalization also plays an important role in our work. Figure 8 visualizes the normalization processing. If we directly set S_x^r to $\hat{S}_x^a$ and then show the keypoint-based pose in its counterpart V_x^a, we will get the result on the left, which is a disaster for the training phase. After normalized processing, even though we don't get the perfect pose estimation of V_x^a, it is roughly right and enough for training in Stage Three.

Stage Three In Stage Three, given $\hat{S}_x^a$ and I^a, we use pose transfer to synthesis the final result V_x^a frame by frame. We first train the model on single anime character and then generalize it to multiply anime characters.

Table 1: Performance of pose estimation on different initialization. 'real19' means finetuning on 19th epoch of real huamn pose estimation. 'mix' means training on both real human and anime data.

Model Name	PCKh
real19	32.05
real69	26.2
real89	27.7
mix	27.48
mobileNet	**17.65**

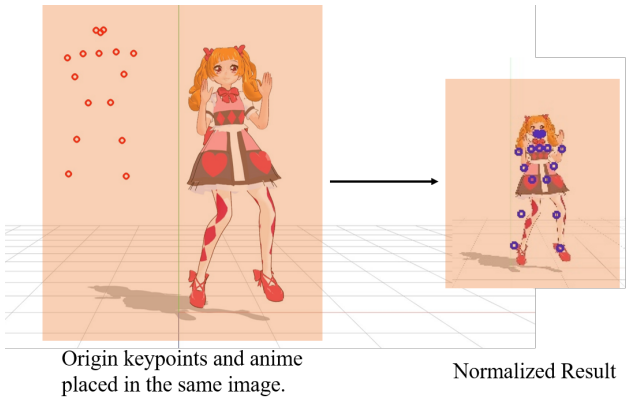


Figure 8: Normalization processing.

We pair two different poses of the same anime character from the database. For example, if anime character a has 100 different pose sampled from original video, then we will have $100(100-1)$ pairs for training because the pose transfer has direction. To reduce redundancy, we randomly eliminate some samples and save the preprocessed training data as NumPy file to avoid repeat processing, which speeds up the long training phase.

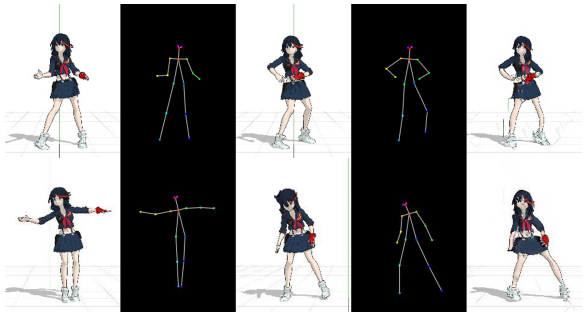


Figure 9: Pose transfer training on single anime character. From left to right: reference image I^a, structure of I^a, the ground truth from V_x^a, target structure from $\hat{S}_x^a$, predicted result from $\hat{S}_x^a$.

Figure 9 visualize the results of training on the same anime character. We can figure out that even the given keypoint-based pose is not perfect, the network learns to

fit the given pose. It works quite well for those facing the camera and having no crossover of limbs. Although some mistakes made in cases more difficult to learn, it still grasp the overall structure. It may be solved by better datasets and more training time on a larger batch size.

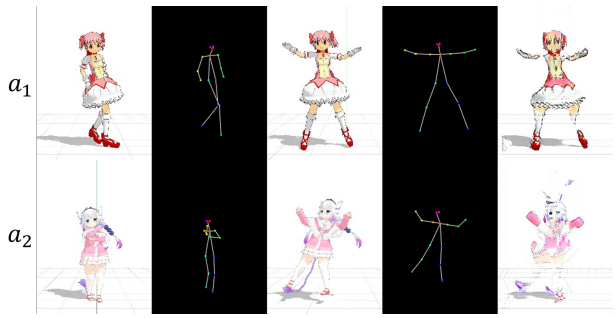


Figure 10: Pose transfer training on multi anime characters with 150 epochs. a_1 and a_2 denotes different anime characters. From left to right: reference image I^a, structure of I^a, the ground truth from V_x^a, target structure from $\hat{S}_x^a$, predicted result from $\hat{S}_x^a$.

Figure 10 visualizes the results of training on multiply anime characters at one time and testing on the anime characters we train. We only train the model with 2 and 4 anime characters because of the time and hardware limitation and Figure 10 is not the final results, which only trains about 150 epochs. We can figure out that the network is able to recognize different anime characters instead of generating a mixture of them. Different performance of different anime characters may originating from the complexity of the character reference picture. You can clear see that costumes and features of anime character a_2, which has tail and horn, are more complicated than a_1 and thus it achieve worse performance.

We also try to train on multiply anime characters but testing on unknown anime characters, getting the general model. In our original idea, if the network is trained on enough anime characters and can operating transfer on them separately after recognition, which has been proved by Figure 10, we have reason to infer that the network can extract universal low-level features of anime and can handle any given anime character. Figure 11 shows the results of testing on unknown anime characters using the model that is trained on only 2 anime characters. The results show that the model only tries to find one of the most similar anime characters, which it learns, as texture information. We speculate that lack of enough anime characters in training degrades the generator to synthesis specific kinds of texture instead of mining the input anime images. We now try to train on 20 anime characters but the training is still ongoing.

5 Conclusions and Future Works

This work focuses on transferring motion from full-body real human to anime character in 2D space. To resolve these, we construct a dataset for real-to-anime pose transfer and propose a three-stage pipeline for generating the final result

Figure 11: Testing on unknown anime characters using multi-anime pose transfer. From left to right: reference image I^a, structure of I^a, the ground truth from V_x^a, target structure from $\hat{S}_x^a$, predicted result from $\hat{S}_x^a$.

step by step. We make full use of available state-of-the-art works in fields of pose estimation and pose transfer. However, because of the limited time and lack of computing resources, we now only partially verified the model with limited anime characters and give inference to the wider conclusion. Better performance of anime pose estimation can be achieved if we have more labelled training data. Besides, fine-tuning the hyper-parameters can also contribute the results. We will dig deeper into these particular issues in future works.

References

Andriluka, M.; Pishchulin, L.; Gehler, P.; and Schiele, B. 2014. 2d human pose estimation: New benchmark and state of the art analysis. In *Proceedings of the IEEE Conference on computer Vision and Pattern Recognition*, 3686–3693.

Balakrishnan, G.; Zhao, A.; Dalca, A. V.; Durand, F.; and Guttag, J. 2018. Synthesizing images of humans in unseen poses. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 8340–8348.

Branwen, G. 2015. Danbooru2019: A large-scale crowdsourced and tagged anime illustration dataset.

Cao, Z.; Hidalgo, G.; Simon, T.; Wei, S.-E.; and Sheikh, Y. 2018. Openpose: realtime multi-person 2d pose estimation using part affinity fields. *arXiv preprint arXiv:1812.08008*.

Chan, C.; Ginosar, S.; Zhou, T.; and Efros, A. A. 2019. Everybody dance now. In *Proceedings of the IEEE International Conference on Computer Vision*, 5933–5942.

dragonmeteor. 2015. Anime Drawings Dataset. `https://github.com/dragonmeteor/AnimeDrawingsDataset/`.

Dvorožnák, M.; Li, W.; Kim, V. G.; and Sỳkora, D. 2018. Toonsynth: example-based synthesis of hand-colored cartoon animations. *ACM Transactions on Graphics (TOG)* 37(4):167.

Goodfellow, I.; Pouget-Abadie, J.; Mirza, M.; Xu, B.; Warde-Farley, D.; Ozair, S.; Courville, A.; and Bengio, Y. 2014. Generative adversarial nets. In *Advances in neural information processing systems*, 2672–2680.

Hamada, K.; Tachibana, K.; Li, T.; Honda, H.; and Uchida, Y. 2018. Full-body high-resolution anime generation with progressive structure-conditional generative adversarial networks. In *Proceedings of the European Conference on Computer Vision (ECCV)*, 0–0.

Howard, A. G.; Zhu, M.; Chen, B.; Kalenichenko, D.; Wang, W.; Weyand, T.; Andreetto, M.; and Adam, H. 2017. Mobilenets: Efficient convolutional neural networks for mobile vision applications. *arXiv preprint arXiv:1704.04861*.

Isola, P.; Zhu, J.-Y.; Zhou, T.; and Efros, A. A. 2017. Image-to-image translation with conditional adversarial networks. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, 1125–1134.

Jin, Y.; Zhang, J.; Li, M.; Tian, Y.; Zhu, H.; and Fang, Z. 2017a. Towards the automatic anime characters creation with generative adversarial networks. *arXiv preprint arXiv:1708.05509*.

Jin, Y.; Zhang, J.; Li, M.; Tian, Y.; Zhu, H.; and Fang, Z. 2017b. Towards the automatic anime characters creation with generative adversarial networks. *arXiv preprint arXiv:1708.05509*.

Karras, T.; Aila, T.; Laine, S.; and Lehtinen, J. 2017. Progressive growing of gans for improved quality, stability, and variation. *arXiv preprint arXiv:1710.10196*.

Karras, T.; Laine, S.; and Aila, T. 2019. A style-based generator architecture for generative adversarial networks. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 4401–4410.

Khungurn, P., and Chou, D. 2016. Pose estimation of anime/manga characters: a case for synthetic data. In *Proceedings of the 1st International Workshop on coMics ANalysis, Processing and Understanding*, 1–6.

Laina, I.; Rupprecht, C.; Belagiannis, V.; Tombari, F.; and Navab, N. 2016. Deeper depth prediction with fully convolutional residual networks. In *2016 Fourth international conference on 3D vision (3DV)*, 239–248. IEEE.

Liu, Z.; Luo, P.; Qiu, S.; Wang, X.; and Tang, X. 2016. Deepfashion: Powering robust clothes recognition and retrieval with rich annotations. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, 1096–1104.

Luo, P. 2016. OpenMMD. `https://github.com/peterljq/OpenMMD`.

Ma, L.; Jia, X.; Sun, Q.; Schiele, B.; Tuytelaars, T.; and Van Gool, L. 2017. Pose guided person image generation. In *Advances in Neural Information Processing Systems*, 406–416.

Ma, L.; Sun, Q.; Georgoulis, S.; Van Gool, L.; Schiele, B.; and Fritz, M. 2018. Disentangled person image generation. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 99–108.

Martinez, J.; Hossain, R.; Romero, J.; and Little, J. J. 2017. A simple yet effective baseline for 3d human pose estimation. In *Proceedings of the IEEE International Conference on Computer Vision*, 2640–2649.

Newell, A.; Yang, K.; and Deng, J. 2016. Stacked hourglass networks for human pose estimation. In *European conference on computer vision*, 483–499. Springer.

Odena, A.; Olah, C.; and Shlens, J. 2017. Conditional image synthesis with auxiliary classifier gans. In *Proceedings of the 34th International Conference on Machine Learning-Volume 70*, 2642–2651. JMLR. org.

Ohnishi, K.; Yamamoto, S.; Ushiku, Y.; and Harada, T. 2018. Hierarchical video generation from orthogonal information: Optical flow and texture. In *Thirty-Second AAAI Conference on Artificial Intelligence*.

Osokin, D. 2019. Global context for convolutional pose machines. *arXiv preprint arXiv:1906.04104*.

Pan, Y.; Qiu, Z.; Yao, T.; Li, H.; and Mei, T. 2017. To create what you tell: Generating videos from captions. In *Proceedings of the 25th ACM international conference on Multimedia*, 1789–1798.

Poursaeed, O.; Kim, V.; Shechtman, E.; Saito, J.; and Belongie, S. 2020. Neural puppet: Generative layered cartoon characters. In *The IEEE Winter Conference on Applications of Computer Vision*, 3346–3356.

Pu, Y.; Gan, Z.; Henao, R.; Yuan, X.; Li, C.; Stevens, A.; and Carin, L. 2016. Variational autoencoder for deep learning of images, labels and captions. In *Advances in neural information processing systems*, 2352–2360.

Qian, X.; Fu, Y.; Xiang, T.; Wang, W.; Qiu, J.; Wu, Y.; Jiang, Y.-G.; and Xue, X. 2018. Pose-normalized image generation for person re-identification. In *Proceedings of the European conference on computer vision (ECCV)*, 650–667.

tdrussell. 2016. IllustrationGAN. `https://github.com/tdrussell/IllustrationGAN/`.

Tome, D.; Russell, C.; and Agapito, L. 2017. Lifting from the deep: Convolutional 3d pose estimation from a single image. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2500–2509.

Wang, X., and Gupta, A. 2016. Generative image modeling using style and structure adversarial networks. In *European Conference on Computer Vision*, 318–335. Springer.

Wei, S.-E.; Ramakrishna, V.; Kanade, T.; and Sheikh, Y. 2016. Convolutional pose machines. In *Proceedings of the IEEE conference on Computer Vision and Pattern Recognition*, 4724–4732.

Xu, F.; Liu, Y.; Stoll, C.; Tompkin, J.; Bharaj, G.; Dai, Q.; Seidel, H.-P.; Kautz, J.; and Theobalt, C. 2011. Video-based characters: creating new human performances from a multi-view video database. In *ACM SIGGRAPH 2011 papers*. 1–10.

Yamamoto, S.; Tejero-de Pablos, A.; Ushiku, Y.; and Harada, T. 2018. Conditional video generation using action-appearance captions. *arXiv preprint arXiv:1812.01261*.

Zhang, H.; Xu, T.; Li, H.; Zhang, S.; Wang, X.; Huang, X.; and Metaxas, D. N. 2017a. Stackgan: Text to photo-realistic image synthesis with stacked generative adversarial networks. In *Proceedings of the IEEE international conference on computer vision*, 5907–5915.

Zhang, L.; Ji, Y.; Lin, X.; and Liu, C. 2017b. Style transfer for anime sketches with enhanced residual u-net and auxiliary classifier gan. In *2017 4th IAPR Asian Conference on Pattern Recognition (ACPR)*, 506–511. IEEE.

Zhang, L.; Li, C.; Wong, T.-T.; Ji, Y.; and Liu, C. 2018. Two-stage sketch colorization. *ACM Transactions on Graphics (TOG)* 37(6):1–14.

Zhu, J.-Y.; Park, T.; Isola, P.; and Efros, A. A. 2017. Unpaired image-to-image translation using cycle-consistent adversarial networks. In *Proceedings of the IEEE international conference on computer vision*, 2223–2232.

Zhu, Z.; Huang, T.; Shi, B.; Yu, M.; Wang, B.; and Bai, X. 2019. Progressive pose attention transfer for person image generation. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2347–2356.