Final Year Project Report

SHO1801 – Motion Retargeting

HUANG Chuanshan

1. Introduction and background

i. Motion Retargeting is a widely applied technique in the film and gaming industry. It appears in the development of CG Animation, game character animation, and the processing of Motion Captured data. However, in most cases, achieving motion retargeting requires laborious work of the editor - he/she needs to adjust the joints of the source one by one in order to facilitates the change of motion.

ii. The search of an automatic motion retargeting method has a long history. Previous methods try to solve it by applying spatial constraints such as setting footsteps, or training deep learning neural network. We tried to achieve motion retargeting by adopting spatial constraints in the source motion, say, fists should punch to certain locations or feet should step on certain places. While we tried to achieve the best visual effects, our results suffered from the problem of smoothness. The movement of the skeleton of some frames is not consistent with their neighboring frames – there exists some absurd flashing gestures of the skeleton.

2. Direction of Final Year Project this semester

i. This semester we experiment on the possible usage of Generative Adversarial Networks (GAN), a promising machine learning approach,

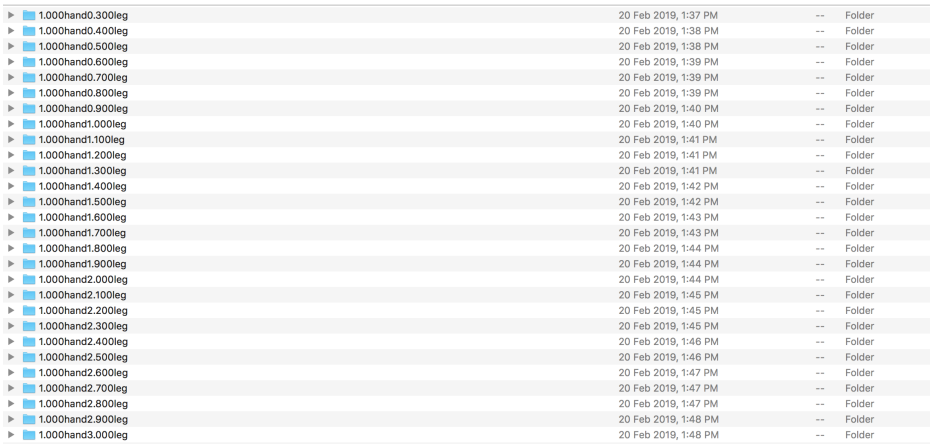
in resolving motion retargeting problems. Although most of the GANs are not controllable, we expect a GAN to learn the “pattern” among sets of motions, which is the motion itself instead of the skeleton.

3. Experiments with Generative Adversarial Network

   i. DCGAN

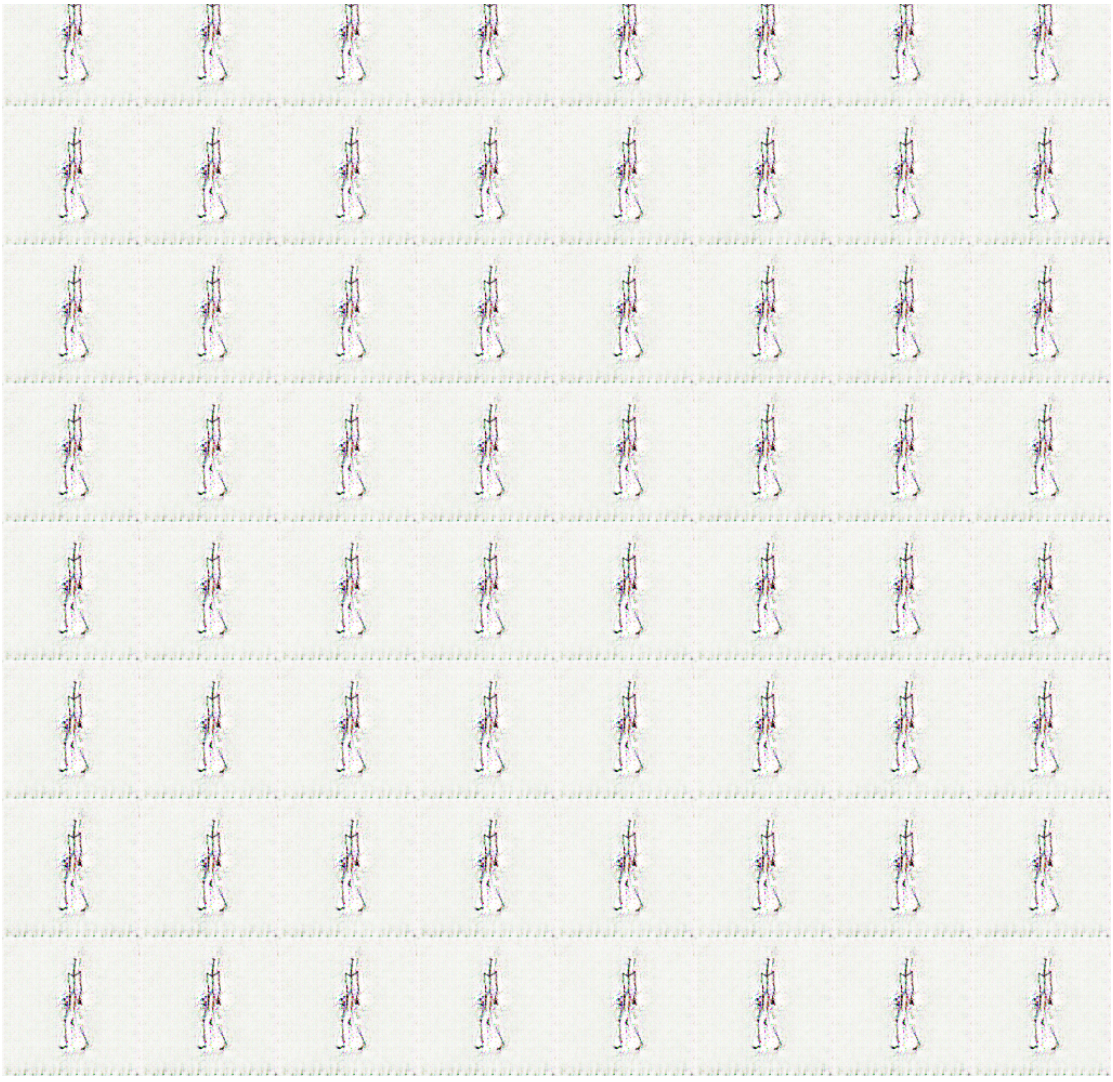
      1) DCGAN is known for its early use of batch normalization and the removal of maxpooling. DCGAN also proves that it can generate figures of same kind, so we want to know if it can extract the motion from the input.

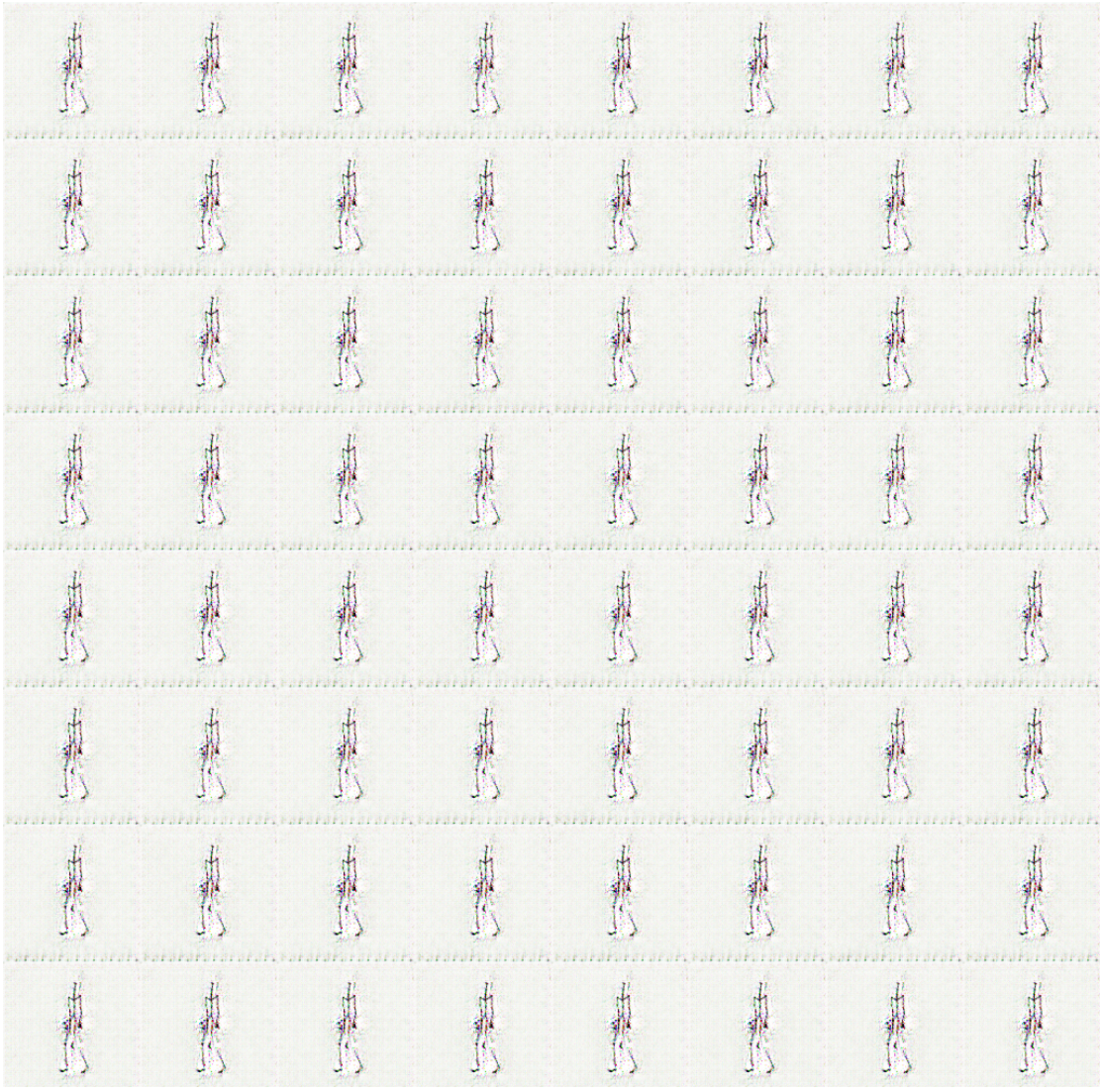
      2) Dataset: We prepare our own dataset based on CMU’s mocap data. We put each motion onto MatLab, visualize it with a white background, then capture an image for each five frames. We then scale the ratio of the length of arms and legs of the skeleton from 1:1 to 1:3 with a constant interval of 0.1, and redo the above procedure. Since it’s experimental stage, we only use the data of running motion. Below is a screenshot of our data:

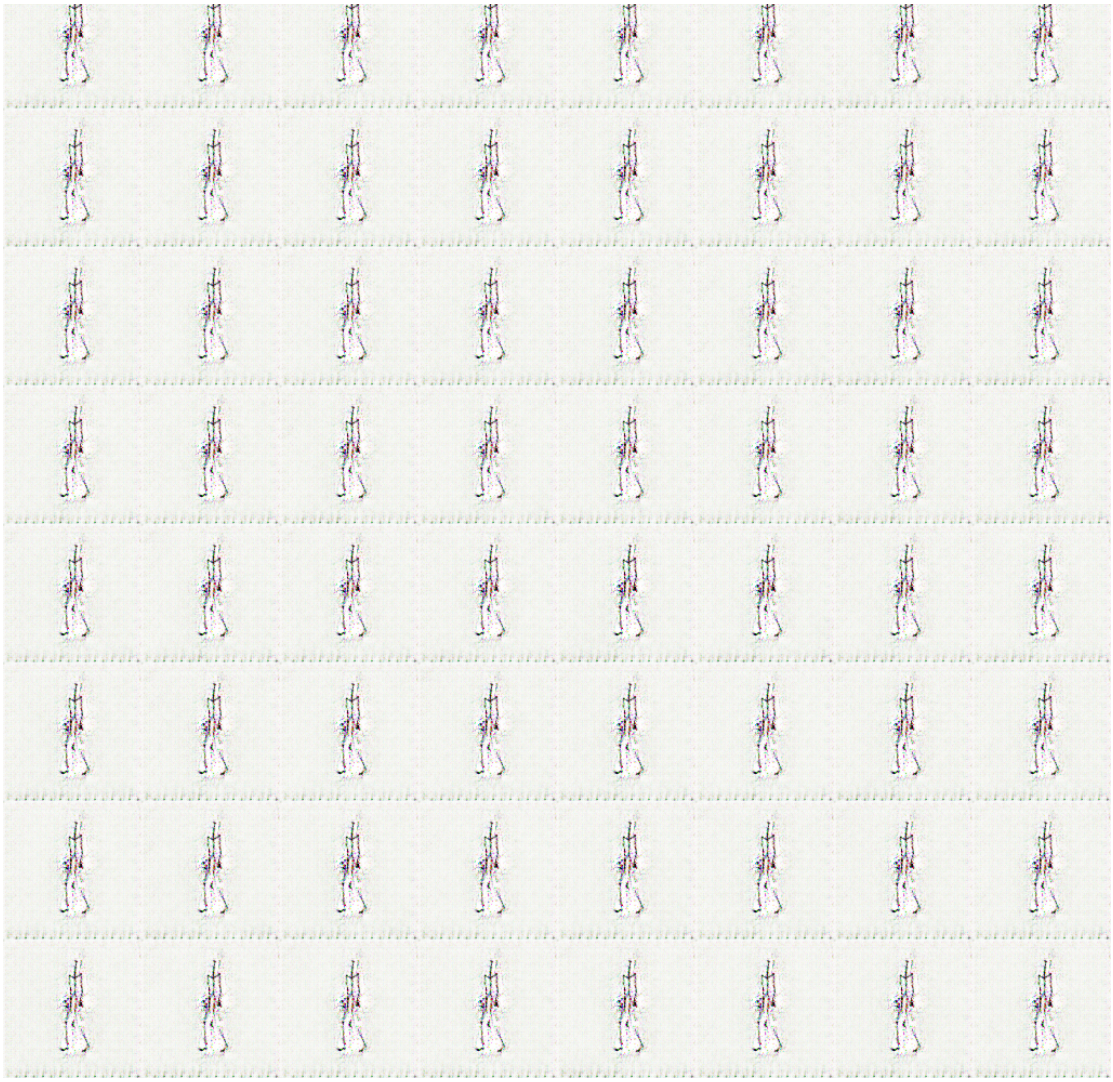


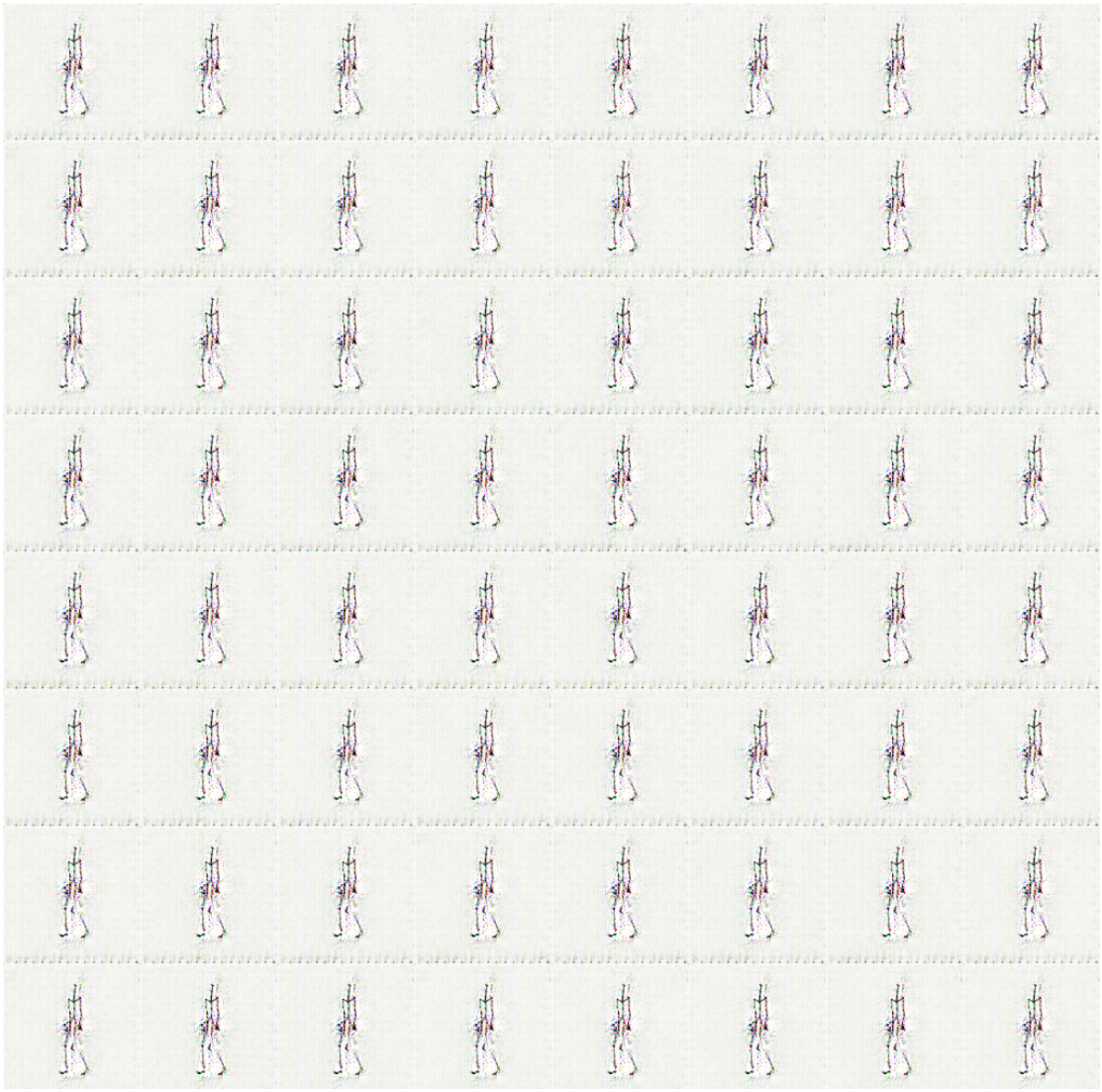
3) Training: We training on CS Department's GPU. The images are cropped to 128x128 using DCGAN's own cropper. We train the network for 150 epochs and obtain the following results (ordered in increasing epoch):

Arms:legs = 1:1

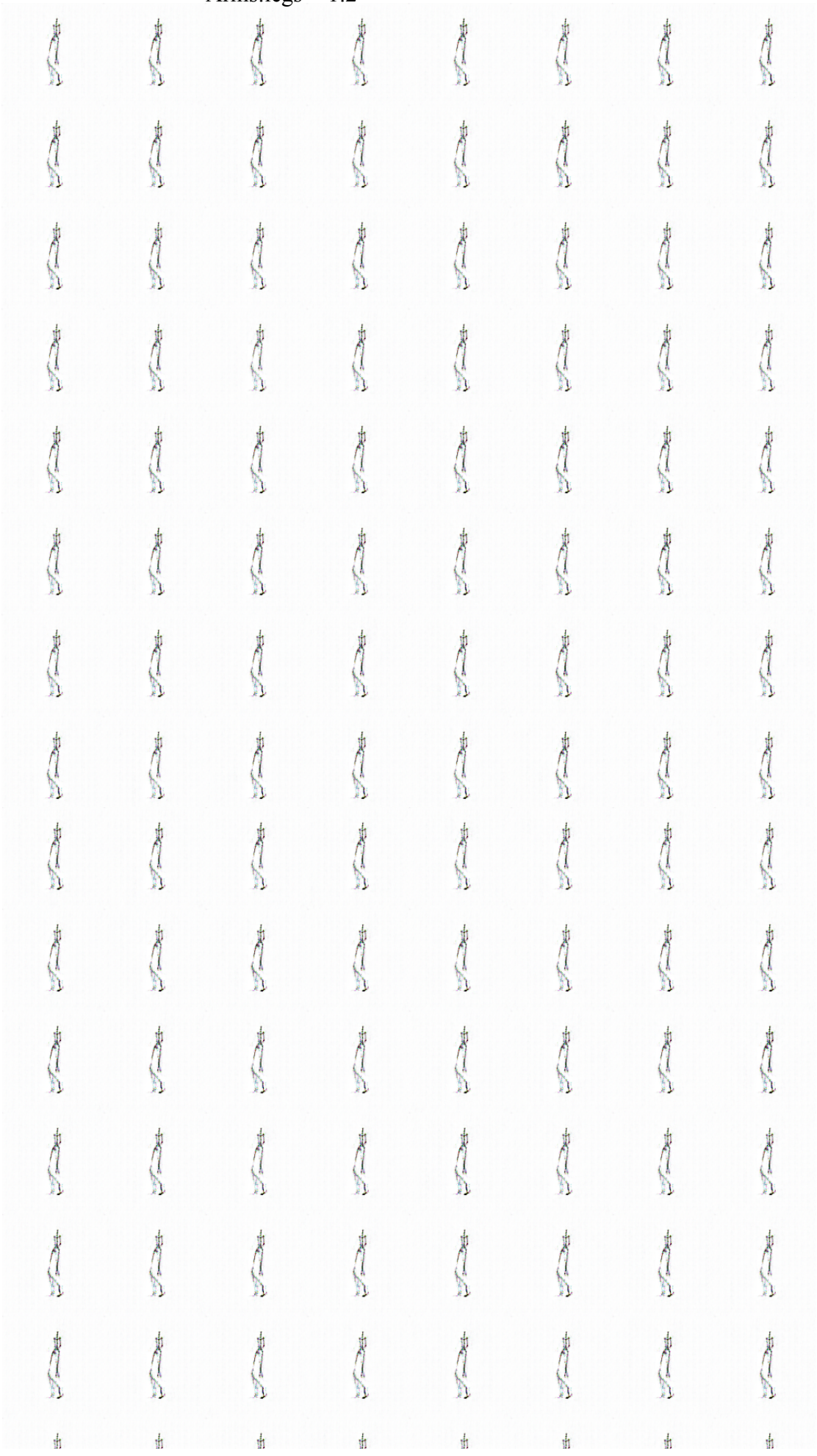


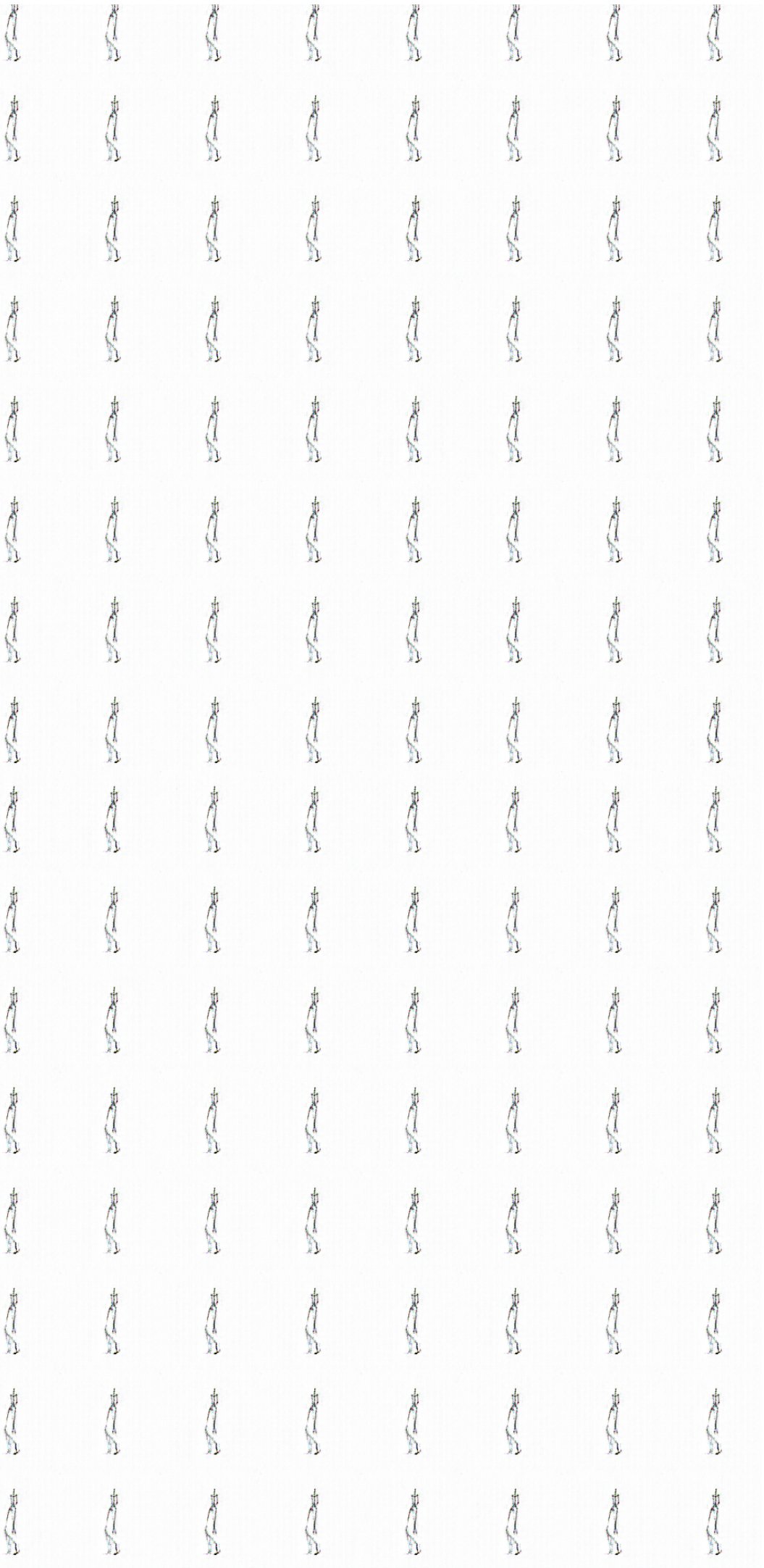






Arms:legs = 1:2

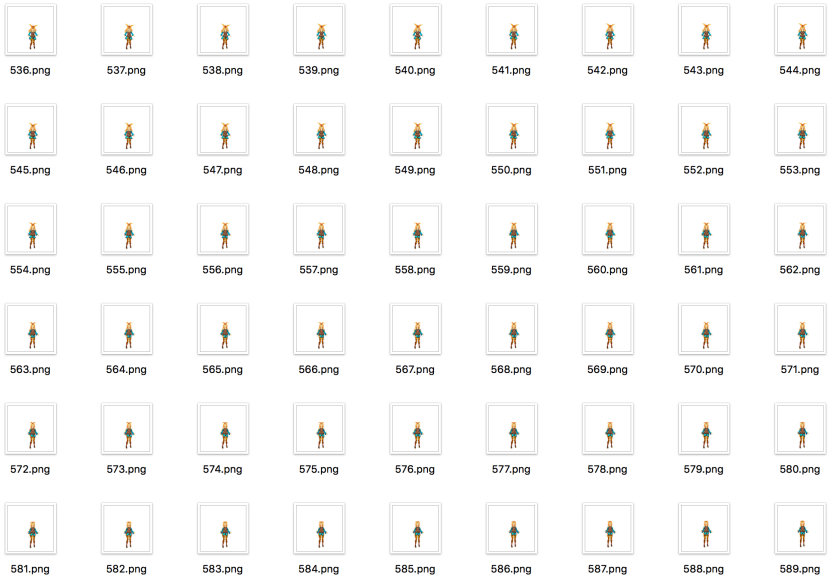




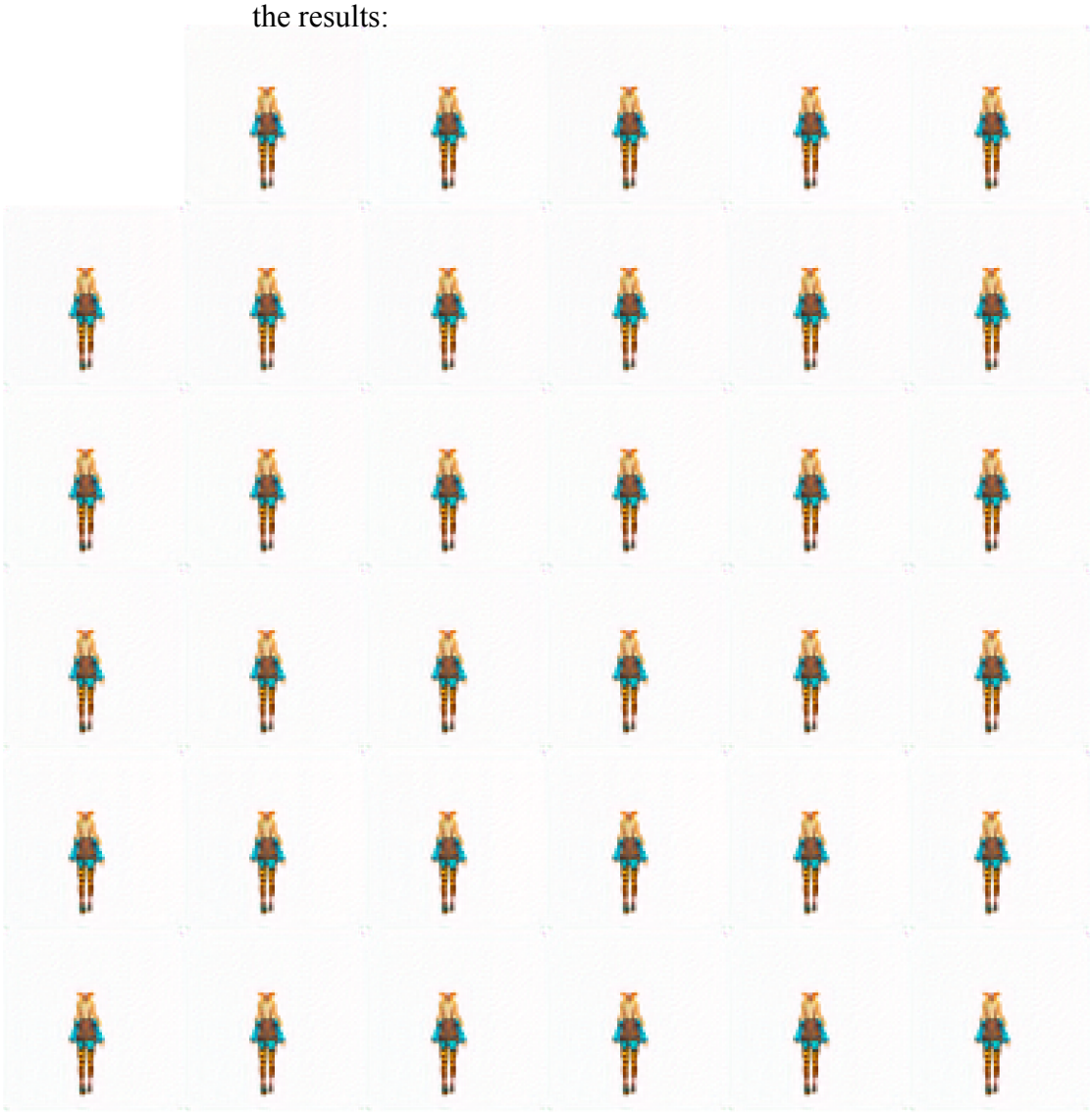
ii. StyleGAN

1) StyleGAN is famous for controllable generator with three types of styles and high-resolution output result. We want to see if it can obtain motion information this time.

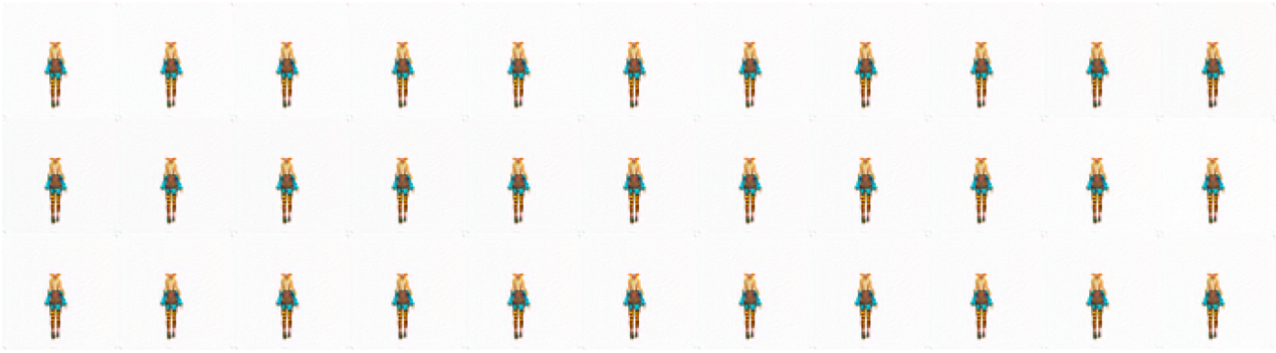
2) Dataset: We use UnityChan, the mascot of Unity Engine as our model. We set her up in an all white environment (no shadow) in Unity, play her animation, and capture the frames. The captured frames are further cropped to 512x512 by ourselves. Below is a screenshot of our data:



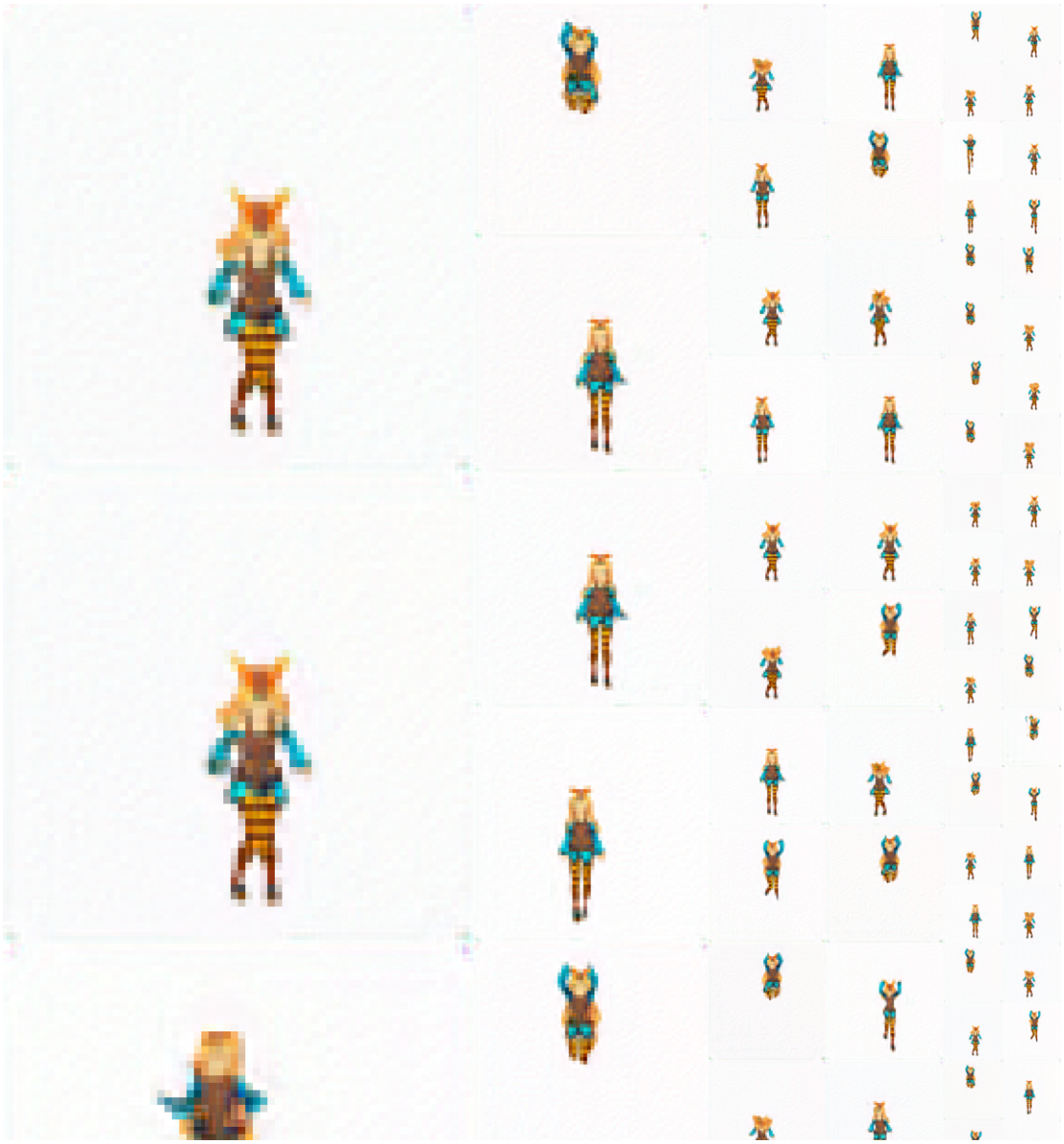
3) Training: I train the network with 1 GPU and Yu-Tang do it with 4 GPU. I train the network until 4344 kimg. Below are the results:



The image above is a style-mixing figure. The first row is the style detected by the network and the first column is the input.



The image above shows the truncation trick. Each row is images generated by tuning a parameter w, which is the strength of the style applied to the input.



The image above is the uncurated image, which shows the training progress from low resolution to high resolution.

4. Discussions and future work

   i. As one can see from above results, there is no visual difference among the test results of the same ratio group in DCGAN. It looks like DCGAN does learn the skeleton itself quite well, but fail to obtain the temporal information about the motion. A possible explanation for this is that in the implementation of DCGAN the author did not include some layers of recurrent neural networks, which is responsible for recognizing temporal information. Also, our frame-by-frame data lack significant different among the images, e.g. the changes of motion between consecutive images are too trivial.

   ii. As for styleGAN, although we expect much better performance, the problems are similar: no recurrent neural layers, and data too homogeneous. But at least styleGAN manages to recognize a figure with much higher resolution.

   iii. We did not run our data on any other GANs due to time limit. We spent too much time on applying GPU account and setting up the running environment. A future direction for this experimental learning is to run the data on a GAN with proper recurrent neural layers and promising ability in generating high-resolution images.