한국컴퓨터그래픽스학회
Korea Computer Graphics Society Vol. 27, No. 5, P. 33~43

변형 자동 인코더를 활용한 모션 스타일 이전

안제원O 권태수*
한양대학교 지능융합학과O 한양대학교 컴퓨터 소프트웨어학과*
ahnjawon@naver.com[1], taesoobear[2]*@gmail.com

Motion Style Transfer using Variational Autoencoder

Jewon AhnO Taesoo Kwon*
Dept. of Intelligence ConvergenceO Dept. of Computer and Software, Hanyang University*

요 약

본 논문에서는 변형 자동 인코더 네트워크(variational autoencoder network)의 잠재 공간 내에 스타일 자동 인코더 네트워크를 적용하여 컨텐츠 캐릭터의 모션에 스타일 캐릭터 모션의 스타일 정보를 이전하는 프레임워크를 제안한다. 이 프레임워크를 사용하면 기존의 변형 자동 인코더를 통해 얻은 모션의 다양성을 스타일 캐릭터 모션의 스타일 정보를 이전하여 증가시킬 수 있다. 또한 입력 데이터 및 출력 데이터에 모션의 속도 정보를 포함시켜 이전 프레임의 모션에 속도를 적분하여 모션을 계산함으로써, 변형 자동 인코더로 인한 샘플링과 잠재 공간 내에서 스타일 정보가 이전된 새로운 잠재 변수의 디코더 네트워크를 통한 확장으로 발생할 수 있는 부자연스러운 동작을 개선할 수 있다.

Abstract

In this paper, we propose a framework that transfers the information of style motions to content motions based on a variational autoencoder network combined with a style encoding in the latent space. Because we transfer a style to a content motion that is sampled from a variational autoencoder, we can increase the diversity of existing motion data. In addition, we can improve the unnatural motions caused by decoding a new latent variable from style transfer. That improvement was achieved by additionally using the velocity information of motions when generating next frames.

키워드: 변형 자동 인코더, 스타일 자동 인코더, 모션 스타일 이전, 생성적인 모델, 속도 적분

Keywords: variational autoencoder, style autoencoder, motion style transfer, generative models, velocity integrate

1. 서론

넓은 범위의 변형과 다른 스타일을 활용하여 캐릭터의 모션을 합성하는 것은 컴퓨터 애니메이션에서 많은 관심을 갖는 연구이다. 이를 위해 머신 러닝에 기반을 둔 모션 합성 접근법들이 큰 관심을 얻고 있지만 이런 모션 합성을 위한 모델들을 구성하는 것은 상당히 많은 양의 예제 데이터들을 요구하게 된다. 그러므로 많은 양의 예제 데이터들에 대한 효율적 사용이 중요해진 만큼, 추가적인 모션 캡쳐 없이 기존에 존재하는 모션 데이터베이스를 활용해 스타일 정보가 입혀진 모션을 만드는 모션 스타일 이전 방법이 많이 연구됐다.

본 논문에서는 모션 데이터베이스 정보들을 사용해 얻은 모션 및 속도 데이터를 통해 학습된 변형 자동 인코더 네트워크의 잠재 공간(latent space) 내에서 스타일 자동 인코더를 활용하여 스타일 이전된 새로운 잠재 변수(latent variable)를 도출할 수 있는 프레임워크를 제안한다. 변형 자동 인코더의 샘플링 된 잠재 변수들을 스타

*corresponding author: Taesoo Kwon/Hanyang University(taesoobear@gmail.com)

Received : 2021.11.06./ Review completed : 1st 2021.11.25. / Accepted : 2021.11.26.
DOI : 10.15701/kcgs.2021.27.5.33
ISSN : 1975-7883(Print)/2383-529X(Online)

일 자동 인코더를 활용해 조합하여 기존 모션 데이터베이스에서 더 다양한 모션을 생성할 수 있다. 또한 새로 형성된 잠재 변수를 디코더 네트워크에서 확장할 때 부자연스러운 모션이 형성되는 현상을 모션의 속도를 활용하여 개선했다는 것을 보였다.

즉, 본 논문에서 제안한 변형 자동 인코더와 스타일 자동 인코더가 결합된 프레임워크의 장점은 다음과 같다.

1. 변형 자동 인코더의 잠재 공간 내에서 스타일 자동 인코더를 활용하여 스타일 모션 이전을 진행했으며, 이를 통해 생성적인 모델(generative models)을 확장하여 기존 모션 데이터에서 더 다양한 모션을 생성 가능하게 한다.

2. 모션의 속도 관련 정보를 활용하여 샘플링과 스타일 이전을 통해 형성될 수 있는 부자연스러운 모션을 개선했다.

3. 저차원 공간으로 프로젝션 된 모션들의 다양체(manifold)가 서로 연관성 있는 잠재 공간 내에서 스타일 이전되므로 서로 다른 두 모션 사이의 시간 축(temporal axis)에 대한 정렬이 요구되지 않는다.

2. 관련 연구

캐릭터 모션 스타일 이전은 여러 방식으로 진행되어 왔다. 우리는 이와 관련된 연구들 중 본 논문과 직접적으로 관련이 있는 모션 스타일 이전, 생성적인 모델과 관련된 연구에 집중한다.

2.1 모션 스타일 이전

하나의 모션으로부터 스타일을 추출하여 다른 모션에 적용하는 모션 스타일 이전은 컴퓨터 애니메이션 분야에서 오래된 문제이다. 한 접근법으로 모션을 주파수 영역(frequency domain)에서 다루는 것이다 [1, 2]. Katherine Pullen과 Christoph Bregler는 모션 데이터의 고주파(high frequency) 요소들을 새로운 모션에 추가하여 스타일 이전을 진행했다 [3]. M. Ersin Yumer와 Niloy Mitra는 스타일을 나타내는 고주파 데이터를 추출하기 위해 빠른 퓨리에 변환(fast Fourier transform)을 적용했다 [4]. 다른 접근법으로 컨텐츠 모션으로부터 스타일 요소를 분리하는 것이 있는데, Brand와 Hertzmann은 스타일 모델을 학습하기 위해 HMMs를 적용하였고, 새로운 스타일 모션을 생성하기 위한 일반 모델을 사용하였다 [5]. 또한 모션 데이터베이스로부터 스타일과 컨텐츠에 대한 매개변수를 학습하기 위한 다중-선형(multi-linear) 모델을 구성하는 연구도 진행됐다 [6]. 딥러닝 또한 스타일 이전에 많이 사용됐다. Daniel Holden은 그램 행렬(Gram matrix)을 사용하여 특징들에서 모션의 스타일을 추출하였으며, 이를 1D 합성곱(convolution)으로 학습시켰다 [7, 8]. 본 논문에서는 위 아이디어에서 방법을 착안하여 그램 행렬을 통해 컨텐츠 모션과 스타일 모션의 정보를 추출했다. H. Du 등은 합성곱 자동 인코더의 잠재 공간 내에서 변형 자동 인코더를 사용하여 모션 스타일 이전을 진행하였다 [9].

2.2 생성적인 모델

캐릭터 모션의 분포를 모델링 하기 위해 은닉 마르코프 모델(Hidden Markov Models)을 사용하여 진행한 연구가 진행됐다 [10]. 또한 다른 방식으로 모션의 분포를 모델링하기 위해 모션 데이터를 낮은 차원의 공간으로 프로젝션 하여, 가우시안 혼합 모델(Gaussian Mixture Model)을 사용해 낮은 차원에서의 모션 분포를 학습하는 연구도 진행됐다 [11]. 나아가 독립적인 잠재 공간이 아닌 연결된 잠재 공간 내 다양체를 형성하기 위해 합성곱 자동 인코더(convolutional autoencoder)를 사용하였고, 이로 인해 모션 특징들에 대한 연속적인 다양체 공간에 대한 학습이 가능하게 됐다 [12]. 합성곱 자동 인코더 외에도 본 논문에서 사용한 변형 자동 인코더를 활용하여 큰 모션 데이터베이스의 분포를 모델링하는 것이 연구되었다 [13]. 이에 시간 축에 대한 순서를 고려하여 모션 데이터의 분포를 모델링 하기 위해 순환 변형 자동 인코더(recurrent variational autoencoder)를 사용한 연구도 진행됐다 [14]. 또한 앞서 나온 연구에 나아가 캐릭터의 포즈를 예측하기 위한 인코더-순환-디코더 모델을 제안하는 연구도 진행됐다 [15]. 본 논문에서는 생성적인 모델을 구현하기 위해 변형 자동 인코더를 사용하여 모션의 분포를 형성했다.

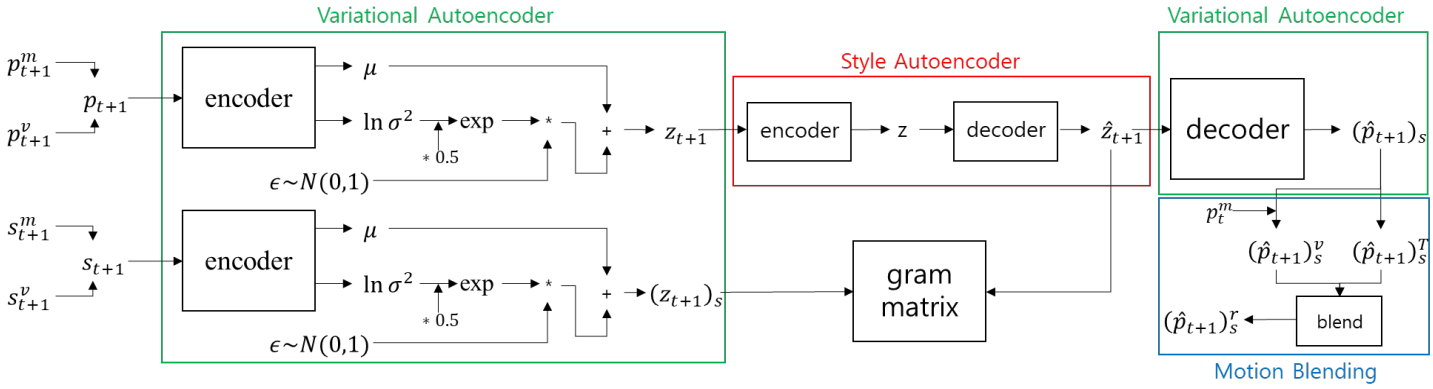


Figure 1: System Overview

3. 개요

본 논문에서는 먼저 스타일 이전을 위한 네트워크를 학습하기 위해 데이터를 준비하였다. 총 73차원으로서, t+1 프레임의 모션 데이터 정보(p^m_{t+1}), 그리고 t 프레임과 t+1 프레임 사이의 속도(p^v_{t+1})로 구성된다. 이에 대한 구체적인 설명은 데이터 준비(section 4)에서 진행하겠다.

Figure 1은 우리가 제안한 모델의 개요이다. 이 모델은 크게 모션을 임베딩(embedding)하는 인코더 네트워크와 임베딩 때 저차원으로 프로젝션 된 잠재 변수가 의미하는 것을 고차원으로 확장시키는 디코더 네트워크를 포함하는 변형 자동 인코더와 임베딩 된 컨텐츠 잠재 변수에 스타일 잠재 변수의 경향성을 이전해 주는 스타일 자동 인코더, 그리고 전체 네트워크를 통해 최종 도출된 모션 데이터를 토대로 t+1 프레임의 모션 데이터를 계산하는 모션 블렌딩(blending)으로 구성된다.

모션 데이터 셋에서 t+1 프레임의 모션 데이터 정보(p^m_{t+1})와 t 프레임과 t+1 프레임 사이의 속도 정보(p^v_{t+1})를 구한 후, 두 정보를 묶어 네트워크를 위한 입력 데이터(p_{t+1})로 사용하게 된다. 입력 데이터는 학습된 변형 자동 인코더의 인코더 네트워크를 통해 정의된 차원의 크기를 갖는 $\mu, \ln\sigma^2$로 인코딩 된다. $\mu, \ln\sigma^2$는 정규 분포 $N(0,1)$을 통해 샘플링 된 ϵ와 같이 잠재 변수(z_{t+1})를 구하는데 사용된다. 도출된 잠재 변수는 학습된 스타일 자동 인코더를 통해 스타일 모션 데이터(s_{t+1})를 나타내는 잠재 변수 정보가 입혀진 컨텐츠 모션(p_{t+1})에 스타일 모션이 이전된 모션을 나타내는 잠재 변수($\hat{z}_{t+1}$)로 변환된다. Figure 1에서 나타낸 스타일 모션 데이터(s_{t+1})의 잠재 변수인 $(z_{t+1})_s$은 스타일 자동 인코더를 학습시킬 때 그램 행렬(Gram matrix)을 활용한 손실 함수(loss function)를 구하는데 사용되며, 학습이 다 된 후에는 사용되지 않는다. 이와 관련된 내용은 스타일 자동 인코더(section 6)에서 자세히 다루겠다. 스타일 자동 인코더를 통해 변환된 새로운 잠재 변수($\hat{z}_{t+1}$)는 변형 자동 인코더의 디코더 네트워크를 통해 총 73차원의 입력 데이터의 구성과 같은 73차원의 출력 데이터($(\hat{p}_{t+1})_s$)로 확장된다. 출력 데이터 중 속도를 나타내는 데이터는 이전에 네트워크를 통해 얻은 t 프레임의 모션 데이터(p^m_t)에 적분 된다. 즉, 이 과정을 통해 t 프레임에 속도가 적용된 t+1 프레임 모션 데이터($(\hat{p}_{t+1})^v_s$)를 얻게 된다. 이 후 $(\hat{p}_{t+1})^v_s$는 출력 데이터 중 t+1 프레임의 타겟 모션 데이터를 나타내는 $(\hat{p}_{t+1})^T_s$와 블렌딩되고, 그 결과 t+1 프레임의 모션 데이터($(\hat{p}_{t+1})^r_s$)를 얻게 된다. 이는 모션 블렌딩(section 7)에서 다루겠다.

4. 데이터 준비

본 논문에서는 네트워크의 입력 데이터로 총 73차원의 데이터를 사용하였다.

4.1 모션 데이터

73차원 중 40차원은 캐릭터의 t+1 프레임에 대한 모션 데이터이다. 1~14 인덱스에는 캐릭터의 골반(root)의 정보가 입력된다. 1~7 인덱스에는 글로벌 이동(global translation(x, y, z)), 그리고 쿼터니언(quaternion(w, x, y, z)) 데이터로 구성되는 캐릭터의 골반 좌표계를 지면에 2D 프로젝션 하여 t 프레임과 t+1 프레임 사이의 델타(delta)를 구한 값이 포함된다. 캐릭터의 골반 좌표계를 2D 프로젝션 시켰으므로, 델타 값은 t 프레임과 t+1 프레임 사이의 y축 회전, x, z 이동(translation)의 차이를 의미하게 된다. 8~14 인덱스에는 t 프레임에서 위에서 구한 델타가 적용된 값인 지면에 2D 프로젝션된 t+1 프레임의 골반 좌표계와 t+1 프레임의 골반 좌표계 사이의 오프셋(offset)을 구한 값이 포함된다. 오프셋은 델타와 다르게 x, z축 회전과 y 이동 값을 갖게 된다. 따라서 네트워크

실행 시 t+1 프레임의 골반 좌표계는 네트워크 출력 데이터의 1~14 인덱스 값을 이전 단계에서 구한 t 프레임의 골반 좌표계의 2D 프로젝션 값에 적용하여 계산된다. 또한 오프셋과 델타로 나누어 차원을 줄여 계산함으로써 에러가 누적되는 것을 방지할 수 있으며, 변화량을 나타냄으로써 모션 재생 시 연속적인 재생이 가능하게 된다. 15~40 인덱스에는 t+1 프레임에 대해 각 관절(joint)의 자유도만큼 상위 관절에 상대적인 회전 값(radian)이 포함된다.

4.2 속도

VAE를 사용하여 잠재 변수가 구해질 때 샘플링이 적용되며, 이로 인해 여러 종류의 모션이 학습됨에 따라 디코딩 된 모션이 불안정 할 수 있다. 또한 스타일 이전 시, 새로 형성된 잠재 변수를 디코더 네트워크가 기존 잠재 변수들 사이에서 보간(interpolation) 할 때 부자연스러운 모션이 형성될 수 있다. 이를 해결하기 위해 t 프레임과 t+1 프레임 모션 사이의 속도 정보를 추가하였다. 1~3 인덱스와 5~7 인덱스는 골반의 공간 속도(spatial velocity)를 나타내며, 각각 선속도(linear velocity)와 각속도(angular velocity) 값이 포함되며 아래의 순서로 계산된다.

골반의 변환 행렬(transformation matrix) T(t)에서 회전 행렬은 해당 프레임의 골반 좌표계의 쿼터니언(q) 값을 활용하여 계산한다. 쿼터니언의 크기가 1임을 확실히 보장하기 위해 정규화(normalize)를 추가한다. 프레임 레이트는 120으로 설정하였다.

$$q = (w, x, y, z)\,, s = \frac{2}{\sqrt{w^2 + x^2 + y^2 + z^2}}$$

일 때

$$\mathrm{R}(t) = \begin{bmatrix} 1 - s(y^2 + z^2) & s(xy - wz) & s(xz + wy) \\ s(xy + wz) & 1 - s(x^2 + z^2) & s(yz - wx) \\ s(xz - wy) & s(yz + wx) & 1 - s(x^2 + y^2) \end{bmatrix}$$

$$\mathrm{T}(t) = \begin{bmatrix} \mathrm{R}(t) & \mathrm{p}(t) \\ 0 & 1 \end{bmatrix}$$

$$\mathrm{v}_{global} = \big(T(t+1) - T(t)\big) * framerate$$

여기서 구한 속도(v_{global})는 글로벌 좌표계에 대한 속도이다. 공간 속도는 이전 프레임 즉, t 프레임의 골반 좌표계에 대해 표현돼야 하므로 아래와 같이 행렬 곱셈을 진행한다. 그 결과 각속도의 3x3 반대칭행렬(skew-symmetric matrix) $[w]$과 선속도 v를 얻게 되며 이를 해당 인덱스에 입력한다.

$$\mathrm{v}_{t\_frame} = \mathrm{T}(t)^{-1} * \mathrm{v}_{global} = \begin{bmatrix} [w] & v \\ 0 & 0 \end{bmatrix}$$

$$[w] = \begin{bmatrix} 0 & -r_3 & r_2 \\ r_3 & 0 & -r_1 \\ -r_2 & r_1 & 0 \end{bmatrix}, v = \begin{bmatrix} v_x \\ v_y \\ v_z \end{bmatrix}$$

인덱스 4, 8~33는 해당 모션 데이터 값(j)의 t 프레임과 t+1 프레임 사이의 속도 값이 포함된다.

$$v_k = (j_{t+1} - j_t) * framerate\,,\ k \in 4, [8{,}33]$$

모션 캐릭터의 골반 좌표계는 글로벌 이동(global translation(x, y, z)), 그리고 쿼터니언(quaternion(w, x, y, z)) 데이터로 구성되며, 총 13개의 관절을 갖는다.

5. 변형 자동 인코더

스타일 자동 인코더를 적용하기 전 미리 학습시키는 변형 자동 인코더는 데이터 준비(section 4)에서 설명한 입력 데이터(p_{t+1})를 정답 데이터로 한 지도 학습(supervised learning)을 활용하여 학습된다. Figure 2는 figure 1에서 스타일 자동 인코더가 제외된 변형 자동 인코더 네트워크를 나타낸다. 학습된 변형 자동 인코더는 t+1 프레임의 모션 정보를 갖는 출력 데이터를 내재적으로 생성하는데, 이 때 생성된 데이터는 정규 분포화 된(normally-distributed) 잠재 변수로 인해 샘플링 돼 가능한 범위 내의 한 분포를 갖게 된다. 즉, 기존에 갖고 있는 모션과는 비슷하지만 샘플링으로 인해 좀 다른 모션이 생성된다. 각 부분별로 요약하면 인코더 네트워크를 통해 샘플링 된 잠재 변수들이 도출되며 이는 디코더 네트워크를 통해 t+1 프레임의 모션 정보를 갖는 출력 데이터로 변환된다.

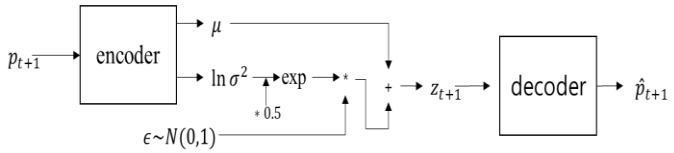


Figure 2: The structure of variational autoencoder which is pretrained before training style autoencoder.

5.1 인코더 네트워크

총 6개의 은닉 레이어(hidden layer)를 포함하는 인코더는 입력 데이터(p_{t+1})를 잠재 변수로 임베딩 하는 역할을 수행한다. 각 은닉 레이어는 입력 레이어에서 출력 레이어 방향으로 256, 256, 128, 64, 32, 16의 은닉 유닛(hidden units)들로 구성되며, 각 은닉 레이어의 활성화 함수로 elu를 사용했다. 출력 레이어에 해당하는 μ와 $\ln\sigma^2$의 차

원은 각각 2로 정하였다. 또한 μ와 $\ln\sigma^2$를 활용하여 잠재 변수를 샘플링 할 때 재매개변수화 트릭(reparameterization trick)을 사용했다. Figure 3은 학습된 인코더를 통과한 모션 별 잠재 변수를 출력한 결과이다. 결과를 보면 알 수 있듯이, 인코딩 된 잠재 공간은 비슷한 센터를 공유하고 있다. 이는 변형 자동 인코더 네트워크 모델이 모션들의 유사성을 알고 있다는 뜻이며, 이로 인해 스타일 이전이 진행될 때 디코더가 모션들 사이에서 스타일이 이전된 모션을 보간 할 수 있게 된다. 만약 figure 3처럼 잠재 공간끼리 서로 관련된 상태가 아닌 서로 분리되어 잠재 공간을 형성한다면, 스타일 이전이 진행될 때 컨텐츠 모션과 스타일 모션 사이의 보간이 디코더에서 이루어지지 않아 컨텐츠 모션에 대한 잠재 공간과 스타일 모션에 대한 잠재 공간 사이에서 연관성 없이 모션이 불연속적으로 형성된다. 이는 스타일 자동 인코더(section 6)에서 설명할 스타일 손실 함수에 대한 가중치를 줄여도 해결되지 않는다. 이 경우 오히려 스타일에 대한 정보가 아예 학습되지 않아 컨텐츠 모션과 거의 같은 모션이 생성된다. 이와 관련된 실험은 평가(section 9)에서 진행하겠다.

$$z = \mu + e^{\frac{1}{2}\ln\sigma^2} * \epsilon, \qquad \epsilon \sim N(0,1)$$

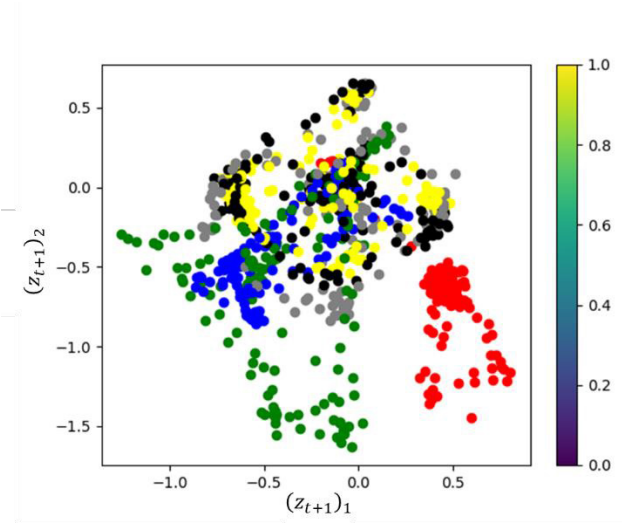


Figure 3: The latent space of variational autoencoder. The x axis represents the first latent variable, and the y axis represents the second latent variable.

5.1 디코더 네트워크

총 6개의 은닉 레이어를 포함하는 디코더는 잠재 변수 (z_{t+1})를 출력 데이터$(\hat{p}_{t+1})$로 변환하는 역할을 수행한다. 각 은닉 레이어는 입력 레이어에서 출력 레이어 방향으로 16, 32, 64, 128, 256, 256의 은닉 유닛들로 구성되며, 각 은닉 레이어의 활성화 함수로 elu를 사용했다. 출력 레이어의 차원은 입력 레이어의 차원과 동일하게 73차원으로 진행했다

5.3 손실 함수

변형 자동 인코더 손실 함수($\mathbf{L}_v$)는 기존 변형 자동 인코더의 손실 함수의 개념대로 설정하였다 [16]. 복원 손실(reconstruction loss) ($\mathbf{L}_{\mathbf{recon}}$)은 입력 데이터$(p_{t+1})$와 출력 데이터$(\hat{p}_{t+1})$ 사이의 평균 제곱 오차(mean squared error)를 사용하여 정의했으며, 정규화 부분(regularization term)인 쿨백-라이블러 발산(KL-divergence) 손실 ($\mathbf{L}_{\mathbf{KL}}$)은 사전 분포(prior distribution)를 정규 분포로 선택했으므로, 아래 식처럼 사용했다.

$$\mathbf{L}_v = \mathbf{L}_{\mathbf{recon}} + \mathbf{L}_{\mathbf{KL}}$$

$$\mathbf{L}_{\mathbf{recon}} = \frac{\sum_{i=1}^{N}((\hat{p}_{t+1})_i - (p_{t+1})_i)^2}{N}, \qquad N = 73$$

$$\mathbf{L}_{\mathbf{KL}} = \frac{1}{2}\sum_{j=1}^{K}\left(\sigma_j + {\mu_j}^2 - 1 - \log\sigma_j\right), \qquad K = 2$$

6. 스타일 자동 인코더

학습된 변형 자동 인코더를 사용하여 스타일 모션의 특징을 컨텐츠 모션에 이전해 두 모션이 섞인 새로운 모션을 생성하기 위해 변형 자동 인코더의 인코더와 디코더 사이에 figure 4와 같은 스타일 자동 인코더를 적용하여 잠재 변수에 대한 제어를 진행했다. 이를 위해 스타일 자동 인코더 학습 시 성분 간의 상관 관계를 나타내는 그램 행렬을 사용한 손실 함수를 추가하여 컨텐츠 잠재 변수가 스타일 자동 인코더를 통과하여 얻어진 잠재 변수$(\hat{z}_{t+1})$내 성분들 사이의 관계가 스타일 모션의 잠재 변수내 성분들 사이의 관계와 유사하도록 이끌었다. 즉, 그램 행렬을 활용한 스타일 손실 함수를 구속 조건(constraint)으로 하여 스타일 자동 인코더를 학습시켜, 컨텐츠 모션의 잠재 변수가 학습된 자동 인코더를 통과할 때 똑같이 나오지 않고 스타일 모션의 잠재 변수들 사이의 경향성을 쫓아가게 조절했다. 이렇게 한개의 구속 조건으로 제어가 가능한 이유는 잠재 변수의 차원이 2차원으로 매우 낮기 때문이다. 잠재 공간의 차원을 줄이게 되면 잠재 공간내 잠재 변수들이 나타내는 다양체의 구조를 단순화시킬 수 있으며, 이는 복잡한 연산 없이도 제어가 가능함을 의미한다. 또한 차원을 낮춤으로 인해 변형 자동 인코더(section 5)가 학습될 때 형성되는 각 모션의 다양체가 서로 연관될 확률이 높아지게

된다. 실제로 figure 3에서 잠재 변수들이 찍힌 모양은 타원이나 원과 같은 단순한 구조를 이루면서 서로 연관되어 있음을 알 수 있다.

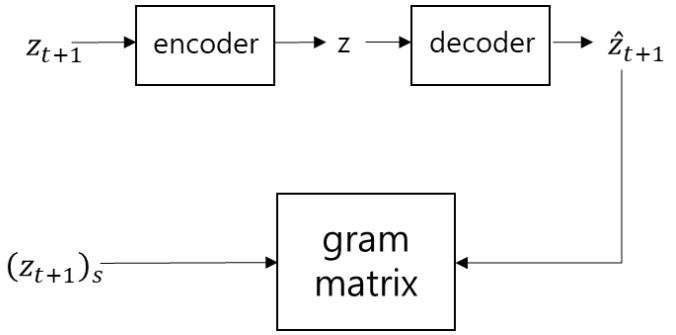


Figure 4: The structure of style autoencoder which is trained by using pretrained variational autoencoder.

스타일 자동 인코더는 변형 자동 인코더와 마찬가지로 인코더 네트워크, 디코더 네트워크로 구성된다. 인코더 네트워크는 총 5개의 은닉 레이어를 포함하며, 각 은닉 레이어는 입력 레이어에서 출력 레이어 방향으로 16, 32, 32, 64, 64의 은닉 유닛들로 구성된다. 인코더 네트워크의 출력 레이어의 차원인 자동 인코더의 잠재 변수(z)의 차원은 8로 설정하였다. 디코더 네트워크 역시 총 5개의 은닉 레이어를 포함하며, 각 은닉 레이어는 입력 레이어에서 출력 레이어 방향으로 64, 64, 32, 32, 16의 은닉 유닛들로 구성된다. 모든 은닉 레이어의 활성화 함수는 elu를 사용했다.

6.1 손실 함수

스타일 자동 인코더 손실 함수($\mathbf{L_s}$)는 컨텐츠 모션의 잠재 변수(z_{t+1})와 비슷하도록 제어하는 컨텐츠 손실 함수($\mathbf{L_{content}}$)와 컨텐츠 잠재 변수가 스타일 자동 인코더를 통과하여 얻어진 잠재 변수($\hat{z}_{t+1}$)와 스타일 잠재 변수($(z_{t+1})_s$)내 상관 관계가 유사하도록 제어하는 스타일 손실 함수($\mathbf{L_{style}}$)로 구성된다. 컨텐츠 손실 함수는 입력 데이터(z_{t+1})와 출력 데이터($\hat{z}_{t+1}$) 사이의 평균 제곱 오차를 사용하여 계산했다. 스타일 손실 함수의 경우, 컨텐츠 잠재 변수가 스타일 자동 인코더를 통과하여 얻어진 잠재 변수($\hat{z}_{t+1}$)의 그램 행렬과 스타일 잠재 변수($(z_{t+1})_s$)의 그램 행렬을 구하여 두 행렬 성분 사이의 평균 제곱 오차를 사용하여 계산했다. 컨텐츠 손실 함수에 대한 가중치 c는 0.1로, 스타일 손실 함수에 대한 가중치 s는 0.01로 설정했다.

$$\mathbf{L_s} = \mathbf{L_{content}} + \mathbf{L_{style}}$$

$$\mathbf{L_{content}} = c\|\hat{z}_{t+1} - z_{t+1}\|_2^2$$

$$\mathbf{L_{style}} = s\|\mathrm{Gram}((z_{t+1})_s) - \mathrm{Gram}(\hat{z}_{t+1})\|_2^2$$

$$\mathrm{Gram}(\mathrm{H}) = \sum_i \mathrm{H_i}\mathrm{H_i^T}$$

7. 모션 블렌딩

입력 데이터가 제안된 네트워크를 통과하게 되면, 변형 자동 인코더 내 샘플링으로 인해 약간 변형된 t+1 프레임의 모션 데이터와 t 프레임과 t+1 프레임 사이의 속도 데이터가 도출된다. 출력 데이터로 얻어진 t+1 프레임의 모션 데이터 정보는 이 후 타겟 모션 데이터 정보 즉, 레퍼런스와 같은 개념으로 사용된다. 출력 데이터들을 활용하여 t+1 프레임의 모션은 아래의 순서로 계산된다.

먼저 데이터 준비(section 4)에서 정리한 속도 계산 순서의 역으로 t 프레임과 t+1 프레임 사이의 모션에 대한 변화량을 출력 데이터 중 t 프레임과 t+1 프레임 사이의 속도를 활용해 계산한다. 계산된 변화량을 t 프레임의 모션 데이터에 적용하면 이전 프레임에 속도가 적분 된 t+1 프레임의 모션 데이터 결과가 생성된다.

속도가 적분 된 t+1 프레임 모션 데이터($(\hat{p}_{t+1})_s^v$)의 골반 좌표계의 정면 방향과 x, z 이동으로 네트워크를 통해 얻은 t+1 프레임의 타겟 모션 데이터($(\hat{p}_{t+1})_s^T$)의 골반 좌표계를 정렬한다. 이 후 정렬된 두 모션 데이터 정보 사이의 가중치(w_k) 만큼의 비율을 갖는 t+1 프레임 최종 모션 데이터($(\hat{p}_{t+1})_s^r$)를 계산한다. w_k는 0.4로 설정하였다.

$$(\hat{p}_{t+1})_s^r = (1 - w_k) * (\hat{p}_{t+1})_s^v + w_k * (\hat{p}_{t+1})_s^T$$

이 전 프레임과 관련된 정보인 속도를 활용하였으므로 최종 계산된 t+1 프레임 모션 데이터는 속도를 적용하지 않을 때 보다 훨씬 더 자연스러운 모션을 형성하며 이는 실험(section 9)에서 비교했다.

8. 구현

본 연구는 32GB RAM, AMD Ryzen 9 3950X 16-Core Processor CPU (32 Cores) @ 2.2GHz, AMD Radeon Pro W5500 등으로 구성된 PC에서 진행되었고, Python ver.3.8.10환경에서 Tensorflow ver.2.4.1, Keras ver.2.4.0을 사용하여 변형 자동 인코더 및 스타일 자동 인코더 네트워크의 구현 및 학습을 수행하였다.

변형 자동 인코더는 총 2085 프레임의 걷기와 약간 빠르게 걷기의 속도를 가지며 뒤로 몸을 젖히기, 보폭을 크게 하기, 팔을 강하게 흔드는 등의 스타일을 나타내는

6종류의 모션 데이터를 통해 에포크(epoch)는 15, 미니 배치 사이즈(minibatch size)는 64, 학습률(learning rate)은 10^{-4}로 설정하여 진행했으며, 총 약 45분이 소요됐다. 학습 에포크가 작은 이유는 이를 감소시키고자 2085 프레임의 모션 입력 데이터를 3000배 똑같이 복사하여 이를 입력 데이터로 학습을 진행했기 때문이다. 이 후 스타일 자동 인코더는 스타일, 컨텐츠 모션을 각각 131프레임으로 진행했다. 에포크는 10으로 설정하였으며, 미니 배치 사이즈 및 학습률은 변형 자동 인코더와 동일하게 진행했으며 총 약 1분이 소요됐다.

9. 실험

본 논문에서 제안하는 프레임워크를 다음의 세가지 항목으로 평가하였다. 먼저 인코딩 시 샘플링이 제대로 이루어지는지에 대해 변형 자동 인코더만을 사용한 결과를 통해 확인하였고, 스타일이 이전된 모션의 결과에 대한 평가를 다른 모션들 과의 비교를 통해 진행하였다. 이 후 잠재 변수가 이루는 다양체가 서로 연관되지 않는 경우와 연관된 경우를 비교하여 인코더 네트워크(section 5.1)에서 설명한 것을 증명하였고, 마지막으로 t+1 프레임의 스타일이 입혀진 최종 모션을 생성할 때 t 프레임과 t+1 프레임 사이의 속도를 적용한 것과 하지 않은 것의 차이를 비교하여 속도를 적용했을 시 더 부드러운 모션을 생성할 수 있음을 확인하였다.

9.1 네트워크 결과

Figure 5는 학습된 변형 자동 인코더만을 사용한 결과이다. 행 마다 가장 오른쪽에 있는 밝은 회색의 캐릭터가 변형 자동 인코더의 입력 모션이며, 나머지 진한 회색의 캐릭터는 변형 자동 인코더의 출력 모션이다. 기존 모션을 비교하여 봤을 때, 인코딩 때의 샘플링으로 인해 기존 모션에서 약간의 다양성이 추가된 것을 확인할 수 있다. Figure 6은 본 논문에서 제안한 프레임워크를 통해 스타일이 이전된 결과를 원본 컨텐츠 및 스타일 모션과 비교한 결과이다. 진한 회색의 캐릭터가 스타일이 이전된 결과이며, 밝은 회색의 캐릭터가 원본 모션이다. 즉, 첫번째 행의 밝은 회색 캐릭터 컨텐츠 모션에 두번째 행의 밝은 회색 캐릭터 스타일 모션의 스타일 정보를 이전한 결과가 첫번째, 두번째 행의 진한 회색의 캐릭터 모션이다. 이를 통해 스타일 정보가 입혀진 즉, 스타일 모션과 컨텐츠 모션이 적절히 조합된 새로운 모션이 생성됨을 알 수 있다. 또한 추가로 제안된 프레임워크를 통해 스타일이 이전된 결과와 학습된 변형 자동 인코더만을 사용하여 컨텐츠 모션과 스타일 모션을 재구성한 결과들을 figure 7에서 비교하였다. 이는 figure 8의 잠재 변수가 찍힌 그래프에서 확인할 수 있듯이 스타일 자동 인코더를 통해 재구성된 잠재 변수(파란색)가 변형 자동 인코더의 인코더 네트워크를 통과한 스타일 및 컨텐츠 잠재 변수들과 가까이 존재하기 때문에 재구성된 잠재 변수가 디코딩 시 컨텐츠나 스타일 중 어느 한 쪽으로 편향되어 확장될 수 있으므로 이를 확인하기 위해 진행하였다. 가장 첫번째 행과 두번째 행은 순서대로 스타일 이전된 캐릭터(진한 회색)와 스타일 모션이 변형 자동 인코더만을 통과한 캐릭터(밝은 회색)를 나타내며, 세번째 행과 네번째 행은 순서대로 스타일 이전된 캐릭터와 컨텐츠 모션이 변형 자동 인코더만을 통과한 캐릭터를 나타낸다. Figure 7의 결과를 확인할 수 있듯이, 어느 한 쪽으로 치우치지 않고 디코딩 시 스타일 다양체와 컨텐츠 다양체 사이에서 잘 보간되어 확장됨을 확인할 수 있다.

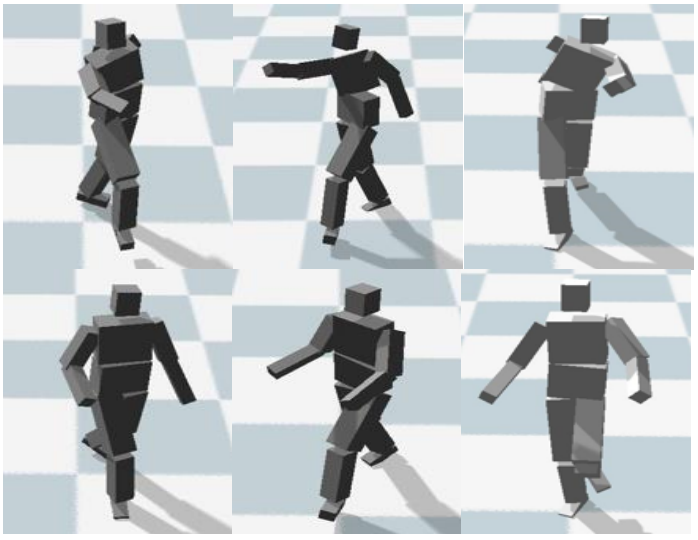
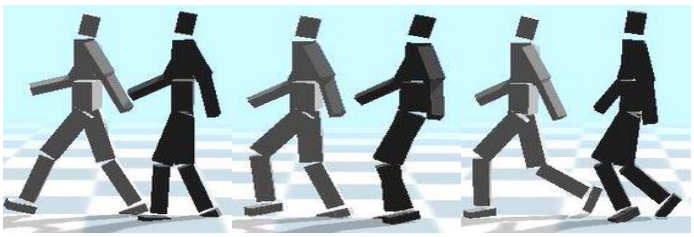


Figure 5: Snapshots of two motions that the dark gray character represent motion that is generated by variational autoencoder without style autoencoder, and light gray character on the right-most in every row is the original motion.



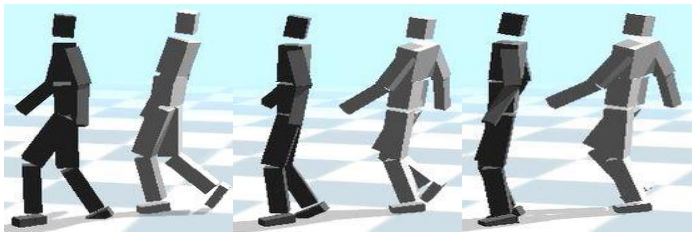


Figure 6: Snapshots of two motions where the dark gray character shows the output of style transfer, and the light gray character shows the original style motion (the second row) and the content motion (the first row). The style information of the original motion (the second row, light gray character) is transferred to the original content (the first row, light gray character).

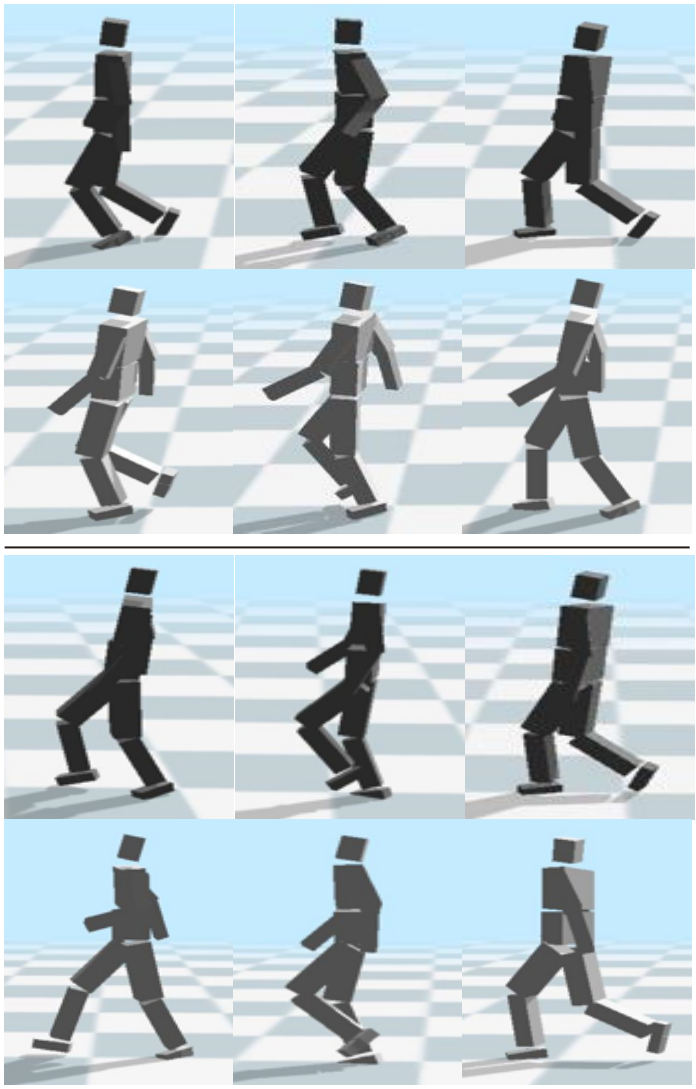
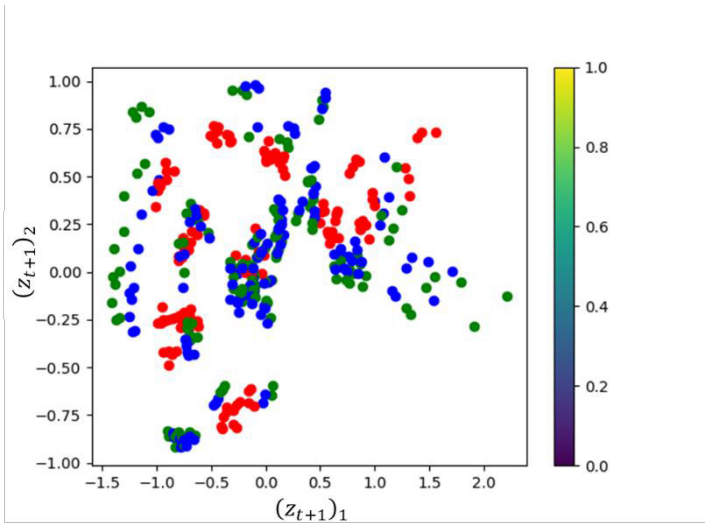


Figure 7: The first and third row (dark gray character) shows the motion of output style transfer. The second row (light gray character) shows the motion of output variational autoencoder without style autoencoder using the original style motion (figure 6, the second row, light gray character) as input data. The fourth row (light gray character) shows the motion of output variational autoencoder without style autoencoder using the original content motion (figure 6, the first row, light gray character) as input data.

9.2 다양체 사이의 연관성

Figure 8은 스타일 및 컨텐츠 모션에 대한 잠재 변수가 이루는 다양체가 서로 연관되어 있는 구조와 아닌 경우의 구조를 비교한 결과이다. 독립적인 구조를 나타내는 두번째 줄과 세번째 줄의 그래프는 각각 스타일 손실 함수의 가중치를 0.01, 200.0으로 설정한 결과이다. 가중치를 낮게 설정하면 컨텐츠 잠재 변수(초록색)내부에만 재구성된 잠재 변수(파란색)이 존재하며, 가중치를 높게 설정하면 연관되지 않는 잠재 공간 사이에 걸쳐서 재구성된 잠재 변수가 존재함을 알 수 있다. Figure 9는 figure 8에서 다양체가 서로 연관되어 있지 않고 독립적으로 구조를 이루는 경우의 모션 결과를 나타낸다. 초록색 캐릭터는 컨텐츠 원본 캐릭터이며, 밝은 회색 캐릭터는 스타일이 이전된 캐릭터이다. 첫째 줄은 스타일 손실 함수의 가중치를 0.01로 한 결과인데 컨텐츠와 거의 동일한 모션이 나옴을 알 수 있으며, 두번째 줄은 가중치를 200.0으로 설정한 결과인데 두번째 결과 화면에서 세번째 결과 화면으로 프레임이 이동할 때 불 연속적인 동작이 나옴을 알 수 있다. 이를 통해 스타일 및 컨텐츠 모션이 저차원으로 프로젝션 된 다양체가 서로 연관되어 있지 않으면 스타일 이전이 원활히 이루어지지 않음을 확인할 수 있다.



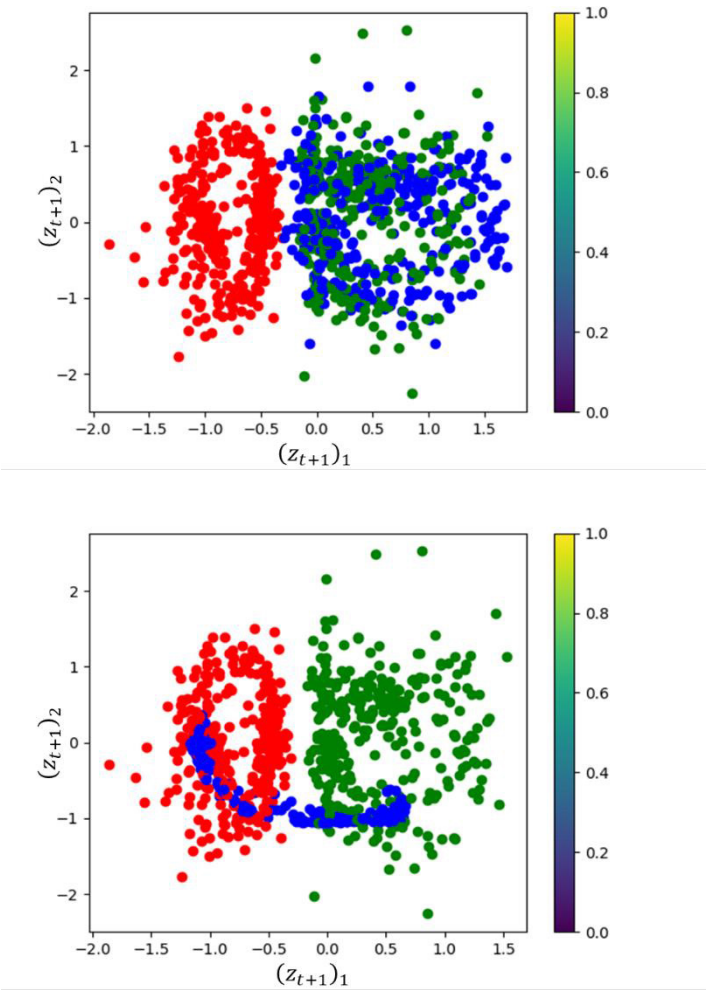


Figure 8: A scatter plot of latent variables. The green points represent the content motion. The red points represent the style motion. The blue points represent the motion of output style transfer.

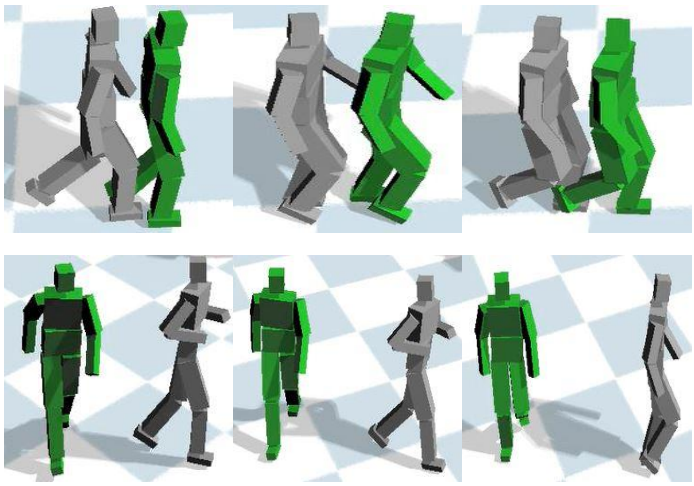


Figure 9: Snapshots of two motions that there is no correlation between latent variable of style and contents. The light gray character is the motion of output style transfer. The green character is content motion. **Top-to-bottom:** 0.01, 200.0 weight of style loss function.

9.3 속도

Figure 10은 t+1 프레임의 스타일이 입혀진 최종 모션을 생성할 때 t 프레임의 모션에 속도를 적분한 것과 속도를 사용하지 않고 네트워크의 출력 데이터의 모션 데이터를 그대로 사용한 것을 1 프레임 단위로 비교한 것이다. 결과를 보면 확인할 수 있듯이 속도를 적분한 경우 더 자연스럽게 모션이 흘러감을 알 수 있다.

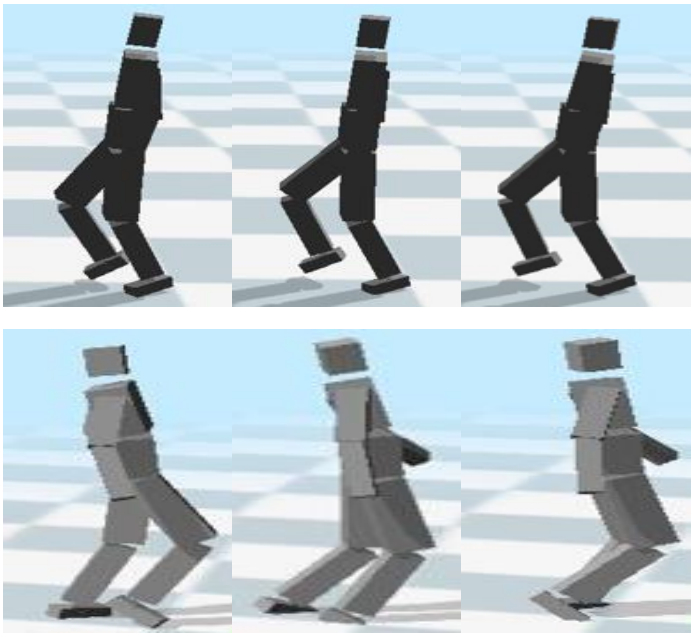


Figure 10: The first row (dark character) represents the result using velocity information. The second row (light gray character) represents the result of not using velocity information.

10. 평가

본 논문에서는 변형 자동 인코더 네트워크와 스타일 자동 인코더 네트워크를 활용해 잠재 공간 내에서 컨텐츠 캐릭터의 모션에 스타일 캐릭터 모션의 스타일 정보를 이전하여 변형 자동 인코더를 통한 모션의 다양성을 증가시켰다. 또한 모션의 속도 정보를 활용하여 더욱 자연스러운 모션을 생성하였다. 하지만 본 논문에서 제안하는 방식은 잠재 공간 내에서 컨텐츠 캐릭터의 모션과 스타일 캐릭터의 모션에 대한 다양체가 서로 연관되어 있지 않은 경우에 대해서 유용하지 않다. 즉, 걷는 모션과 뛰는 모션 같이 서로 완전히 스타일이 다른 모션들에 대해서는 스타일 이전이 원활히 이루어지지 않는 단점을 갖는다. 그러므로 이와 관련하여 잠재 공간 내에서 인코딩 된 두 모션에 대한 다양체 사이에서의 연결을 정의하여 완전히 다른 두 모션의 스타일 이전과 같은 추가적인 연구가 진행될 수 있다.

감사의 글

이 논문은 정부(과학기술정보통신부)의 재원으로 한국연구재단의 지원(NRF-2019R1A4A1029800) 및 정보통신기획평가원의 지원(No.2021-0-00320, 실 공간 대상 XR 생성 및 변형/증강 기술 개발)을 받아 수행된 연구임.

References

[1] Unuma, Munetoshi, Ken Anjyo, and Ryozo Takeuchi. "Fourier principles for emotion-based human figure animation." *Proceedings of the 22nd annual conference on Computer graphics and interactive techniques*, pp. 91–96, 1995.

[2] Bruderlin, Armin, and Lance Williams. "Motion signal processing." *Proceedings of the 22nd annual conference on Computer graphics and interactive techniques*, pp. 97–104, 1995.

[3] Pullen, Katherine, and Christoph Bregler. "Motion capture assisted animation: Texturing and synthesis." *Proceedings of the 29th annual conference on Computer graphics and interactive techniques*, pp. 501–508, 2002.

[4] Yumer, M. Ersin, and Niloy J. Mitra. "Spectral style transfer for human motion between independent actions." *ACM Transactions on Graphics (TOG)*, vol. 35, no. 4, pp. 1–8, 2016.

[5] Brand, Matthew, and Aaron Hertzmann. "Style machines." *Proceedings of the 27th annual conference on Computer graphics and interactive techniques*, pp. 183–192, 2000.

[6] Min, Jianyuan, Huajun Liu, and Jinxiang Chai. "Synthesis and editing of personalized stylistic human motion." *Proceedings of the 2010 ACM SIGGRAPH symposium on Interactive 3D Graphics and Games*, pp. 39–46, 2010.

[7] Holden, Daniel, et al. "Fast neural style transfer for motion data." *IEEE computer graphics and applications*, vol. 37, no. 4, pp. 42–49, 2017.

[8] Gatys, Leon A., Alexander S. Ecker, and Matthias Bethge. "A neural algorithm of artistic style." *arXiv preprint arXiv:1508.06576*, 2015.

[9] Du, Han, et al. "Stylistic Locomotion Modeling with Conditional Variational Autoencoder." *Eurographics (Short Papers)*, pp. 9–12, 2019.

[10] Bowden, Richard. "Learning statistical models of human motion." *IEEE Workshop on Human Modeling, Analysis and Synthesis, CVPR*, vol. 2000, 2000.

[11] Min, Jianyuan, and Jinxiang Chai. "Motion graphs++ a compact generative model for semantic motion analysis and synthesis." *ACM Transactions on Graphics (TOG)*, vol. 31, no. 6, pp. 1–12, 2012.

[12] Holden, Daniel, et al. "Learning motion manifolds with convolutional autoencoders." *SIGGRAPH Asia 2015 Technical Briefs*, pp. 1–4, 2015.

[13] Motegi, Yuichiro, Yuma Hijioka, and Makoto Murakami. "Human motion generative model using variational autoencoder." *International Journal of Modeling and Optimization*, vol. 8, no. 1, 2018.

[14] Habibie, Ikhsanul, et al. "A recurrent variational autoencoder for human motion synthesis." *28th British Machine Vision Conference*, 2017.

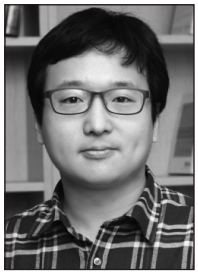
[15] Fragkiadaki, Katerina, et al. "Recurrent network models for human dynamics." *Proceedings of the IEEE International Conference on Computer Vision*, pp. 4346–4354, 2015.

[16] Kingma, Diederik P., and Max Welling. "Auto-encoding variational bayes." *arXiv preprint arXiv:1312.6114*, 2013.

〈저자소개〉

안 제 원

- 2015-2019 한양대학교 기계공학부 학사
- 2020-현재 한양대학교 지능융합학과 석사
- 관심 분야: Motion style transfer, Deep learning techniques, Physically-Based Character Control.
- https://orcid.org/0000-0002-4151-5372



권 태 수

- 1996-2000 서울대학교 전기컴퓨터공학부 학사
- 2000-2002 서울대학교 전기컴퓨터공학부 석사
- 2002-2007 한국과학기술원 전산학전공 박사
- 관심 분야: Physics-based models, Machine learning techniques.
- https://orcid.org/0000-0002-9253-2156