2020 년 가을 POSTECH 컴퓨터공학과 과제연구 최종 보고서

[3D Garment Reconstruction from 2D Image to Humanoid Avatar]

학 번: 20170702
이 름: 손주은
연구 지도교수: 이승용 교수님
학 과: 컴퓨터공학과

연구 목적(Problem Statement)

본 연구의 주제는 사용자가 입력한 garment image 를 input humanoid 가 착용할 수 있는 형태로 reconstruct 하는 것이다. 옷 가지 수는 티셔츠와 긴 바지의 두 종류로 제한하며, armature 가 있는 obj 및 fbx 파일 형태의 input humanoid model 에 대해 동작하는 것을 확인했다.

본 연구는 VR 게임 등에서 의복을 착용한 게임 캐릭터를 대량생산할 때 유용하게 사용될 수 있을 것이다. 게임 캐릭터의 복장은 게임 내 몰입도를 높이는데 중요한 역할을 하는데, 캐릭터 하나하나의 의복을 모델링하는 것은 전문적인 지식과 기술을 필요로 하는 일이다. 본 프로그램을 활용을 하면 많은 시간과 자원을 절약할 수 있을 것으로 예상한다.

연구 배경(Motivation and Background)

본 연구는 garment reconstruction 및 garment transfer 분야를 바탕으로 한다. Garment Capture from a photograph [1] 의 연구진들은 옷을 착용한 마네킹의 사진을 이용해 옷의 pattern draft 를 자동으로 만들어주는 연구를 진행했다. 의류학에서의 pattern draft theory 를 이용했다는 점에서 참신하나 one-repeat texture 만을 extract 할 수 있다는 점과 마네킹이 착용한 것을 촬영한 이미지가 구하기 쉽지 않다는 점에서 응용력이 떨어진다고 할 수 있다.

DeepFashion3D: A dataset and benchmark for 3d garment reconstruction from single images [2] 의 연구진들은 딥러닝 기술을 이용해 single-image 만으로 garment geometry 를

reconstruct 하는 연구를 진행했다. Template garment mesh 를 사용하지 않아 geometric detail 을 잘 보존할 수 있다는 점에서 의의가 있으나 texture extraction 은 하지 않았다는 한계가 있다.

3D Virtual Garment Modeling from RGB Images [3] 의 연구진들은 딥러닝을 이용해 garment landmark 를 알아내고, 이를 바탕으로 template garment 를 free-form deformation 방식으로 deform 하고 texture extraction 을 진행했다. 본 연구에서도 landmark 정보를 바탕으로 texture extraction 을 진행하지만, 단순히 garment modeling 에서 끝나지 않고 input humanoid 의 몸에 맞게 변형한다.

Customizing 3D garments based on volumetric deformation [4] 의 연구진들은 Volumetric Graph Laplacian 을 이용해 기존의 garment model 을 deform 함으로써 input avatar 에 맞게 의복을 변형했다. Cross-section 을 sampling 해 기존에 옷을 착용하고 있던 모델과 input 모델 간의 mapping 을 구하고, 이를 바탕으로 따로 만들어준 tetrahedral mesh 를 deform 함으로써 input model 에 맞는 garment 를 구한다. 본 연구에서는 cross-section 을 통해 sample point 를 구하는 방법을 이용하나, tetrahedral mesh 를 통해 간접적으로 garment 를 변형하지 않고 직접 mesh geometry 에 control point 를 추가해 laplacian deformation 을 적용한다.

연구 방법(Design and Implementation)

본 연구에서는 사용자가 입력한 garment image 를 반영한 garment 를 recreate 하기 위해 template garment mesh 를 deform 하는 방식을 이용한다. 또한 딥러닝을 활용해 사진으로부터 얻어낸 garment landmark 정보를 이용해 texture map 으로 이미지를 옮긴 뒤, deform 된 garment mesh 에 texture 를 적용할 수 있다. Template Mesh Preparation, Texture Extraction, Garment Transfer 의 세 단계로 나누어볼 수 있다.

Template Mesh Preparation

Input humanoid model 의 몸에 맞는 옷을 만들어주기 위해 template garment mesh 를 deform 하는 방식을 택한다. Default humanoid model 는 Free3D [5]라는 3d 모델 사이트를 통해 구했다. 티셔츠와 바지의 3D 모델은 Berkeley garment libraries 에서 제공한 tshirt 와 trousers 모델을 사용한다. 두 garment model 을 Blender 를 이용해 default humanoid 가 착용할 수 있도록 수작업으로 mesh 를 edit 해주었다. 각각의 garment 모델에 대해 한 번씩만 이

작업을 진행해준다. 이 단계가 완료되면 default humanoid 와 그에 맞는 tshirt 및 trousers 모델이 준비된다.

Figure 1 Default model 이 tshirt 및 trousers 을 착용한 모습이다

Texture Extraction

Texture Extraction 단계는 사용자가 입력한 garment image 를 garment model 에 바로 적용할 수 있는 texture map 으로 변환하는 단계이다. UV unwrapping, Garment Landmark Detection, Image Transfer to Texture Map 으로 나누어볼 수 있다.

UV map 이란 3d 모델 표면의 texture 정보를 2 차원 이미지로 나타낸 것으로, 각 vertex 가 어느 point 에 대응되는지 정보를 담고 있다. 3d 모델로부터 이러한 uv map 을 구하는 과정을 uv unwrapping 이라고 한다. Blender 를 통해 아래 그림과 같이 각 garment model 의 uv map 을 구해주었다. 이후 단계에서 garment 를 구성하는 vertex 를 옮겨도 큰 왜곡 없이 같은 texture map 을 적용시킬 수 있다.

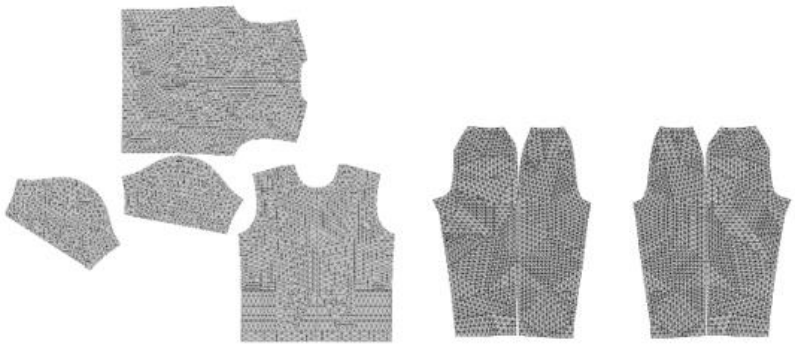

Figure 2 티셔츠와 바지의 UV Map 이다

다음으로 사용자가 입력한 garment 사진으로부터 texture map 을 만들어준다. 딥러닝을 이용한 garment landmark detection 을 통해 garment image 에서의 landmark 정보를 얻어낸다. 티셔츠의 경우 소매 시작점, 몸통 중간 지점 등의 landmark 가 있다. 이러한 landmark 를 이용해 원본 garment 이미지를 조각조각 deform 함으로써 texture map 을 만든다.

Garment landmark detection network 의 경우 직접 train 하지 않고 Jingdong Wang 과 연구진들의 모델 [6] 을 사용한다. 이 모델을 통해 티셔츠 이미지의 경우 25 개, 긴 바지의 경우 14 개의 landmark 좌표를 얻을 수 있다. 그런 다음 각 vertex 의 uv map 에서의 대응 위치를 수작업으로 잡아준 뒤, OpenCV 라이브러리의 affine transformation 기능을 통해 garment image 의 triangle 들을 texture map 으로 옮겨준다.

Figure 3 티셔츠와 바지 이미지로부터 texture map 을 만든 모습이다.

Garment Transfer

Garment Transfer 단계는 default humanoid 에 맞던 template garment mesh 를 input humanoid 의 몸에 맞게 변형하는 단계이다. 크게 Segmentation, Alignment, Sampling, Ray-casting, Laplacian Deformation 단계로 나눌 수 있다. 구현은 Blender Python Scripting API 를 이용해 진행한다.

Segmentation

본격적 pipeline 을 위한 준비 단계로 Segmentation 을 진행한다. Segmentation 이란 model 의 각 vertex 가 어느 segment 에 속하는지 group 을 정해주는 것이다. 두 garment model 과 humanoid model 들에 대해 아래 표대로 segmentation 을 진행해주었다. Segmentation 이 완료되면, 아무 group 에도 속하지 않는 vertex 들, 즉 target garment 과 관련이 없는 영역은 삭제를 함으로써 region of interest 를 뽑아낸다.

Garment Type	Segmentation	Etc.
Tshirt	TORSO UPPER ARM (R, L)	Avatar 의 두 shoulder joint 을 경계로 가운데 영역은 torso, 바깥에 속한 vertex 는 좌표에 따라 left 와 right arm 로 나누어주었다.
Pants	WAIST UPPER LEG (R, L) LOWER LEG (R, L)	Avatar 의 pelvis joint 와 knee joint 의 높이를 경계로 위에서부터 차례대로 waist, upper leg, lower leg 으로 분류해주었다. Tshirt 와 마찬가지로 right, left 에 따른 분류도 해준다.

Figure 4 Segmentation Chart

Alignment

Aligning 은 세세한 변형을 진행하기에 앞서 humanoid 간의 개략적인 pose 를 맞춰주는 단계이다. 두 humanoid 의 armature 정보를 통해 Default humanoid 를 input humanoid pose 에 맞추고, 이때 default humanoid 에 적용한 transformation 을 garment 에 그대로 적용함으로써 일차적인 deform 이 진행된다. 같은 transformation 을 적용하기 때문에 Alignment 단계 진행 후에도 default humanoid model 과 garment model 사이에는 일정한 spacing 이 유지된다. Transformation 은 각 segment 별로 Figure 5 의 표대로 적용한다.

*두 humanoid 모두 y 축을 향하고 있고, 높이는 z 축, 왼쪽을 -x 축을 향하고 있다 가정한다.

Segment	Transformation
TORSO	✧ X-axis SCALE: 어깨 너비 맞추기 위함 ✧ Z-axis SCALE: 척추 길이 맞추기 위함 ✧ TRANSLATION: Neck joint 의 위치 맞추기 위함
ARM	✧ TORSO 에 적용한 transformation 을 ARM 에도 적용 ✧ YZ SHEAR: UPPER ARM 의 각도 (Elbow joint 의 위치)를 맞추기 위함 ✧ X-axis SCALE: UPPER ARM 의 길이 맞추기 위함
**PANTS_PARENT	✧ X-axis SCALE: 골반 너비 맞추기 위함 ✧ TRANSLATION: Hip Joint 위치 맞추기 위함
WAIST	✧ PANTS_PARENT 적용 ✧ Z-axis SCALE: 허리 길이 맞추기 위함
UPPER LEG	✧ PANTS_PARENT 적용 ✧ Z-axis SCALE: 허벅지 길이 맞추기 위함 ✧ XY SHEAR: Knee joint 의 위치 맞추기 위함
LOWER LEG	✧ UPPER_LEG 에 적용한 transformation 을 LOWER LEG 에도 적용 ✧ Z-axis SCALE: 종아리 길이 맞추기 위함 ✧ XY SHEAR: Ankle joint 의 위치 맞추기 위함

**PANTS_PARENT 는 segment 는 아니지만 pants 하위 segment 에 해당 transformation 을 공통으로 적용

Figure 5 Segment 별로 가하는 transformation

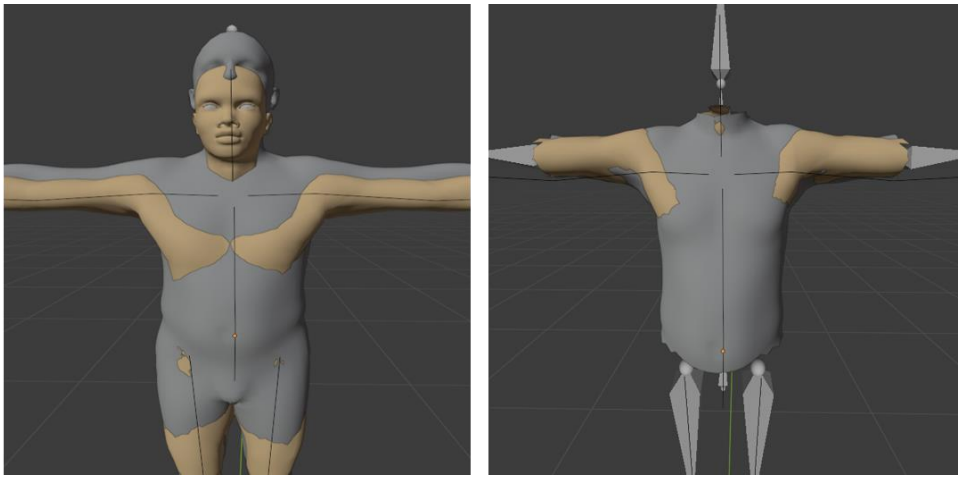

Figure 6 Default(베이지) 모델이 input(회색)과 align 된 모습이다

Sampling

두 humanoid model 의 신체적 특징을 보다 세부적으로 파악하기 위해 sampling 단계를 진행한다. Model 의 bone structure 를 따라 plane 을 여러 개 정의하면, 각 plane 별로 model 과의 intersect 하는 curve 인 cross-section curve 가 생긴다. 이 curve 를 N 등분했을때의 각 교점을 sample points 로 삼는다. Curve 를 구성하는 vertex 의 무게중심을 구하고, 무게중심을 지나고 사용자가 바라보고 있는 정면을 기준으로 N 에 따라 등각으로 평면을 회전시키면서 curve 와의 교점을 sampling point 로 잡는다. 본 연구에서는 N=18 으로 잡았다. 같은 방식으로 default 와 input humanoid 에 대해 sampling 을 진행해줌으로써 각 point 별로 일대일대응이 생기게 되는데, 두 point 의 position difference 를 이용해 이후 단계에서 deformation 을 진행한다.

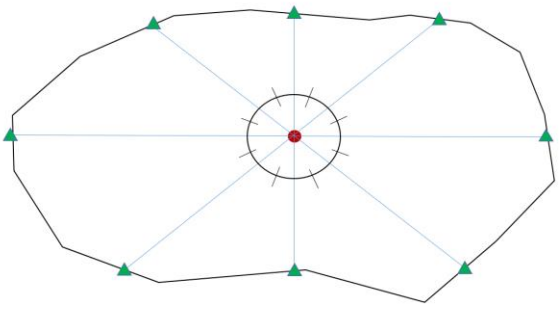

Figure 7 무게중심(빨간 점)을 기준으로 equiangular 하게 cross-section curve 를 나눈 모습으로, sample point 는 초록 삼각형으로 표시했다. (N=8 인 경우)

현재는 직접 사용자가 sampling 에 이용되는 plane 을 정의해야한다. Alignment 단계를 마친 상태이기 때문에 두 humanoid model 의 기본적인 골격은 크게 다르지 않다고 가정하고 sampling 단계를 진행한다. 상의와 하의 각각 plane 의 개수는 아래와 같이 14 개, 8 개로 잡아주었다.

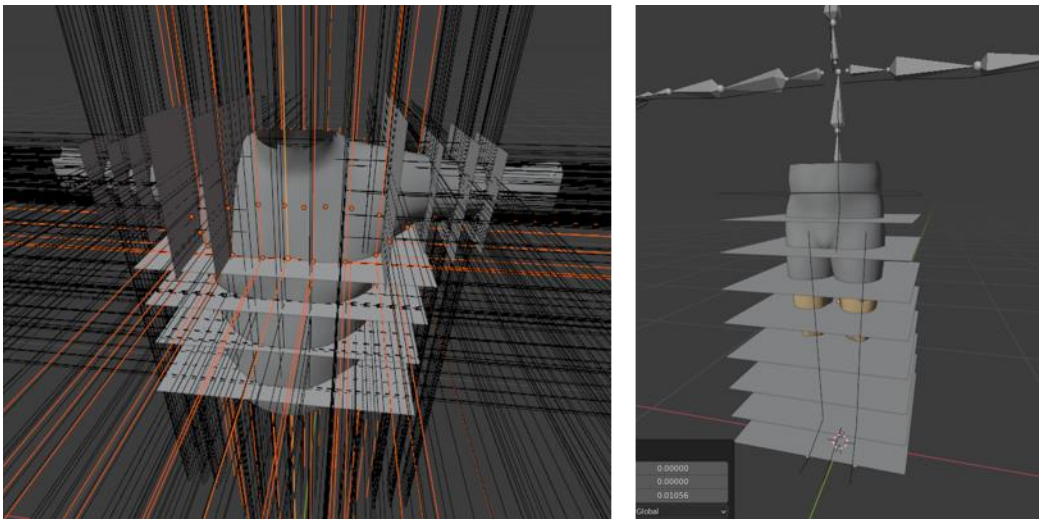

Figure 8 상의와 하의의 sampling plane 및 sampling 을 진행하는 모습이다

Ray-Casting

위의 sampling 단계가 두 humanoid model 의 sample point 를 구하는 과정이었다면 ray casting 은 garment model 을 sampling 하는 단계이다. Ray casting 은 ray 와 surface 의 intersection 을 확인하는 알고리즘인데, 이 방법을 통해 garment transfer 의 마지막 단계인 laplacian deformation 에서 사용될 control point 를 구한다.

이전 단계에서 구해진 default humanoid model 의 각 sample points 에 대해 humanoid 에서의 각 vertex 의 normal 방향을 향해 garment model 로의 ray-casting 을 진행한다. 이때 생기는 교점이 control point 가 되는 것이다. Alignment 가 완료된 뒤에도 default humanoid model 과 garment model 간의 spacing 이 유지되기 때문에 항상 ray casting 결과가 존재함을 가정할 수 있다.

Laplacian Deformation

Laplacian Deformation [7] 이란 surface 의 geometric detail 을 최대한 보존하면서 control points 를 기준으로 deform 하는 방법이다. 위의 ray-cast 단계에서 deformation 이전 control points 를 설정해주었다. 각 control point 의 target position 은 다음과 같은 식을 이용해 구할 수 있다.

$$\text{Target} = \text{hook} + (\text{sampled\_input} - \text{sampled\_default})$$

Hook 의 좌표에 input 과 default 의 sample points 의 차를 더함으로써 각 vertex 의 target 좌표를 구할 수 있다.

연구 결과 및 평가 (Methodology and Evaluation)

Figure 9 은 본 연구에서 사용된 default 와 input humanoid 모델의 사진이다. Figure 10 은 default humanoid 의 몸에 맞던 garment template 을 input 이 착용할 수 있도록 deform 한 모습이다. Default 모델이 garment 을 착용했을 때 두 mesh 간에 충돌이 없었지만 garment deformation 후 input 과 garment 가 충돌하는 부분이 생김을 확인할 수 있다. (파란색 살이 튀어나온 모습)

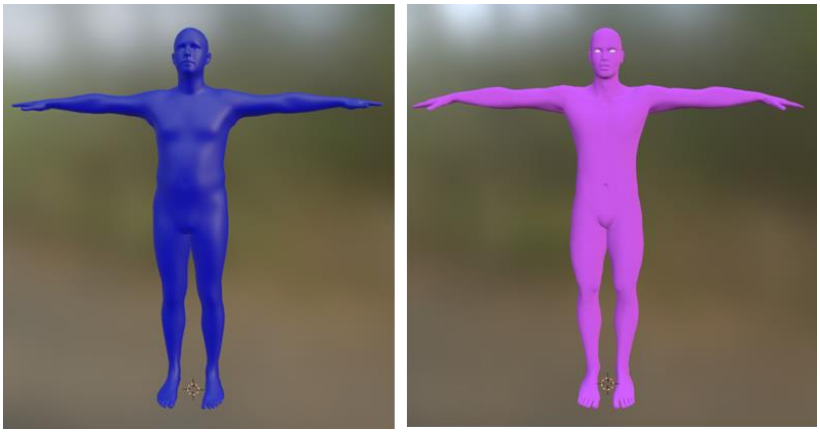

Figure 9. Input 과 Default mesh 의 모습

Figure 10. 하의와 상의를 input model 에 맞게 deform 한 결과이다.

티셔츠보다 바지에서 더욱 심하게 spacing 이 망가지는 것을 확인할 수 있는데 허리와 허벅지 부분이 그러하다. 허벅지의 경우 다리가 붙어있는 모델들을 사용했기 때문에 비교적 단순한 구조였던 상의에서와는 달리 자체적으로 충돌이 일어나고 있다. 또한 허리 부분의 경우에는 상의와 하의에 걸쳐있는 부분인데, 상의와 겹치지 않게 무리해서 하의를 humanoid 에 밀착시켜 trousers template mesh 를 수정해 나타난 현상으로 볼 수 있다. Humanoid 와 하의에 적당한 spacing 을 만들어준 뒤, 하의와 상의에 spacing 을 주는 순으로 작업을 했다면 더욱 정밀한 결과를 얻을 수 있었을 것이다.

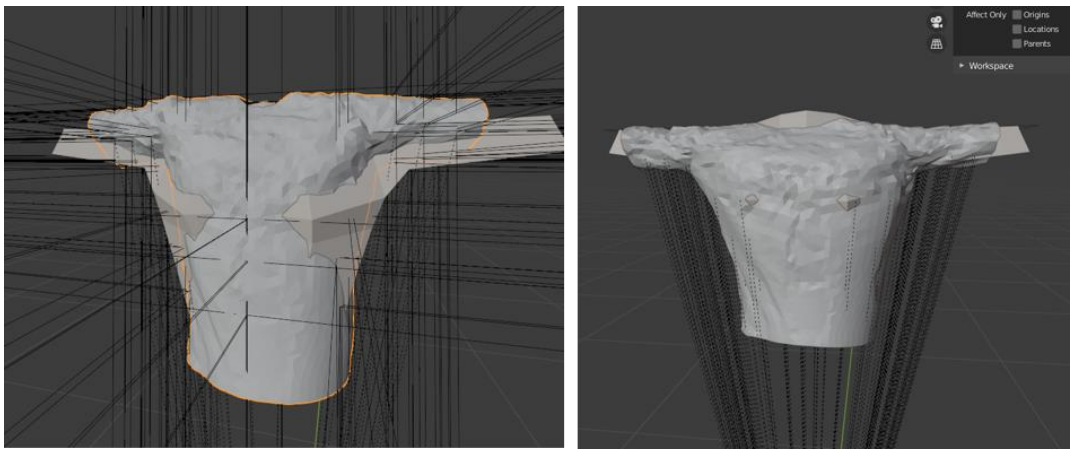

Figure 11. N 과 plane 의 수를 달리해가며 deform 해본 결과로, N 과 plane 의 수가 클수록 더욱 정밀한 deform 결과를 얻을 수 있음을 확인할 수 있다.

Figure 11 은 low poly input model 에 대해 N (한 curve 당 sample point 의 수) 의 값을 바꿔가며 deform 해본 결과이다. 첫번째가 N=6 인 경우, 두번째가 N=18 인 경우이다. 후자인 경우에는 sampling plane 의 수도 segment 당 하나씩 늘려주었다. 후자의 경우, 여전히

garment 와 model 간의 spacing 이 완전히 보존되지는 않았지만 전자에 비해 humanoid 와 충돌하는 부분이 더 적은 것을 확인할 수 있다.

Figure 12. Texture Extraction 과 Garment Transfer 단계를 합친 결과이다.

Figure 12 는 texture extraction 결과와 garment transfer 단계를 합친 결과이다. Blender 를 이용해 garment model 에 texture 를 적용해 rendering 했다. Figure 13 는 이를 두 garment 모델을 humanoid model 과 함께 armature 에 연결해 animation 을 적용한 모습이다. 다양한 포즈에도 humanoid 와 함께 garment 가 deform 되는 모습을 확인할 수 있다.

Figure 13. Unity 에서 garment 를 착용한 humanoid 에 animation 을 적용한 모습이다

토론 및 전망 (Discussion and Future Work)

본 연구에서는 사용자가 입력한 garment image 를 input humanoid avatar 에 몸에 맞게 재구성하는 프로그램을 만들어주었다.

현재 template garment mesh 를 deform 함으로써 input humanoid 에 맞는 옷을 만들어주는데, 옷에 따른 garment geometry 의 차이는 없다고 할 수 있다. 본 연구에서 사용된 티셔츠와 긴 바지는 비교적 단순한 garment type 이기 때문에 texture 만으로도 이미지의 느낌을 잘 전달할 수 있지만 원피스 등 모양이 훨씬 다양한 옷의 경우에는 geometry 도 차이가 있어야할 것으로 생각된다. Texture extraction 을 위해 진행했던 garment landmark detection 결과를 활용해 landmark 간의 거리 관계를 이용해 template garment mesh 를 deform 하는 방법을 고려해볼 수 있을 것이다.

Garment Transfer 의 sampling 단계에서 plane 을 사용자가 직접 설정해야한다는 점은 보완할 여지가 있다. 현재 직접 설정해줌으로써 input avatar 에 더욱 적합한 cross-section 을 구할 수 있다는 장점이 있지만 사용자가 수작업으로 해야하는 일이기 때문에 자동화한다면 프로그램의 완성도를 높일 수 있을 것이다. 두 mesh 의 bone structure 를 이용하면 쉽게 구현할 수 있을 것으로 예상한다. 한 bone 과 나란한 방향으로 M 등분을 해서 cross section plane 을 잡아주는 방법이 있을 것이다.

또한 현재 최종 deform 된 garment 의 경우 몇몇 부분에서는 default humanoid 와 garment 간의 spacing 이 제대로 보존되지 않았다. 이것을 고칠 수 있는 방법으로 두가지를 제시해본다. 먼저 sampling plane 및 각 curve 마다 sampling 하는 point 의 수 (N) 를 늘리는 방법이다. 위 연구 결과에서도 언급했듯이, sampling plane 수 및 N 을 더 늘린다면 spacing 는 더욱 잘 보존될 것으로 예상한다.

또한 현재 garment transfer 의 laplacian deformation 단계에서 control point 의 target 좌표를 계산할 때 input 과 default humanoid 에서의 sample point 의 difference 만큼을 그대로 더해주었다. 하지만 여기에 변수 알파 (α)를 추가해 1.5 등으로 잡은 뒤 (현재는 1.0 인 셈이다) difference 에 곱해준 뒤 target 을 계산하면 spacing 문제를 해결할 수 있을 것으로 보인다.

참고 문헌

[1] M.-H. Jeong, D.-H. Han 그리고 H.-S. Ko, "Garment capture from a photograph," %1 *Computer Animation and Virtual Worlds*, 2015.

[2] H. Zhu, "Deep Fashion 3D: A Dataset and Benchmark for 3D Garment Reconstruction from Single Images," *arXiv prepint arXiv*, 제 2003, 번호: 12753, 2020.

[3] Y. Xu, "3d virtual garment modeling from rgb images," *IEEE International Symposium on Mixed and Augmented Reality (ISMAR)*, 2019.

[4] J. Li 그리고 G. Lu, "Customizing 3D Garments based on volumetric deformation," *Computers in Industry*, 제 62, 번호: 7, pp. 693-707, 2011.

[5] rito-kun, "free3d.com," [온라인]. Available: https://free3d.com/ko/3d-model/human-base-riged-85678.html. [액세스: 16 12 2020].

[6] J. Wang, K. Sun, T. Cheng, B. Jiang, C. Deng, Y. Zhao, D. Liu, Y. Mu, M. Tan, X. Wang, W. Liu 그리고 B. Xiao, "Deep High-Resolution Representation Learning for Visual Recognition," *Computer Vision and Pattern Recognition*, 2019.

[7] O. Sorkine, "Laplacian Surface Editing," *Eurographics Symposium on Geometry Processing*, 2004.