Format-Compatible 3D Metahuman Modeling from a Single Image

Sojin Yun[1], Soyoung Yoon[1], and In Kyu Park[1][0000-0003-4774-7841]

Department of Information and Communication Engineering, Inha University
Incheon 22212, Korea
sj0524sj@gmail.com, thdud679@gmail.com, pik@inha.ac.kr

Abstract. In this paper, we propose a method for generating 3D metahumans reflecting the body shape and clothes texture estimated from a single general image. The body and clothes mesh are automatically transformed in the blender, using body shape estimates obtained through SMPL-X. The obtained clothes' UV Map and mesh reflecting the body shape are automatically called to the Unreal Engine to create a metahuman. The generated metahuman can be seen as an animation implemented on Blueprint of the Unreal engine.

Keywords: Format-compatible · 3D metahuman · Single image.

1 Introduction

Metaverse is a combination of Meta and Universe, and means a virtual world linked to reality. Recently, more and more people are communicating in digital spaces since COVID-19, centering on the generation familiar with the digital environment. In the current existing metaverse, it is difficult to create the same metahuman that is 3D Virtual Humans on metaverse, because you choose the limited created type of characters and clothes. Therefore, this work proposes a 3D metahuman generation method from one's own single image.

Recent studies [13, 15] have modeled clothes with the body volumetrically using voxels or point cloud, which is not compatible with game engines and authoring tools. In this paper, metahumans can be imported to Unreal engines, the commonly used game engines. Our previous work [7] estimates the body type with one image and models the clothes to generate metahumans, but there are limitations in research. The gender and clothes categories of the model are limited, so various types of characters can't be reflected. In addition, UV Map generation is not accurate for images that are not frontal and static. Also, both the body shape deformation code in the blender and the metahuman generation and animation in the unreal engine are manually performed. This study aims to make up for these limitations.

The main contributions can be summarized as follows.

– A proof of concept for format-compatible and animatable 3D metahuman generation from a single image
– Automatic shape and texture modeling for widely-used commercial game engines, *e.g.* Unreal engines

2 Proposed Method

Fig. 1 shows the proposed pipeline that generates 3D metahumans through body shape and clothes texture extracted from a single image.

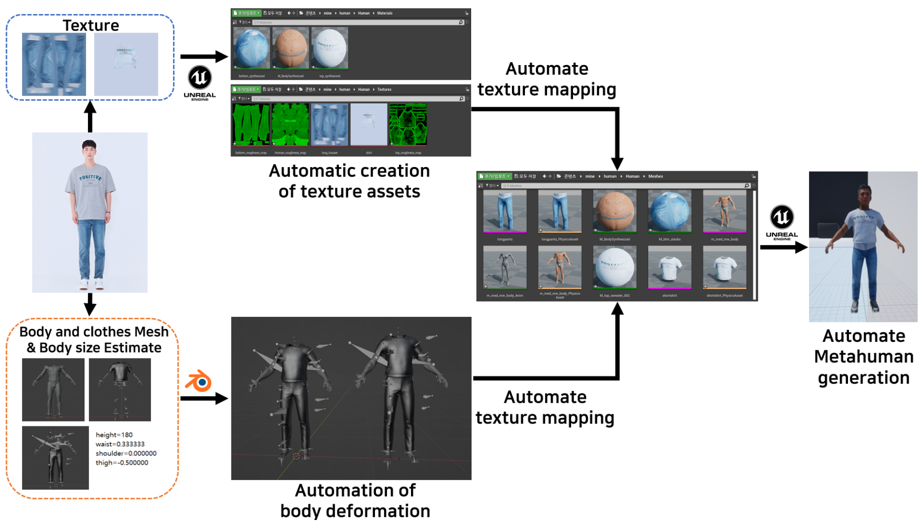

Fig. 1: 3D metahuman generation pipeline with body and clothing textures estimated from a single image. Test image is from [1].

2.1 3D Face Reconstruction

We follow the general pipeline that was proposed in our previous work [10]. Basic face rendering features utilize 3DMM [5] and CNN. Landmarks are detected on faces, and 3DMM coefficients are estimated using them. Alignment between 2D and 3D landmarks is performed via RBF to restore a rough 3D face. With CNN-based fine grained depth information estimation, wrinkles and other elements are included in the 3D face. Face texture is also generated using landmarks and CNN by removing hair and ears and filling the background of the image with skin color and texture. It generates a low resolution diffuse reflection texture from a single basic face image. [10]

Additionally, cavity map, normal map, and roughness map are used for natural and realistic face rendering. When using only one sheet of reflection texture, there is a problem that is not realistic when texture mapping to the face mesh model. Therefore, additional cavity map and normal map representing light reflection and skin irregularities, and roughness maps representing the resulting skin roughness, are used. Through the above additional information, realistic skin expression that the face mesh model cannot express is possible. [8, 9]

2.2 3D Body Reconstruction

We follow the general pipeline that was proposed in our previous work [14]. Extracting body type information from a single image utilizes SMPL [11] and SMPL-X [12]. Detailed information of the person is estimated using SMPL-X from the input image. It is converted to SMPL for measuring body type parameters and posture parameters of body models. In this conversion, the SMPL model modified to a straight posture is obtained by changing only the value of a specific posture parameter without converting the body shape parameter. Using the SMPL model, which modified the posture, the dimensions of the height, shoulder, waist, and thigh set as the main parts of the body are measured. The basic body model provided by the metahuman creator is converted through the measured size. [9, 14]

We use body size estimates to deform the mesh of the body and clothes. The body and clothes files provided by the Metahuman Creator consist of mesh and skeleton. For each skeleton, an affected set of vertices is defined. As the skeleton rotates, the affected set of vertices moves together. Therefore, all vertex sets defined in the skeleton set are calculated and selected. Thereafter, the deformation is performed on the selected set of vertices. The Python code of [7] deforms the mesh in the blender. Adjusting the height uses a function that scales in the z-axis direction relative to the origin. For body parts other than height, the operation of reducing or enlarging the surface based on the normal vector is performed.

By adding from [7], the import and export process of the mesh file is automatically performed, and the mesh deformation is also automatically performed. The biggest difference from the existing method is that it transforms the necessary mesh files one by one. Using a method of importing and deforming all meshes at once, it is impossible to map textures separately from the Unreal engine because the export function stores all the meshes currently in place. Therefore, the mesh for the body, top, and bottom is all recalled separately, and each is deformed and exported. In between this process, each modified and exported mesh is deleted so that it does not affect other meshes.

2.3 3D Cloth Reconstruction

The basic costume clothes mesh uses a Metahuman Creator. Various clothes that are not provided, such as skirts, dresses, and outerwear, are produced using 3D tools such as Marvelous Designer and Blender. The basic metahuman is set as an avatar on Marvelous Designer and clothes are fitted to the body. Metahuman's skeleton is planted in this, combined with the costume, and weight paint is performed on each bone to enable costume animation.

From the input image, the clothes segmentation is carried out into 13 costume categories provided by Deepfashion2 [6]. From this result, keypoint tailored to the characteristics of each costume is detected. With the coordinate information of the keypoint, perspective transformation mapping is performed with coordinates to be mapped to the UV Map. The texture of the invisible side is reflected in the UV Map by distinguishing whether it is pattern or logo-centered. [16]

2.4 3D Metahuman Generation

First, we perform the task of loading files of various types of extensions, such as obj, fbx, jpg, and png. Textures have extensions such as jpg, png, and bmp, and are stored in the 'Textures' folder upon loading.

The imported texture file is used to create the Synthesized Asset. Synthesized asset is a type of material instance that allows for more sophisticated texture mapping by setting normal, roughness, and cavity. The created synthesized asset is stored in the "Materials" folder, and the mesh of the body, top, and bottom is stored in the "Meshes" folder. Each mesh is mapped in texture with a set synthesized set. The final metahuman is implemented in the viewport of the Unreal engine. The final metahuman is implemented in the viewport of the Unreal Engine by reflecting the position and rotation. At this time, meshes such as shoes, faces, and hair are also called in to create the final metahuman [4].

3 Experimental Results

The methods proposed by the paper are automatic conversion of body and clothes mesh and automatic generation of metahuman within an unreal engine. Body and clothes mesh is converted separately and then stored to facilitate texture mapping when meta-humans were generated. Mesh to which the texture is mapped is created in Viewport of the Unreal Engine where you can check the appearance and animation of the asset, to confirmation of the final appearance.

Fig. 2 shows the experimental results of the method proposed by the paper. Unlike previous studies [7], the experiment is conducted with a general image. UV Maps without mixing background colors is created even though they is not static, and female models and clothes such as skirts is added to create more diverse metahumans. Also, unlike the manual experiments on the blender and unreal engines, metahumans are generated through automation codes. When using the automation process, body shape conversion takes about 4 seconds and metahuman generation takes about 30 seconds. Using the existing code [7], body shape conversion takes about 10 seconds each for the body, top, and bottom, taking a total of 30 seconds. Metahuman generation takes about 5 minutes if you are familiar with the Unreal engine because all the work is done manually. Using the code proposed by the paper, time can be greatly reduced compared to previous studies and the results can be easily checked.

Animation can be implemented because it uses the same skeleton as the existing metahuman. However, the animation is done manually on Blueprint. Blueprint that implements the animation of the basic metahuman is duplicated to add the necessary body and costume assets to the component. The same animation is applied by setting it to the child class where the legging work has been done. Application of animation to the generated metahuman can be found in Fig. 3.

The metahuman's face and hands generated tend to be larger than the entire body. Through SMPL and SMPL-X, the body and costume mesh is converted

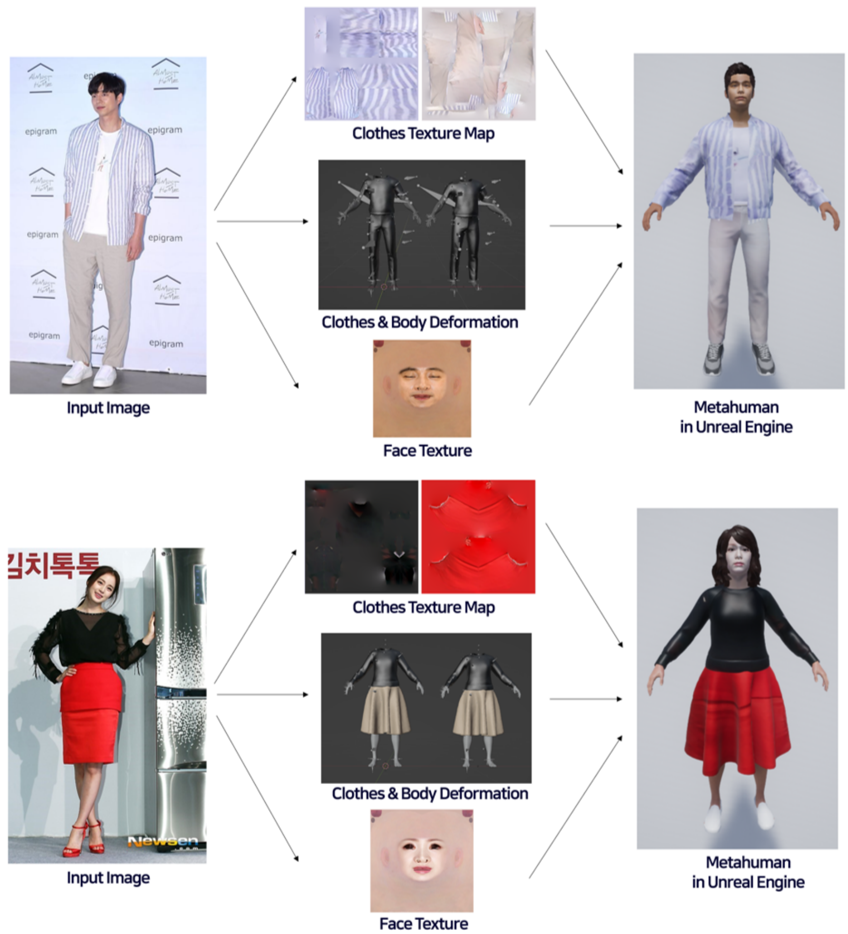

Fig. 2: Experimental results on typical images. Extract costume UV Map and body type estimates from general images. Body estimates are used for body shape deformation of body mash. The final metahuman is created by mapping the clothes UV Map to the deformed body mesh. In this process, body shape transformation and metahuman generation are implemented by automation code. Test images are from [2, 3]

(a) (b)

Fig. 3: Application of animation to the generated metahuman. It uses the same skeleton as the basic metahuman, so that basic metahuman animations such as running and jumping can be executed through the direction keys. (a) Gong Yoo. (b) Kim Tae Hee

from the body dimensions of the input image, but the accuracy is poor because the ratio of the basic metahuman is not correct. In addition, it tends to be less accurate because the face texture is applied to the face mesh of the existing metahuman, not the face mesh used in [10], for the animation of the face.

4 Conclusion

In this paper, we proposed a method for generating 3D metahumans through body shape and clothes textures estimated from a single image. The resultant model could be imported onto a popular game engine like Unreal and authoring tool like Maya for metaverse applications. The model was observed to be animatable on those platforms.

Acknowledgements This work was partly supported by the National Research Foundation of Korea (NRF) grant funded by the Korea government (MSIT) (No. 2022R1A4A1033549). This work was partly supported by Institute of Information & communications Technology Planning & Evaluation (IITP) grant funded by the Korea government (MSIT) (No. 2022-0-00981, Foreground and background matching 3D object streaming technology development and No.RS-2022-00155915, Artificial Intelligence Convergence Innovation Human Resources Development (Inha University)).

References

1. https://www.musinsa.com/app/goods/2546982, Accessed on December 12, 2022
2. https://www.newsen.com/news_view.php?uid=201310311519262910, Accessed on December 12, 2022
3. http://osen.mt.co.kr/article/G1110913218, Accessed on December 12, 2022
4. Unreal Python API Documentation https://docs.unrealengine.com/4.27/en-US/PythonAPI/, accessed August 24, 2022
5. Blanz, V., Romdhani, S., Vetter, T.: Face identification across different poses and illuminations with a 3D morphable model. In: Proc. of IEEE International Conference on Automatic Face and Gesture Recognition. p. 202–207 (August 2002)
6. Ge, Y., Zhang, R.and Wang, X., Tang, X., Luo, P.: Deepfashion2: a versatile benchmark for detection, pose estimation. In: Proc. IEEE/CVF Conference on Computer Vision and Pattern Recognition(CVPR). p. 5337–5345 (2019)
7. Kim, H.W., Kim, D.E., Kim, Y., Park, I.K.: 3D clothes modeling of virtual human for metaverse. Journal of Broadcast Engineering **27**(5), 638–653 (December 2022)
8. Kim, Y.: A study on multimodal facial texture generation for 3d human modeling. Master's thesis, Inha University (August 2022)
9. Kim, Y., Park, I.K.: Controllable facial micro-element synthesis using segmentation maps. In: Proc. IEEE International Conference on Automatic Face and Gesture Recognition (January 2022)
10. Lee, J., Lumentut, J.S., Park, I.K.: Holistic 3D face and head reconstruction with geometric details from a single image. Multimedia Tools and Applications **81**(26), 38217–38233 (November 2022)
11. Loper, M., N. Mahmood, J.R., Pons-Moll, G., , Bla, M.J.: SMPL: A skinned multi-person linear model. ACM Trans. on Graphics **34**(6), 248:1–248:16 (Nov 2015)
12. Pavlakos, G., Choutas, V., Ghorbani, N., Bolkart, T., Osman, A.A.A., Tzionas, D., Black, M.J.: Expressive body capture: 3D hands, face, and body from a single image. In: Proc. IEEE/CVF Conference on Computer Vision and Pattern Recognition(CVPR). p. 10975–10985 (2019)
13. Saito, S., Huang, Z., Natsume, R., Morishima, S., Kanazawa, A., Li, H.: PIFu: Pixel-aligned implicit function for high-resolution clothed human digitization. In: IEEE International Conference on Computer Vision(ICCV). pp. 2304–2314 (2019)
14. Santoso, J., Williem, , Park, I.K.: Holistic 3D body reconstruction from a blurred single image. IEEE Access **10**, 115399–115410 (Nov 2022)
15. Wu, B., Wang, Z., Wang, H.: A GPU-based multilevel additive schwarz preconditioner for cloth and deformable body simulation. ACM Trans. on Graphics **41**(4), 63:1–63:14 (July 2022)
16. Yoon, S., Yun, S., Park, I.K.: Game engine compatible 3D clothes modeling from a single image. In: Proc. International Workshop on Frontiers of Computer Vision (Febuary 2023)