COMP4801 Final Year Project Final Report

fyp23084

Real-Time Virtual Try-On Using Computer Vision and AR

Kong Chun Yung (3035788895)
Supervised by Dr. Wong, Kenneth K.Y.

Abstract

As the e-commerce market keeps growing with fashion as the major revenue contributor, the importance of virtual try-on emerges. Current solutions cannot satisfy customers' needs in terms of efficiency and variety of garment choices. In this project, a mobile solution accepting any image input and allowing users to try any garments on is proposed. The mobile app will transform the image input into a 3D garment model using the xCloth framework, a CNN-based computer vision generative solution. Then, it will transfer the skinning weights from the user-provided alignment information such that it can be animated by the rotation of joints. Finally it uses a pretrained body tracking model to estimate the pose in real-time. Three contributions are made: implementation of automatic pipeline for movable assets, implementation of a real-time body-tracking using remote procedure call and implementation of a highly scalable Android applications. Further work on the fundamental algorithm for each task can be done to improve the performance.

Contents

List of Figures

List of Abbreviations

AR Augmented Reality

CNN Convolution Neural Network

SMPL Skinned Multi-Person Linear

RPC Remote Procedure Call

PCD Point Cloud

PSR Poisson Surface Reconstruction

1 Introduction

1.1 Background

E-commerce has been a substantial part of the market as online payment and logistics advanced. In addition, COVID-19 contributed to a global growth in online shopping and the trend still exists even in the post-pandemic time (ITA, 2021). Among all the online shopping categories, fashion was the top revenue contributor with almost 9 billion US dollars in revenue and shared almost half of the whole e-commerce market in Hong Kong in 2022 (Statisca, n.d.). The market is even predicted to have a two-fold growth in 2027 (Statisca, n.d.). Trying clothes on is an indispensable part of fashion to check whether the selected clothes are suitable for oneself, but it becomes challenging when the clothes are not physically accessible for online shopping. Therefore, virtual try-on providing a fitting experience without needing to put the real clothes is of great help.

Many products such as photo editing software and AI image generators can achieve virtual try-on. Yet, they share a common problem of being static. That is, one has to retake and reprocess a new photo to view a different posture or make any changes. There are also products like AvatorCloud that create an avatar of the user and allow the user to change the avatar's clothing (NeXR Technologies, n.d.). However, the avatar may not be realistic enough to fully demonstrate the clothing and the postures may also be limited to the predefined templates. Some brands like Farfetch have also published real-time virtual try-on features using augmented reality (AR) on Snapchat (Farfetch, n.d.), but the choices are limited to their own products or predefined dressing. One may need multiple applications or services for different brands and categories. Thus, this project proposes a mobile app solution to solve the above problems.

1.2 Objectives

The objective of this project is to create an Android mobile app that allows users to try any clothes on in real time regardless of the brands and types. The workflow of the app is described below.

1.2.1. The user can upload images of any clothing to the app and align the the images to a predefined 3D human model to provide information about the corresponding body parts and size of the clothes. The user can adjust the pose of the 3D human model so that it match the shape of the clothes provided.

1.2.2. The app will transform the 2D images of the clothes into a 3D garment model using CNN. The user can adjust the model parameters to obtain different results. Once the user confirms to use which reconstructed 3D garment model, the corresponding model will be rigged such that it is animatable by the movement of the user.

1.2.3. The app will track the user's motion and transform the 3D clothes model in realtime to handle different postures.

Thus, the main features of the app to be implemented are:

1. **3D garment reconstruction** (Step 1.2.1, 1.2.2)
2. **3D garment overlay** (Step 1.2.3)
3. **Human motion tracking** (Step 1.2.4)

A server is also implemented for handling heavy computation tasks in the above features. The results will be sent back to the mobile app for further local computation.

1.3 Outline

This project makes 3 major contributions. First, it implemented a full pipeline for animatable 3D garment model reconstruction from a monocular RGB image. Such pipeline can be used on different applications other than the virtual try-on proposed in the project such as 3D animations or game asset creations. Second, it introduced a framework for real-time body tracking on mobile device via server-client structure and Remote Procedure Call (RPC). This eases the lack of computation resources of a mobile device and allows advanced algorithms to be used. Third, it implemented a scalable real-time virtual try-on application using MVVM pattern and allowing replacement of each module on the server.

This report proceeds as follows: Section 2 discusses the existing methods to achieve the main objectives and compares them. Section 3 describes the implementation detail of the selected methods and models, as well as the system architecture. Section 4 presents the experiment results and final product. Section 5 discusses future plans and potential improvements for the current methods. Section 6 summarizes this report and states the conclusion.

2 Related Works

This section discusses the possibility of using existing works including image-based virtual try-on (Section 2.1), 3D garment reconstruction networks (Section 2.2), 3D garment fitting algorithm (Section 2.3) and 3D human reconstruction and tracking (Section 2.4) to achieve the objective.

2.1 Image-Based Virtual Try-on

There are multiple models working on 2D image-to-image generation such as Flow-Style-VTON and GarmentGAN (He et al., 2022; Raffiee and Sollami, 2020). Although the results are more realistic and detailed than the 3D approach, there are concerns about (1) expensive computation cost for per-frame generation, (2) size misalignment, and (3) truncation of the input image to fit the shape of the original clothes. In Figure 1 below, the lower part of the target clothes is cut off to fit the shape of the original input. Since this project aims at real-time fitting and it is also one of the major features to keep the original clothes size and shape as much as possible, the 3D model-based approach is more suitable.

Figure 1: Example results from Flow-Style-VTON (He et al., 2022). The left is the input, the middle is the target clothes and the right is the result.

2.2 3D Garment Reconstruction Networks

There are 2 major machine-learning approaches to reconstruct 3D garments from a single image: (1) mapping the image onto a predefined template and (2) generating new models without templates. The template-based approach such as JFNet relies on the segmentation of garment parts and the deformation of templates (Xu et al., 2019). Although it generates a controllable result and is more convenient to define movable parts, it is not practical to exhaust all possible combinations of garment parts and their corresponding 3D templates; The template-free approach such as xCloth and AnchorUDF can reconstruct any garments in their original shape (Srivastava et al., 2022; Zhao et al., 2021). However, the result is greatly dependent on the input pose and it may not recognize the separable parts of the clothes. See Figure 2, for an input image with the sleeves overlapping with the body, the resulting 3D model had the two sleeves connected to the body part. Ideally, the sleeves should be separated from the main body. In this project, the template-free xCloth framework will be used for extracting 3D garments because of its capability of extracting garments in random shapes with texture.

Figure 2: Example results from AnchorUDF (Zhao et al., 2021). The output sleeves are merged into the main body.

2.3 3D Garment Fitting Algorithms

There are two common approaches for overlaying garments: (1) physics simulation and (2) per-frame synthesis. Using physics simulation requires movable 3D garments and collision boxes. It can show the dynamics of cloth in continuous motion. However, it can become computationally expensive to simulate physics in real time on a mobile phone for a detail-rigged model. There are also concerns about model glitching; Using per-frame fitting can omit the rigging part and directly deform the 3D garment to fit the human pose. AgentDress has shown the possibility of real-time fitting using CPU only without GPU (Wu et al., 2021). TailorNet is another per-frame synthesis approach using neural network (Patel et al., 2020). However, these algorithms require predefined 3D templates of garments such that the clothes can be represented in parameters or functions. Since the choice for 3D reconstruction is a template-free approach, these algorithms are not suitable for this project. This project applies skinning weights transfer to generate a movable 3D garment mesh and deforms the mesh directly according to the joints rotation without collision boxes so that it can fit arbitrarily shaped 3D garment mesh to the user's body without too much overhead for computing physics.

2.4 3D Human Reconstruction and Tracking

Real-time body tracking can be achieved in 2 ways, (1) 3D landmarks estimation and (2) full 3D human reconstruction. For 3D landmarks estimation, it estimate the world coordinates for specific entities which are commonly human joints. BlazePose can achieve local real-time inferences on mobile devices (Bazarevsky et al., 2020). However, to deform the garment mesh based on the pose, rotation information is required and it is difficult to deduce from just 3D world coordinates. There is also an requirement that the shape and skeleton (or joints) of the 3D human model must align with the landmarks. It could lead to extra overhead in blending shape or mapping joints. For 3D human reconsturction, a common format to represent 3D humans is the SMPL model which is a parametric human model format that contains skin surfaces and a skeleton (Loper et al., 2015). Its pose and shape can be altered conveniently by just modifying some parameters. It will also be used in this project to represent human captures in AR space. Many models such as ProHMR and PyMAF-X generate accurate SMPL estimation from a single image (Pavlakos et al., 2021; Zhang et al., 2021). However, these models cannot achieve real-time performance which is against the objective of real-time human body tracking. ROMP is a monocular SMPL estimation neural network that is possible for real-time inferences (Sun et al., 2021). Although the computation resources of a mobile device is inadequate to run ROMP locally, these can be eased by introducing a server for inferences and using video streaming to input images. Thus, this project uses ROMP as the motion tracking algorithm.

3 Methodology

The model built for 3D garment reconstruction will be trained on an existing dataset (Section 3.1). Off-the-shelf methods will be used to implement 3D garment overlay (Section 3.2) and human motion tracking (Section 3.3). High-performance frameworks will be adapted to the system (Section 3.4).

3.1 3D Garment Reconstruction

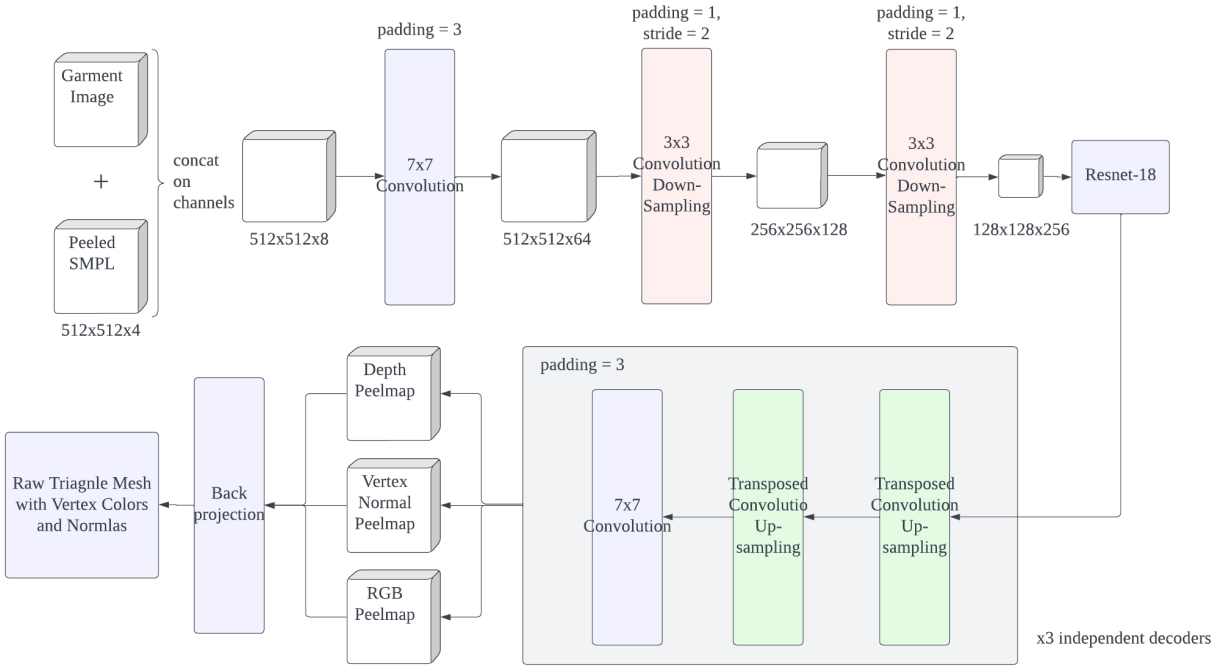


Figure 3: The CNN model structure adapting xCloth framework (adapted from Srivastava et al., 2022).

A model using the xCloth framework was created for the reconstruction framework. The model structure can be viewed in Figure 3. It accepts a garment image with background being black and a prior SMPL depth estimation in peeled human representation (peelmap) as the input. The peeled human representation is a multilayer bitmap where the i-th layer is an image of i-th intersection of rays projected from the front camera to the garment and human mesh for $i \in \{0, 1, 2, 3\}$ (Srivastava et al., 2022). The model consists of 1 shared encoder and 3 independent decoders predicting the depth, color and vertex normal information. The input is encoded using a convolution layer followed by 2 downsampling (convolution) layers and 18 resnet blocks. Then, The encoded features are decoded using 2 upsampling (transposed convolution) layers and a final convolution layer. Sigmoid activation is used for the final convolution for the depth encoder while the vertex normal and rgb decoders use *tanh* activation. Between each convolution, the ReLU activation is used.

The output consists of 3 four-layered peelmap which stores the predicted depth, vertex normal and RGB per pixel. The 0-th layer of the RGB peelmaps is the input image. Each pixel represents a vertex and a point cloud can be generated by projecting the pixels in depth peelmap back to the world coordinates. The back-projected vertices are filtered using a threshold distance away from a low resolution mesh reconstructed using Poisson Surface Reconstruction (PSR). The faces are reconstructed by linking the 8-neighbours of each pixel. The unfilled gaps and holes will be filled by sampling vertices and faces around the gaps from the PSR mesh.

The model was trained on the Deep Fashion3D V2 dataset (Heming et al., 2020). The dataset contains over 1000 3D meshes with texture and 3D meshes of upper clothes, pants and dresses (Heming et al., 2020). Since there is no image in the dataset, pictures of the front view of the 3D meshes will be rendered in 512 × 512 PNG image to form {2D image, 3D meshes} pairs for training. The pairs are further preprocessed into peelmaps to match the input and output format. The objective function is the sum of L1 loss of the depth and RGB peelmaps, and L2 loss of the vertex normal peelmap, i.e.

$$L = \sum_{i=0}^{3} \left\| \hat{P}^i_{depth} - P^i_{depth} \right\| + \sum_{i=0}^{3} \left\| \hat{P}^i_{RGB} - P^i_{RGB} \right\| + \sum_{i=1}^{3} \left\| \hat{P}^i_{normal} - P^i_{normal} \right\|^2$$

where $\hat{P}^i$ represent the i-th layer of the predicted peelmaps and P^i represent the i-th layer of the ground truth peelmaps (Srivastava et al., 2022). This is obtained by removing the segmentation part from the original equation and setting all weights to 1.

Model Input: A 2D garment image with black background and a SMPL human aligned to the image.
Model Output: Triangle mesh containing vertices, faces, vertex colors and vertex normal vectors.

3.2 3D Garment Overlay

The garment mesh is directly rendered on the screen and moves along with the pose of the user. To add dynamics in the garment mesh, the skinning weights of the SMPL human are transferred to the reconstructed garments using the alignment and pose information from Section 3.1. The skinning weights represent the ratio of movements of the vertices to each joint. It is used in linear blend skinning which computes the position of a vertex given joint movements. Using the algorithm by Abdrashitov et al. (2023), every vertices of the reconstructed garment mesh are separated in 2 subsets, S_{match} and $S_{nomatch}$. A vertex is in S_{match} if its closest point on the SMPL human is within 0.05× distance of the diagonal bounding box and 35° from the face normal, otherwise, it is in $S_{nomatch}$.

For all vertices in S_{match}, their weights are copies of weights of the closest point on the SMPL human computed using barycentric coordinates (Abdrashitov et al., 2023). For all vertices in $S_{nomatch}$, the weights are estimated via the following optimization equation:

$$\arg\min_{\mathbf{W}} \quad trace(\mathbf{W}^T(-\mathbf{L} + \mathbf{L}\mathbf{M}^{-1}\mathbf{L})\mathbf{W})$$

where $\mathbf{W} \in \mathbb{R}^{|V| \times m}$ for $|V|$ is the number of vertices, m is the number of bones, $\mathbf{L}$ is the contangent Laplacian matrix and $\mathbf{M}$ is the diagonal lumped mass matrix (Abdrashitov et al., 2023). This algorithm allows a smoother deformation for loose clothing.

The weight assignments and mesh registration are delegated to Blender so that it can be exported in GLB, a common transmission format for 3D scenes. For local rendering on the mobile device, Google Filament is used since it is a lightweight 3D engine with native support for Android device.

Step Input: Reconstructed static garment mesh and aligned SMPL human from Section3.1
Step Output: Animatable garment mesh based on joints of the SMPL human in GLB format

3.3 Human Motion Tracking

The pretrained ROMP is used for real-time motion tracking by estimating SMPL poses. HRNet-32 is used as the backbone model and achieved lower error (Sun et al., 2021). The model is hosted on the server due to the lack of computation power on the mobile device. Real-time mobile video captures are

streamed to the server. To reduce the transmission delay yet preserve the accuracy, the resolution of the video stream is set to 720×1280. It estimates the user's shape and poses online and sends the results back to the mobile. The results are in minimum form which is a float array of size 75 containing the root translation in xyz axis, and the axis-angle of the 24 joints. The pose are used for directly deforming the garment mesh by setting the transformation matrix generated from the axis-angles and translation of each joints.

Step Input: Video stream of 720×1280 resolution
Step Output: Float array of global translation, global rotation and joint rotations

3.4 System Architecture

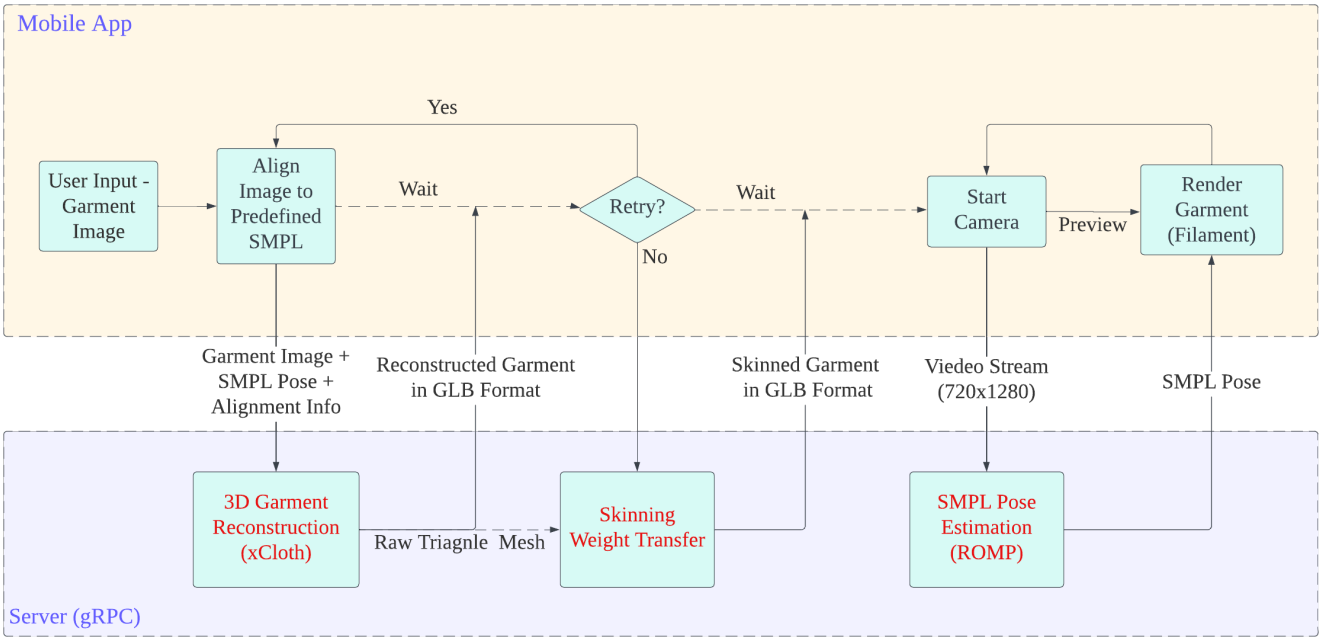


Figure 4: Detailed flowchart of the whole system

A gRPC server was implemented in Python for providing services of the 3 modules, xCloth, skinning weights transfer and ROMP. gRPC framework has high performance for handling large data flow and speeds up the data transfer by using a unified format (gRPC, n.d.). This can be used for efficient 3D model transmission. The detailed workflow can be viewed in Figure 4, the user starts by uploading the image and aligning it to the predefined SMPL human. Then the mobile app waits for the reconstruction to complete and send a signal to the server to run skinning weights transfer if the user accepts the output. The mobile app waits again and start the camera upon receiving the skinned garment model. Now the ROMP model on the server starts receiving inputs and the mobile streams the camera capture to it to estimate the user's pose. This step repeats until the user quit the session.

The Android mobile app is implemented in MVVM pattern using dependency injection to achieve high scalability. It also provides high abstraction such that the modules on the server can be replaced by other algorithms.

4 Results

This section presents and discusses the result of applying proposed frameworks and algorithm in Section 3. The performances of the xCloth, skinning weights transfer and the entire system are listed in Section 4.1, Section 4.2 and Section 4.3 respectively,.

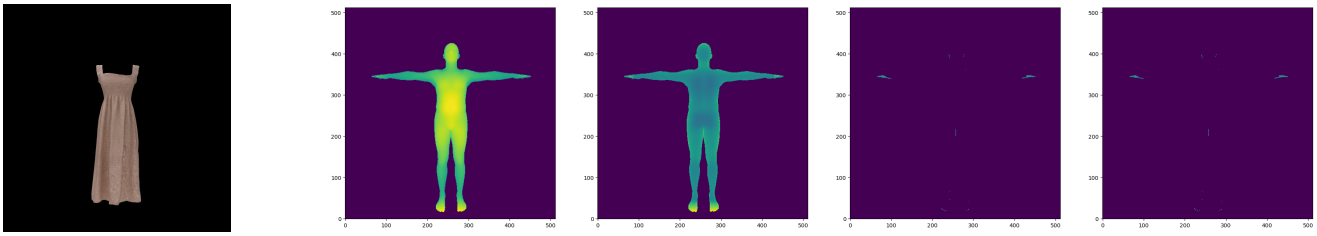
4.1 xCloth Performance

4.1.1 Settings

The xCloth was trained using 20 epochs with exponentially decaying learning rate starting from 0.05 using Adam optimizer. It took 4 hours on a Nvidia RTX3090 graphic card on CS GPU Farm. The dataset was split into train and test dataset with around 1000 and 200 data respectively, splitting by the garment id.

4.1.2 Reconstruction Results

Figure 5 shows an example of the input. It contains the RGB front view of a dress and the SMPL human depth peelmaps.

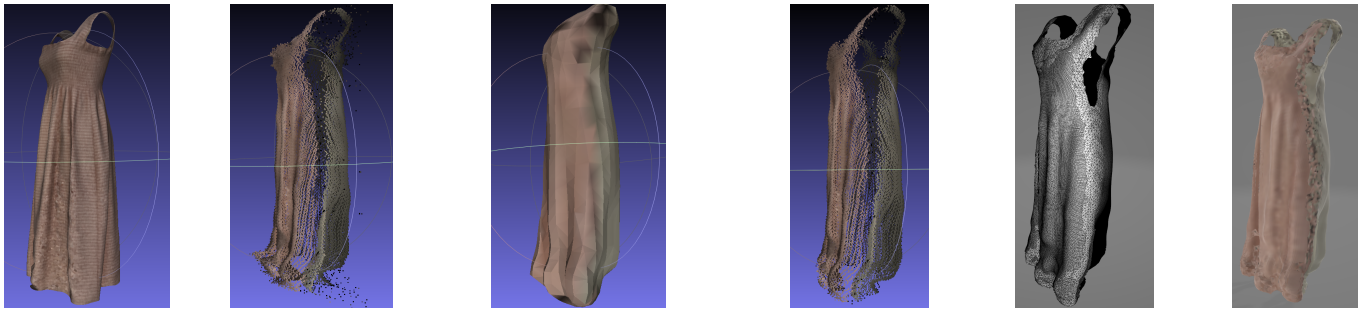


(a) RGB garment image input

(b) SMPL depth prior peelmaps

Figure 5: Input for the garment reconstruction network

Figure 6 shows the corresponding reconstructed garment mesh. The back-projected peelmaps formed a Point Cloud (PCD) with a gap on the side in-between each layer. This is because the z-axis (depth) was almost orthogonal to the face normal. Also, PSR cannot restore the original mesh and still leave some gaps unfilled (See Figure 6e). Because of the lack of indication about expected openings and unexpected gaps, it is difficulty to fill larger holes while preserving the original openings. Although it performed rather poorly on the side, the overall shape was able to reflect shape the ground truth after smoothing and it is acceptable for the scope of this project.



(a) Ground truth (b) Raw PCD (c) Low resolution PSR (d) Filtered PCD (e) Mesh (f) Textured

Figure 6: Output of the garment reconstruction network

The predict texture on the back also deviated from the ground truth by a large extent (See Figure 6). The idea of using the RGB peelmaps was to attempt to capture the similarity between the color of

the front and the back. However, it failed to capture such relationship and generated a monochromatic image. In the original paper, texture extension was used to estimate the texture on the back Srivastava et al., 2022. However, it requires an UV-atlas which cannot be computed in the reconstructed results due to high discontinuity of vertices and overlapping of faces. It is more preferred to introduce extra inputs about different views of the garment so that the reconstructed model can capture the most accurate textures.

There is also a problem about distinguishing whether the neckline of the clothes is at the front or at the back (See Figure 7). It is difficult to address since there is no information about its position. One straight forward solution is to remove the region that belongs to the back of the clothes. This makes little impact to the whole virtual try-on experience since the region removed is negligible comparing to the whole garment.

(a) Input clothes with the neckline shaded

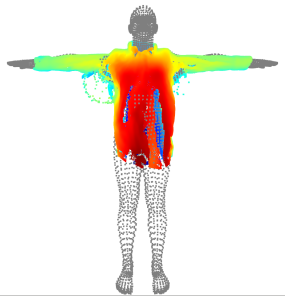
(b) Reconstructed garment with no opening around the neckline

Figure 7: Reconstructed mesh with problem around the neckline

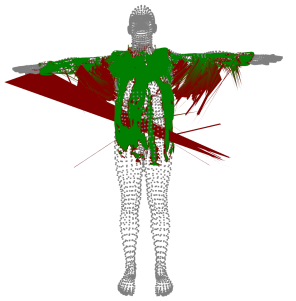
4.2 Skinning Weight Transfer Results

(a) Original mesh

(b) Segregated vertices



(c) Deformed PCD



(d) Deformed mesh

Figure 8: Skinning weights transfer results

Figure 8 showed the results applying the robust skinning weights transfer algorithm. This algorithm aims at adding looseness to the mesh such as the armpit position in Figure 8c. However, the offsets was too large and looked unnatural. Furthermore, in Figure 8d, the faces were ill-deformed. One reason could be that the weight matrix $\mathbf{W}$ is sparse and the target to optimize $\mathbf{W}^T(-\mathbf{L} + \mathbf{L}\mathbf{M}^{-1}\mathbf{L})\mathbf{W}$ could result in a singular matrix. Using the Cholesky decomposition as proposed by Abdrashitov et al. (2023) to estimate the least square solution often result in zero and ,therefore, the veritces collapse to the center origin (0, 0, 0). Another factor is the quality of the reconstructed garment mesh. If some parts of the mesh are merged but the ground truth are separable, they are considered a whole rigid body and produces stretched faces.

The major reason is deduced to be incorrect matching of closest points. The reconstructed mesh often penetrates the SMPL human (See Figure 9). The vertex might be matched to a point that is far away from the actual point, as demonstrated in Figure 10. The error produced could be very large and leads

to incorrect skinning weights copied or interpolated. The current approach lacks shape blending for the reconstructed clothes. Using the native approach by find closest points on the SMPL human and move the vertices along the normal direction towards the closest points could result in the same problem. This problem is left unsolved and could be investigated in the future works.

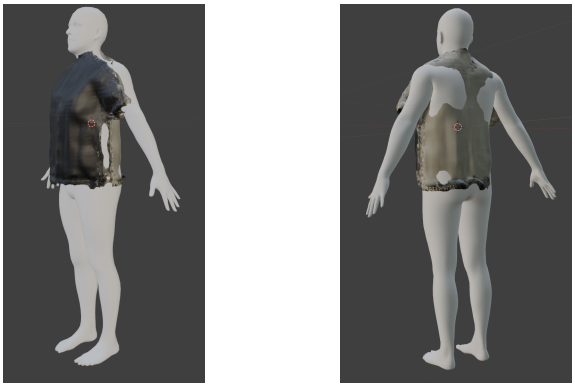


Figure 9: Reconstructed mesh penetrating the SMPL human

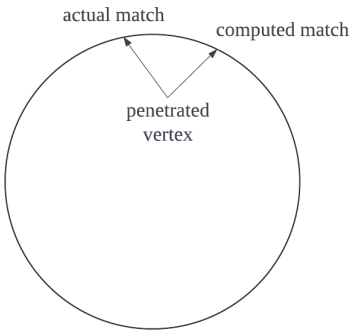


Figure 10: Demonstration of penetration error

4.3 ROMP and System

4.3.1 ROMP Performance

Figure 11 showed the estimated SMPL human and the mobile screen when the garment is rendered. The accuracy depends heavily on the proportion of the body captured. The pose was incorrectly estimated as half-sitting while the user was standing. However, the garment is attached to the upper body, so the result had little error. While the estimated pose may deviate, the user can always adjust the camera angle to obtain a better return. The average inference time is $0.081s$, i.e. around 12 FPS on a PC with Nvidia RTX2080Ti GPU and AMD Ryzen 9 4900HS CPU. This can be considered as semi-real-time, that is, the video is smooth enough for the user to notice only a lag.

4.3.2 Server Delay

The round trip time is measured to be $0.104s$ on the mobile device Samsung Galaxy S21+ using a 5GHz local Wifi network. This greatly affect the FPS of the application, resulting in an average FPS of around 5 to 6 FPS. The user will observe clear lag and delay. The reason is that gRPC framework spends time on serializing and deserializing messages so that the communication is the same across different device. Yet, for the video stream part, the mobile sends raw RGB bytes to the server. This makes the serialization feature unnecessary and an extra overhead. Such overhead could slow down the performance by half as shown in Figure 12 (Costa, 2018).

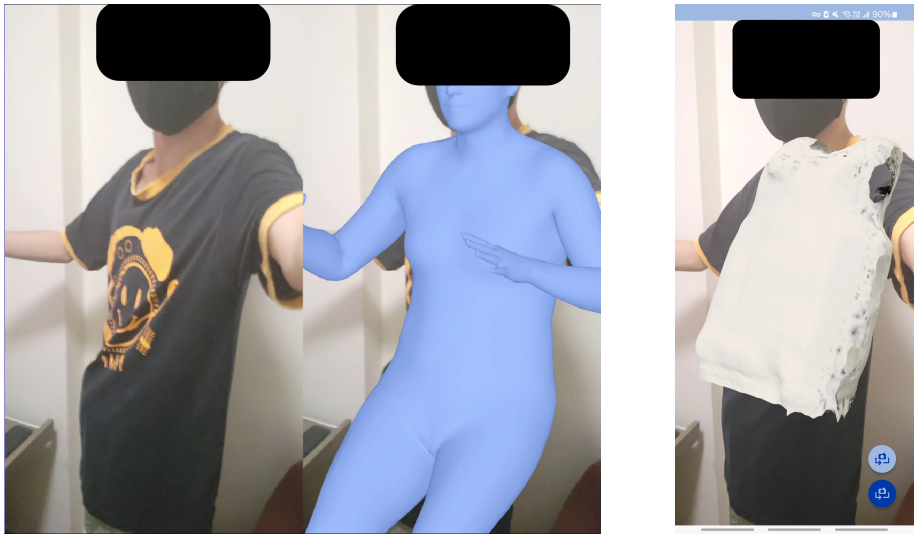


Figure 11: Estimated SMPL pose and rendered garment mesh

On device algorithm such as MLkit's pose detection was attempted, but it provides only 3D coordinates of landmarks and failed to find a mapping from these landmarks to SMPL poses.

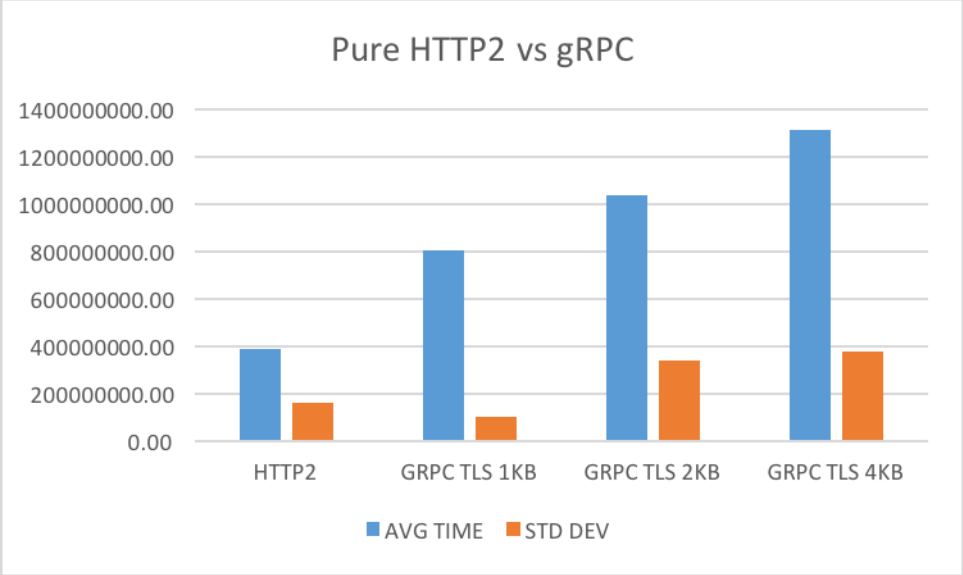


Figure 12: Performance comparison between gRPC and pure HTTP/2 streaming

4.3.3 Android Implementation

Figure 13 shows the UI of the Android application and the feature of each interactive elements are describe in the figures. The application is implemented using Jetpack Compose for the UI and Dagger Hilt for dependency injections.

(a) Home screen (b) Settings screen (c) Alignment screen

(d) Alignment screen (e) Confirmation screen (f) Preview and render

Figure 13: Mobile UI

5 Future Plan

This project is a continuous development. The possible possible improvements are listed as follows:

- Allow multiview input for the garment reconstruction network
- Allow user to manually create/remove vertices or faces from the reconstructed model
- Allow user the specify position of certain region of the clothes
- Add shape blending for the reconstructed garment mesh
- Replace the skinning weight transfer algorithm
- Investigate better models for real-time SMPL estimation

- Replace the server by faster framework such as ZeroMQ or Pure HTTP/2 for efficient raw bytes transfer
- Investigate cross platform solution (all frameworks used are compatible with multiple platform)

6 Conclusion

While an increasing trend of online shopping for fashion is observed, existing tools cannot provide a perfect try-on experience for online clothes. This project proposed a mobile solution using CNN in encoder-decoder sequence, numerical algorithms and motion tracking with AR environment to improve customer experience in online shopping in the fashion category.

The performance for the 3 main features: 3D garment reconstruction, 3D garment overlay and body tracking shows the possibility for real-time mobile virtual try-on application. The xCloth architecture show excellent result in reconstructing 3D garment meshes of highly similar shape to the ground truth. Although the texture generation and manual filtering is required for better result, it is enough for virtual try-on applications. The skinning weights transfer algorithm fails to work as expected due to the lack of shape blending for the garment mesh. This should be investigated in the future work. The biggest difficulties for the body tracking module is the transmission speed, which is slowed down by the serialization feature of the framework chosen, i.e. gRPC. A better framework for efficient transmission of bytes and video stream should be researched. Currently this project provides the minimum working example by connecting all these modules.

7 References

References

Abdrashitov, R., Raichstat, K., Monsen, J., & Hill, D. (2023). Robust skin weights transfer via weight inpainting. *SIGGRAPH Asia 2023 Technical Communications*. https://doi.org/10.1145/3610543.3626180

Bazarevsky, V., Grishchenko, I., Raveendran, K., Zhu, T., Zhang, F., & Grundmann, M. (2020). Blazepose: On-device real-time body pose tracking.

Costa, C. S. (2018, January). Sending files via grpc. https://ops.tips/blog/sending-files-via-grpc/

Farfetch. (n.d.). Off-white ar try-on lens. *Snapchat*. Retrieved September 13, 2023, from https://www.snapchat.com/lens/d4b06ac9f35546d9808fb1ac65b6e93e

gRPC. (n.d.). *A high performance, open source universal rpc framework*. https://grpc.io/

He, S., Song, Y.-Z., & Xiang, T. (2022). Style-based global appearance flow for virtual try-on. *The Institute of Electrical and Electronics Engineers, Inc. (IEEE) Conference Proceedings*.

Heming, Z., Yu, C., Hang, J., Weikai, C., Dong, D., Zhangye, W., Shuguang, C., & Xiaoguang, H. (2020). Deep fashion3d: A dataset and benchmark for 3d garment reconstruction from single images. *Computer Vision – ECCV 2020*, 512–530.

ITA. (2021, October). Impact of covid pandemic on ecommerce. international trade administration. https://www.trade.gov/impact-covid-pandemic-ecommerce

Loper, M., Mahmood, N., Romero, J., Pons-Moll, G., & Black, M. (2015). Smpl: A skinned multi-person linear model. *ACM transactions on graphics*, *34*(6), 1–16.

NeXR Technologies. (n.d.). Avatarcloud. Retrieved September 20, 2023, from https://avatar.cloud/

Patel, C., Liao, Z., & Pons-Moll, G. (2020). Tailornet: Predicting clothing in 3d as a function of human pose, shape and garment style. *arXiv.org*.

Pavlakos, G., Jayaraman, D., & Daniilidis, K. (2021). Probabilistic modeling for human mesh recovery. *The Institute of Electrical and Electronics Engineers, Inc. (IEEE) Conference Proceedings*.

Raffiee, A. H., & Sollami, M. (2020). Garmentgan: Photo-realistic adversarial fashion transfer. *arXiv.org*.

Srivastava, A., Pokhariya, C., Jinka, S. S., & Sharma, A. (2022). Xcloth: Extracting template-free textured 3d clothes from a monocular image. *arXiv.org*.

Statisca. (n.d.). Ecommerce - hong kong —statisca market forecast. Retrieved September 13, 2023, from https://www.statista.com/outlook/dmo/ecommerce/hong-kong

Sun, Y., Bao, Q., Liu, W., Fu, Y., Michael J., B., & Mei, T. (2021). Monocular, One-stage, Regression of Multiple 3D People. *ICCV*.

Wu, N., Chao, Q., Chen, Y., Xu, W., Liu, C., Manocha, D., Sun, W., Han, Y., Yao, X., & Jin, X. (2021). Agentdress: Realtime clothing synthesis for virtual agents using plausible deformations. *IEEE transactions on visualization and computer graphics*, *27*(11), 1–1.

Xu, Y., Yang, S., Sun, W., Tan, L., Li, K., & Zhou, H. (2019). 3d virtual garment modeling from rgb images. *arXiv.org*.

Zhang, H., Tian, Y., Zhou, X., Ouyang, W., & Liu, Y. (2021). Pymaf: 3d human pose and shape regression with pyramidal mesh alignment feedback loop. *The Institute of Electrical and Electronics Engineers, Inc. (IEEE) Conference Proceedings*.

Zhao, F., Wang, W., Liao, S., & Shao, L. (2021). Learning anchored unsigned distance functions with gradient direction alignment for single-view garment reconstruction. *arXiv.org*.