TRIBHUVAN UNIVERSITY

INSTITUTE OF ENGINEERING
IOE CENTRAL CAMPUS, PULCHOWK

MAJOR PROJECT MID-TERM PROGRESS REPORT

The Future of Clothing: A Technological Approach

Submitted By:

Dinesh Devkota (073BCT516)
Maharshi Bhusal (073BCT522)
Rajat Parajuli (073BCT532)
Sushant Thapa (073BCT547)

Submitted To:

Department of Electronics and Computer Engineering

March 5, 2020

Acknowledgement

We would like to express our gratitude to the Department of Electronics and Computer Engineering of the Institute of Engineering, Pulchowk Campus for providing a platform for exchanging knowledge and develop one's personal creativity. All guidance and resources provided by the college has been crucial in our vision that we present today. By assigning a major project as part of the fulfilment of the Bachelors' Degree in Computer Engineering, it has develop technical skills and convey the necessities in handling real life projects in the future.

We are indebted to Dr Jyoti Tandukar for being our supervisor and guiding us towards a feasible project roadmap. His guidance, experiences and expertise have been a boon for our group and crucial in developing our project to the stage it has reached. We would also like to extend our gratitude to the staff of Alternative Technology who have supported us throughout our project timeline.

Abstract

The project “The Future of Clothing: A Technological Approach” is a project that aims to use existing technology to improve the experience of custom designed clothing. Here, a program is written for the design of the clothes and generation of a human body model, both on the basis of user input and preference. The program further wraps the generated clothing on the body model and enables the end-user to view this model in a 3-D environment.

Table of Contents

List of Figures

Introduction

Technology in the sector of image processing has matured enough for developers to create interesting applications. With the increase in computation power, software development tools, developer communities, etc. and decrease in the cost of computer hardware it has become easier for technology creators and consumers to create and consume technology for a better quality of life.

In this project, the attention is focused towards using technology in the field of image processing and applying the said technology towards clothing. The main aim of the project is to create a system whereby a user can virtually wrap a computer generated cloth to a computer generated body of the end user. The body of the subject is based on the description provided by the users themselves. Not only is the clothing wrapped around a user defined body wireframe, but the clothing and the pattern of clothing is also user defined providing a high degree of customization from the perspective of the user.

Patterns and designs used in the clothing can be generated by using a generative adversarial network (GAN) which will synthesize brand new designs and patterns based on the user's preference. Such a GAN can be trained by supplying training data according to the user preference.

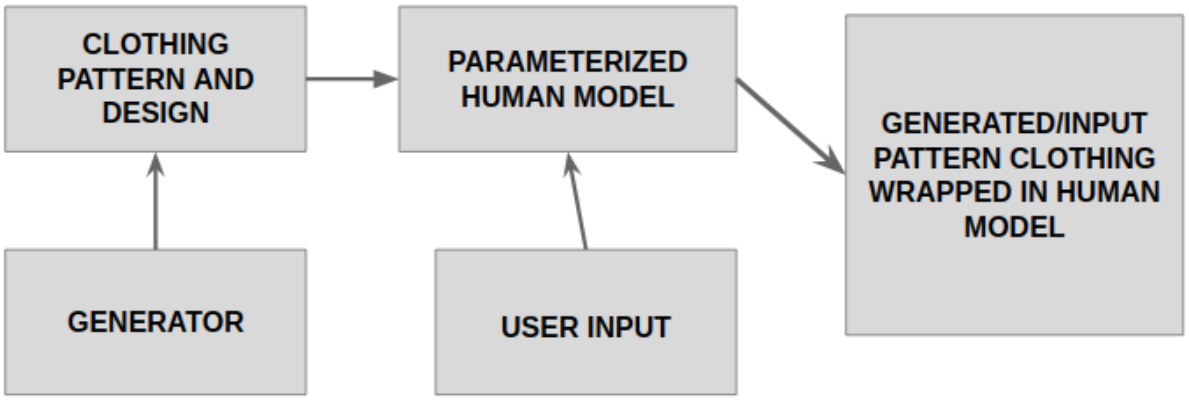

Figure 1- Basic Architecture of Proposed System

Literature Review

Generating Pictures from text description has been done before quite successfully. There have also been a lot of projects in which the image was generated from parameters which can be closely related to image generation from text. Our project revolves around taking input from the user using some text and then using that information to generate images. The project also works to create a design from existing design ideas and gives the user a new unique design instantaneously, provided that the network is trained.

Generative Adversarial Networks (GANs) are a class of the deep neural networks which works by competing two networks namely Generative and Adversarial networks. These networks compete with each other in a race to fool the other network. Because of this, the generative network gets trained to generate an image to fool the adversarial network which is generally a classifier based on Convolutional Deep Neural Network (CDNN). The basic structure of the GAN was taken from [1].

The results of the experiments till now have been satisfactory especially in the generation of natural images. However, few problems have been noticed in the generation of high quality images and geometric images which includes shapes with precise lines and angles.

We have looked into several variants of GANs. As given in [2], Stack GANs can be considered an acute solution to the upscaling problem. Stack GAN stacks the layers of GANs on top of one another to upscale the image from smaller size to larger size provided that the data in the dataset permits that size.

Furthermore we still have problems regarding the graphics element of the motifs and others. However, in this matter several work has already been done. As per [3] , generation of animation faces, which is similar in structure to motifs and designs are difficult to create using simple GAN architecture including the vanilla GAN architecture. The author also moves on to explain that the creation of the faces is faster and more accurate using a variant of GAN called StyleGAN.

Style GAN is a Style-Based Generator Architecture for Generative Adversarial Networks is an alternative generator architecture for generative adversarial networks, borrowing from style

transfer literature. The new architecture leads to an automatically learned, unsupervised separation of high-level attributes.

Furthermore, we have also looked into the ProGAN as per [4] which discuss a new and faster way to train GANs.

From these research papers, the experiment is underway to test if a better learning method is possible that combines the benefit of Style GAN and ProGAN to provide training results faster and then layer the obtained GAN to create a very high resolution image.

In 3D computer graphics and computer vision, a depth map is an image or image channel that contains information relating to the distance of the surfaces of scene objects from a viewpoint. The term is related to and may be analogous to depth buffer, Z-buffer, Z-buffering and Z-depth. The "Z" in these latter terms relates to a convention that the central axis of view of a camera is in the direction of the camera's Z axis, and not to the absolute Z axis of a scene.

Depth Maps usually follow two conventions. While both of them show the relative distance between objects in an image, one convention chooses that points closer to the focal plane are shaded lighter than those further away. The other convention does the opposite, i.e., points closer to the focal plane are shaded darker than those further away.

Although depth maps have a multitude of use cases, the one that is being explored for this project is that of automatic conversion from 2D to 3D. In computer vision single-view or multi-view images depth maps, or other types of images, are used to model 3D shapes or reconstruct them. Depth maps in general can be generated by 3D scanners or reconstructed from multiple images.

There are existing manual and semi-automatic techniques to make 3D images from 2D still images using photo editing tools like Gimp or Photoshop.

A point cloud, as the name suggests, is a collection of points in three dimensions. The most common implementation of point clouds are usually a series of coordinates, and each element in the list is treated as a point with the embedded position. Point clouds generally are not the end product. Likewise, here we further process the point cloud to generate a mesh of triangles.

We researched various algorithms like marching cube algorithm and Delaunay triangulation that generate meshes from point clouds. However, we did not have to use any of those complicated and

computationally intensive algorithms as we already know the relation between the points as they were simply processed from the image.

A 3D model is comprised of many triangles and sometimes other polygons. The computational power required to render triangles increases linearly with the number of triangles in the system. However, the rate of increase of triangles with the resolution is of a parabolic nature. Using a full HD image yields over 4 million triangles. While this is theoretically feasible, it is not very practical. The framerate drop is very noticeable. Triangle reduction yields yet another problem, the mapping of the textures. Once again, we didn't have this problem because simply lowering the resolution of the depth map gave very good results.

UV mapping is the process of mapping textures from an image into the surfaces of a 3D object. We cannot simply attach a 2D image to a 3D object we have to map every surface to a portion of the image. Since we knew where the triangles were taken from in the image, we knew exactly which portion of the image should be mapped to which surface.

There has been a lot of work done in the construction, use and manipulation of three-dimensional human body models for various purposes. Many researchers have used 3D full body scanners, 3D cameras and even the Xbox Kinect camera to capture and map human models. Similarly, a lot of research has be done to manipulate and generate new human models from the assets available by training regression functions to correlate semantically significant values [5] or by employing the use of principal component analysis on feature curves and segment patches drawn onto the model and thus modified [5]. Our project will employ a mix of principal component analysis and linear regression models to generate parameters on the human body model and allow general modifications.

Among the many datasets available, the MPII Human Shape dataset [6] is a readily available free dataset that was used which was based on the widely used statistical body representation and learned from the CAESAR dataset, the largest commercially available scan database to date. The Polygon File Format (.ply) files obtained from the dataset formed the generic point cloud shape of the human body which was further processed in a commercially available software MeshLab [7, 8] for the calculation of normal and triangulation. Further work could employ Delaunay triangulation and also spherical parameterization of 3D meshes [9] for more optimized mesh construction, manipulation and also for better texture mapping.

Completed Work

Synthesis of Images using GAN:

The first experiment involved generating the image of a dog. Dog images of size 64-by-64 px were generated using the GAN architecture given from the GitHub repositories [10] and [11]. The dataset used was the Stanford standard dog dataset which consisted of more than 25000 dog images and annotations in xml format. The first successful generation of the dog image was given from the kernel from [12].

On the next step of our experiment we tried to generate a dandelion image from the standard flower dataset which consisted of about 1000 images of dandelion. The kernel from the dog generator was modified and used to fit our new dataset. The new generated image was also 64-by-64 px in size.

On our next experiment we tried to make a classed generator which takes input a number and outputs the image of different classes on different numbers inputted. The kernel used was a modified version of the TensorFlow tutorial kernel which was taken from the GitHub repository [13]. However, we did not achieve much success with this kernel and thus we used a different architecture which was made by modifying the dog generator kernel.

The latest experiment involves generation of monochrome motif designs. The motifs are converted to monochrome and then fed to the TensorFlow kernel which did not give us the expected result.

Figure 2- Program Generated Output for Dogs

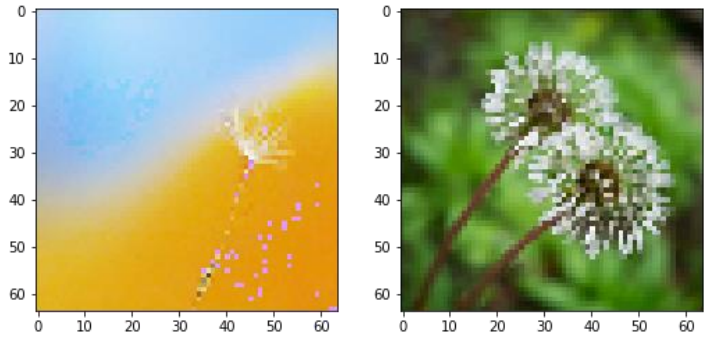

Figure 3- Program Generated Output for Dandelion Flower

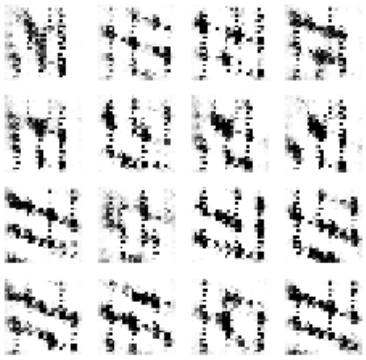

Figure 4- Unsuccessful Shape Generation using DCGAN

Figure 5- Ground Truth for Shape Generation

Generation of Human Body:

From the MPII Human Shape dataset, the collection of 4308 models were converted from a .mat format (binary data containers that are used to include variables, functions, arrays, and other information) which contained a 6449x3 matrix of vertices in three-dimensional Cartesian coordinates into a .ply standard polygonal format for the representation of point clouds. These point clouds contained only the basic positional information of the vertices that defined the human body shape. The generated .ply files were imported into a commercially available tool called MeshLab where the point normals were generated. Then the surface was constructed using Screened Poisson Surface Construction method, both of which are available tools in the program. The model was then exported as a Wavefront object (.obj) and ready for further editing.

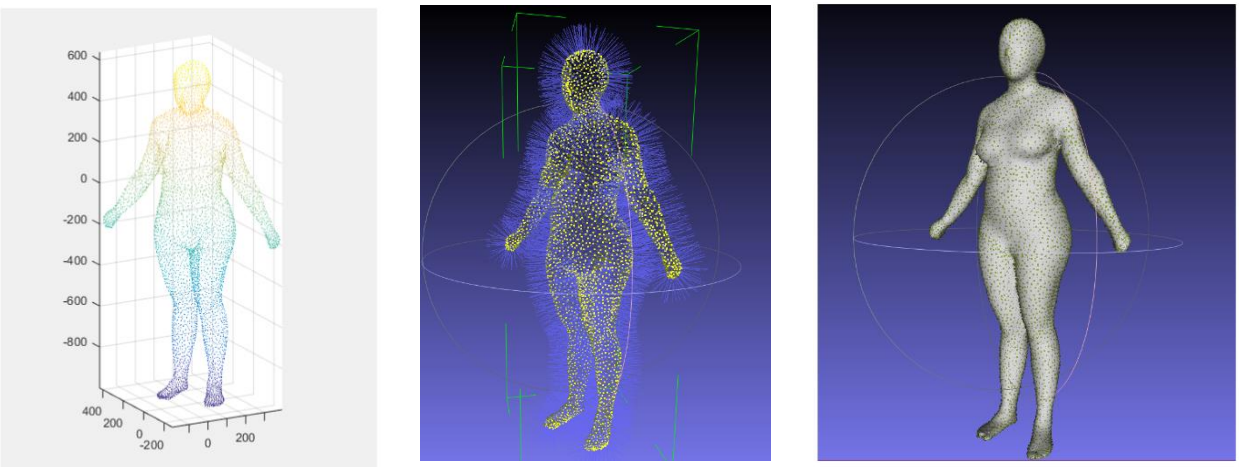

Figure 6- (Left) the extracted point cloud. (Middle) The calculated point normal in MeshLab. (Right) The mesh resulting from the screened Poisson surface reconstruction

Depth Map Extraction:

A small amount of time was dedicated to studying the ways to manually create segmentation and depth maps. Image editing tools like GIMP and Photoshop offered a way to create depth maps by manually segmenting the image into various segments and applying a gray scale according to the proximity of the image perceived by the user.

There were also algorithms for semi-automatic segmentation and depth-map creation with the concept of random walker. These methods were semi-automatic at best, so other approaches were considered.

The next phase of the experiment shifted from generating a depth map to extracting a depth map from images in which such depth maps are encoded in the form of metadata. After a brief research, it was seen that on versions of Android OS below Android 10, the information of how the device stored depth related data is not stored in a certain universal standard. Starting from Android 10, data related to depth can be accessed and used as they are stored in a format called Dynamic Depth Format [14]. Since familiarization and expertise on Android development seemed to take time, a new approach was taken to extract the depth image.

A command line tool, called ExifTool was used to extract the depth information. It is a platform-independent Perl library plus a command-line application for reading, writing and editing meta-

information in a wide variety of files. ExifTool supports many different metadata formats including EXIF, GPS, IPTC, XMP, JFIF, GeoTIFF, ICC Profile, Photoshop IRB, FlashPix, AFCP and ID3, as well as the maker notes of many digital cameras by Canon, Casio, DJI, FLIR, Fujifilm, GE, GoPro, HP, JVC/Victor, Kodak, Leaf, Minolta/Konica-Minolta, Motorola, Nikon, Nintendo, Olympus/Epson, Panasonic/Leica, Pentax/Asahi, Phase One, Reconyx, Ricoh, Samsung, Sanyo, Sigma/Foveon and Sony.

ExifTool was successful in extracting the depth map from a JPEG image, which can be seen below.

Figure 7- (Left) the original image with depth information. (Right) Extracted depth map using the ExifTool

Depth Map to Point Cloud

Another experiment was conducted on a depth-full image from which point cloud was obtained. The depth contains information regarding how far the pixel was. The pixels can then be converted into points. It is assumed that the camera is Z units in front of the image. Then we a line is drawn from the camera to every pixel. The pixel contains information regarding how far or near the object is. Then the pixels are pulled or pushed according to the proximity. In the image above, closer the object, the whiter it is. So the lamp, which was the whitest in the depth map, was pulled closer the furthest.

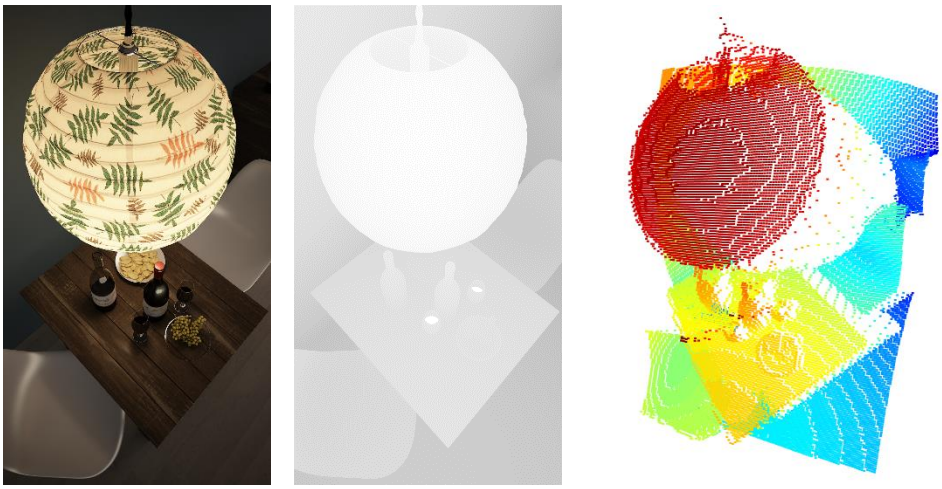

Figure 8- (Left) A normal .jpeg image with depth information. (Middle) The depth map extracted from the image. (Right) The point cloud obtained from the depth map.

Point Cloud to Triangle Mesh

Since the points were generated from the pixels and they correspond to the pixels, points were simply connected according to the pixels which are connected. In our example, the first triangle consists of the points corresponding to the pixels in the black triangle and the second triangle consists of the points corresponding to the pixels in the blue triangle.

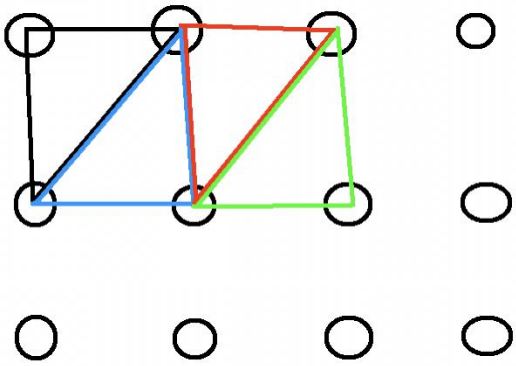

Figure 9- Triangulation of points from their corresponding pixels

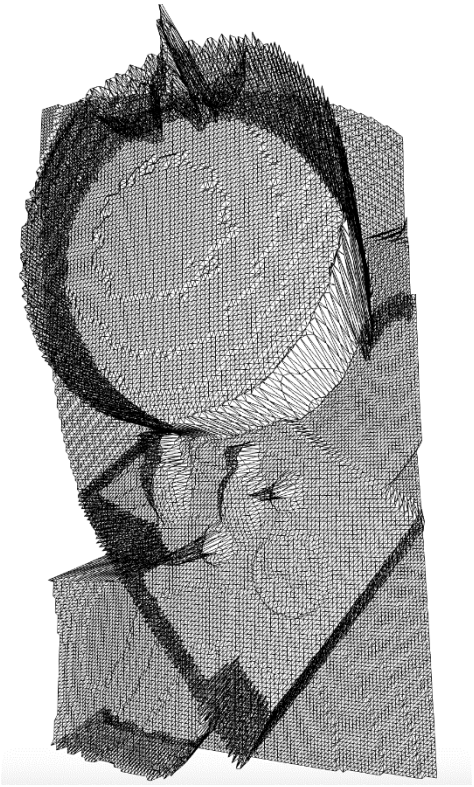

Figure 10- Triangular mesh generated from the point cloud

UV Mapping

The image above contains a lot of triangles. These triangles now need to be filled with segments from an image. Since the points in the triangles directly correspond to the pixels in the image, the corresponding segments of the surfaces were easily extracted.

Future Work

According to our project vision, the project's main aim is to give as much customization as possible to the user. The various components, viz., design pattern synthesis, human modelling, depth map and point cloud extraction, triangulation and texture wrapping are currently being developed, tested and experimented on individually. The complete project will include these components communicating with each other such that the user is presented with a cohesive experience with virtual clothing.

Besides the incorporation of various independent components of the project, some of the components themselves need work before they can match our vision. In particular:

1. The GAN itself needs to be improved so that it can generate pictures of better quality. Experiments need to be done on various different kinds of designs with different characteristics. Although we successfully generated low quality images of dogs and flowers using GAN, ultimately we will have to focus on generation of patterns and designs that are more fitting to clothing. Also currently, the experiments have been focused towards generation of a single design. Work remains to solve problems regarding tiling the said design in a way that is visually and aesthetically pleasing.
2. Work remains to manipulate the human model according to the required physique using parameterisation. User input is paramount to shape the human model according to some parameters so that the user can exercise greater freedom in the customization process. Optionally, since some phones take photos and store the depth map as a metadata, generating the model using that metadata remains to be done.
3. The design generated using GAN still has to be wrapped around the human model realistically so that simulation of clothing can be done realistically. Work remains to be done so as to reduce the amount of errors for a more efficient representation of clothing on a human model.

Bibliography

[1] I. J. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. Courville and Y. Bengio, "Generative Adversarial Networks," 2014.

[2] H. Zhang, T. Xu, H. Li, S. Zhang, X. Wang, X. Huang and D. Metaxas, "StackGAN: Text to Photo-realistic Image Synthesis with Stacked Generative Adversarial Networks," *arXiv e-prints*, p. arXiv: 1612.03242, 2016.

[3] G. Branwen, "Making Anime Faces with StyleGAN," 4 February 2019. [Online]. Available: https://www.gwern.net/Faces.

[4] T. Karras, T. Aila, S. Laine and J. Lehtinen, "Progressive Growing of GANs for Improved Quality, Stability, and Variation," *arXiv e-prints*, p. arXiv: 1710.10196, 2017.

[5] C.-H. Chu, Y.-T. Tsai, C. C. Wang and T.-H. Kwok, "Exemplar-based statistical model for semantic parametric design of human body," *Computers in Industry*, vol. 61, no. 6, pp. 542-549, 2010.

[6] L. Pishchulin, S. Wuhrer, T. Helten, C. Theobalt and B. Schiele, "Building Statistical Shape Spaces for 3D Human Modeling," *arXiv e-prints*, p. arXiv: 1503.05860, 2015.

[7] C. P., C. M., C. M., D. M., G. F. and R. G., "MeshLab: an Open-Source Mesh Processing Tool," in *Sixth Eurographics Italian Chapter Conference*, 2008.

[8] M. Kazhdan and H. Hoppe, "Screened Poisson Surface Reconstruction," *ACM Transaction on Graphics*, vol. 32, p. 29, 2013.

[9] C. Gotsman, X. Gu and A. Sheffer, "Fundamentals of spherical parameterization for 3D meshes," *SIGGRAPH '03: ACM SIGGRAPH 2003 Papers*, pp. 358-363, 2003.

[10] J. Martinez, "GAN Keras Dog Generator," 4 August 2019. [Online]. Available: https://github.com/javiermzll/GAN-Dog-Generator/blob/master/data.py.

[11] A. Gupta, “Kaggle Competition: Generative Dog Images using GANs,” 24 July 2019. [Online]. Available: https://github.com/RoyMachineLearning/Generative-Dog-Images.

[12] C. Deotte, “Dog Memorizer GAN,” 6 July 2019. [Online]. Available: https://www.kaggle.com/cdeotte/dog-memorizer-gan.

[13] B. Lamberta, Y. Katariya, M. Daoust and D. , “Deep Convolutional Generative Adversarial Network,” [Online]. Available: https://github.com/tensorflow/docs/blob/master/site/en/tutorials/generative/dcgan.ipynb.

[14] Google LLC., “Dynamic Depth,” 20 3 2019. [Online]. Available: https://developer.android.com/training/camera2/Dynamic-depth-v1.0.pdf.