Future of Clothing, A technological approach

Dinesh Devkota, Maharshi Bhusal, Rajat Parajuli, Sushant Thapa

May 1, 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 developing 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, the Department has helped us 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

1 Introduction

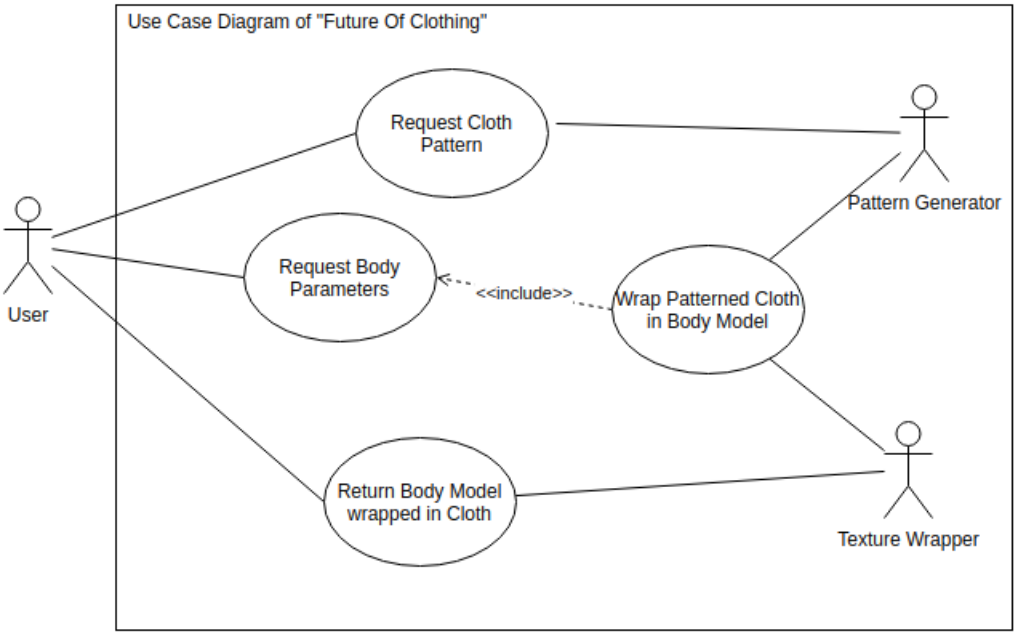

Figure 1: Use Case Diagram of the system

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 new designs and patterns based on the user's preference. Such a GAN can be trained by supplying training data based on existing design.

2 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.

2.1 Generative Adversarial Networks

Generative Adversarial Networks (GANs) are a class of the deep neural networks which works by competing two networks namely Generative and Adversarial networks.This network was defined and introduced to the world by Ian Goodfellow in 2014 in a now legendary paper.[**CITATION NEEDED**]. Generative Adversarial Network consist of two networks. A generative network and an Adversarial network. The generative network is a simple network which takes in gaussian noise as input and then scales and modifies the noise to create a new generated image. This generated image is the result of the GAN we need. New examples from the GANS are generated on the basis of this noise and the network. The adversarial network on the other hand is generally a classifier based on Convolutional networks. The architecture of the generative and adversarial network is not defined anywhere but generally a modified Convolutional network for adversarial and Transpose Convolutional network for generative network. However the original paper introducing GANS [CITATION I.Goodfellow14] was a fully connected network for both generative and adversarial networks.[**CITATION NEEDED**]. The Generative and Adversarial network compete in a game to fool one another, sort of like a race. On each epochs,the generative network or generator generates new images based on the input noise the adversarial network learns from the training data and then uses that information to verify image produced by the generative network. The real image is classified as true for the discriminator while the generated image is classified as false. Now the loss is calculated and is used to update both discriminator and the generator. If all goes well, after a few epochs the generator is able to fool the discriminator with a generated image and we get a very lifelike and real image [**citation needed**].

Our architecture aims to provide a high quality geometric shapes as output. So vanilla GANs with fully connected networks cannot work as the one required as per our project. Thus we have looked into several other variation of GANs which we hope to mix in our architectures.

2.1.1 Deep Convolutional GAN

Deep Convolutinal GAN is a variation of GAN which is based on Deep Convolutional architecture. It uses Deep Convolutinal architecture for the dicriminator as well as Transpose Convolutional to scale the image from the noise in the generator. [1] 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 [**CITATION NEEDED**] , 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 [**CITATION NEEDED**] 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.

2.2 3D and Depth Maps

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.

2.3 UV Mapping

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 [**CITATION**] or by employing the use of principal component analysis on feature curves and segment patches drawn onto the model and thus modified [**CITATION NEEDED**]. 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 **CITATION NEEDED**] 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 **CITATION NEEDED**] for the calculation of normal and triangulation. Further work could employ Delaunay triangulation and also spherical parameterization of 3D meshes []**CITATION NEEDED**] for more optimized mesh construction, manipulation and also for better texture mapping.

3 Methodology

3.1 Basic Structure of System

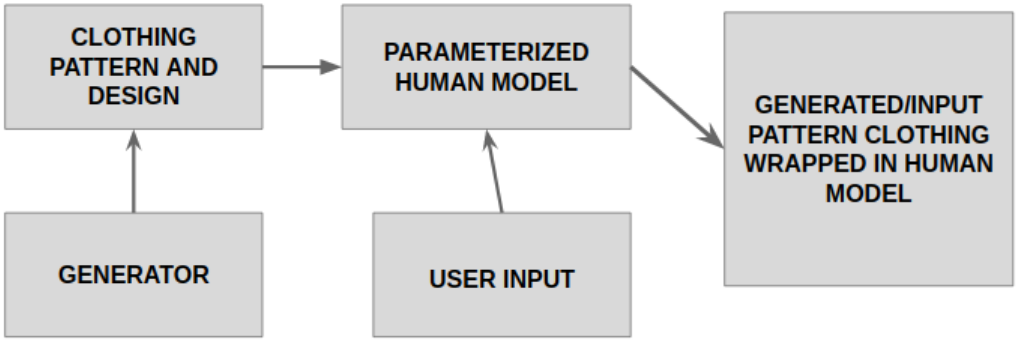

Figure 2: basic architecture of system

3.2 Image Synthesis 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 [**CITATION Mar19 ł1033**] and [**CITATION Gup19 ł1033**]. 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 [**CITATION Deo19 ł1033**].

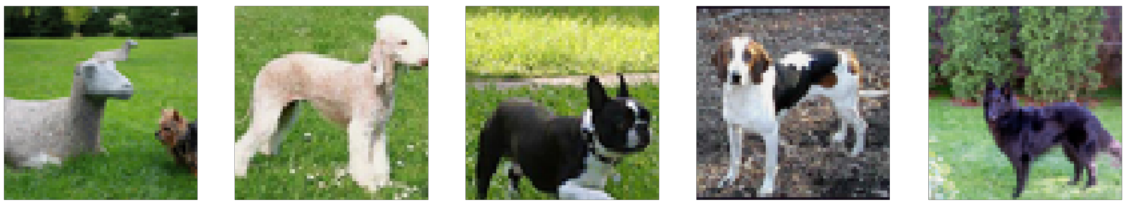

Figure 3: Images generated by GAN

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 [**CITATION Lam ł1033**]. 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.

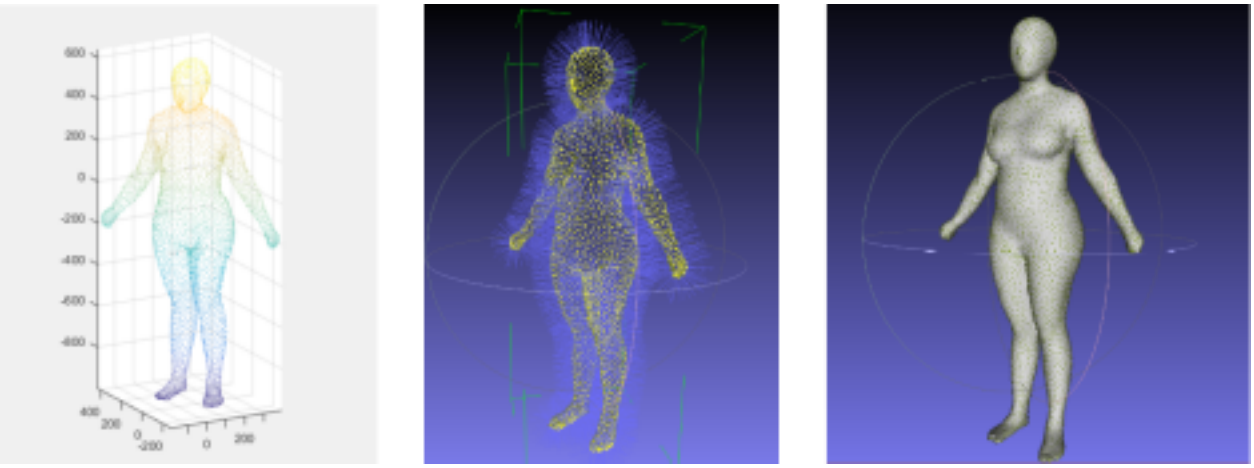

Figure 4: (Left) the extracted point cloud. (Middle) The calculated point normal in MeshLab. (Right) The mesh resulting from the screened Poisson surface reconstruction

3.3 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.

3.4 Depth Maps to 3D

A depth map is to distance, as a thermal image is to temperature. The closer the object, the darker it is, or lighter depending on the exact algorithm used. Just looking at a depth map does not say a lot except how far the objects are. We do not want to just know how far the object is, we want to recreate its 3D object and manipulate it or view it. An important thing to note here is that the depth map only provides us with one side of the 3d object, as evident by the point cloud below.

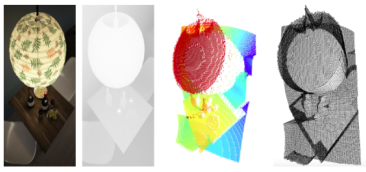

Figure 5: Image with corresponding Depth Map,Point Cloud and Triangle Mesh

3.4.1 Depth Map Extraction

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.

Figure 6: Sample image along with extracted depth map

3.4.2 Depth Map to Point Cloud

4 Output

5 Performance and Analysis

6 Conclusion

References

[1] Alec Radford, Luke Metz, and Soumith Chintala. Unsupervised representation learning with deep convolutional generative adversarial networks, 2015.

7 lines to be added

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.