Deep Learning 3D Scans for Footwear Fit Estimation from a Single Depth Map

by

Nolan Lunscher

A thesis
presented to the University of Waterloo
in fulfillment of the
thesis requirement for the degree of
Master of Applied Science
in
Systems Design Engineering

Waterloo, Ontario, Canada, 2017

Statement of Contributions

Three publications have resulted from the work presented in this thesis.

1. [49]: Nolan Lunscher and John Zelek. Point Cloud Completion of Foot Shape from a Single Depth Map for Fit Matching using Deep Learning View Synthesis. *The IEEE International Conference on Computer Vision (ICCV) Workshop - Computer Vision for Fashion*, 2017.

2. [47]: Nolan Lunscher and John Zelek. Deep Learning Anthropomorphic 3D Point Clouds from a Single Depth Map Camera Viewpoint. *The IEEE International Conference on Computer Vision (ICCV) Workshop - 3D Reconstruction Meets Semantics*, 2017.

3. [48]: Nolan Lunscher and John Zelek. Foot Depth Map Point Cloud Completion using Deep Learning with Residual Blocks. *Journal of Computational Vision and Imaging Systems*, 2017.

Abstract

In clothing and particularly in footwear, the variance in the size and shape of people and of clothing poses a problem of how to match items of clothing to a person. This is specifically important in footwear, as fit is highly dependent on foot shape, which is not fully captured by shoe size. 3D scanning can be used to determine detailed personalized shape information, which can then be used to match against product shape for a more personalized footwear matching experience. In current implementations however, this process is typically expensive and cumbersome. Typical scanning techniques require that a camera capture an object from many views in order to reconstruct shape. This usually requires either many cameras or a moving camera system, both of which being complex engineering tasks to construct.

Ideally, in order to reduce the cost and complexity of scanning systems as much as possible, only a single image from a single camera would be needed. With recent techniques, semantics such as knowing the kind of object in view can be leveraged to determine the full 3D shape given incomplete information. Deep learning methods have been shown to be able to reconstruct 3D shape from limited inputs in highly symmetrical objects such as furniture and vehicles.

We apply a deep learning approach to the domain of foot scanning, and present methods to reconstruct a 3D point cloud from a single input depth map. Anthropomorphic body parts can be challenging due to their irregular shapes, difficulty for parameterizing and limited symmetries. We present two methods leveraging deep learning models to produce complete foot scans from a single input depth map. We utilize 3D data from MPII Human Shape based on the CAESAR database, and train deep neural networks to learn anthropomorphic shape representations. Our first method attempts to complete the point cloud supplied by the input depth map by simply synthesizing the remaining information. We show that this method is capable of synthesizing the remainder of a point cloud with accuracies of 2.92$\pm$0.72 mm, and can be improved to accuracies of 2.55$\pm$0.75 mm when using an updated network architecture. Our second method fully synthesizes a complete point cloud foot scan from multiple virtual view points. We show that this method can produce foot scans with accuracies of 1.55$\pm$0.41 mm from a single input depth map.

We performed additional experiments on real world foot scans captured using Kinect Fusion. We find that despite being trained only on a low resolution representation of foot shape, our models are able to recognize and synthesize reasonable complete point cloud scans. Our results suggest that our methods can be extended to work in the real world, with additional domain specific data.

Acknowledgements

I would like to thank my supervisor, Professor John Zelek, who provided invaluable guidance throughout my masters, and has been a joy to work with. I would also like to thank my thesis readers, Professor David Clausi and Professor Alex Wong, for their time and input. I would also like to thank the Systems Design Department Staff for their help with my admittance, and all their administrative help throughout my degree.

I would like to thank Talometrix, and NSERC for supplying resources and financial support that made this thesis possible.

Dedication

This thesis is dedicated to my parents who were there for me my whole life, and supported me through my degrees, even while I was away from home. I would also like to dedicate this to my friends from JZ's Corner, as well as the VIP lab and surrounding labs that made everyday in the office enjoyable.

Table of Contents

List of Tables

List of Figures

Chapter 1

Introduction

In the footwear market, there are countless brands and models that come in all shapes and sizes. Similarly, individual feet vary widely and pairing a person to the best-suited footwear is not obvious [35]. Finding the optimal footwear can be of importance to consumers as their fit largely determines performance and comfort. Additionally, ill fitting footwear can cause pain and various foot deformities [27]. The American Podiatric Medical Association recommends properly fitted shoes as a preventative measure against foot pain [73].

In footwear, typically the only indicator used to estimate fit is shoe size, which does not fully characterize the profile of a shoe or a foot [33]. Foot morphology for describing foot shape can be complex and include measures for various lengths, widths, girths and angles [31]. This makes it difficult to determine how some footwear will fit without trying them on. This can be especially inconvenient with the rise of online shopping, where items cannot be tried-on before purchase. This process could be improved if the precise 3D shape of a foot could be virtually fitted with the 3D volumetric shape of a shoe cavity.

The first step of this process includes determining the shape of a person's feet beyond what is captured by a simple shoe size measurement. 3D scanning presents a great solution to this problem, as it can provide an accurate and complete model of a person's foot. Such systems have already started to hit the retail space, including the Vorum Yeti [18] and the Volumental scanner [78], however they tend to be expensive or cumbersome to operate.

In developing a cheap and simple 3D scanner, RGBD cameras are compelling as they are affordable with sufficient accuracy and easy to operate [16, 54]. These cameras use an infrared projector and camera pair that can measure the depth of a surface using a structured light or a time of flight system. 3D scanning using RGBD cameras does however present a number of challenges. A RGBD camera can only capture points from a single viewpoint at a time, and obtaining a full 3D scan of an object requires either a moving camera or multiple fixed cameras. When scanning a person, stationary cameras are a better solution, as they are not as mechanically complex but also faster and leave less opportunity for a person to move during scanning. When using multiple RGBD cameras however, interference between their projector patterns prevents them from being able to capture simultaneous images, which can still allow for some movement by the person [46]. Another complication is that in order to form a 3D scan, the data from each view point needs to be registered to produce a properly aligned scan. This process typically works best when there is significant overlap between views, thus requiring many views that are close together. When using fixed cameras, more required views results in more required camera hardware (e.g., 8 cameras are needed to have just one at each 45 degree interval around an object).

Many of the described complications in building a 3D scanner have to do with the need to capture every aspect of an object to reconstruct a model. Humans do not tend to be limited in this same way. We have the ability to form a complete mental model of an object from far less information. This has been shown in experiments that demonstrate our ability to perform "mental rotation" on 3D objects [71]. We seem to be able to leverage prior information and semantics about objects to more efficiently create models and to fill in missing information. Brain inspired neural networks and deep learning approaches have been shown to perform well in a multitude of vision related tasks [44] by leveraging abstract representations to implicitly learn and classify models from large databases. Deep learning methods have been shown to be able to model 3D shape from limited inputs through voxel volume representations [14, 22, 66, 75, 81, 83, 84], primitive shapes [91] and with view synthesis [6, 63, 76, 88, 89] in objects such as furniture and vehicles.

Here we explore how deep learning techniques can be used to address the shortcomings of current scanning systems. We build deep models which learn to produce complete foot scans from partial scans given in the form of a single depth map. These models learn to extract the necessary information, including knowledge about left vs. right foot, and various foot structures and their measures from a partial representation and apply it to produce a full 3D reconstruction of the overall foot. Our initial models are based on previous view synthesis architectures [76], which we subsequently improve upon using components of ResNet [34].

Due to a lack of freely available foot scan data, a depth map dataset was constructed from renderings of body models from MPII Human Shape [64] built from the CAESAR database [67]. The feet of each body model were isolated, and virtually scanned at various azimuth angles, elevation angles, roll angles and at varying radii around the profile of a foot. We virtually scan using conditions that are representative of a real world scanning system, such that systems trained on this data can suggest their capabilities when applied to the real world.

We implement two methods of complete foot scanning from a single depth map. In one technique, we use a deep learning model to learn only to complete the point cloud provided as input. In this method, we keep the process as simple as possible, by requiring only a single forward pass through a deep network to generate all required points, in such a way that they are automatically aligned to the input. Using this method, we are able to achieve point cloud accuracies of up to 2.55±0.75 mm.

In our second technique, we use a deep learning model to learn to fully synthesize the foot point cloud. This method requires multiple forward passes through our network and alignment steps to produce the overall point cloud, but is able to produce very accurate results. Using this method, we are able to achieve complete point cloud accuracies of up to 1.55±0.41 mm.

Our models are trained using MPII Human Shape models, which only gives a rough idea of how the system might work in a real setting. To get a better idea of our methods in the real world, we collect a small set of real scans using Kinect Fusion [58]. We show that our models are able to understand real world data however are not fully generalized to support this domain. We will discuss further work that could be used to adapt our methods for real world use.

1.1 Contributions

Overall, the aim of this thesis is to describe, implement and demonstrate methods of producing a complete point cloud foot scan from only a single depth map input.

1. We propose a method for leveraging a profile depth map of a foot to produce a complete point cloud with direct alignment to the input points (Section 3.2).
2. We propose a method for fully synthesizing a complete anthropomorphic point cloud using view synthesis principles (Section 3.3).
3. We demonstrate the impact and benefits of utilizing residual block network architectures in our method of point cloud completion (Section 3.4/4.4).
4. We discover the limitations of how our deep models generalize from MPII Human Shape data to real world scanning, which primarily revolve around a requirement to match real foot pose and posture to that seen in training (Section 3.5/4.5).

5. We are the first to apply deep learning to implicitly learn and synthesize 3D shape of anthropomorphic body parts.

Three publications have resulted from the work presented in this thesis:

1. [49]: Nolan Lunscher and John Zelek. Point Cloud Completion of Foot Shape from a Single Depth Map for Fit Matching using Deep Learning View Synthesis. *The IEEE International Conference on Computer Vision (ICCV) Workshop - Computer Vision for Fashion*, 2017.
2. [47]: Nolan Lunscher and John Zelek. Deep Learning Anthropomorphic 3D Point Clouds from a Single Depth Map Camera Viewpoint. *The IEEE International Conference on Computer Vision (ICCV) Workshop - 3D Reconstruction Meets Semantics*, 2017.
3. [48]: Nolan Lunscher and John Zelek. Foot Depth Map Point Cloud Completion using Deep Learning with Residual Blocks. *Journal of Computational Vision and Imaging Systems*, 2017.

1.2 Thesis Outline

We will review the relevant literature and technologies in Chapter 2. We will then discuss our various algorithms and methods in Chapter 3. Here we will discuss creating a large foot shape dataset (3.1), and present our point cloud completion (3.2), and point cloud synthesis (3.3) methods. We will also discuss improvements to the network architecture (3.4) and finally the collection of a small set of real world scans (3.5). In Chapter 4 we will discuss our results and compare our methods. Here we will present our metrics (4.1) and discuss the results of each of our methods (4.2, 4.3, 4.4). We will also discuss how our methods performed on real world data, and suggest how to tune our methods such that they can be deployed in the real world (4.5). Finally, Chapter 5 will contain our final discussions and conclusions.

Chapter 2

Background

2.1 Foot Measurements

Foot shape is complex and in order to properly characterize it requires many measures of lengths, widths, girths and angles [31]. Shoe size systems currently in use typically only rely on a single length measurement which does not do enough to characterize shape.

Foot measurements classically are done using caliper rules for heights, widths and other length dimensions, as well as tape measures for various circumferences [31]. The Brannock Device® [15], shown in Figure 2.1, is a commonly used foot measuring instrument, which packages a caliper for foot length and breadth into an easy to use tool. This device is used in many retail stores as a way to quickly determine a customer's shoe size.

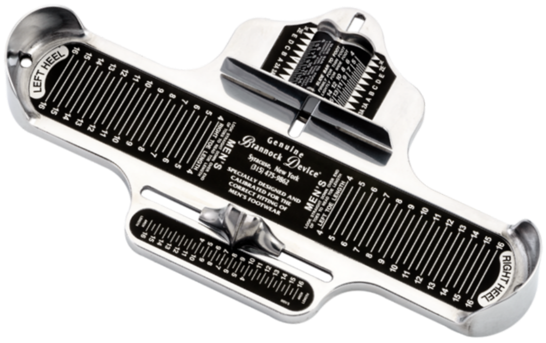


Figure 2.1: The Brannock Device® [15].

Pedographs are another simple way to collect measurements of the foot. This tool measures the shape and pressure distribution under the feet. Originally, this system works by having a person stand on a pad with ink on its reverse side that would be imprinted on a pedograph paper placed underneath [31], although more advanced electronic plantar pressure measurement systems have since been developed. These systems are largely used for orthotic and footwear design, as well as for foot health monitoring [3]. These machines have also been used in retail environments, such as the Dr. Scholls Custom Fit Orthotics Center [37], which recommends orthotic products to customers based on their individual pressure profile. Examples of these systems are shown in Figure 2.2.

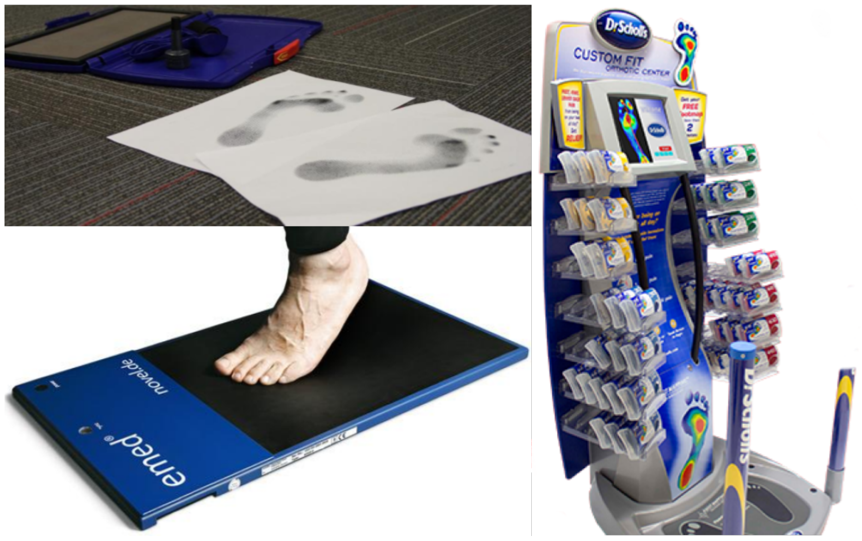


Figure 2.2: Examples of pedograph systems. Top-Left: The ink pedograph [57], Bottom-Left: The emed®electronic pedograph system [60], Right: The Dr. Scholls Custom Fit Orthotics Center [37].

In describing overall foot morphology, a collection of measures are typically used. These measures are based on anatomical structures and use various landmarks on the foot to define the precise areas to measure. The measures used in describing foot morphology are shown in Table 2.1 and includes 19 measures of various lengths, widths, circumferences, heights and angles. Another separate set of measures also exists for describing foot posture and arch structure, which are shown in Table 2.2. This data is valuable in designing footwear, but typically is not used to fit customers with specific shoes that may have been designed for similar morphology. Statistical analysis of foot morphology among the population has shown that there foot proportions significantly vary between feet of the same

Table 2.1: Foot measures describing foot morphology. Reprinted by permission from Macmillan Publishers Ltd: International Journal of Obesity [52], copyright 2008.

Category	Measure
Lengths	Foot length
	Ball-of-foot length
	Outside ball-of-foot length
	Toe length
	Heal to medial/lateral malleolus
Widths	Ball-of-foot width
	Orthogonal ball-of-foot width
	Heel width
	Orthogonal heal width
	Plantar arch width
	Bimalleolar width
Circumferences	Ball girth
	Minimum arch girth
	Heel girth
Heights	Medial/lateral malleolus height
	Dorsal arch height
Angles	Ball angle
	Hallux angle
	Digitus minimus angle

size [42]. For this reason, more detailed measurements are needed for designing and fitting shoes, as using the mean measurement values is not suitable for the majority of people. These measure are unfortunately difficult and time consuming to measure manually, and often require repeated measurements to collect, as individual measures can be inconsistent [33]. On top of the many complications in manual measurements, shape can also change with the load placed on the foot (e.g. standing vs. sitting) [69], adding to the variability.

Table 2.2: Foot measures describing foot posture and arch structure [31].

Category	Measure
Anthropometric Values	Arch height
	Plantar arch width
	Rearfoot angle
	Arch angle
Foot Indices	Arch Index
	Chippaux-Smirak Index
	Staheli Index

2.2 Shoe Shape Design

The shape of a shoe is largely determined by a shoe-last, such as those shown in Figure 2.3, which are used extensively in the manufacturing process of shoe-making [26]. Traditionally, lasts were made of wood and were usually based on a small set of reference measurements. The shoe-last maker would shape and sand the model until the measures were correct, but the specifics of how it was sanded would also impact the fit. This may take several iterations, and the skill of the shoe-last maker largely determines the quality of the shoe-last. With modern technologies, CAD software is used to more efficiently and accurately design shoe-lasts, with CNC machines used to create the physical last.

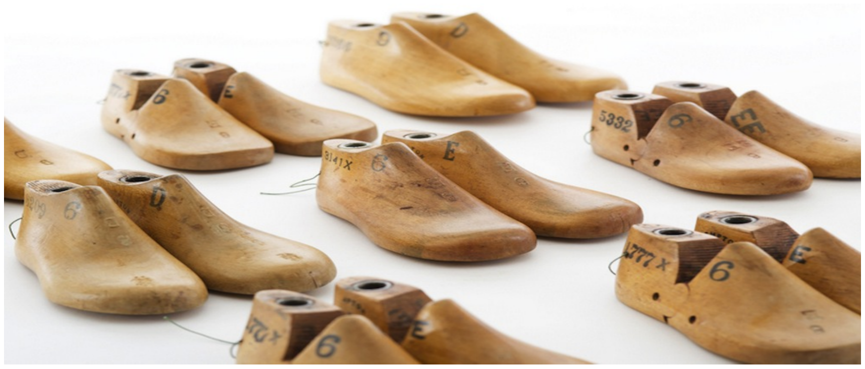


Figure 2.3: Examples of shoe-lasts [72].

Shoe-last shape is not necessarily the same as foot shape. Shoe-lasts are a more regular shape, and are often designed to obtain a specific shoe appearance, as well as to make

Table 2.3: Increments used in various shoe sizing systems [31].

Sizing System	Starting Unit	Length Increment	Girth Increment
English	4 in. = 101.6 mm	1/3 in. = 8.46 mm	1/4 in. = 6.35 mm
American	3 11/12 in. = 99.48 mm	1/3 in. = 8.46 mm	1/4 in. = 6.35 mm
Continental	100 mm	2/3 cm = 6.66 mm	5 mm
Chinese	90 mm	5 mm	3.5 mm
Mondopoint	-	10 mm	5 mm

constructing the shoe around the last easier [31]. Shoe-lasts can have many measurements describing their shape, including those shown in Figure 2.4. The American Footwear Manufacturers Association defines as many as 61 last dimensions [26], however these may differ based on designer or manufacturer.

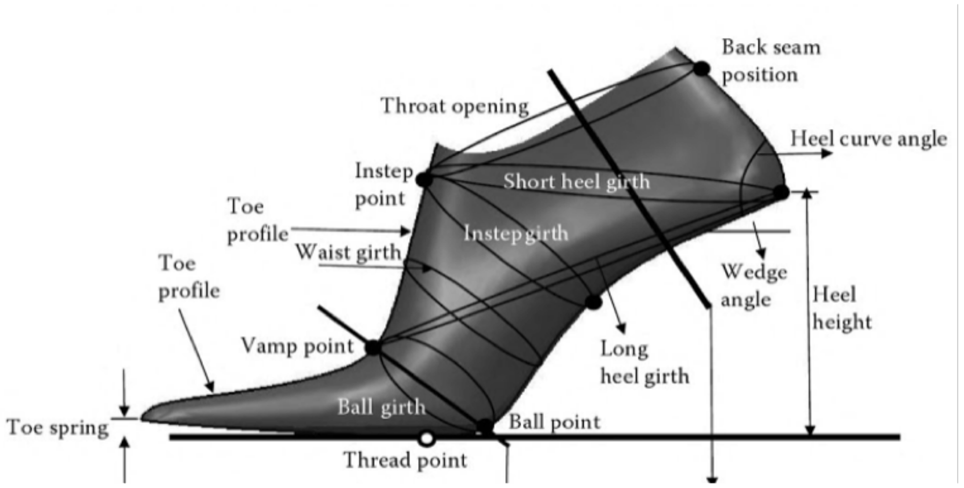


Figure 2.4: Various shoe-last shape measurements [26].

The sizing systems used to describe shoes primarily uses foot length and girth. There are differing systems around the world, with the Continental (common in Western Europe) and American systems being the most widely used [31]. Table 2.3 describes the standard increments used in length and width for each system, however these sometimes differ based on the individual manufacturer. These measures only encompass one length and one girth measurement, and neglects any measure of circumference, height, angle or arch.

2.3 Foot Shape Modeling

Collecting foot morphology measures are used in shoe design, but rarely used in shoe fitting. This is largely due to practical reasons, as capturing the necessary measurements is time consuming and requires some skill.

An alternative approach to taking the full set of morphological measurements is to collect only sufficient information, such that accurate estimates of the overall shape can be made. It has been shown that the parameters of foot shape can be compacted using statistical models while still containing sufficient information to reconstruct the overall shape from as few as 4 input measurements [50]. This was done by characterizing foot shape by 4 measurements at 99 cross sections along the foot as shown in Figure 2.5. Each of the 4 measurements for each cross section are used to deform a standard foot, which can be used to estimate the overall foot shape by predicting the 4 measurements of all cross sections given the measures from only one cross section. A similar outcome can also be achieved by using plantar and profile foot outlines, which contains information about cross section widths and heights [51].

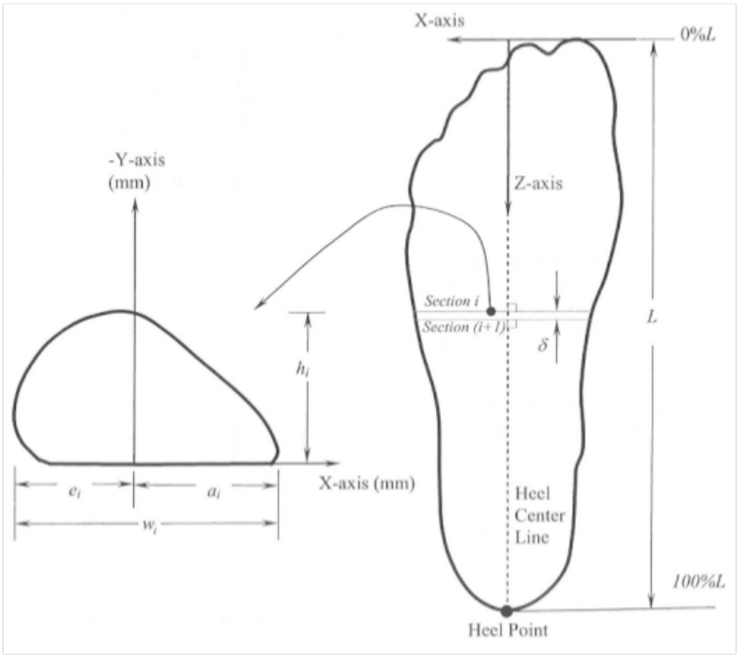


Figure 2.5: Model parameters for a foot used by Luximon et al. [50].

In the application of building body shape models that can be manipulated, such as for creating personal avatar characters, other works have similarly explored creating parameterized models of bodies [9, 64, 92]. These methods work by fitting a reduced set of vertices to a set of complete body scans, and isolate body shape from pose. Using PCA, the parameters characterizing overall body shape can be even further compressed.

2.4 3D Reconstruction and Scanning

To properly characterize the shape of a foot, it is best to capture its entire surface. This surface contains all the measurements previously described, while also capturing aspects not typically measured. This is useful because in feet, with the possible exception of the space between the toes, all surfaces are important in most cases [31]. Various techniques exist for capturing 3D information about an object, many of which have been applied to capturing 3D foot shape.

2.4.1 Passive Triangulation

In passive triangulation techniques, objects or scenes can be reconstructed in 3D by finding correspondence of points across images taken from different poses [56]. Based on where a corresponding point appears in at least two image, as shown in Figure 2.6, its relative 3D position to the cameras can be determined using triangulation. Passive triangulation typically is done either using stereo cameras, or with a moving camera. In the stereo case, two cameras with known relative pose, typically with the same orientation but separated by some baseline distance, are used. With stereo, the 3D position of corresponding points found in images from both cameras can easily be found. A similar process can be done using multiple images taken from a moving camera, and is known as structure-from-motion. In this case, the 3D position of corresponding points found in images across time from different camera poses can be triangulated (assuming a static object or scene). In either case however, they rely on the ability to accurately determine corresponding points across images, which works best on feature rich objects.

Passive triangulation on its own is not popular in the application of foot scanning, as feet tend to be smooth and featureless. In order to mitigate this issue, a simple solution has been to apply a texture pattern to the object being scanned, for examples a foot can wear a feature rich sock. This is the strategy used by the Lightbeam Scanner by Corpus.e, shown in Figure 2.7. This system works with a yellow and green sock, with a single camera and light source mounted on an arm which moves around the foot in a circular path [19].

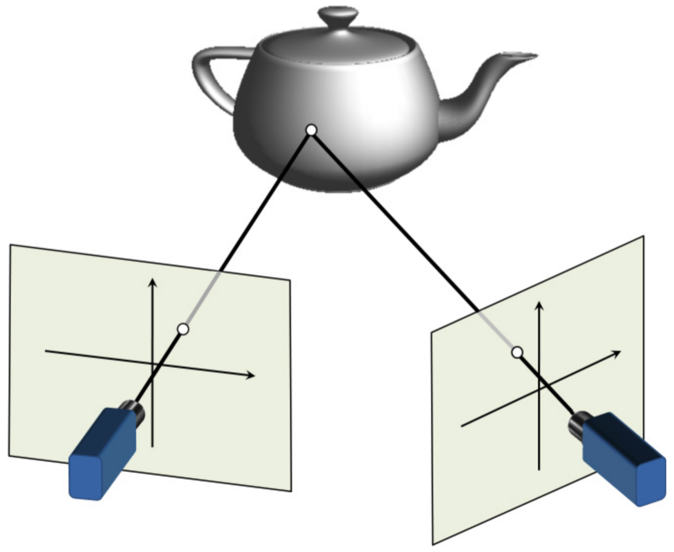


Figure 2.6: Passive triangulation where a point is seen in two separate positions in images taken from different poses. If the relative poses of the images are known, then the relative 3D position of the point can be determined [56].

2.4.2 Active Triangulation

Active triangulation techniques use various methods to simplify the correspondence problem in stereo triangulation. They do this by replacing one of the cameras in the system with a light source that projects a point or pattern from a calibrated position into the environment, where it can be easily found in the image of a paired camera, as shown in Figure 2.8.

Laser Scanning

Laser scanning is a form of active triangulation, where a calibrated laser and sensor pair are used to determine 3D geometry. Typical laser scanning works with a combination of a laser source and camera system. A laser projects a line or set of lines onto an object, where a camera tuned to the specific wavelength is used to measure the deformation of the line [56]. This information is used to turn each line into a 3D curve. The laser is moved along the entire surface of the object, such that the captured 3D curves can form the entire 3D surface [31].

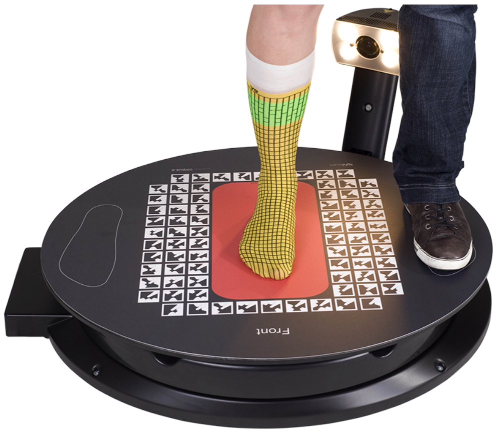


Figure 2.7: The lightbeam 3D scanner [19].

Laser scanners typically come as either stationary scanners or as handheld scanners. A notable example of a stationary scanner is the Yeti 3D Scanner by Vorum Research Corperation [18], shown in Figure 2.9. The Yeti scanner uses four laser sources with eight cameras to capture 3D shape, and has a scanning process that takes around four seconds [31]. In the handheld scanner segment, a notable example is the scanGogh, also by Vorum Research Corperation [17], shown in Figure 2.9. This uses a single laser source and a single camera to capture shape, with a sensor that tracks the position and orientation of the device using magnetic fields.

Structured Light

Active triangulation systems can also further simplify the 3D reconstruction using images. In these systems a projector projects a pattern onto a surface, where a camera at a known relative position can view and measure the distortions in the pattern to determine 3D positions. This technique, often referred to as structured light, simplifies the correspondence problem by projecting spatially or temporally coded patterns [56]. Using a camera, the positions of points in the pattern seen on a surface can be easily looked up to find their position in the reference pattern, and thus be used for triangulation.

A notable example of a scanner using structured light is the FotoScan 3D, by Precision 3D [1]. The scanner uses a set of projector camera pairs placed all the way around a foot, as well as below, for a total of five pairs. The system can capture the complete 3D surface

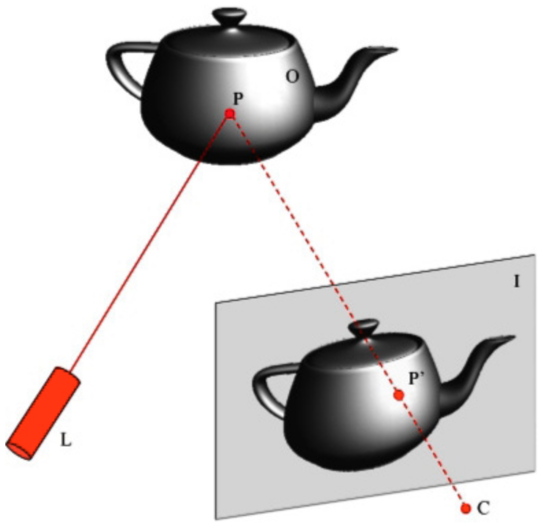


Figure 2.8: Active triangulation where a light source L projects onto a point P in the environment, where it can be seen and easily distinguished in image I [56].

of a foot, and takes approximately 3-4 seconds to complete. A similar system was also produced by Volumental [78]. Both scanner systems are shown in Figure 2.10.

2.4.3 Time of Flight

Time-of-flight sensors are another method of capturing 3D coordinates. They work by projecting light into a scene or onto an object, and measuring how long that light takes to reflect and return to a sensor [56]. This often uses scanning laser such as LIDAR systems, which collects individual points from the surrounding environment. This same idea also exists with time-of-flight cameras, where each pixel on the camera sensor measures the time the light takes to reach it. With time-of-flight cameras, 3D images (depth maps) can be captured, with a similar result as structured light cameras.

2.4.4 RGBD Cameras

RGBD cameras have become very popular in recent years for many applications, most notably with the Kinect [54], which was originally developed as an input device for the XBOX 360 game console. Intel has recently also been developing their own Intel RealSense

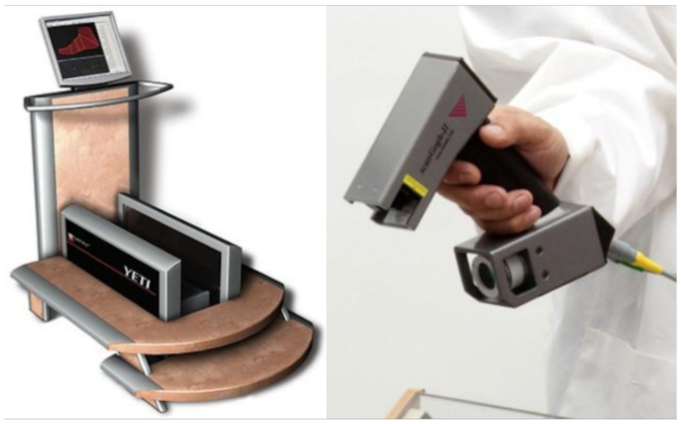


Figure 2.9: Examples of laser scanners. Left: Yeti 3D Scanner [18], Right: scanGogh scanner [17].

RGBD camera systems, with the intention to equip devices to better understand their environments [16]. Examples of RGBD cameras are shown in Figure 2.11. These cameras typically utilize either a structured light or time-of-flight system to supply a depth map image, which is paired with an RGB camera to supply color information.

Due to the abilities of RGBD cameras to quickly deliver depth maps, while being available at a low cost, they have become very popular in 3D scanning. One widely used method is the Kinect Fusion algorithm [58], which reconstructs a scene from a moving RGBD camera's video. This system captures many frames from a scene and is able to produce high quality scans of objects; however, the process can take a long time to complete as the camera must be moved through all necessary viewpoints. Additionally, these moving camera scanning algorithms often suffer from tracking stability issues as well as loop closure problems when circling an object. Multiple RGBD cameras can be used to provide faster scans [1, 12, 46, 78], however a large camera apparatus is then needed, as well as complex calibration and registration techniques to produce final scans, making these systems less ideal for a real world use.

RGBD cameras offer an accessible and capable system for capturing object shape from one prospective. Extending this to capture whole shape using traditional reconstruction techniques brings with it various limitations and complications. With the speed of advances in technology today, recent research demonstrates that limited inputs can be extrapolated in intelligent ways, which may alleviate some complications, and allow for more efficient depth map scanning.

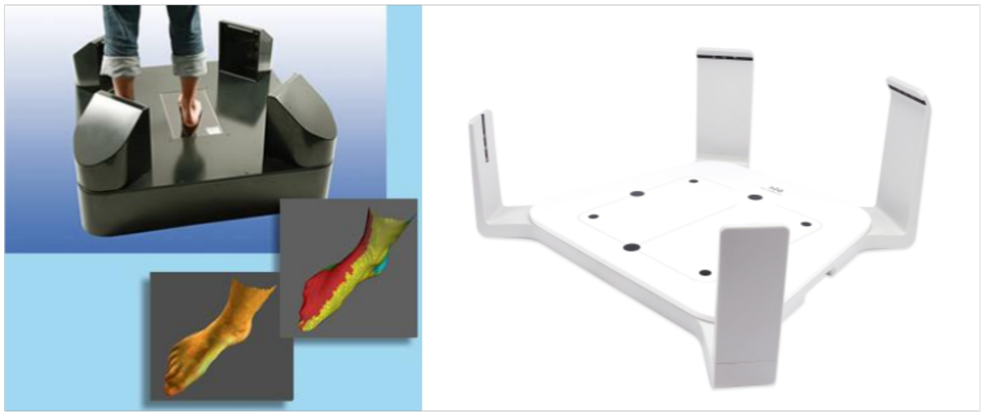


Figure 2.10: Examples of structured light scanners. Left: FotoScan 3D [1], Right: Volumental scanner [78].

2.5 Deep Learning

Machine learning techniques have been in use for many years and applied to a vast number of problems. Despite this, these algorithms have been limited in their ability to process unstructured inputs such as raw natural data. In order to overcome this limitation, the typical solution has been to collect domain specific expertise and to carefully engineer feature extractors that can transform raw data into a form more suitable to traditional machine learning methods [53, 70, 90], such as support vector machines [20], random forests [10] or multilayer feedforward networks [36].

More recently, deep learning has begun to revolutionize the machine learning world, and enable new applications. Deep learning refers to large neural networks that work by combining many layers of neurons which can form many levels of representations describing the input data. The major advantages of deep learning techniques are that they do not require any designed features or data representations, and instead have a very large capacity to learn powerful representations on their own using only a general purpose learning process [44].

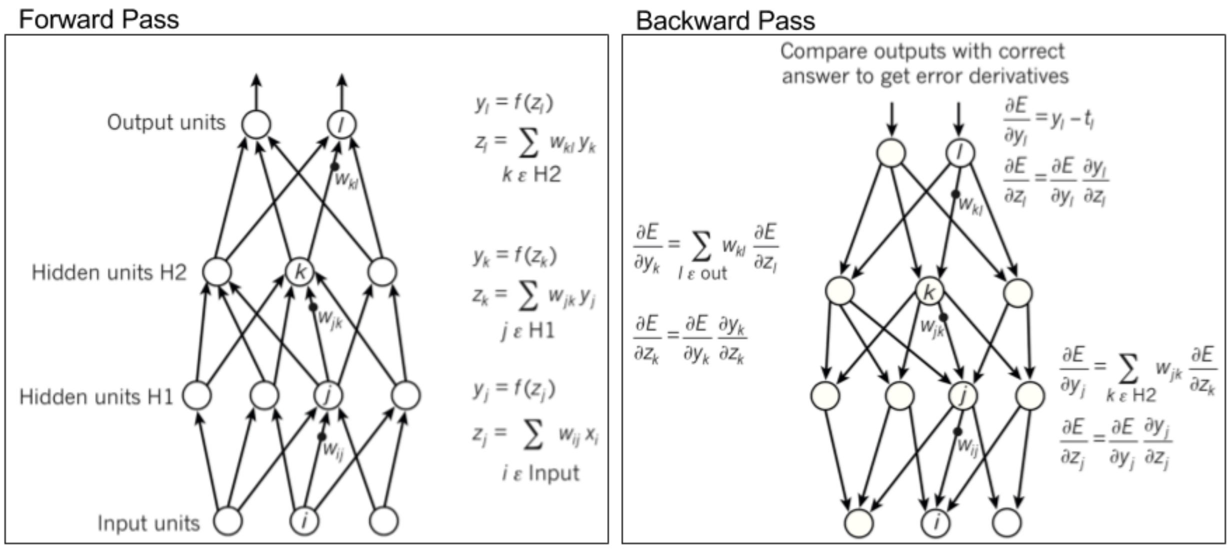
Deep networks are organized by stacking layers of neurons. Input layers are connected to the input data, and feed forward into a set of hidden layers (middle layers) which process the data. The final layer of a deep network is the output layer, which tries to predict some desired output or produce a desired value. Every neuron has a number of input connections, each of which have an associated learned weight, which is used to transform the data in

Figure 2.11: Examples of RGBD cameras. Left: Kinect V1, Right: Intel RealSense F200.

some way. In a standard neural network, each neuron's input is fully connected to the outputs of every neuron in the previous layer. The output of any neuron is computed by first calculating the weighted sum of all inputs, then by applying a non-linear activation function [44]. This non-linear activation function makes each layer non-linear as well as the neural network as a whole non-linear, allowing it to approximate non-linear functions and decision boundaries. In many cases neurons also have a learned bias term that is added to the weighted sum before applying the non-linearity.

Deep neural networks, like most other artificial neural networks are trained using backpropagation [79]. The backpropogation algorithm allows for errors to be propagated from the output layers of a neural network in a backwards fashion, such that the amount of error each parameter in the network is contributing can be determined. This information is useful, as it can be used to update the parameters of any neuron in the network, including those in hidden layers, through optimization algorithms such as gradient descent. Backpropogation works by computing the derivative of total error, with respect to each parameter in the network. This is achieved first by calculating this derivative for the output layer, and using the chain rule to subsequently calculate derivatives of previous layers all the way to the input layer. The process was first described for use in machine learning in 1986 [79]. The forward and backward pass processes are shown in Figure 2.13.

Early deep learning neural networks followed architectures where each layer of neurons were fully connected to previous layers. These networks performed well at tasks such as speech recognition [21, 55], but are less efficient when dealing with data that has local structure such as images. Convolutional Neural Networks (CNN) are a much more efficient and easier to train class of deep network [44]. These networks learn weights for each neuron that are only connected to a local patch of neurons on the previous layer, and all weights



Forward Pass
Output units
Hidden units H2
Hidden units H1
Input units
$y_l = f(z_l)$
$z_l = \sum_{k \in H2} w_{kl} y_k$
$y_k = f(z_k)$
$z_k = \sum_{j \in H1} w_{jk} y_j$
$y_j = f(z_j)$
$z_j = \sum_{i \in Input} w_{ij} x_i$
Backward Pass
Compare outputs with correct answer to get error derivatives
$\frac{\partial E}{\partial y_l} = y_l - t_l$
$\frac{\partial E}{\partial z_l} = \frac{\partial E}{\partial y_l} \frac{\partial y_l}{\partial z_l}$
$\frac{\partial E}{\partial y_k} = \sum_{l \in out} w_{kl} \frac{\partial E}{\partial z_l}$
$\frac{\partial E}{\partial z_k} = \frac{\partial E}{\partial y_k} \frac{\partial y_k}{\partial z_k}$
$\frac{\partial E}{\partial y_j} = \sum_{k \in H2} w_{jk} \frac{\partial E}{\partial z_k}$
$\frac{\partial E}{\partial z_j} = \frac{\partial E}{\partial y_j} \frac{\partial y_j}{\partial z_j}$

Figure 2.12: The forward and backward pass processes for a typical neural network. Forward pass: each neuron computes z as a weighted sum of its input connections, then applies a non-linear activation function $f(z)$ to compute y as its output. Backward pass: the error derivative with respect to each neuron and parameter is computed using the chain rule in a backwards direction. Reprinted by permission from Macmillan Publishers Ltd: Nature [44], copyright 2015.

learned by neurons on a certain layer are shared across the entire layer. In this way, the neurons of a layer essentially learn a filter bank which is convolved across the previous layer's data representation. The weights learning by a CNN layer equate to learning a set of filter kernels. In this way, CNNs are naturally translation invariant, and offer powerful computations with fewer numbers of parameters than previous fully connected networks. An example of a CNN is shown in Figure 2.13. This technique has proven to be extremely powerful on numerous application where data has local structure, most notably in images for computer vision [34, 43, 44, 45].

Despite the foundations of deep learning having existed for many years, a few factors prevented these networks from being viable in many tasks until relatively recently. One of these factors was that deep networks with many parameters took extremely long to train on conventional computers. It wasn't until 2009 that it was demonstrated that deep networks could be trained using relatively affordable and publicly available GPUs, with substantial increases in speed [65]. The second major factor holding back deep learning was

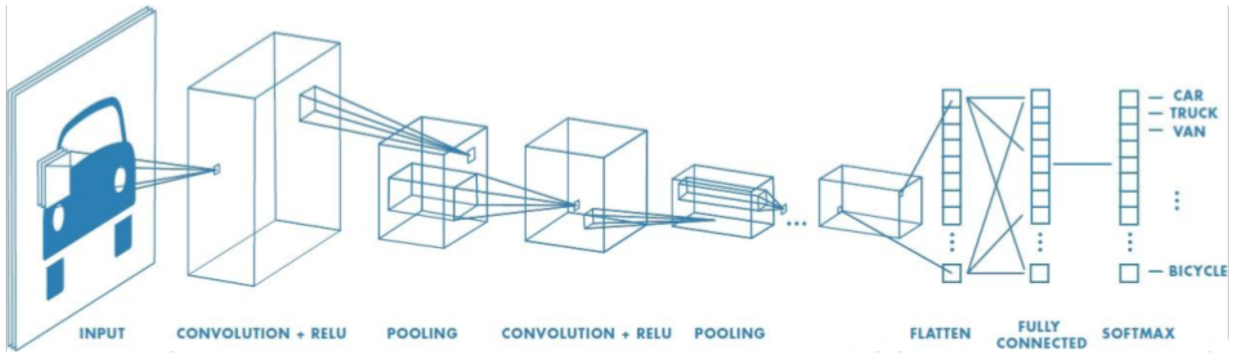


Figure 2.13: An example of a convolutional neural network. CNNs use a set of convolutional layers and often feed into fully connected layers near the output layer for the final processing calculations [38].

that when training a model to perform on raw natural data, especially in computer vision tasks, extremely large training sets were required. Such datasets only became available relatively recently. One such dataset is ImageNet [23], a subset of which forms the basis of the ILSVRC competitions [68]. In 2012, a deep learning model won the classification competition, outperforming the previous state of the art by more than ten percent [43]. This success proved the effectiveness of deep learning models, which have since become the leading technique in almost all recognition and detection tasks [44].

2.6 Shape Completion and View Synthesis

In 3D reconstruction and scanning, there are many complications associated with cost and usability. In recent years, the interest in accessible scanners has drastically increased for applications such as prototyping in 3D graphics to avatar creation to shape capture for 3D printing. From this need, an increasing number of researchers have been exploring methods of producing full object shapes from limited inputs, sometimes referred to as shape completion.

In some cases, object parameters can be captured and used to describe its shape. If available, these parameterized models can be leveraged in learning methods to produce whole models from images by determining a mapping from an image or images to a set of parameters used to deform a template model [24, 25]. The main drawback to these methods is that they are dependent on a complete set of predefined parameters to characterize the object being scanned. In foot shape modeling, these parameters are not easily measured, and thus are not publicly available in any current large datasets paired with images and 3D models. In other words, the general disadvantage of these methods is that they cannot learn shape directly from a set of arbitrary scans or shape models.

In order to learn shape directly, the typical approach involves representing a shape using a voxel volume representation, as shown in Figure 2.14. In deep learning, these voxels can be operated on using 3D convolutional neural networks. These networks have been explored to work directly on limited voxel inputs [22, 66, 81], or from limited input images [14, 66, 75, 80, 83, 84], to form a completed 3D shape voxel representation. These methods have been shown to perform well in shape completion tasks, however their usefulness in applications that require accurate 3D measurement is far more limited. Voxel representations are computationally intensive in deep learning, limiting the output resolutions used in current implementations (usually 32x32x32 voxels or less), although new techniques are starting to be developed to address this [22, 80, 84].

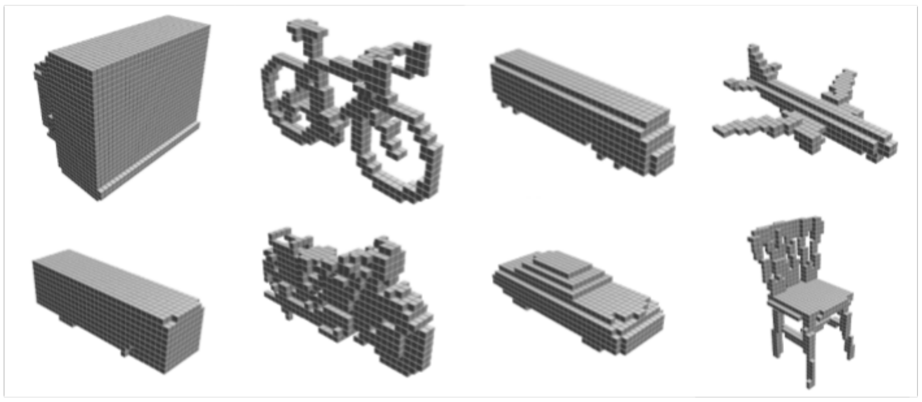


Figure 2.14: Examples of objects represented using voxel volumes (32x32x32) [14].

Another approach to acquiring missing object information is to treat it as a view synthesis problem. The goal of view synthesis is to synthesize a novel view of an object or scene given a single or a set of arbitrary views. This technique is based on how people are able to form complete mental object models, and perform "mental rotation" such that we can imagine objects from unseen perspectives [71]. An example of a mental rotation task is shown in Figure 2.15. Applying this idea to shape completion, missing views of an object can be synthetically scanned to complete it. With this approach, no explicit object parameters are required, making it more generally applicable, and in deep learning, images can contain higher resolutions than voxels for similar computational complexity.

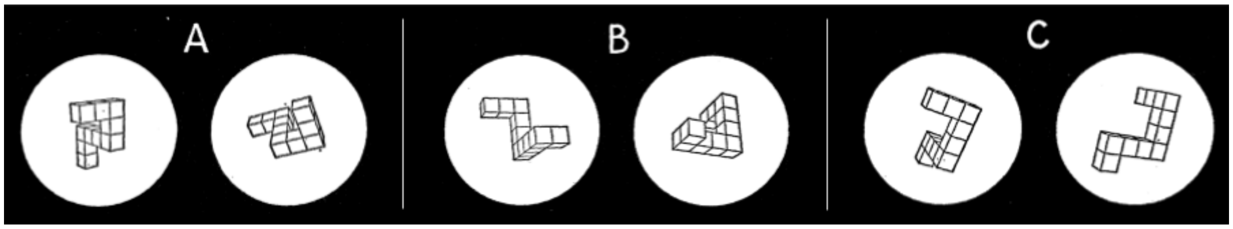


Figure 2.15: Mental rotation example: by imagining each object rotating, it can be seen that the two images shown in case A and case B are of the same object, while the images shown in case C are not [71].

A number of methods have been explored to maximize the visual appearance of a synthesized view, including: appearance flow [63, 89], where the output view samples pixels from the input view, and adversarial networks [88], where the output view attempts to fool a discriminator network [30]. These approaches focus on RGB views however, making it difficult to utilize these methods to extract 3D object shape. Tatarchenko et al. [76] demonstrated how view synthesis can be used to create useful 3D object representations using the process shown in Figure 2.16. Their approach uses a convolutional neural network that takes an RGB view as input and produces a RGBD view as output. This network learned to produce the color and depth values of objects from scratch, which resulted in blurrier reconstructions than those in some other methods [63, 88, 89] but its inclusion of the depth component allowed for the reconstruction of an object from a single image. This concept of using depth map views to represent object shape has also since been explored in generative models [6].

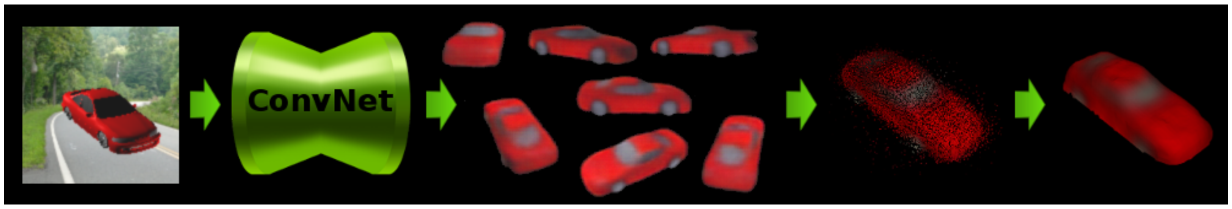


Figure 2.16: View synthesis for object reconstruction by Tatarchenko et al. [76]. A single input image is used to create multiple synthesized RGBD views and form a complete point cloud model.

2.7 Summary

Feet are complex shapes that vary between people, including between people with the same shoe size. The systems used today in characterizing foot and shoe size is insufficient for most people, making finding well fitting shoes a difficult process. 3D scanning can offer more accurate fitting, but the products available today are expensive or slow and cumbersome to operate. The recent advances in machine learning technologies have shown great promise in solving these limitations. With deep learning and view synthesis techniques, there is the potential to create accurate scanners that can operate using only a single input image from one view, thus circumventing many of the limitations and complications of traditional methods.

Chapter 3

Algorithms and Methods

Our primary goal is to develop and implement a method that can alleviate many of the complications associated with 3D object scanning, and apply it to foot scanning for footwear matching. We present two methods that can achieve this by extending the limited input of a single depth map to a full point cloud scan through the use of deep learning. In this chapter we will describe our datasets as well as each of our methods.

3.1 Foot Shape Depth Map Data

There are a large number of complications in designing a low cost 3D foot scanner with as few components as possible. With the advances in machine learning techniques in recent year, it has been shown that we can instead design systems are able to learn shape from limited inputs [6, 14, 22, 63, 66, 75, 76, 81, 83, 84, 88, 89, 91]. In order to apply this idea to creating a simple foot scanning system, arguably the most important component is to obtain a relevant dataset. In this case, a large dataset containing foot shape data from a diverse set of people is required.

The largest available database of 3D human shape scans is the Civilian American and European Surface Anthropometry Resource Project (CAESAR) database [67]. This database was created by the Air Force Research Laboratory (AFRL) with a number of collaborators. The purpose of the database is to overcome limitations with previous shape databases comprised of 2D measures, which lack the ability to accurately extend to volumes and surface areas. The CAESAR database contains a number of demographic information and traditional measurements as well as complete full body scans in three body postures shown in Figure 3.1. The complete CAESAR database is available to anyone, but costs \$20,000 for the complete set of data.

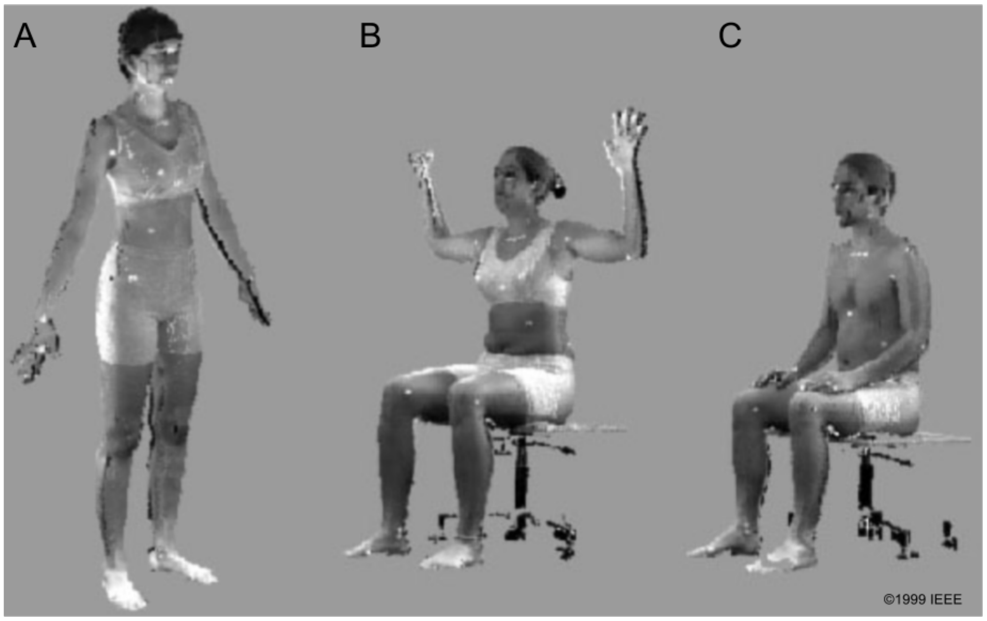


Figure 3.1: The three poses captured in the CAESAR database [67]. A: Standing posture, B: Sitting posture, C: Comfortable working posture.

The CAESAR database has been used by a number of research teams on body shape projects. One such work was that by Pishchulin et al. [64] who developed the MPII Human Shape body models from CAESAR. These body models are fitted statistical models, with average vertex errors of less than 1 cm, learned from the CAESAR 3D scans. These body models learn shape and pose separately, and thus can be manipulated like a puppet. Body models fitted to scans from the CAESAR database are freely available online through their website[1].

We use the 4301 mesh body models from MPII Human Shape [64] that were fit to the standing posture, using the posture normalization of Wuhrer et al. [82]. Each full body model consists of 6449 vertex points, with about 800 vertex points in each foot up to the knee. While the MPII Human Shape models are technically a parameterized shape representation, we do not incorporate these parameters anywhere in our method.

[1]humanshape.mpi-inf.mpg.de

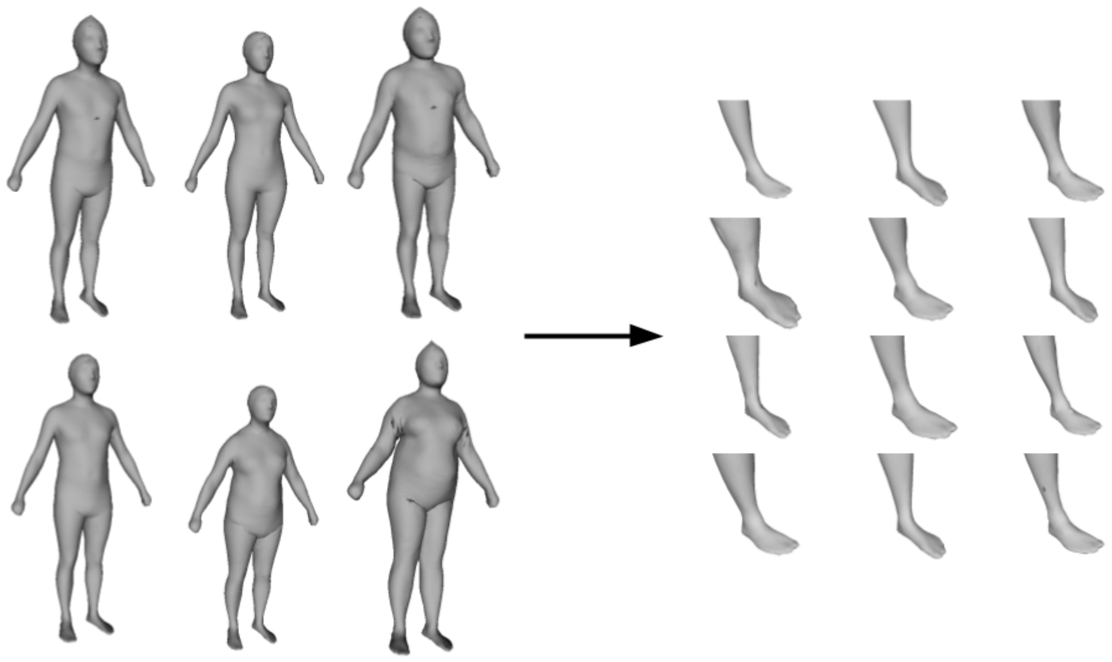


Figure 3.2: The feet of each MPII Human Shape body model [64] is separated and used as an individual foot shape objects for training and testing.

For each of the meshed body models, we isolate the points associated to the left and right feet up to the knee. We then transform each set of foot points such that the origin is the center of the second toe and the heel. This process results in 8602 meshed foot objects to train and test our network, samples are shown in Figure 3.2. Depth map images of the foot objects are rendered using Panda3D [77] before training, with the intrinsic parameters shown in Table 3.1. The input and output views of the foot objects are randomly rendered using pose parameters specific to which of our methods is being trained.

Table 3.1: Intrinsic camera parameters used in the virtual scanning camera.

Parameter	Value (in pixels)
Image width and height	128
Focal length X and Y	192
Optical center X and Y	64
Distortions	none

Our dataset of 8602 feet was separated by individuals such that both of a person's feet would be in the same set. We do this to ensure that our models do not extend any information it could have learned on one of a person's legs in the training set to the other in the test set. In this way, at test time all instances are from completely new people, similar to what would be seen if it were to be deployed into the wild. 80% of the data was used for training and 20% for testing.

3.2 Point Cloud Completion

Following the idea that learning models can extend limited inputs about an object's shape and provide complete reconstructions, we develop a technique for completing an object point cloud, given incomplete information. In our point cloud completion method, our goal is to incorporate the information from an input depth map as much as possible into our final point cloud. We do this by taking advantage of the shape of a foot to select specific views that maximize our ability to fill in missing points using as few synthesized views as possible. In this method, we also introduce a way to synthesize a missing object view without the need to align the new view to that of the input when reconstructing the overall point cloud.

We frame the problem of completing a limited foot scan point cloud as a depth map view synthesis problem. We leverage the power of deep learning to implicitly learn foot shape and the relationships in how it can appear between views. Our input depth map view configuration is shown in Figure 3.3. A foot is placed at the origin, such that depth map images can be taken from camera poses at various azimuth, elevation and roll angles, as well as at varying radii.

Unlike general view synthesis problems, we restrict our input views to be only profile views of the foot rather than any arbitrary view. The profile of a foot contains a significant amount of information about overall shape [51], which a learning system can take advantage of. Additionally, our synthesized output views are always of the surface on the direct opposite side of the foot from the camera, produced as if it could be seen through the near side of the foot as shown in Figure 3.4. Due to how feet are shaped, these two opposite profile views of the foot contain the vast majority of points on the object's surface, as the foot has minimal self occlusions along this direction. In framing the problem in this way, the synthesized depth map can also be re-projected to the same coordinate system as the input view to produce a near complete point cloud of the foot surface without the need for any extrinsic parameters or alignment.

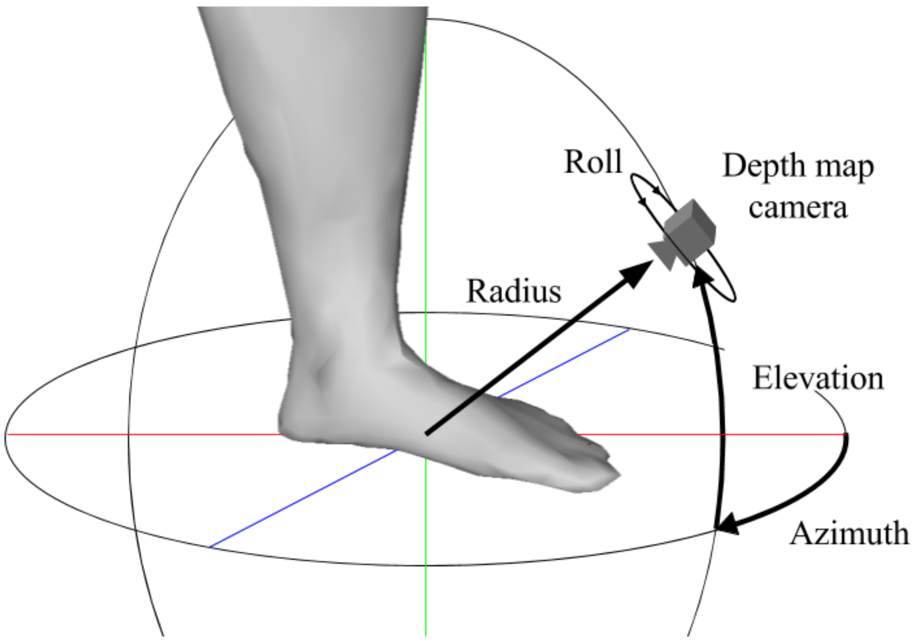


Figure 3.3: Point cloud completion depth camera pose configuration.

Depth maps are rendered using the camera poses described in Table 3.2. Additional Kinect noise [59] is added to the input depth map. To generate the output depth map data, we first render a depth map from the same radius but opposite azimuth, elevation and roll angles as the input. The 3D position of each of these points is then projected onto the image plane of the input depth map, with depth values calculated based on relative pose. This opposite depth map is rendered at double resolution (in this case 256x256), to try and mitigate quantization artifacts when projecting points, although some of this still appears in the final ground truth depth maps.

Table 3.2: Camera pose parameters for the network input.

Pose Parameter	Value Range	Step
Radius (mm)	640 to 700	15
Azimuth (deg)	70 to 110 or 250 to 290	1
Elevation (deg)	-20 to 20	1
Roll (deg)	-5 to 5	1

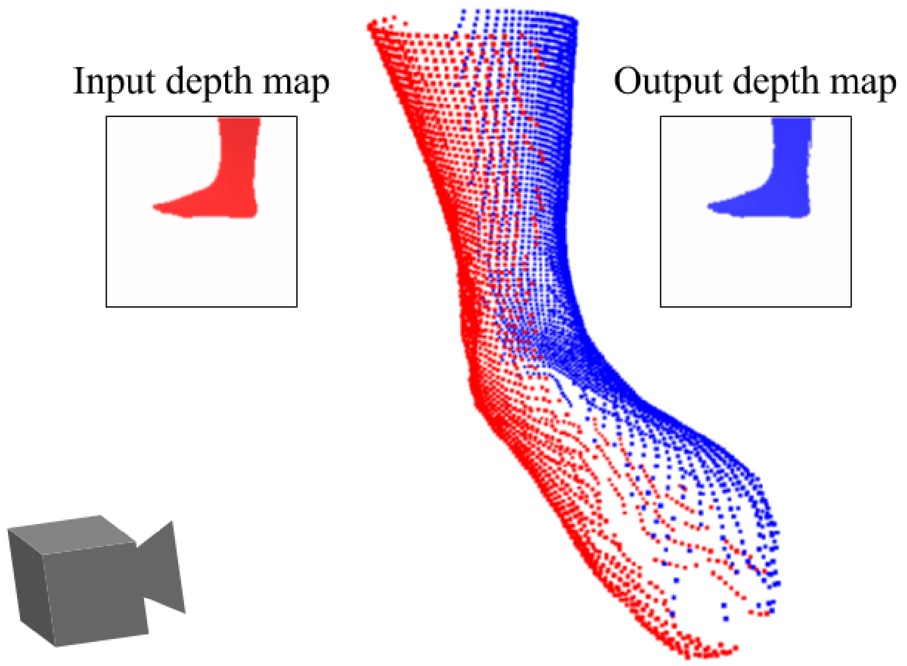


Figure 3.4: Depth map input/output configuration. Red: points from the input depth map, Blue: points from the synthesized output depth map.

Our initial basic network architecture is similar to that by Tatarchenko et al. [76]. We use a deep neural network of convolutions followed by deconvolutions with fully connected layers in the middle to process an input depth map and synthesize an output depth map as shown in Figure 3.5. We train directly on a one channel depth map input, to produce a one channel depth map output that corresponds to the opposite foot profile surface. In this implementation, we do not incorporate any color information that would be present in a typical RGBD image. We discuss changes to this network architecture in Section 3.4.

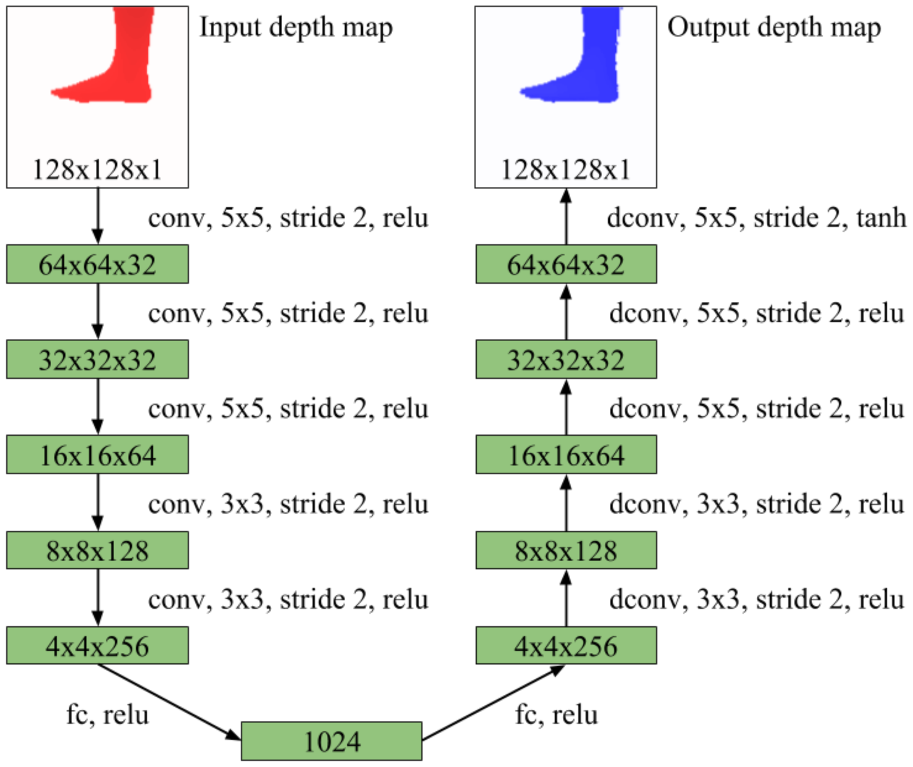

Input depth map
128x128x1
conv, 5x5, stride 2, relu
64x64x32
conv, 5x5, stride 2, relu
32x32x32
conv, 5x5, stride 2, relu
16x16x64
conv, 3x3, stride 2, relu
8x8x128
conv, 3x3, stride 2, relu
4x4x256
fc, relu
1024
fc, relu
4x4x256
dconv, 3x3, stride 2, relu
8x8x128
dconv, 3x3, stride 2, relu
16x16x64
dconv, 5x5, stride 2, relu
32x32x32
dconv, 5x5, stride 2, relu
64x64x32
dconv, 5x5, stride 2, tanh
128x128x1
Output depth map

Figure 3.5: Point cloud completion network architecture.

When reconstructing the complete point cloud, we re-project the input depth map and the synthesized output depth map using the same camera parameters, as they are already aligned. We also remove outliers and clean the point clouds using 3D cropping and MATLAB's *pcdenoise* function. Merging the points from the input and synthesized output produces a near complete foot point cloud.

We implemented our network in Tensorflow [2] on a Linux machine running an Nvidia K80 GPU. Training was done using a mini batch size of 64 and the Adam optimizer [41] with a learning rate of 5e-5. Our loss function was the mean L_1 distance between the output depth map pixels values and ground truth. Our results applying this point cloud completion method are presented and discussed in Section 4.2.

3.3 Point Cloud Synthesis

In our point cloud synthesis method, our goal is to train a model to fully learn the overall shape of an object from one input depth map view. We frame the problem of extrapolating a limited foot scan into an entire point cloud as a view synthesis problem. We leverage the power of deep learning to implicitly learn foot shape, and relationships between how it can appear from view to view.

Unlike our point cloud completion method, the input depth map is not aligned or added to the final point cloud. We do this so as not to complicate matters by worrying about the pose of the input view. This technique synthesizes depth maps from many views around the surface of the foot, covering the entire surface with overlap between views. This approach prevents regions with missing points, but also mitigates the effects of inaccurate regions, by allowing redundancy with view overlap.

Our approach is similar to Tatarchenko et al. [76], however we focus on the 3D information in the depth map as input as opposed to RGB values, and apply it to foot scanning with a single depth map camera. We propose a method that uses a deep neural network to take as input the shape information from a depth map and synthesizes novel shape information through an output depth map. This shape to shape approach should contain more information about the 3D structure of a foot than the image to image approach seen in other view synthesis approaches, and thus be more practical in object scanning. Additionally our model is not told explicitly how to represent shape, allowing it to be trained directly from a set of non-parameterized arbitrary scans or shape models.

Our depth map view configuration is shown in Figure 3.6. A foot is placed at the origin, and depth map images are taken from camera poses at various azimuth and elevation angles, as well as at varying radii. We also allow for variations that reflect real world imperfect camera mounting, where its orientation can have additional roll, pitch angle offset and heading angle offset, rather than always looking directly at the foot object. Objects are first scaled down to 0.003 size (from mm units) before rendering. In training, input views of the foot are randomly distributed in azimuth angle, elevation angle and radius, while also randomly having some degree of roll, pitch offset and heading offset. Our method then estimates any desired depth map view from another camera pose of the same foot object at an arbitrary azimuth angle and elevation angle. We train our network to produce output views that do not contain any imperfect framing from roll or offset. Table 3.3 describes the camera pose parameters used to render the input and output depth map images.

Our network architecture is shown in Figure 3.7, and is similar to [76]. Our network takes in as input an arbitrary depth map view of an object denoted x, and an encoding

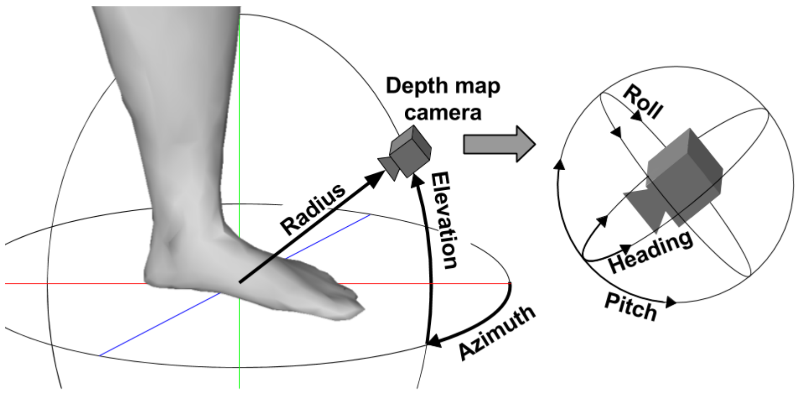


Figure 3.6: Point cloud synthesis depth camera pose configuration.

Table 3.3: Rendering camera pose parameter ranges for the network input and output (angles are in degrees).

Parameter name	Input	Output
Azimuth	0, 5, . . . , 355	0, 20, . . . , 340
Elevation	-30, -25, . . . , 40	-30, -20, . . . , 40
Roll	-5, -4, . . . , 5	0
Radius	1.9, 1.95, . . . , 2.1	2
Pitch offset	-2, -1.5, . . . , 2	0
Heading offset	-2, -1.5, . . . , 2	0

specifying the desired view camera pose denoted θ. From this input, the network provides an estimate for the depth map at the desired pose denoted y. We train a convolutional neural network encoder and decoder, where the encoder encodes the input depth map x into a vector representation. The desired view camera pose θ is encoded by a set of fully connected layers before being appended to x's vector representation. The decoder processes this new vector representation, and uses deconvolutions [86] to synthesize an output depth map of the same size as the input depth map.

In order to completely reconstruct the foot object from a single view, k forward passes through our network are required. For any input image x, a set of desired view camera poses $\theta_{1,2,\ldots,k}$ must be specified that would sufficiently cover the object being scanned, where including overlap between views can produce a cleaner scan. Our network is then used to estimate the scans $y_{1,2,\ldots,k}$, through k forward passes. These k depth map estimates

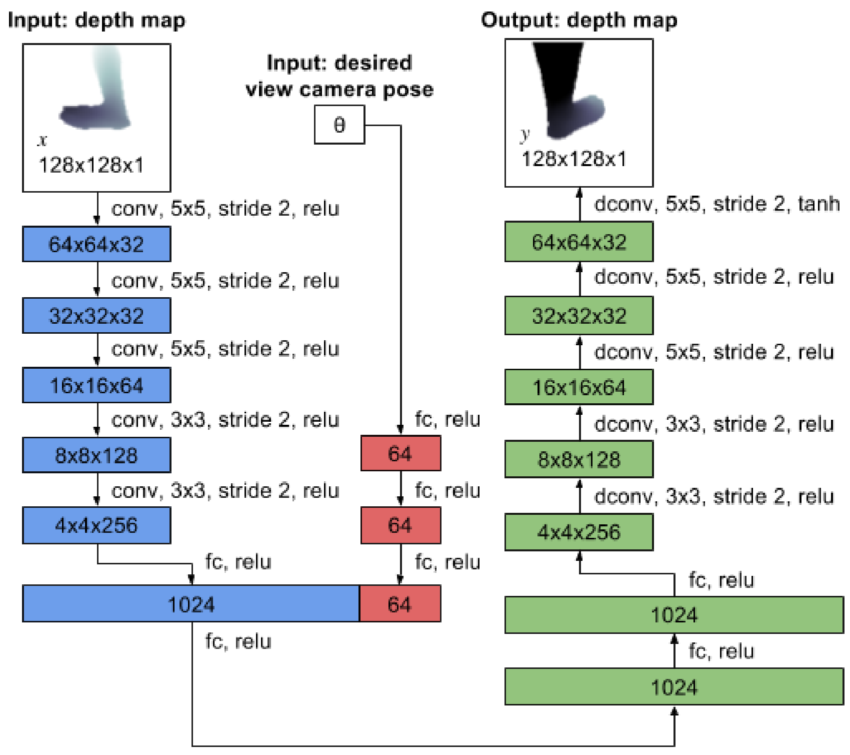

Input: depth map
x
128x128x1
Input: desired view camera pose
θ
Output: depth map
y
128x128x1
conv, 5x5, stride 2, relu
64x64x32
conv, 5x5, stride 2, relu
32x32x32
conv, 5x5, stride 2, relu
16x16x64
conv, 3x3, stride 2, relu
8x8x128
conv, 3x3, stride 2, relu
4x4x256
fc, relu
1024
fc, relu
fc, relu
64
fc, relu
64
fc, relu
64
dconv, 5x5, stride 2, tanh
64x64x32
dconv, 5x5, stride 2, relu
32x32x32
dconv, 5x5, stride 2, relu
16x16x64
dconv, 3x3, stride 2, relu
8x8x128
dconv, 3x3, stride 2, relu
4x4x256
fc, relu
1024
fc, relu
1024

Figure 3.7: Point cloud synthesis network architecture.

are then reprojected and registered using the intrinsic parameters of the camera used to create the training data and the camera poses encoded in $\theta_{1,2,\ldots,k}$, resulting in a complete point cloud.

Our network was implemented in Tensorflow [2] on a Linux machine with an Nvidia K80 GPU. Training was done with mini batch sizes of 64 using the Adam optimizer [41] and a learning rate of 5e-5. Our loss function was the mean L_1 difference between the output depth map pixels and the ground truth.

When reconstructing the entire point cloud of each foot object we use a set of $k = 24$ desired viewpoint scans. We run our network to estimate scans positioned at a radius of 2, every 45 degrees in azimuth angle for elevation angles of -30, 0 and 30 degrees. We further use 3D cropping and MATLAB's *pcdenoise* function to remove outliers and clean our final representation. Our results applying this point cloud synthesis method are presented and discussed in Section 4.3.

3.4 Network Architecture Improvements

In this work, we expand on our point cloud completion work and explore the impact of using different network architectures. We implement two new architectures utilizing residual blocks, which are a fundamental component of ResNet [34], one of the most powerful deep networks to date. Residual blocks are a layer configuration characterized by a set of convolutional layers, with a skip connection passing around them, as shown in Figure 3.8. The motivation behind the residual block is that it allows for information to flow around a set of layers, giving that block the possibility to become closer to an identity mapping if that is optimal. The benefit to this idea is that in deeper models (which have more powerful learning capacities), additional layers can approximate identity mappings if needed, and thus should have at most the same error as a shallower network [34], rather than degrade in performance as is typical when adding many layers.

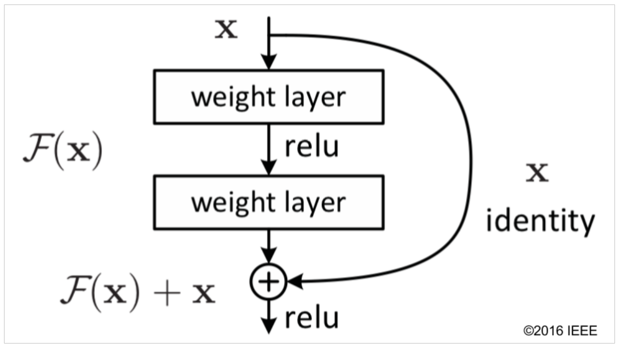


Figure 3.8: A residual block [34].

We train a deep neural network to take as input a depth map of the profile view of a foot, and to synthesize a depth map of the same foot from the opposite viewpoint (as discussed in Section 3.2). The points from this second depth map can be merged with the points from the input depth map to create a near complete 3D point cloud. To simplify the process of registering the two sets of points, we synthesize the points from the second depth map from the same camera pose as the input. Figure 3.4 illustrates this process. We focused our network architecture exploration on our point cloud completion method primary because this method has more room to improve in terms of point cloud accuracy than our point cloud synthesis method (discussed further in Sections 4.2 and 4.3).

Our first architecture, denoted Net1, closely follows that used in previous RGBD view synthesis works [76], and was used in our previous works from Section 3.2. This network encodes information all the way down to a compact 1 dimensional representation before decoding the output depth map. This network does not use any batch normalization [39].

Our second architecture, denoted Net2, uses an architecture similar to ResNet-34 [34] as its encoder and decoder. Sixteen residual blocks of two 3x3 convolutional layers are used in both the encoder and decoder, with projection shortcuts used to match dimensions. Upscaling in the decoder's residual blocks is done using strided deconvolutional [86] layers. We do not include any fully connected layers in Net2 in order to save parameters, as well as to allow for more 2D spatial information to pass through the network. Batch normalization was used in all layers except for the last three residual blocks and the final deconvolution. We found that batch normalization on the final layers made it difficult for the output depth map pixels to take on the necessary values. This network contains less parameters than Net1, but contains far more processing at each representation size.

Our third architecture, denoted Net3, uses less residual blocks than Net2, and does not downscale its representations to as small a size, allowing it to have far less parameters than either Net1 or Net2. Here we again use batch normalization everywhere except on the last three residual blocks and the final deconvolution. In all three architectures, ReLU activations are used in all layers except for the final deconvolution, which uses tanh. Our network architectures are shown in Table 3.4.

We implement our networks in Tensorflow [2] on a Linux machine with an Nvidia K80 GPU. Training was done with mini-batch sizes of 64 and the Adam optimizer [41] in all cases. We used the mean L1 difference between the output and the ground truth pixels as the loss function. Net1 used a learning rate of 5e-5, while Net2 and Net3 used 5e-4.

As described previously, we construct a complete point cloud using our networks, we first project both the input depth map and output depth map to world coordinate points. We then post process the points using MATLAB's *pcdenoise* function and 3D cropping. The combination of the input depth map points, and the synthesized depth map points create a near complete overall point cloud. Our results applying these network architectures to our point cloud completion method are presented and discussed in Section 4.4.

3.5 Real World Data Capture

As an additional experiment to explore the utility of our methods, we explore their performance on real world scan data. In order to evaluate this, we collected a small set of scans from the feet of several graduate students. The collected scans are used purely to test with, as collecting enough data to train on would be a very large undertaking. We

Table 3.4: Network architectures used. Abbreviations used: Convolution (c), Deconvolution (d), Stride of 2 (/2), Fully connected (fc), Residual Block (res), Residual Block with deconvolutional up-scaling (dres).

Output Size	Net1	Net2	Net3
128x128	Input	Input	Input
64x64	c 5x5, 32, /2	c 7x7, 16, /2	c 7x7, 16, /2
	-	res, 16	res, 16
	-	res, 16	res, 16
	-	res, 16	res, 16
32x32	c 5x5, 32, /2	res, 32, /2	res, 32, /2
	-	res, 32	res, 32
	-	res, 32	res, 32
	-	res, 32	res, 32
16x16	c 5x5, 64, /2	res, 64, /2	res, 64, /2
	-	res, 64	res, 64
	-	res, 64	res, 64
	-	res, 64	-
	-	res, 64	-
	-	res, 64	-
8x8	c 3x3, 128, /2	res, 128, /2	-
	-	res, 128	-
	-	res, 128	-
4x4	c 3x3, 256, /2	-	-
1024	fc	-	-
4096	fc	-	-
8x8	d 3x3, 128, /2	res, 128	-
	-	res, 128	-
16x16	d 3x3, 64, /2	dres, 64, /2	-
	-	res, 64	-
	-	res, 64	-
	-	res, 64	-
	-	res, 64	res, 64
	-	res, 64	res, 64
32x32	d 5x5, 32, /2	dres, 32, /2	dres, 32, /2
	-	res, 32	res, 32
	-	res, 32	res, 32
	-	res, 32	res, 32
64x64	d 5x5, 32, /2	dres, 16, /2	dres, 16, /2
	-	res, 16	res, 16
	-	res, 16	res, 16
	-	res, 16	res, 16
128x128	d 5x5, 1, /2	d 7x7, 1, /2	d 7x7, 1, /2
	Output	Output	Output
Parameters	9,286,977	2,666,785	581,153

do not expect to observe performance at the same level as was seen when testing on our MPII Human Shape foot dataset. In this case, models are trained on MPII Human Shape data but tested on real world data, which are similar but not necessarily from the same domain, and thus are likely to not be fully recognized or understood by our networks.

We utilize a Kinect V1 RGBD camera and Kinect Fusion [58] to collect our scans. We originally experimented with a multiple-camera system, but found Kinect Fusion to be simpler to achieve accurate scans (under controlled conditions). We scan each persons feet while they are wearing socks, as the MPII Human Shape feet are largely smooth, and do not contain many fine features such as toes (as shown in Figure 3.2). Each person is scanned while sitting on an elevated surface with their feet hanging well above the ground, as shown in Figure 3.9. This is done such that we can fully scan the entire surface of the foot, including the underside. We try to match MPII Human Shape as much as possible, and instruct each person to keep their foot at a 90 degree angle to their leg, with as little toe flex as possible.

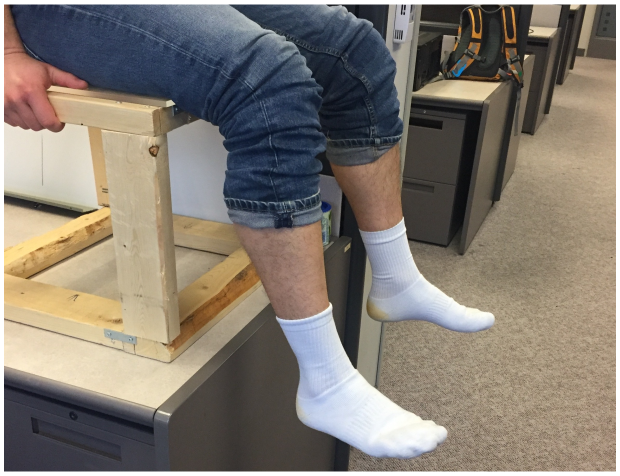


Figure 3.9: Seated pose used to capture a complete foot scan using Kinect Fusion.

We found that great care must be taken during scanning to capture usable data. One major difficulty with Kinect Fusion is that it often loses tracking during a scan, so scans must be done very slowly, without large movements. We also found that scans often suffered from holes as well as loop closure issues, where the surfaces of the foot do not line up in the final scan, resulting in a distorted shape, as shown in Figure 3.10. For these reasons, each foot scan often required multiple attempts to properly capture, and even so, despite our best efforts, some distortions are still present in our final scans. These findings are in line with the limitations motivating the development of our single view scanner methods. Once scanned, foot models are manually isolated, by cropping away background elements and the mesh coordinates are transformed to be centered at the origin. Examples of the real world scans we captured are shown in Figure 3.11. Each scan mesh has around 800,000 veticies and 250,000 faces. Our results applying both of our single view scanning methods to this scan data are presented and discussed in Section 4.5.

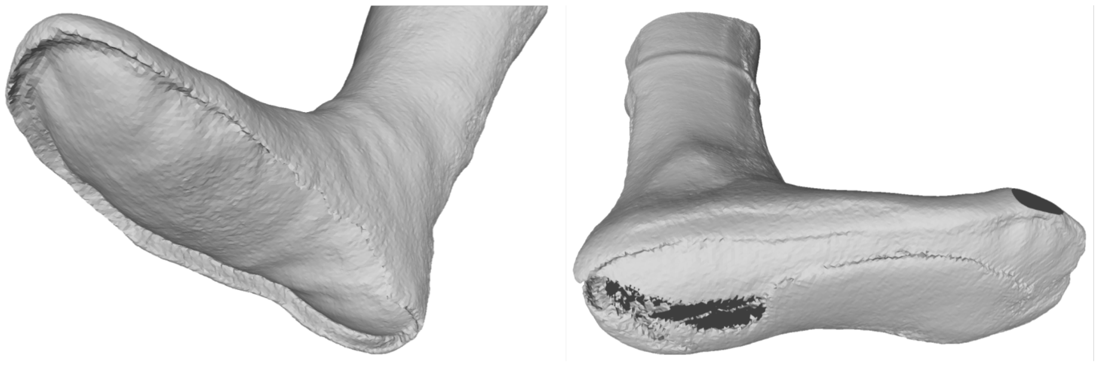


Figure 3.10: Common failures we observed while using Kinect Fusion. Left: Distortions in object shape caused by loop closure issues. In this case the bottom surface of the foot is not aligned to the outer surfaces. Right: Holes in the objects surface. In this case holes can be seen on the heel and on the toe.

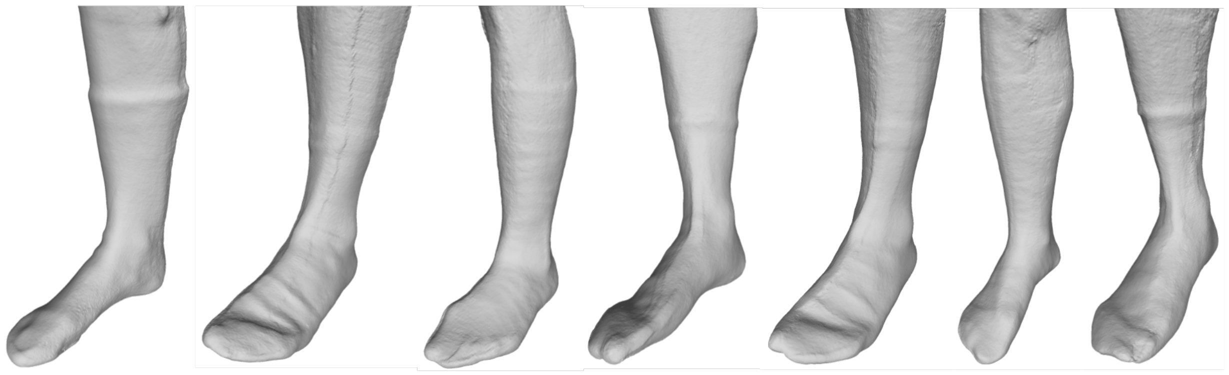


Figure 3.11: Foot meshes from real people captured using Kinect Fusion [58].

3.6 Summary of Methods

We have presented two deep learning based methods that are designed specifically to eliminate many of the complexities associated with 3D foot scanners that use traditional techniques. Utilizing the MPII Human Shape [64] body models of the CAESAR database [67], we create a dataset of 3D foot objects. We present two separate methods of leveraging our foot dataset to create deep models that can produce complete scans from a single depth map input. We additionally discuss network architecture improvements that may improve performance in our methods. In order to explore the usefulness of our method in the real world, we capture a small set of foot scans to test our methods on. Results of our experiments and testing of each of our methods will be discussed in Chapter 4.

Chapter 4

Results

We have presented two different deep learning methods of producing full point cloud scans from a single input depth map. Each method is trained using only depth maps generated from a set of 3D objects. In this chapter we will describe the metrics we use to evaluate performance, and discuss the results of each of our methods applied to both MPII Human Shape foot data [64], as well as our real world scans.

4.1 Error Metrics

Our test set consists of 1720 random foot objects not used during training. The accuracy of our outputs are evaluated in two ways. First, we evaluate the network's ability to generate new depth map views given an input depth map view. Second, we evaluate our method's ability to reconstruct a foot point cloud given a single input depth map view.

In order to evaluate our network's ability to generate depth maps, we use 64 random input-output pairs for each foot in the test set. Our depth map error measure is the mean L_1 distance between the output depth map pixels values $\hat{d}_i$ and ground truth d_i values.

$$\mathcal{L} = \sum_i \|d_i - \hat{d}_i\|_1. \tag{4.1}$$

Our point cloud shape error measure is similar to that used by Luximon et al. [50], who used a statistical model to parameterize a foot point cloud based on four measures. We use a two directional nearest neighbor Euclidean distance metric to measure similarity between point clouds. For each point $\mathbf{p}_{syn,i}$ in the synthesized point cloud, we calculate its

Euclidean distance to the nearest point in the ground truth point cloud, and for each point $\mathbf{p}_{gt,j}$ in the ground truth point cloud, we calculate its Euclidean distance to the nearest point in the synthesized point cloud using the following equations:

$$e_{syn,i} = min_j \|\mathbf{p}_{syn,i} - \mathbf{p}_{gt,j}\|_2, \tag{4.2}$$

$$e_{gt,j} = min_i \|\mathbf{p}_{gt,j} - \mathbf{p}_{syn,i}\|_2, \tag{4.3}$$

where $e_{syn,i}$ is the error for point $\mathbf{p}_{syn,i}$ in the synthesized point cloud to the ground truth, and $e_{gt,j}$ is the error for point $\mathbf{p}_{gt,j}$ in the ground truth to the synthesized point cloud. We normalize the distance measures by the number of points in each cloud, then average the two measures to form our overall error of our point cloud estimate using the following equation:

$$e_{total} = \frac{\frac{1}{N}\sum_i e_{syn,i} + \frac{1}{M}\sum_j e_{gt,j}}{2}, \tag{4.4}$$

where N and M are the number of points in the synthesized and ground truth point clouds respectively, and e_{total} is the total error reported for a synthesized point cloud. In order to calculate this point cloud metric efficiently, a KDTree [7] implementation is crucial for locating nearest neighbour points in 3D space with reasonable computational complexity. In our works, our point clouds use mm units and thus we report our error measurements in mm units.

Neither of these metrics are invariant to object pose or position. In our point cloud error metric, our metric is most accurate when both object point clouds are centered and oriented in the same way. In our case, the training data foot objects have a fairly consistent pose that our networks can predict, so this sensitivity in the metric has a minimal effect on the reported error.

4.2 Point Cloud Completion Results

To test our point cloud completion method described in Section 3.2, we generated random images for each foot object in the test set using the same camera pose parameters that were used in training. After training, our depth map error on the test set was 0.0054. Samples of synthesized depth map results are shown in Figure 4.1, along with the distribution of error within the depth maps. Looking at the error distributions, it can be seen that a large portion of the error is due to pixels on the foot outline. It appears that the network is uncertain whether pixels in these regions would be a part of the background or of the foot.

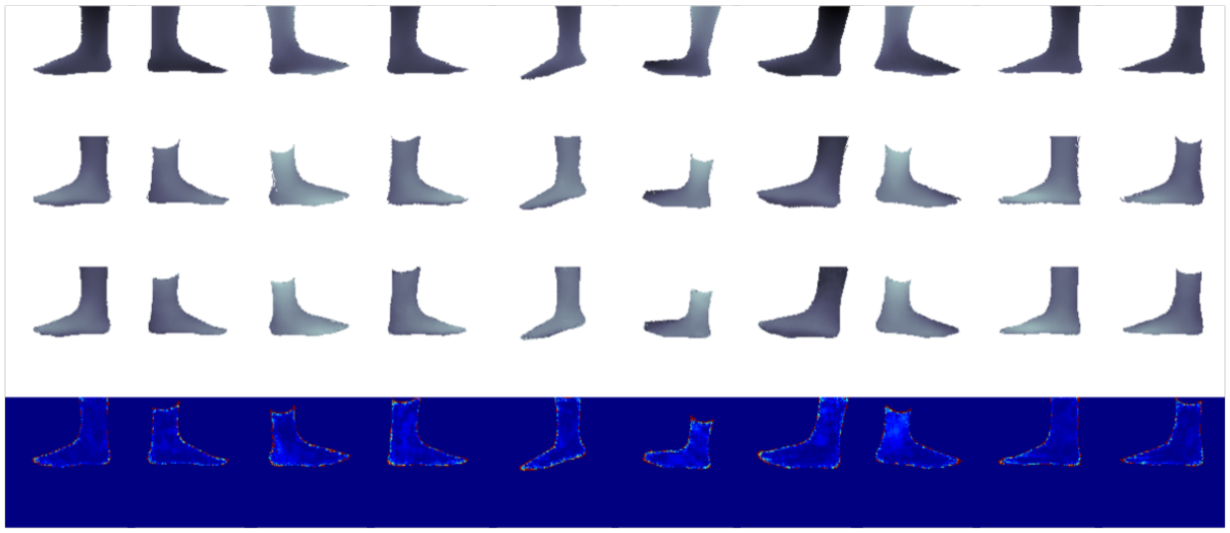


Figure 4.1: Sample point cloud completion depth map results. First row: input depth map, Second row: ground truth, Third row: synthesized output depth map, Fourth row: output depth map error distribution.

For each test image, we additionally evaluated the network outputs in their ability to create shape accurate point clouds. Each depth map from the test set is re-projected to a point cloud in mm units. Sample point clouds are shown in Figure 4.2, compared with the ground truths. We found that across all the images in our test set, our method produced points clouds with an average error of 2.92 mm with a standard deviation of 0.72 mm. These results are more precise than the 4.23 mm half size increment used in the English and American shoe sizing systems, however they may not be accurate to the half sizes of 3.33 mm in the European sizing system [31]. Looking more closely at Figure 4.2, it can be seen that the synthesized point clouds perform accurately in areas that are dense with points, such as along the side of the foot, but has more difficulty producing points in sparse regions, such as along the top and bottom of the foot. These sparse regions corresponded to the high error points along the outlines of the depth maps and always occur in these areas as a by-product of how we chose our profile camera poses. We can additionally see the seam between the input and synthesized point clouds. This seam occurs along the surface of the foot that is at very high angles to the depth map camera, where reliable depth measurements cannot be taken by the input camera [59] and where the synthesis network becomes uncertain. Due to this, points in this region are often noisy and filtered out when creating the final point cloud.

This seam also appears in the ground truth point clouds, as there will always be a point along the foot surface that is perpendicular to both profile camera views. This seam prevents this method from being able to fully reconstruct the foot point cloud on its own, but additional methods can be leveraged to fill in the missing regions [12], so long as the halves are aligned. Even leaving the seam as is, accurate measurements can be taken across the two surfaces and operations such as virtual fitting are still possible.

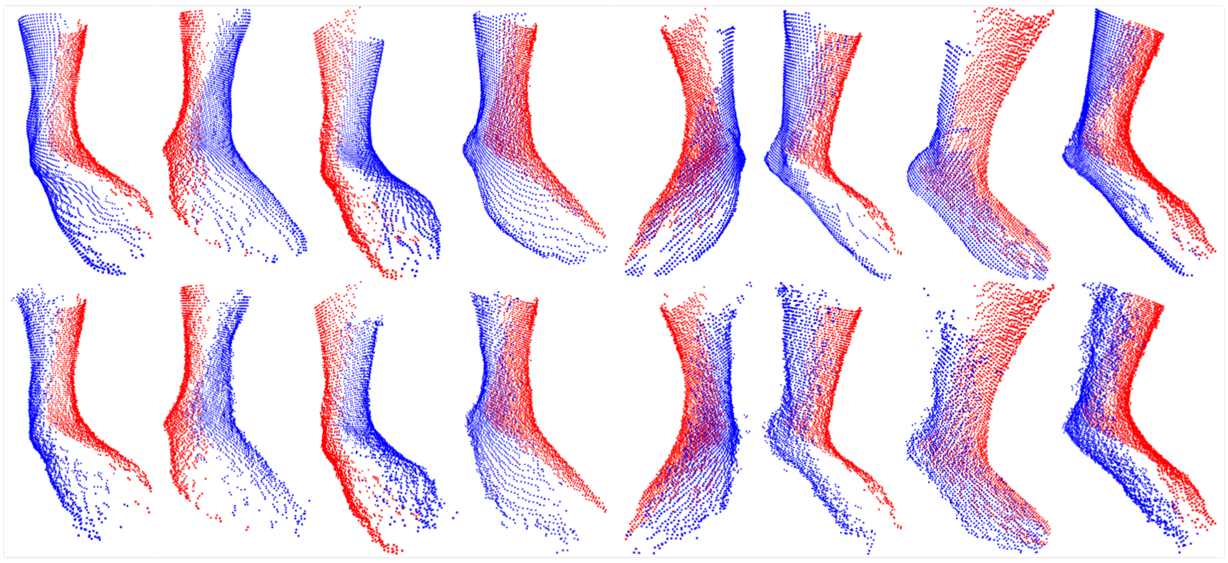


Figure 4.2: Sample point cloud completion results. First row: ground truth, Second row: synthesized point cloud. Red: input depth map points, Blue: ground truth/synthesized output depth map points.

4.3 Point Cloud Synthesis Results

In order to evaluate our point cloud synthesis method described in Section 3.3, we first test our network's ability to generate novel depth map views, we use 64 random input-output pairs for each foot object in the test set. These random test images are strictly used to test the view synthesis ability of our network, and are not used to reconstruct the complete 3D foot shape. After training, our error on this test set was 0.0072. Samples of the depth maps from the network are shown in Figure 4.3, as well as the distribution of the error across the depth maps. It can be seen that the majority of the error comes from the pixels around the outline of the foot. It appears that in these regions the network is uncertain whether these pixels should belong to the foot or the background. These high error outline pixels do not pose much of a problem when forming the foot point cloud however, as they are easily filtered out in our post processing step. Unlike in our point cloud completion approach, filtering out these points does not create sparse regions or holes in the point clouds to the same extent. This is because in this approach we synthesize more depth maps and thus more points on the foot surface. These additional views provide additional overlap and redundancy allowing for a more dense point cloud.

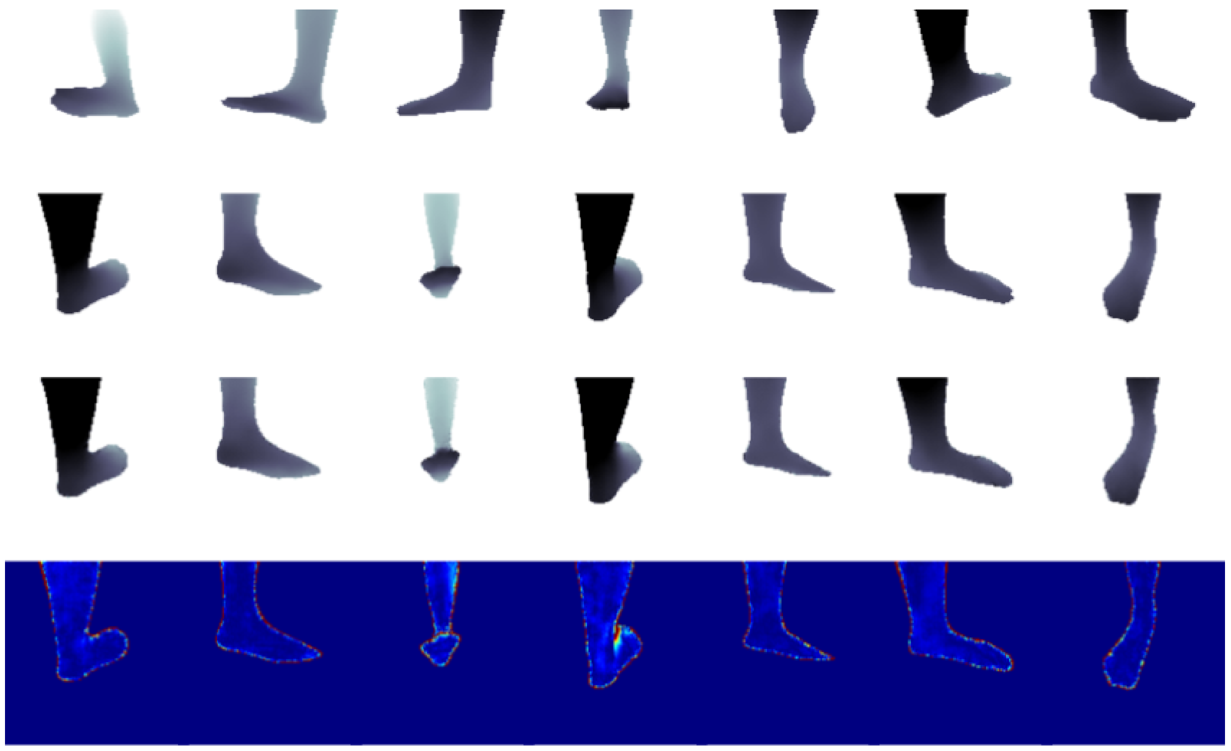


Figure 4.3: Sample point cloud synthesis depth map results. First row: input depth map, Second row: ground truth, Third row: synthesized output depth map, Fourth row: output depth map error distribution.

In order to evaluate our network's ability to reconstruct the complete foot point cloud, we used a single input depth map, and generate a point cloud from 24 synthesized depth maps, as described previously. We additionally test using 24 different camera poses as the single input viewpoint to explore what camera placement works best for foot scanning.

Our point cloud error is calculated by comparing the estimated point cloud against the point cloud formed by the ground truths for the same 24 depth maps. We found that from the best input view point, our method produced point clouds with an average error of 1.55 mm with a standard deviation of 0.41 mm. These results are significantly more precise than the 4.23 mm half size increment used in the English and American shoe sizing systems, as well as the half sizes of 3.33 mm in the European sizing system [31]. Sample point clouds are shown in Figure 4.4. A breakdown of the point cloud error for estimates generated from each of the 24 input views explored is shown in Figure 4.5.

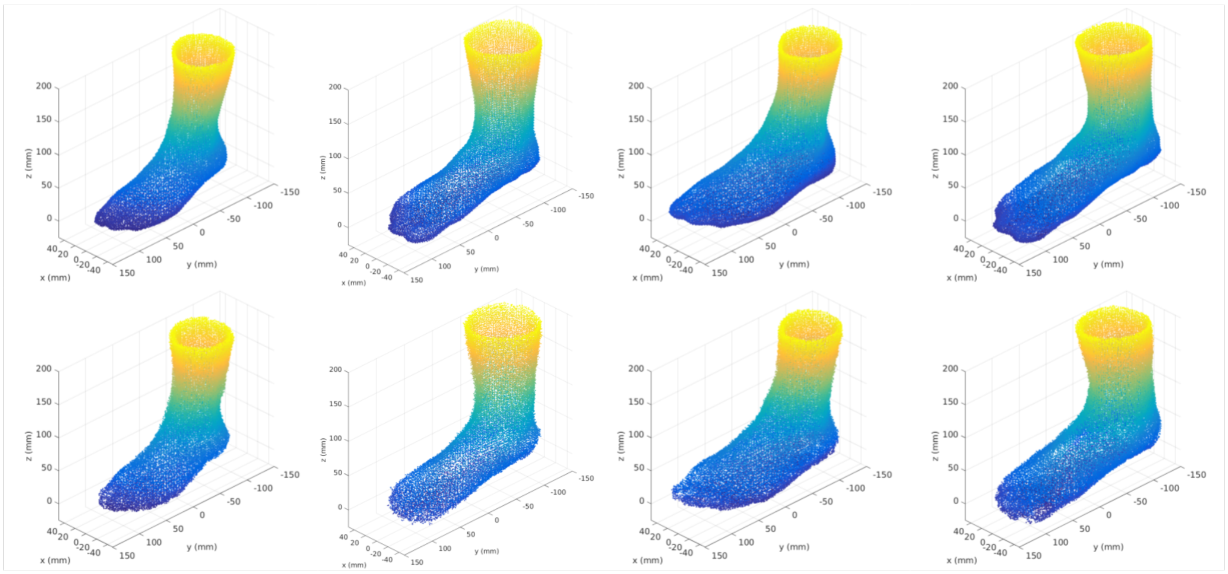


Figure 4.4: Sample point cloud synthesis results. First row: ground truth, Second row: synthesized point cloud.

Looking at the synthesized point clouds, we observe that their overall shape and scale match that of the ground truth very closely. Some regions such as the toes, appear to be less densely populated with points than the ground truth. We found our method to be less accurate here, and thus many of those points are filtered out in post-processing.

Looking more closely at Figure 4.5, it can be seen that point clouds generated when given an input view of the bottom of the foot were generally more accurate than when given a view of the top of the foot. This suggests that the most important information about foot shape is contained on the bottom surface of the foot rather than the top. Interestingly however, the most accurate point clouds are formed when given a profile view of the foot,

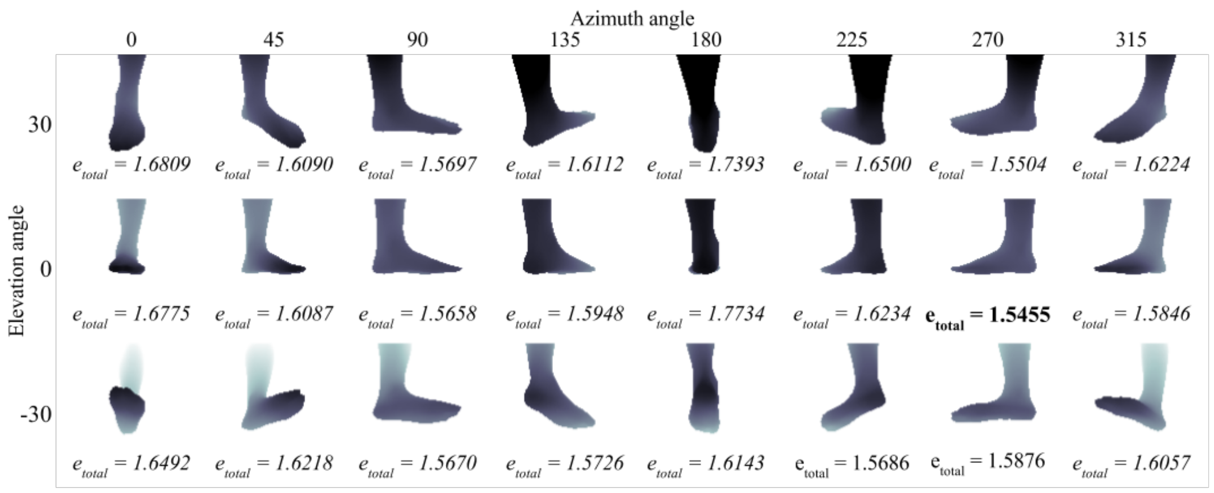


Figure 4.5: The 24 input depth map views explored and their corresponding synthesized point cloud error on the test set.

with 0 elevation angle. This result suggests that when building a scanner with a single camera, the camera should be mounted at this profile view position. This finding is also inline with those from Luximon et al. [51], who found that having information from the foot profile was important for overall shape reconstructions.

4.4 Impacts of Network Architectures

We evaluate our updated network architectures (as discussed in Section 3.4) using the depth map and point cloud error metrics. Samples of depth maps from each architecture are shown in Figure 4.6 along with their distribution of error across the image. Samples of synthesized point clouds from each of the three networks are shown in Figure 4.7. Our results are shown in Table 4.1. As can be seen, Net2 and Net3 both significantly outperform Net1 in both metrics, while requiring less training time. The use of residual blocks with batch normalization allows for higher learning rates enabling this reduction in training time. The lack of fully connected layers allows for smaller models with less parameters, while the residual block skip connection allows for deeper networks, with more processing at each representation size. In both cases of Net2 and Net3, the improved networks were able to become more precise in their point cloud accuracies than the 3.33 mm increment from the European continental sizing system [31].

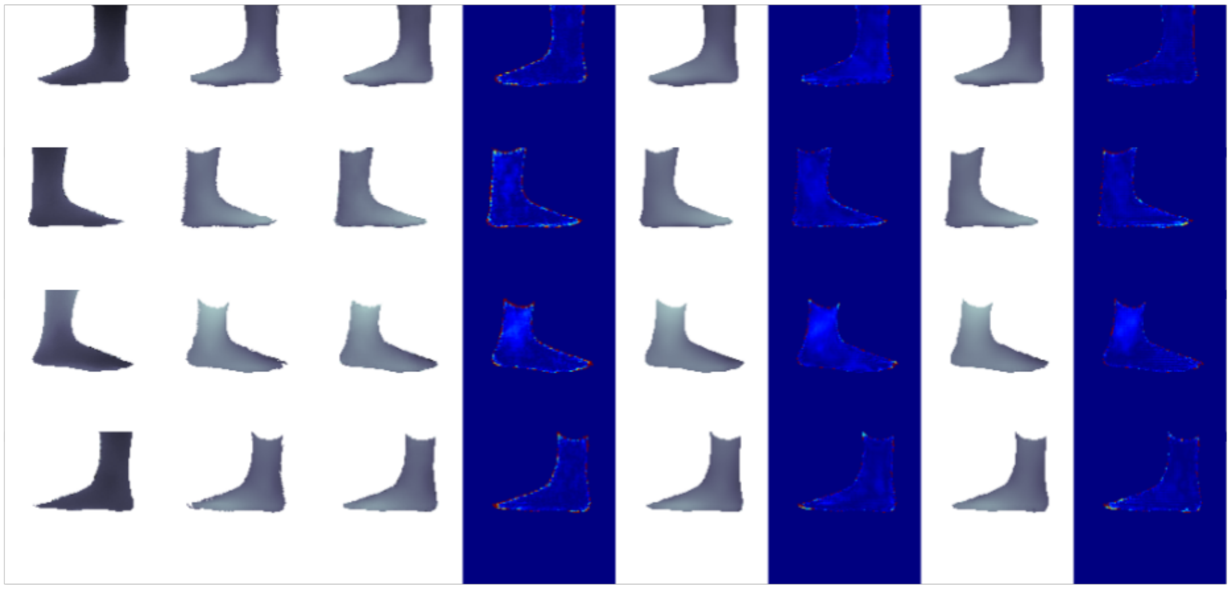


Figure 4.6: Depth map samples. Column 1: input depth map, Column 2: ground truth, Column 3: Net1 output depth map, Column 4: Net1 depth map error, Column 5: Net2 output depth map, Column 6: Net2 depth map error, Column 7: Net3 output depth map, Column 8: Net3 depth map error.

Net2 and Net3 preformed very similar with Net2 being the best overall, however it has a slightly wider standard deviation. Net3 contained the least number of parameters and could be trained in the least amount of time. Looking at the depth map errors in Figure 4.6, it can be seen that Net2 and Net3 produce less errors along the outlines of the foot, where Net1 struggles. Looking at the point clouds in Figure 4.7, Net2 and Net3 produce points that appear less noisy than Net1. The point clouds of Net2 appear to cover slightly more of the foot surface in sparse regions such as the toe, however it also tends to have more outliers in some cases, as seen on the heel in row 3. Overall our updated network architectures were able to improve performance in our point cloud completion method of single view foot scanning.

Table 4.1: Network architecture results.

Metric	Net1	Net2	Net3
Depth map error	0.0054	0.0047	0.0049
PC Error (mm)	2.92±0.72	2.55±0.75	2.58±0.71
Training Iterations	1,300,000	390,000	330,000

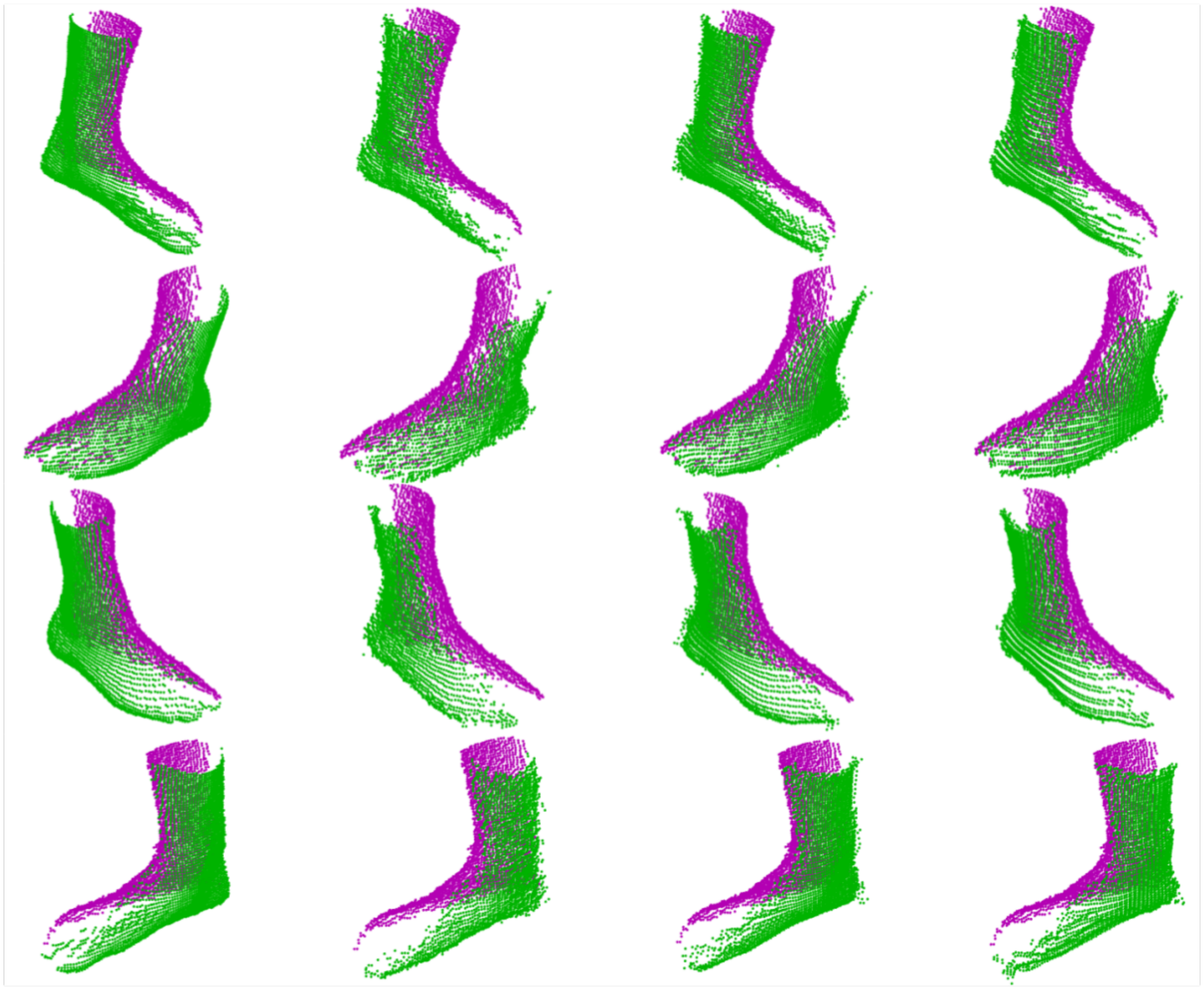


Figure 4.7: Point cloud samples. Column 1: ground truth, Column 2: Net1 point cloud, Column 3: Net2 point cloud, Column 4: Net3 point cloud. Purple: points from the input depth map, Green: points from the ground truth/synthesized output depth map.

4.5 Real World Data Results

4.5.1 Point Cloud Completion on Real World Data

Examples of the depth maps generated using our point cloud completion method on our real world scans are shown in Figure 4.8. We use our original network architecture described in Section 3.2. We found that our model generates depth maps that qualitatively look fairly accurate, however miss some finer features. The most obvious missing feature is the high arch, which makes for a sharper heel (most noticeable in column 9). Finer features such as these were likely not present in the MPII Human Shape data. The original CAESAR scans in the standing posture were taken from a weight bearing pose, where as we captured from a floating, non-weight bearing pose such that we could capture the plantar surface with the Kinect. The error distribution in Figure 4.8 shows that the synthesized views struggle to produce accurate toe regions as well. We suspect the cause of this is similar to that previously discussed, as the toe and heel shape as well as the pose can differ significantly based on posture. Overall based on 64 random input views, our model achieves an average depth map error of 0.01394, which is lower than on our original test set, however not so far off as to be a hard failure.

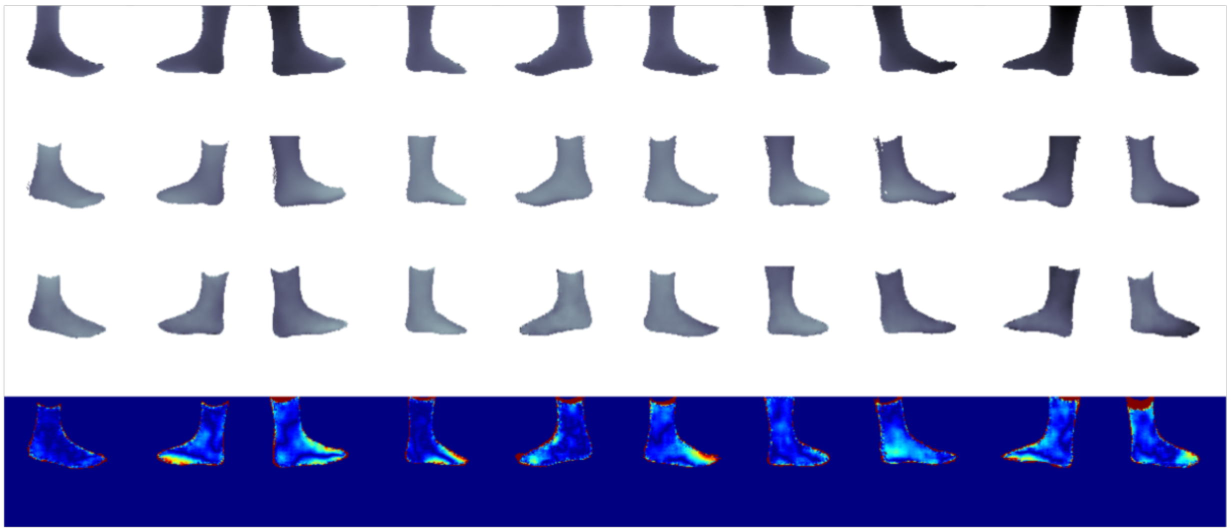


Figure 4.8: Depth map result samples using point cloud completion on real world scans. First row: input depth map, Second row: ground truth, Third row: synthesized output depth map, Fourth row: output depth map error distribution.

Point clouds examples are shown in Figure 4.9. With the full 3D point cloud, we can more clearly see where the network struggles. The lack of a sharp heel can clearly be seen here, as well as the misshapen toes. Interestingly, the overall size and placement of the synthesized depth map resembles the ground truth to a reasonable extent. This suggests that the network somewhat understands the input, and thus may be able to preform better if the input data could be matched more closely to the training data poses. This also suggests that this deep model could likely preform well with fine tuning on real data, as the current model shows signs of being capable in this domain. We find that our point clouds typically achieve an accuracy of around 8.53 mm, however in some of our tests we were able achieve accuracies of 6.48 mm.

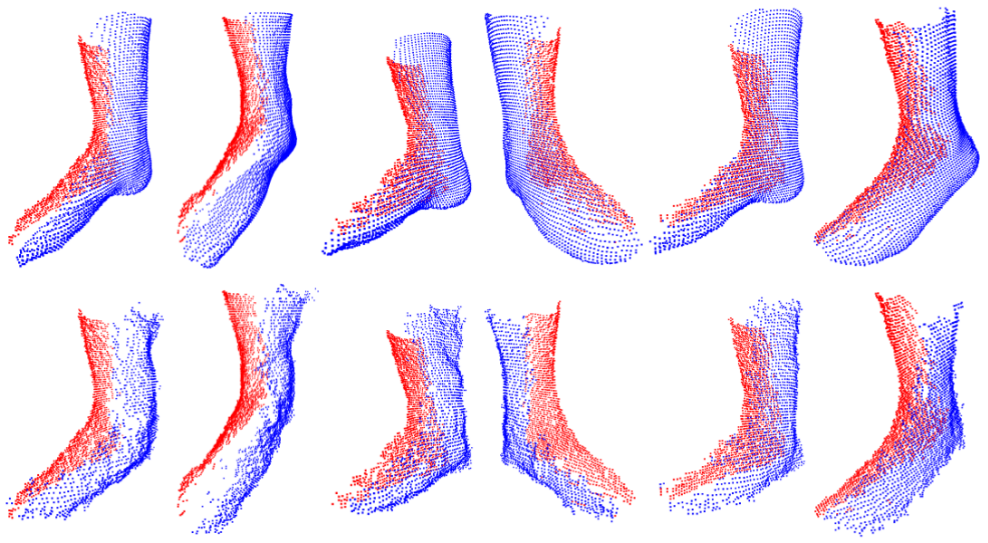


Figure 4.9: Point cloud result samples using point cloud completion on real world scans. First row: ground truth, Second row: synthesized point cloud. Red: input depth map points, Blue: ground truth/synthesized output depth map points.

4.5.2 Point Cloud Synthesis on Real World Data

Examples of the depth maps generated using our point cloud synthesis method on our real world scans are shown in Figure 4.10. Using this method, we observe a similar pattern to our point cloud completion method. Qualitatively the depth maps appear to be relatively accurate, minus certain features. Inconsistencies seem to be the sharp heel (noticeable in column 2), and a flatter foot profile (noticeable in column 1 and 7). Looking at the error distributions, we can see that the toes again seem to cause a lot of error. We also see more error around the outline of the foot, in this case it seems to be caused by the inconsistencies in foot thickness, likely caused by the weight bearing vs non-weight bearing issue discussed previously. Overall based on the reconstruction views, our model achieves an average depth map error of 0.03589, this is lower than on our original test set, but again, not necessarily a hard failure.

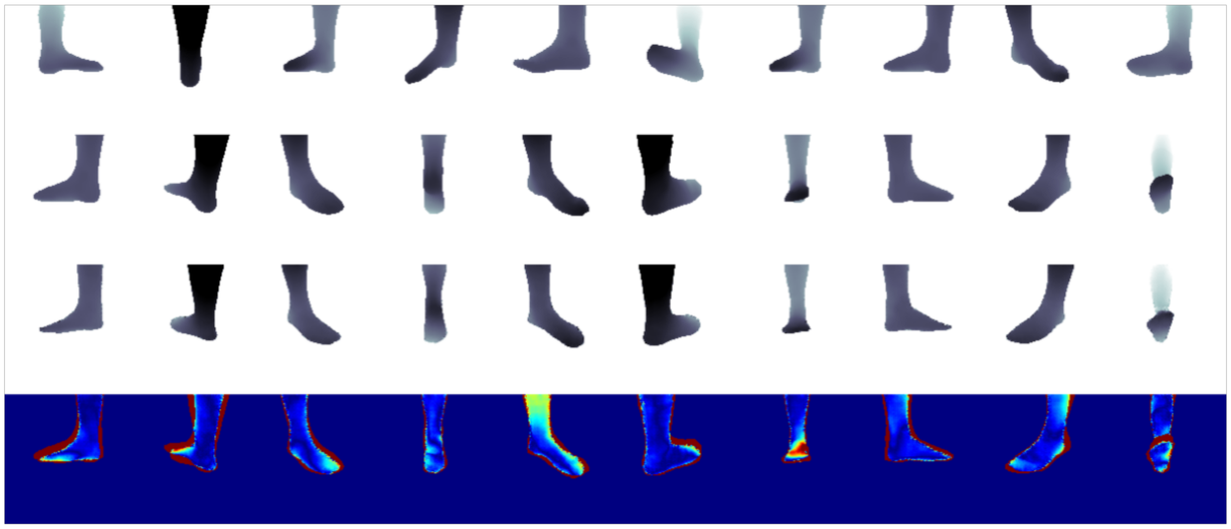


Figure 4.10: Depth map result samples using point cloud synthesis on real world scans. First row: input depth map, Second row: ground truth, Third row: synthesized output depth map, Fourth row: output depth map error distribution.

Fully synthesized point cloud examples are shown in Figure 4.11. As can be seen, the overall size of the synthesized point clouds are similar to that of the ground truth scan, however many of the finer features differ. We find that our point clouds achieve an accuracy of 4.73 mm, with some instances achieving 3.58 mm.

4.5.3 Real World Data Discussion

Using both of our methods, we were able to produce reasonable depth maps and complete point clouds. We have reported depth map and point cloud accuracy numbers for both

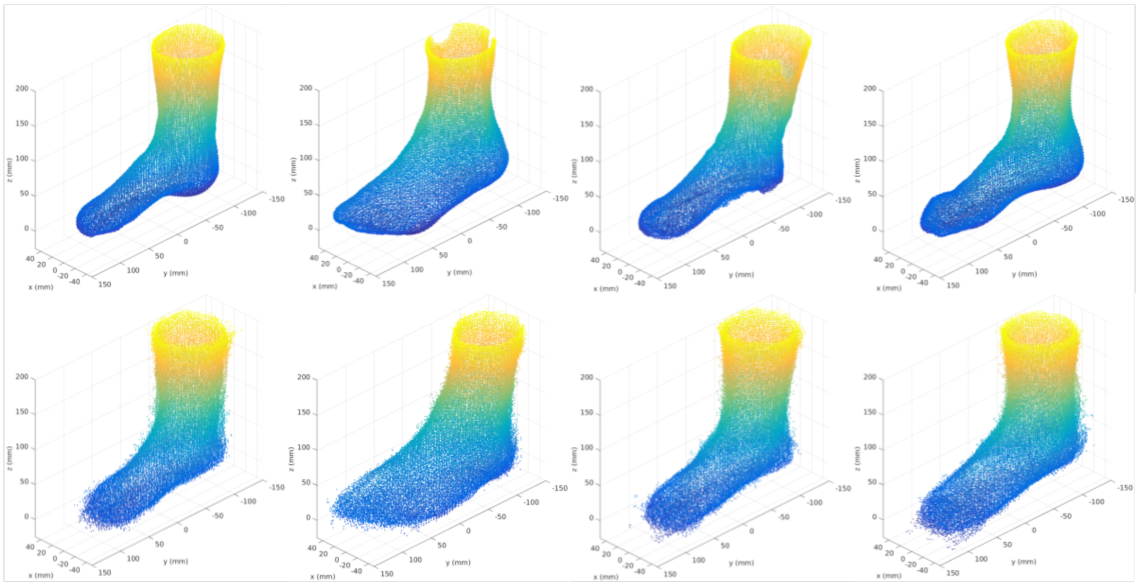


Figure 4.11: Point cloud result samples using point cloud synthesis on real world scans. First row: ground truth, Second row: synthesized point cloud.

numbers, however these should only be taken as very rough estimates. As we have discussed, our networks seem to be sensitive to foot pose, which affects their ability to produce accurate point clouds, however the pose also impacts the calculation of the metrics directly. Neither our depth map or point cloud metrics are particularly invariant to pose or position. In our MPII Human Shape testing, pose and position were more predictable, so they did not cause to much inflation of error in our metrics. In the real world data however, since pose impacts rigid shape and since foot objects were isolated and aligned by hand, our metrics are more easily inflated. For these reasons, we suspect that our reported errors on real world data represent an upper bound rather than the true shape error.

Overall we found that our point cloud completion method was able to produce more accurate depth maps than our point cloud synthesis method, but less accurate point cloud reconstructions. This result is inline with the results found when tested on MPII Human Shape data. The depth maps from our point cloud completion method qualitatively appear to be very similar to the ground truth. The network correctly determines which points belong on the foot surface in most cases, resulting in an accurate foreground shape (Figure 4.8). We do not see the same level of success in the depth maps from our point cloud synthesis method, where the foreground shape is more commonly incorrect (Figure 4.10).

We suspect this to be the primary reason why the depth map error for the point cloud synthesis method was found to be so high.

Looking at the point clouds produced, we see that our point cloud completion method produces recognizable feet, but not necessarily accurate ones (Figure 4.9). The network seems to be uncertain about the precise shape, as can be seen by the noisy surface, and seems to resort to producing surfaces that are smaller than they should be (most noticeable in column 4). Interestingly, when we look at the point clouds from our point cloud synthesis method (Figure 4.11), they are far more accurate. We found that these point clouds don't necessarily follow the correct foot pose and omit finer features, but are typically of roughly the correct size. The network in this method seems to have learned a more coherent representation of foot shape, allowing it to perform more strongly under new conditions.

In both of our methods, reasonable depth maps and point clouds were produced, considering the new data was quite different than that in training. These results suggest that our methods will be capable of performing well in real world conditions, if sufficient data from that domain can be acquired. Ideally, a few thousand real world scans could be used to train our methods from scratch, something that may be achievable with the raw CAESAR data. If this is not possible however, our results with MPII Human Shape suggest that our current models could be fine tuned using a much smaller set of real world scans, to adapt them to that domain. We suspect that using these methods, the accuracy levels we demonstrated testing MPII Human Shape feet should be achievable when applied to real world scanning.

4.6 Summary of Results

We have shown that both of our methods are viable solutions to 3D foot scanning from a single input depth map. Our point cloud completion method was shown to be able to produce near complete point clouds with an accuracy as high as 2.55 mm. Our point cloud synthesis method was found to be even more powerful, being able to produce complete point clouds with an accuracy of 1.55 mm. In either case, our results were demonstrated to be more precise than half sizes in the English American shoe sizing systems (4.23 mm), as well as half sizes in the European continental sizing system (3.33 mm) [31]. When applied to real world scans, we found that both our methods were able to recognize and produce reasonable depth maps and point clouds. Here we again found our point cloud synthesis method to be more accurate, and seems to be able to produce more coherient foot shapes under unfamiliar conditions. Our results suggest that are methods should perform strongly in the real world if trained or fine tuned on such data.

Overall we found that our point cloud synthesis method performed better than point cloud completion. We suspect this has mostly to do with its use of points from multiple synthesized depth maps. With more points, there is more redundancy in predictions of where a surface should be. This redundancy allows for inaccurate points produced from one view, to be made up for by more accurate point predictions from another. The overall point cloud is also more dense with points, which allows us to more aggressively filter away noisy points without leaving behind wholes and bare patches in the point cloud.

Chapter 5

Conclusions

We have presented two novel methods for leveraging deep learning to perform 3D foot scanning from a single depth map input view. Our networks were successfully able to learned the 3D shape of an anthropomorphic body part from incomplete information. These models were shown to be able to determine the missing information required to accurately generate a near complete or totally complete point cloud representation of a foot from only a single depth map.

Our point cloud completion method was found to have a reconstruction accuracy of 2.92±0.72 mm, which is more precise than the 4.23 mm half size increment used in the English and American shoe sizing systems, however it may not be accurate to the 3.33 mm half size in the European continental sizing system [31]. Upon updating our network to use residual blocks however, we were able to achieve accuracies of 2.55±0.75 mm, which comes in just below the continental half sizes. Our point cloud synthesis method was able to reconstruct full point clouds with an accuracy of 1.55±0.41 mm, which is significantly smaller than the English and American shoe sizing half size increment, as well as the continental half size.

Our models trained on MPII Human Shape data suggest the capabilities of deep learning for single view 3D foot scanning but also show that using this dataset alone may not be enough to fully build such systems. Subtle differences in the posture and shape of real world scan data seems to be enough to confuse the deep networks in our methods. This was shown to result in reconstructions with reduced accuracy. Our experiments with real world scans show that our models can understand real data, but for them to perform properly in a real setting requires that they are fully trained or at least fine tuned on real world data.

Overall, we have shown that our methods allow for foot scanners to avoid many of the complications associated with multi-camera and moving-camera systems. This includes complexities associated with extrinsic calibration, point cloud alignment, slow operation, mechanical complexity and high cost. Our point cloud completion method in particular, is simple, even compared with other forms of deep learning view synthesis for object reconstruction. We focused on how our network can supplement the existing data from the input depth map. By taking advantage of the general shape of a foot, and found that only a single additional view would be sufficient to reconstruct overall shape, requiring only a single forward pass of our network. Additionally, due to how we synthesize the additional view's depth map pixels from the same camera pose as the input, no additional steps to align the synthesized view with the original input are required to reconstruct the foot.

Due to both of our method's ability to require only a single input view, these can be applied to scanners that are significantly cheaper and faster at capturing shape. The simplicity of our scanning method could make scanners far more accessible than current products on the market, and allow for more widespread use of scanning for footwear matching. These methods also have potential in applications such as analysis of foot dynamics and loading from video, which can be useful in determining fit during motion and in footwear design.

Despite our promising results, our methods have some limitations over more traditional RGBD scanning methods. The point clouds produced by these methods are not as accurate as a scan using multiple viewpoints would be, as those would contain the true information about the object's shape from all views. Our methods are also limited to only scanning new instances of objects that are mostly similar to those seen during training. This method will fail if for example someone for whatever reason has a particularly unique foot shape, or if there is any sort of abnormality on a part of the foot not seen by the camera's single viewpoint. Specific to our point cloud completion method, we take advantage of foot shape to select the two depth maps views used. For more complex objects with more self occlusions, a different implementation would be required to capture the whole surface. For these reasons, our methods may not be practical to completely replace more traditional scanning method in all cases, however in our application, for most feet it is sufficient to capture an accurate representation.

Further work is needed to adapt our methods to working in the real world. The most important aspect to this would be to collect real world data to train with. We suggest acquiring the true CAESAR database scans, as well as scans using the specific camera hardware intended to be used in the final system. MPII Human Shape data and CAESAR data can be used to pre-train deep models, which should then be fine tuned on sensor specific data.

Additional future works and ways to expand upon this thesis work include further investigations in network architecture as well as additional methods of pre-processing and post-processing the data. Expanding this work to explore how color cameras could be used in single view scanning could prove useful as well, as they are significantly cheaper, more readily available and often have higher resolutions than RGBD depth maps.

References

[1] Precision 3D. Fotoscan 3d foot scanner. http://www.precision3d.co.uk/fs.htm. Accessed: 2017-09-28.

[2] Martín Abadi, Ashish Agarwal, Paul Barham, Eugene Brevdo, Zhifeng Chen, Craig Citro, Greg S. Corrado, Andy Davis, Jeffrey Dean, Matthieu Devin, Sanjay Ghemawat, Ian Goodfellow, Andrew Harp, Geoffrey Irving, Michael Isard, Yangqing Jia, Rafal Jozefowicz, Lukasz Kaiser, Manjunath Kudlur, Josh Levenberg, Dan Mané, Rajat Monga, Sherry Moore, Derek Murray, Chris Olah, Mike Schuster, Jonathon Shlens, Benoit Steiner, Ilya Sutskever, Kunal Talwar, Paul Tucker, Vincent Vanhoucke, Vijay Vasudevan, Fernanda Viégas, Oriol Vinyals, Pete Warden, Martin Wattenberg, Martin Wicke, Yuan Yu, and Xiaoqiang Zheng. TensorFlow: Large-scale machine learning on heterogeneous systems, 2015. Software available from tensorflow.org.

[3] Abdul Hadi Abdul Razak, Aladin Zayegh, Rezaul K Begg, and Yufridin Wahab. Foot plantar pressure measurement system: A review. *Sensors*, 12(7):9884–9912, 2012.

[4] Faraj Alhwarin, Alexander Ferrein, and Ingrid Scholl. Ir stereo kinect: improving depth images by combining structured light with ir stereo. In *Pacific Rim International Conference on Artificial Intelligence*, pages 409–421. Springer, 2014.

[5] Edmée Amstutz, Tomoaki Teshima, Makoto Kimura, Masaaki Mochimaru, and Hideo Saito. Pca based 3d shape reconstruction of human foot using multiple viewpoint cameras. In *International Conference on Computer Vision Systems*, pages 161–170. Springer, 2008.

[6] Amir Arsalan Soltani, Haibin Huang, Jiajun Wu, Tejas D. Kulkarni, and Joshua B. Tenenbaum. Synthesizing 3d shapes via modeling multi-view depth maps and silhouettes with deep generative networks. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, July 2017.

[7] Jon Louis Bentley. Multidimensional binary search trees used for associative searching. *Communications of the ACM*, 18(9):509–517, 1975.

[8] Paul J Besl and Neil D McKay. A method for registration of 3-d shapes. *IEEE Transactions on pattern analysis and machine intelligence*, 14(2):239–256, 1992.

[9] Federica Bogo, Javier Romero, Matthew Loper, and Michael J Black. Faust: Dataset and evaluation for 3d mesh registration. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 3794–3801, 2014.

[10] Leo Breiman. Random forests. *Machine learning*, 45(1):5–32, 2001.

[11] Yang-Sheng Chen, Yu-Chun Chen, Peng-Yuan Kao, Sheng-Wen Shih, and Yi-Ping Hung. Estimation of 3-d foot parameters using hand-held rgb-d camera. In *Asian Conference on Computer Vision*, pages 407–418. Springer, 2014.

[12] Yin Chen, Gang Dang, Zhi-Quan Cheng, and Kai Xu. Fast capture of personalized avatar using two kinects. *Journal of Manufacturing Systems*, 33(1):233–240, 2014.

[13] Gyeongmin Choe, Jaesik Park, Yu-Wing Tai, and In So Kweon. Refining geometry from depth sensors using ir shading images. *International Journal of Computer Vision*, pages 1–16, 2016.

[14] Christopher B Choy, Danfei Xu, JunYoung Gwak, Kevin Chen, and Silvio Savarese. 3d-r2n2: A unified approach for single and multi-view 3d object reconstruction. In *European Conference on Computer Vision*, pages 628–644. Springer, 2016.

[15] The Brannock Device Company. Genuine barnnock device. https://brannock.com/pages/about-us. Accessed: 2017-09-27.

[16] Intel Corporation. Intel realsense technology: Observe the world in 3d. https://www.intel.ca/content/www/ca/en/architecture-and-technology/realsense-overview.html. Accessed: 2017-09-29.

[17] Vorum Research Corporation. Scangogh 3d prosthetics and orthotics scanner. http://vorum.com/cad-cam-prosthetic-orthotic/scangogh-3d-scanner/. Accessed: 2017-09-28.

[18] Vorum Research Corporation. Yeti 3d scanner. http://vorum.com/footwear/yeti-3d-foot-scanner/. Accessed: 2017-09-27.

[19] Corpus.e. Lightbeam profile and convenience you customer. http://www.corpus-e.com/en/home/products/lightbeam.html. Accessed: 2017-09-28.

[20] Corinna Cortes and Vladimir Vapnik. Support-vector networks. *Machine learning*, 20(3):273–297, 1995.

[21] George E Dahl, Dong Yu, Li Deng, and Alex Acero. Context-dependent pre-trained deep neural networks for large-vocabulary speech recognition. *IEEE Transactions on audio, speech, and language processing*, 20(1):30–42, 2012.

[22] Angela Dai, Charles Ruizhongtai Qi, and Matthias Nießner. Shape completion using 3d-encoder-predictor cnns and shape synthesis. *arXiv preprint arXiv:1612.00101*, 2016.

[23] Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. Imagenet: A large-scale hierarchical image database. In *Computer Vision and Pattern Recognition, 2009. CVPR 2009. IEEE Conference on*, pages 248–255. IEEE, 2009.

[24] Endri Dibra, Himanshu Jain, Cengiz Öztireli, Remo Ziegler, and Markus Gross. Hs-nets: Estimating human body shape from silhouettes with convolutional neural networks. In *3D Vision (3DV), 2016 Fourth International Conference on*, pages 108–117. IEEE, 2016.

[25] Endri Dibra, Cengiz Öztireli, Remo Ziegler, and Markus Gross. Shape from selfies: Human body shape estimation using cca regression forests. In *European Conference on Computer Vision*, pages 88–104. Springer, 2016.

[26] Vincent G Duffy. *Digital human modeling*. Springer, 2007.

[27] Carol Frey, Francesca Thompson, Judith Smith, Melanie Sanders, and Helen Horstman. American orthopaedic foot and ankle society women's shoe survey. *Foot & ankle*, 14(2):78–81, 1993.

[28] S. Garrido-Jurado, R. Mu noz Salinas, F.J. Madrid-Cuevas, and M.J. Marín-Jiménez. Automatic generation and detection of highly reliable fiducial markers under occlusion. *Pattern Recognition*, 47(6):2280 – 2292, 2014.

[29] S. Garrido-Jurado, R. Mu noz Salinas, F.J. Madrid-Cuevas, and R. Medina-Carnicer. Generation of fiducial marker dictionaries using mixed integer linear programming. *Pattern Recognition*, 51:481 – 491, 2016.

[30] Ian Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. Generative adversarial nets. In Z. Ghahramani, M. Welling, C. Cortes, N. D. Lawrence, and K. Q. Weinberger, editors, *Advances in Neural Information Processing Systems 27*, pages 2672–2680. Curran Associates, Inc., 2014.

[31] Ravindra S Goonetilleke. *The science of footwear.* CRC Press, 2012.

[32] Mohammadul Haque, Avishek Chatterjee, Venu Madhav Govindu, et al. High quality photometric reconstruction using a depth camera. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 2275–2282, 2014.

[33] Michael R Hawes and Daniela Sovak. Quantitative morphology of the human foot in a north american population. *Ergonomics*, 37(7):1213–1226, 1994.

[34] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 770–778, 2016.

[35] Elizabeth Holmes. Feet are getting bigger, and many people wear shoes that don't fit right. *The Wall Street Journal*, 2014.

[36] Kurt Hornik, Maxwell Stinchcombe, and Halbert White. Multilayer feedforward networks are universal approximators. *Neural networks*, 2(5):359–366, 1989.

[37] Bayer Inc. Dr. scholls custom fit orthotic inserts. https://www.drscholls.com/products/pain-relief/custom-fit-orthotic-inserts/. Accessed: 2017-09-28.

[38] The MathWorks Inc. Convolutional neural network. https://www.mathworks.com/discovery/convolutional-neural-network.html. Accessed: 2017-10-14.

[39] Sergey Ioffe and Christian Szegedy. Batch normalization: Accelerating deep network training by reducing internal covariate shift. In *International Conference on Machine Learning*, pages 448–456, 2015.

[40] Achuta Kadambi, Vage Taamazyan, Boxin Shi, and Ramesh Raskar. Polarized 3d: High-quality depth sensing with polarization cues. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 3370–3378, 2015.

[41] Diederik P. Kingma and Jimmy Lei Ba. Adam: a Method for Stochastic Optimization. In *International Conference on Learning Representations 2015*, pages 1–15, 2015.

[42] I Krauss, S Grau, M Mauch, C Maiwald, and T Horstmann. Sex-related differences in foot shape. *Ergonomics*, 51(11):1693–1709, 2008.

[43] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. ImageNet Classification with Deep Convolutional Neural Networks. *Advances In Neural Information Processing Systems*, pages 1–9, 2012.

[44] Yann LeCun, Yoshua Bengio, and Geoffrey Hinton. Deep learning. *Nature*, 521(7553):436–444, 2015.

[45] Yann LeCun, Léon Bottou, Yoshua Bengio, and Patrick Haffner. Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, 86(11):2278–2324, 1998.

[46] Shuai Lin, Yin Chen, Yu-Kun Lai, Ralph R Martin, and Zhi-Quan Cheng. Fast capture of textured full-body avatar with rgb-d cameras. *The Visual Computer*, 32(6-8):681–691, 2016.

[47] Nolan Lunscher and John Zelek. Deep learning anthropomorphic 3d point clouds from a single depth map camera viewpoint. In *The IEEE International Conference on Computer Vision (ICCV) Workshops*, 2017.

[48] Nolan Lunscher and John Zelek. Foot depth map point cloud completion using deep learning with residual blocks. In *Journal of Computational Vision and Imaging Systems*, 2017.

[49] Nolan Lunscher and John Zelek. Point cloud completion of foot shape from a single depth map for fit matching using deep learning view synthesis. In *The IEEE International Conference on Computer Vision (ICCV) Workshops*, 2017.

[50] Ameersing Luximon and Ravindra S Goonetilleke. Foot shape modeling. *Human Factors*, 46(2):304–315, 2004.

[51] Ameersing Luximon, Ravindra S Goonetilleke, and Ming Zhang. 3d foot shape generation from 2d information. *Ergonomics*, 48(6):625–641, 2005.

[52] M Mauch, S Grau, I Krauss, C Maiwald, and T Horstmann. Foot morphology of normal, underweight and overweight children. *International journal of obesity*, 32(7):1068, 2008.

[53] Thomas Mensink, Jakob Verbeek, Florent Perronnin, and Gabriela Csurka. Metric learning for large scale image classification: Generalizing to new classes at near-zero cost. *Computer Vision–ECCV 2012*, pages 488–501, 2012.

[54] Microsoft. Meet kinect for windows. https://developer.microsoft.com/en-us/windows/kinect. Accessed: 2017-10-15.

[55] Abdel-rahman Mohamed, George E Dahl, and Geoffrey Hinton. Acoustic modeling using deep belief networks. *IEEE Transactions on Audio, Speech, and Language Processing*, 20(1):14–22, 2012.

[56] Theo Moons, Luc Van Gool, Maarten Vergauwen, et al. 3d reconstruction from multiple images part 1: principles. *Foundations and Trends® in Computer Graphics and Vision*, 4(4):287–404, 2010.

[57] FIT Muscle and Joint Clinic. Custom foot orthotics. https://www.fitmjc.com/chiropractic-services/custom-foot-orthotics/. Accessed: 2017-09-28.

[58] Richard A Newcombe, Shahram Izadi, Otmar Hilliges, David Molyneaux, David Kim, Andrew J Davison, Pushmeet Kohi, Jamie Shotton, Steve Hodges, and Andrew Fitzgibbon. Kinectfusion: Real-time dense surface mapping and tracking. In *Mixed and augmented reality (ISMAR), 2011 10th IEEE international symposium on*, pages 127–136. IEEE, 2011.

[59] Chuong V Nguyen, Shahram Izadi, and David Lovell. Modeling kinect sensor noise for improved 3d reconstruction and tracking. In *3D Imaging, Modeling, Processing, Visualization and Transmission (3DIMPVT), 2012 Second International Conference on*, pages 524–530. IEEE, 2012.

[60] Novel.de. The emed-systems. http://www.novel.de/novelcontent/emed. Accessed: 2017-09-28.

[61] Roy Or-El, Rom Hershkovitz, Aaron Wetzler, Guy Rosman, Alfred M Bruckstein, and Ron Kimmel. Real-time depth refinement for specular objects. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 4378–4386, 2016.

[62] Roy Or-El, Guy Rosman, Aaron Wetzler, Ron Kimmel, and Alfred M Bruckstein. Rgbd-fusion: Real-time high precision depth recovery. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 5407–5416, 2015.

[63] Eunbyung Park, Jimei Yang, Ersin Yumer, Duygu Ceylan, and Alexander C Berg. Transformation-grounded image generation network for novel 3d view synthesis. *arXiv preprint arXiv:1703.02921*, 2017.

[64] Leonid Pishchulin, Stefanie Wuhrer, Thomas Helten, Christian Theobalt, and Bernt Schiele. Building statistical shape spaces for 3d human modeling. *CoRR*, abs/1503.05860, 2015.

[65] Rajat Raina, Anand Madhavan, and Andrew Y Ng. Large-scale deep unsupervised learning using graphics processors. In *Proceedings of the 26th annual international conference on machine learning*, pages 873–880. ACM, 2009.

[66] Danilo Jimenez Rezende, SM Ali Eslami, Shakir Mohamed, Peter Battaglia, Max Jaderberg, and Nicolas Heess. Unsupervised learning of 3d structure from images. In *Advances in Neural Information Processing Systems*, pages 4996–5004, 2016.

[67] Kathleen M Robinette, Hans Daanen, and Eric Paquet. The caesar project: a 3-d surface anthropometry survey. In *3-D Digital Imaging and Modeling, 1999. Proceedings. Second International Conference on*, pages 380–386. IEEE, 1999.

[68] Olga Russakovsky, Jia Deng, Hao Su, Jonathan Krause, Sanjeev Satheesh, Sean Ma, Zhiheng Huang, Andrej Karpathy, Aditya Khosla, Michael Bernstein, Alexander C. Berg, and Li Fei-Fei. ImageNet Large Scale Visual Recognition Challenge. *International Journal of Computer Vision (IJCV)*, 115(3):211–252, 2015.

[69] Bonnie Yuk San Tsung, Ming Zhang, Yu Bo Fan, and David Alan Boone. Quantitative comparison of plantar foot shapes under different weight-bearing conditions. *Journal of rehabilitation research and development*, 40(6):517, 2003.

[70] Jorge Sánchez and Florent Perronnin. High-dimensional signature compression for large-scale image classification. In *Computer Vision and Pattern Recognition (CVPR), 2011 IEEE Conference on*, pages 1665–1672. IEEE, 2011.

[71] R. N. Shepard and J. Metzler. Mental rotation of three-dimensional objects. *Science*, 171(3972):701–703, 1971.

[72] Begg Shoes. Shoe lasts. https://www.beggshoes.com/blog/shoe-last/. Accessed: 2017-11-12.

[73] Harvey Simon and David Zieve. Foot pain in-depth report. *The New York Times*, 2013.

[74] Karen Simonyan and Andrew Zisserman. Very deep convolutional networks for large-scale image recognition. *arXiv preprint arXiv:1409.1556*, 2014.

[75] Shuran Song, Fisher Yu, Andy Zeng, Angel X. Chang, Manolis Savva, and Thomas Funkhouser. Semantic scene completion from a single depth image. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, July 2017.

[76] Maxim Tatarchenko, Alexey Dosovitskiy, and Thomas Brox. Multi-view 3d models from single images with a convolutional network. In *European Conference on Computer Vision*, pages 322–337. Springer, 2016.

[77] Carnegie Mellon University. Panda3d. https://www.panda3d.org/. Accessed: 2017-12-17.

[78] Volumental. Foot scanning. https://www.volumental.com/foot-scanning/. Accessed: 2017-09-27.

[79] DRGHR Williams and Geoffrey Hinton. Learning representations by back-propagating errors. *Nature*, 323(6088):533–538, 1986.

[80] Jiajun Wu, Yifan Wang, Tianfan Xue, Xingyuan Sun, William T Freeman, and Joshua B Tenenbaum. MarrNet: 3D Shape Reconstruction via 2.5D Sketches. In *Advances In Neural Information Processing Systems*, 2017.

[81] Zhirong Wu, Shuran Song, Aditya Khosla, Fisher Yu, Linguang Zhang, Xiaoou Tang, and Jianxiong Xiao. 3d shapenets: A deep representation for volumetric shapes. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 1912–1920, 2015.

[82] Stefanie Wuhrer, Chang Shu, and Pengcheng Xi. Posture-invariant statistical shape analysis using laplace operator. *Computers & Graphics*, 36(5):410–416, 2012.

[83] Xinchen Yan, Jimei Yang, Ersin Yumer, Yijie Guo, and Honglak Lee. Perspective transformer nets: Learning single-view 3d object reconstruction without 3d supervision. In *Advances in Neural Information Processing Systems*, pages 1696–1704, 2016.

[84] Bo Yang, Hongkai Wen, Sen Wang, Ronald Clark, Andrew Markham, and Niki Trigoni. 3d object reconstruction from a single depth view with adversarial learning. In *The IEEE International Conference on Computer Vision (ICCV)*, Oct 2017.

[85] Pietro Zanuttigh, Giulio Marin, Carlo Dal Mutto, Fabio Dominio, Ludovico Minto, and Guido Maria Cortelazzo. Operating principles of structured light depth cameras. In *Time-of-Flight and Structured Light Depth Cameras*, pages 43–79. Springer, 2016.

[86] Matthew D Zeiler, Dilip Krishnan, Graham W Taylor, and Rob Fergus. Deconvolutional networks. In *Computer Vision and Pattern Recognition (CVPR), 2010 IEEE Conference on*, pages 2528–2535. IEEE, 2010.

[87] Ziyu Zhang, Alexander G. Schwing, Sanja Fidler, and Raquel Urtasun. Monocular object instance segmentation and depth ordering with CNNs. *Proceedings of the IEEE International Conference on Computer Vision*, 11-18-December-2015:2614–2622, 2016.

[88] Bo Zhao, Xiao Wu, Zhi-Qi Cheng, Hao Liu, and Jiashi Feng. Multi-view image generation from a single-view. *arXiv preprint arXiv:1704.04886*, 2017.

[89] Tinghui Zhou, Shubham Tulsiani, Weilun Sun, Jitendra Malik, and Alexei A Efros. View synthesis by appearance flow. In *European Conference on Computer Vision*, pages 286–301. Springer, 2016.

[90] Alexander Zien, Gunnar Rätsch, Sebastian Mika, Bernhard Schölkopf, Thomas Lengauer, and K-R Müller. Engineering support vector machine kernels that recognize translation initiation sites. *Bioinformatics*, 16(9):799–807, 2000.

[91] Chuhang Zou, Ersin Yumer, Jimei Yang, Duygu Ceylan, and Derek Hoiem. 3d-prnn: Generating shape primitives with recurrent neural networks. In *The IEEE International Conference on Computer Vision (ICCV)*, Oct 2017.

[92] Silvia Zuffi and Michael J Black. The stitched puppet: A graphical model of 3d human shape and pose. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 3537–3546, 2015.