Supervised Approaches on Human Pose Estimation from 3D Point Clouds

handed in
MASTER'S THESIS

B. Eng. Alexander Schwer

born on the 06.06.1991
living in:
Felsennelkenanger 21
80937 Munich
Tel.: 08293/7429

Human-centered Assistive Robotics
Technical University of Munich

Univ.-Prof. Dr.-Ing. Dongheui Lee

Supervisor:	Prof. Dongheui Lee, Shile Li, Christoph Baur
Start:	12.12.2018
Intermediate Report:	01.07.2019
Delivery:	14.10.2019

TECHNISCHE UNIVERSITÄT MÜNCHEN
Human-centered Assistive Robotics
UNIV.-PROF. DR.-ING. DONGHEUI LEE

October 10, 2019

M A S T E R ' S T H E S I S
for
Alexander Schwer
Student ID 03679869, Degree EI

Supervised Approaches for Human Pose Estimation from 3D Point Clouds

Problem description:

Human Pose Estimation is of vital importance for tracking human subjects in public surveillance applications or for establishing human safety in robotic environments. A plethora of work has been devoted to human pose estimation from RGB images, depth sensors and a combination of both. Thereby, the methods can be primarily grouped into sparse skeleton-based [2][3] and lately also dense approaches [1], the latter trying to establish correspondences between pixels in 2D images and 3D points on a human surface model. In this project, the ultimate goal is to identify human pose from 3D point clouds rather than 2D data, and to establish a mapping of these points to a 3D canonical human surface model. Therefore, State-of-the-Art (SOTA) methods first have to be examined and potentially transfered to the domain of 3D data. Moreover, for dense correspondence estimation among 3D points, labeled training data is not readily available. Existing 2D↔3D correspondence methods [1] can potentially be leveraged to create such a ground-truth, which is also subject to investigation.

Tasks:

- Literature review on Human Pose Estimation and search for a suitable dataset
- Setup of the development environment and reproducing suitable state-of-the-art on RGB/RGB-D data
- Adapting and exploring the applicability of SOTA methods for 3D point clouds
- Wrap-Up

Bibliography:

[1] R. A. Güler, N. Neverova, and I. Kokkinos. Densepose: Dense human pose estimation in the wild. *arXiv preprint arXiv:1802.00434*, 2018.

[2] D. Mehta, S. Sridhar, O. Sotnychenko, H. Rhodin, M. Shafiei, H.-P. Seidel, W. Xu, D. Casas, and C. Theobalt. Vnect: Real-time 3d human pose estimation with a single rgb camera. *ACM Transactions on Graphics (TOG)*, 36(4):44, 2017.

[3] C. Zimmermann, T. Welschehold, C. Dornhege, W. Burgard, and T. Brox. 3d human pose estimation in rgbd images for robotic task learning. *arXiv preprint arXiv:1803.02622*, 2018.

Supervisor:	M.Sc. Shile Li
Start:	12.12.2018
Intermediate Report:	01.07.2019
Delivery:	14.10.2019

(D. Lee)
Univ.-Professor

Abstract

In this work, we present multiple approaches for establishing correspondences between the human body surface and a canonical representation of it via human pose estimation, formulated as body part segmentation task. To address the lack of a suitable and sufficiently large data set, showing point clouds of walking people without clothes, we propose a fully automatic data synthesis pipeline based on classical animation approaches and also provide a method for automatic segmentation label creation. Based on this pipeline we create a data set of 60 thousand samples showing 200 different synthetic people in 40 different walking scenarios. For the human pose estimation itself, we adapt multiple convolutional neural network architectures from the domain of segmentation working on textureless point clouds, as well as on volumetric representations of these point clouds. Furthermore, we evaluate our models on the synthesized data and provide quantitative results for all architectures as well as qualitative results in terms of visualizations of predictions of the networks. Thereby we show that the proposed architectures are capable of solving the human pose estimation task and discover that the proposed 3D CNN architectures are superior to the point-based approaches in terms of segmentation performance, but suffer from much higher inference time.

Zusammenfassung

In dieser Arbeit stellen wir mehrere Ansätze zur Ermittlung von Korrespondenzen zwischen der menschlichen Körperoberfläche und einer kanonischen Darstellung derselben mittels menschlicher Posenschätzung vor, die als Aufgabe der Segmentierung von Körperteilen formuliert ist. Um dem Mangel an geeigneten und ausreichend großen Datensätzen, die Punktwolken von gehenden Menschen ohne Kleidung zeigen, zu begegnen, schlagen wir eine vollautomatische Datensynthesepipeline vor, die auf klassischen Animationsansätzen basiert und auch ein Verfahren zur automatischen Segmentierungslabelerstellung bereitstellen. Basierend auf dieser Pipeline erstellen wir einen Datensatz von 60.000 Proben, der 200 verschiedene synthetische Menschen in 40 verschiedenen Gehszenarien zeigt. Für die menschliche Posenschätzung selbst adaptieren wir mehrere faltungsneuronale Netzwerkarchitekturen aus dem Bereich der Segmentierung, die auf texturlosen Punktwolken arbeiten, sowie auf volumetrischen Darstellungen dieser Punktwolken. Darüber hinaus bewerten wir unsere Modelle auf den synthetisierten Daten und liefern quantitative Ergebnisse für alle Architekturen sowie qualitative Ergebnisse in Form von Vorhersagen der Netzwerke. Dabei zeigen wir, dass die vorgeschlagenen Architekturen in der Lage sind, die Aufgabe der menschlichen Posenschätzung zu lösen und entdecken, dass die vorgeschlagenen 3D-CNN-Architekturen den punktbasierten Ansätzen in Bezug auf die Segmentierungsleistung überlegen sind, aber eine viel höheren Verarbeitungszeit benötigen.

Contents

Chapter 1

Introduction

While security regulations are tightened in many public places such as airports, stadiums and also on concerts and other events, security scanners gain in importance and attention. Due to the lack of the ability of metal scanners to detect non-metallic material per definition, other approaches based on microwave imaging technology are in the focus through their capability of detecting a large variety of other potentially dangerous materials. These scanners create an image of the human body surface by transmitting electromagnetic waves in the microwave range and by capturing their reflection. From the fact that waves of these wavelengths can't pass the human skin and hence be reflected, a detailed 3D radar image of the human body surface is created. A special type of these scanners, which are investigated in this thesis, use the multistatic radar principle, that distinguishes themselves from others by surrendering moving parts, hence the name static. These devices use a large number of antennas to scan a volumetric area.

For most applications, there is security personnel that has to operate the device. In order to protect the scanned people's privacy the microwave images aren't shown to the security personnel directly. Instead, a canonical avatar is shown, which creates the necessity for software that detects potential threats automatically in the image, as well as to locate the threat on the avatar. The latter is addressed by human pose estimation. It also has been shown that pose estimation as a feature for the detection software is beneficial to improve detection results. This further motivates the need for human pose estimation and leads to the problem statement in 1.1.

But human pose estimation is not a problem that is specific for body scanners, it is one of the great problems in Computer Vision and is as well not only applied to point clouds. It is of great practical value in paradigms such as human-robot interaction, surveillance or healthcare. Recent years have seen a vast amount of successful approaches towards human pose estimation from primarily RGB and RGB-D data (see section 3.1), thanks to great advancements in the field of Machine Learning. The main facilitator behind these approaches is Deep Learning, which allows

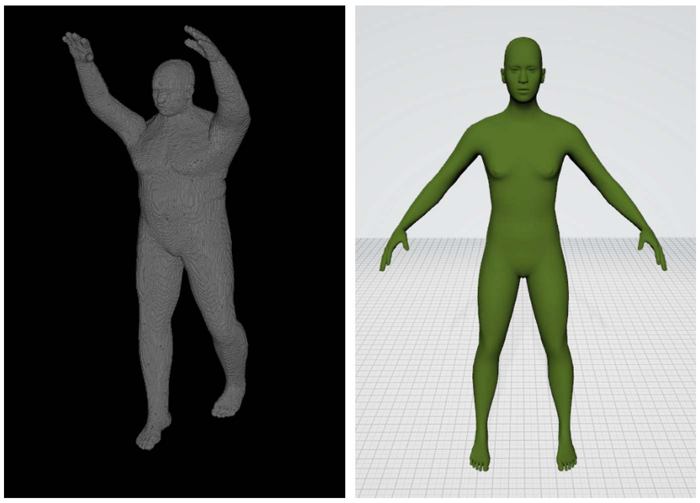

(a) Point Cloud (b) Avatar

Figure 1.1: Problem Statement Data

establishing highly complex mappings to a desired output via learning a powerful, non-linear feature extraction pipeline. Motivated by progresses that were made in Pose Estimation via Machine Learning on RGB or RGB-D and due to the little effort that was invested into working on pose estimation from point clouds, the following problem statement and research questions were formulated.

1.1 Problem Statement and Research Questions

Given a 3D point cloud with regular spacing, the goal is to estimate the correspondences between points in the point cloud and points on the surface model of the human body.
These correspondences define a mapping between a point in the point cloud and another one, located on an avatar.
Or mathematically formulated: The point cloud can be interpreted as a set $\mathbb{A} \subseteq \mathbb{R}^3$, which undergoes the map:

$$\phi : \mathbf{x} \rightarrow \phi\left(\mathbf{x}\right), \mathbf{x} \in \mathbb{A} \tag{1.1}$$

onto another set $\mathbb{B} \subseteq \mathbb{R}^3, \phi\left(\mathbf{x}\right) \in \mathbb{B}$. This makes this map a vector field assigning each point of the input set again a vector. Therefore the goal of the master thesis is to find an approach that estimates the map $\phi\left(\mathbf{x}\right)$ for a given point cloud.
Considering the huge amount of data in point clouds of such body scanner (67 Mio Voxels) and the therefore required immense effort for creating the ground truth, also an approach is considered where the map is not estimated directly. Instead, each

point is classified into a class corresponding to a body part or background. This can be viewed as a rough estimate of the correspondence. These body part segmentation can then also ultimately be used to estimate the map $\phi(x)$. Creating labels for body zones instead of a per voxel vector map eases up the labeling effort significantly.

This thesis aims to build CNN models coping with the body part segmentation task without using additional algorithms or processing steps slowing down the application's speed. 3D CNNs were successfully applied to perform a binary segmentation on medical data [18, 53], as well as point-based 2D CNNs were applied for other segmentation tasks [41, 67]. But it is the first attempt (to my knowledge) to utilize such networks for the formulated classification task. Therefore, the question that arises is: *Can 3D and 2D CNNs be successfully applied to the formulated classification task? How do they compare in terms of timing?*

1.2 Thesis Structure

The thesis is structured in 8 chapters including the introduction, where the problem statement was motivated:

- A quick introduction into the imaging system, pose estimation, machine learning, neural networks and CNNs is given in Chapter 2
- Chapter 3 reviews the related literature in pose estimation and point cloud segmentation
- Chapter 4 describes the novel dataset creation pipeline
- A detailed description of the proposed CNN architectures and their implementations are given in Chapter 5
- Chapter 6 and 7 explain completed experiments, provide technical details and discuss obtained results
- Finally, Chapter 8 concludes this thesis stating the goals achieved, limitations and future work

Chapter 2

Theoretical Background

2.1 Imaging System

Driven by the enormous advances in semiconductor technology, achieved in the years before 2012, a modern imaging system in the millimeter-wave (mm-wave) range[1] with a high number of channels was developed [1]. The imaging system works completely contact-less (in comp. to ultrasonic imaging) and also free of health concerns (in comp. to X-Ray Technology). It's capable of a spatial resolution up to 2mm in the lateral direction, therefore it is well suited for personal screening at airport security checkpoints. The resolution and dynamic range are the main performance measures for imaging devices, talking about both no comparable device is on the market so far. For such mm-wave imaging systems it is possible to work with passive or active imaging concepts. A passive concept would use the characteristic radiation of an object and the reflected background radiation without the need for illumination by electromagnetic energy. But for indoor applications this concept suffers from low radiometric contrast and is hence not applicable. This is why the imaging system was designed to use an active concept, i.e. the system is emitting electromagnetic waves and capturing their reflections.

2.1.1 Basic Imaging Principle

Looking a little bit closer at the basic principle of the imaging system, it can be derived from Maxwell's equations (appendix A.1) with certain assumptions about the imaged material and certain approximations. A quick overview of the derivation of the basic imaging equations according to [79] is given in the following.

Assuming material with negligible conductivity, one can assume the scattering re-

[1] 1-10mm (30-300 GHz)

gion[1] to be source-free. Further assumption of time-harmonic electrical and magnetic field, leads to the following equation, which can be derived from Maxwell's equation as in [79]:

$$\nabla^2 \mathbf{E} + \omega^{\mathbf{2}} \varepsilon(\mathbf{r}) \mu(\mathbf{r}) \mathbf{E} = -\left(\nabla \ln \mu(\mathbf{r})\right) \times \left(\nabla \times \mathbf{E}\right) + \nabla\left(\left(\nabla \ln \varepsilon(\mathbf{r})\right) \cdot \mathbf{E}\right), \tag{2.1}$$

where ∇ denotes the nabla operator, $\cdot$ the scalar product, $\times$ the cross product and ω the angular frequency. $\mathbf{E}$ denotes the electrical field, $\varepsilon(\mathbf{r})$ the permittivity and $\mu(\mathbf{r})$ the permeability, which in general all can depend on their position in space $\mathbf{r}$.

With further assumption of $\mu(\mathbf{r})$ being constant and equal to μ_0, i.e. the permittivity in vacuum and the term $\nabla \ln(\varepsilon(\mathbf{r}))$ being low compared to $\mathbf{E}$, one can completely remove the right hand side of (2.1), as a approximation, which leads to the following equation, which is called the homogeneous Helmholtz equation:

$$\nabla^2 \mathbf{E} + k_0^2 \varepsilon_r^c(\mathbf{r}) \mathbf{E} = 0, \tag{2.2}$$

where $k_0 = \omega\sqrt{\mu_0 \varepsilon_0}$ denotes the wave-number in free space and $\varepsilon_r^c(\mathbf{r})$ the complex relative permittivity. From that equation it can be observed that Cartesian field components of $\mathbf{E}$ are decoupled, which suggests to work just with a scalar field, that can stand for any Cartesian component in $\mathbf{E}$ and will be denoted as $U = U^s + U^i$, where U^s and U^i are the scattered wave and the incident wave respectively, resulting in:

$$\nabla^2 U + k_0^2 \varepsilon_r^c(\mathbf{r}) U = 0. \tag{2.3}$$

When incorporating that U^i is a solution to (2.3), where $\varepsilon_r^c(\mathbf{r}) = 1$, then (2.3) can be rewritten to:

$$\nabla^2 U^s + k_0^2 U^s = -O(\mathbf{r}) U, \tag{2.4}$$

where $O(\mathbf{r}) = k_0(\varepsilon_r^c - 1)$, which is called the scattering potential and describes spatial variations in permittivity, and thus is the target function one wants to capture to create an image.
When the right hand side is replaced by a negative dirac delta $-\delta(\mathbf{r})$, the solution for U^s in (2.4) is given by the green function $G(\mathbf{r}) = \frac{\exp(-jk_0\|\mathbf{r}\|)}{4\pi\|\mathbf{r}\|}$, and therefore , since (2.4) is linear, the solution for U^s in general for (2.3) is given by the convolution between $G(\mathbf{r})$ and $O(\mathbf{r}) U$:

$$U^s(\mathbf{r}) = \left(O(\mathbf{r}) U(\mathbf{r})\right) * G(\mathbf{r}) \approx \left(O(\mathbf{r}) U^i(\mathbf{r})\right) * G(\mathbf{r}), \tag{2.5}$$

where the right part of (2.5) is achieved by applying the Born approximation to get a solution for U^s.

[1]region where the electromagnetic wave is reflected

To solve for the desired quantity $O\left(\mathbf{r}\right)$ one has to invert (2.5), this can be done be deconvolution or Fourier transform.
An estimate of $O\left(\mathbf{r}\right)$ can be achieved through spatial sampling of U^s with the help of an antenna array, which is given by:

$$\hat{O}\left(\mathbf{r}\right) = \sum_k \sum_{n_t} \sum_{n_r} U^s\left(\mathbf{r_t}, \mathbf{r_r}, k\right) e^{-jk\left(\|\mathbf{r}-\mathbf{r_t}\|+\|\mathbf{r}-\mathbf{r_r}\|\right)} \tag{2.6}$$

where n_t and n_r denote the number of transmit and receive antennas and r_t and r_r the positions of the corresponding transmit and receive antennas in space.
This is called the focusing equation, where the backscattered signals U^s, that are received by the receive antennas, are superimposed linearly after weighting by a phase term, that incorporates the distance between transmit/receive antennas to the target, one wants to image.

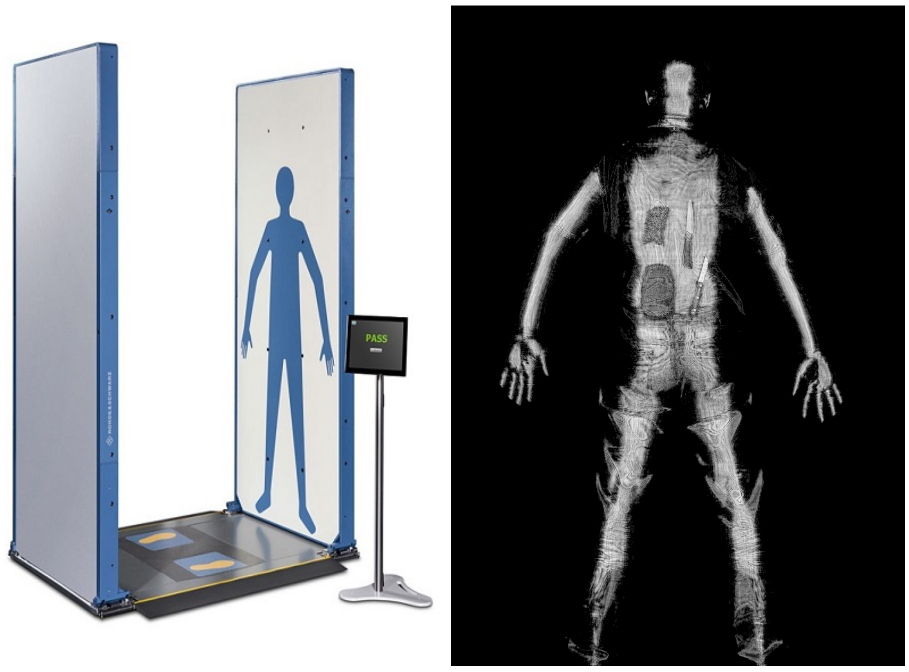
2.1.2 Relation to Human Pose Estimation

The values $\hat{O}\left(r\right)$, as determined with (2.6), give then the image of the object, one wants to reconstruct. This values can be interpreted as a three-dimensional point cloud with reflection values assigned that are represented by $\hat{O}$.
The reconstruction according to (2.6) is performed on a regular grid for r, such that the final point cloud has a regular spacing, resulting in a volume.
As already mentioned above this imaging system is used for security scanner applications (figure 2.1), where it is utilized to picture the human body.
In figure 2.1 also a so-called maximum intensity projection (MIP) of this point cloud can be seen, showing a person in standard pose for airport application. For that application, it is necessary to determine the body pose of the people visible in such a reconstructed images.

2.2 Human Pose Estimation

Definition:
"Human pose estimation is the process of estimating the configuration of the body (pose) from a single, typically monocular, image." [81]

Nevertheless, there are different kind of images/data types (see Section 3) and also variants of estimating the body pose, which are described in the following.



(a) QPS201 Security Scanner (b) MIP projection of a mm-wave-image

Figure 2.1: Imaging System

2D Skeleton Pose Estimation. Quite a huge amount of work was done on estimating the 2D skeleton pose, which refers to the task of finding the 2D location of a certain number of body joints in an image. This task can be considered to be solved, since there are many well working solutions out there, as stated in [37].

3D Skeleton Pose Estimation. But there is also 3D skeleton pose estimation, which refers to the task of estimating body joints in a 3D space. Since there exist quite less labelled datasets and estimating the 3D pose is challenging depending on the underlying data types, there is still work to be done in that area.

3D Shape Estimation. Another variant is shape estimation as used in [3], where it is introduced as the registration process, that maps the body joints of the SMPL model [48] onto the estimated body joints of people in an image. Therefore it can be viewed as a two-step procedure, where first a 2D skeleton pose estimation is performed, and then a registration of body joints of the SMPL model is used to align the shape of the model with the shape of a person in the image. This gives in the end also a mapping for pixel coordinates.

Dense Pose Estimation. Dense Pose Estimation as introduced in [71, 72] is most related to the problem statement of this master thesis, it also refers to a direct mapping of the coordinates of the input data to a canonical representation. But there the mapping refers to mapping 2-dimensional coordinates of pixels onto coordinates of a 2-dimensional space describing the surface of a human body model.

But in the end it all boils down to assigning some kind of input data coordinates to a model of the human body, which could be 2D or 3D joints as well as models used for dense pose estimation or shape estimation.

2.3 Machine Learning (ML)

As with any concept, machine learning may have a slightly different definition, depending on whom you ask. There is a collection of valid definitions:

"Machine learning is the science of getting computers to act without being explicitly programmed." - Stanford

"Machine learning is based on algorithms that can learn from data without relying on rules-based programming." - McKinsey & Co.

"Machine learning algorithms can figure out how to perform important tasks by generalizing from examples." - University of Washington

"The field of Machine Learning seeks to answer the question 'How can we build computer systems that automatically improve with experience, and what are the fundamental laws that govern all learning processes?'" - Carnegie Mellon University

"Machine learning research is part of research on artificial intelligence, seeking to provide knowledge to computers through data, observations and interacting with the world. That acquired knowledge allows computers to correctly generalize to new settings." - Dr. Yoshua Bengio

For the thesis, we stick with the last definition, since it seems to stay most general and hence might be applicable to a wide range of learning algorithms.

There are many different types of machine learning algorithms, with hundreds published each day, and they're typically grouped by either learning style (i.e. supervised learning, unsupervised learning, semi-supervised learning) or by similarity in form or function (i.e. classification, regression, decision tree, clustering, deep learning, etc.).

Since a neural network is some special form of ML algorithm, which will be utilized in this thesis, the basics of such a network will be described in the following.

2.4 Neural Networks and Deep Learning

One very interesting field of machine learning are neural networks. This concept is inspired by biology and the human brain, where neurons are the basic building blocks that are connected via synapses. Depending on their received signal neurons start to "fire", i.e. emit signals to connected neurons.

2.4.1 Artificial Neurons

Similarly one can copy this concept and define a so called artificial neuron[59], which algebraic structure is defined as follows :

$$y = g\left(\mathbf{w}^{\mathbf{T}}\mathbf{x} + \mathbf{b}\right), \tag{2.7}$$

where $\mathbf{x}$, y denote the input and output of the neuron and $\mathbf{w}$, b the parameters called weight and bias respectively. The so called activation function $g\left(\cdot\right)$ is successively applied onto the affine function $\mathbf{w}^T\mathbf{x} + b$.
Nowadays there are different activation functions in use, a famous one is e.g. the sigmoid activation function, which is defined as:

$$\sigma\left(x\right) = \frac{1}{1 + e^{-x}}, \tag{2.8}$$

But due to the problem of vanishing gradients, which appears, when $|x|$ gets big, the sigmoid activation is not widely used in practice, instead the *ReLU*, *leakyReLU* or *tanh* activations are used.

$$ReLU(x) = \begin{cases} x & \text{if } x \geq 0 \\ 0 & \text{else} \end{cases}, \tag{2.9}$$

$$leakyReLU(x) = \begin{cases} x & \text{if } x \geq 0 \\ ax & \text{else} \end{cases}, \tag{2.10}$$

$$tanh(x) = \frac{e^{+x} - e^{-x}}{e^{+x} + e^{-x}}, \tag{2.11}$$

2.4.2 Multilayer Perceptron (MLP)

It is straight forward to define a network from its basic building block the neuron. One can just use them in parallel on the same input $\mathbf{x}$ to create different features of the same data, which results in a so-called layer with vector-valued output $\mathbf{y}$, where all the scalar outputs y of each neuron are stacked together. Cascading such layers results in a network which is called multilayer perceptron, which characterized by having fully connected layers, i.e. each neuron is connected to each neuron in the next layer.
The overall mathematical description of a MLP with N hidden layers is given by the following equation:

$$MLP(\mathbf{x}) = \mathbf{g}_N\left(\mathbf{W}_N\mathbf{g}_{N-1}\left(\mathbf{W}_{N-1}...\mathbf{g}_0\left(\mathbf{W}_0\mathbf{x} + \mathbf{b}_0\right)... + \mathbf{b}_{N-1}\right) + \mathbf{b}_N\right), \tag{2.12}$$

where $\mathbf{g}_i$: is the element-wise applied activation function at layer i and $\mathbf{W}_i$ the weight matrix and $\mathbf{b}_i$ the bias vector.

One very interesting property of such an MLP is, that an MLP with a single hidden layer and a finite number of neurons can approximate any continuous function on a compact subset of $\mathbb{R}^{\mathrm{n}}$, which is stated by the universal approximation theorem[14, 27].

2.4.3 Deep Learning (DL)

To get something useful of such a MLP, one has to determine its parameters W_i and b_i, this is what is called training or learning. Therefore the gradients of the objective functions with respect to W_i and b_i are of interest and can be calculated via back-propagation very efficiently. The objective functions are a measure of performance of the network and hence are optimized.
Experience has shown that building such networks very deeply improves their performance. The field dealing with such networks is called Deep Learning. But actually, there is no number of how many layers such a network must have to be called "deep".

2.5 Convolutional Neural Networks (CNNs)

Convolutional neural networks can be viewed as a special type of an MLP, where the weight matrices are sparse, such that each neuron does not depend on all neurons of the previous layer. As a further restriction, weights are shared for several neurons, e.g. the weight matrices W_i have a structure such that certain weights within that matrices are forced to be the same. This weight sharing and sparseness leads to a less overall effective number of parameters and is hence easier to train.

Becoming more clear convolutional neural networks use convolutions to realize a linear relationship between input and output. Thereby the input of each layer is convolved with a filter, which results in the above mentioned weight sharing and less connections, since each neuron then just depends on as many neurons from the previous layer as the filters have weights and all the neurons within one feature map share the same filter and hence the same weights. After filtering also activations are applied to each neuron.

This kind of network has been very successfully applied in the area of image processing and computer vision, where one has to deal with images, which are 2D structures. For images, 2D convolutions have always been a useful operation to detect local features, as for example, corners and edges. And as these classical feature detectors make use of this local dependencies in an image also neural networks can exploit these dependencies by going from fully connected layers to convolutional layers.
But not only 2D convolutions are of interest in general, 3D convolutions can be very useful when dealing with 3D data. That is why they are utilized for the task of human pose estimation in this thesis, where we are working with 3D body scans.

Chapter 3

Related Work

In this chapter a quick overview about the literature in human pose estimation is given as well as literature for segmentation is reviewed.
By doing so, the related work section is structured in a way such that the discussed literature is getting more related to the thesis problem statement while proceeding through the sections.

3.1 Human Pose Estimation

Due to the rich literature on human pose estimation, a full review, giving a complete overview of the literature in human pose estimation of the last 20 years is out of the scope of the thesis. In order to get a wider overview, consult [55, 66, 64, 77]. Nevertheless, several related approaches shall be reviewed in the following.

Pose estimation on the human body and its parts has gotten a lot of attention in the last 10 - 20 years, which was for sure supported by the fact that machine learning found application in a wide range of problems, such as pose estimation. A huge amount of work has been done on human pose estimation in the last years, people worked not only on estimating the whole body pose [3, 13, 50, 51, 21, 22, 23, 31, 32, 33, 34, 35, 42, 44, 48, 54, 58, 72, 80, 85, 88, 90, 93, 94, 100], but also focused on specific parts, such as the hand [46, 60, 92, 95, 98] and the face [71].

Latest works focus on improving multi-person pose estimation[58], the performance in more realistic backgrounds (in the wild)[90], occlusion handling [78], exploitation of temporal information [28] or new forms of pose estimation like Shape Estimation [3] or Dense Pose Estimation [71, 72].
Nie et al.[58] seeks to improve multi-person pose estimation with a single-stage approach, meaning instead of splitting the problem into people detection and subsequent joint prediction or wise-versa, he estimates multiple skeletons at once by using

dense confidence and displacement maps for root and body joints. Wang et al.[90] improves human pose estimation in more realistic backgrounds and environments with help of data set synthesized through 2D to 3D Joint uplifting. Sarandis et al. [78] also uses data synthesis to augment images with occlusion for improved occlusion handling. On the other hand Hossain et al.[28] tries to improve the uplifting of body joints from 2D to 3D by exploiting temporal information. A two-step approach for simultaneous 3D pose and shape estimation is developed by Zanfir et al.[3] by first estimating 2D and 3D Joints and subsequently fit them with additional silhouette information to the SMPL model. Güler et al.[72] formulates pose estimation on RGB images in a new way, first for estimation on the face and then on the whole body by using massive data annotation to create UV-maps on real images for part segmentation and simultaneous UV-map prediction, resulting in a dense map.
Furthermore, there is work done on many different types of data such as RGB images, Depth images, point clouds, volumetric data and combinations of them. In the following approaches on different data types will be described more closely.

Human Pose Estimation on RGB

On RGB data it is very challenging to infer the 3D pose, since an image results from a 2D perspective projection of the real world and hence a single RGB image doesn't contain real 3D information. To infer 3D information and pose from RGB images it is necessary to have prior knowledge about the object visible in a scene. What a neural network can do is for example to recognize certain patterns of the human body and with the help of prior knowledge, which can be a body model one can infer a 3D pose. This is for example done in the approach of [3], where he estimates the shape of the human body in an RGB images with the help of the SMPL model. Also Güler et al.[72] uses a body model but in a more implicit way. He uses a surface model for annotation to create UV-map ground truth data. The creation of rich ground truth for the COCO-Dataset is the main contribution of that approach apart from the dense formulation of pose estimation and the proposed network architecture for simultaneous prediction of part segmentation and UV-map. In contrast to that newer forms of pose estimation there still is a huge amount of work focusing on the classical approach, predicting skeleton joints from RGB images [6, 2, 11, 50, 51, 22, 26, 28, 54, 58, 77, 78, 86, 90, 93, 94].

Human Pose Estimation on Depth Images and RGB-D

Working on depth images directly [15, 23, 33, 34, 42, 31, 44, 46, 60, 80, 88, 92, 95, 98] or exploiting additional depth information through RGB-D images[13, 47, 52] eases up the problem significantly, since real 3D information is used. But still, self-occlusion and sparseness due to the viewpoint hinder the data to provide full 3D information.
Where [15, 23, 33, 34, 42] estimate the pose in a dense sense, other approaches on depth images estimate the classical skeleton [31, 44, 46, 60, 80, 88, 92, 95, 98].

Through the very popular application in console games the RGB-D camera system Kinect has gotten a lot of attention. The algorithm used for pose estimation in Kinect was developed by Shotton et al.[31] based on pure depth information. He solves the 3D Pose estimation problem by using as intermediate representation a body part segmentation to finally predict body joints.
For estimating the body pose in a dense sense Wei et al.[42] provides an algorithm for establishing dense correspondences. He is learning a per depth image pixel descriptor by training a neural network on a body part classification task. Correspondences between images are then established by a nearest neighbour search in the descriptor space.

Human Pose Estimation on Point Clouds and Volumes

Pose Estimation on Point Clouds [32, 61, 62, 85, 32, 61, 62, 85] and volumes[56] has seen less effort. This and the fact that complete body scans of the human body provide full 3D information and are best represented as point clouds or volumes motivate the research in that area and the proposed method of the thesis.

3.2 Segmentation

Segmentation on 3D Point Clouds and volumes

As mentioned before it is beneficial to think about a way how to partition the human body into zones, which in fact gives rough correspondences. That directly leads to the task of point cloud segmentation. So far there is a lot of work in this area. Several approaches directly work on point clouds such as PointNet, [67],PointNet++[68], 3D Capsule Net[96] and AtlasNet[25], whereas other approaches like Vnet[53], VoxNet[49], 3D-Unet[89] are designed to work on volumes.

Chapter 4

Data set

In order to be able to develop and test algorithms for pose estimation at least one data set is necessary. First of all the requirements on the data set are given in Section 4.1, then Section 4.2 contains a quick summary of the results of the research on data sets and synthesis methods in the area of pose estimation. Finally, the data set synthesis is described in detail in Section 4.3.

4.1 Requirements on the data set

The following requirements have been set to meet the actual data format of the body scanner

- Point cloud in volumetric representation, i.e. regular spaced 3D grid
- Resolution: 512x256x512 Voxels
- Voxelsize: 0.5mm x 0.5mm x 0.5mm
- The point cloud in volumetric representation can be written as a tensor

$$D \in \{d \mid d \in R^{512\times256\times512}, d_{ijk} \in \{0,1\}\,\forall i,j,k\}, \tag{4.1}$$

  where D contains the complete surface of different human bodies in varying walking poses (see figure 4.1) and the d_{ijk} represent the scattering potential as introduced in (2.6) for a volume representation a certain space. It has to be noted, that the tensor D was restricted to be binary, whereas the scattering potential (2.6) is not binary in general. But to get a non-binary output, the imaging property of the imaging system, which is mainly set by the antenna array geometry and the imaged material, would have to be incorporated for data synthesis, which is out of the scope of the thesis. Nevertheless, this restriction rather makes the problem of pose estimation a little harder, because clues from the imaging properties, which are space depended are not being used.

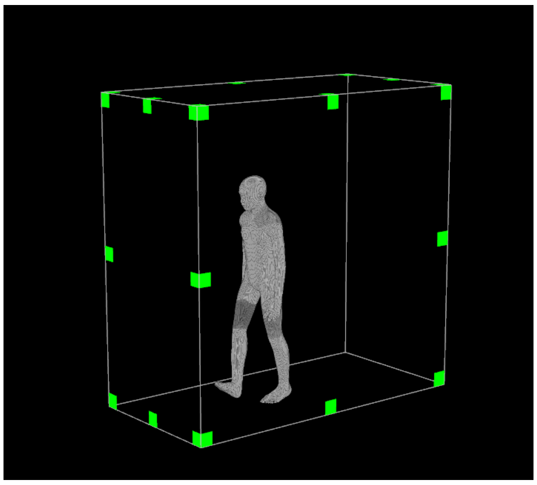


Figure 4.1: Exemplary Point Cloud in volumetric representation

- The labels for every point in the point cloud are given by the tensor

$$L \in \{l \mid l \in R^{512 \times 256 \times 512}, l_{ijk} \in \{1, ..., C\} \, \forall i, j, k\}, \tag{4.2}$$

hereby the label indicates a certain body part.

4.2 Related Data Sets and Synthesis methods

In literature there can be found a lot of data sets in pose estimation, but no data set addressing the actual identical problem. The data sets are examined in Subsection 4.2.1. Also related methods for synthesis, that inspired the suggested method for data set synthesis, are described in Subsection 4.2.2 and finally the motivation for the proposed data set synthesis method is given in Subsection 4.2.3.

4.2.1 Related Data Sets

The main problem with the existing data sets is that they rather are based on different physical data, e.g. RGB, RGB-D, pure depth images, sparse point clouds from ToF-Sensors or Laser Scans, or that they have no appropriate labels showing a clustering of the human body in a regular spaced dense point cloud format. As well the data sets don't exclusively contain humans for that very constrained walking scene in a clutter-free environment.
Through the table 4.1 an overview of several datasets, found through our research, is given.

MPII-HUMAN POSE[4] is a benchmark data set for 2D skeleton human pose estimation on RGB images. It was created by performing Youtube queries on certain activities structured in a hierarchical way. Afterward, the relevant frames were extracted manually by human annotators to end up with 40522 images of different humans. These humans were annotated manually with 2D body joints and 3D viewpoint of the head and torso.

On the other side, there are also purely synthetic data sets like *SURREAL* [87], which was created with a focus on creating a large scale data set with realistic images of people. To create the data set the SMPL model[48] was used. Shape parameter were extracted from *CAESAR*[73] data set and for animation MoCap data was used from the CMU MoCap database. From these 3D animations, RGB images were rendered with background from the LSUN data set.

Providing also 3D meshes for humans captured with RGB cameras *UP-3D* [37] is a data set that builds on existing data sets like *MPII-HUMAN POSE*[4] to estimate 3D shape in RGB images with an extended version of the SMPlify[21] method. It also provides 3D Joints and body part segmentation in 2D.

A further data set dealing with meshes is the *SCAPE*[5] dataset, which contains 71 registered meshes of a single subject in different poses. Since the meshes are reconstructed from real data, they are more realistic (e.g., they do not have the same local shape features). The meshes were registered using only geometric information. Hence it is unclear how accurate the deformations in this dataset are.

Speaking of shape information and mesh data *FAUST*[7] is a data set of registered human body meshes, that was built with the aim to accurately provide correspondence information between different persons and poses. These meshes are registered using a novel texture-based registration technique, where real humans had to place a certain high-frequency texture on to their skin. The humans then are recorded by a 3D multi-stereo system and a single RGB camera. One limitation of *FAUST* is that is restricted for research purposes.

DYNA[65] is an extension to the *SCAPE* model, that was build from the *SCAPE* data set. Where *SCAPE* only approximates static surface deformation, *DYNA* was build to include full body deformations driven by the motion of the body. The model used to build *DYNA* is a mathematical model that relates deformations of the human body surface to changing the poses in time. In order to build that model from data, the *DYNA* dataset with 40k registered meshes was captured. For registration only geometry information was used.

Building on ideas from *FAUST* and *DYNA*, *D-FAUST*[8] can be viewed as extension to *FAUST* that also takes care of soft-tissue deformations and hence captures information of the human body in dynamic situations. For the creation of the registered meshes, *D-FAUST* combines 2D correspondences in texture space with a model-based registration approach dealing with temporally-offset streams of geometry and texture data. But it lacks in a dense sampling of the shape space since it also provides just 10 people.

A further data set with registered mesh data is the *CAESAR* [73] data set, which build with the purpose to create a database about civilian anthropometry. It was captured with a laser-driven triangulation system, that creates a full-body scan and also rough correspondences from 100 landmarks.

TOSCA [10] is a synthetic dataset that is widely used for evaluation of mesh registration methods. It provides 80 artificially created meshes of animals and people (with 3 subjects in a dozen different poses each). Meshes in the same class share the same topology, so ground-truth correspondences are immediately defined. The meshes and the deformations, however, are unrealistic and there is no noise or missing data.

Further data sets build for 3D Joint estimation are the *HUMAN3.6M*[30] data set and the *HUMANEVA*[82] data set. *HUMAN3.6M* and *HUMANEVA* are both data sets that were build in a laboratory setup. In *HUMAN3.6M* 11 people were recorded with 4 digital video cameras, one ToF-Sensor and 10 motion capture cameras. In *HUMANEVA* only 6 people were recorded in various activities and also with simultaneous capture of video and motion information. *HUMANEVA* with about 40k images is a rather small data set compared to *HUMAN3.6M* with 3600k images.

To summarize, as can be seen from the table, many data sets are build to have RGB images as input provide no body shape information in form of meshes at all[4, 30, 82, 87] and hence are not useful. Some data sets with RGB images leverage [87] or provide[37] mesh models. But the latter data set is very small and the meshes are just built with the help of another shape estimation approach[21] and hence are inaccurate and also don't fit the required pose. Also, depth images are provided through some data sets[87, 30], but they don't cover the human body completely and also don't provide any correspondence information among the depth images.

Data Set	**Input Type**	**Scenes**	**Clothes**	**Labels**	**Synthetic (input/label)**	**#Samples**
MPII-HUMAN POSE	RGB Images	800 activities	Yes	2D Joints	No/No	∼40k Images
SURREAL	RGB Images	Indoor with varying poses	Yes	3D Joints, Depth Map, Part Segmentation 2D	Yes/Yes	∼6536k Images
UP-3D	RGB Images	Various activities	Yes	3D Joints, 3D Meshes, Part Segmentation 2D	No/Yes	8,5k Images
FAUST	Full Body Scans (static, 172k points)	Various poses	Yes	Registered 3D Meshes (very accurate)	No/Yes	0,3k scans of 10 people
D-FAUST	Full Body Scans (motion Sequences, 150k points)	Motion Sequences with 60fps	Yes	Registered 3D Meshes (very accurate)	No/ Yes	40k scans of 10 people
HUMAN3.6M	RGB Images	Various activities (urban and office scenes)	Yes	3D Joints, Depth Map	No/No	∼900k x 4 Images of 11 people (∼64k x 4 of walking)
HUMANEVA	RGB Images	Various activities (jogging, gesturing, boxing,combo,...)	Yes	3D Joints	No/No	∼40k Images of 6 people
CAESAR	Full Body Scans	3 static poses	Yes	Texture information, 3D Mesh (100 landmarks)	No/No	4,8k x 3 Scans of 4,8k people
SCAPE	Full Body Scans	Various poses	Yes	3D Joints, Registered 3D Meshes	No/Yes	1x70+ 37x1 scans (people x pose)
TOSCA	Meshes	-	-	-	Yes/-	80 Meshes of people and animals
DYNA	Full Body Scans as motion Sequences	Motion Sequences with 60fps	Yes	Registered 3D Meshes (not accurate)	No,Yes	∼40k scans of 10 people

Table 4.1: Overview about related data sets

Then there are the data sets[5, 7, 8, 65, 73] that provide full body scans with registered mesh models or pure mesh models [10]. Where the *SCAPE* and *TOSCA* data set are simply too small, the *CAESAR* data set lags in variance in the poses. This leaves us with *FAUST*, *D-FAUST* and *DYNA*. All three of them lack in sampling the shape space densely, since they all provide only scans from 10 different persons. Furthermore, *FAUST* doesn't suit, because it captures just various static poses. The most interesting data set is *D-FAUST* since it captures humans in dynamic scenes like walking and outperforms DYNA in terms of registration precision, but it still lacks in sampling the shape space and pose space for that walking scene densely.

To conclude, no data set is meeting the requirement to fit the input tensor D or L and the problem setting directly. Hence, in any case, one has to build a pipeline to create the tensors D and L.

4.2.2 Related Data Set Synthesis methods

Even if explaining related data sets also means describing how they are created or synthesized, in the following a quick overview about related synthesis methods utilizing mesh models for various applications shall be given.

For the training of the pose estimation algorithm of Kinect Shotton et al.[31] renders 2D depth images based on MoCap data and meshes of human bodies to create a data set. Also for optical flow estimation, 3D models of chairs have been used and rendered by [16] into RGB images. Furthermore, real RGB images have been augmented by rendered 3D models for object detection[63] and viewpoint estimation has been profited from augmenting real images with 3D models of different objects in different viewpoints[84]. Fanello et al.[19] renders infra-red and depth images from 3D models using Poser to generate 100K hand and face image pairs. Also for modeling of hands with a subsequent rendering of depth images 3D models were used by [57] for hand pose estimation.

When it comes to modelling of the human body meshes were used by several researcher for creating synthetic data for 2D pose estimation[39, 75, 91] , 3D pose estimation[12, 17, 20, 24, 74, 83, 97], detection[38, 40, 45] or action recognition[69, 70].

4.2.3 Motivation for Data Set Synthesis

The most important reason for creating an own pipeline for data synthesis is the fact that no other data set meets all the requirements discussed in Section 4.1. Hence it is not possible to use these data sets out of the box.

The main focus in creating the data set lies not in creating a very detailed model of all highly realistic deformations of real human bodies, where the data sets [7, 8, 65] have focused on, since voxelization, which is necessary for that problem, reduces the shape information in the models. The focus rather lies in a dense sampling of possible shape and pose variations for that very constrained walking scene, where the mentioned data sets have a lack in. As well as the people for that walking scene, should not wear clothes, since they are not visible through the imaging technology. But all mentioned data sets show dressed people. A further benefit for the release and use of a data set, that avoids scans or images of real people, is that one does not run into anonymity issues.

To get more real-world deformations of the meshes into the scans the SMPL model could have been used since it is compatible with the synthesis process, that will be explained in 4.3. But as we focus not on creating highly realistic real-world models and using the SMPL commercially could mean some license issues, it was not used.

And there are the general benefits of data synthesis as well. Data synthesis makes the data creation very flexible. Through the provided pipeline, it is just necessary to sample some human body models and with the help of MOCAP data, thousands of samples can be created immediately without any further effort. This means we end up with a nearly infinite data source, which is just limited by the number of mesh models and MOCAP data. As a further benefit, the pipeline provides skeleton and mesh information, which could be used for multi-task learning.

Furthermore, the creation of some data sets involves manual human annotation, which is always questionable, since it depends on the annotator itself and introduces errors. Therefore by using data synthesis human annotation was avoided.

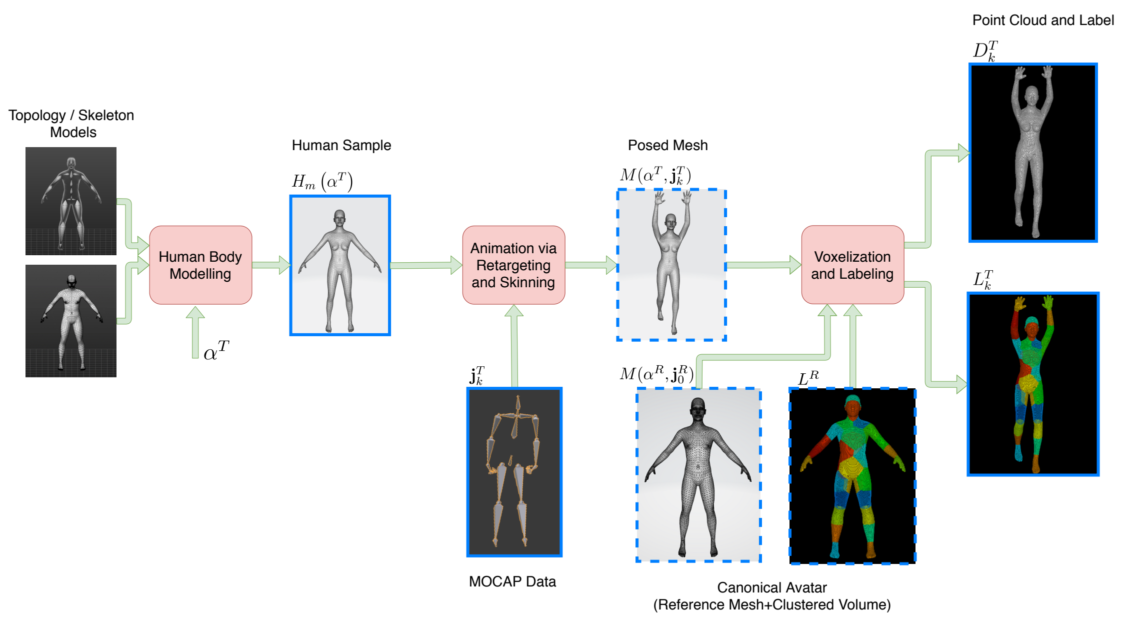

Topology / Skeleton Models
Human Body Modelling
α^T
Human Sample
$H_m(\alpha^T)$
Animation via Retargeting and Skinning
$\mathbf{j}_k^T$
MOCAP Data
Posed Mesh
$M(\alpha^T, \mathbf{j}_k^T)$
$M(\alpha^R, \mathbf{j}_0^R)$
L^R
Canonical Avatar (Reference Mesh+Clustered Volume)
Voxelization and Labeling
Point Cloud and Label
D_k^T
L_k^T

Figure 4.2: Overview about the data synthesis pipeline, the basic parts are depicted as red boxes

4.3 Data Set Synthesis

To be flexible in terms of synthesis, a completely automatic toolchain for data set synthesis was developed. This toolchain allows creating point clouds of humans posing arbitrary in space, as well as to create segmentation labels for an arbitrary number of body zones. This toolchain was developed as a first step to create a dataset according to the needs of the problem setting, i.e. according to Section 4.1 An overview of the parts of the synthesis pipeline is given in figure 4.2.
Human Body Modelling is discussed in 4.3.2, while Animation is explained in detail in 4.3.3 and Voxelization and Labelling is described in 4.3.4

4.3.1 Tooling

In order to be able to create sequences of different humans walking differently through the body scanner, a good tooling was necessary for every step in the data synthesis pipeline. For the selection of tools freeware or open source solutions were preferred.

Human Body Modelling

For the creation of human body models according to Section 4.1 the only free tool with appropriate degrees of freedom (age, gender, height, weight, proportions,etc.) in modelling humans was MakeHuman [9, 43] according to the results of our research, hence it was used for modeling.

Further Tools that were examined are Poser, Manuel Bastioni Lab, and Virtual Caliper.

Motion Capture Data

For the acquisition of motion capture data, i.e. time sequences of skeletons, one could use databases like Cologne Motion Capture Database or any other database. To make the dataset suit the application custom MOCAP data was collected by using Kinect and a tool for MOCAP data recording, called Brekel.

Animation

The landscape of tools for animation is huge through the wide use of these tools in the gaming industry. Considered tools were Blender, Poser, Unity, Collada, Maya, 3DS and Lightwave.
Through the interfaces provided to MakeHuman and the compatibility between MakeHuman and Blender, Blender was chosen to be the tool used for animation. It also provided a plugin for automatic retargeting of motion capture skeletons onto the skeletons of the human body model.

Mesh Processing

For Voxelization and mesh processing the Visualization Tool Kit (VTK) was used in python.

4.3.2 Human Body Modelling

One big sub-field of Computer Graphics is 3D Modelling, which refers to the task of finding a mathematical representation of any surface of an object. The resulting surface model is described by a so-called mesh, which consists of vertices and edges, both together define faces and normals on the faces .
Such a polygonal mesh representation was used to describe the human body surface, i.e. a human body model consisted of:

- a skeleton $S = \{B, J\}$
  - B: Bones
  - J: Joints (points were bones meet)
- a mesh $M = \{V, E, F\}$
  - V: Vertices (points on the skin)
  - E: Edges (lines between vertices)
  - F: Faces (closed sets of edges)

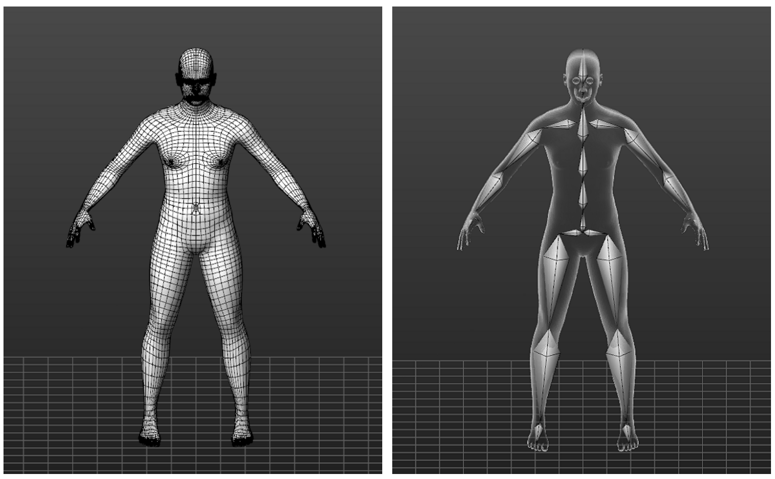

(a) Mesh M of the model (b) Skeleton S of the model

Figure 4.3: Human Body Model

- a group G
  - Control skinning through weights w in animation

This results in a model $m = \{S, M, G\}$ with a certain mesh topology defined through vertices and edges and also a topology for the skeleton. These topologies are fixed ones before the animation and shape modeling process is done.
The shape modelling process of human mesh models can be a very tedious task. Designing different meshes by hand through adjusting manually vertices positions in space is not only very time consuming but also difficult when we want to achieve natural-looking results.

Therefore good tools are necessary, that enable the alignment of points and morphing of a mesh into natural-looking shapes. Such tools are still really time-consuming when it comes to modelling individual humans in detail, with different body shapes, considering differences like height, weight, proportions and shapes, etc. For that degree of detail in designing mesh models, tools like MakeHuman[9] help to model people by just adjusting different shape parameters. Humans $H_m(\alpha)$ therefor all share the same m, but differ in their shape parameters α. If one still wants to save time in modelling through adjusting shape parameters, one can randomly sample parameters in the parameter space, after defining a certain distribution over the parameters.
In the easiest case the parameters are sampled from a uniform distribution. This approach was actually taken, but in order to be able to sample and adjust these parameters, it was necessary to adapt the source code from MakeHuman to handle the automatic adjustment of relevant parameters through scripts.

Through the tool it was possible to select from a set of 8 mesh topologies. The topology was chosen to be generic in gender, which just left us with 2 topologies, from which the one with the higher number of vertices ($|V| = 13378$) was chosen to achieve a more detailed modelling, which helped for further processing.
The skeleton "Cmu MB" was chosen in order to be compatible with further animation of the meshes.

4.3.3 Animation

The following steps in the animation process are necessary to create a single animation scene corresponding to one human $H_m(\alpha)$ and one skeletal animation sequence.

Acquisition of MOCAP data

In order to consider the real scenario of walking through a body scanner, Kinect and its 3D pose estimation software was used to acquire motion capture data, i.e. an estimate of the skeleton joints J^{MOCAP} for each RGB-D image. This resulted in a time series of joint positions and rotations:

$$\mathbf{j}^{MOCAP} = \left\{\mathbf{j}_1^M, \mathbf{j}_2^M, \mathbf{j}_3^M, ..., \mathbf{j}_K^M\right\} \tag{4.3}$$

Retargeting

The goal of retargeting is to match a motion of the skeleton joints J^{MOCAP} onto the target skeleton joints of the model J^{TARGET}. For retargeting the MakeWalk Add-On in Blender was applied, which resulted in an equivalent series for the target skeleton:

$$\mathbf{j}^{TARGET} = \left\{\mathbf{j}_1^T, \mathbf{j}_2^T, \mathbf{j}_3^T, ..., \mathbf{j}_K^T\right\} \tag{4.4}$$

Skinning

The task of skinning is the calculation of vertices corresponding to the underlying skeleton. When talking of time step k with target joint positions and rotations $\mathbf{j}_k^T$ skinning calculates the corresponding vertex coordinates $\mathbf{v}_k^T$, resulting in a series:

$$\mathbf{v}^{TARGET} = \left\{\mathbf{v}_1^T, \mathbf{v}_2^T, \mathbf{v}_3^T, ..., \mathbf{v}_K^T\right\} \tag{4.5}$$

Here the vertex coordinates can be calculated with one of the following methods:

- Rigid Skinning (RS):
  - Each vertex V_l^T at time step k with position $\mathbf{v}_{k,l}^T$ corresponds to only one bone B_i

  - If bone B_i gets transformed by a euclidean transform given by $\mathbf{A}_{k,i}$, the new vertex position calculates to:

$$\mathbf{v}_{k,l}^{T} = \mathbf{A}_{k,i}\mathbf{v}_{0,l}^{T} \quad \forall k \in \{1, ..., K\}, l \in \{1, ..., |V|\}, \tag{4.6}$$

    where $\mathbf{v}_{0,l}^{T}$ denotes the position of vertex V_l^T in a reference pose defined by $\mathbf{j}_0^T$. The transformation $\mathbf{A}_{k,i}$ can be calculated from $\mathbf{j}_{k,i}^T$ and $\mathbf{j}_{0,i}^T$ by traversing through a kinematic tree.

- Linear Blend Skinning(LBS):
  - Each vertex V_l^T at time step k with position $\mathbf{v}_{k,l}^T$ corresponds to several bones
  - The influence of bone B_i onto vertex V_l is controlled by weights $w_{l,i}$ and the position of vertex V_l^T in time step k calculates to:

$$\mathbf{v}_{k,l}^{T} = \sum_i w_{l,i}\mathbf{A}_{k,i}\mathbf{v}_{0,l}^{T} \quad \forall k \in \{1, ..., K\}, l \in \{1, ..., |V|\}, \tag{4.7}$$

- Dual Quaternion Skinning (DQS) and others

For animation in blender the LBS algorithm was used.

4.3.4 Voxelization and Labeling

Voxelization

Given a time series of meshes $M^{TARGET} = \{M_1, M_2, M_3, ..., M_K\}$ with a corresponding time series of vertex coordinates $\mathbf{v}^{TARGET}$ as in (4.5), one has to voxelize the meshes for each time step to get a point cloud with regular spacing.
This voxelization is performed with a linear extrusion filter on the mesh and subsequent application of a stencil function. The extrusion filter expands the mesh into two meshes, where the volume between the two meshes defines the skin of the human. The stencil function then sets the voxels within the skin to one, the rest is left to be zero. So from each time frame in M^{TARGET} we get a tensor D as described in (4.1), hence we end up with a series of tensors

$$D^{TARGET} = \left\{D_1^T, D_2^T, D_3^T, ..., D_K^T\right\} \tag{4.8}$$

Definition of a reference avatar

For the creation of the labels the definition of a reference mesh was necessary. Shape parameters α^R and reference pose $\mathbf{j}_0^R$ were chosen, ending up with a reference mesh $M(\alpha^R, \mathbf{j}_0^R)$. This mesh then also was rendered into a volume given by the tensor D^R.

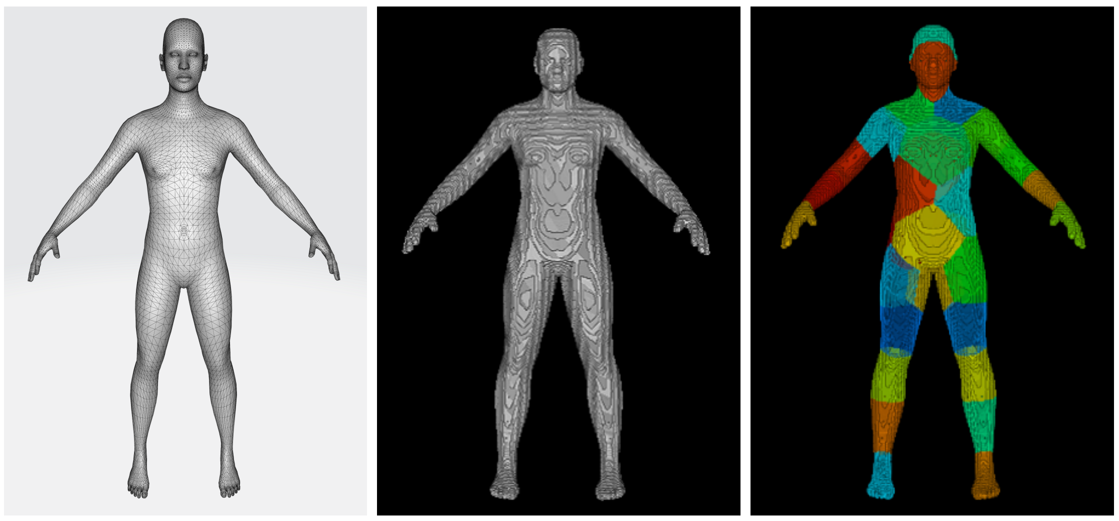

(a) Reference mesh $M(\alpha^R, \mathbf{j}_0^R)$ (b) Reference volume D^R (c) Reference label L^R

Figure 4.4: Reference avatar

To create body part labels the k-Means clustering algorithm was applied on the coordinate vectors of the non-zero voxels in D^R , such that we end up with a Voronoi tesselation of the body surface as depicted in figure 4.4, which gives the labels L^R.

Labeling of a target tensor

With the help of this tensor D_k^T the labels L_k^T were created. To create labels for different time steps and persons $H_m(\alpha)$, one has to re-identify the same body zones from the reference avatar on different poses and persons. To do so the fixed topology of the mesh was used, i.e. through the fixed topology, there exists a correspondence of vertices.
This correspondence defines a map between the surfaces of different people and poses, i.e. for each vertex V_l in $\mathbf{v}_k^T$, there exists a displacement vector $\mathbf{v}_{k,l}^T - \mathbf{v}_l^R$, where $\mathbf{v}_l^R$ is the position of vertex V_l of the reference mesh $M(\alpha^R, \mathbf{j}_0^R)$.

These displacement vectors can be interpreted as resulting from the map $\phi(\mathbf{x})$ (1.1), such that we can define a displacement map

$$\gamma : \mathbf{x} \rightarrow \gamma(\mathbf{x}) = \phi(\mathbf{x}) - \mathbf{x} \quad \in \mathbb{R}^3, \mathbf{x} \in \mathbb{R}^3, \tag{4.9}$$

which maps a vertex of the reference mesh $M(\alpha^R, \mathbf{j}_0^R)$ with position $\mathbf{v}_l^R$ onto its displacement vector $\mathbf{v}_{k,l}^T - \mathbf{v}_l^R$. Through the $|V|$ displacement vectors the displacement map is just defined at positions $\mathbf{v}_l^R$, hence we have to interpolate the displacement field onto the positions of the non-zero voxels in tensor D^R to transfer the labels L^R

onto the labels L_k^T for each frame.

Interpolation can be understood as finding the underlying map γ, which was estimated with the help of thin plate spline interpolation. The overall vector valued interpolation function γ is split into three scalar functions:

$$\gamma_n(\mathbf{x}) = \sum_{l=0}^{N} w_{l,n}\varphi(\|\mathbf{x} - \mathbf{x}_l\|) \quad n \in \{1,2,3\}, \tag{4.10}$$

where φ denotes the radial basis function(RBF), which was chosen to be $\varphi(r) = r^2 \log(r)$, such that we get a thin plate spline. Enforcing the following

$$\gamma_n\left(\mathbf{v}_l^R\right) = \left(\mathbf{v}_{k,l}^T - \mathbf{v}_l^R\right)_n \quad \forall l \in \{1,...,|V|\}, n \in \{1,2,3\}, \tag{4.11}$$

results in three linear system of equations, that are solved to get the parameters $w_{l,n}$ for the three scalar interpolation functions. The interpolated displacement map γ can then be evaluated at the non-zero voxel positions of D^R to transform the labels L^R to their position in time step k and then apply a further 1NN interpolation to get the labels at non-zero voxel positions of D_k^T, resulting in L_k^T. Repeating this for all k yields the time series:

$$L^{TARGET} = \left\{L_1^T, L_2^T, L_3^T, ..., L_K^T\right\} \tag{4.12}$$

The schematic figure 4.5 gives an overview about the voxelization and labeling process.

4.3.5 Timing improvements on the Labeling Process

The problem with implementing the labeling process directly as described in the previous section is the immense amount of time the label creation needs per frame. This time is caused by the fact, that for finding the interpolation function three linear systems of equations with $|V|$ unknowns and $|V|$ equations each have to be solved. Hence we have chosen the number of vertices for the mesh to be $|V| = 13380$, we get huge linear systems of equations, where finding a solution for all systems of equations needs several minutes on the working machine.
With the help of the k-Means clustering algorithm the number of points was reduced. Through empirically adapting the number of clusters/points for k-Means, it was found that a number of 1000 vertices was sufficient for transferring the labels.
In figure 4.6 it can be seen what happens if the number of vertices used for the interpolation function is too small. No precise transfer of the labels is possible, when choosing just 200 vertices for interpolation.
By reducing the number of vertices by a factor of 13 the time needed for interpolation on the target pc was just 6 seconds. By parallelizing the solution of the three systems of equations or the whole computation of one system of equations, the time could

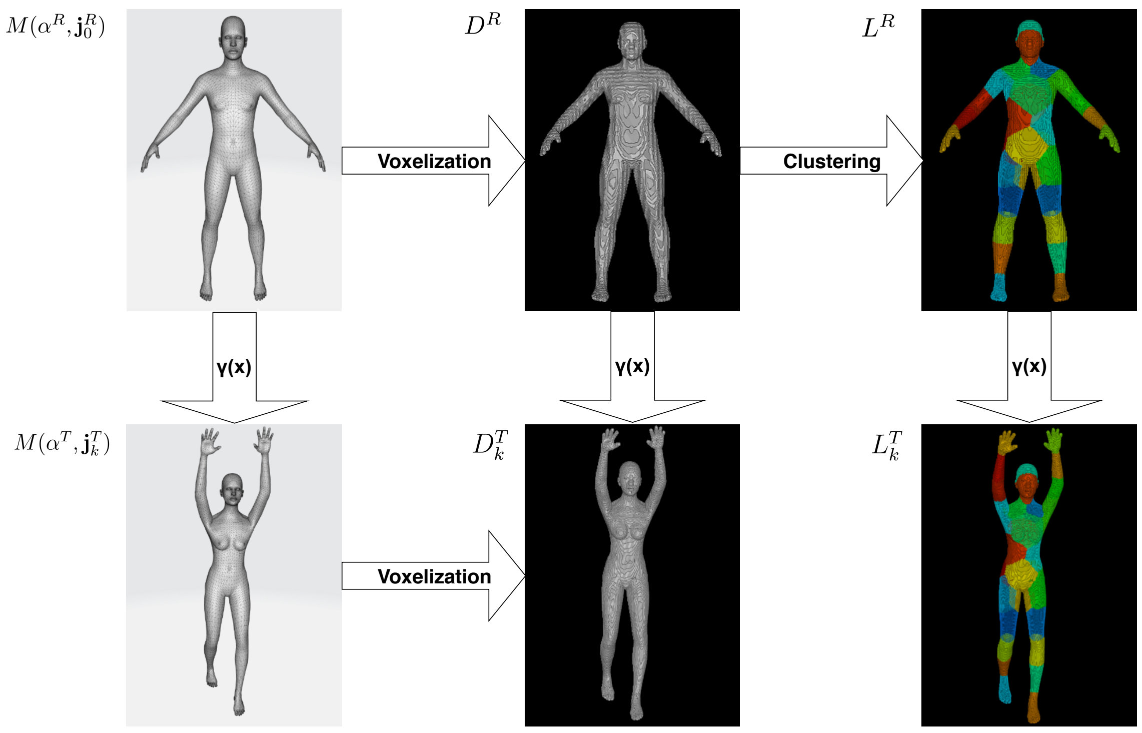


Figure 4.5: Voxelization and Labeling

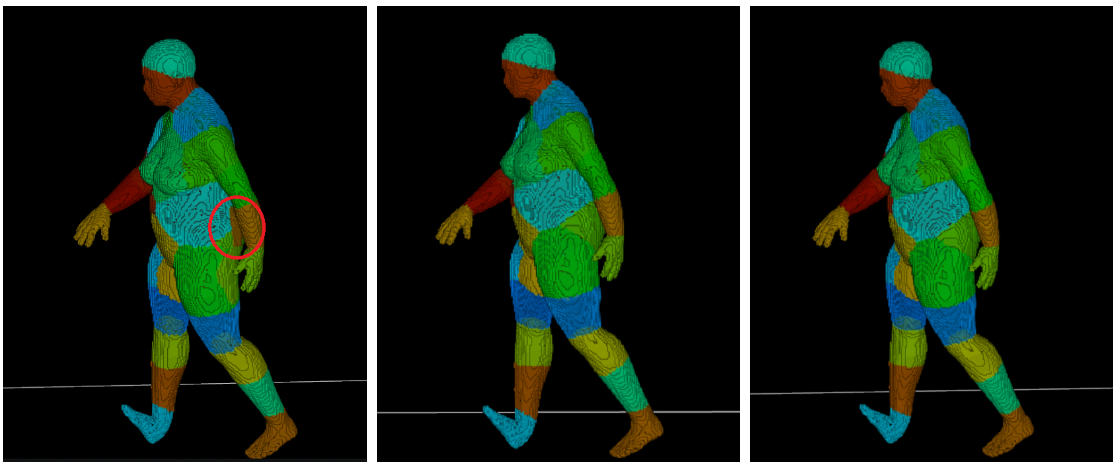
have been reduced further, but due to the fact that the time for voxelization lies in the same order of magnitude as the time for interpolation then, further improvements on interpolation, wouldn't have had a big impact anymore.

4.4 Definition of a Data Set

With the help of the synthesis pipeline as described in Section 4.3 we are able to create a data set very flexible by just collecting motion capture data and sampling humans from the body model to feed both into the pipeline.
The composition of the introduced dataset can be taken from table 4.2 .
The parameter sets $\Lambda_1, \Lambda_2, \Lambda_3$,where the shape parameters $\alpha_{train} \in \Lambda_1, \alpha_{eval} \in \Lambda_2, \alpha_{test} \in \Lambda_3$ originate from, and the motion capture data sets $\Omega_1, \Omega_2, \Omega_3$ with $j_{train}^{MOCAP} \in \Omega_1, j_{eval}^{MOCAP} \in \Omega_2, j_{test}^{MOCAP} \in \Omega_3$ are mutually disjoint each, i.e. it holds the following:

$$\Lambda_i \cap \Lambda_j = \emptyset \quad \forall i, j \in \{1, 2, 3\} \wedge i \neq j \tag{4.13}$$

$$\Omega_i \cap \Omega_j = \emptyset \quad \forall i, j \in \{1, 2, 3\} \wedge i \neq j \tag{4.14}$$



(a) Labeling with 200 vertices (b) Labeling with 1000 vertices (c) Labeling with 13000 vertices

Figure 4.6: Timing improvements

	Body Models	**Motion Capture Sequences**	**Overall tensor pairs (D,L)**
Training	*100 Humans* $H_m(\alpha_{train})$	*20* j^{MOCAP}_{train} *a 20 frames*	*40k*
Evaluation	*50 Humans* $H_m(\alpha_{eval})$	*10* j^{MOCAP}_{eval} *a 20 frames*	*10k*
Testing	*50 Humans* $H_m(\alpha_{test})$	*10* j^{MOCAP}_{test} *a 20 frames*	*10k*
Cumulative	*200 Humans*	*40 a 20 frames*	*60k*

Table 4.2: Data Set Composition

Chapter 5

Technical Approaches

For the formulated body part segmentation problem several approaches were incorporated. Point-based approaches, working directly on point clouds are explained in Section 5.2 and also volumetric approaches, that work on a volumetric representation of the point cloud, are presented in Section 5.1.

5.1 3D Convolutional Neural Networks

3D convolutions, as a natural extension to in image processing widely used 2D convolutions, are perfectly suited to 3-dimensional data and hence extensively utilized in the proposed methods.

5.1.1 Basic Building Blocks

The symbols for the basic building blocks of the utilized networks can be taken from figure 5.6 and will be explained in the following.

3D Convolution. If in context of deep learning someone is speaking of convolution, he means cross-correlation, which just differs in a flipped sign, which changes the

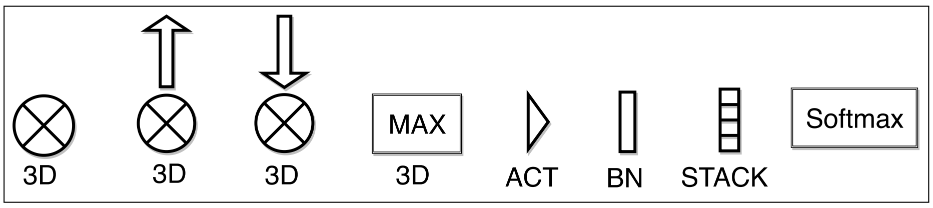


Figure 5.1: Basic Building Blocks from left to right: 3D Convolution, 3D Transposed Convolution, 3D Convolution with stride, 3D Maxpooling, Activation Function, Batch Normalization, Stacking

direction in which the filter slides over the input. The 3D discrete cross-correlation is given by the following equation:

$$g\left[-x,-y,-z\right] * h\left[x,y,z\right] = \sum_{k=0}^{K}\sum_{l=0}^{L}\sum_{m=0}^{M} h\left[k,l,m\right] g\left[x+k,y+l,z+m\right] \tag{5.1}$$

where g and h denote 3-dimensional discrete functions, that are cross-correlated. The cross-correlation is expressed as a convolution between h and a point reflected g. Hereby $*$ denotes the convolution operation. One of the signals g or h can be viewed as one channel of the feature map of the input and the other signal represents one filter learned by the neural network. The output for one channel of the output feature map is then defined by the element-wise sum over all input channels after application of (5.1).

3D Convolution with stride. The 3D convolution with stride is equivalent to first applying equation (5.1) and then sub-sampling the resulting signal by a certain factor, called stride(for filter size equal to the sampling factor). The factor used in the proposed methods is 2. Additionally the number of output channels is kept equal to the number of input channels.

3D Transposed Convolution. The transposed convolution, sometimes called De-Convolution is not the inverse operation to convolution with or without stride. It just reverses the downsizing effect of strided convolution, by expanding the signals again. The name transposed convolution stems from the fact that the 3D convolution can be expressed by a matrix multiplication between a matrix originating from the filter and an into a vector flattened input. The transposed convolution then just uses the transposed of this matrix to again enlarge the volume. For a better understanding, the 3D transposed convolution can be viewed as the 3D equivalent of what the Kronecker product is in 2D. For illustration the Kronecker product between two matrices is defined as:

$$A \otimes B = \begin{pmatrix} a_{11}B & \cdots & a_{1n}B \\ \vdots & \ddots & \vdots \\ a_{m1}B & \cdots & a_{mn}B \end{pmatrix} \tag{5.2}$$

thereby B would denote the filter matrix that is applied on a 2-dimensional input represented by the matrix A.

3D max-pooling. 3D max-pooling with filter size equal to the stride, splits a volume into sub-regions equal to the filter size. In these sub-regions just the maximum value is kept. We use filter size of 2.

Activation Functions. The activation function is a scalar function applied element-wise to a volume, introducing non-linearities, as already introduced in 2.4.1.

3D Batch Normalization. Batch normalization is the operation that normalizes each activation in a layer by first subtracting the sample mean and then dividing by the sample variance. As proposed by Ioffe et al.[29] for batch normalization in convolutional neural networks the mean and variance are calculated overall dimensions except the channel dimension of the corresponding tensor, which is passed into the batchnorm layer.

3D Stacking. The 3D stacking stacks two 4D tensors along the channel dimension.

3D Softmax. The softmax activation function is scaling the input between 0 and 1, producing an output that can be interpreted as a probability for a certain class. The softmax function is therby amplifying high values and weakening low value. That is why it is called the softmax function. The formula relating the input to the output of the softmax function is given by:

$$\sigma(\mathbf{z})_j = \frac{e^{z_j}}{\sum_{k=1}^{K} e^{z_k}} \quad j = 1, ..., K \tag{5.3}$$

Note that for a volumetric input the softmax is applied along the channel dimension.

5.1.2 V-net and variants

V-net[53] is a architecture used for prostate segmentation in MRT images. As segmentation of prostates is a binary segmentation problem, modifications were necessary to apply the network to our body part segmentation problem. Thus we first describe the baseline architecture as introduced in the paper and then the necessary modifications.

Baseline Architecture

The V-net as depicted in figure 5.2 is, in general, an encoder-decoder architecture. The left part of the model is the encoder path that compresses the input of 128x64x128 down to a resolution of 8x4x8. This is achieved through 3D convolutions with stride. On the right part there is the decoder path, that up-samples the volume back to a resolution of 128x64x128. The up-sampling is computed via transposed

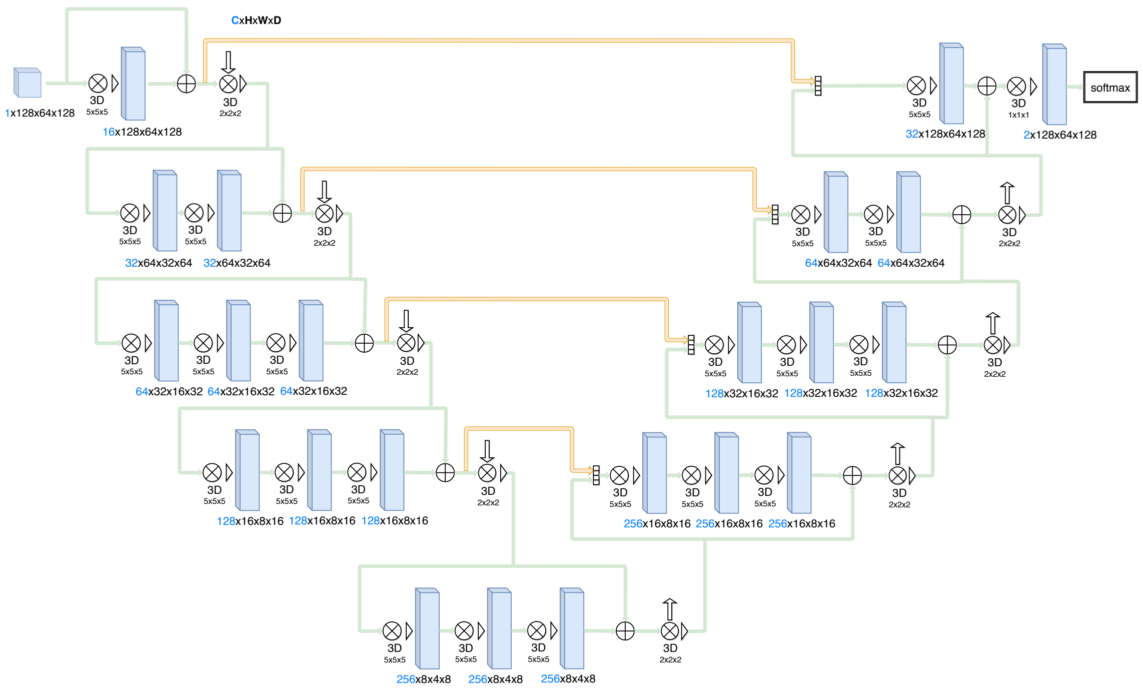


Figure 5.2: V-net Baseline Model for Binary Segmentation(∼70Mio par.)

convolutions. Thus each stage produces features at different resolutions, at both the compression and decompression path. Stages from the encoder and the decoder path are connected where they share the same resolutions, i.e. features from the encoder path are directly forwarded to the decoder path. These connections enable the network to ”skip” a further way down the network. This improves convergence due to short-cuts the gradient flow can take, while back-propagating through the network, as well as information from the feature maps of the compression path is provided, which would have been lost due to compression. The stages are formulated to learn residual functions by first forwarding the signal to several convolutional layers and then adding the unprocessed signal to the result of the convolutional layers. All convolutions use 5x5x5 filter kernels except the ones, used for down-sampling, and the one convolution at the output layer. The convolutions for down-sampling have a kernel of size 2x2x2 and the kernel at the output layer is 1x1x1. For all convolutions, zero-padding is used. As the filter in each convolution with stride is only applied once for each non-overlapping region of 2x2x2 the size of the feature maps is halved. The activation function used for the network is leakyReLU. For each class of the segmentation problem, a channel in the output layer is provided. Since for the prostate, the segmentation is binary, two channels are provided with subsequent softmax activation to scale the value into a range between 0 and 1.

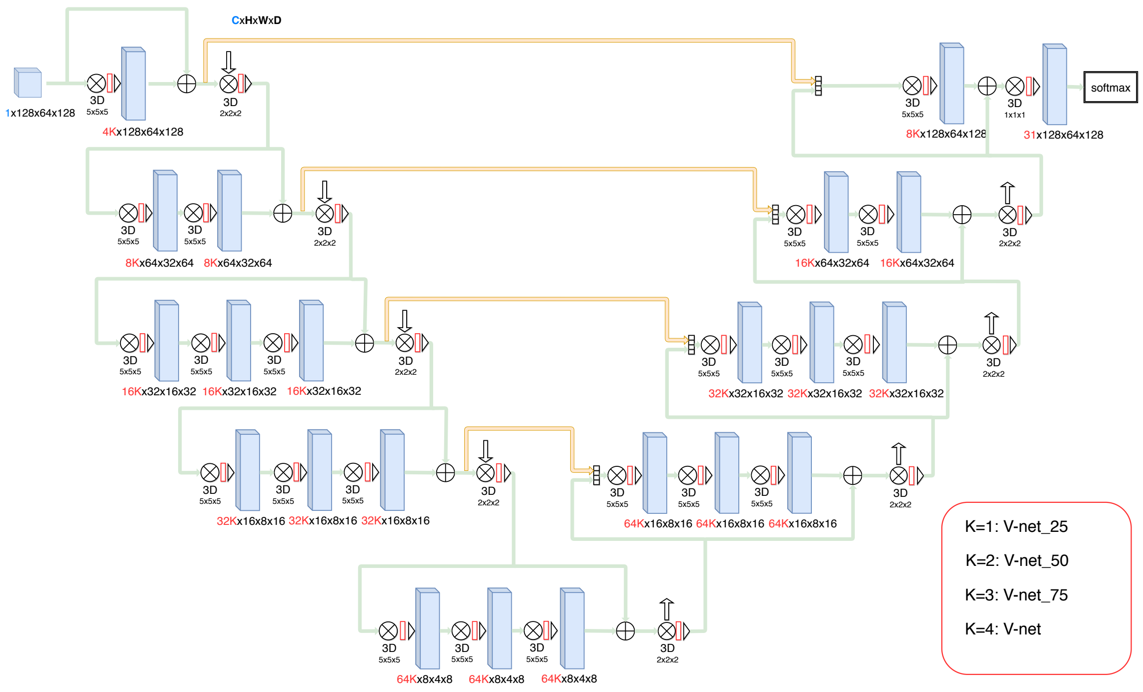

CxHxWxD
1x128x64x128
3D 5x5x5
4Kx128x64x128
3D 2x2x2
8Kx64x32x64 8Kx64x32x64
16Kx32x16x32 16Kx32x16x32 16Kx32x16x32
32Kx16x8x16 32Kx16x8x16 32Kx16x8x16
64Kx8x4x8 64Kx8x4x8 64Kx8x4x8
64Kx16x8x16 64Kx16x8x16 64Kx16x8x16
32Kx32x16x32 32Kx32x16x32 32Kx32x16x32
16Kx64x32x64 16Kx64x32x64
8Kx128x64x128
3D 1x1x1
31x128x64x128
softmax
K=1: V-net_25
K=2: V-net_50
K=3: V-net_75
K=4: V-net

Figure 5.3: V-net Model with Modifications (∼4.4Mio,∼17.8Mio,∼39.9Mio,∼70Mio and ∼3.3Mio par.)

Modified Architectures

To apply the network to our pose estimation problem, we developed different variants of the network and adapted where necessary. To handle 31 classes(30 body parts and 1 background class) the last layer was extended to have 31 output channels. Furthermore, each convolutional layer was extended with a batchnorm layer for faster convergence. Furthermore, several networks with 25, 50, 75 and 100 percent of the original number of channels were examined as can be seen in figure 5.3

Furthermore, a shallower version of V-net was implemented which is just a version for K=4, where after the second down-sampling stage, an up-sampling stage is following again. By doing so, we try to explore if 2 down-sampling stages are sufficient to still achieve a good performance of the network.

5.1.3 3D U-Net

Also coming from the medical domain, the 3D U-Net architecture proposed by Wang et al.[89] uses 3D convolutions extensively but was applied for a 3 class segmentation problem on a 3 Channel input. It builds on the 2D U-Net architecture by Ronneberger et al.[76] by replacing all 2D operations with their 3D counterparts. In particular, it uses 3D convolutions, 3D max-pooling and 3D transposed convolutions.

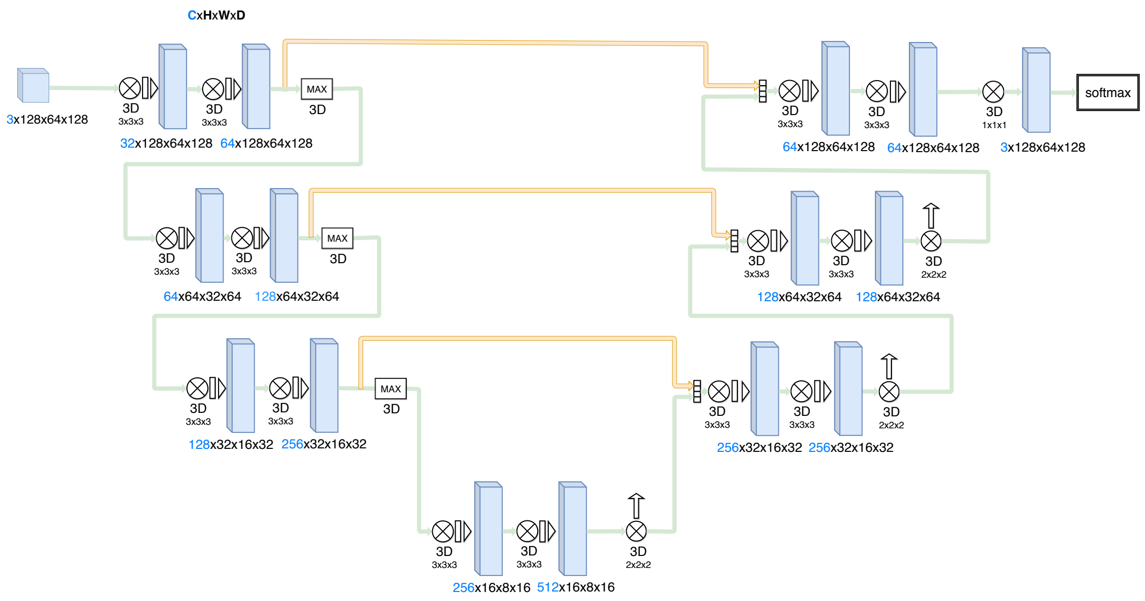


Figure 5.4: 3D U-Net Baseline Model (∼19 Mio par.)

Baseline Architecture

The baseline architecture as depicted in figure 5.4 consists as well as the V-net architecture of an encoder-decoder structure. But it disclaims to learn residual blocks. Furthermore, it uses 3D max-pooling instead of strided convolution and just contains 4 different stages with different resolutions compared to V-net with 5 stages. Each stage of the encoder path contains two 3x3x3 convolutional layers followed by a 2x2x2 max-pooling layer with stride 2, whereas each stage of the decoder path contains two 3x3x3 convolutional layers and one 2x2x2 transposed convolutional layer with stride 2. Batch normalization followed by the ReLU activation function is applied after each convolutional layer, except the last convolutional layer. Furthermore similar as in V-net skip-connections are provided between the encoder and decoder path at stages with identical resolution.

Modified Architecture

The modifications at the network can be taken from figure 5.5. The ReLU activations were replaced by leakyReLU and also additional batchnorm and activation layers were introduced after each transposed convolution. In order to cope with our body part segmentation class the output layer got 31 channels, one for each body part and background.

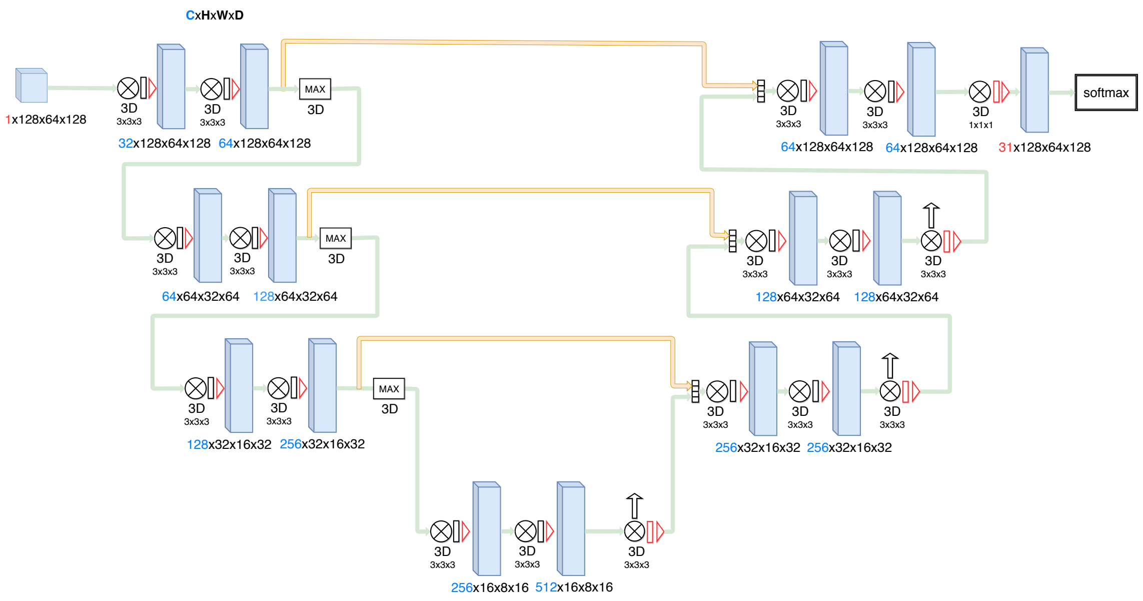


Figure 5.5: 3D U-Net Model with Modifications (∼19Mio par.)

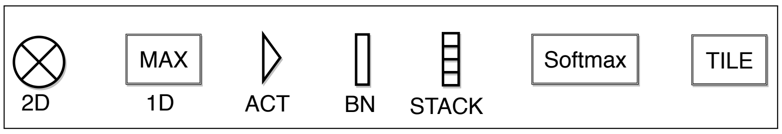


Figure 5.6: Basic Building Blocks from left to right: 2D Convolution, 1D Maxpooling, Activation Function, Batch Normalization, Stacking, Softmax, Tiling

5.2 Point Based Neural Networks

Instead of representing the space with a volumetric voxel grid, one can code the volume in a format that just points with a certain property are represented. In our case background points are ignored. Only points representing the human body surface are represented with their coordinate, hence we end up with a point cloud. To process such point clouds other forms of networks are necessary. The main challenge of such a network is that it has to deal with an arbitrary number of unordered points. In the following two architectures dealing with that challenge are explained. But first, the basic building blocks are described.

5.2.1 Basic Building Blocks

The symbols for the basic building blocks of the utilized networks can be taken from figure 5.6 and will be explained in the following.

2D Convolutions. Similar to the 3D convolution as introduced in Subsection 5.1.1

we can define the 2D convolution in deep learning as:

$$g\left[-x,-y\right]*h\left[x,y\right]=\sum_{k=0}^{K}\sum_{l=0}^{L}h\left[k,l\right]g\left[x+k,y+l\right] \tag{5.4}$$

1D max-pooling. The 1D max-pooling with filter size Nx1 takes the maximum value along the axis the feature vectors are stacked. Thereby it reduces a tensor of shape Nx1xK to 1x1xK.

Activation Functions. The activation function again is a scalar function applied element-wise to a volume, introducing non-linearities.

2D Batch Normalization. Batch normalization again is implemented as proposed in[29] with normalization overall dimensions except the channel dimension.

Tile. The tile block repeats the tensor along one dimension. In our case, it repeats it along the first dimension.

5.2.2 PointNet

PointNet[67], compared to other approaches, is a network that can work directly on an irregular format such as a point cloud. It is invariant to permutation the input points and also can take an arbitrary number of points. The basic idea is to use a symmetric function on identically transformed elements of the point set, i.e. we get a overall function describing the network:

$$f\left(\{x_1,...,x_N\}\right)=g\left(h\left(x_1\right),...,h\left(x_N\right)\right), \tag{5.5}$$

where $\{x_1,...,x_N\}$ is the point set and g is a composition of a single variable function and a symmetric function, which was chosen to be the max-pooling function. The function h is realized with a series of layers with 2D convolutions and subsequent activation function. In this way, a stack of MLPs is realized that takes the point cloud as an input and outputs a feature vector for each point. The MPLs all share the same weights hence our function h is one MLP itself.

The original network was applied on data sets for semantic segmentation on indoor scenes to classify a room into its parts like furniture, chair, floor, table, etc. and also on part segmentation problems for diverse rigid objects such as bags, laptops, airplanes, etc.

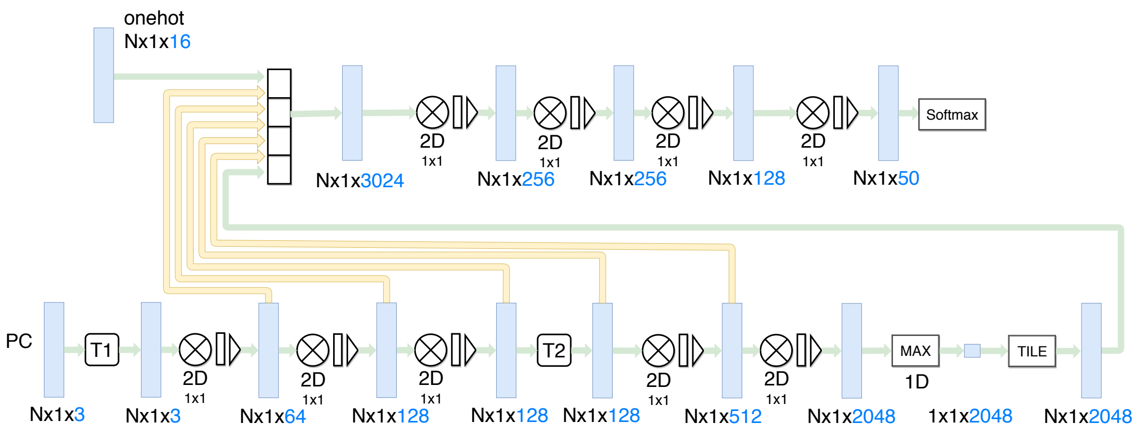


Figure 5.7: PointNet Baseline (∼2Mio par. without T1 and T2)

Baseline Architecture

The baseline architecture (see figure 5.7) contains five 1x1 2D convolutional layers with subsequent batch normalization and activation function at the first stage until the max-pooling layer. On the input the transformation T1 and after the third convolutional layer the transformation T2 is applied to achieve invariance of the features against rigid transformations of the objects that shall be segmented. At the end of the first stage, the 1D max-pooling is applied and the result is replicated for every point and stacked together with local point-wise features from earlier layers and a one-hot vector describing the object class. Then in the second stage, this Nx1x3024 tensor again is passed into a stack of MLPs realized through 2D convolutions and finally, a softmax activation is applied.

Modified Architecture

The modifications to the baseline model are the removal of transformation networks T1 and T2 as well as an additional convolutional layer. Also for our body part segmentation problem stacking a one-hot vector with an object category was not necessary as well as the last layer was adapted to have 31 output channels. Thus we end up the model as depicted in figure 5.8.

5.2.3 Residual PEL Network

A further architecture proposed by Li et al.[41] was utilized for the formulated segmentation task. Similar as PointNet the concept of permutation invariant layers(PEL) is used with the help of an MLP and 1D max-pooling. But instead of using very deep MLPs he introduces a layer just containing a single layer MLP stack, which can again be implemented through 2D convolution and max pooling layer. Furthermore instead of stacking he aims at learning a residual function by

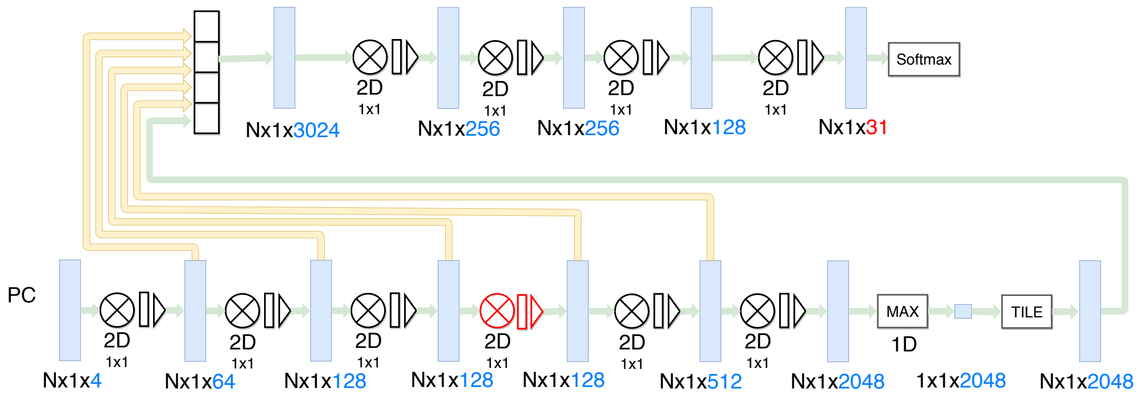


Figure 5.8: PointNet Modified (∼2Mio par.)

adding the unprocessed signal to the signal passed through max-pooling, tiling, 2D convolution and the activation function. With this layer, which he calls PEL layer, he builds a very deep network consisting of 39 PEL layers, which can be seen in figure 5.9.

Baseline Architecture

The baseline architecture, as depicted in figure 5.9, consists of 3 residual blocks with 64, 256 and 2048 channels at the output. Each residual block itself consists of 13 PEL layers, that contain as many channels as the output of the residual block has channels.

Modified Architecture

In order to be able to train the network on larger point clouds, the network was reduced by the last residual block compared to the baseline model. Furthermore, additional batchnormalization layers were introduced, one after each 2D convolution. Additionally, an output layer with a 2D convolution and softmax was added to end up with tensor of shape Nx1x31, which can be seen from figure 5.10.

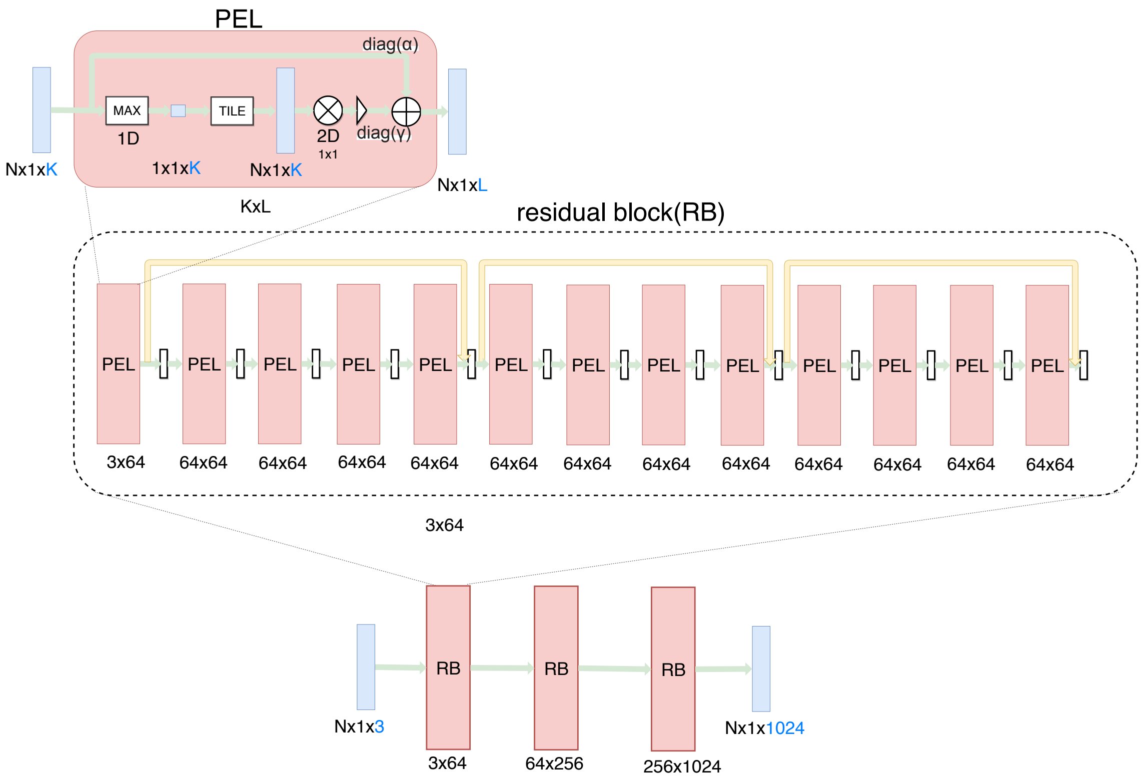


Figure 5.9: Residual PEL Network Baseline (∼13Mio par.)

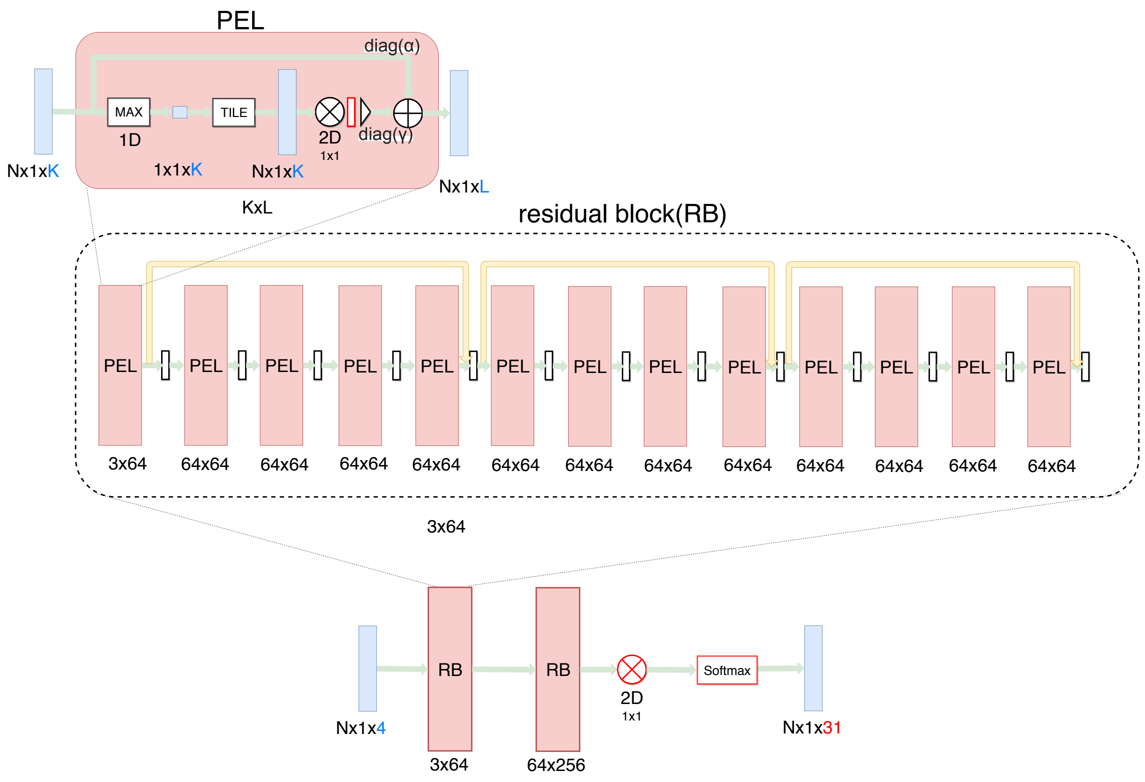


Figure 5.10: Residual PEL Network Modified (∼0.8Mio par.)

Chapter 6

Evaluation

6.1 Training

For training and evaluation of the neural networks, an NVidia Quadro P5000 with 16GB graphics memory was used. The time measurements were performed with an NVidia GTX 1070. The parameters for training can be taken from table 6.1. To optimize the networks a cross-entropy loss was applied to the outputs of the networks, described in Section 5. Furthermore, the data set used for training can be taken from table 4.2 in Chapter 4. Note, that to make training possible for all networks in terms of graphics memory, the original resolution of 512x256x512 was reduced to 128x64x128 by a simple downsampling procedure. This lower resolution was then used to train all networks. Also, random shuffling of the data set was applied for training.

Parameters	**V-net**	**3D U-Net**	**PointNet**	**Residual PEL Network**
initial learning rate	0.05	0.05	0.0005	0.0001
optimizer	SGD	SGD	SGD	SGD
learning rate decay	0.7	0.7	0.7	0.7
learning rate decay step	5000	5000	5000	5000
momentum	0.99	0.99	0.99	0.99
batch size	4	1	16	16
points for training	10k	10k	10k	10k
resolution	128x64x128	128x64x128	128x64x128	128x64x128

Table 6.1: Training Parameters

6.2 Evaluation Measures

To evaluate the performance of the investigated networks on the segmentation task classical evaluation measures for segmentation were applied such as the true positive rate (TPR) (6.4) and the dice score (6.2).

$$TPR = \frac{TP}{TP + FN} \tag{6.1}$$

$$F_1 = \frac{2TP}{2TP + FP + FN} \tag{6.2}$$

Hereby the TP, TN, FP and FN denote the true positives, true negatives, false positives and false negatives respectively.
The macro average TPR for class K calculates to:

$$TPR^K_{avg} = \frac{1}{N}\sum_{n=1}^{N} TPR^K_n \tag{6.3}$$

where N denotes the number of samples and TPR^K_n the true positive rate for class K of a sample with index n . Similarly the macro average dice score is defined:

$$F^K_{1,avg} = \frac{1}{N}\sum_{n=1}^{N} F^K_{1,n} \tag{6.4}$$

In addition to these performance measures of the segmentation the inference time was measured for all network types as well.

6.3 Quantitative Results

After training, the proposed networks were evaluated on the test set according to the table 4. In figure 6.1 one can see the distribution of the macro average TPR over the classes for all proposed networks, achieved on the test set. In terms of macro average TPR, the best network is the shallow version of the V-net with a mean TPR of 0.943. But all V-net architectures perform quite similar. The 3D U-net and PoinNet lie a little behind with around 0.90 and 0.89 TPR. The modified residual PEL architecture seems to be the bottom of the proposed architectures. The F_1-score gives an identical ranking as the TPR in terms of segmentation performance as can be seen from table 6.2 and figure 6.2
Also for measurements of the inference time, one can take a look at table 6.2. It has to be noted that the point-based approaches outperform the volumetric approaches significantly.

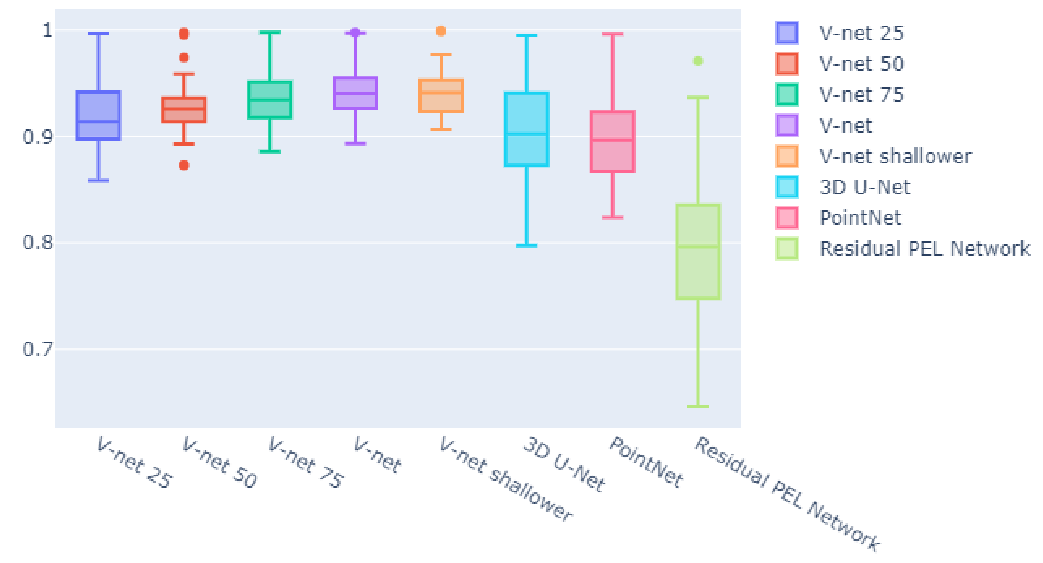


Figure 6.1: Macro average TPR distribution over classes

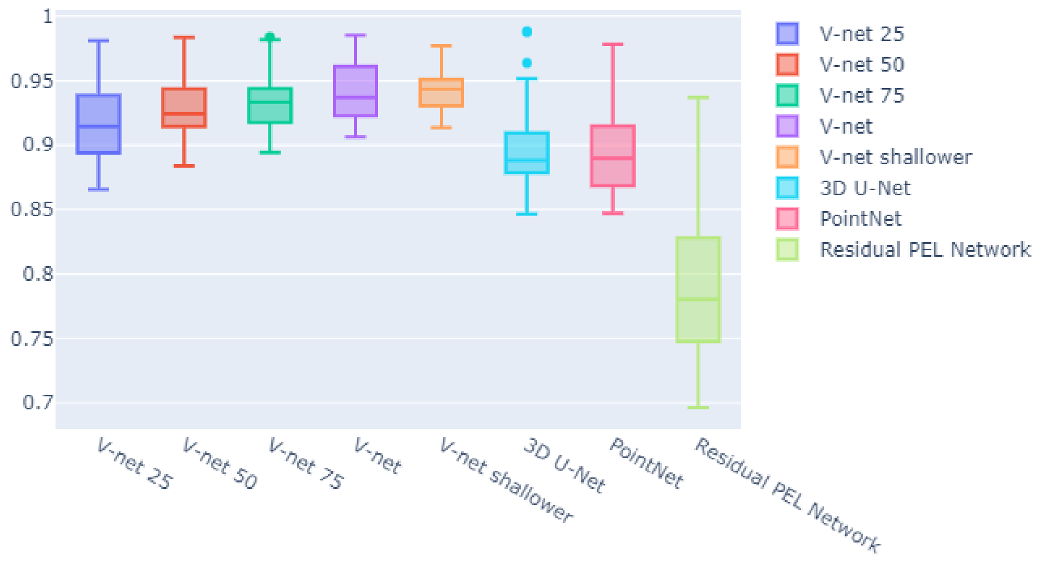


Figure 6.2: Macro average dice score distribution over classes

Measures	**V-net (25/50/75/100/Shallow)**	**3D U-Net**	**PointNet**	**Residual PEL Network**
mean TPR^{K}_{avg}	0.919/0.928/0.934/0.941/0.943	0.902	0.897	0.794
mean $F^{K}_{1,avg}$	0.918/0.927/0.933/0.940/0.943	0.900	0.895	0.790
inference time(ms)	∼83/∼222/∼400/∼544/∼430	∼719	∼18	∼25

Table 6.2: Measurements

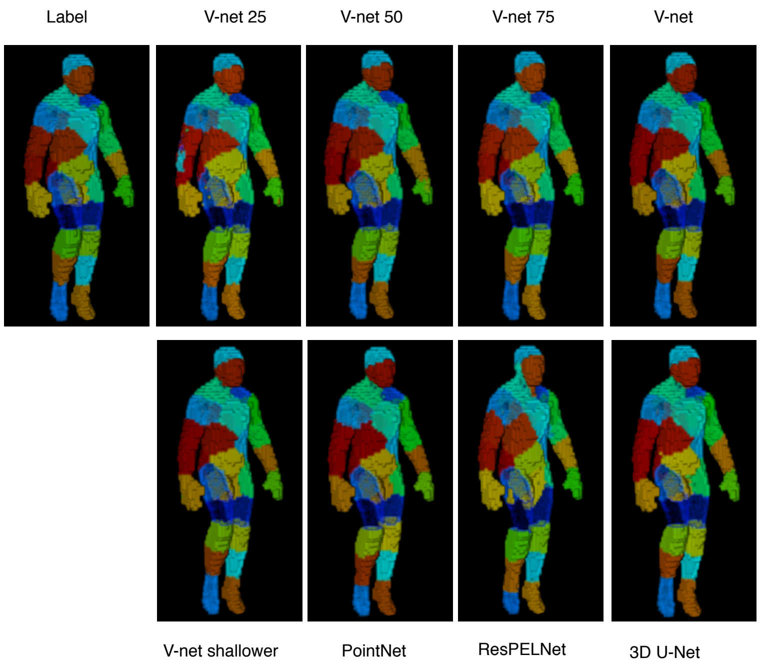


Figure 6.3: Visualization of Predictions and Label Image 1

6.4 Visual Results

In order to gain a visual impression of the performance of the networks several predictions with their corresponding ground truth are provided in figure 6.3-6.5. It can be seen that images show visible errors for the V-net 25 and both point-based approaches. Furthermore, a difference between the point-based networks and the 3D convolutional networks has to be noted. The point-based networks tend to keep the topology the same, whereas speckle occurs at some positions at the convolutional networks, e.g. at the V-net 25.

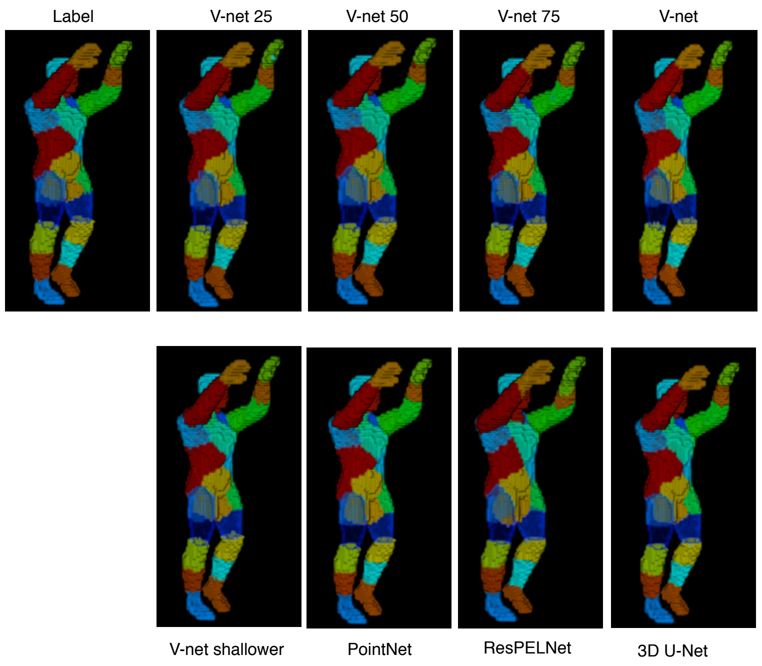


Figure 6.4: Visualization of Predictions and Label Image 2

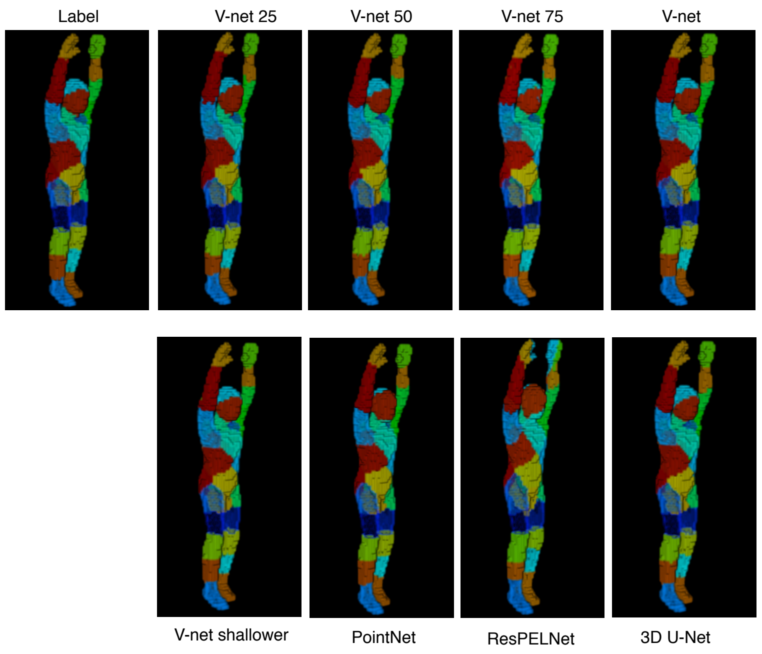


Figure 6.5: Visualization of Predictions and Label Image 3

Chapter 7

Discussion

As can be seen from the results all networks achieve quite good TPR/F1-score on the test set, both measures are around 0.9 and above for all networks except the second point based architecture. That's why they are suitable for the formulated segmentation task. Furthermore from the fact that the performance measures for the V-net decrease when the number of filters is reduced, but even increases when the network is built shallower, we can conclude, that we can reduce the number of parameters significantly for the formulated body part segmentation task. But instead of reducing the number of filters it is more beneficial to build the network shallower to achieve less complexity. The fact that a shallower network performs well, also makes sense from the viewpoint of scales, since people usually never differ in height more than in a factor of 4 (baby with 50 cm compared to a full-grown man of 200cm height). Hence two down-sampling steps are sufficient. As a further result we can conclude, that it is beneficial to learn residual functions in 3D networks since V-net outperforms the 3D U-Net slightly. The fact that PointNet outperforms the residual PEL network significantly has the following reason. The residual PEL network was built to perform on a lower number of points compared to 10k points. As more points contain more information about the scene, the compression bottleneck should be wider. This compression bottleneck is given by the length of the feature vector after the max-pooling layer and the length of this vector is crucial for a good segmentation performance, as also stated in [67] for general permutation invariant layers. Due to the fact that we reduced the length of this feature vector from 2048 to 64, we end up with a worse segmentation performance. As can be seen from the presented timing measurements the point-based approaches outperform the 3D CNN approaches. That can be explained from the fact that the point-based approaches act on a more compact representation of the volume and hence need fewer computations for inference.

Chapter 8

Conclusion

8.1 Summary and Findings

To sum things up, as a first step we approached the formulated problem statement as body part segmentation problem, which was inspired by extensive literature research in human pose estimation. To cope with the problem of having no suitable public data set or appropriate labels for real-world body scans, we developed a fully automatic pipeline for data set synthesis based on classical animation methods. Furthermore, we approached the problem of having no labels by developing a method for generating rich ground truth for volumetric rendered dense point clouds for an arbitrary number of body parts. Then there is the data set itself which was built by defining a canonical avatar, sampling persons from the body model, recording MOCAP data and finally feeding that data into the synthesis pipeline. Thereby we also defined a further restriction on the data set, namely being textureless. Learning on textureless data enables us to directly apply those methods on real-world body scans after binarization.
For approaching the problem of correspondence estimation itself, we combined inspirations like the body part problem formulation and network architectures from various domains such as the medical domain, the general segmentation domain, and the human pose estimation domain.
To conclude we propose several methods for the body part segmentation problem. Whereas the point-based approaches are superior in terms of timing, the 3D CNNs outperform point base approaches in precision. In the context of the described body scanners real-time approaches are preferred and hence the point-based approaches are more suitable for that application.

8.2 Limitations

The proposed data set is limited by the realism of the animations. i.e. it does not include real word soft-tissue deformations as the data sets like FAUST[7], *DYNA*[65]

and *D-FAUST*[8]. Also, MOCAP data was only captured with Kinect, and not a professional motion capture setup. Furthermore, no information from the texture space is used, which might be beneficial for the correspondence estimation.
Since these limitations affect the data set they also affect the data-driven models and the performance on real-world scans.
Furthermore, from a conceptional point of view only rough correspondences are provided due to the body part segmentation approach. Nevertheless, by increasing the number of body parts the correspondences become more dense.

8.3 Future Work

To address the shortcomings of realism one could try to incorporate the SMPL model if it comes to non-commercial use or try to register meshes on real-world body scans from the imaging system. This both would increase the realism of the data set. Furthermore, a professional motion capture setup could be considered for animation to gain more naturally looking skeleton motions without jitter.
To address the dense correspondence limitation one could build on work from [15, 36], which both try to establish shape correspondences. For establishing dense correspondences also UV-maps as utilized from Güler et al.[72] could be used to estimate correspondences or a direct estimation of an underlying mesh could be targeted.
Finally, future work could go into the direction of the incorporation of texture, which is provided by the body scans. This, for example, could be done via designing a CycleGANs[99] for mapping between synthesized data and real data or via registration of meshes onto real-world scans.

Appendix A

Electromagnetics

A.1 Maxwell's Equations in differential form

The following equations, called Maxwell's equations, describe the basic electrical field relations in a differential form.

$$\nabla \cdot \mathbf{E} = \frac{\rho}{\varepsilon_0} \tag{A.1}$$

$$\nabla \cdot \mathbf{B} = 0 \tag{A.2}$$

$$\nabla \times \mathbf{E} = -\frac{\partial \mathbf{B}}{\partial t} \tag{A.3}$$

$$\nabla \times \mathbf{B} = \mu_0 \left(\mathbf{J} + \varepsilon_0 \frac{\partial \mathbf{E}}{\partial t} \right) \tag{A.4}$$

List of Figures

Acronyms and Notations

2D Two-Dimensional

3D Three-Dimensional

CNN Convolutional Neural Network

CycleGAN Cycle Generativ Adversarial Network

D-FAUST Dynamic Fine Alignment Using Scan Texture

FAUST Fine Alignment Using Scan Texture

GAN Generativ Adversarial Network

FN False Negatives

FP False Positive

MLP Maximum Intensity Projection

MLP Multi-Layer Perceptron

MOCAP Motion Capture

PEL Permutation Invariant Layer

PReLU Parametric Rectified Linear Unit

ReLU Rectified Linear Unit

RBF Radial Basis Function

ReLU Rectified Linear Unit

RGB Red-Green-Blue(color model)

RGB-D Red-Green-Blue-Depth

SGD Stochastic Gradient Descent

SMPL	Skinned Multi-Person Linear Model
TN	True Negatives
TP	True Positive
TPR	True Positive Rate

Bibliography

[1] Sherif Sayed Ahmed, Andreas Schiessl, Frank Gumbmann, Marc Tiebout, Sebastian Methfessel, and Lorenz-Peter Schmidt. Advanced microwave imaging. 13(6):26–43, 2012.

[2] Alejandro Newell, Kaiyu Yang, and Jia Deng. Stacked hourglass networks for human pose estimation. 2016.

[3] Andrei Zanfir, Elisabeta Marinoiu, and Cristian Sminchisescu. Monocular 3d pose and shape estimation of multiple people in natural scenes. 2018.

[4] Mykhaylo Andriluka, Leonid Pishchulin, Peter Gehler, and Bernt Schiele. 2d human pose estimation: New benchmark and state of the art analysis. In *2014 IEEE Conference on Computer Vision and Pattern Recognition*, pages 3686–3693. IEEE, 23.06.2014 - 28.06.2014.

[5] Dragomir Anguelov, Praveen Srinivasan, Daphne Koller, Sebastian Thrun, Jim Rodgers, and James Davis. Scape. 24(3):408, 2005.

[6] Vasileios Belagiannis. Human pose estimation in complex environments, 2015.

[7] Federica Bogo, Javier Romero, Matthew Loper, and Michael J. Black. Faust: Dataset and evaluation for 3d mesh registration. In *2014 IEEE Conference on Computer Vision and Pattern Recognition*, pages 3794–3801. IEEE, 23.06.2014 - 28.06.2014.

[8] Federica Bogo, Javier Romero, Gerard Pons-Moll, and Michael J. Black. Dynamic faust: Registering human bodies in motion. In *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 5573–5582. IEEE, 21.07.2017 - 26.07.2017.

[9] Leyde Briceno and Gunther Paul. Makehuman: A review of the modelling framework. In Sebastiano Bagnara, Riccardo Tartaglia, Sara Albolino, Thomas Alexander, and Yushi Fujita, editors, *Proceedings of the 20th Congress of the International Ergonomics Association (IEA 2018)*, volume 822 of *Advances in Intelligent Systems and Computing*, pages 224–232. Springer International Publishing, 2019.

[10] Alexander M. Bronstein, Michael M. Bronstein, and Ron Kimmel. *Numerical geometry of non-rigid shapes*. Monographs in computer science. Springer, 2008.

[11] James Charles, Tomas Pfister, Derek Magee, David Hogg, and Andrew Zisserman. Personalizing human video pose estimation, 20.11.2015. CVPR 2016.

[12] Wenzheng Chen, Huan Wang, Yangyan Li, Hao Su, Zhenhua Wang, Changhe Tu, Dani Lischinski, Daniel Cohen-Or, and Baoquan Chen. Synthesizing training images for boosting human 3d pose estimation, 10.04.2016.

[13] Christian Zimmernmann, Tim Welschehold, Christian Dornhege, and Wolfram Burghard und Thomas Brox. 3d human pose estimation in rgbd images for robotic task learning. 2018.

[14] George Cybenko. Approximation by superpositions of a sigmoidal function. 2(4):303–314, 1989.

[15] Davide Boscaini, Jonathan Masci, Emanuele Rodolà, and Michael Bronstein. Learning shape correspondence with anisotropic convolutional neural networks. 2016.

[16] Alexey Dosovitskiy, Philipp Fischer, Eddy Ilg, Philip Hausser, Caner Hazirbas, Vladimir Golkov, Patrick van der Smagt, Daniel Cremers, and Thomas Brox. Flownet: Learning optical flow with convolutional networks. In *2015 IEEE International Conference on Computer Vision (ICCV)*, pages 2758–2766. IEEE, 07.12.2015 - 13.12.2015.

[17] Yu Du, Yongkang Wong, Yonghao Liu, Feilin Han, Yilin Gui, Zhen Wang, Mohan Kankanhalli, and Weidong Geng. Marker-less 3d human motion capture with monocular image sequence and height-maps. In Bastian Leibe, Jiri Matas, Nicu Sebe, and Max Welling, editors, *Computer vision - ECCV 2016*, volume 9908 of *Lecture Notes in Computer Science*, pages 20–36. Springer, 2016.

[18] Bora Erden, Noah Gamboa, and Sam Wood. 3d convolutional neural network for brain tumor segmentation. 2017.

[19] Sean Ryan Fanello, Tim Paek, Cem Keskin, Shahram Izadi, Pushmeet Kohli, David Kim, David Sweeney, Antonio Criminisi, Jamie Shotton, and Sing Bing Kang. Learning to be a depth camera for close-range human capture and interaction. 33(4):1–11, 2014.

[20] David Forsyth, Philip Torr, Andrew Zisserman, Ryuzo Okada, and Stefano Soatto, editors. *Relevant Feature Selection for Human Pose Estimation and Localization in Cluttered Images*. Springer Berlin Heidelberg, 2008.

[21] Frederica Bogo, Angjoo Kanazawa, Christoph Lassner, Peter Gehler, Javier Romero, and Michael J. Black. Keep it smpl: Automatic estimation of 3d human pose and shape from a single image. 2016.

[22] Gerard Pons Moll. Dissertation gerard pons moll. 2014.

[23] Gerard Pons-Moll, Jonathan Taylor, Jamie Shotton, Aaron Hertzmann, and Andrew Fitzgibbon. Metric regression forests for correspondence estimation. 2015.

[24] Mona Fathollahi Ghezelghieh, Rangachar Kasturi, and Sudeep Sarkar. Learning camera viewpoint using cnn to improve 3d body pose estimation, 18.09.2016. To appear at the International Conference on 3D Vision (3DV), 2016.

[25] Thibault Groueix, Matthew Fisher, Vladimir G. Kim, Bryan C. Russell, and Mathieu Aubry. Atlasnet: A papier-mâché approach to learning 3d surface generation, 15.02.2018.

[26] Peng Guan, Alexander Weiss, Alexandru O. Balan, and Michael J. Black. Estimating human shape and pose from a single image. In *2009 IEEE 12th International Conference on Computer Vision*, pages 1381–1388. IEEE, 29.09.2009 - 02.10.2009.

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

[28] Mir Rayat Imtiaz Hossain and James J. Little. Exploiting temporal information for 3d pose estimation. 11214(7):69–86, 2018.

[29] Sergey Ioffe and Christian Szegedy. Batch normalization: Accelerating deep network training by reducing internal covariate shift, 2/11/2015.

[30] Catalin Ionescu, Dragos Papava, Vlad Olaru, and Cristian Sminchisescu. Human3.6m: Large scale datasets and predictive methods for 3d human sensing in natural environments. 36(7):1325–1339, 2014. Journal Article Research Support, Non-U.S. Gov't.

[31] Jamie Shotton, Andrew Fitzgibbon, Mat Cook, Toby Sharp, Mark Finocchio, Richard Moore, Alex Kipman, and Andrew Blake. Real-time human pose recognition in parts from single depth images. 2011.

[32] Jinlong Yang, Jean-Sébastien Franco, Franck Hétroy-Wheeler, and Stefanie Wuhrer. Estimation of human body shape in motion with wide clothing. 2016.

[33] Jonathan Masci, Davide Boscaini, Michael M. Bronstein, and Pierre Vandergheynst. Geodesic convolutional neural networks on riemannian manifolds. 2015.

[34] Jonathan Taylor, Jamie Shotton, Toby Sharp, and Andrew Fitzgibbon. The vitruvian manifold: Inferring dense correspondences for one-shot human pose estimation. 2012.

[35] Evangelos Kalogerakis, Aaron Hertzmann, and Karan Singh. Learning 3d mesh segmentation and labeling. 29(4):1, 2010.

[36] Vladimir G. Kim, Yaron Lipman, and Thomas Funkhouser. Blended intrinsic maps. 30(4):1, 2011.

[37] Christoph Lassner, Javier Romero, Martin Kiefel, Federica Bogo, Michael J. Black, and Peter V. Gehler. Unite the people: Closing the loop between 3d and 2d human representations, 10.01.2017.

[38] Leonid Pishchulin, Arjun Jain, Christian Wojek, Mykhaylo Andriluka, Thorsten Thormählen, and Bernt Schiele. Learning people detection models from few training samples. 2011.

[39] Leonid Pishchulin, Arjun Jain, Mykhaylo Andriluka, Thorsten Thormählen, and Bernt Schiele. Articulated people detection and pose estimation: Reshaping the future. 2012.

[40] Leonid Pishchulin, Arjun Jain, Mykhaylo Andriluka, Thorsten Thormählen, and Bernt Schiele. Articulated people detection and pose estimation: Reshaping the future. 2012.

[41] Shile Li and Dongheui Lee. Point-to-pose voting based hand pose estimation using residual permutation equivariant layer. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 11927–11936, 2019.

[42] Lingyu Wei, Qixing Huang, Duygu Ceylan, Etienne Vouga, and Hao Li. Dense human body correspondences using convolutional networks. 2016.

[43] Bastioni Manuel, Re Simone, and Misra Shakti. Ideas and methods for modeling 3d human figures. 2008.

[44] Mao Ye, Xianwang Wang, Ruigang Yang, Liu Ren, and Marc Pollefeys. Accurate 3d pose estimation from a single depth image. 2011.

[45] Javier Marin, David Vazquez, David Geronimo, and Antonio M. Lopez. Learning appearance in virtual scenarios for pedestrian detection. In *2010 IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, pages 137–144. IEEE, 13.06.2010 - 18.06.2010.

[46] Markus Oberweger, Paul Wohlhart, and Vincent Lepetit. Training a feedback loop for hand pose estimation. 2015.

[47] Enrique Martinez-Berti, Antonio José Sánchez Salmerón, Carlos Ricolfe Viala, Oliver Nina, and Mubarak Shah. Human pose estimation for rgbd imagery with multi-channel mixture of parts and kinematic constraints. 15:279–286, 2016.

[48] Matthew Loper, Naureen Mahmood, Javier Romero, Gerard Pons-Moll, and Michael J. Black. Smpl: A skinned multi-person linear model. 2015.

[49] Daniel Maturana and Sebastian Scherer. Voxnet: A 3d convolutional neural network for real-time object recognition. In *2015 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 922–928. IEEE, 28.09.2015 - 02.10.2015.

[50] Dushyant Mehta, Oleksandr Sotnychenko, Franziska Mueller, Weipeng Xu, Srinath Sridhar, Gerard Pons-Moll, and Christian Theobalt. Single-shot multi-person 3d pose estimation from monocular rgb. In *2018 International Conference on 3D Vision (3DV)*, pages 120–130. IEEE, 2018.

[51] Dushyant Mehta, Srinath Sridhar, Oleksandr Sotnychenko, Helge Rhodin, Mohammad Shafiei, Hans-Peter Seidel, Weipeng Xu, Dan Casas, and Christian Theobalt. Vnect: Real-time 3d human pose estimation with a single rgb camera. 36(4):44, 2017.

[52] Damien Michel, Ammar Qammaz, and Antonis A. Argyros. Markerless 3d human pose estimation and tracking based on rgbd cameras. In Unknown, editor, *Proceedings of the 10th International Conference on PErvasive Technologies Related to Assistive Environments - PETRA '17*, pages 115–122. ACM Press, 2017.

[53] Fausto Milletari, Nassir Navab, and Seyed-Ahmad Ahmadi. V-net: Fully convolutional neural networks for volumetric medical image segmentation. In *2016 Fourth International Conference on 3D Vision (3DV)*, pages 565–571. IEEE, 25.10.2016 - 28.10.2016.

[54] Mir Rayat Imtiaz Hossain. Understanding the sources of error for 3d human pose estimation from monocular images and videos. 2017.

[55] Thomas B. Moeslund, Adrian Hilton, and Volker Krüger. A survey of advances in vision-based human motion capture and analysis. 104(2-3):90–126, 2006. PII: S1077314206001263.

[56] Gyeongsik Moon, Ju Yong Chang, and Kyoung Mu Lee. V2v-posenet: Voxel-to-voxel prediction network for accurate 3d hand and human pose estimation

from a single depth map. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 5079–5088, 2018.

[57] Natalia Neverova, Christian Wolf, Florian Nebout, and Graham Taylor. Hand pose estimation through semi-supervised and weakly-supervised learning, 2017. 13 pages, 10 figures, 4 tables.

[58] Xuecheng Nie, Jianfeng Zhang, Shuicheng Yan, and Jiashi Feng. Single-stage multi-person pose machines, 25.08.2019. To appear in ICCV 2019.

[59] Michael A. Nielsen. *Neural networks and deep learning*, volume 25. Determination press San Francisco, CA, USA, 2015.

[60] Markus Oberweger, Paul Wohlhart, and Vincent Lepetit. Hands deep in deep learning for hand pose estimation. 2015. In Proceedings of 20th Computer Vision Winter Workshop (CVWW) 2015, pp. 21-30 added link to source https://github.com/moberweger/deep-prior.

[61] Maks Ovsjanikov, Mirela Ben-Chen, Justin Solomon, Adrian Butscher, and Leonidas Guibas. Functional maps: a flexible representation of maps between shapes. 31(4):30, 2012.

[62] Maks Ovsjanikov, Quentin Mérigot, Facundo Mémoli, and Leonidas Guibas. One point isometric matching with the heat kernel. 29(5):1555–1564, 2010.

[63] Xingchao Peng, Baochen Sun, Karim Ali, and Kate Saenko. Learning deep object detectors from 3d models, 22.12.2014.

[64] Xavier Perez-Sala, Sergio Escalera, Cecilio Angulo, and Jordi Gonzàlez. A survey on model based approaches for 2d and 3d visual human pose recovery. 14(3):4189–4210, 2014. Journal Article Research Support, Non-U.S. Gov't Review.

[65] Gerard Pons-Moll, Javier Romero, Naureen Mahmood, and Michael J. Black. Dyna. 34(4):120:1–120:14, 2015.

[66] Ronald Poppe. Vision-based human motion analysis: An overview. 108(1-2):4–18, 2007. PII: S1077314206002293.

[67] Charles R. Qi, Hao Su, Kaichun Mo, and Leonidas J. Guibas. Pointnet: Deep learning on point sets for 3d classification and segmentation, 02.12.2016. CVPR 2017.

[68] Charles R. Qi, Li Yi, Hao Su, and Leonidas J. Guibas. Pointnet++: Deep hierarchical feature learning on point sets in a metric space, 08.06.2017.

[69] Hossein Rahmani and Ajmal Mian. Learning a non-linear knowledge transfer model for cross-view action recognition. In *2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 2458–2466. IEEE, 07.06.2015 - 12.06.2015.

[70] Hossein Rahmani and Ajmal Mian. 3d action recognition from novel viewpoints. pages 1506–1515, 2016.

[71] Riza Alp Guler, George Trigeorgis, Epameinondas Antonakos, Patrick Snape, Stefanos Zafeiriou, and Iasonas Kokkinos. Densereg: Fully convolutional dense shape regression in-the-wild. 2017.

[72] Riza Alp Guler, Natalia Neverova, and Iasonas Kokkinos. Densepose: Dense human pose estimation in the wild. 2018.

[73] K. M. Robinette, H. Daanen, and E. Paquet. The caesar project: a 3-d surface anthropometry survey. In *Second International Conference on 3-D Digital Imaging and Modeling (Cat. No.PR00062)*, pages 380–386. IEEE Comput. Soc, 4-8 Oct. 1999.

[74] Grégory Rogez and Cordelia Schmid. Mocap-guided data augmentation for 3d pose estimation in the wild, 07.07.2016. 9 pages, accepted to appear in NIPS 2016.

[75] Javier Romero, Matthew Loper, and Michael J. Black. Flowcap: 2d human pose from optical flow. In Juergen Gall, Peter Gehler, and Bastian Leibe, editors, *Pattern Recognition*, volume 9358 of *Lecture Notes in Computer Science*, pages 412–423. Springer International Publishing, 2015.

[76] Olaf Ronneberger, Philipp Fischer, and Thomas Brox. U-net: Convolutional networks for biomedical image segmentation, 18.05.2015. conditionally accepted at MICCAI 2015.

[77] Nikolaos Sarafianos, Bogdan Boteanu, Bogdan Ionescu, and Ioannis A. Kakadiaris. 3d human pose estimation: A review of the literature and analysis of covariates. 152:1–20, 2016. PII: S1077314216301369.

[78] István Sárándi, Timm Linder, Kai O. Arras, and Bastian Leibe. How robust is 3d human pose estimation to occlusion?, 28.08.2018. Accepted for IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS'18) - Workshop on Robotic Co-workers 4.0: Human Safety and Comfort in Human-Robot Interactive Social Environments.

[79] Sherif Sayed Aboelyazeed Ahmed. Electronic microwave imaging with planar multistatic arrays. 2013.

[80] Shihong Xia, Zihao Zhang, and Le Su. Cascaded 3d full-body pose regression from single depth image at 100 fps. 2018.

[81] Leonid Sigal. Human pose estimation. pages 362–370, 2014.

[82] Leonid Sigal, Alexandru O. Balan, and Michael J. Black. Humaneva: Synchronized video and motion capture dataset and baseline algorithm for evaluation of articulated human motion. 87(1-2):4–27, 2010. PII: 273.

[83] C. Sminchisescu, A. Kanaujia, and D. Metaxas. Learning joint top-down and bottom-up processes for 3d visual inference. In *2006 IEEE Computer Society Conference on Computer Vision and Pattern Recognition - Volume 2 (CVPR'06)*, pages 1743–1752. IEEE, 17-22 June 2006.

[84] Hao Su, Charles R. Qi, Yangyan Li, and Leonidas J. Guibas. Render for cnn: Viewpoint estimation in images using cnns trained with rendered 3d model views. In *2015 IEEE International Conference on Computer Vision (ICCV)*, pages 2686–2694. IEEE, 07.12.2015 - 13.12.2015.

[85] Tamal K. Dey, Bo Fu, Huamin Wang, and Lei Wang. Automatic posing of a meshed human model using point clouds. 2015.

[86] Timo von Marcard, Roberto Henschel, Michael Black, Bodo Rosenhahn, and Gerard Pons-Moll. Recovering accurate 3d human pose in the wild using imus and a moving camera. 2018.

[87] Gül Varol, Javier Romero, Xavier Martin, Naureen Mahmood, Michael J. Black, Ivan Laptev, and Cordelia Schmid. Learning from synthetic humans. pages 4627–4635, 2017. Appears in: 2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR 2017). 9 pages.

[88] Varun Ganapathi, Christian Plagemann, Daphne Koller, and Sebastian Thrun. Real time motion capture using a single time-of-flight camera. 2010.

[89] Chengjia Wang, Tom MacGillivray, Gillian Macnaught, Guang Yang, and David Newby. A two-stage 3d unet framework for multi-class segmentation on full resolution image, 12.04.2018.

[90] Luyang Wang, Yan Chen, Zhenhua Guo, Keyuan Qian, Mude Lin, Hongsheng Li, and Jimmy S. Ren. Generalizing monocular 3d human pose estimation in the wild, 11.04.2019.

[91] Weichao Qiu. Generating human images and ground truth using computer graphics ew, 2016.

[92] Jan Wöhlke, Shile Li, and Dongheui Lee. Model-based hand pose estimation for generalized hand shape with appearance normalization, 02.07.2018.

[93] Xingyi Zhou, Qixing Huang, Xiao Sun, Xiangyang Xue, and Yichen Wei. Towards 3d human pose estimation in the wild: a weakly-supervised approach. 2017.

[94] Yebin Liu, Juergen Gall, Carsten Stoll, Qionghai Dai, Hans-Peter Seidel, and Christian Theobalt. Markerless motion capture of multiple characters using multiview image segmentation. 2013.

[95] Shanxin Yuan, Guillermo Garcia-Hernando, Bjorn Stenger, Gyeongsik Moon, Ju Yong Chang, Kyoung Mu Lee, Pavlo Molchanov, Jan Kautz, Sina Honari, Liuhao Ge, Junsong Yuan, Xinghao Chen, Guijin Wang, Fan Yang, Kai Akiyama, Yang Wu, Qingfu Wan, Meysam Madadi, Sergio Escalera, Shile Li, Dongheui Lee, Iason Oikonomidis, Antonis Argyros, and Tae-Kyun Kim. Depth-based 3d hand pose estimation: From current achievements to future goals, 11.12.2017.

[96] Yongheng Zhao, Tolga Birdal, Haowen Deng, and Federico Tombari. 3d point capsule networks, 27.12.2018. As published in CVPR 2019 (camera ready version), with supplementary material.

[97] Xiaowei Zhou, Menglong Zhu, Spyridon Leonardos, Kosta Derpanis, and Kostas Daniilidis. Sparseness meets deepness: 3d human pose estimation from monocular video, 30.11.2015. Published in CVPR2016.

[98] Xingyi Zhou, Qingfu Wan, Wei Zhang, Xiangyang Xue, and Yichen Wei. Model-based deep hand pose estimation, 22.06.2016.

[99] Jun-Yan Zhu, Taesung Park, Phillip Isola, and Alexei A. Efros. Unpaired image-to-image translation using cycle-consistent adversarial networks, 30.03.2017. An extended version of our ICCV 2017 paper, v6 updated the implementation details in the appendix. Code and data: https://github.com/junyanz/pytorch-CycleGAN-and-pix2pix.

[100] Silvia Zuffi and Michael J. Black. The stitched puppet: A graphical model of 3d human shape and pose. In *2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 3537–3546. IEEE, 07.06.2015 - 12.06.2015.

License