TU WIEN Informatics

Optimizing Realism in Synthetic Data for Training 2D Human Pose Estimation Algorithms

DIPLOMARBEIT

zur Erlangung des akademischen Grades

Diplom-Ingenieur

im Rahmen des Studiums

Media and Human-Centered Computing

eingereicht von

Fabian Stiedl, BSc
Matrikelnummer 01029117

an der Fakultät für Informatik

der Technischen Universität Wien

Betreuung: Ao.Univ.Prof. Dipl.-Ing. Mag. Dr. Margrit Gelautz
Mitwirkung: Dr. Michael Hödlmoser

Wien, 23. November 2020

Fabian Stiedl	Margrit Gelautz

TU WIEN Informatics

Optimizing Realism in Synthetic Data for Training 2D Human Pose Estimation Algorithms

DIPLOMA THESIS

submitted in partial fulfillment of the requirements for the degree of

Diplom-Ingenieur

in

Media and Human-Centered Computing

by

Fabian Stiedl, BSc
Registration Number 01029117

to the Faculty of Informatics

at the TU Wien

Advisor: Ao.Univ.Prof. Dipl.-Ing. Mag. Dr. Margrit Gelautz
Assistance: Dr. Michael Hödlmoser

Vienna, 23rd November, 2020

Fabian Stiedl | Margrit Gelautz

Erklärung zur Verfassung der Arbeit

Fabian Stiedl, BSc

Hiermit erkläre ich, dass ich diese Arbeit selbständig verfasst habe, dass ich die verwendeten Quellen und Hilfsmittel vollständig angegeben habe und dass ich die Stellen der Arbeit – einschließlich Tabellen, Karten und Abbildungen –, die anderen Werken oder dem Internet im Wortlaut oder dem Sinn nach entnommen sind, auf jeden Fall unter Angabe der Quelle als Entlehnung kenntlich gemacht habe.

Wien, 23. November 2020

Fabian Stiedl

Danksagung

Ich möchte mich bei allen bedanken, die zu dieser Arbeit beigetragen haben. Vielen Dank, Florian Seitner und Michael Hödlmoser, die mir erste Ideen für diese Arbeit gegeben und mich auf dem ganzen Weg unterstützt haben. Vielen Dank an meine Betreuerin Margrit Gelautz, die immer für mich da war, wenn ich Hilfe brauchte.

Vielen Dank an meine Kollegen bei emotion3D, insbesondere an Sebastian Beyer, der Einblicke und Wissen teilte, als ich gerade anfing. Vielen Dank an meine Familie, die mich bei allem unterstützt, was ich tue. Vielen Dank an meine Freundin Nathalie, die es geschafft hat, mich zu motivieren, als keine Motivation gefunden wurde.

Besonderer Dank geht an meine Nichten Hannah und Lily für ihre Freude und ihr Lachen, den wahrscheinlich besten Stressabbau, den es gibt.

Acknowledgements

I would like to thank everyone who contributed to this thesis. Thank you Florian Seitner and Michael Hödlmoser who gave me initial ideas for this thesis and supported me all the way. Thanks to Margrit Gelautz, my advisor who was always there for me when I needed help.

Thanks to my colleagues at emotion3D, especially Sebastian Beyer who provided insight and knowledge when I was just starting out. Thanks to my family who support me in everything I do. Thanks to my girlfriend Nathalie who managed to motivate me when no motivation was found.

Special thanks to my nieces Hannah and Lily for their joy and laughter, the best stress relief there is.

Kurzfassung

Das Generieren von synthetischen Daten, die Menschen enthalten, ist ein immer wichtiger werdendes Thema in der Bildverarbeitung. Im Kontext von Deep Learning werden große Mengen an annotierten Daten für das Trainieren von Algorithmen benötigt. Das Problem fehlender realer Trainingsdaten kann mithilfe von synthetischen Daten gelöst werden, da beim Generieren dieser synthetischen Daten als Nebenprodukt Annotierungen erzeugt werden. Die vorliegende Arbeit untersucht die Generierung synthetischer Bilder, die Menschen enthalten, im Zusammenhang mit der Schätzung menschlicher 2D Körperhaltung. Der Fokus dieser Arbeit liegt auf dem Design und der Implementierung eines mit der Spiel-Engine Unity entwickelten Frameworks, das synthetische Trainingsdaten für das Trainieren eines Deep Learning Algorithmus und dessen Benchmarks auf dem COCO Datensatz erzeugt. Im Mittelpunkt des Frameworks steht die Möglichkeit, eine Reihe von Parametern bei der Generierung synthetischer Daten zu ändern. Insbesondere evaluieren wir, wie sich Veränderungen von Haaren, Hintergründen und Texturen der menschlichen Modelle auf die Genauigkeit des trainierten Algorithmus auswirken. Unsere Ergebnisse zeigen, dass die Vielfalt der Variationen der ausgewerteten Aspekte bis zu einem gewissen Punkt von Bedeutung sind.

Abstract

Generating synthetic data containing humans is a subject of growing importance in computer vision. In the context of deep learning algorithms, large amounts of annotated data are needed for training. Synthetic data can help overcome a frequent lack of real annotated training data with ground truth annotations, which are produced as a by-product of the data synthesis process. This thesis examines the generation of synthetic images containing human characters in connection with human 2D joint estimation algorithms. The focus of this work lies in designing and implementing a framework using the game engine Unity for generating synthetic training data for training and benchmarking a deep learning algorithm on the COCO dataset. We develop the framework in a way that allows us to change a number of parameters when generating synthetic data. In particular, we evaluate the impact of hair, background and human model textures on the accuracy of our estimator. Our evaluations show that increasing the variability of clothing textures, hairstyles and background images clearly improves the results up to a certain point.

Contents

CHAPTER 1

Introduction

1.1 Motivation

In recent years, computer vision research has achieved important advances in trainable deep learning models, more powerful hardware and the increasing number of labeled training data. Research results show that the performance of deep learning models on computer vision tasks (in terms of the accuracy a model obtains when applied on a specific task) improves by increasing the size of available training data. [GLP18]

Increasing the number of annotated training samples can be achieved in two major ways. First, real images can be annotated. Second, synthetic images can be generated. Generating labels for real images is often done by hand with the use of labeling tools, which is a time-consuming task [APGS14]. Synthetic data is generated automatically by computer programs. These programs can often output additional meta-information, such as body joint positions, object labels or bounding boxes. However, real data's biggest advantage over synthetically generated data is photo-realism. Therefore, improving photo-realism is one of the major goals when generating synthetic data. [GLP18]

Humans are important subjects in computer vision applications. Moeslund et al. describe the importance of humans as follows: *"Understanding human activity from video is one of the central problems in the field of computer vision. It is driven by a wide variety of applications in communications, entertainment, security, commerce and athletics. At its foundations are a set of fundamental computer vision problems that have largely driven the great progress that the field has made during the past few decades."* [MebHKS11]

This thesis examines the generation of synthetic data for one such task which humans play an important role in, namely human 2D pose estimation. Human 2D pose estimation deals with detecting the position of human body joints in an image and is an important problem in computer vision. Because humans and their movements are diverse and can be captured by cameras from a wide range of angles under different lighting conditions,

images containing humans show high variability. Therefore, machine learning models need a large number of diverse training samples in order to cover the variability shown in these images. Generating synthetic images which contain this high degree of variability is an ongoing research area. Current challenges in generating synthetic images containing humans together with ground truth annotations involve generating images of clothed human characters with a high degree of photo-realism, dynamic occlusions and lighting scenarios similar to those found in the real world.

1.2 Goal

The goal of this thesis is to gain insight into how synthetic data should be generated and which variations of the data need to be considered to improve the accuracy of selected machine-learning tasks. To achieve this goal, a synthetic data generation framework will be implemented using the game engine Unity [Unia], which generates data for the task of human 2D pose estimation. The framework will enable the application of a variety of parameter variations on the synthetic data. It is expected that the framework will enable the identification of the impact of specific variations, such as different human clothing, changing backgrounds and hair variations, on the accuracy of human 2D pose estimation. The impact of these variations will be analyzed by benchmarking a state-of-the-art deep learning algorithm trained on the generated data. By training the algorithm a multitude of times on different variants of the same dataset, but with changes to specific parameters involved in the generation of the data, the impact of these variations can be inferred. To facilitate comparison, the evaluation will include the publicly available dataset Common Objects in Context (COCO) [LMB+14] and a state-of-the-art 2D human pose estimation algorithm.

It is expected that the results of this thesis will help to identify aspects that are relevant when generating synthetic data for training human 2D body pose estimation algorithms.

1.3 Thesis Structure

This thesis consists of five chapters, which are briefly outlined here. In Chapter 1, the introduction provides background information about the task this thesis focuses on, that is, generating synthetic data for computer vision tasks. Furthermore, it outlines the role of synthetic data in the context of deep learning and the specific goals of this thesis. In Chapter 2, related works which employ state-of-the-art methods to generate synthetic images containing humans are highlighted. The focus of the literature review lies on identifying methods which enable controllable changes on the generated output data. Chapter 3 provides information about the selected rendering framework to facilitate the chosen rendering techniques. In addition, design decisions and implementation details of the rendering pipeline are explained. In Chapter 4, synthetically generated datasets are evaluated using accuracy measures on the publicly available COCO dataset. The evaluation focuses on the controlled change of a variety of control parameters,

which affect the generated image, such as numbers of human textures and number of background images. Lastly in Chapter 5, the evaluation results are discussed and possible improvements to the rendering framework are suggested. Furthermore this last chapter gives an outlook to possible future research in the field.

CHAPTER 2

Related work

Exploring new ways to generate and make use of synthetic data is a rapidly expanding research area. There is a large variety of methods to generate synthetic images containing humans. This chapter will provide information about state-of-the-art methods to generate clothed synthetic humans. In addition, exemplary papers that use different strategies to generate synthetic data containing humans are reviewed. This literature review will act as a basis for the design decisions regarding the framework which generates the synthetic data, and will provide a basis for the experiments and the evaluation of the framework.

2.1 Synthetic Human Models

Humans come in many different shapes and wear a variety of clothing. Synthetic data used for training deep learning algorithms needs to encompass humans with a large variety in shape and clothing. Recent advances in computer graphics have managed to produce highly photo-realistic humans (Figure 2.1), which do however require extensive artist effort to create. When it comes to generating synthetic images for training deep learning models current challenges lie in generating a large number of diverse, photo-realistic humans which encompass the rich variation found in real humans without the need of hand-crafting photo-realism. In this chapter two main approaches to generating human models are presented.

The first group utilizes a fixed set of human objects from a library of pre-created human objects. These human models are usually hand-crafted using 3D modeling software [DWB+19, BCC+18]. The main benefit of this approach is that the objects present in the library are prepared by hand and therefore offer high photo-realism, depending on the artist's skill and effort. Having a library that can be set up rather quickly benefits pioneer studies and developing prototypes for research purposes. It offers great control over shape and texture of each human since every object is handcrafted. The downside to

(a) Video game trailer [Hel] (b) Game engine [Unrb] (c) Movie [Sta16]

Figure 2.1: Three examples of highly photo-realistic humans showing recent advancements in CGI in various media.

this approach is that a fixed set of human objects does not represent the high variability in shape and clothing present in real humans.

Da Cruz et al. [DWB+19] create a set of human objects using the open-source software make-human [Mak]. They use these pre-defined human models to create a domain-specific synthetic dataset focusing on rear seat occupancy detection in vehicles. The dataset is specifically created to test the generalization of machine learning algorithms to new surroundings. This kind of research question benefits from the controllable nature concerning photo-realism that a library of hand-made objects offers.

Barbosa et al. [BCC+18] study the usefulness of synthetic training data for the task of re-identification of individuals. Similarly to Da Cruz et al. they use make-human to generate a limited set of humans in various shapes and clothing. The task of re-identification benefits from the offered control over the created humans in a dedicated library for the task.

The second group of techniques tries to overcome the limited variability a set of pre-defined human objects suffers from by creating a parametric human model learned from 3D body scans. These parametric human models can output a large variety of human shapes. The general workflow to creating these parametric human body models consists of gathering a large amount of 3D body scan data, fitting a template body mesh to each scan and learning a linear model of parameters representing the mesh.

Reed et al. [RRTP14] developed the University of Michigan Transportation Research Institute (UMTRI) human shape, a parametric human body model which can be scaled by inputting standard anthropometric dimensions such as stature and body weight, based on a statistical analysis of 3D laser scans of hundreds of bodies of men and women. They fit a human template mesh to each body scan and conduct Principal Component Analysis (PCA). Subsequently, a regression analysis is conducted to predict principal component values from 11 standard anthropometric dimensions.

Loper et al [LMR+15] present Skinned Multi-Person Linear Model (SMPL), a parametric

human body model which includes pose-dependent shape variations. The body scan data is taken from the Civilian American and European Surface Anthropometry Resource Project (CAESAR) dataset [RBD+02], which consists of approximately 4000 3D body scans for men and women. A linear model is learned using PCA. The resulting principal components become blend shapes which represent the morphological differences of the body shapes. In addition, pose-dependent blend-shapes are trained on posed body-scan data.

Several works use these parametric human body models to create synthetic datasets containing images of humans with high variations for different tasks [VRM+17, CWL+16, HTBT19]. Varol el al. [VRM+17] generate Synthetic hUmans foR REAL tasks (SURREAL) a large scale dataset containing more than 6 million images together with ground truth 2D and 3D pose as well as depth and segmentation masks. The synthetic humans are created using the SMPL body model. They approximate the body shape by selecting a random CAESAR scan and the first 10 SMPL principal shape components.

A different approach tries to skip the 3D modelling part of human shapes and go straight to images of synthetic people with the use of a Generational Adversarial Network (GAN) [GPAM+14] (see Chapter 2.8 Deep Generative Models).

2.2 Human Model Textures

Human models need a variety of textures to be of effective use in generating synthetic images containing humans. Rendering engines that deal with 3D models of humans have a UV map for every object that is rendered. Every pixel on a UV map corresponds to a pixel on the 3D mesh. Classical library of human object approaches deal with specific UV maps for each character in the library, corresponding to face, skin and clothing textures. Dynamically generated parametric human models need a multitude of different textures for the same base mesh.

Newest 3D scanning technology can capture hundreds of thousands of points on the human body surface in a few seconds. Depending on the scanner the scanned points include color information from which UV maps can be derived. Varol et al. [VRM+17] generate SMPL textures directly from real body scans. These maps vary in skin color and face texture, but the quality is rather low due to low resolution. The subjects in the CAESAR scan data wear the same uniform tight fitting clothing which lack variation. To provide additional variety, Varol et al. extract textures from a second 3D scan set with normal clothing. Due to privacy concerns face textures are inpainted with the average CAESAR face which gets blended smoothly with the original map which leads to realistic anonymized body texture, but uniform face textures.

Chen et al. [CWL+16] present a novel approach to generate a large amount of textures from real images of clothes onto human 3D models. Their technique makes use of the vast amount of product images of clothes available online. These clothing items often appear in the same canonical poses without occlusions and background clutter. This

allows Chen et al. to collect large amounts of those clothing images, extract them from the background easily and transfer them onto the texture map of the human 3D model. This transfer is done by warping the clothing items into the UV map's space. First, a matching of source contour and target contour is established. Second, the contour is warped. Per texture, Chen et al. extract the contour twice, once for the upper part of the body and once for the lower part. Using that technique they can transfer shirts, tracksuits, pants, etc. generating thousands of clothing textures for their base mesh.

2.3 Synthetic Scenery

Having a synthetic human is only one part of achieving photo-realistic images containing humans. The surroundings in which the human is placed play a big role in generating synthetic images. Humans in real images can appear in many different environments under various lighting conditions and they can be occluded by a number of different objects. In general, there are two main groups of approaches to generate synthetic scenes containing humans.

The first group generates a complete synthetic 3D scene and places a 3D model of a human into it [MCL+18, DWB+19]. The main advantage of this approach is that in graphics sequences made up of 3D objects everything is known (e.g. 3D shape, position, motion, lighting ...) and can be represented as ground truth for effective use as training data. But, because everything in the scene is synthetic, photo-realism is a challenge for both humans and surroundings.

Butler et al. [BWSB12] generate an entirely synthetic dataset acquired from a short open source animated 3D movie called Sintel. The dataset features ground truth for optical flow containing 1064 training frames. It contains a high degree of photo-realism and includes natural image degradations such as fog and motion blur. Although the quality of the dataset is high, the dataset is quite small for training deep learning models. Mayer et al. [MIH+16] investigate a different approach to synthetic data generation. In contrast to Sintel, which features a small set of natural photo-realistic data, they create a large scale dataset containing 25 000 frames together with ground truth data. In order to generate a wide variety of variations they rely on the random juxtaposition of 35 927 detailed 3D models from ShapeNet [CFG+15].

The second group places the generated synthetic human into a real image [VRM+17, HTBT19]. This provides photo-realism of the surroundings but causes the challenge of placing the human in respect to already existing geometry and lighting conditions.

Varol et al. [VRM+17] present such an approach in their paper "Learning from Synthetic Humans". They render small video sequences of different humans in front of a static background image. They use a subset of the LSUN dataset [YSZ+15] which features 400 000 images of kitchens, living rooms, bedrooms and dining rooms. Their work does not feature occlusions. Similarly, Hoffmann et al. [HTBT19] use an approach to synthetic scenery, by placing the synthetic humans in front of background images from the SUN397

dataset [XHE+10]. These background images also feature scenery outside of buildings. In addition they explore occlusions, by re-using humans as occlusion objects and placing multiple layers of them in front of the background image. Hattori et al. [HLB+18] consider a specific scenario for their synthetic training data. They generate training data for pedestrian observation in a new environment. Their use-case specific rendering engine generates synthetic pedestrians in front of the same background image. They solve occlusions by focusing on a specific use-case where the camera background is known before rendering pedestrians in front of the image. Subsequently, they can annotate spots in the image by hand where occlusions are possible and avoid placing humans in these areas of the background image.

2.4 Lighting a Synthetic Scene

A high number of variations in images of the real world come from the different lighting circumstances that can be captured with a camera. Therefore it is theorized that synthetic data generation with use-cases in the real world needs to offer a similarly high degree of lighting variation to be effective.

Works that deal with completely synthetic scenes can use the 3D information of surroundings and humans to render realistic lighting conditions, including shadows from various light sources which bounce off the scenery. Works that place humans in front of background images rely on the lighting variation of the already existing background images. In that case, dynamic lighting only affects the body of the human placed into the scene and can differ from lighting conditions of the surrounding scenery.

Particularly interesting regarding lighting variation are extreme cases where parts of the human model are occluded by shadows or bright light. These cases have been studied by computer vision researchers and play an important role in the variation of human-appearances. [YYR+20]

2.5 Posing a Human Model

Humans are flexible and non-rigid. They can appear in many different poses which subsequently need to be represented by the synthetic data to be of effective use in training deep learning models. It is common practice to infer poses from a motion capture dataset, which already covers a wide range of human pose space. A widely used motion capture dataset in deep learning, the CMU MoCap dataset [CMUa], which is also used by researchers in the field of synthetic data containing humans [VRM+17, HTBT19, CWL+16]. 2D and 3D poses are inferred from the motion capture data, which contains more than 2000 sequences of different high-level actions (e.g. walking, sitting, climbing, fighting ...).

As these motion capture datasets can not cover all possible poses, Chen et al. [CWL+16] employ a Bayesian network from MoCap data to cover even more poses. The learned

network can composite substructures of the network to generate new poses. Only valid substructures of the network are composited when appropriate, which in turn generates new valid poses.

Another challenge lies in the intricacies of hand poses. Hoffman et al. [HTBT19] add variations to hand poses, which conventional MoCap systems do not record. They use the "embodied hands" dataset [RTB17] to obtain realistic hand poses and fuse them with the poses from the CMU MoCap dataset.

2.6 Rendering Engines

Most works that deal with generating synthetic images containing humans use already existing rendering engines [DWB+19, VRM+17, HTBT19, CWL+16]. This allows researchers to focus on their specific research area in the field. Blender [Ble], a free open-source rendering-engine, is often used because it of its accessibility and ease of use.

Another tool, that researchers in the field frequently use are game engines. These engines combine real-time-graphics rendering with accurate physics simulation and a high degree of photo-realism in a toolset originally set out to be used by game-designers and programmers. The nature of game-engines enables researchers to implement functionalities that build upon the source-code extending the functionalities tailored for the researcher's need.

Müller et al. [MCL+18] develop a framework built with Unreal Engine [Unra] featuring physics-based cars, unmanned aerial vehicles and animated human actors in a set of diverse urban and suburban 3D environments. In their paper "Sim4CV" they focus on two case studies: autonomous UAV-based tracking of moving objects and autonomous driving using supervised learning. Their system incorporates a tracking algorithm in combination with a benchmark evaluation tool and a deep neural network for training vehicles to drive autonomously. The framework generates synthetic images with high visual fidelity in addition to automatic ground truth annotations with large variety in the generated synthetic worlds. Their automatically generated worlds are completely synthetic and are achieved by placing a select number of objects from a pre-defined library into the scene semi-automatically.

Gaidon et al. [GWCV16] build a framework using Unity [Unia] which focuses on generating images of virtual outdoor worlds together with ground-truth annotations including object detection, tracking, scene and instance segmentation, depth and optical flow. They cite Unity's strong community, which developed many "assets" which are publicly available on Unity's Asset store, as reason for choosing this game-engine. Making use of already existing assets, in their case, realistic 3D models and materials of objects, reduces the manual labor involved in generating their virtual worlds. In addition, they make use of the flexible nature of game-engines by changing a number of parameters concerning the scripts involved in generating a multitude of different variations of their virtual world, such as speed, color and model of cars placed into the virtual world.

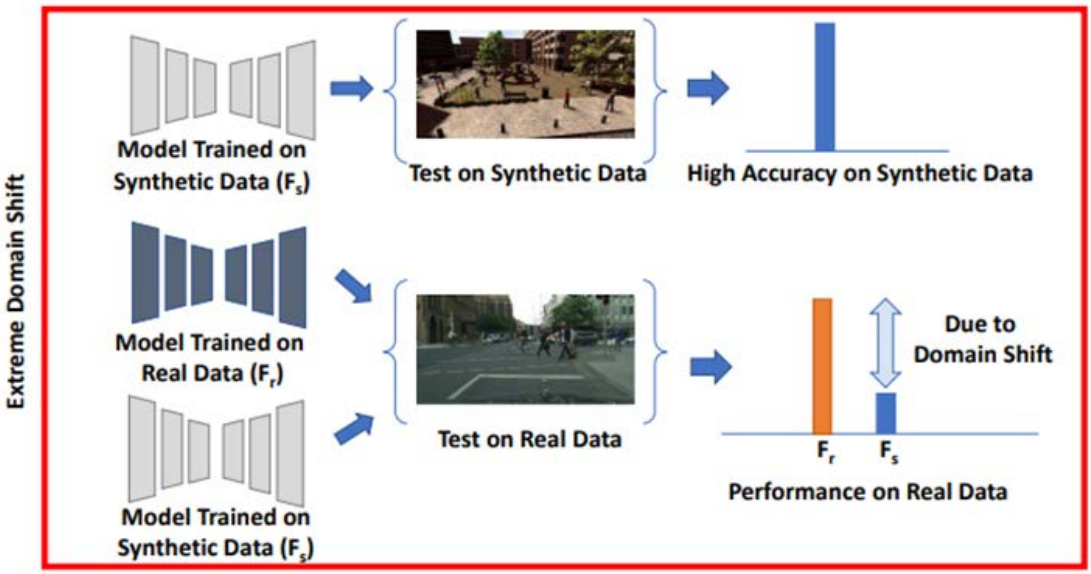


Figure 2.2: Domain Shift induces a difference in accuracy on real data. [SBJ+18]

2.7 The Gap between Synthetic and Real Data

Computer graphics have advanced considerably in the last decade. The photo-realism of synthetic data containing humans is improving rapidly. But there is still a gap between real and synthetic images used for training deep learning algorithms [PUS+18]. Models that learn from synthetic data need to transfer to real data, essentially overcoming the gap of photo-realism and difference in variation. This means that models which are trained on synthetic data usually perform worse on real data (Figure 2.2). Approaches for domain adaptation minimize some of the differences between the source and target images. Recent work by Sankaranarayanan et al. [SBJ+18] focuses on an adversarial approach based on GANs, which transfers images from the synthetic domain into the real domain. By improving the photo-realism of synthetic data, with the goal of essentially mimicking the real world, the gap in photo-realism between synthetic and real images can be closed.

2.8 Deep Generative Models

Generating photo-realistic images by using deep learning models is a rapidly evolving research area. The general idea is to make a computer program generate a variety of new images which are indistinguishable from real images. In addition, gaining control over the generated photo-realistic images is a major goal.

One generative method is the Variational Autoencoder (VAE) proposed in 2014 by Kingma and Welling [KW14]. It is a generative model learned from input data. It

consists of two parts: an encoder converting an image into a higher representation called latent space and a decoder which converts the latent space back into the initial image. By forcing the encoder to generate latent vectors which roughly follow a unit Gaussian distribution it is possible to generate new images by sampling a latent vector from the same distribution and feeding it to the decoder. VAE's strengths lie in generating a variety of new images, but challenges lie in generating high-resolution images.

Lassner et al. [LPMG17] generate people in clothing learned from the Chictopia10K [LXS+17] dataset. Their method is a two step-approach. First, they make use of the strengths of VAE's in generating a variety of conditional segmentation masks. Then, this mask gets filled in by an image-to-image translation method in a subsequent step. The generated image's photo-realism is not yet on par with traditional 3D rendering approaches, but the quality in images generated by recent variants of the VAE shows that this approach could produce higher quality results in the future [PAD20].

Another generative method called GAN was proposed by Goodfellow et al. [GPAM+14]. It is a generative technique that features two neural networks which contest each other in a "game". Given a training set, a generative network tries to generate new data similar to this set and another discriminative network evaluates the samples. The generative network tries to produce data which fools the discriminator into believing the data is real (i.e. from the original training set). The initial idea has been applied to many different computer vision problems. The role of GANs in the generation of synthetic images containing humans usually involves a convolutional neural network for image generation and a deconvolutional neural network as a discriminator. The architecture of both networks is a rapidly evolving one, steadily improving the photorealism of the output images.

Karras et al. [KLA+20] generate high-resolution, photorealistic images with StyleGAN trained on a variety of different datasets. One such dataset is the Flickr Faces High Quality (FFHQ) dataset [KLA19], which contains 70 000 high-resolution images of human faces (see Figure 2.3 for generated high-resolution images of faces).

Recent work by Pidhorskyi et al. [PAD20] tries to combine recent improvements in GAN architecture, which produce high-definition images, in combination with the control and variety of Autoencoders producing high definition face images with comparable quality of StyleGAN but means of manipulating the generated images. Generating photo-realistic images of people in clothing is yet to be achieved by these generative methods. Additionally it is a complex task to gain control over the generated output including the generation of ground truth data when generating images using generative methods.

Figure 2.3: StyleGAN2 trained on the FFHQ dataset produces high-resolution images of people's faces. [KLA19]

CHAPTER 3

Implementation

This chapter provides information about our implementation of the rendering framework. It applies suitable techniques for generating images containing humans described in Chapter 2. The focus lies on developing a rendering framework which enables controllable (see Section 3.2 for explanation) changes to the generated data. Methods described in Chapter 2 were chosen in a way which facilitates this controlled change of the generated output data but additionally offers high photo-realism. Images containing synthetic humans are rendered with added ground truth human 2D pose annotations. By randomizing parameters for each frame, a large variety of human textures, body poses, lighting, viewpoints and occlusions is generated. Each of the following subchapters focuses on one aspect of the generated output data explaining a single step of the rendering pipeline in more detail and how we randomize each aspect. Specific values and percentages for randomization were selected empirically during initial evaluation of the generated dataset. At the start of this chapter, an overview of the rendering pipeline in Unity is given, then the generation of human shapes, human textures and poses, the scene structure, lighting and post-processing are explained.

3.1 Rendering Pipeline

The rendering pipeline is integrated in [Unia]. The game engine allows the framework to build upon already existing features, such as photo-realistic shaders. In addition, Unity's ease of use thanks to many tutorials, a large community and the possibility to compile and test inside the engine itself made it the first choice. Furthermore, an asset from Unity's built-in store is used to render hair on the models. The framework was built using Unity version 2019.3.13f1 and the Universal Render Pipeline (URP). The URP is a pre-built version of the Scriptable Render Pipeline (SRP) and provides photo-realistic shaders out of the box. To render objects in the scene, Unity's built-in standard shader as well as the built-in lighting features and post-processing effects were used. Meshes are the main

representations of 3D objects in Unity and describe the shape of an object. A shader is a script which contains mathematical calculations and algorithms for calculating the color of each rendered pixel depending on the lighting input and the material configurations. A material defines how a surface of a mesh should be rendered by including references to the textures, as well as additional information which, depending on the shader, is used for rendering objects. Textures are images that a material uses as a reference and can be used by the material's shader for calculating the surface color of a mesh [Unib]. Unity calculates color information for each frame and for every object in the scene, using the specific materials and shaders used by the objects. By hooking C# scripts into Unity's frame-by-frame life cycle, the scene can be altered at every frame by changing numerous different parameters, such as the objects in the scene, the textures which are used for rendering the objects or the specific positions of objects in the scene.

A simplistic view of the steps necessary to generate new images containing a large variety of synthetic humans is illustrated in Figure 3.1. A single pass through this rendering pipeline is performed each frame. First, a random human shape is generated, which deforms a human base mesh. Then a texture from a pool of previously generated human textures is applied to the material used by our human base mesh. In the same step, one of the previously defined hairstyles is added to the mesh. After additionally varying the hair length and color of the hair's material properties, the human character is posed and placed into the scene. Placing the character into the scene causes lighting to be applied depending on the background image. Then occlusions are added by rigid objects as well as other human characters. In a final step, post-processing is applied and the finished frame is saved together with ground truth 2D body joint positions in the COCO format.

3.2 Parametrization

Since the rendering framework will be used to evaluate the effect of changes to parameters, which lead to the generation of the synthetic scenes containing humans, the framework needs to generate the data in a deterministic manner. In a preliminary step, rendering techniques which allow this deterministic generation of the data are selected. Gradual changes to the generated data are achieved by changing parameters in a script which can be handily toggled through Unity's inspector system. Each parameter is tied to a seeded random number generator which enables the user to specifically change certain parameters while other variations to the data stay the same (Figure 3.2). Furthermore, using a specific seed, results which were generated previously can be reproduced. This leads to the framework's ability to produce in principle the same synthetic dataset over and over again, but also to change a number of different parameters, for example the texture and hairstyles applied to the human avatars, and to compare their contribution to the accuracy of machine learning algorithms. This leads to greater insight into which factors of the synthetic images contribute to the resulting accuracy on machine learning tasks.

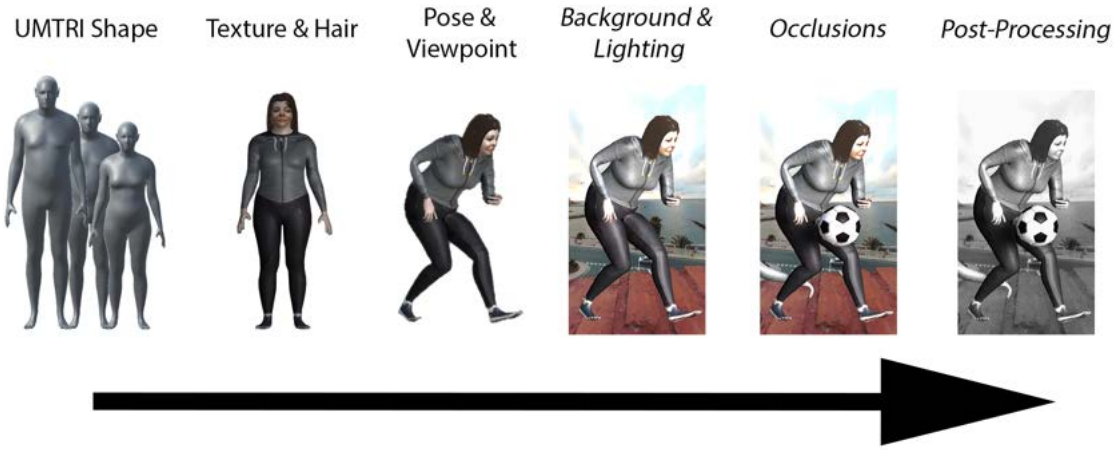


Figure 3.1: The rendering pipeline. For each frame, the steps are processed in sequence. First a UMTRI Shape is generated from random anthropometric values. Second, a random texture is applied and hair is added for the chosen shape. Then the model is posed and placed into the scene. Furthermore, scene lighting and a background image are added. Subsequently, rigid occlusion objects and human occlusions are added to the scene. Finally, post-processing is applied to the whole image, resulting in the final frame.

Figure 3.2: Example images of controllable changes to the generated dataset. The four images on the left show changes to the background. The four images in the middle visualize changes to the hair and the images on the right show clothing variations.

3.2.1 Human Shape

Humans can appear in many different shapes in real images. Therefore, the framework needs to incorporate a way to generate different human shapes which can later be posed, textured and captured by the virtual camera in the synthetic scene under various lighting conditions and occlusions. Humanoid objects in Unity need to include a skeleton of joints to control the movement of the model. The process of attaching a skeleton bone structure to an object for animation is called rigging. Each skeleton bone, which can also be called a joint, represents a controllable anchor point for Unity's underlying animation system. Most works which were reviewed in Chapter 2 focus on generating a base shape learned from body scans which are rigged and can be morphed into different human shapes (e.g. tall, slim, ...) by changing a set of parameters and are compatible with standard rendering engines. In the framework, the synthetic human shapes are created using UMTRI Human Shape developed by Reed et al. [RRTP14](see Chapter 2.1). The rendering framework uses two separate base models for men and women. These base models or base meshes can be changed by using different anthropometry values and act as a basis for the future change to the models. The body shapes for the base meshes are randomized for each frame by randomly selecting anthropometric measures from the full range of possible shapes outlined by Reed et al. For every frame either the male or female base mesh is used as the main subject for which ground truth data is generated. In subsequent steps texture and hair are applied to these base models after their shapes have been adapted.

3.2.2 Human Texture

After generating a specific random shape, textures are applied to the untextured base mesh for the male and female characters. In Unity, each object which is rendered has materials which link one or multiple corresponding texture maps to the object. Texture maps where each point on the map relates to a point on the object providing color information are called UV coordinate maps. By changing the corresponding UV map in the material linked to the human objects, the untextured base mesh can be changed to a clothed character. Therefore, in a preliminary step, a large variety of textures are generated which can be applied to the model's materials for each frame. The textures are generated similarly to Chen et al. [CWL+16] where images of clothing items in canonical poses are warped onto the UV map of the same base meshes of our human characters. The framework works with two separate base models for male and female shapes. Therefore, UV maps for men and women are differentiated and these textures are generated separately. In addition, faces are stitched onto the UV maps which were generated by StyleGAN2 [KLA+20] trained on the FFHQ [KLA19] dataset. Moreover, since the framework encompasses two base models, male and female faces generated by StyleGAN are differentiated. This leads to the generation of a large variety of face and clothing textures for both base meshes which are combined into numerous variations of full human textures for the models.

In detail the UV map is generated by stitching clothing items together. Therefore the UV

map is segmented into four regions of interest for stitching: upper-body, lower-body, face and shoes. For the upper-body, lower-body and shoes 936 clothing images in canonical poses were downloaded from Google images. Clothing items which are displayed on online-shopping websites often appear in the same viewing angle and shape (Figure 3.3). For the upper and lower-body regions of interest, two categories of clothing items are defined. The upper-body features long and short-sleeved items and the lower-body features pants and shorts. These clothing items are then warped from their original appearance into the space of the UV maps. The pipeline for warping a single clothing item can be seen in Figure 3.4 First, images in canonical poses were rescaled to fit the dimensions of the UV map. Furthermore, the warping was done by segmenting the contour of the clothing images into several parts for better warping accuracy. The segmentation of the contour was done by handcrafting simple feature detectors in clothing items for each region of interest. After the contour was segmented, each contour part was then matched by finding pairwise matchings (Figure 3.5) between each point of the source contour and the target contour, in which case the source is the canonical image downloaded from the internet and the target is the contour of the UV map. Subsequently, rigid Moving Least Squares (MLS) warping [SMW06] was performed using the contour points source and target positions for each segment of the contour. The MLS algorithm creates smooth deformations which deform the entire plane that the image lies in. Some clothing items which were too distorted after warping were discarded by hand.

By randomly combining warped clothing items and stitching them onto a single UV map, a set of 1248 UV maps for each of the male and female base models is generated (see Figure 3.7 for example UV maps). Since the upper-body and lower-body regions of interest feature two categories of clothing items, long and short sleeves as well as pants and shorts, these categories get mixed to facilitate every possible combination of clothing categories (e.g. long sleeves in combination with pants and shorts or short-sleeved items in combination with pants and shorts). In addition, each UV map is assigned a random face from the 285 faces which were generated by StyleGAN2 trained on the FFHQ dataset. The generated faces are rich in diversity and have a high degree of photo-realism. Some of the generated faces also feature glasses and beards. Figure 3.6 shows a number of generated faces which were masked and are ready for warping and subsequent stitching. The images generated by the GAN are in similar canonical pose, looking straight into the virtual camera. This was important because when warping into the UV map's space, the faces should contain as little distortion as possible. Moreover, this uniformity in the generated faces is vital for future keypoint annotation by ensuring the eyes, mouths and noses are properly positioned. Since the faces were generated by a GAN, a large variety of face textures could be used for the human texture generation of the models. The generated faces also vary in skin color. Specific spots in the UV map are reserved for skin color of the hands, necks, calves and lower arm areas of the human models. For each face, the average skin color of the UV map is adapted by calculating a k-means cluster of the face colors [RA09]. The parts reserved for skin are filled with this calculation of prototypical average face color. In addition, random white noise is added to the skin parts in the UV map to provide some texture to the skin of the models.

Figure 3.3: Example images of clothing items in canonical poses which were downloaded from the internet using Google image search. Each clothing item in its category: upper-body, lower-body and shoes, exhibit a similar canonical appearance (e.g. t-shirts are shown from the front). This similarity of viewpoint and shape provides the possibility of warping a multitude of items into the same specific parts of the texture space of the base-models with little distortion.

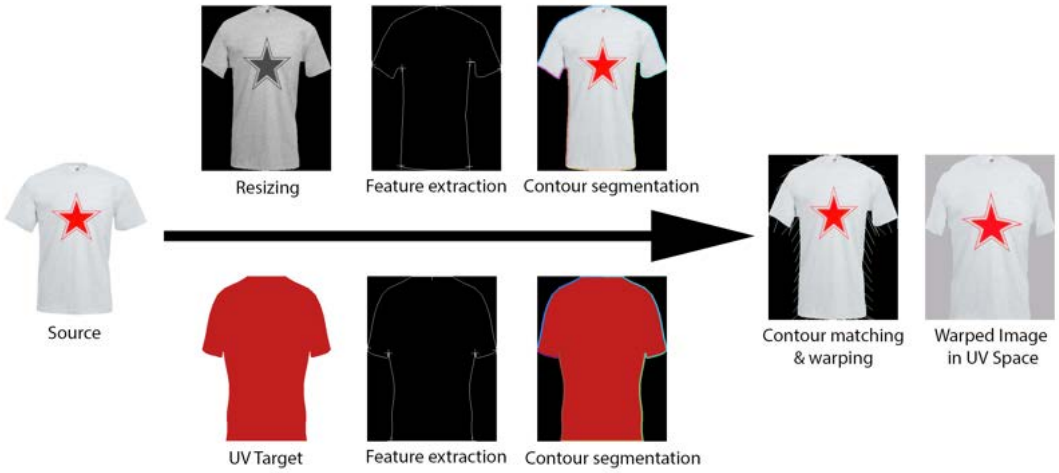


Figure 3.4: The warping pipeline shows the warping process from the source image, which was downloaded from the internet, to the warped image in UV space, which gets stitched on the UV map of the male and female base models. First, the image is reshaped to fit the dimensions of the UV target. Then a simple feature extractor marks the points on the contour for subsequent contour segmentation. Contour parts are then matched pairwise and warped using MLS warping. The resulting warped image is in UV space and is ready to be applied to a UV map.

3.2.3 Hair

The hair of the models is created with the Unity Asset Hairdesigner [Hai], which was downlaoded from the Unity asset store. It is a tool for creating realistic looking hair for specific models inside the Unity Editor by hand. In total, five different hairstyles for the male and female base models were created. Each hairstyle was designed by hand and can be further varied by changing a number of Hairdesigner's built-in scripts. Specifically, the hair is manipulated by changing length and color, for each frame (Figure 3.8). This leads to additional variance in resulting hairstyles.

Changing the length either cuts of the strands at specific lengths or scales the textures used for rendering the hair, resulting in two different looks for length adjustment. Each adjustment features hair ranging from bald to full hair. The base-length of full hair is dependent on the specific hairstyle. For the hair color, 24 colors featuring a wide range of color variation were defined. For each frame, one of the modeled hairstyles is selected and placed onto the human models. Additionally, the hair of the models is varied in terms of color and length. The hair for the main human subject, for which ground truth data is generated, as well as the hair of the other human subjects which provide occlusions are varied separately.

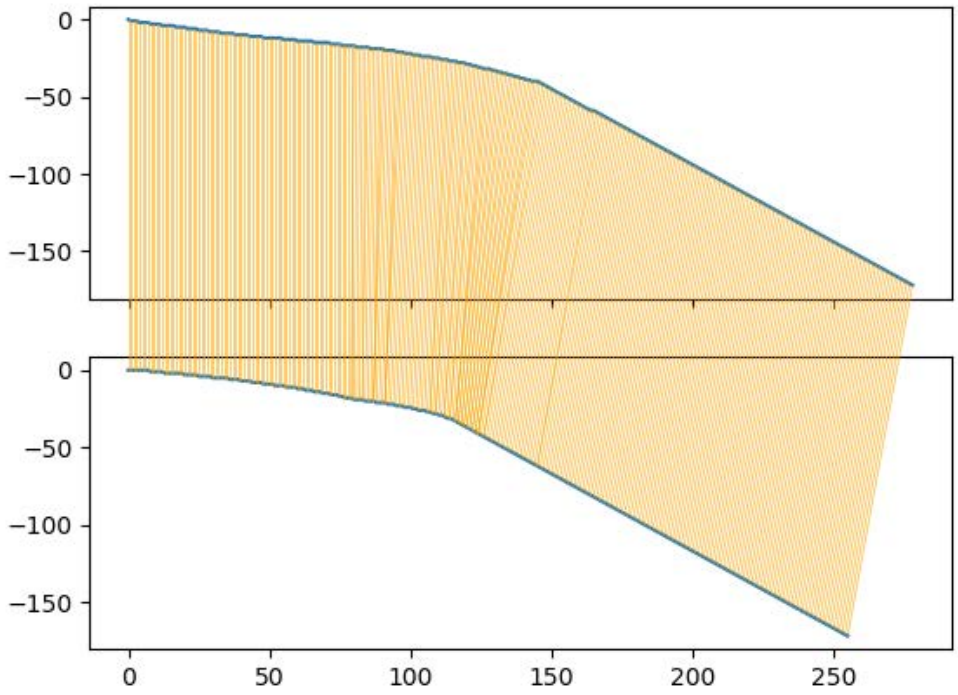


Figure 3.5: A matching between two contour parts. The orange lines show pairwise matchings between the contour lines. Contour parts can be of varying length.

3.2.4 Body Pose

The human models are posed using motion capture data from the Carnegie Mellon University Motion Capturing (CMU MoCap) dataset [CMUa] featuring high-level actions such as running, jumping, fighting and talking. A subset of this dataset is used which contains 2537 motion capture sequences in the Filmbox (FBX) format which are compatible with Unity's animation system [CMUb]. In addition, 50 animations were added which were downloaded from Mixamo [Mix]. These animations were selected by hand and feature high variance in supplementary actions. This adds up to a total of 2587 animations for posing the human models. A random frame from a random motion capture is selected for each human pose. The poses of the main human subject for which ground truth data is generated as well as the other human models which are placed into the scene for occlusion purposes are varied separately.

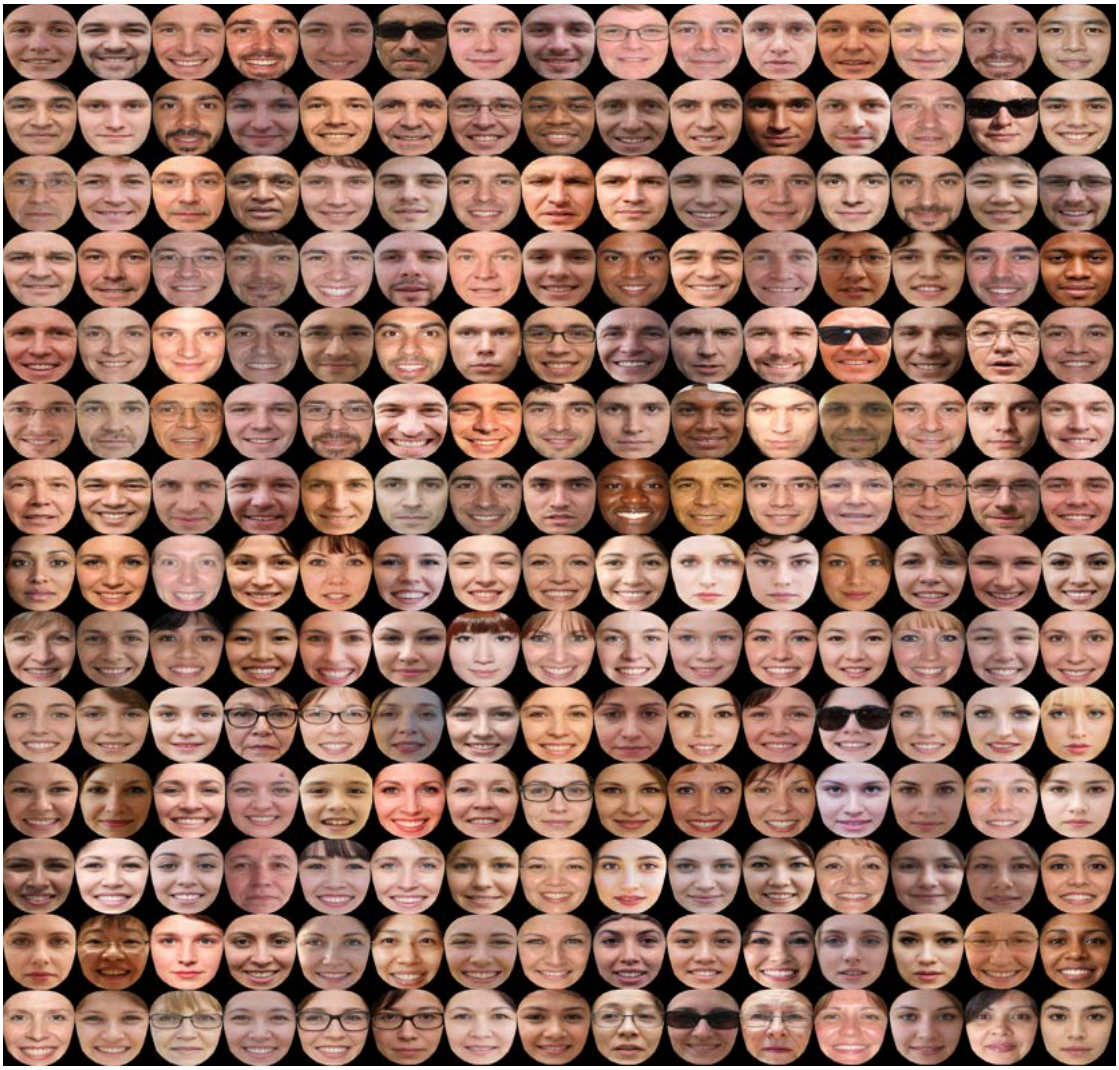


Figure 3.6: A sample from the face textures generated by StyleGAN2, taken from an online source [Sty]. These are subsequently warped onto the UV maps. Note that the faces are generated in similar canonical pose so that the warping onto the UV map distort them as little as possible through the warping. Additionally the generated faces feature glasses and beards, a rich diversity of different face structures and a high degree of photo-realism.

Figure 3.7: Example UV maps for the female base mesh (left) and the male base mesh (right).

3.2.5 Backgrounds

The scene consists of a background image in front of which the human models are placed. The background image is a skybox, which enables the use of Unity's built-in features for manipulating lighting. A skybox represents texture information on the horizon in Unity. The skybox in Unity is typically a cube map containing textures for six sides of a cube, where each side of the texture represents a background texture of a specific direction. In total, the framework features 52 skyboxes, which were downloaded from an online source [Sky]. The skyboxes feature indoor and outdoor scenes, including dark environments during the night and indoor scenes with sparse lighting. Having skyboxes which simulate dark environments is particularly important because the amount of ambient light present in the scenes is derived from the brightness present in the skybox. In the framework, the camera is static. In order to simulate different camera viewpoints the selected skybox is rotated randomly between 0 and 360 degrees on the horizontal axis.

3.2.6 Camera

The virtual camera captures a resolution of 768x1024 pixels with a vertical Field Of View (FOV) of 70°. In the framework, the camera's position or angle is not changed, but the objects and background images, which make up the final captured image, are varied. The main human model's position in front of the camera is changed randomly for each frame. The placement of the human follows a three-dimensional normal distribution. The mean of this distribution is in the middle of the image, two meters away from the camera. Depending on the distance of the human model to the camera, the horizontal and vertical

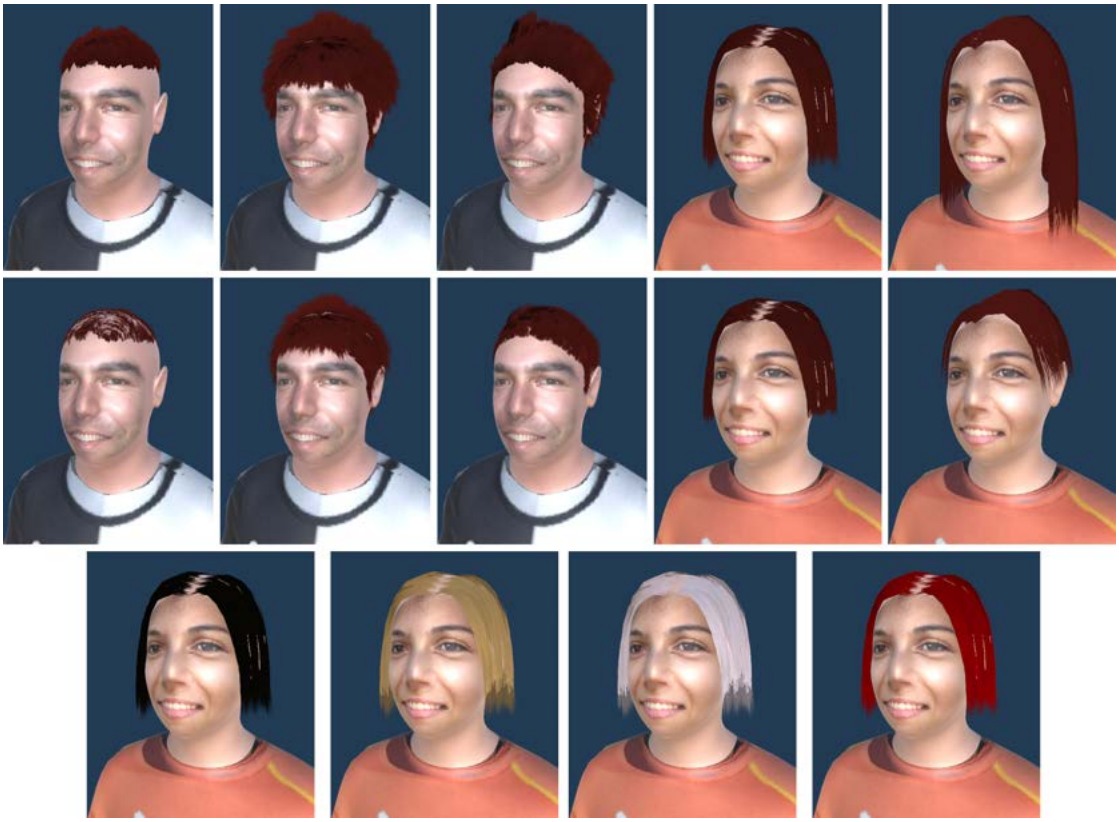


Figure 3.8: A visualization of hair variations. The top row presents all five basic hairstyles. The middle row shows the same hairstyle as in the first row but with changes to the length. The third row shows a sample of color variations to a single hairstyle.

ranges, means and standard deviations of the normal distribution are adjusted to fit the main human model anywhere between the edges of the captured image and up to 12 meters away. This variance in distances and positions within the image ensures the presence of depth blur which appears when the main subject is far away from the camera. Moreover, it ensures that a different number of pixels represent the final synthetic person which is captured by the virtual camera. In addition, moving the main human subject to the edge of the captured image facilitates partial occlusions by the edges of the image (e.g. the lower body of the main subject is out of bounds so the camera can only capture the upper-body).

3.2.7 Occlusions

Occlusions are modeled for two separate cases: occlusions by rigid objects and occlusions by other human models. Both cases fulfill the same need. That is for the algorithm to learn that humans can be partly occluded by other objects or humans. Moreover, occluding the main human subject with other human models is the key to distinguishing

Figure 3.9: Example images of the rigid occlusion objects used by the framework. The objects were selected by hand to facilitate high degree of variation and photo-realism. [Tur]

connected body parts which belong to the same human in images where more than one human is present. Therefore, the framework features occlusions by other human models. Up to two additional human models are placed into the scene with a 30% chance to appear in a frame in front or behind the main human subject for which ground truth data is generated. These human models vary in clothing, hair and pose, similar to the main subject. The humans get placed close to the main subject, at most one meter away, every which direction. In addition, 50 rigid occlusion objects were prepared which were downloaded from an online source [Tur]. The objects were selected by hand to facilitate high variance and high photo-realism in this relatively small set (Figure 3.9). Similarly to the human occlusion models these rigid objects have a chance of 30% to appear in any captured frame. The rigid objects get placed in front of the main human subject at random offsets in every direction up to a meter away. Each occlusion object (humans and rigid occlusion) is randomly rotated 360°around the upwards pointing (z) axis and tilted between -10°and +10°around the horizontal (x and y) axes.

3.2.8 Lighting

The synthetic scene features two main light sources. First, ambient lighting values are automatically set by Unity depending on the brightness levels present in the used skybox. Figure 3.10 shows different ambient lighting that the framework generates depending on the brightness levels present in the skybox. This helps to blend the objects which were placed into the scene together and acts as a basis for future highlights. In addition, this basic emission by the skybox is further varied to generate a larger variety. The framework varies the emission of the surrounding skybox by changing the skybox emission values ranging from 0.7 to 1.5 randomly for each frame. To produce a variety of bright highlights, a directional light source was implemented. For every frame, this directional light changes in regards to the intensity values and angle at which the light shines. Specifically, the framework uses Unity's standard directional light and activates it every second frame. The rotatation changes at a random angle, ranging from 0°to 360°, in every direction. The intensity of the directional light is also varied for each frame by using Unity's standard intensity values ranging from 0.5 to 1.5.

3.2.9 Post-Processing

Unity's integrated post-processing step augments the generated synthetic image right at the end of the rendering stage and impacts the final rendered image. The post-processing effects are used for two things: adding more variety to the final images and blending the objects into the surroundings. Since the post-processing effect is performed on the whole image, the background texture (i.e. skybox) and the foreground objects (i.e. humans and rigid occlusion objects) are equally affected, blending the foreground into the background. Together with the ambient lighting calculated by the skybox this blends the scene together. Seven different effects are used, which already exist in Unity's built in post-processing-stack (see 3.11). Each effect is applied with a 30% chance per rendered frame and is varied by changing standard intensity values ranging from 0 to 1.

- Greyscale: produces a greyscale image from the standard Red Green Blue (RGB) image
- Bloom: bright lights in the skybox glow, which makes light shine onto other surfaces
- Chromatic Abberation: simulates the failure of a lens to focus all colors to the same spot, this disperses colors among boundaries between dark and light areas of the image
- Focal Lenth: adjusting the virtual focal length simulates blur in relation to the depth of objects in the image
- Grain: adds grainy film noise to the image
- Color Adjustment: adjustments such as tint, saturation and contrast are changed

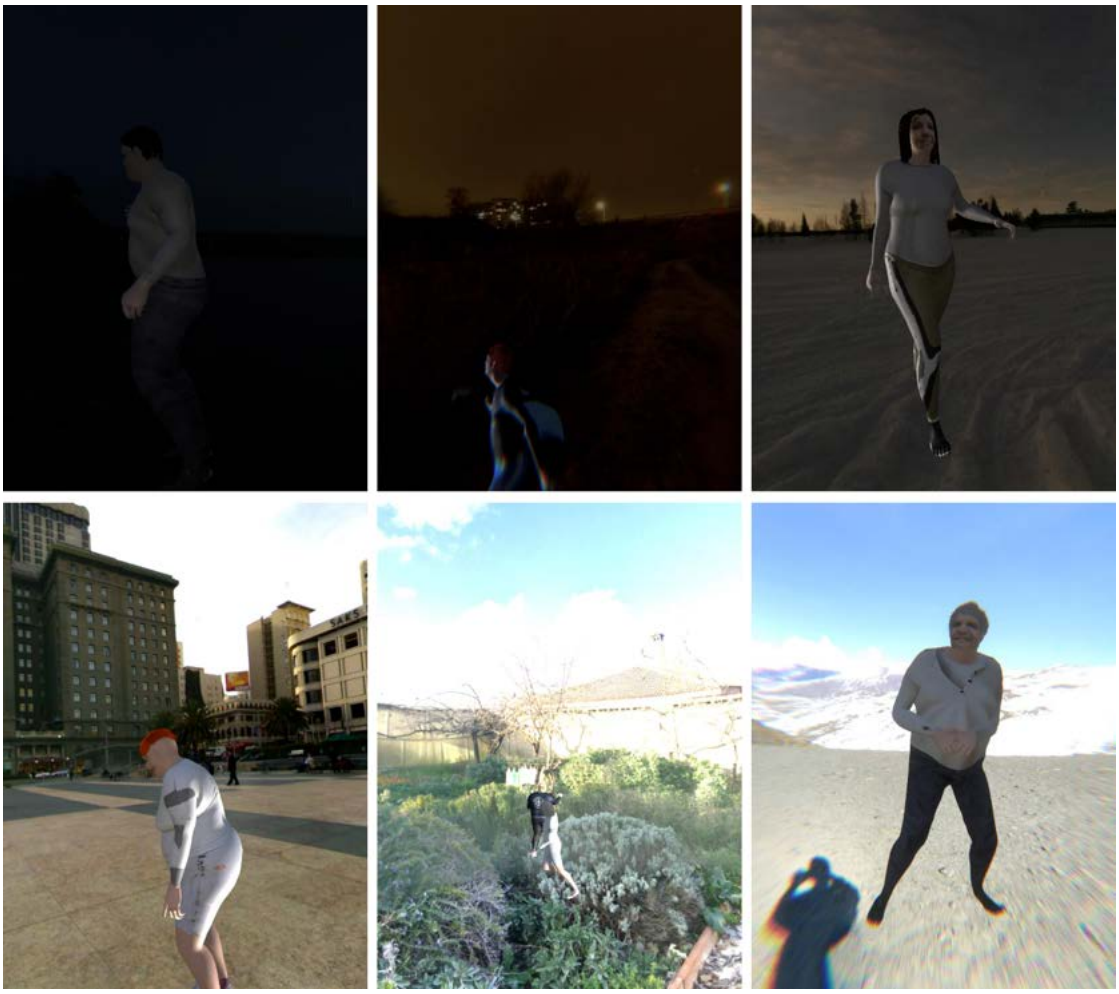
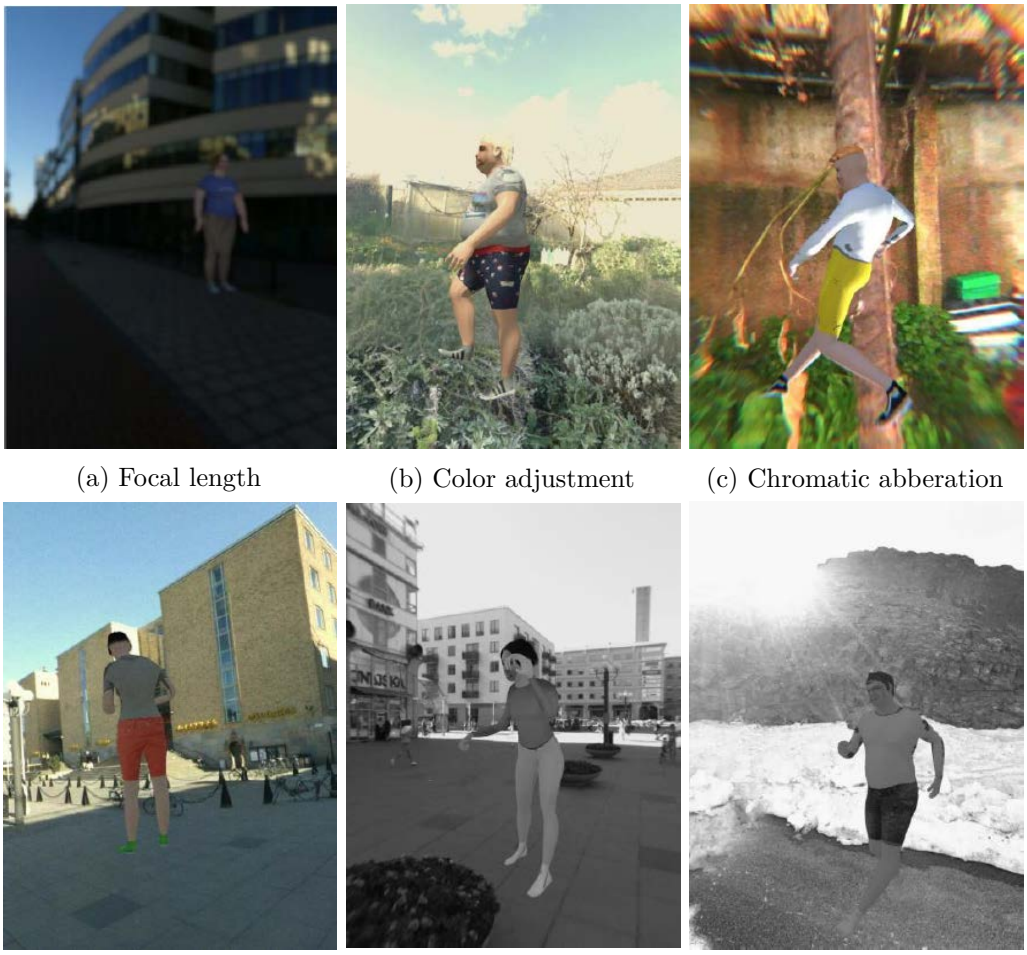


Figure 3.10: The framework is capable of calculating the amount of ambient light present in a scene and features dark and light environments. Additional highlights are rendered using Unity's built-in directional light, which shines in a random direction in every second frame.



(a) Focal length (b) Color adjustment (c) Chromatic abberation

(d) Grain (e) Grayscale (f) Bloom

Figure 3.11: Subfigures (a) through (f) show example images for each post processing effect which are used in generating our datasets.

3.3 Ground Truth

In order to train deep learning algorithms using supervised learning, ground truth data needs to be generated for the datasets. Ground truth human 2D pose annotations are generated in the COCO format [LMB+14] for every frame. This format encompasses 17 human body joints (eyes, nose, ears, shoulders, ...) which are tracked by the framework at data generation. Figure 3.12 shows all 17, 2D human body joints in the COCO format. The 2D positions of the human body joints are calculated by projecting the 3D positions of the joints onto the final image. But since the 3D positions are known, these can also be documented as ground truth data. Furthermore, by using specific textures on the materials linked to the objects in the scene, pixel-wise instance segmentation masks can be generated, from which bounding boxes of objects can be derived. In addition, it is

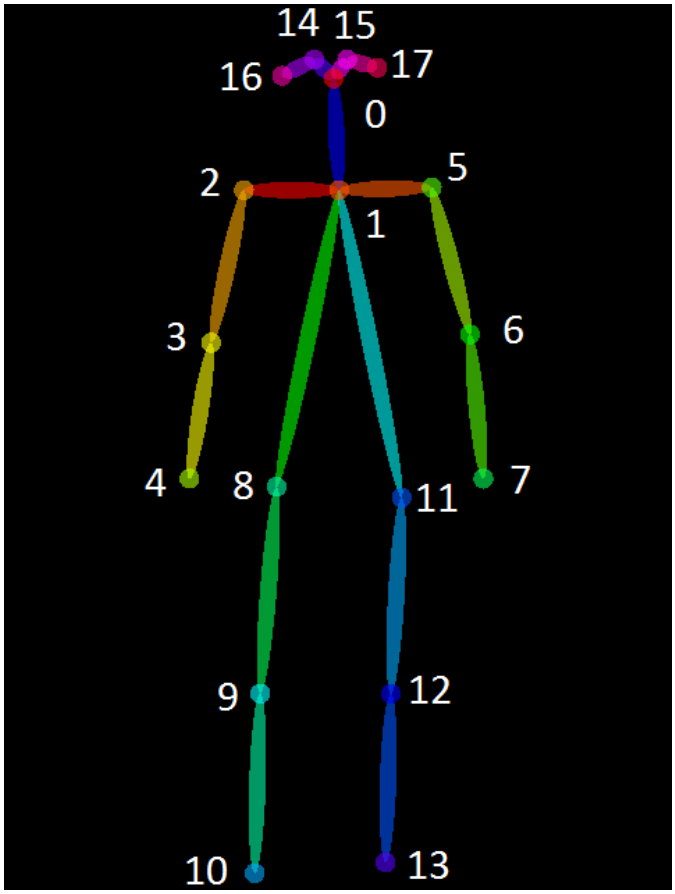


Figure 3.12: All 17 human body joints of the COCO format, forming a human skeleton structure.[COC]

possible to document the anthropometric parameters which led to the human model shape generation, such as stature, Body Mass Index (BMI), age and sitting height to stature ratio.

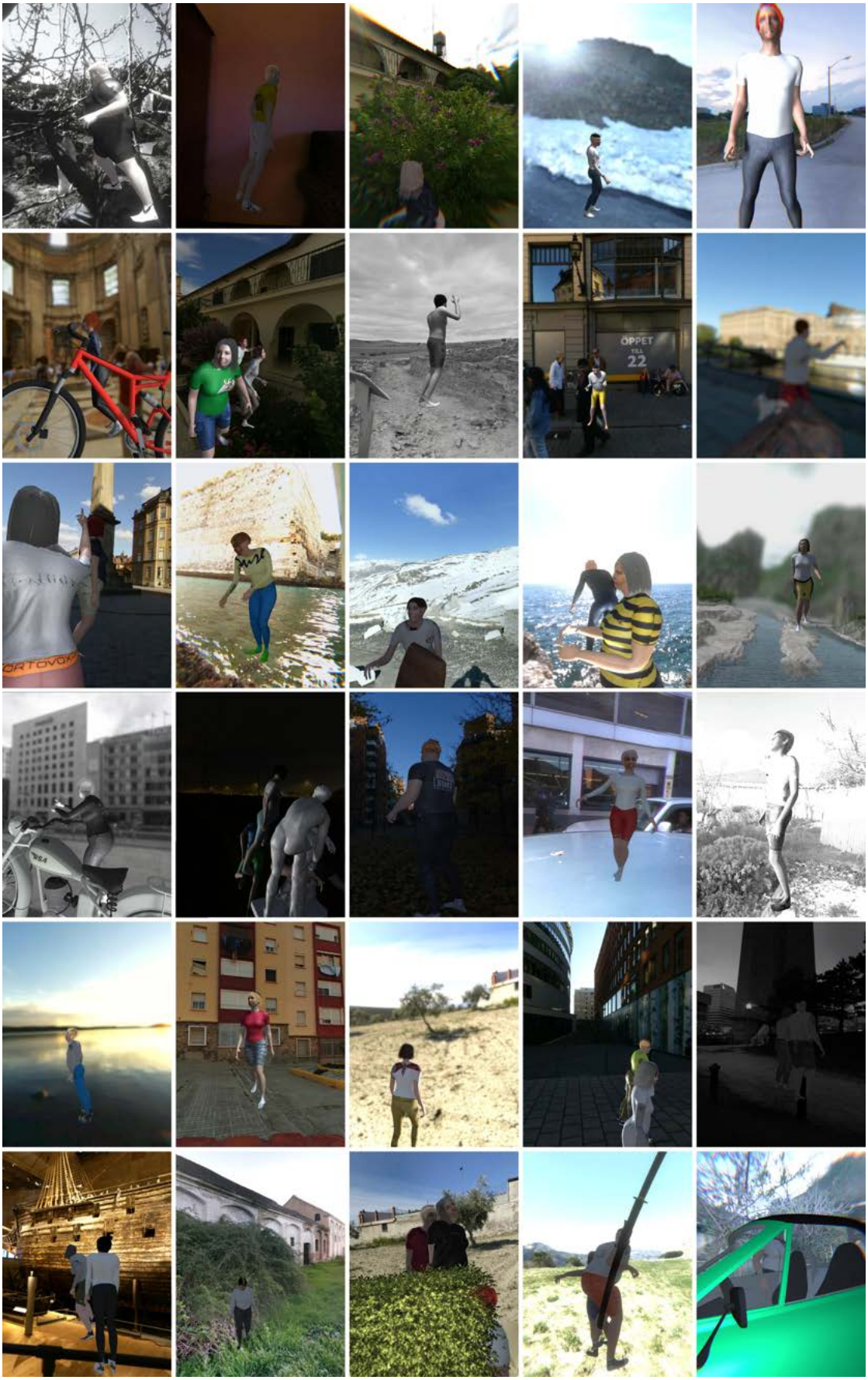


Figure 3.13: Sample frames from the generated dataset with a large variety of viewpoints, clothing, hairstyles, backgrounds, lighting, occlusion objects and post-processing effects.

CHAPTER 4

Evaluation and Results

In this chapter, a quantitative and qualitative analysis of the developed rendering framework is presented. During research and implementation, several questions arose regarding the effectiveness of the generated dataset on the accuracy of human 2D pose estimation algorithms. Questions such as: "How does the accuracy increase with increasing numbers of human textures?", "How does the accuracy increase with increasing numbers of background textures?" and "Is the hair photo-realistic enough to make a noticable difference in accuracy?" came up during research and implementation and motivated the set-up of the experiments.

4.1 Evaluation Strategy

The evaluation uses a proprietary, deep learning 2D pose estimation algorithm provided by Emotion3D with the goal to produce results which can be compared to other 2D pose estimation algorithms trained on synthetic data as well as real data. The proprietary algorithm's input are images which only contain parts where the human is present (i.e. the bounding box which surrounds the human). During training the deep learning algorithm learns from synthetic images together with ground truth 2D joint positions, which are both generated by the framework described in Chapter 3. At inference the algorithm outputs a prediction of 2D joint positions. For each experiment the algorithm is trained on different variants of generated synthetic images. The way the algorithm is trained stays the same for all experiments. In detail, the number of images used for training and validation as well as hyperparameters settings, such as learning rate, batch size and number of epochs, stay the same when training the algorithm. The accuracy of the datasets is evaluated quantitatively, in terms of Average Precision (AP) achieved on the COCO dataset, and qualitatively by comparing the predicted human 2D body joints of the algorithms which were trained on different variants of the synthetic data.

Figure 4.1: A selection of COCO images with accurate predictions by the algorithm which was trained on the main dataset. Colored dots represent predicted joint positions. A specific color relates to the type of predicted joint (e.g. red for the left ankle joint). Blue lines connect predicted joint positions to form a skeleton structure (e.g. the left wrist is connected to the left elbow). The predicted 2D joint positions for this selection of images worked particularly well and properly formed skeleton structures can be seen.

The metrics used in the evaluation of the trained algorithms follow the COCO standards described in detail by Ronchi et al. [RP17]. The similarity between predicted and ground truth 2D joint positions is defined as the Object Keypoint Similarity (OKS), a similarity measure which ranges from 0 to 1, and is computed by a Gaussian function centered on the ground-truth position of the joints. The Gaussian's standard deviation is specific to the joint type (e.g. nose or hip) in relation to human precision at the specific joint type. Using a specific threshold for the OKS similarity measure leads to classified predictions in True or False Positives (either above or below the threshold). False Negatives are defined as unmatched annotations, when ground truth joints are not predicted by the algorithm. With this classification, Precision and Recall values can be calculated. The overall AP this thesis reports on is the area under the Precision Recall curve averaged over all OKS threshold values (.5:.05:.95). The notation describes that the OKS threshold is incremented by 0.05 between the OKS thresholds of 0.5 and 0.95 and subsequently averaged to get the final AP metric, which is the standard COCO challenge metric. In addition, we report a variant of the standard COCO metric, the AP_{50} metric, a looser metric with an OKS threshold value of 0.5. In simple terms, the looser metric allows predicted keypoints to be further off from ground truth keypoints to count as a True Positive.

Figure 4.2: A selection of COCO images with inaccurate predictions by the algorithm which was trained on the main dataset. Colored dots represent predicted joint positions. A specific color relates to the type of predicted joint (e.g. orange for the left wrist joint). Blue lines connect predicted joint positions to form a skeleton structure (e.g. the left wrist is connected to the left elbow).

4.2 The Main Dataset

A main dataset which includes all the variations described in Chapter 3 was generated. An example of this main dataset can be seen in Figure 3.13. The main dataset features 20 000 training images and 2000 validation images, including ground truth annotations of human 2D body joints in the COCO format [LMB+14]. The dataset includes different human shapes, body poses, hair and clothing textures, a variety of viewpoints, different lighting conditions, post-processing augmentations and occlusions by rigid objects as well as other human models. The main dataset features all the variations made possible by the generation framework and achieves the highest accuracy on the COCO dataset with an AP of 21.9 and an AP_{50} of 43.7. It serves as a benchmark for the experiments, where changes to specific parameters, such as number of human textures, in the generation of this dataset were made to produce a multitude of similar datasets which only vary with regards to these specific aspects.

Figure 4.1 and Figure 4.2 show qualitative evaluation of predictions the algorithm outputs when trained on the main dataset. Overall, the trained algorithm which uses the main dataset works on images with different lighting conditions, poses and viewpoints. It can even generalize to unseen data by making accurate predictions for people riding bikes, which is not in the synthetic training data. The trained algorithm achieves good results on images which are similar to the ones generated in the main dataset. But it still

(a) Dataset example images with 2 human model textures.

(b) Dataset example images with 4 human model textures.

(c) Dataset example images with 8 human model textures.

(d) Dataset example images with 20 human model textures.

(e) Dataset example images with 2048 human model textures.

Figure 4.3: A comparison of the different variations in human model textures on example images from each dataset. Subfigures (a) through (e) show an increasing number of human textures on the same image sequence of the datasets.

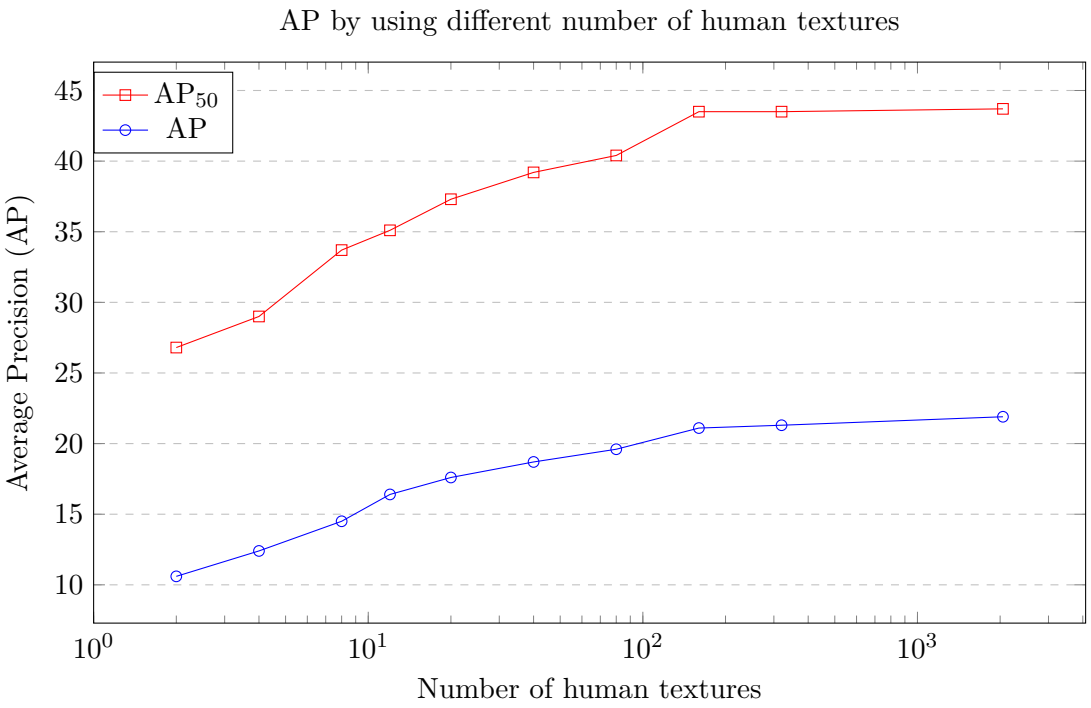


Figure 4.4: Resulting AP and AP_{50} on the COCO dataset for various numbers of human textures used when training the model. In general, the AP increases when using more textures up until a certain point, which in our experiments was around 160 different clothing and face textures of our human models.

struggles with predicting joints in images which feature variations not present in the data. It can be seen that the images which the trained algorithm has trouble with are visibly different in viewing angle and human appearance than the generated dataset. Specifically the amount of viewing angles differ to a great extent in the images which the algorithm fails at. Close-ups of heads, hands and backs are specifically hard to predict. Furthermore the trained algorithm fails at clothes such as safety vests and hooded sweatshirts. The COCO dataset encompasses a greater variety in viewing angles, textures, lighting and occlusions than the main dataset which is generated by our framework.

Looking at the two presented metrics, i.e. the more strict AP and the looser AP_{50} metric, we see that all experiments including the trained algorithm on the main dataset show a large difference in AP and AP_{50}. The gap between the AP and the AP_{50} for the algorithm which was trained on the main dataset is 21.8. This means that the trained algorithm often predicts 2D joint positions which are in the correct area, but are too far away from the ground truth position to count as a True Positive in the stricter AP metric.

Number of textures	2	4	8	12	20	40	80	160	320	2048
AP	10.6	12.4	14.5	16.4	17.6	18.7	19.6	21.1	21.3	21.9
AP_{50}	26.8	29	33.7	35.1	37.3	39.2	40.4	43.5	43.5	43.7

Table 4.1: Evaluation of the datasets which feature variation in the number of human model textures used. We used two metrics to characterize the performance of our detectors on the COCO dataset. AP_{50} at OKS=50 (loose metric) and AP at OKS=.5:.05:.95 (primary challenge metric). Using two textures results in an AP of 10.6. Increasing the number of textures available to the framework which generates image data increases the AP.

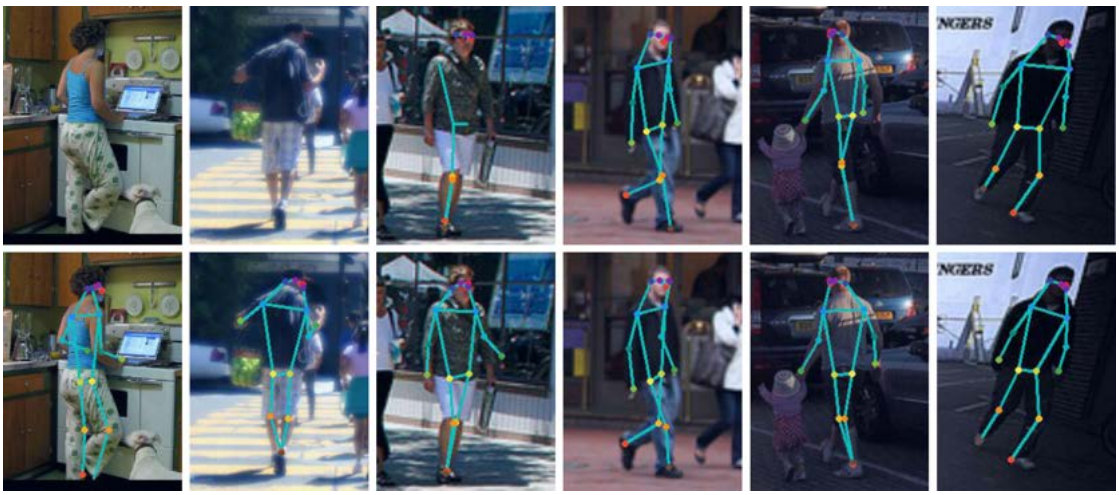


Figure 4.5: A comparison of predicted joints on validation images of the COCO dataset by the trained model on two different datasets of the clothing experiments. The first row shows the model trained on synthetic data with just one human texture per male and female model and the second row was trained using 160 different human textures.

4.3 Human Texture Experiment

The human texture experiments show the impact texture variability has on the resulting accuracy of the trained 2D pose estimation algorithm. For this experiment a multitude of datasets with a different number of UV maps for the human models were generated. Starting with only one UV map for the male and human base model we increase the number of textures incrementally up until we reach our maximum available textures, 2048. These UV maps represent the clothing and face textures of the models. Figure 4.3 shows samples from the generated datasets with a varying number of textures. It can be seen that only the human textures change but everything else stays the same. Figure 4.4 shows a visualization of the AP and AP_{50} on the COCO dataset depending on the number of textures used. Table 4.1 shows the resulting accuracies in numbers. It can be seen that at 160 textures, the accuracy no longer increases significantly compared to

Number of backgrounds	1	3	8	20	52
AP	16	17.8	19.8	21.6	21.9
AP_{50}	34.6	37.6	40.5	43.6	43.7

Table 4.2: Evaluation of the datasets which feature a variation in the number of skyboxes used as backgrounds. We used two metrics to characterize the performance of our detectors on the COCO dataset. AP_{50} at OKS=50 (loose metric) and AP at OKS=.5:.05:.95 (primary challenge metric). The increase from 20 background to 52 backgrounds only changed the resulting AP by 0.3.

experiments using a lower number of textures. At 160 textures the AP is 21.1 and at 2048 textures 21.9. In general, while the amount of different clothing and face textures matters, it only does so up to a certain point. When looking at the predicted joints in the images of two different datasets, it is noticeable that the increase in clothing and face textures help in accurately predicting 2D joint positions in well-lit scenes where clothing variations are visible in the images (Figure 4.5). In darker scenes, the algorithm, which is trained on only two clothing and face textures, predicts joints relatively well compared to the algorithm trained on a large number of textures. This can be probably attributed to the fact that these datasets still feature a large variety of lighting and post-processing variation which also affect the human texture.

4.4 Background Experiment

The background experiments were conducted similarly to the clothing experiments. We increased the number of skyboxes acting as background images and generated a multitude of datasets. Figure 4.6 shows samples from the generated datasets where changes to the number of available skyboxes were made while everything else was kept the same. Visually, changing the number of backgrounds has a large impact on the resulting images. Figure 4.7 visualizes the AP and AP_{50} which the trained algorithm achieves when using different number of background images. Table 4.2 shows the resulting accuracies in numbers.The use of only one skybox as a background image already achieves satisfying results. At 20 background images the trained algorithm approached the best result which was achieved at 52 background images in total. This may be attributed to the already high variation of color space augmentations in the post-processing step and the fact that a single skybox can offer a relatively high variation in background texture because it provides texture information surrounding the whole scene. Figure 4.8 shows the prediction of two different algorithms trained on variants of the background datasets. A general improvement across all kinds of different images can be seen when increasing the number of background images.

(a) Dataset example images with 1 skybox as a background image.

(b) Dataset example images with 3 skyboxes as a background image.

(c) Dataset example images with 8 skyboxes as a background image.

(d) Dataset example images with 20 skyboxes as a background image.

(e) Dataset example images with 52 skyboxes as a background image.

Figure 4.6: A comparison of the different variations of skyboxes as background images on example images from each dataset. Subfigures (a) through (e) show an increasing number of backgrounds on the same image sequence of the datasets.

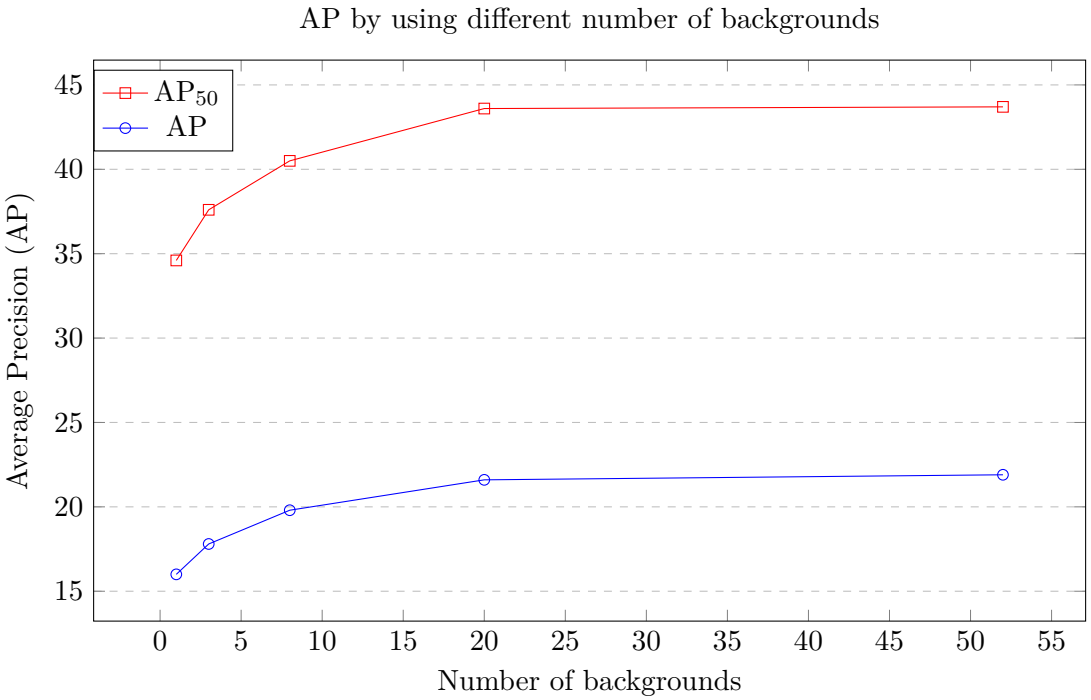


Figure 4.7: Resulting AP and AP_{50} on the COCO dataset for various numbers of skyboxes used, which act as background to our human characters when training the model. The AP increases when using more backgrounds up until a certain point, which in our experiments was around 20 different background skyboxes.

4.5 Hair Experiment

For the hair experiments, the effect that variations to the hair have on the resulting accuracy of the datasets was analyzed. For that, similarly to the other experiments, different variants of the same dataset that only feature changes to the hair of the human models were generated. It is of particular interest to see whether changes to the hair color or changes to the hair geometry are more impactful. In total, four different datasets with different changes to the hair were generated. Figure 4.9 shows a sample from each generated dataset.

The first dataset features the main human models without any hair at all. The second dataset was rendered with just one hairstyle and hair color for the male and one for the female human models. The third dataset features the same hairstyles as the former but includes variation to the hair colors. The fourth dataset uses the same hair color as the second dataset but hairstyles vary in appearance and length. This results in the described four datasets, each with a different kind of variation to the hair. Figure 4.10 visualizes the achieved AP and AP_{50} for each dataset. Table 4.3 shows the resulting accuracies in numbers. The hair, even though its photo-realism is relatively low, does

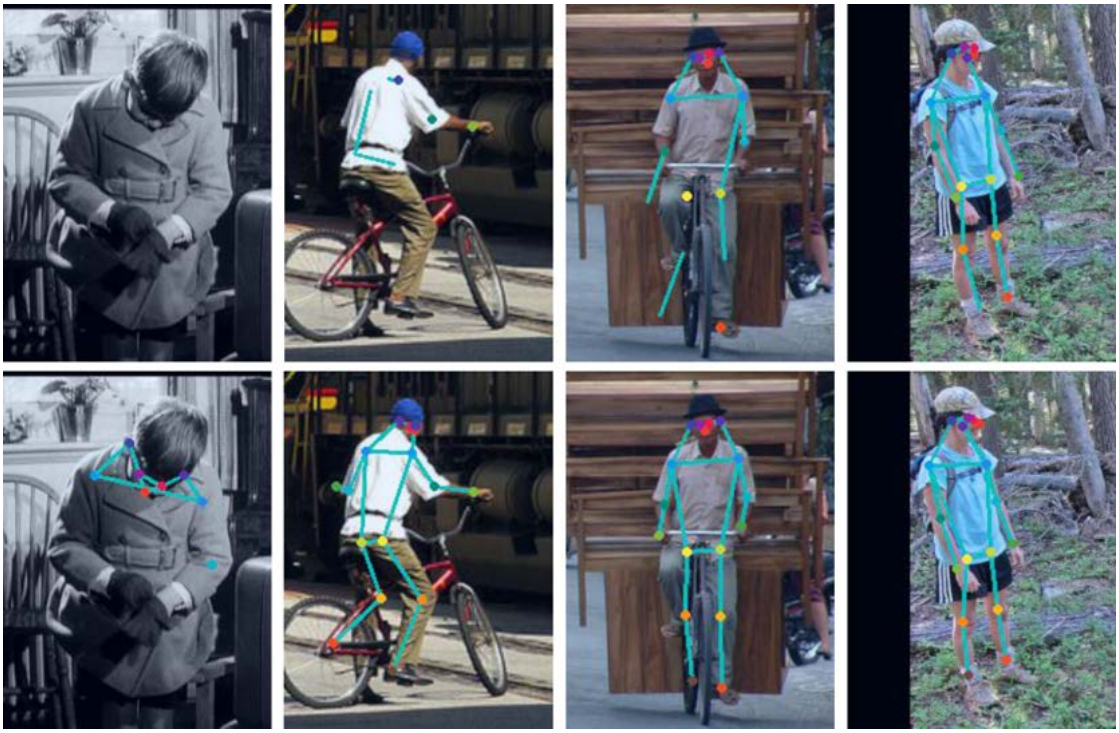


Figure 4.8: A comparison of predicted joints on validation images of the COCO dataset by the trained model on two different datasets of the background experiments. The first row shows the model trained on data with just one background and the second row was trained using 20 different backgrounds.

Hair variations	No Hair	One hairstyle and one hair color	One hairstyle and 24 hair colors	Length variation and one hair color	Length variation and 24 hair colors
AP	13.8	16.5	21.2	20.9	21.9
AP_{50}	28.8	34.5	42.8	41.7	43.7

Table 4.3: Evaluation of the datasets which feature variation to the hair of our human models. We used two metrics to characterize the performance of our detectors on the COCO dataset. AP_{50} at OKS=50 (loose metric) and AP at OKS=.5:.05:.95 (primary challenge metric). The difference between using no hair and one hair style is 2.7 AP. The difference between two types of variation: color and length, is 0.3 AP.

(a) Example images from the dataset without hair.

(b) Example images from the dataset with 1 hairpiece and 1 color.

(c) Example images from the dataset with 1 hairpiece and 24 colors.

(d) Example images from the dataset with hair length variation and 1 color.

(e) Example images from the dataset with hair length variation and 24 colors.

Figure 4.9: Comparing the different variations in hair on example images from each dataset. Subfigures (a) through (e) show each variation on the same image sequence of the datasets. Relevant parts of the image were zoomed in for visualization purposes.

have an impact on the resulting accuracy of the trained algorithm. Additionally, it can be seen that compared to no hair at all, just using a single hairstyle with the same color in every image of the dataset has a relatively strong impact on the resulting accuracy, namely a change of 5.7 AP_{50}. In addition, the difference in accuracy of the third and fourth datasets, one which features only color variation and one which features only geometry variation, is relatively small, only changing the accuracy by 0.9 AP_{50}.

When looking at the resulting prediction of the joints qualitatively, a general improvement in the detection of joints all across multiple different images can be noticed. Figure 4.11 shows two algorithms trained on different variants of the hair datasets. The addition of hair to the models helped the algorithm gain a better understanding of human appearances even though the photo-realism of the hair is rather low.

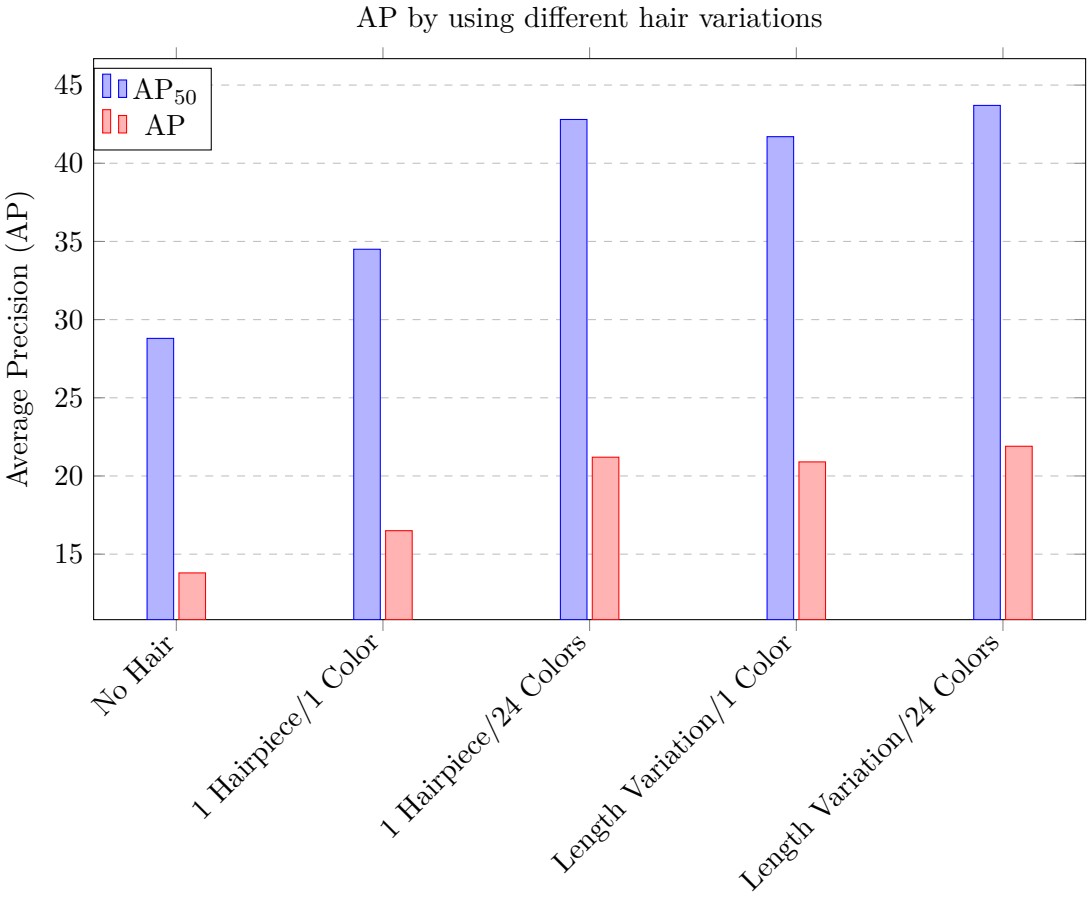


Figure 4.10: Resulting AP and AP_{50} on the COCO dataset for different variations of hairstyles on our human characters when training the model. Simply adding one hair style for male and one for the female characters increased the AP_{50} by 5.7. Specific variations to only one aspect of the hair, color or length, achieves similar variations in AP and AP_{50}.

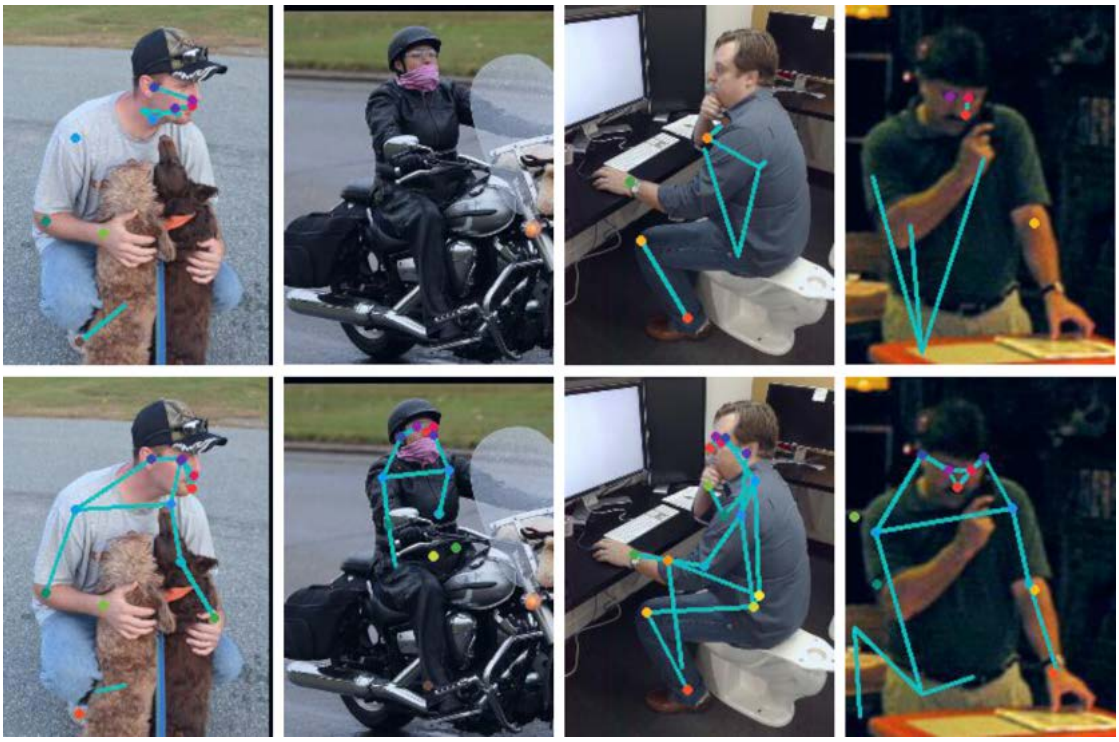


Figure 4.11: A comparison of predicted joints on validation images of the COCO dataset by the trained model on two different datasets of the hair experiments. The first row used no hair at all and the second row featured a variation in hair-length but a fixed color. There is a general improvement in the prediction of the joints. Additionally, the hair helps in improving the results on people with helmets and hats.

CHAPTER 5

Conclusion and Future Work

In this chapter, the contents of this diploma thesis are summarized by recapping the chapters and the design decisions of the developed framework. Furthermore, the main points of the evaluation results are discussed. At the end of this work, an outlook is given to possible future topics of research which could build upon the work of this thesis.

5.1 Chapter Summaries

Chapter 1 provides an introduction to the task of generating synthetic data for computer vision tasks. This thesis focused on one such task, human 2D pose estimation. The introduction further outlines the importance of synthetic humans and the challenges that lie in generating them. The goal is briefly outlined: developing a framework for generating synthetic images containing humans for human 2D pose estimation with the possibility to control certain parameters in its generation of the data.

Chapter 2 provides background information on different state-of-the-art techniques and use cases for generating synthetic scenes containing humans for various other computer vision tasks. Each subchapter focuses on one aspect of the generation of synthetic scenes. The role of domain shift which occurs when training deep learning algorithms on synthetic data and performing tasks on real data is highlighted. Additionally, deep generative models which can produce highly photo-realistic images are analyzed.

Chapter 3 describes the developed framework for synthesizing training images containing humans. First, we gave an overview of the rendering pipeline which is implemented in Unity. Subsequently, similar to Chapter 2, we described how each aspect of the synthetic scene is generated. Since the data we generate features random variation each frame, we described what random means for each variation. The framework features a main human subject which is captured in front of a variety of background images including occlusions by other humans as well as rigid objects and dynamic lighting. Using seeded random

variables, the framework was set up to facilitate the evaluations done in the following chapter.

Chapter 4 evaluates three aspects involved in generating the synthetic scenes: the number of human textures, the number of background images and variations to the hair styles of our human models. Each aspect is evaluated by training the same deep learning algorithm on different variants of the generated datasets. Quantitative evaluation is done by providing precision metrics on the publicly available COCO dataset. Qualitative evaluation is done by comparing the predicted 2D human joint positions of algorithms trained on variants of the generated datasets.

5.2 Framework Review

During the research of related works, it became clear that synthetic training data containing humans for the 2D pose estimation task should be generated using parametric human shapes because these shapes learned from body scan data encompass the rich diversity human shapes possess. We used an approach similar to Chen et al. [CWL+16] where traditional approaches which generate textures from body scan data are circumvented by warping clothing images in canonical poses onto a vast number of different texture maps. This approach provides a large variety of clothing textures which can later be used in the rendering process. In addition, the generation of photo-realistic human faces by a GAN was easily combined with the texture warping approach. Another decision was the choice of a rendering engine. Using Unity, it was possible to use photo-realistic shaders out of the box. Especially interesting was the possibility to adjust scene lighting with regards to the brightness present in the background image. Moreover, an asset from the community driven asset store was used for the design of the hair on our models. Furthermore, the existing seeded random number generation system made the controlled variation, which was necessary for the evaluation of the generated data, possible.

5.3 Synopsis of the Evaluation Results

The knowledge gained through the evaluation of the experiments can be summarized as follows: Generating synthetic training data for 2D pose estimation algorithms is a challenging task. Quantitative evaluation of the training data on the COCO dataset shows that synthetic data can be of effective use for 2D pose estimation. A rich diversity in human appearance, viewpoints and lighting conditions needs to be met in order to achieve high accuracy. Especially important to synthetic data containing humans where real backgrounds are used is the discrepancy between the human model and the background environment. By using the post-processing effects of Unity's rendering pipeline, which are performed on the whole image, the framework was able to better blend humans and background together. The experiments show that the variety of the evaluated parameters matter when generating synthetic data but only up to a certain point, beyond which the added variability of clothing textures, hairstyles and background images will not improve

the overall results anymore. Training with a smaller set of hand-crafted photo-realistic human models could be beneficial in overcoming the gap in photo-realism compared to the complex struggle of adding human geometry and texture to parametric human shapes. By evaluating qualitatively, it can be seen that variations which are not present in our generated data are hard to predict for the trained algorithm, especially unseen viewpoints and variations which are not present such as changes to clothing geometry.

5.4 Future Work

In the future, we plan to continue investigating the impact certain parameters which produce synthetic training data have on the accuracy of human 2D pose estimation algorithms. It would be interesting to further look into different variations to the dataset including similar experiments with regards to changing lighting conditions, viewpoints and occlusion objects. Additionally, using a GAN for generating faces for the human textures worked particularly well. We suggest that the use of GANs for texture creation could be researched further by going one step beyond warping generated faces for texture maps. We suggest creating whole texture maps including skin and clothing textures with recent advancements in gaining control over GANs' generational output. Besides improving photo-realism of parametric human models, it would be interesting to see how a smaller variety of highly photo-realistic human models compete against the vast variety of parametric human shapes in terms of accuracy achieved on 2D pose estimation algorithms.

List of Figures

List of Tables

Acronyms

AP Average Precision. 33–35, 37–39, 41, 42, 45, 53, 55

BMI Body Mass Index. 30

CAESAR Civilian American and European Surface Anthropometry Resource Project. 7

CMU MoCap Carnegie Mellon University Motion Capturing. 22

COCO Common Objects in Context. 2, 16, 29, 30, 33–35, 37–39, 41, 42, 45, 46, 48, 52, 53, 55

FBX Filmbox. 22

FFHQ Flickr Faces High Quality. 12, 13, 18, 19, 51

FOV Field Of View. 24

GAN Generational Adversarial Network. 7, 12, 19, 48, 49

MLS Moving Least Squares. 19, 21, 51

OKS Object Keypoint Similarity. 34, 38, 39, 42, 55

PCA Principal Component Analysis. 6, 7

RGB Red Green Blue. 27

SMPL Skinned Multi-Person Linear Model. 6, 7

SRP Scriptable Render Pipeline. 15

SURREAL Synthetic hUmans foR REAL tasks. 7

UMTRI University of Michigan Transportation Research Institute. 6, 17, 18, 51

URP Universal Render Pipeline. 15

Bibliography

[APGS14] Mykhaylo Andriluka, Leonid Pishchulin, Peter Gehler, and Bernt Schiele. 2D Human Pose Estimation: New Benchmark and State of the Art Analysis. In *Proceedings of Conference on Computer Vision and Pattern Recognition*, pages 3686–3693. IEEE, 2014.

[BCC+18] Igor Barros Barbosa, Marco Cristani, Barbara Caputo, Aleksander Rognhaugen, and Theoharis Theoharis. Looking beyond Appearances. *Computer Vision and Image Understanding*, 167(C):50–62, 2018.

[Ble] Blender. `https://www.blender.org/`. Accessed: 2020-11-18.

[BWSB12] Daniel J. Butler, Jonas Wulff, Garrett B. Stanley, and Michael J. Black. A Naturalistic Open Source Movie for Optical Flow Evaluation. In *Proceedings of European Conference on Computer Vision*, pages 611–625. Springer Berlin Heidelberg, 2012.

[CFG+15] Angel X. Chang, Thomas Funkhouser, Leonidas Guibas, Pat Hanrahan, Qixing Huang, Zimo Li, Silvio Savarese, Manolis Savva, Shuran Song, Hao Su, Jianxiong Xiao, Li Yi, and Fisher Yu. ShapeNet: An Information-Rich 3D Model Repository. *arXiv e-prints*, 2015. arXiv: 1512.03012.

[CMUa] Carnegie-Mellon Mocap Database. `http://mocap.cs.cmu.edu/`. Accessed: 2020-11-18.

[CMUb] Subset of Carnegie-Mellon Mocap Database in FBX Format. `https://3deeplearner.com/cmu-fbx/`. Accessed: 2020-10-08.

[COC] COCO Keypoints Sample Skeleton. `https://blog.csdn.net/happyhorizion/article/details/77894205`. Accessed: 2020-11-18.

[CWL+16] 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. In *Proceedings of International Conference on 3D Vision*, pages 479–488. IEEE, 2016.

[DWB+19] Steve Dias Da Cruz, Oliver Wasenmüller, Hans-Peter Beise, Thomas Stifter, and Didier Stricker. An Overview of the SVIRO Dataset and Benchmark. In *Proceedings of Computer Science in Cars Symposium*. ACM, 2019.

[GLP18] Adrien Gaidon, Antonio Lopez, and Florent Perronnin. The Reasonable Effectiveness of Synthetic Visual Data. *International Journal of Computer Vision*, 126(9):899–901, 2018.

[GPAM+14] Ian J. Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. Generative adversarial nets. In *Proceedings of International Conference on Neural Information Processing Systems*, NIPS'14, pages 2672–2680. MIT Press, 2014.

[GWCV16] Adrien Gaidon, Qiao Wang, Yohann Cabon, and Eleonora Vig. Virtual Worlds as Proxy for Multi-object Tracking Analysis. In *Proceedings of Conference on Computer Vision and Pattern Recognition*, pages 4340–4349. IEEE, 2016.

[Hai] Hair designer Unity Asset Store App. `http://www.kalagaan.com/#hairdesigner`. Accessed: 2020-11-18.

[Hel] Hellblade 2 Trailer. `https://www.youtube.com/watch?v=2TR0gaG01do`. Accessed: 2020-11-18.

[HLB+18] Hironori Hattori, Namhoon Lee, Vishnu Naresh Boddeti, Fares Beainy, Kris M. Kitani, and Takeo Kanade. Synthesizing a Scene-Specific Pedestrian Detector and Pose Estimator for Static Video Surveillance. *International Journal of Computer Vision*, 126(9):1027–1044, 2018.

[HTBT19] David T. Hoffmann, Dimitrios Tzionas, Michael J. Black, and Siyu Tang. Learning to Train with Synthetic Humans. In *Proceedings of Conference on Pattern Recognition*, pages 609–623. Springer Cham, 2019.

[KLA19] Tero Karras, Samuli Laine, and Timo Aila. A Style-Based Generator Architecture for Generative Adversarial Networks. In *Proceedings of Conference on Computer Vision and Pattern Recognition*, pages 4396–4405. IEEE, 2019.

[KLA+20] Tero Karras, Samuli Laine, Miika Aittala, Janne Hellsten, Jaakko Lehtinen, and Timo Aila. Analyzing and Improving the Image Quality of StyleGAN. In *Proceedings of Conference on Computer Vision and Pattern Recognition*, pages 8107–8116. IEEE, 2020.

[KW14] Diederik P. Kingma and Max Welling. Auto-Encoding Variational Bayes. In *Proceedings of International Conference on Learning Representations*, 2014.

[LMB+14] Tsung-Yi Lin, Michael Maire, Serge Belongie, James Hays, Pietro Perona, Deva Ramanan, Piotr Dollár, and C. Lawrence Zitnick. Microsoft COCO: Common Objects in Context. In *Proceedings of European Conference on Computer Vision*, pages 740–755. Springer Cham, 2014.

[LMR+15] Matthew Loper, Naureen Mahmood, Javier Romero, Gerard Pons-Moll, and Michael J. Black. SMPL: A Skinned Multi-Person Linear Model. *Transactions on Graphics*, 34(6), 2015.

[LPMG17] Christoph Lassner, Gerard Pons-Moll, and Peter V. Gehler. A Generative Model of People in Clothing. In *Proceedings of International Conference on Computer Vision*, pages 853–862. IEEE, 2017.

[LXS+17] Xiaodan Liang, Chunyan Xu, Xiaohui Shen, Jianchao Yang, Jinhui Tang, Liang Lin, and Shuicheng Yan. Human Parsing with Contextualized Convolutional Neural Network. *Transactions on Pattern Analysis and Machine Intelligence*, 39(1):115–127, 2017.

[Mak] Make Human. `http://www.makehumancommunity.org/`. Accessed: 2020-11-18.

[MCL+18] Matthias Müller, Vincent Casser, Jean Lahoud, Neil Smith, and Bernard Ghanem. Sim4CV: A Photo-Realistic Simulator for Computer Vision Applications. *International Journal of Computer Vision*, 126(9):902–919, 2018.

[MebHKS11] Thomas B. Moeslund, Adrian edited by Hilton, Volker Krüger, and Leonid Sigal. *Visual Analysis of Humans*. Springer Nature, 2011.

[MIH+16] Nikolaus Mayer, Eddy Ilg, P. Häusser, Philipp Fischer, D. Cremers, Alexey Dosovitskiy, and Thomas Brox. A Large Dataset to Train Convolutional Networks for Disparity, Optical Flow, and Scene Flow Estimation. In *Proceedings of Conference on Computer Vision and Pattern Recognition*, pages 4040–4048. IEEE, 2016.

[Mix] Adobe Mixamo Animations Database. `https://www.mixamo.com/#/?page=1&type=Motion%2CMotionPack`. Accessed: 2020-11-18.

[PAD20] Stanislav Pidhorskyi, Donald A. Adjeroh, and Gianfranco Doretto. Adversarial Latent Autoencoders. In *Proceedings of Conference on Computer Vision and Pattern Recognition*, pages 14092–14101. IEEE, 2020.

[PUS+18] Xingchao Peng, Ben Usman, Kuniaki Saito, Neela Kaushik, Judy Hoffman, and Kate Saenko. Syn2Real: A New Benchmark for Synthetic-to-Real Visual Domain Adaptation. *Computing Research Repository*, abs/1806.09755, 2018.

[RA09] KS Ravichandran and B Ananthi. Color Skin Segmentation using K-Means Cluster. *International Journal of Computational and Applied Mathematics*, 4(2):153–158, 2009.

[RBD+02] Kathleen Robinette, Sherri Blackwell, Hein Daanen, Mark Boehmer, and Scott Fleming. Civilian American and European Surface Anthropometry Resource (CAESAR), Final Report. Summary. *Technical Report AFRL-HEWP-TR-2002-0169*, 1:74, 2002.

[RP17] Matteo Ruggero Ronchi and Pietro Perona. Benchmarking and Error Diagnosis in Multi-instance Pose Estimation. In *Proceedings of International Conference on Computer Vision*, pages 369–378. IEEE, 2017.

[RRTP14] Matthew P. Reed, Ulrich Raschke, Rishi Tirumali, and Matthew B. Parkinson. Developing and Implementing Parametric Human Body Shape Models in Ergonomics Software. In *Proceedings of International Digital Human Modeling Symposium*, 2014.

[RTB17] Javier Romero, Dimitrios Tzionas, and Michael J. Black. Embodied Hands: Modeling and Capturing Hands and Bodies Together. *Transactions on Graphics*, 36(6), 2017.

[SBJ+18] Swami Sankar, Yogesh Balaji, Arpit Jain, Ser-Nam Lim, and Rama Chellappa. Learning from Synthetic Data: Addressing Domain Shift for Semantic Segmentation. In *Proceedings of Conference on Computer Vision and Pattern Recognition*, pages 3752–3761. IEEE, 2018.

[Sky] Skyboxes by Emil Persson. `http://www.humus.name/index.php?page=Textures`. Accessed: 2020-11-18.

[SMW06] Scott Schaefer, Travis McPhail, and Joe Warren. Image deformation using moving least squares. *Transactions on Graphics*, 2006.

[Sta16] Rogue One: A Star Wars Story, 2016. Movie directed by: Gareth Edwards, Producers: Kathleen Kennedy and Allison Shearmur and Simon Emanuel, published by: Walt Disney Studios Motion Pictures.

[Sty] This Person Does Not Exist. `https://thispersondoesnotexist.com/`. Accessed: 2020-11-18.

[Tur] Turbosquid 3D Objects. `https://www.turbosquid.com/`. Accessed: 2020-11-18.

[Unia] Unity Game Engine. `https://unity.com/`. Accessed: 2020-11-18.

[Unib] Unity Shader Basics. `https://docs.unity3d.com/Manual/Shaders.html`. Accessed: 2020-11-18.

[Unra] Unreal Engine. https://www.unrealengine.com/. Accessed: 2020-11-18.

[Unrb] Unreal Engine Showcase. https://www.youtube.com/watch?v=mkkWCmljMSA. Accessed: 2020-11-18.

[VRM+17] Gül Varol, Javier Romero, Xavier Martin, Naureen Mahmood, Michael J. Black, Ivan Laptev, and Cordelia Schmid. Learning from Synthetic Humans. In *Proceedings of Conference on Computer Vision and Pattern Recognition*, pages 4627–4635. IEEE, 2017.

[XHE+10] Jianxiong Xiao, James Hays, Krista A. Ehinger, Aude Oliva, and Antonio Torralba. SUN database: Large-scale Scene Recognition from Abbey to Zoo. In *Proceedings of Computer Society Conference on Computer Vision and Pattern Recognition*, pages 3485–3492. IEEE, 2010.

[YSZ+15] Fisher Yu, Ari Seff, Yinda Zhang, Shuran Song, Thomas Funkhouser, and Jianxiong Xiao. LSUN: Construction of a Large-scale Image Dataset using Deep Learning with Humans in the Loop. *arXiv e-prints*, page arXiv:1506.03365, 2015.

[YYR+20] Wenhan Yang, Ye Yuan, Wenqi Ren, Jiaying Liu, Walter J. Scheirer, Zhangyang Wang, Taiheng Zhang, Qiaoyong Zhong, Di Xie, Shiliang Pu, Yuqiang Zheng, Yanyun Qu, Yuhong Xie, Liang Chen, Zhonghao Li, Chen Hong, Hao Jiang, Siyuan Yang, Yan Liu, Xiaochao Qu, Pengfei Wan, Shuai Zheng, Minhui Zhong, Taiyi Su, Lingzhi He, Yandong Guo, Yao Zhao, Zhenfeng Zhu, Jinxiu Liang, Jingwen Wang, Tianyi Chen, Yuhui Quan, Yong Xu, Bo Liu, Xin Liu, Qi Sun, Tingyu Lin, Xiaochuan Li, Feng Lu, Lin Gu, Shengdi Zhou, Cong Cao, Shifeng Zhang, Cheng Chi, Chubing Zhuang, Zhen Lei, Stan Z. Li, Shizheng Wang, Ruizhe Liu, Dong Yi, Zheming Zuo, Jianning Chi, Huan Wang, Kai Wang, Yixiu Liu, Xingyu Gao, Zhenyu Chen, Chang Guo, Yongzhou Li, Huicai Zhong, Jing Huang, Heng Guo, Jianfei Yang, Wenjuan Liao, Jiangang Yang, Liguo Zhou, Mingyue Feng, and Likun Qin. Advancing Image Understanding in Poor Visibility Environments: A Collective Benchmark Study. *Transactions on Image Processing*, 29:5737–5752, 2020.