FACULDADE DE ENGENHARIA DA UNIVERSIDADE DO PORTO

BlanketGen: Synthetic Blanket Occlusion Generation for the Augmentation of Motion Capture Datasets

João Carmona

Mestrado Integrado em Engenharia Eletrotécnica e de Computadores

Supervisor: João Paulo Cunha

November 17, 2022

Resumo

Introdução: Epilepsia é uma doença neurológica que afeta dezenas de milhões de pessoas globalmente. O diagnóstico geralmente requere a avaliação da semiologia de crises epilépticas. Esta semiologia é baseada na interpretação subjetiva por parte de epileptologistas dos movimentos que acontecem durante a crise. A automatização nesta área seria útil para melhorar a eficácia e eficiência desta análise da semiologia. Nesta linha de investigação e desenvolvimento, a automatização não invasiva sem contacto com o corpo do doente tem vindo a utilizar sistemas baseados em visão computacional, normalmente baseados em aprendizagem computacional profunda (DL). Uma das abordagens mais comuns para este tipo de sistemas envolve a utilização de um sistema de *human pose estimation* (HPE), que deteta automaticamente as poses dos sujeitos, para depois trabalhar com os resultados desse sistema em vez das imagens originais. No entanto, sistemas de HPE foram maioritariamente estudados e desenvolvidos para uso em situações em que os sujeitos estão de pé, sem qualquer oclusão relativamente às câmaras vídeo e não deitados e ocultos por cobertores, como acontece na análise de semiologia de crises epilépticas. Desta forma, a oclusão por cobertores é o principal tipo de cenário que impede a utilização de sistemas de visão computacional DL pré-treinados e disponíveis na literatura. Assim, esta dissertação explora o potencial de aumentar os *datasets* existentes que não contenham oclusões de cobertores com oclusões de cobertores geradas por computador (CG) na melhoria do desempenho de sistemas de HPE que utilizem *deep learning*.

Métodos: Foi implementado um *pipeline* para aumentar artificialmente com oclusões de cobertores simulados um conhecido e muito usado *dataset* da literatura (3DPW) que inclui vídeos RGB de pessoas em cenários normais com o correspondente *ground truth* da posição e forma do corpo. O *dataset* resultande - a que chamamos "BlanketGen-3DPW" - foi depois usado para treinar um sistema de HPE com *deep learning* que foi avaliado e comparado quantitativamente com o mesmo sistema treinado apenas no *dataset* original. Para além disso, foi criado um novo *dataset* - a que chamamos "BlanketSet" - com gravações de vídeos 3D (RGBD) de pessoas deitadas numa cama de uma Unidade de Monitorização de Epilepsia realizando vários movimentos, tanto com e sem oclusões de cobertores reais. Este *dataset* foi utilizado para a avaliação qualitativa dos resultados.

Resultados: Um novo modelo DL de HPE treinado com o nosso BlanketGen-3DPW obteve um melhor desempenho perante um conjunto de dados que incluiam exemplos de cenas com e sem oclusões de cobertores simulados (52.46 PA-MPJPE vs 52.70). Como esperado, este novo modelo teve um desempenho pior quando avaliado no dataset original 3DPW (sem oclusões)(50.95 PA-MPJPE vs 44.97). Estes resultados concordam com os resultados da avaliação qualitativa elaborada com o BlanketSet, na qual o novo modelo obteve melhores resultados nos vídeos em que o corpo inteiro do participante estava coberto por um cobertor (0.2 ± 0.15).

Conclusões: Aumentar datasets com oclusões de cobertores CG poderá ser uma abordagem viável para melhorar sistemas de DL HPE em cenários onde cobertores tendem a cobrir os sujeitos - como nos cenários hospitalares, mas há necessidade de investigar abordagens para evitar reduzir a precisão quando não há oclusões.

Palavras-chave: estimação de pose humana, motion capture, oclusões sintéticas, simulação de tecido, deep learning, aumento sintético de datasets

Abstract

Introduction: Epilepsy is a neurological disorder that affects tens of millions of people worldwide. Proper diagnosis can include the analysis of the semiology of epileptic seizures. This analysis is based on the subjective interpretation by epileptologists of the movements that occur during recorded seizures. Automatization in this area would be helpful to improve the accuracy and efficiency of the semiology analysis. Non-intrusive contact-less automation relies on characterization of the movements during seizures with vision-based systems. A promising approach is to first use a human pose estimation (HPE) system and then work with its output instead of the original video. However, HPE has mostly been studied and developed for use cases where the subjects are standing up and are unoccluded, whereas in the case of the analysis of semiology of epileptic seizures the subjects are usually lying down in hospital beds and are often occluded by blankets. This dissertation explores the potential of augmenting datasets without blanket occlusions with computer-generated (CG) blanket occlusions in improving the performance of HPE deep learning systems.

Methods: A pipeline was implemented to augment the 3DPW dataset with CG blanket occlusions, this dataset was named BlanketGen-3DPW. The 3DPW dataset includes RGB videos of people in normal everyday scenarios with body mesh and position ground truths. The augmented dataset was used to train an HPE deep learning system, which was then evaluated and compared quantitatively to the same system trained only on the original dataset. Besides this, a dataset which was named BlanketSet was created with RGB-D recordings of people in an Epilepsy Monitoring Unit bed performing various movements both with and without real blanket occlusions. This dataset was used for qualitative evaluation of the results.

Results: A model fine-tuned on BlanketGen-3DPW underperformed when evaluated on the original 3DPW dataset compared to a model without the fine-tuning (50.95 PA-MPJPE vs 44.97), but provided slightly better results on BlanketGen-3DPW (52.46 PA-MPJPE vs 52.70), which includes people with and without synthetic blanket occlusions. These results were corroborated by the results of the qualitative evaluation on BlanketSet, on which there was a statistically significant performance improvement on videos with full-body blanket occlusions (0.20±0.15).

Conclusions: Augmenting datasets with synthetic blanket occlusions could be a viable approach to improve DL HPE systems in scenarios where blanket occlusions are common, but more research needs to be done to address the downgrade in accuracy in situations where blanket occlusions are not present.

Keywords: human pose estimation, motion capture, synthetic occlusions, cloth simulation, deep learning, dataset augmentation

Acknowledgements

First I would like to thank my supervisor Professor João Paulo Cunha for his guidance and for coordinating C-BER (and for telling Dr.Ricardo to answer my emails), as well as my pseudo-supervisor Tamás Karácsony for meeting with me weekly, for coming up with the idea for BlanketGen, and for pushing me to keep up with deadlines. Duarte Dias for helping me settle down in the lab and figure out the bureaucracy necessary to record a dataset at a hospital during a pandemic, as well as all the members of C-BER with whom I interacted throughout the semester. Jiefeng Li for helping me work with HybrIK and even updating the github to address some of the issues I had.

I would also like to express my gratitude to all the friends with whom I hung out, online and offline, for helping me handle the stress of looming deadlines. And the friends who put up with me randomly calling them on Discord, often at unreasonable hours, to complain about how I spent hours trying to troubleshoot some issue only to find out the solution was actually very easy (or worse, extremely difficult).

I'm thankful to my parents and my sister for their support and for not complaining about the constantly shifting final deadline of this dissertation forcing them to redo our holiday plans over and over again. To my brother who I only realized was such a positive presence in my life this past month while he was away.

I'm grateful to Dr.Ricardo Rego for letting me record BlanketSet at the EMU, and all the staff who helped me while I was at the hospital.

Finally I'd like to thank all of the people who showed up to record BlanketSet. They chose to spend an hour in a hospital bed doing what effectively amounted to a workout routine during some of the hottest days of the year when they could have gone to the beach instead.

João Carmona

"While it is always best to believe in oneself,
a little help from others can be a great blessing."

Uncle Iroh

Contents

List of Figures

List of Tables

List of Abbreviations

3DPW	3D Poses in the Wild
AGORA	Avatars in Geography Optimized for Regression Analysis
AUC	Area Under the PCK-Threshold Curve
BIP	Blanket Initial Position
BRDF	Bidirectional Reflectance Distribution Function
CAESAR	Civilian American and European Surface Anthropometry Resource
CG	Computer-Generated
CHUSJ	Hospital Center of São João
CNN	Convolutional Neural Networks
DL	Deep Learning
DNN	Deep Neural Networks
EEG	Electroencephalogram
EMS	Electromagnetic Measurement System
EMU	Epilepsy Monitoring Unit
FEUP	Faculty of Engineering of the University of Porto
GPS	Geodesic Point Similarity
GPU	Graphics Processing Unit
HPE	Human Pose Estimation
IMU	Inertial Measurement Unit
IPS	Image Processing System
IR	Infrared
JPG	Joint Photographic Experts Group
LSTM	Long Short-Term Memory
MPJAE	Mean Per Joint Angle Error
MPJPE	Mean Per Joint Position Error
MS COCO	The Microsoft Common Objects in Context
NN	Neural Network
OMS	Optoelectronic Measurement System
PA-MPJAE	Procrustes-Aligned Mean Per Joint Angle Error
PA-MPJPE	Procrustes-Aligned Mean Per Joint Position Error
PCA	Principal Component Analysis
PCK	Percentage of Correct Keypoints
ReLU	Rectified Linear Unit
RGB	Red, Green, and Blue
RGB-D	Red, Green, Blue, and Depth
RNN	Recurrent Neural Network
SDK	Software Development Kit
SMPL	Skinned Multi-Person Linear
USA	United States of America

Chapter 1

Introduction

1.1 Context

Human motion analysis has always been a topic of interest for academics, but up until fairly recently computer-powered automatic systems for it weren't viable. With the processing power technological boom of the past decades, automatic systems for motion analysis have been improving drastically, well into the realm of viability. Due to this, automatic human motion analysis has become a hot topic in several areas of research.

One such area is the semiology of epileptic seizures: accurate and effective diagnosis and classification of epilepsy requires visiting an Epilepsy Monitoring Unit (EMU), where the patients are monitored with video-electroencephalogram (video-EEG) systems; the outputs of these systems during epileptic seizures are then subjectively analyzed by epileptologists. Human motion analysis can be used to aid the epileptologists with quantitative results [10][11].

Human Pose Estimation (HPE) is a task within the broader concept of human motion analysis which focuses exclusively on the position of the subjects being analyzed at any given moment. Different approaches have been studied to tackle the task of HPE, from the use of inertial measurement units (IMUs) to video-based systems [9]. For the specific use-case of seizure semiology in EMUs, IMUs aren't practical as they are uncomfortable and patients sometimes need to stay for considerable periods of time, they are also difficult to keep properly attached during violent seizures, so video-based systems are the norm.

However, most of the research effort invested into HPE has been focused on the most common scenarios of subjects standing up and moving either inside or outside with few occlusions and with variable camera angles [12]; whereas the scenario considered in this dissertation has the subjects lying down in beds, usually with blankets covering them at least partially, and a fixed camera. The blanket occlusions in particular are of concern since accurate estimation of occluded joints is especially difficult. Blanket occlusions do provide some information as to the location of the occluded joints, but current state-of-the-art systems don't take advantage of this information, as they generally make use of Deep Learning (DL) which is extremely reliant on the datasets used for training and blanket occlusions aren't present in any dataset used by these systems [12].

In order to allow DL HPE systems to make use of the information hidden in blanket occlusions, this dissertation proposes BlanketGen, a pipeline to augment a dataset with computer-generated (CG) blanket occlusions. The augmented dataset was used to train a DL HPE system so that the improvement brought by the augmentation could be analyzed.

1.2 Goals and Objectives

The goal of this dissertation is to study CG blanket occlusion augmentations as a potential approach to aid DL HPE systems in scenarios where blanket occlusions are present.

In order to fulfill this goal the following steps were taken:

- Technical background study
- State of the art review
- Development of a python script that loads the 3DPW dataset [13] onto blender, simulates a blanket on top of a subject, then renders the scene.
- Acquisition of BlanketSet, a dataset with real blanket occlusions at the EMU of the University Hospital Center of São João (CHUSJ)
- Training of the DL HPE system HybrIK [14] with and without the augmented dataset to evaluate the results.

1.3 Structure of the Dissertation

This dissertation has 7 more chapters besides the Introduction.

In chapter 2 the technical background is established which explains the background knowledge necessary to understand the work of the dissertation; it is split into 5 sections: section 1 covers Blender, which was used for the simulation and rendering of the augmentations, section 2 covers the camera model used to superimpose CG objects onto the videos properly, section 3 covers the basics of Deep Learning, section 4 covers the Azure Kinect and the Azure Kinect SDK which were used to capture BlanketSet, and section 5 covers the different measurements used to evaluate HPE systems.

Chapter 3 explores the current state-of-the-art solutions and datasets for HPE as well as previous studies that have looked specifically at CG blanket occlusion augmentations as a potential approach to improve HPE in cases where real blanket occlusions are present.

Chapter 4 explains the BlanketGen pipeline, as well as all the design decisions made and the reasoning behind them; besides this it also explains the challenges that were faced in the implementation of the pipeline and the solutions found.

Chapter 5 describes BlanketSet, the dataset acquired to help evaluate the performance improvement of training the DL HPE system on the augmented dataset. It explains the design decisions of the dataset and includes visualizations and explanations for all the movement sequences recorded in the dataset. This chapter was edited from the Data Acquisition Protocol, which was a document used to propose and explain the dataset and its acquisition plan.

Chapter 6 explains the choice of DL HPE system used as well as the methods chosen to evaluate the BlanketGen pipeline and their results. It covers the quantitative methods employed to compare the performance of DL HPE models on the original and the augmented dataset, as well as the qualitative methods used to evaluate the performance of the models on BlanketSet.

Chapter 7 comments on the results laid out in the previous chapter, it also speculates on the causes of the successes and failures of the work of the dissertation.

Chapter 8 summarizes the information of all the previous chapters and draws conclusions based on that information, including a section about the work to be done in the future.

Chapter 2

Technical Background

2.1 Blender

Blender is a free and open-source program for creating 3D computer graphics, it provides all of the tools necessary for 3D animation.

It was chosen for the implementation of BlanketGen over other 3D computer graphics programs due to being free, having a robust cloth simulation implementation, and providing a python API that supports nearly every capability of the program.

2.1.1 Python in Blender

The default installation of Blender includes its own python distribution as well as an IDE which can be accessed directly from Blender. Scripts can be run from this IDE, which allows for rapid testing and development.

In order to interact with Blender through python code, the Blender Python API must be used, this package is included in the python distribution provided.

The Blender Python API provides extensive control over all the different functionalities of Blender, so it includes an extremely large set of methods, classes, and functions.

To help with using such a complex API a few tools are provided:

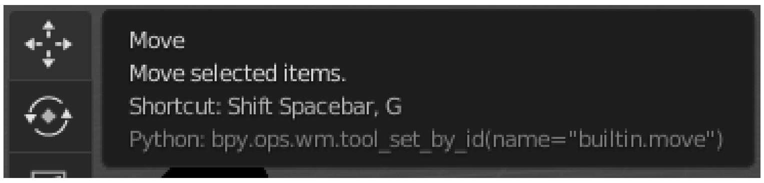
1. Official documentation [15]

2. On-hover tooltips on nearly every button, slider, textbox, etc, with the data-path of what is being hovered (figure 2.1)

3. A python console with auto-complete suggestions (figure 2.2)

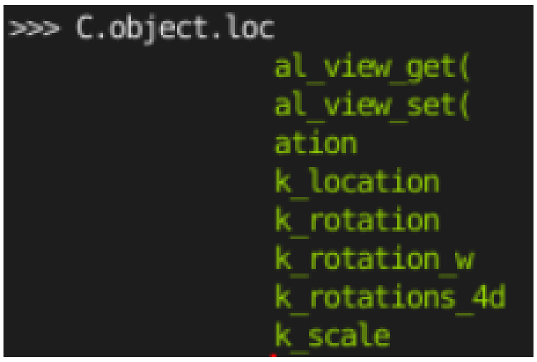
4. A window that displays python code which would have the same effect as each action taken within the GUI (figure 2.3).

These tools prove themselves invaluable when working with Blender Python, since it has its own very particular structure. This chapter is an attempt at a brief overview of this structure.



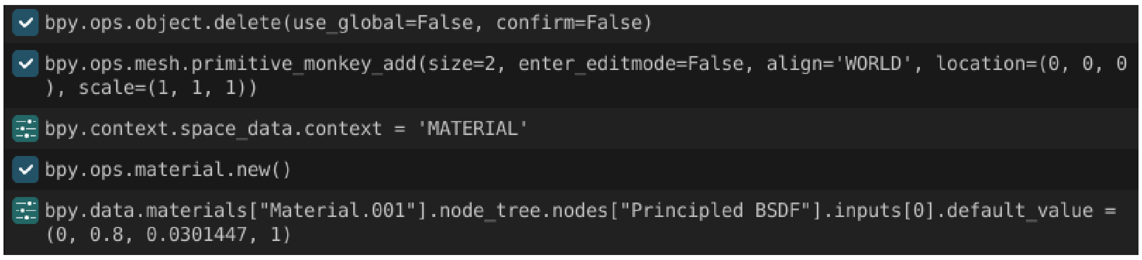
```
Move
Move selected items.
Shortcut: Shift Spacebar, G
Python: bpy.ops.wm.tool_set_by_id(name="builtin.move")
```

Figure 2.1: An example on-hover tooltip, the data-path for this button is provided in the last line



```
>>> C.object.loc
                al_view_get(
                al_view_set(
                ation
                k_location
                k_rotation
                k_rotation_w
                k_rotations_4d
                k_scale
```

Figure 2.2: An example of auto-complete suggestions from the built-in python console



```
✓ bpy.ops.object.delete(use_global=False, confirm=False)
✓ bpy.ops.mesh.primitive_monkey_add(size=2, enter_editmode=False, align='WORLD', location=(0, 0, 0
  ), scale=(1, 1, 1))
  bpy.context.space_data.context = 'MATERIAL'
✓ bpy.ops.material.new()
  bpy.data.materials["Material.001"].node_tree.nodes["Principled BSDF"].inputs[0].default_value =
  (0, 0.8, 0.0301447, 1)
```

Figure 2.3: Python code that replicates the action of deleting the initial cube, adding a monkey, and changing its material.

It's important to note that in Blender some terms have specific meanings that don't necessarily match the meaning usually ascribed to them. In this dissertation, terms are displayed in italics when they're using Blender's definition.

A glossary with definitions of terms as they are used in Blender can be found in [16].

2.1.2 Context

In Blender Python, *context* is a class which describes the state of the entire project at any given moment. This includes information about the currently selected *object*, the current active window, the current scene, etc.

2.1.3 Operators

Operators in Blender Python are methods that interact with the project in some way, be it by transforming a given object, by adding an animation *keyframe*, etc.

Operators take a *context* object as an input, by default the current *context* of the project. Some operators use the current window selection from the *context*. This proved to be a recurring problem, as the BlanketGen pipeline was at every point developed to be run from the command line in headless mode, which doesn't open up the GUI, and thus, doesn't have any windows to even be selected. The pipeline ended up only being run in scenarios where a GUI was permitted, but this was not guaranteed from the start, so operators were usually avoided wherever possible.

2.1.4 Translation and Rotation

In Blender Python, translation and rotation transformations can be either performed directly (i.e. by changing the location values of a given object) or through an operator. For the reason explained in the previous subsection, operators were avoided in BlanketGen.

Each object in the scene has a *world matrix* defining its rotation and position in relation to the global axes, this matrix has the following format:

$$M_w(\theta_x,\theta_y,\theta_z,x,y,z)=\begin{bmatrix}\cos\theta_y\cos\theta_z & \cos\theta_z\sin\theta_x\sin\theta_y-\cos\theta_x\sin\theta_z & \sin\theta_x\sin\theta_z+\cos\theta_x\cos\theta_z\sin\theta_y & x\\ \cos\theta_y\sin\theta_z & \cos\theta_x\cos\theta_z+\sin\theta_x\sin\theta_y\sin\theta_z & \cos\theta_x\sin\theta_y\sin\theta_z-\cos\theta_z\sin\theta_x & y\\ -\sin\theta_y & \cos\theta_y\sin\theta_x & \cos\theta_x\cos\theta_y & z\\ 0 & 0 & 0 & 1\end{bmatrix}$$

where $(\theta_x,\theta_y,\theta_z)$ are the Euler angles in XYZ format and (x,y,z) are the global coordinates.

To apply a rotation and/or translation transformation to an object, a matrix of the transformation can be calculated with the same format and then the transformation can be applied by multiplying the *world matrix* of the object with the transformation matrix.

Both rotation and translation transformations can also be applied by altering other attributes of the object. Translations by altering the global coordinates attribute, and rotations by altering the rotation attribute of the object, with the supported formats being Euler angles, quaternion, and axis angles.

2.1.5 Image Rendering

Image rendering is any process used to generate a 2D image. In the context of this dissertation, it refers specifically to the task of generating realistic 2D images from a 3D scene as if they were captured by a camera.

Blender provides 2 different rendering engines called Eevee and Cycles; the main differences between these engines are the time they take to render an image with passable quality and the quality of said image.

Eevee renders images much faster than Cycles, but images rendered by it are much less accurate than images rendered by Cycles (as seen in figure 2.4). This is because Eevee uses a technique called rasterization, whereas Cycles uses raytracing instead.

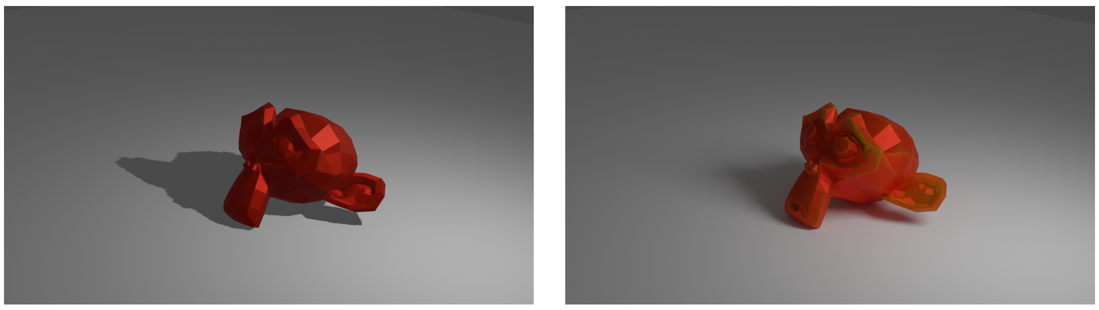


Figure 2.4: The same scene rendered by Eevee (on the left) and Cycles (on the right), note the red indirect lighting below the monkey, as well as the smooth shadow and the yellow subsurface scattering

2.1.5.1 Rasterization

Rasterization is a process for rendering that projects all the triangles in the scene into the 2D screen (illustrated in figure 2.5) then figures out which ones are visible and which aren't based on their distance to the camera, this step is called back-face culling. The color of each pixel is then calculated by the shader of the *material* of the *object*. The simplest operation that shaders can implement is a light level calculation, this is usually by scaling the light level with the cosine of the angle between the normal of the triangle and the direction of the light.

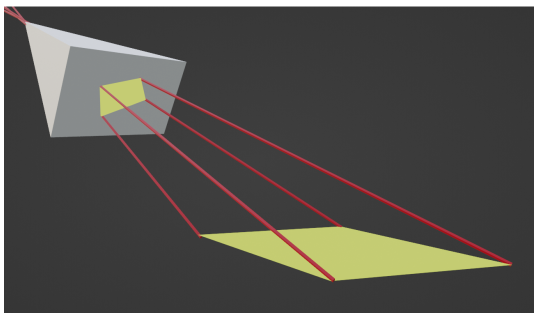


Figure 2.5: An illustration of projecting an object from 3D space to a 2D camera plane.

To create shadows, the same process used for back-face culling is done from the perspective of every light to create a map of which areas are hit by the light and which aren't.

Rasterization is very fast since for every pass it only needs to iterate through the vertices and through the pixels, however, it doesn't include many physical phenomena that affect real images taken by real cameras.

One such phenomenon is indirect lighting: in real life, light bounces off of materials so objects can light up other nearby objects even if they don't emit light.

Other phenomena include subsurface scattering, which refers how light scatters inside objects instead of only interacting with their surface, and smooth shadows caused by non-point lights.

2.1.5.2 Raytracing

Raytracing is a rendering process based on light ray simulations [1]. At its core, it attempts to solve the render equation:

$$L_o(P,\omega_o) = \int_{S^2} f(P,\omega_o,\omega_i)L_i(P,\omega_i)|\cos\theta_i|d\omega_i$$

Where L_o is the radiance leaving the surface at point P in direction ω_o, and the surface property f is the bidirectional reflectance distribution function (BRDF). L_i is the incoming light along direction ω_i and the angle between the surface normal and the incoming light direction is θ_i.

The BRDF is a function that defines the distribution of the light reflected off of a material in every direction. Different materials have different BRDFs, matte materials reflect light equally in all directions regardless of the direction of the source, whereas mirrors reflect all of the light in a single direction (this is illustrated in figure 2.7).

L_i requires solving the same equation for every surface point visible from the point being calculated, which means that the formula must be solved recursively.

In practice, this integral is usually approximated using the Monte Carlo method. Multiple rays are cast through every pixel and all their interactions with the scene are simulated up until they either reach a light or bounce a specified number of times (illustrated in figure 2.6). Rays can cast their own rays and usually do upon colliding with materials according to the BDRF of the material at the point of the collision, this way the radiance at the point of the collision is also approximated.

The number of rays cast at each step defines how long the image takes to render as well as the overall quality of the render, with more rays increasing both the quality and the render time.

The maximum number of bounces also affects both the render time and quality of the render, but the quality increase gained by increasing it diminishes very quickly if there are no mirror objects in the scene.

Because a Monte Carlo approximation is used, the pixels all end up with random noise, however, adjacent pixels usually have similar colors so denoising can be done, for example with a low pass filter, to improve the accuracy of the final result.

2.1.6 Cloth Simulation

Blender uses a spring network model for cloth simulation. Between each pair of adjacent vertices there's a tension spring and a compression spring, between non-adjacent vertices of every face a shear spring, between adjacent faces angular bending springs (as seen in figure 2.8).

Springs are modeled linearly:

$$F = -kd$$

The physical simulation of the cloth, including the springs as well as the internal and external collisions, is done in time steps, and the length of these steps is a parameter that can be set. Longer time steps provide worse results but simulate faster. The number of vertices in the cloth can also be controlled, with the quality and time it takes to simulate increasing with the number of vertices.

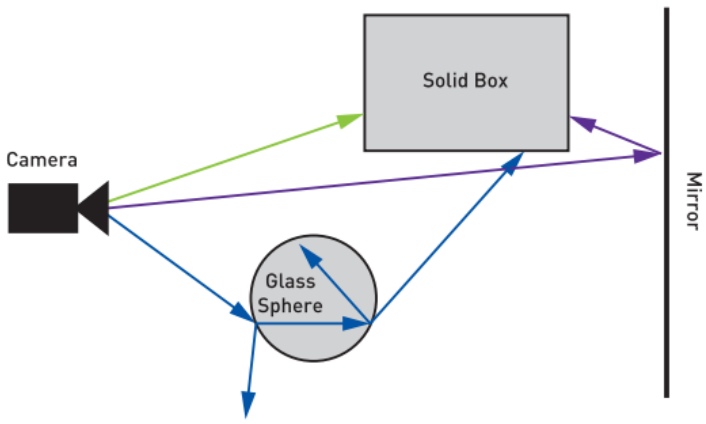


Figure 2.6: An illustration of rays being cast from a camera from [1]. It is somewhat incomplete as it doesn't include a light: none of these rays would find a light so the image would be completely black.

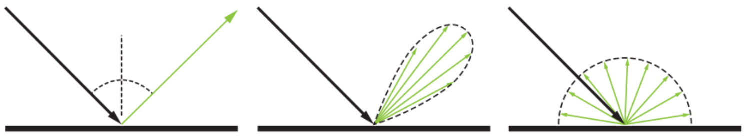


Figure 2.7: A visualization of the BDRF of different materials from [1]. Respectively, a mirror, glossy, and diffuse material.

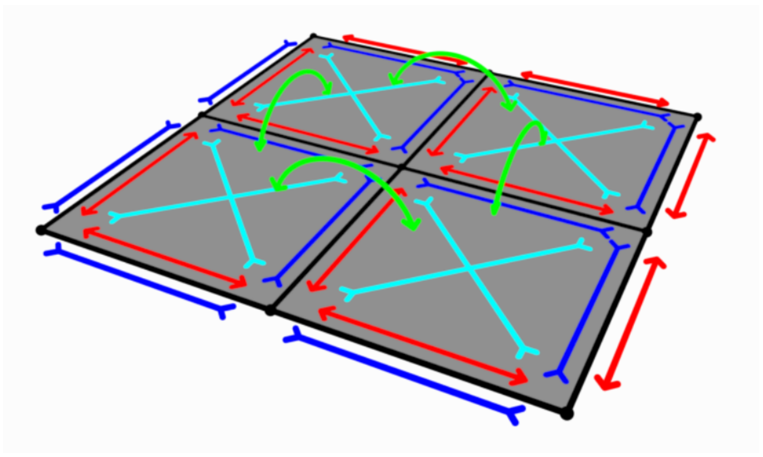


Figure 2.8: An illustration of cloth springs from [2]; tension springs (blue), compression springs (red), shear springs (cyan), and angular bending springs (green).

Generally, the biggest inaccuracies that can come from these trade-offs are missed collisions: if the time steps are too long it's possible for a vertex to phase through a face or vertex without a collision being detected if the vertex is moving fast enough. Missed collisions are a big issue and a single one can completely ruin a simulation since once the vertex has phased through something it was supposed to collide with, it can usually only phase back out to where it should be if it gathers enough velocity, which rarely happens.

In practice, a single missed collision tends to lead to many other missed collisions, which themselves also lead to more missed collisions. This escalates into a completely inaccurate simulation (as seen in figure 2.9).

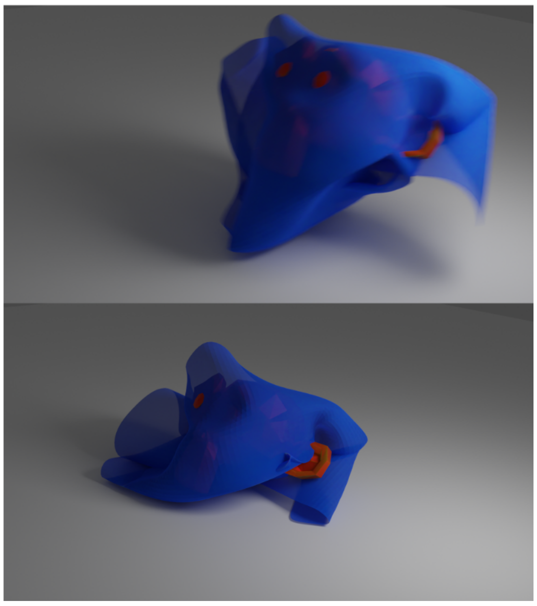


Figure 2.9: A couple of renders from a simulation; the monkey phased through the cloth while it was moving quickly, and even after it stopped moving the simulation never recovered from the error.

2.2 Camera Model and Parameters

Cameras create 2D images from a 3D world, this process can be described as a series of linear transformations [17]. This model is based on pinhole cameras and thus assumes no lens distortions. Despite this, it's still fairly accurate and is the model used to describe the camera parameter in most datasets.

$$\begin{bmatrix} x^s \\ y^s \\ 1 \end{bmatrix} = P \begin{bmatrix} X \\ Y \\ Z \\ 1 \end{bmatrix}$$

$$P = KR[I_3|-\mathbf{X_O}]$$

Where K is the intrinsic matrix, R is the rotation matrix, and $\mathbf{X_O}$ is the location vector of the principal point.

The projection matrix P contains 11 parameters and maps positions in the 3D world to the 2D image plane.

2.2.1 Extrinsic Parameters

Extrinsic parameters are the parameters that relate to the position of the camera in the 3D world. There are a total of 6 extrinsic parameters, 3 for the location of the camera and 3 for its orientation.

These parameters form two matrices, one for rotation and another for location.

$$R = \begin{bmatrix} \cos\theta_y \cos\theta_z & -\cos\theta_y \sin\theta_z & \sin\theta_y \\ \cos\theta_x \sin\theta_z + \cos\theta_z \sin\theta_x \sin\theta_y & \cos\theta_x \cos\theta_z - \sin\theta_x \sin\theta_y \sin\theta_z & -\cos\theta_y \sin\theta_x \\ \sin\theta_x \sin\theta_z - \cos\theta_x \cos\theta_z \sin\theta_y & \cos\theta_z \sin\theta_x + \cos\theta_x \sin\theta_y \sin\theta_z & \cos\theta_x \cos\theta_y \end{bmatrix}$$

With $(\theta_x, \theta_y, \theta_z)$ being the XYZ Euler angles.

$$[I_3|-\mathbf{X_O}] = \begin{bmatrix} 1 & 0 & 0 & -X_O \\ 0 & 1 & 0 & -Y_O \\ 0 & 0 & 1 & -Z_O \end{bmatrix}$$

With X_O, Y_O, and Z_O being the 3D coordinates of the camera.

2.2.2 Intrinsic Parameters

Intrinsic parameters are used to project the points into the image plane. There are a total of 5 intrinsic parameters. but for digital cameras the skew is usually 0.

These parameters form the K matrix:

$$K = \begin{bmatrix} c & cs & p_x \\ 0 & c(1+m) & p_y \\ 0 & 0 & 1 \end{bmatrix}$$

Where c is the distance between the center of projection O and the principal point H, s is the skew (0 for rectangle pixels), m is the scale in the x and y axes, and (p_x, p_y) are the coordinates of the principal point (the point of the pinhole in a pinhole camera).

2.3 Deep Learning

Deep learning is a subsection of machine learning which utilizes neural networks with several layers.

While the theory behind deep neural networks (DNNs) was first researched decades ago, they have only gained popularity more recently. This is because the computational power required for them wasn't feasible for most use-cases, but with recent advances in computing, particularly in devices specialized for massively parallel computing such as GPUs and TPUs, it has become less of a bottleneck.

Besides computational power, DNNs also require very large datasets to avoid over-fitting and the datasets have to be diverse enough for the DNN to be able to generalize and perform well in real-world situations.

As DNNs became more advanced, they started seeing usage in new fields. Convolutional neural networks (CNNs) in particular have become central to computer vision [18].

2.3.1 Neural Networks

A neural network is essentially a directed graph of nodes (as represented in figure 2.10) where the nodes themselves execute some operation given their inputs and pass their outputs to the next nodes, which take them as inputs [3].

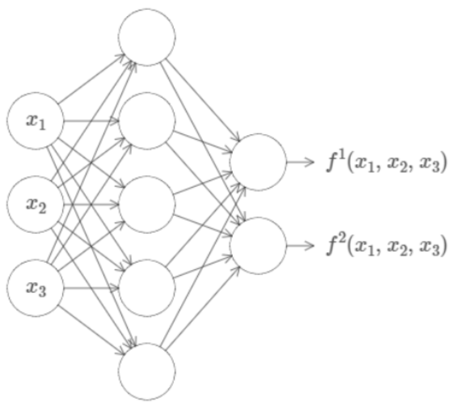


Figure 2.10: Example of a neural network from [3]

NNs generally have 2 sets of parameters to be optimized, the weights and the biases. The weights scale the output of each node before passing it to the input of the next node, there's a weight for every edge in the network. The biases are added to the output of each node, there's one for every node in the network.

The functions of the nodes are called activation functions. Common activation functions are the rectified linear unit (ReLU) function and the sigmoid function (these functions are shown in figure 2.11).

$$ReLu : f(x) = max(0, x)$$

$$Sigmoid : f(x) = \frac{1}{1 + e^{-x}}$$

With this information, the formula for the output of the j^{th} node in the l^{th} layer with the activation function f(x) can be defined:

$$a_j^l = f\left(\sum_k w_{jk}^l a_k^{l-1} + b_j^l\right)$$

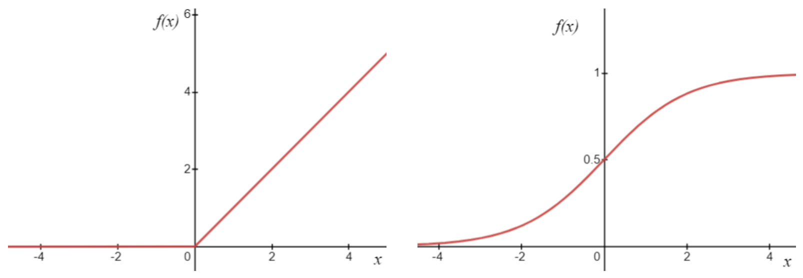


Figure 2.11: ReLU activation function on the left and sigmoid on the right

A cost function C is chosen to evaluate the network. Optimizing the network is done by minimizing this cost function. To minimize the cost an algorithm based on gradient descent can be used. This algorithm consists of iteratively updating the parameters in the direction opposite that of the gradient. This means each parameter takes a step negatively proportional to its partial derivative.

$$w_{k+1} = w_k - \eta \frac{\partial C}{\partial w_k}$$

$$b_{k+1} = b_k - \eta \frac{\partial C}{\partial b_k}$$

Where η is the step size hyper-parameter. Repeatedly taking steps like this gradually minimizes the cost function and hopefully finds a minimum of the cost function, although it's not guaranteed to be the global minimum.

In order to accelerate the calculation of these partial derivatives, an algorithm called back-propagation was invented which takes advantage of the chain rule: since the layers of the network are connected, we can store the partial derivatives of each layer and use them in the calculation of the partial derivatives of the next layer, this avoids repeating calculations.

In order to train a NN, first the training dataset is split into batches. Then the batches are iterated through, with the cost and the gradient of the cost being calculated at every iteration and the trainable parameters being adjusted accordingly. Iterating once through all the batches of the training dataset is called an epoch; NNs are usually trained for several epochs, usually with different hyper-parameters, such as the learning rate. To keep track of the performance of the NN throughout the training a validation dataset can be used to evaluate the NN repeatedly as it trains.

After training is complete, a test dataset on which the NN hasn't been trained or evaluated is used to evaluate the performance of the NN.

2.3.2 Convolutional Neural Networks

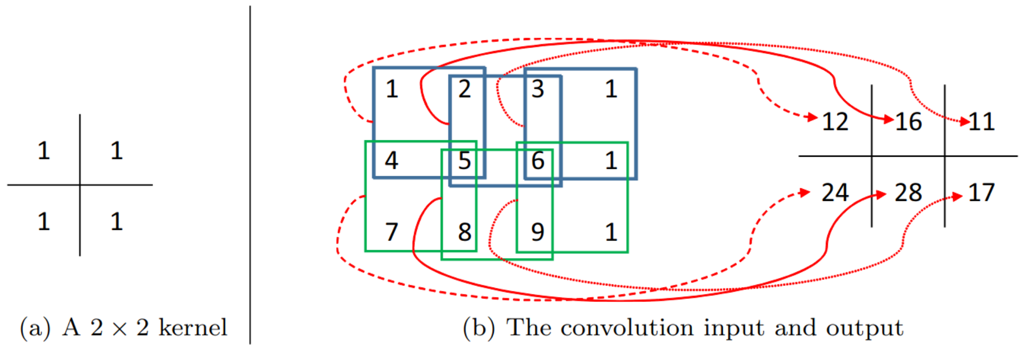
Convolutional neural networks (CNNs) are a type of neural network which uses convolutional layers, which are layers where the function executed by each node is a convolutional filter.

Convolutional filters perform the convolution of a kernel with the image to be processed.

Output of the convolution of an image I with a kernel K of size (m,n) at pixel (i,j) is given by

$$(I * K)_{i,j} = \sum_{a=-\lfloor \frac{m}{2} \rfloor}^{\lfloor \frac{m}{2} \rfloor} \sum_{b=-\lfloor \frac{n}{2} \rfloor}^{\lfloor \frac{n}{2} \rfloor} I_{i-a,j-b} K_{\frac{m}{2}+a,\frac{n}{2}+b}$$

A visual example of a convolution is presented in figure 2.12.



(a) A 2 × 2 kernel (b) The convolution input and output

Figure 2.12: An illustration of the convolution operation from [4]

The outputs of a convolutional layer are called activation maps.

Each convolutional layer performs the convolution of filters with the activation maps (or color channels) passed to it as inputs and the outputs of these convolutions are summed to create the output activation maps.

In order to reduce the size of activation maps, pooling layers are used, these layers break the activation map into chunks with a specified kernel size and then reduce each chunk to a single value, usually by taking either the max or the average of the chunk (illustrated in figure 2.13).

Both the convolutional layers and the pooling layers can also have a stride parameter, which defines how many pixels the kernels skip as they are iterated through the activation map.

In computer vision scenarios, the advantage of convolutional layers over fully connected layers is the massive reduction of the number of trainable parameters: a convolutional layer that takes a 200 by 200 RGB image as an input and uses it to generate an activation map with 7 by 7 kernels has 147 weights to train, whereas a fully connected layer that takes the same image and calculates the output of a single node has 120000 weights and 1 bias to train.

Convolutional layers also have the advantage of being invariant to translation: a CNN that learns to identify objects in an image, for example, is equally able to identify them regardless of where in the image they are.

This type of neural network is exceptional at tasks such as object recognition in images, in which it considerably outperforms densely connected networks [18].

2.3.3 Recurrent Neural Networks

Recurrent neural networks (RNNs) are a type of neural network developed for sequential data. On each member of the sequence the network takes a memory state as an input and produces as an output another memory state, which is then passed to the next member.

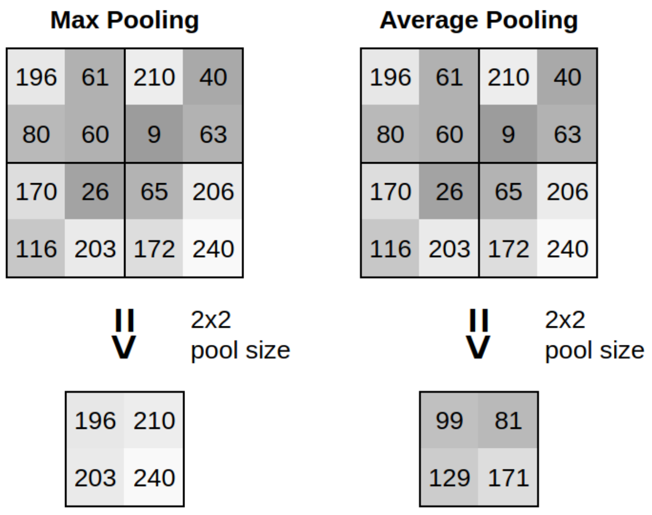


Figure 2.13: Examples of pooling operations with a 2 by 2 kernel and a stride of 2.

As seen in figure 2.14: at each step the cell takes an input x as well as a memory state h and the biases c to calculate an output o, which is then compared to the target y by the loss function L.

The weight matrix U maps the inputs to the current memory state, the weight matrix W maps the previous memory state to the current one, and the weight matrix V maps the memory state to the cell output. These weight matrices are the trainable parameters of the network.

$$a^t = b + Wh^{t-1} + Ux^t$$

$$h^t = \tanh a^t$$

$$o^t = c + Vh^t$$

$$L^t = L\left(o^t, y^t\right)$$

An issue with this approach appears during the calculation of the partial derivatives that make up the gradient of the loss function: since they're made up of long products of initially arbitrary values they tend to either explode or vanish.

To address this issue, Long Short-Term Memory (LSTM) cells, among others, were proposed [19].

LSTM cells are used in RNNs replacing the nonlinear units (referred to as h in the previous subsection).

Figure 2.15 illustrates an LSTM cell, it contains a memory cell and three gates, the input gate i, the forget gate f, and the output gate o.

These are the formulas for the components of an LSTM cell:

$$i^t = \sigma\left(W_{xi}x^t + W_{hi}h^{t-1} + W_{ci}c^{t-1} + b_i\right)$$

$$f^t = \sigma\left(W_{xf}x^t + W_{hf}h^{t-1} + W_{cf}c^{t-1} + b_f\right)$$

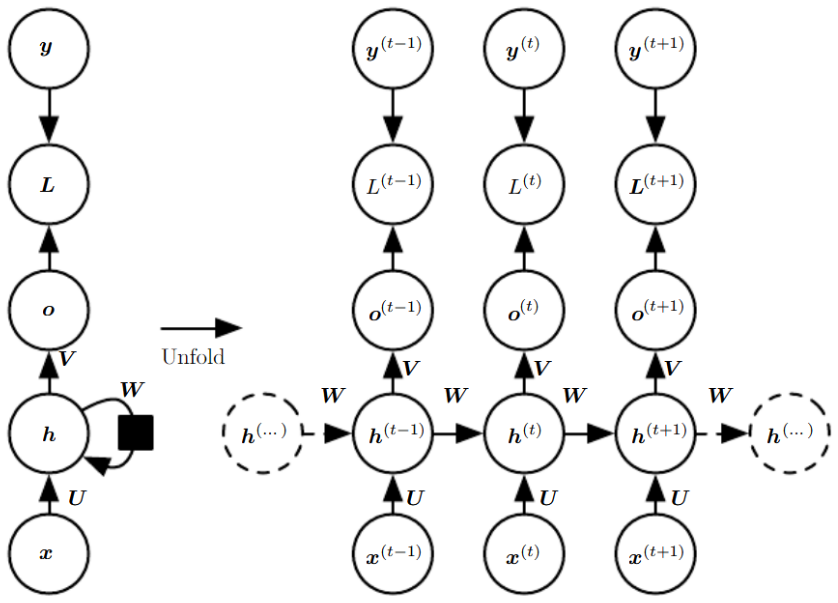


Figure 2.14: Illustration of an RNN from [5].

$$c^t = f^t c^{t-1} + i^t \tanh\left(W_{xc}x^t + W_{hc}h^{t-1} + b_c\right)$$

$$o^t = \sigma\left(W_{xo}x^t + W_{ho}h^{t-1} + W_{co}c^t + b_i\right)$$

$$h^t = o^t \tanh\left(c^t\right)$$

σ is the sigmoid function, the W matrices are the weight matrices that map units to each other and the b scalars are the biases. The weights and the biases are the trainable parameters of the network.

Since they were introduced in 1997 [19], LSTM cells have become a very powerful tool in the field of Deep Learning, especially in areas with sequential data such as natural language processing [20].

2.4 Azure Kinect

The Azure Kinect is the latest sensor in Microsoft's Kinect line. It includes an RGB camera and an infrared (IR) camera which, along with IR emitters, is used to estimate a depth map. It also includes a 7-microphone array, a gyroscope, and an accelerometer, with the latter 2 being used to track the position of the sensor (figure 2.16).

Microsoft provides a Software Development Kit (SDK) for working with the Azure Kinect, which includes applications to update the firmware, view both saved recordings and live data, and record data.

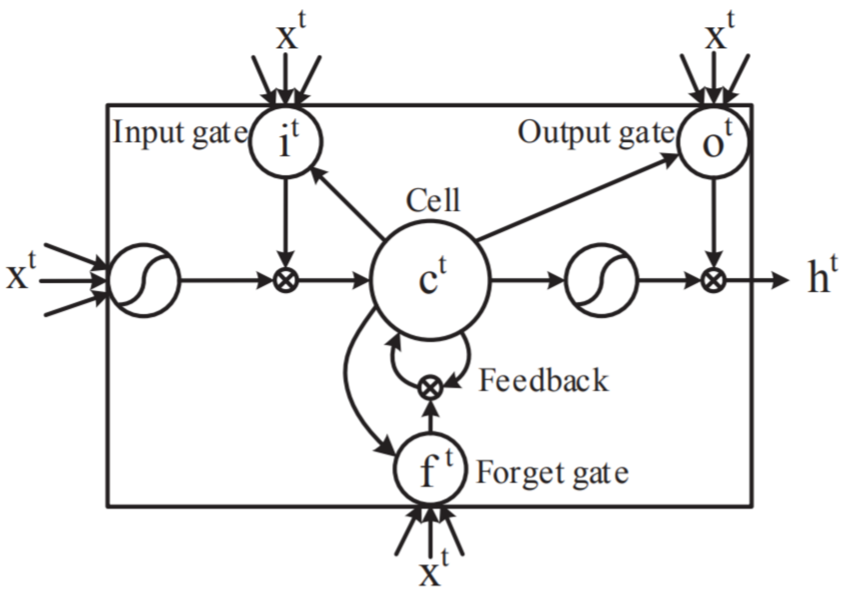


Figure 2.15: Illustration of an LSTM cell from [6].

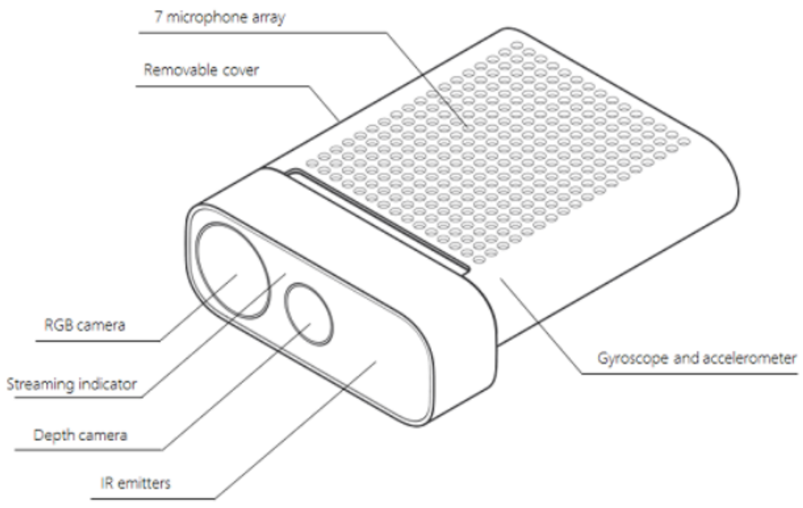


Figure 2.16: A diagram of the Azure Kinect from [7]

2.5 HPE Evaluation Measurements

There are many different ways to evaluate pose estimation and body shape estimation.

The Percentage of Correct Keypoints (PCK) is the percentage of joints within a specified distance of their correct position.

The area under the PCK-threshold curve is called AUC and is calculated by computing PCKs for a range of threshold values.

Mean Per Joint Position Error (MPJPE) is the mean Euclidean distance between the ground truth and the predicted location of each joint. It can be calculated with the following formula:

$$MPJPE = \frac{1}{N}\sum_{i=1}^{N}\sqrt{(p_{i,x}-\hat{p}_{i,x})^2+(p_{i,y}-\hat{p}_{i,y})^2+(p_{i,z}-\hat{p}_{i,z})^2}$$

where N is the total number of joints used, p_i is the ground truth position of joint i, and $\hat{p}_i$ is the estimated position of joint i.

Mean Per Joint Angle Error (MPJAE) is the error in degrees between the ground truth orientation of each body part and the prediction. It can be calculated with the following formula:

$$MPJAE = \frac{1}{N}\sum_{i=1}^{N}|\theta_i - \hat{\theta}_i|$$

where N is the number of body parts, θ_i is the angle of the body part i, and $\hat{\theta}_i$ is the predicted angle of the body part i.

Geodesic Point Similarity (GPS) is calculated with the following formula:

$$GPS = \frac{1}{|P|}\sum_{p_i \in P}\exp\frac{-d(\hat{p}_i,p_i)^2}{2\kappa(p_i)^2}$$

where $-d(\hat{p}_i,p_i)$ is the geodesic distance between estimated $\hat{p}_i$ and ground truth p_i human body surface points, and $\kappa(p_i)$ is a per-part normalization factor, defined as the mean geodesic distance between points on the part.

2.5.1 Procrustes Alignment

Procrustes alignment consists of aligning an object in space to another object by the use of linear transformations.

It can be seen as a scaling transformation, followed by a translation and then a rotation as seen in figure 2.17.

It's common when evaluating HPE systems to use Procrustes alignment first before measuring the error in the joint positions, when this is done the measurement is called Procrustes Aligned MPJPE (PA-MPJPE). When reporting the MPJAE it's also common to report it after Procrustes alignment, in which case it's called the PA-MPJAE.

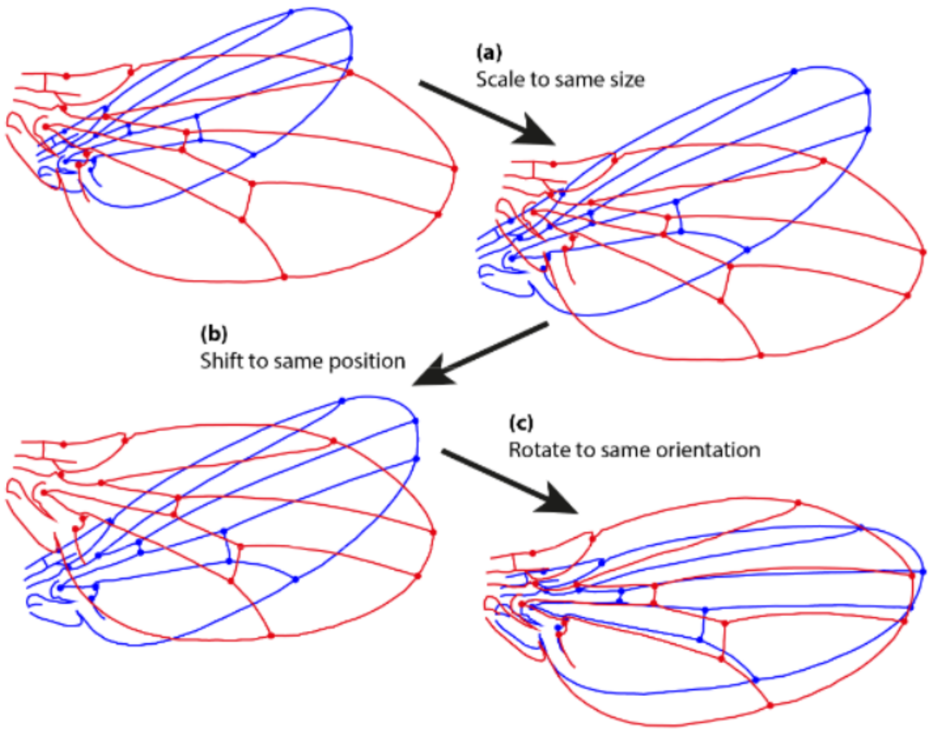


Figure 2.17: An illustration of a Procrustes alignment from [8]

Chapter 3

State of the Art

3.1 Human Pose Estimation

There are several different approaches to HPE, each having its advantages and disadvantages.

[9] is a great review of motion capture technology used in sports, although much of it is more general and its contents are relevant in other scenarios.

The four main technologies used for motion capture are as follows:

- Electromagnetic Measurement Systems (EMS) use radio transponders to communicate with fixed base stations, which estimate the location of the transponders using the time-of-flight of the radio waves [21]. EMS can function in large volumes and they do not require line-of-sight, but do not provide the best accuracy. Moreover, they're sensitive to ferromagnetic materials [22] and thus aren't ideal for scenarios where interactions with these materials are expected.

- Image Processing Systems (IPS) utilize optical cameras alongside computer vision programs to track the subject's motion. IPS require more local setup than EMS but can provide more accurate results [9]. IPS are very diverse and thus their performance, as well as their costs, vary wildly. This is both an advantage in the sense that each situation can use an IPS tailored to it, and a disadvantage in the sense that they require more setup. However, even with very minimalist setups good results have been achieved, for example in [23] exceptional results were accomplished with only a single camera and no markers.

- Optoelectronic Measurement Systems (OMS) utilize custom markers that either emit or reflect pulses of light which are then detected by fixed cameras, which use the time-of-flight of the pulses to estimate the location of the markers. OMS produce the most accurate results out of these technologies. The accuracy, however, has its costs: OMS require markers and line-of-sight between the cameras and the markers [24][25], and the cameras can be fairly expensive. Moreover, they are particularly sensitive to accidental movement of the cameras [26] and can run into issues if used outside during the day [24] as the sunlight can cause interference.

- Inertial Measurement Units (IMUs) make use of accelerometers, magnetometers, and gyroscopes. The data from the different sensors is fused to estimate the angle of the IMU [27]. Several IMUs can be attached to a subject and their relative position can be estimated by assuming the body of the subject to be rigid [28][29]. This means IMUs are not restricted in space like the other three options are, since they don't require fixed base stations or cameras, and they do not have line-of-sight constraints. Their downsides are their reliance on the sensor fusion algorithms, which can make them difficult to work with, as well as the issues inherent to the sensors used.

These approaches are neatly organized in figure 3.1.

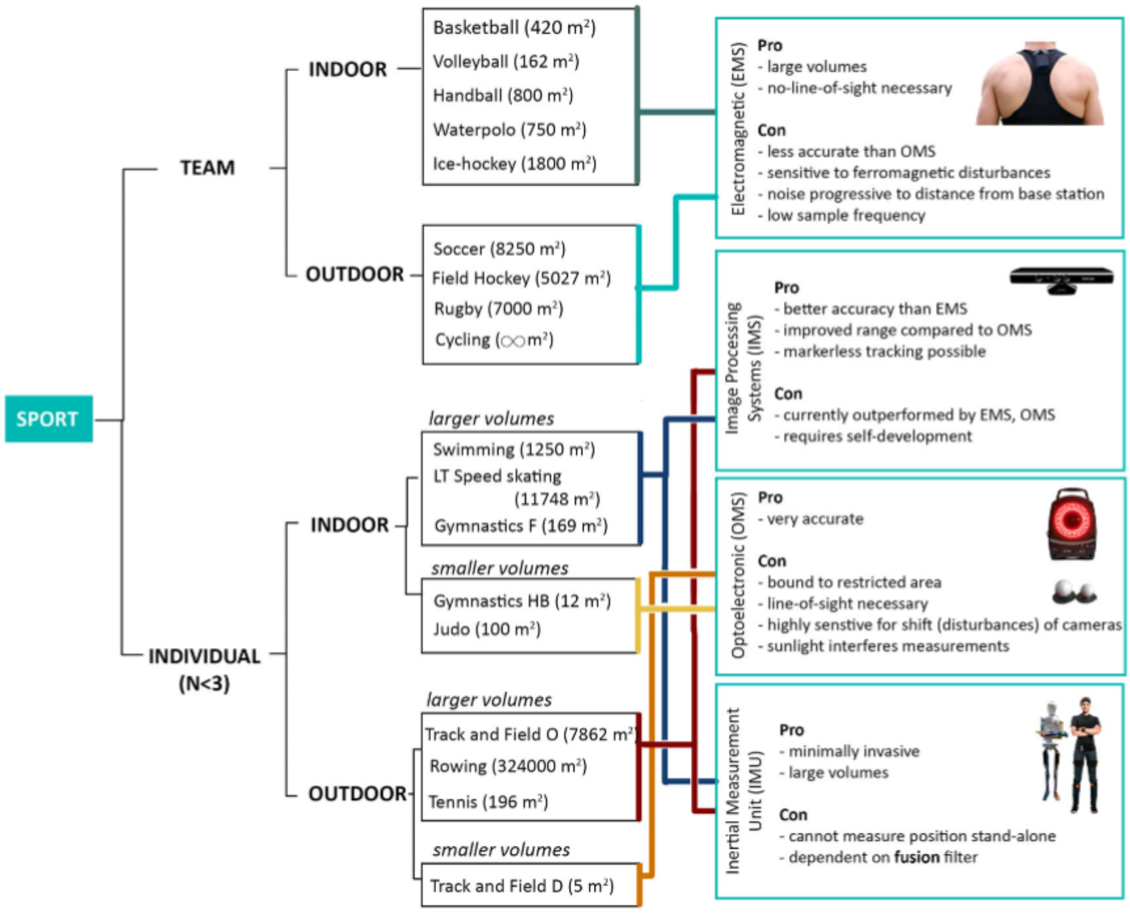

SPORT
TEAM
INDOOR
Basketball (420 m²)
Volleyball (162 m²)
Handball (800 m²)
Waterpolo (750 m²)
Ice-hockey (1800 m²)
OUTDOOR
Soccer (8250 m²)
Field Hockey (5027 m²)
Rugby (7000 m²)
Cycling (∞m²)
INDIVIDUAL (N<3)
INDOOR
larger volumes
Swimming (1250 m²)
LT Speed skating (11748 m²)
Gymnastics F (169 m²)
smaller volumes
Gymnastics HB (12 m²)
Judo (100 m²)
OUTDOOR
larger volumes
Track and Field O (7862 m²)
Rowing (324000 m²)
Tennis (196 m²)
smaller volumes
Track and Field D (5 m²)
Electromagnetic (EMS)
Pro
- large volumes
- no-line-of-sight necessary
Con
- less accurate than OMS
- sensitive to ferromagnetic disturbances
- noise progressive to distance from base station
- low sample frequency
Image Processing Systems (IMS)
Pro
- better accuracy than EMS
- improved range compared to OMS
- markerless tracking possible
Con
- currently outperformed by EMS, OMS
- requires self-development
Optoelectronic (OMS)
Pro
- very accurate
Con
- bound to restricted area
- line-of-sight necessary
- highly senstive for shift (disturbances) of cameras
- sunlight interferes measurements
Inertial Measurement Unit (IMU)
Pro
- minimally invasive
- large volumes
Con
- cannot measure position stand-alone
- dependent on fusion filter

Figure 3.1: A diagram displaying the different technologies used for HPE in sports from [9].

Out of these technologies, the ones that are most appealing for motion capture in hospital beds are IPS [11] and IMUs [30]. Image processing systems have been gaining more attention in recent years because of advancements in deep learning for computer vision.

3.1.1 Datasets

Several different datasets have been acquired that are useful for the task of HPE.

Human3.6M [31] contains 3.6 million human poses annotated with 10 high-speed motion cameras, as well as video recorded by 4 other video cameras and time of flight data recorded by a sensor placed next to one of the video cameras. 11 actors (6 male, 5 female) performed in 17 common scenarios (having an argument, talking on the phone, etc) so that the data is varied and conforms to what's common in the real world.

3D Poses in the Wild (3DPW) [13] contains video from multiple situations outdoors, each with 1 or 2 actors, recorded with a moving phone camera. Ground truth poses were obtained by combining data from the videos and IMUs, and were used to generate ground truth body shapes using the SMPL model. It also contains ground truth camera intrinsics and extrinsics so that the model can be mapped onto the screen even with a moving camera. Because the videos were recorded outside in public spaces, there are many unannotated people in them, and because the camera and the actors were moving and interacting with the world there are many occlusions. 3DPW has a total of over 51 thousand image frames.

MPI-INF-3DHP [32] contains data from constrained indoor and outdoor scenes, 8 actors performed 8 different activities each filmed with 14 cameras. Each scene lasts about 4 minutes and a total of more than 1.3 million frames were recorded. It contains pose ground truth as well as the intrinsic parameters for the cameras used.

The Microsoft Common Objects in Context dataset (MS COCO) [33] is a general dataset for object detection, segmentation, key-point detection, and captioning. It contains 330 thousand images, of which over 200 thousand are annotated. Despite not being its main focus, it also includes images of 250 thousand people with masks annotating the area of the image that they cover.

Avatars in Geography Optimized for Regression Analysis (AGORA) [34] is a fully synthetic dataset, which means that. while it's not quite as realistic as other datasets, it can provide ground truth that is much more accurate and detailed. Due to being synthetic it can also have annotated data that would be extremely difficult to annotate in the real world, such as scenarios up to 15 people are visible to the camera simultaneously. It uses scanned models of 4240 people, with 257 of them being children. A total of 17 thousand frames were rendered, with each containing at least 5 annotated synthetic humans.

Civilian American and European Surface Anthropometry Resource (CAESAR) [35] is a dataset containing 3D body scans of 5 thousand people between the ages of 18 and 65 in realistic poses.

SizeUSA [36] is a dataset with 3D scans of over 10 thousand subjects. It was recorded to aid in the fashion industry in the United States of America (USA) and subjects were scanned in 12 different cities

3.1.2 SMPL

SMPL (Skinned Multi-Person Linear) is a skinned vertex-based model of the human body [37]. It describes any human body in any position by starting off with a base 3D mesh of the human body and then deforms that mesh linearly with pose and shape parameters. The linearity and independence of pose and shape make SMPL particularly useful for most challenges related to describing the human form in some way.

After fitting the base mesh to scans from the CAESAR dataset, principal component analysis (PCA) was used to reduce the dimensions required to describe the variations in the mesh. Body shape deformations were found to follow a mostly Gaussian distribution within each gender, which means that a very large dimension reduction could be attained with PCA. In fact, despite the base mesh having 21000 vertices, with only 10 components over 90% of the variation in shapes can be covered. The authors of [37] made both the male and female SMPL models available as well as a gender-neutral one, all with 300 principal components.

In order to correct for shape deformations caused by poses, SMPL models shape corrections based on the poses which are then added to the shape parameters of the model.

Because of its robustness and ease of use, SMPL is used by most HPE systems, and SMPL parameters are often provided as ground truth in datasets.

The SMPL model was extended to include hand poses in SMPL+H [38] and then to include facial expressions in SMPL-X [39].

In 2020, a new model called STAR was introduced [40], which improves on SMPL by defining per-joint pose correctives and learning which vertices are affected by the different joint movements. This way more realistic deformations are achieved and the number of model parameters can be reduced without negatively affecting the results. Besides this, STAR was also trained on both CAESAR and SizeUSA.

3.1.3 Image Processing Systems for Human Pose Estimation

While most recent research on the task of HPE with IPS makes use of deep learning, not all of it does, particularly in cases where large datasets are not yet available. For example, the system proposed in [11] uses RGB-D data from a Kinect system to semi-automatically track body joints. Through a software tool called KiMA, ellipsoid masks are manually selected that delimit the parts of the body to be tracked. Optical flow is then used to keep track of the selected ellipsoids, requesting manual corrections when needed.

However, for most applications, the focus of research has been on improving DL HPE systems to make them faster, more accurate, more robust, and to allow them to work off of less information.

The authors of [41] were able to robustly extract the 3D pose of the human body as well as its 3D shape from a single image. This was accomplished by first using the DeepCut method introduced in [42], which uses CNNs to estimate 2D body joints from the image, then optimizing 3D body joints generated with the SMPL model [37], using the 2D distance between the 2D projection of these joints and the DeepCut 2D joints as the error.

HybrIK [14] first finds 3D joint positions with a convolutional network as well as SMPL pose and shape parameters with a fully connected network. It then uses a hybrid analytical-neural approach to calculate inverse kinematics to fuse the 3D joint locations and SMPL parameters, which leads to an improvement on both.

DynaBOA [43] uses online unsupervised learning to adapt to data from new sources as it's being tested; this allows it to adapt to new domains with different attributes such as camera intrinsics and background colors.

Most DL HPE systems calculate a bounding box around the subject as a first step [43, 14, 42, 41, 44], this is known as a top-to-bottom approach. The opposite, first finding the joints and then organizing them to fit the subjects and the image, is called a bottom-to-top approach.

Due to only analysing the contents of the image inside the bounding box, top-to-bottom approaches lose the information as to where in the image the subject is. To address this, CLIFF [45] keeps the bounding box information and uses it in the estimation of the global orientation of the subjects.

The authors of PARE [44] found that most state-of-the-art 3D pose and shape estimation methods rely on global features, which makes them sensitive to occlusions, even small ones. They addressed this issue by creating an architecture that learns to predict attention masks guided by body parts which led to improved performance on visible body parts whenever other body parts are occluded.

Deciwatch [46] makes use of temporal information to improve the speed of other DL HPE systems on videos. It does so by downsampling the videos and feeding only those frames to the DL HPE system used as a backbone, then it denoises these poses with transformer-based encoder modules, and finally, it again makes use of transformer-based encoder modules to fill in the dropped frames. With this method and PARE as the backbone, results were improved (due to the temporal information being used) even when only 10% frames were analyzed, which means a 10x speedup was achieved at no cost.

3.2 Computer Generated Blanket Occlusions

The idea of synthetically generating blanket occlusions to augment datasets is not novel. A couple of papers exploring it have been published.

In [47] synthetic blanket occlusions were used to augment depth video recorded of patients performing movement sequences in a bed. This dataset (with and without synthetic occlusions) was used to train and evaluate systems based on random forests, convolutional neural networks, and recurrent neural networks, which make use of temporal information. Training on the augmented dataset improved the accuracy of all 3 systems, with recurrent neural networks in particular improving the most.

In [48] the authors made use of synthetic blanket occlusions to augment a dataset consisting of depth images of subjects lying down in beds. They then used this augmented dataset to train a CNN that generates SMPL parameters; from these SMPL parameters, pressure maps were estimated.

Chapter 4

BlanketGen

In order to explore the potential of synthetic blanket occlusion augmentations in improving the performance of DL HPE systems, a pipeline was implemented that augments the 3DPW dataset with synthetic blanket occlusions, this pipeline was then used to create a dataset with synthetic blanket occlusions. This pipeline was named BlanketGen and the dataset created with it BlanketGen-3DPW.

The pipeline was implemented in Blender because it is free, has robust cloth simulation, and proves a python API which supports most of the capabilities of the program.

3DPW was chosen as the dataset to augment as it has a reasonable size, provides ground truth body models, and because there are state-of-the-art DL HPE systems with open source code that used it for training and testing.

4.1 Pipeline Structure

BlanketGen takes the 3DPW dataset and generates videos with CG blankets simulated on top of the subjects of the videos.

In order to do this, first it loads a video from 3DPW and fetches the ground truth data associated with it. Then it loads the body model onto the scene, fixes its position to the center of the axes, and adjusts the position of the camera accordingly; a rectangle cuboid is also placed behind the body model from the camera's perspective as a makeshift bed. After the camera, body model, and bed are positioned properly a blanket is simulated on top of the body model for a short duration in order for the video to start with the blanket already resting on the person.

After this setting time, the model is animated using shape keys with the blanket continuously being simulated on top of it. If the blanket falls off the body model then the simulation is stopped and the pipeline continues without simulating the rest of the video. When this happens the next video is started where the previous left off with the blanket back in a resting position.

After the cloth simulation is done the scene is rendered with the original frames composited as the background and both the body model and bed as holdouts, which are transparent masks. The video is then saved in its own folder as a sequence of frames.

After the video is saved the process repeats until there are no more videos to augment.

The pipeline is illustrated in figure 4.1

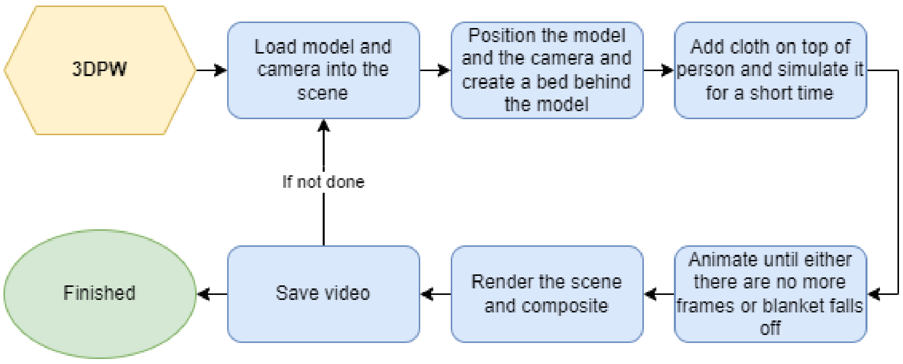

3DPW
Load model and camera into the scene
Position the model and the camera and create a bed behind the model
Add cloth on top of person and simulate it for a short time
If not done
Finished
Save video
Render the scene and composite
Animate until either there are no more frames or blanket falls off

Figure 4.1: Diagram of the BlanketGen pipeline

4.2 Data Loading

In 3DPW the videos are saved as frames with each video in its own folder, ground truth for each video is provided separately in a sequence file.

This sequence file contains the global location of the subject and the camera both at 30Hz and 60Hz, the SMPL pose and shape parameters (with 300 principal components), with the poses being provided at 30Hz and 60Hz, the camera intrinsics and extrinsics, with the extrinsics being provided only at 30Hz, the genders of the subjects, the 2d joint locations both at 30Hz and 60Hz, a detailed 3D mesh of the subjects with clothes (only at 30Hz), and texture maps for the detailed 3D mesh.

The detailed 3D mesh isn't provided for every subject, and the texture maps aren't provided for all of the detailed 3D meshes. Because of this, and because of the increased simplicity, BlanketGen uses the SMPL model instead.

The camera position isn't accurate for every frame, so a list of all the frames where it is accurate is also provided.

In order to load the SMPL models onto Blender, the SMPL-X [39] python package is used to calculate the vertices and faces of the model.

To avoid collision errors caused by the SMPL models intersecting with each other, which negatively affect the cloth simulation, only one model was loaded for each augmented video, with the blanket being created on top of the model. This leads to the blanket only interacting with one of the subjects, which isn't ideal as some videos contain two subjects, but it's a practical way to avoid cloth simulation errors that ruin the entire video.

A camera is created and added to the scene. The created camera's attributes are then altered to match the intrinsic matrix provided in the sequence file.

A sun light, which is a light source that shines parallel rays of light is also added.

In figure 4.2 an example of a frame at this stage of the pipeline is shown.

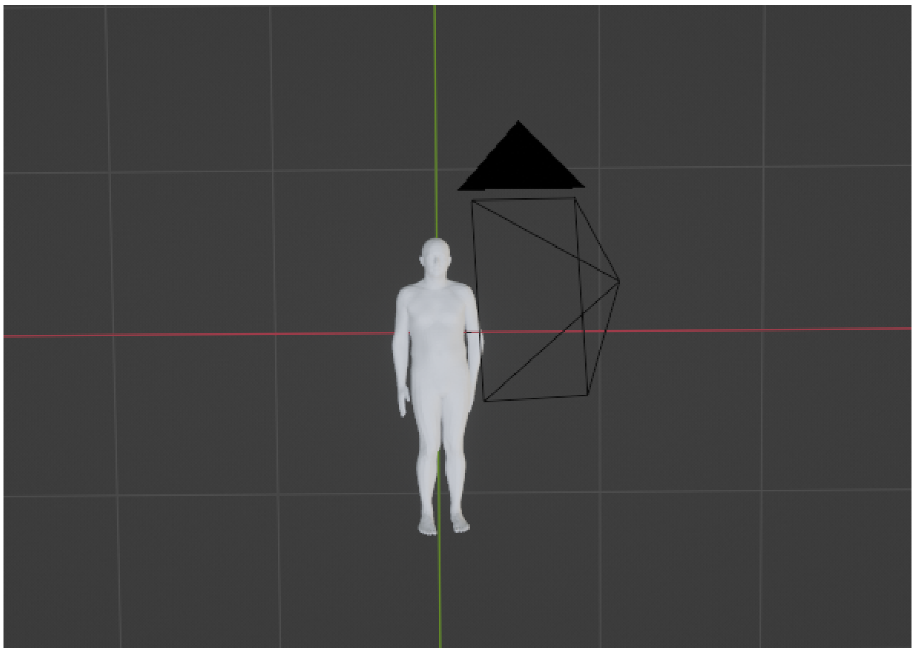


Figure 4.2: Example of a frame after the data is loaded.

4.3 Positioning

While the positions provided as ground truth are accurate relative to each other, the global positions have low-frequency errors and so the models drift over time. To address this, the model of the subject is fixed at position (0,0,0) and the camera is re-positioned so that their relative positions remain correct.

Besides this, some problems arose from specific Blender interactions and format differences.

The format used for the *world matrix* in Blender doesn't match the format of the extrinsic matrix provided as ground truth for the camera. To address this the *world matrix* **W** of the camera has to be calculated from the provided extrinsic matrix **E** as shown:

$$\mathbf{E} = \begin{bmatrix} R_{11} & R_{12} & R_{13} & T_x \\ R_{21} & R_{22} & R_{23} & T_y \\ R_{31} & R_{32} & R_{33} & T_z \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

$$\mathbf{W} = \begin{bmatrix} R_{11} & R_{12} & R_{13} & -T_x \\ -R_{21} & -R_{22} & -R_{23} & -T_y \\ -R_{31} & -R_{32} & -R_{33} & -T_z \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

The rotation of the SMPL model is provided in the axis-angle format, which describes the rotation of the object with a vector perpendicular to the plane of the rotation that has magnitude equal to the angle of the rotation. Unfortunately, axis-angle rotations in Blender also require use of an operator that's not available in headless mode. Instead, rotation quaternions had to be used. Given the axis-angle rotation vector V the following calculations were performed to obtain the rotation quaternion Q:

$$Q = \begin{bmatrix} \cos\frac{\|V\|}{2} & \hat{V}_x \sin\frac{\|V\|}{2} & \hat{V}_y \sin\frac{\|V\|}{2} & \hat{V}_z \sin\frac{\|V\|}{2} \end{bmatrix}$$

The center of the body model in Blender doesn't match the one in the ground truth, so it has to be adjusted. Fortunately, Blender allows the center of an object to be relocated, unfortunately, it relies on operators that can't be used while running headless. Instead this problem was fixed by iterating through all the vertices of the model and displacing them by the offset necessary for their location relative to the center of the model to be correct.

While iterating through all the vertices, the vertex most distant from the camera is found. A new set of axes is created with this vertex at the center and with one of the axis pointing directly away from the camera. The rectangle cuboid bed is then placed in the scene using these axes, this way the distance between the bed and the body model can be a defined constant; this is done to avoid the subject's model clipping into the bed and causing the cloth simulation to fail.

The position of the simulated blanket is initially set a short distance away from the body model in the direction of the camera, and its orientation is set to match the orientation of the bed.

In figure 4.3 an example of a frame at this stage of the pipeline is shown.

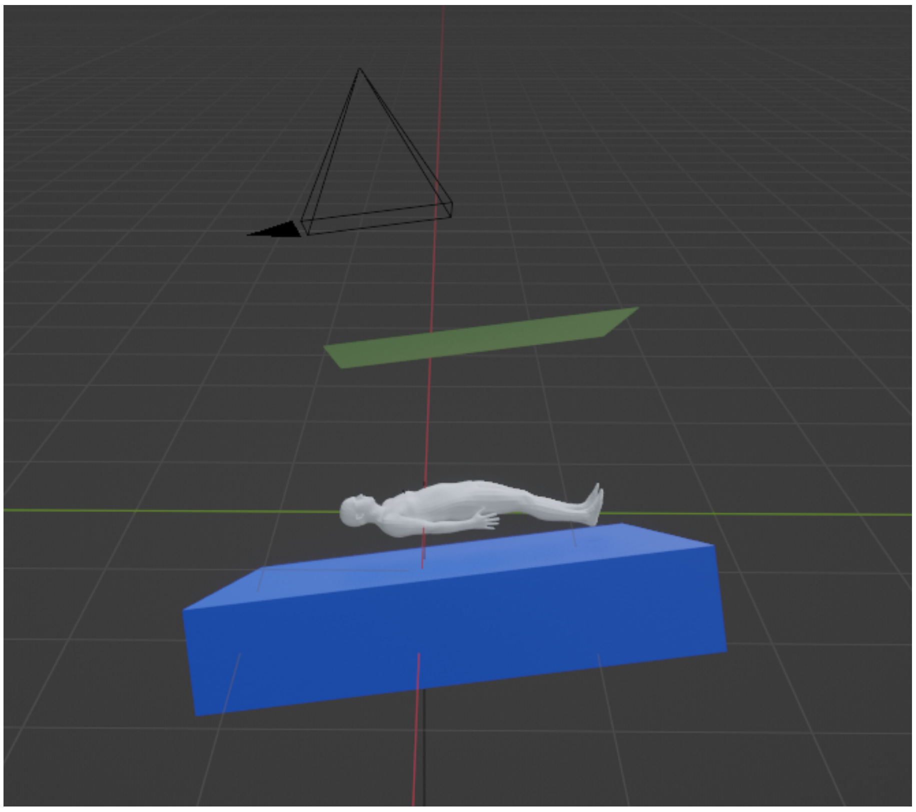


Figure 4.3: Example of a frame after all the objects are placed into position.

4.4 Cloth Simulation

In order to create a realistic blanket, first a rectangular plane is added, then this plane is cut into 5625 (75^2) smaller rectangles, then the *subdivision* modifier is used, which further subdivides the plane and smooths the subdivisions, but which doesn't affect the cloth simulation. With 2 levels of subdivision this further increases the number of squares in the blanket to 90000, but with only the original 5625 being used in the cloth simulation.

The blanket is then set as a cloth with the collision calculations being performed in 15 increments per frame, and the spring calculations being performed in 10 increments per frame. The collision distance is set at half a millimeter and the gravity is set perpendicular from the bed.

The cloth simulation is run for 24 frames before the video starts in order to let the blanket come to a rest on top of the body model.

The simulation of the blanket is updated frame by frame and at every frame the vertex closest to the body model is found, if the distance between this vertex and the body model is too large the simulation is stopped and the video goes on in the pipeline. The start of the next video is set to be the maximum of the frame where the current video ended and the first frame of the video plus 48, this is done to avoid the pipeline getting stuck in cases where the blanket falls off either during the time the blanket is allowed to come to a rest or in cases where it glitches through the body model. In the generation of BlanketGen-3DPW there was a bug in this which made it so that the next video was always started 48 frames after the start of the current video, this lead to significant overlap in the final dataset, as many frames were part of multiple generated videos.

In figure 4.4 an example of a frame with a simulated blanket is shown.

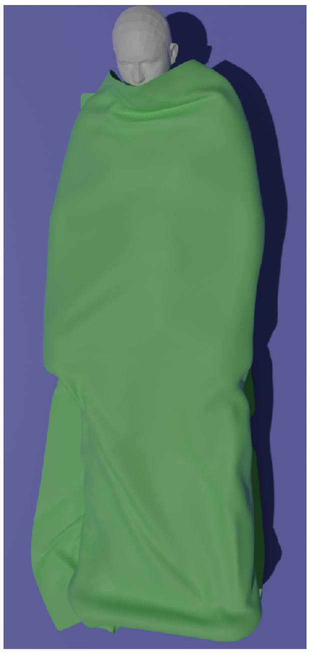


Figure 4.4: Example of a cloth simulated with the parameters specified in this section.

4.5 Rendering and Compositing

Rendering is done with the *Cycles* rendering engine, which uses raytracing to generate physics-based renders. Due to the ease of parallelization of Monte Carlo estimations the render engine can use GPUs to speed up considerably. A total of 1000 samples are calculated per pixel, with a time limit of 10 seconds per frame.

The material of the blanket uses the Principled BSDF model, which is the default option in Blender and is generally fairly robust. The color is randomized uniformly in the RGB space for each video generated, the sheen is set to 1, and the specularity is set to 0.16.

The bed and the body model are set as holdouts, which means they function as transparent masks. This means that when compositing the rendered scene with the original frames, only the blanket is visible, and when the blanket goes behind the body model from the camera's perspective the portion that's behind the body model becomes invisible. With this the blanket realistically disappears when the subject goes in front of it.

The compositing is done with the node tree system that blender provides, an alpha-over node is used to layer the rendered scene on top of the original video.

In figure 4.5 a frame from a generated video is shown.

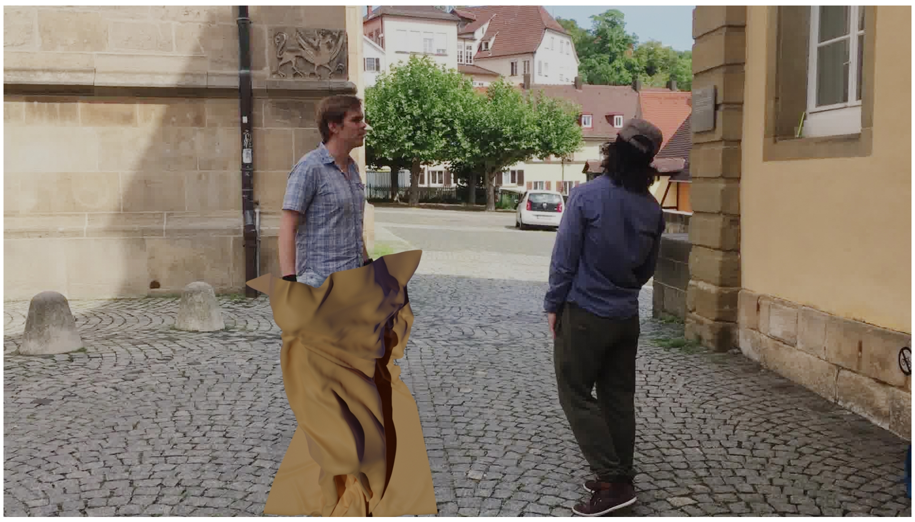


Figure 4.5: A frame from BlanketGen-3DPW.

4.6 Format

The output of the compositor is saved as a sequence of JPG images in a folder for every video. Each video folder is contained within a folder for the original video from which it was generated. Furthermore, these folders are separated into three folders for the three dataset partitions (train, test, and validation).

4.7 BlanketGen-3DPW

BlanketGen-3DPW is an augmented dataset generated with the BlanketGen pipeline. It took 3 weeks to generate running on 2 desktop PCs using 3 Nvidia GPUs. In total it contains 1037 generated videos with synthetic blanket occlusions, a total of 96399 frames.

4.7.1 Format

BlanketGen-3DPW was formatted to be used by HybrIK as a drop-in replacement for 3DPW. Each video is a sequence of frames in its own folder, and the ground truth is contained in one large file using the COCO format [49].

4.7.2 Excluded Videos

Despite the precautions taken, in some videos the cloth simulation failed, usually due to the body model clipping into itself or moving too fast for the simulation to keep up. Because of this, 66 generated videos had to be excluded totaling 14332 frames.

The ground truth for the camera location on 3DPW isn't accurate in every frame, to deal with this BlanketGen just ignores frames with invalid camera positions and reuses the last valid frame. In some videos there are many frames in a row without accurate ground truth camera position, which leads to very inaccurate generated videos. This happened in 172 generated videos, with 23691 frames being excluded for this reason.

This means that 38023 of the original 134422 frames had to be excluded, a waste of 28% of the generated frames.

Chapter 5

BlanketSet

5.1 Objective

The goal of this acquisition is to create a dataset (BlanketSet) that can be utilized with deep learning computer RGB HPE systems to evaluate the benefit of augmenting datasets with blanket occlusions. Beyond this initial use, it could also be used in further research in the future.

5.2 Design and Procedure

5.2.1 Design

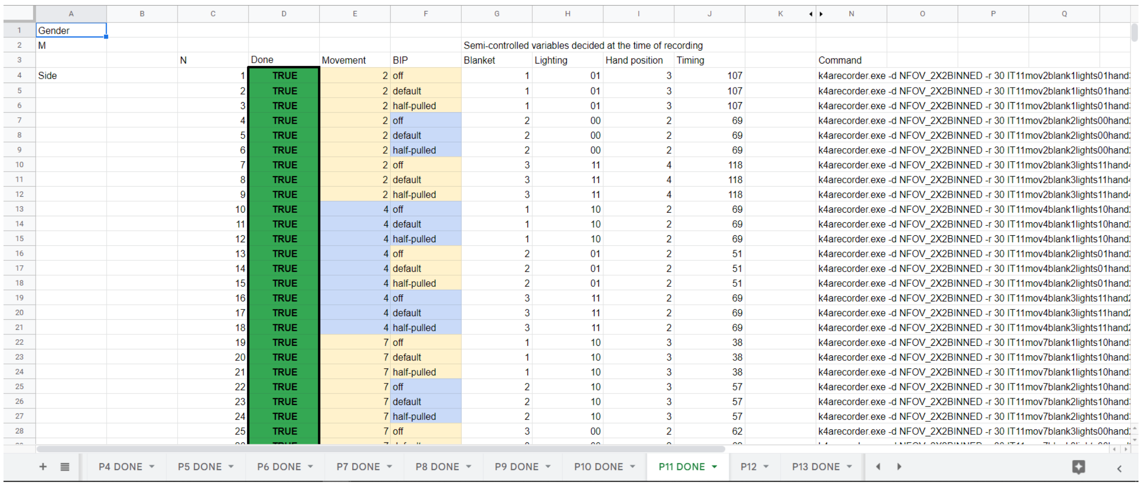
Sequences of movements were recorded with an Azure Kinect which records RGB and Infrared video, which it uses to create depth estimations of the area it's recording. Every sequence recorded was recorded both with and without real blanket occlusions.

While BlanketSet was originally planned to be used for quantitative analysis of the improvement of HPE systems caused by BlanketGen, annotation proved to be more of a challenge than what was feasible, so it was instead used only for qualitative analysis. Annotation approaches are further discussed in the future work chapter.

5.2.2 Procedure

Each participant had the dataset acquisition protocol explained to them, as well as the information related to their privacy rights, they were then asked to sign the informed consent form (appendix A.1). Then for each movement sequence recorded they were instructed on how to do them, afterwards they executed the movement sequence in all the planned variations. Movements were repeated whenever they couldn't be executed properly; whether the movements were executed correctly or not was decided subjectively at the time of recording.

A spreadsheet was used to keep track of the movements each participant had to perform and to streamline the recording process.



	A	B	C	D	E	F	G	H	I	J	K	N
1	Gender											
2	M						Semi-controlled variables decided at the time of recording					
3			N	Done	Movement	BIP	Blanket	Lighting	Hand position	Timing		Command
4	Side		1	TRUE	2	off	1	01	3	107		k4arecorder.exe -d NFOV_2X2BINNED -r 30 IT11mov2blank1lights01hand
5			2	TRUE	2	default	1	01	3	107		k4arecorder.exe -d NFOV_2X2BINNED -r 30 IT11mov2blank1lights01hand
6			3	TRUE	2	half-pulled	1	01	3	107		k4arecorder.exe -d NFOV_2X2BINNED -r 30 IT11mov2blank1lights01hand
7			4	TRUE	2	off	2	00	2	69		k4arecorder.exe -d NFOV_2X2BINNED -r 30 IT11mov2blank2lights00hand
8			5	TRUE	2	default	2	00	2	69		k4arecorder.exe -d NFOV_2X2BINNED -r 30 IT11mov2blank2lights00hand
9			6	TRUE	2	half-pulled	2	00	2	69		k4arecorder.exe -d NFOV_2X2BINNED -r 30 IT11mov2blank2lights00hand
10			7	TRUE	2	off	3	11	4	118		k4arecorder.exe -d NFOV_2X2BINNED -r 30 IT11mov2blank3lights11hand
11			8	TRUE	2	default	3	11	4	118		k4arecorder.exe -d NFOV_2X2BINNED -r 30 IT11mov2blank3lights11hand
12			9	TRUE	2	half-pulled	3	11	4	118		k4arecorder.exe -d NFOV_2X2BINNED -r 30 IT11mov2blank3lights11hand
13			10	TRUE	4	off	1	10	2	69		k4arecorder.exe -d NFOV_2X2BINNED -r 30 IT11mov4blank1lights10hand
14			11	TRUE	4	default	1	10	2	69		k4arecorder.exe -d NFOV_2X2BINNED -r 30 IT11mov4blank1lights10hand
15			12	TRUE	4	half-pulled	1	10	2	69		k4arecorder.exe -d NFOV_2X2BINNED -r 30 IT11mov4blank1lights10hand
16			13	TRUE	4	off	2	01	2	51		k4arecorder.exe -d NFOV_2X2BINNED -r 30 IT11mov4blank2lights01hand
17			14	TRUE	4	default	2	01	2	51		k4arecorder.exe -d NFOV_2X2BINNED -r 30 IT11mov4blank2lights01hand
18			15	TRUE	4	half-pulled	2	01	2	51		k4arecorder.exe -d NFOV_2X2BINNED -r 30 IT11mov4blank2lights01hand
19			16	TRUE	4	off	3	11	2	69		k4arecorder.exe -d NFOV_2X2BINNED -r 30 IT11mov4blank3lights11hand
20			17	TRUE	4	default	3	11	2	69		k4arecorder.exe -d NFOV_2X2BINNED -r 30 IT11mov4blank3lights11hand
21			18	TRUE	4	half-pulled	3	11	2	69		k4arecorder.exe -d NFOV_2X2BINNED -r 30 IT11mov4blank3lights11hand
22			19	TRUE	7	off	1	10	3	38		k4arecorder.exe -d NFOV_2X2BINNED -r 30 IT11mov7blank1lights10hand
23			20	TRUE	7	default	1	10	3	38		k4arecorder.exe -d NFOV_2X2BINNED -r 30 IT11mov7blank1lights10hand
24			21	TRUE	7	half-pulled	1	10	3	38		k4arecorder.exe -d NFOV_2X2BINNED -r 30 IT11mov7blank1lights10hand
25			22	TRUE	7	off	2	10	3	57		k4arecorder.exe -d NFOV_2X2BINNED -r 30 IT11mov7blank2lights10hand
26			23	TRUE	7	default	2	10	3	57		k4arecorder.exe -d NFOV_2X2BINNED -r 30 IT11mov7blank2lights10hand
27			24	TRUE	7	half-pulled	2	10	3	57		k4arecorder.exe -d NFOV_2X2BINNED -r 30 IT11mov7blank2lights10hand
28			25	TRUE	7	off	3	00	2	62		k4arecorder.exe -d NFOV_2X2BINNED -r 30 IT11mov7blank3lights00hand

Figure 5.1: A screenshot of the spreadsheet used to aid with the acquisition of BlanketSet

BlanketSet was recorded in the Epilepsy Monitoring Unit of the University Hospital Center of São João.

5.2.3 Dimensions

There were too many variables to evenly cover all the combinations. So the variables were split between controlled variables, semi-controlled variables, and uncontrolled variables.

Controlled variables were decided manually to cover their variations as evenly as possible.

Semi-controlled variables were decided randomly in an attempt to avoid correlations between them.

Uncontrolled variables were ignored as they were too difficult to control, but some that could affect the quality of the dataset are still listed.

The gender distribution of the subjects was attempted to be controlled, but it ended up being unbalanced.

Due to the time between recordings and Covid restrictions lifting between them, in some recordings the subjects had masks and in others they didn't. Besides this, in the videos without masks, different bedding (including blankets) was used.

One person participated both before and after the restrictions were lifted, and therefore both with and without a mask, that person was both IT03 and IT17.

5.2.3.1 Controlled variables

- Movement sequences: 8 movement sequences were performed, they are explained section 5.3.

- Blanket Initial Position (BIP): There were 3 different initial positions for the blanket: no blanket, fully covering the subject (referred to as Full in chapter 6), and pulled down so that only about the feet and lower legs were covered (referred to as Half in chapter 6). This last

BIP was decided to simulate a failed attempt to remove the blanket, as sometimes happens in real situations in the EMU.

- Blanket: Originally 3 different blankets were alternated between, of different thicknesses colors, and weights (black heavy, white light, and green heavy). In the recordings after the restrictions were lifted, different blankets were used (grey heavy, white light, orange heavy).

5.2.3.2 Semi-controlled variables

- Lighting: 4 different lighting setups were used: having the blinds up or down, and having the lights on or off. However, the blinds in the window next to the bed were broken so they were always up; after the restrictions lifted a blanket was used to cover the window, so some control was gained (blanket tied up or not). The light above the bed was also not able to fully turn on, which reduced control over the lighting.
- Hand position: 4 hand positions were covered, they are covered in subsection 5.2.3.4.
- Time between position changes (referred to as timings from here on out): 0.5 to 2 seconds (timings were selected in beats per minute with integer precision).

5.2.3.3 Uncontrolled variables

- Body shape
- Clothes
- Hair color/length
- Time of day

5.2.3.4 Hand positions

Hand positions could affect HPE systems and as such 4 variations were included: relaxed, fingers spread, fingers stretched touching, and closed fist.

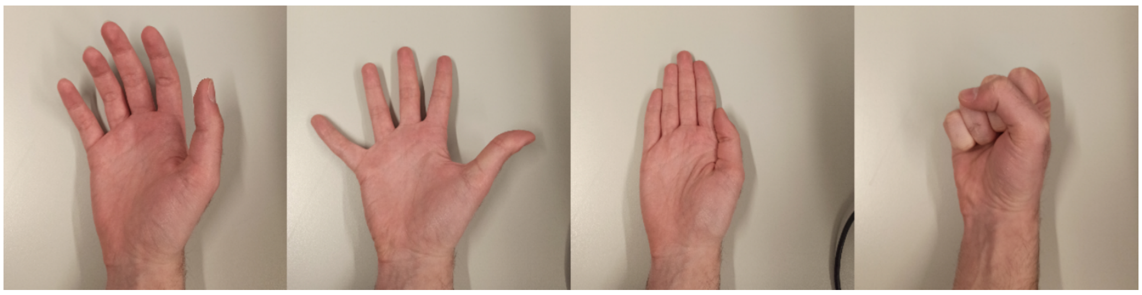


Figure 5.2: The 4 hand positions from left to right: relaxed, fingers spread, fingers stretched touching, and closed fist

5.3 Movement Sequences

Each motion sequence consists of switching from position to position once per timing for 20 timings (as defined previously). The emphasis was on ensuring that the motions were repeated properly over ensuring they were done perfectly as described: movements that were slightly different from the description but which were consistent with the previous performances of the participant were accepted over ones which better matched the description but not the previous performances. This was because there were originally plans to annotate BlanketSet by making use of a state-of-the-art HPE system on the sequences with no blankets and then use the same annotations with some manual adjustments to annotate the sequences with blankets. However, this wasn't feasible as the repetitions weren't similar enough.

With each movement taking 20 timings, and each timing taking an average of 1 second, each sequence takes an average of 20 seconds. In total, 27 sequences were recorded per subject; initially, 10 subjects were expected to be recorded (5 men, 5 women), but in the final dataset, 14 subjects total were recorded (9 men, 5 women). Therefore, 270 sequences were expected to be recorded; in total 405 sequences were recorded, since one person recorded 27 sequences before the restrictions were lifted and 27 after.

In order to cover all the variations of the controlled variables, each sequence was repeated at least 9 times.

With the original 270 planned recordings, 30 different movement sequences would allow coverage of every combination of the controlled variables, 15 to also evenly cover both genders. However, to better cover the semi-controlled variables, only 8 sequences were covered.

Considering the context of the thesis, recording movements of the lower half of the body was more important than the upper half, since it's more likely to be covered by a blanket in real-world scenarios, therefore the majority of the motion sequences included the legs. It was also important to cover as much of the range of motion of each joint as possible.

The sequences were planned to be easily repeated to facilitate annotation, so each motion generally only moved either the legs or the arms since coordinating arm and leg movements with the timings and with good precision can be very challenging. Each motion sequence included a simple motion being repeated several times in a row.

Repeated motions with the weight of a blanket on top can get tiring, so movement sequences were chosen to avoid tiresome positions, this helped with their repeatability as well.

For the sake of making BlanketSet more challenging, self-occlusions were also included in the sequences.

After careful consideration, 8 motion sequences were selected, 5 involving the legs and 4 involving the arms (1 involves both the arms and the legs). Each of them was given a simple name. They are as follows:

5.3.1 Foot to knee

The participant starts in a default resting position facing upwards, after 1 timing they lift their left foot and place it on top of the right knee, after 1 more timing they reset back to the default position. They stay in the resting position for 1 timing, then repeat the same movement with the opposite foot and knee. Each repetition of both legs takes 4 timings, so there are 5 repetitions per sequence.

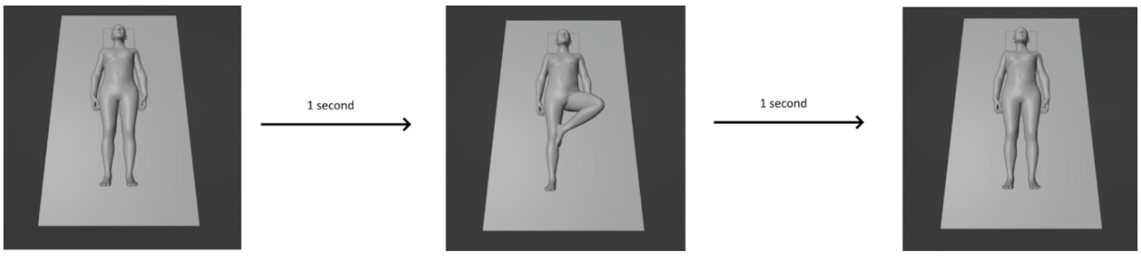


Figure 5.3: A 3d model performing sequence 1

This sequence was chosen because it is challenging as the foot occludes the knee. Besides that, when it is done under a blanket it very significantly displaces the blanket with every repetition, so if a system is able to extract information from the displacement of the blanket, it will have a lot of information to take from it.

5.3.2 Knee bend

The participant starts in a default resting position facing upwards with their right knee bent and their right foot on the bed next to the left knee, after 1 timing they bend their left knee to the same position, lifting it up while keeping the foot on the bed and simultaneously they stretch the right leg back down. After 1 timing they repeat the same movement but with the other knee. Each repetition of both legs takes 2 timings, so there are 10 repetitions per sequence.

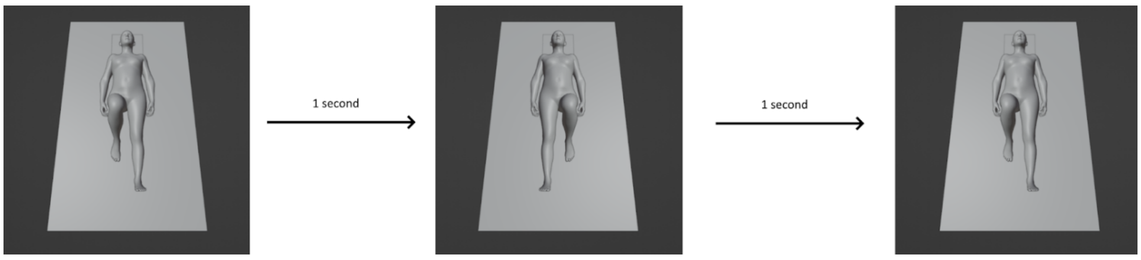


Figure 5.4: A 3d model performing sequence 2

This sequence was chosen because it is somewhat challenging, as the knee is moving almost in the direction of the camera, but also the deformation of the blanket can partially indicate how high the knee was bent, which should be useful for a system that has learned to extract information from blanket displacement.

5.3.3 Swinging legs

The participant starts in a default resting position on the side, with one knee stretched and the other bent at a 90º angle. After 1 timing they switch the legs, bending the stretched knee to 90º and stretching the bent one. The side on which the participants lie down is randomly decided at the time of recording. Each repetition takes only 1 timing, so there are a total of 20 repetitions.

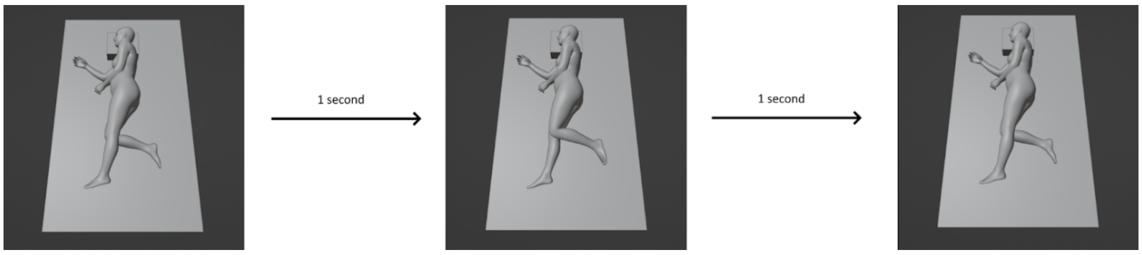


Figure 5.5: A 3d model performing sequence 3

This sequence was chosen because it has a lot of self-occlusions, half of the body is occluded by the other half, and the lower half of the bottom leg passes under the top leg with every repetition. Beyond that, the location of the bottom leg is very occluded during the swinging motion, so being able to make use of both temporal information and information from the blanket displacement will probably be required for good results.

5.3.4 Hands to shoulders

The participant starts in a default resting position facing upwards, after 1 timing they bring both hands to the opposite shoulders simultaneously with the right arm in front of the left, then after 1 timing they reset back to the default position. They stay in this position for 1 timing, then repeat the same movement with the opposite arm in front. Each repetition with both arms in front takes 4 timings, so there are 5 repetitions per sequence.

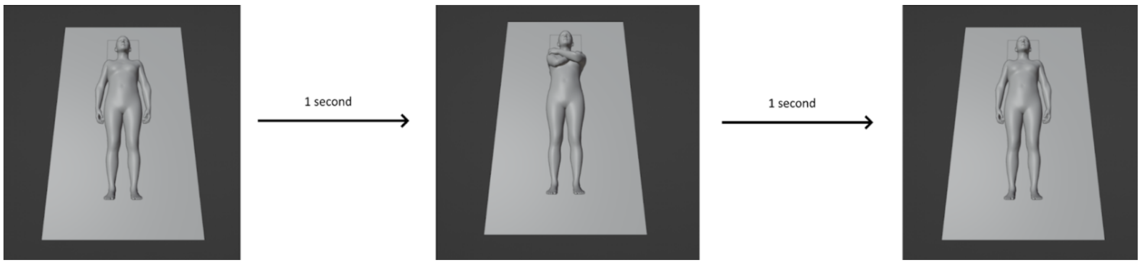


Figure 5.6: A 3d model performing sequence 4

This sequence was chosen because it is very challenging since the arms cross over each other and the hands occlude the shoulders. Besides that, the hands can interact with the blanket, so there will be samples where the blanket is moving but the legs aren't, which increases the variety of the data in the dataset.

5.3.5 Belly-down spread

The participant starts in a default resting position facing down, after 1 timing they spread their arms and legs to the edges of the bed, then after 1 timing they reset to the default position. Each repetition takes 2 timings, so there are 10 repetitions per sequence.

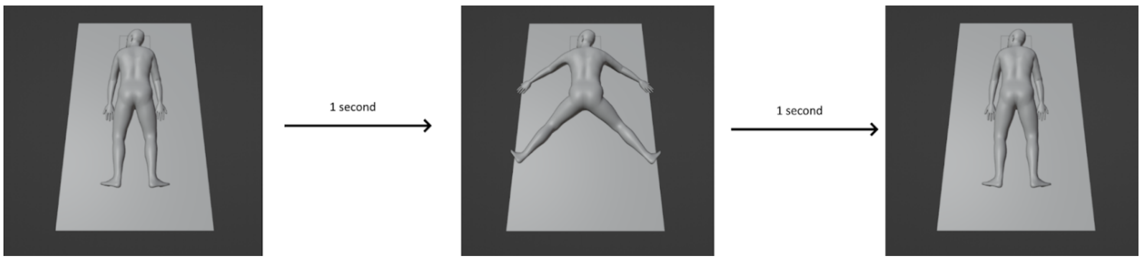


Figure 5.7: A 3d model performing sequence 5

This sequence was chosen because it significantly displaces the blanket with each repetition, and because it involves moving both the arms and legs simultaneously without being difficult to coordinate.

5.3.6 Torso lean

The participant starts in a leaning position with the right forearm lying down on the bed and the left arm stretched. Both hands are placed symmetrically on the sides and do not move throughout the whole sequence. After each timing the participant switches which side they're leaning on by stretching one arm and bending the other until the elbow reaches the bed. Each repetition takes 1 timing, so there are 20 repetitions.

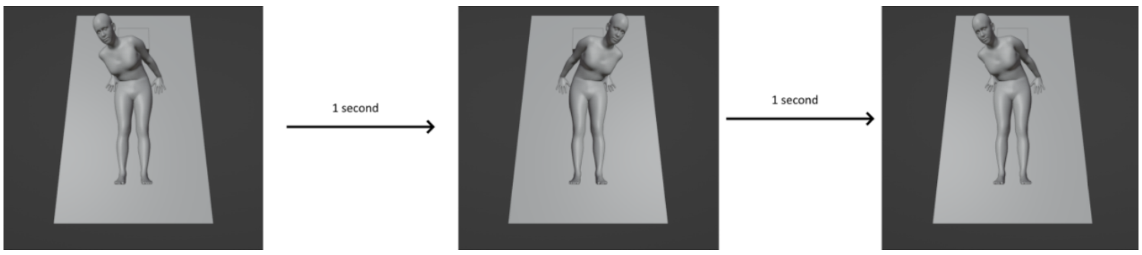


Figure 5.8: A 3d model performing sequence 6

This sequence was chosen because it includes movement of the torso, which is generally static in the other sequences, also one elbow is always occluded, which increases the difficulty of the dataset.

5.3.7 Stretched arms

The participant starts in a default resting position facing up with their arms stretched next to them. After 1 timing they raise their arms upwards until they're perpendicular to the bed, then after 1 timing they bring their arms down, still stretched, to the side to a T-pose. 1 timing later they raise their arms back up to the previous position then a timing later they lower their arms back down to the default resting position. Each repetition takes 4 timings, so there are 5 repetitions

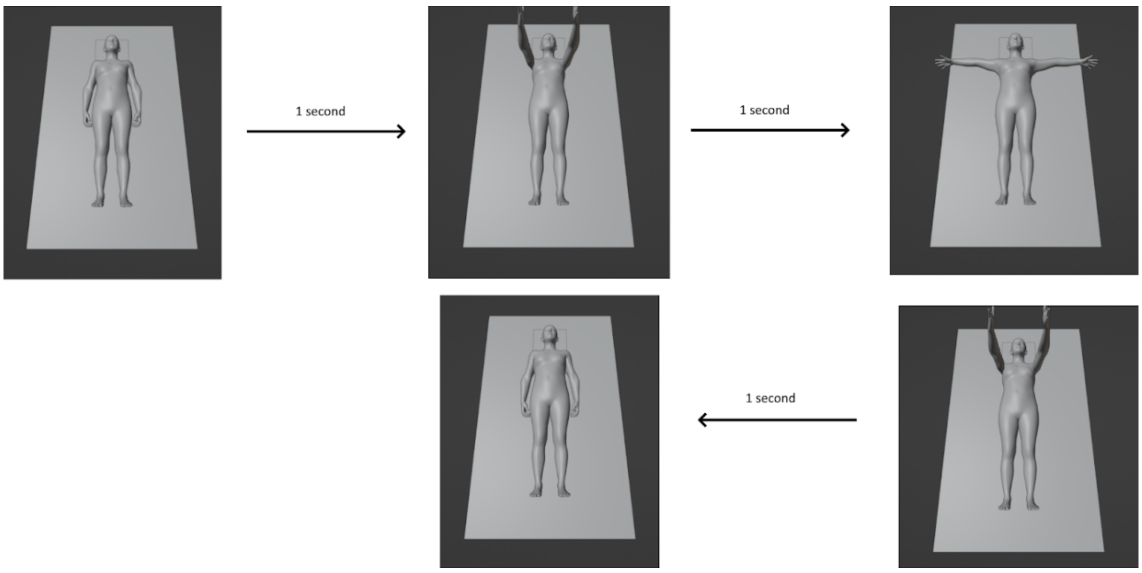


Figure 5.9: A 3d model performing sequence 7

This sequence was chosen because it includes a large portion of the range of motion of the shoulders plus it is partially vertical, which is a dimension the other sequences don't prominently explore.

5.3.8 Feet stretched

The participant starts in a default resting position facing up with their feet at 90º to the rest of the leg. After 1 timing they stretch their feet to be as in line with the legs as they can. They hold this position for 1 timing then reset back to the default position. Each repetition takes 2 timings, so there are 10 repetitions total.

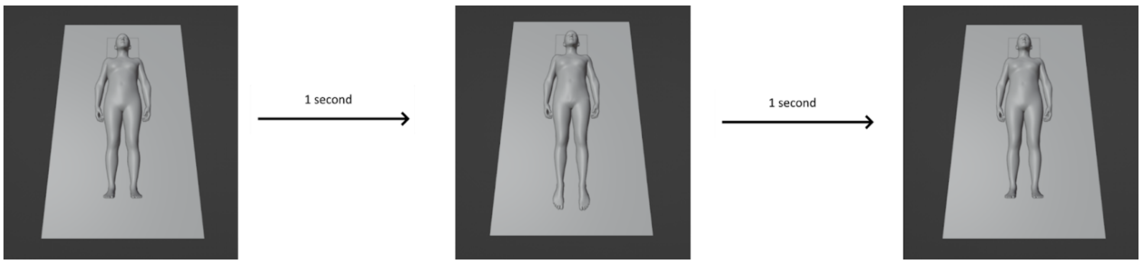


Figure 5.10: A 3d model performing sequence 8

This sequence was chosen because it consists only of movements of the feet, which are notoriously difficult to annotate under blankets.

Chapter 6

Evaluation Methods and Results

6.1 HybrIK

In order to evaluate whether CG blanket occlusion augmentations could improve existing DL HPE systems, one such system was chosen.

HybrIK [14] is a state-of-the-art DL HPE system and has been explained in chapter 3.

It was chosen for the evaluation of BlanketGen over other state-of-the-art alternatives as it has an open source implementation with training code as well as a pre-trained model. Time was a scarce resource in the writing of this dissertation so having a pre-trained model was necessary, and implementing a DL HPE system from scratch was beyond the scope of the dissertation, so having open source code was also a necessity.

Unfortunately, due to time constraints, it wasn't feasible to evaluate BlanketGen on any of the DL HPE systems that make use of the temporal information contained in videos.

The pre-trained model provided was originally trained on a joint dataset consisting of training data from Human3.6M, MPI-INF-3DHP, 3DPW, and MSCOCO. It was trained on this dataset for 90 epochs with a learning rate of $1e{-}3$, then for 30 epochs with a learning rate of $1e{-}4$, and finally for 80 epochs with a learning rate of $1e{-}5$.

For the evaluation of BlanketGen it was then trained only on training data from BlanketGen-3DPW for 10 epochs with a learning rate of $1e{-}3$, then 10 epochs with a learning rate of $1e{-}4$, and finally 10 epochs with a learning rate of $1e{-}5$.

The architecture doesn't just predict human pose in 3 dimensions, so the loss function is a weighted average of different cost functions pertaining to the body model, the bounding box, and the 2D and 3D joint positions. The loss over the course of fine-tuning is plotted in figure 6.1.

This fine-tuned model was compared with the original pre-trained model.

6.2 Quantitative Evaluation

Quantitative evaluation was done by testing both models on the testing sets of both the original 3DPW as well as BlanketGen-3DPW.

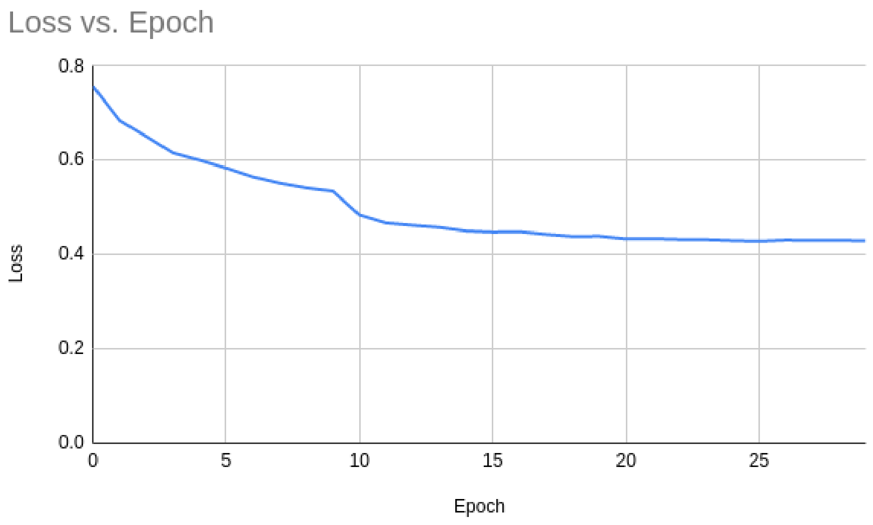


Figure 6.1: A plot of the loss over the epoch number during the fine-tuning.

The average PA-MPJPE was recorded for all test cases and is reported in table 6.1.

Model	PA-MPJPE on 3DPW↓	PA-MPJPE on BlanketGen-3DPW↓
Pre-trained	**44.97**	52.70
Fine-tuned	50.95	**52.46**

Table 6.1: Quantitative results

6.3 Qualitative Evaluation

For qualitative evaluation, videos from BlanketSet were annotated using both models. For this, one sequence from each participant was selected, with each movement sequence, as well as each blanket, being present at least once in this selection.

All faces present in the annotated videos were then blurred to protect the privacy of the participants. Figure 6.2 is a frame from one of these videos.

A google forms survey was then created and made public where the annotated videos could be rated on a scale from 0 to 10, with 0 being a failure to even locate the subject and 10 being perfect annotations.

This survey was advertised to all FEUP students with the dynamic email service provided by the university and gathered a total of 29 responses.

Table 6.2 describes which recordings were used for the survey as well as the average rating each recording got when annotated with either model, rounded to two decimal cases.

As expected, the determining factor for the quality of the annotations was the BIP; table 6.3 summarizes these results as well as the rating difference between both models and a symmetric 95% confidence interval for these differences calculated with a two-tailed paired t-test.

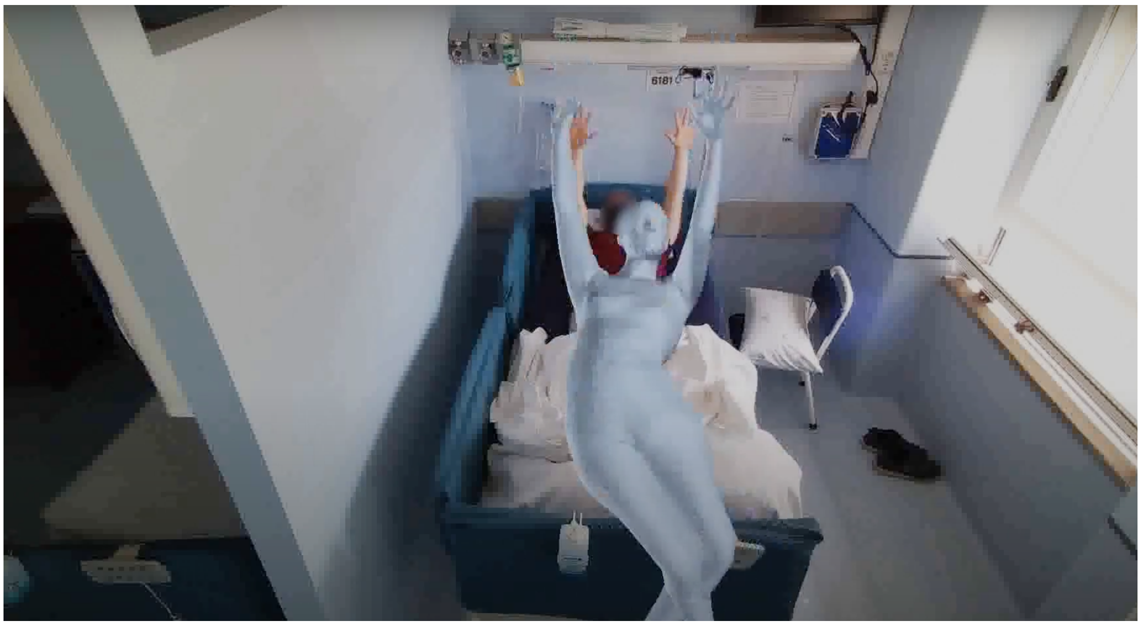


Figure 6.2: A frame from one of the annotated videos with the face of the participant blurred.

Participant	BIP	Sequence	Blanket	Pre-trained↑	Fine-tuned↑
IT01	Half	2	1	**7.28**	6.79
IT02	Full	6	2	3.55	**4.34**
IT03	Half	1	3	5.52	**6.07**
IT04	Full	4	1	0.28	**0.38**
IT05	Half	5	2	**5.10**	4.86
IT06	Full	8	3	**0.69**	0.45
IT07	Half	7	1	5.62	**6.03**
IT08	Full	3	2	1.00	**1.10**
IT09	Half	1	6	8.38	**8.52**
IT11	Full	2	1	0.55	**0.66**
IT13	Half	4	5	8.34	**8.48**
IT15	Full	6	6	**5.62**	5.34
IT16	Half	3	4	**6.07**	5.59
IT17	Full	8	5	0.72	**1.52**
IT20	Half	5	6	8.14	**8.34**

Table 6.2: Recordings selected for evaluation and qualitative results of their annotations

BIP	Pre-trained↑	Fine-tuned↑	Difference
Half	6.81	**6.84**	0.03±0.14
Full	1.77	**1.97**	0.20±0.15
Aggregate	4.46	**4.57**	0.11±0.09

Table 6.3: Summary of the qualitative results, separated by BIP

Chapter 7

Discussion

From the quantitative results, there was a slight improvement in accuracy on the augmented dataset, but it came at a much bigger decrease in accuracy on the original dataset.

It's important to note that BlanketGen-3DPW only has synthetic occlusions on one subject per video, which means that on all of the videos which included two subjects there was at least one person who was at least mostly unoccluded, so the accuracy measured on it is a mix of the accuracy on data with and without blanket occlusions. Taking into consideration the decrease in accuracy on the original dataset, the results of fine-tuning on the augmented dataset seem promising for cases where blanket occlusions are present.

Figures 7.1 and 7.2 display annotations both from both models as well as the original image. Note that on figure 7.1 the fine-tuned model more accurately predicted the location of the occluded foot. And on figure 7.2 it predicted the location of the feet, at the cost of positioning the shoulders too far down. This is consistent with the quantitative results: the fine-tuned model performs better than the pre-trained one when there are blanket occlusions, but worse when there are no occlusions.

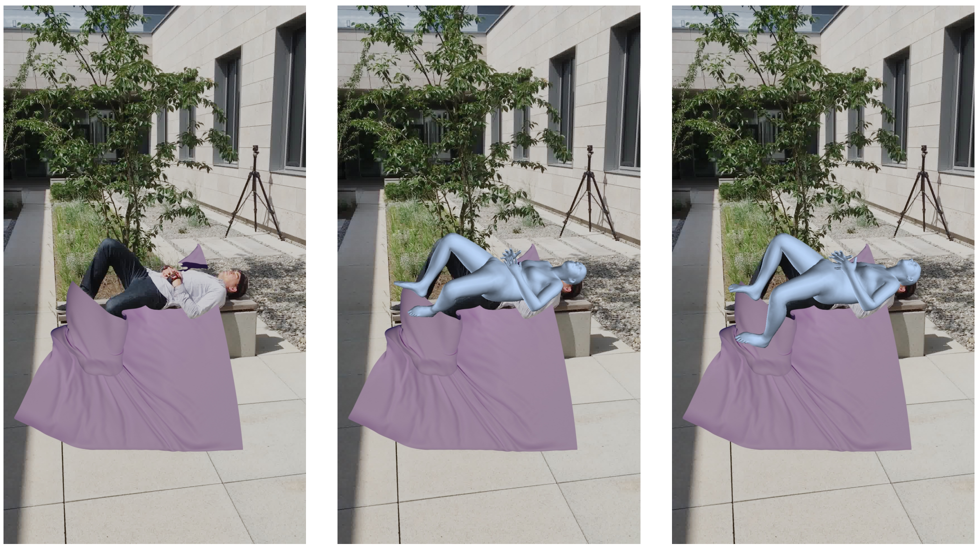


Figure 7.1: From left to right: A frame from BlanketGen-3DPW, the same frame with body model annotations generated with the pre-trained model, and with the fine-tuned model.

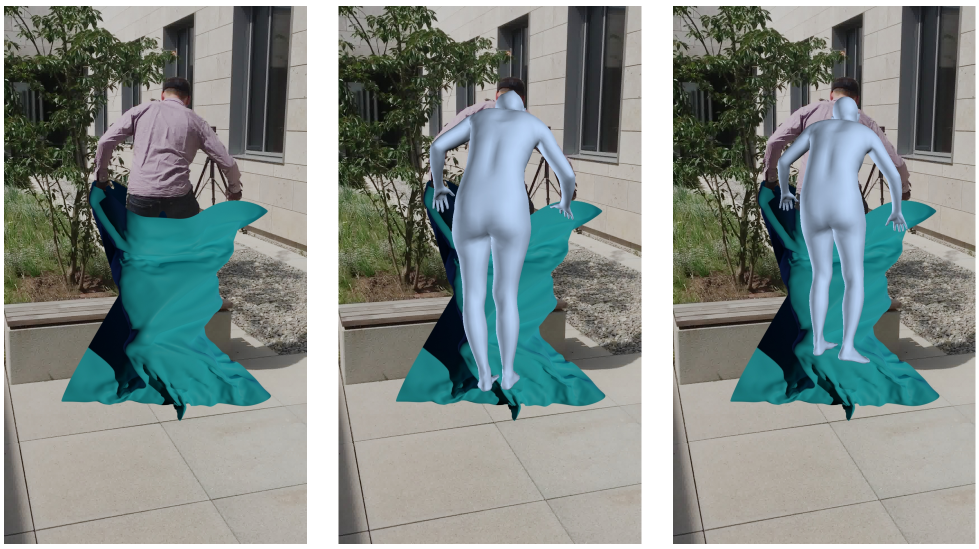


Figure 7.2: From left to right: A frame from BlanketGen-3DPW, the same frame with body model annotations generated with the pre-trained model, and with the fine-tuned model.

The qualitative results corroborate this, since the fine-tuned model only performed better than the pre-trained one when there were full blanket occlusions. This improvement in performance in cases with full real blanket occlusions bodes well for this approach as a solution to HPE in real world scenarios, although the performance in these cases was still underwhelming.

The decrease in accuracy on unoccluded subjects could be caused by the 30 epochs of fine-tuning not including data from the other datasets that the model was originally trained on, as well as the fairly low number of epochs of fine-tuning. If this is the case then this issue could be addressed by including data from Human3.6M, MPI-INF-3DHP, 3DPW, and MSCOCO in the fine-tuning, although this would drastically increase the processing time required.

All in all, computing power was the main limiting factor of this dissertation, as both Blanket-Gen and training the architecture are extremely computationally expensive.

Besides the results, this dissertation also introduced two datasets, BlanketSet, and BlanketGen-3DPW.

BlanketSet is a large dataset with RGB, IR, and depth video of well-defined movement sequences in an EMU bed, with varied lighting, movement frequencies, and blankets. Because of this, it can function as an effective benchmark dataset to evaluate the performance of HPE systems on EMUs. If ground-truth annotations are created in the future BlanketSet could also be used for training said systems.

BlanketGen-3DPW is a version of 3DPW with synthetic blanket occlusions and as such can fulfill the same purposes as 3DPW, but in contexts where blanket occlusions are present. Because it is formatted the same as 3DPW, it can be used as a drop-in replacement for 3DPW, as it was in this dissertation.

For these reasons, both BlanketSet and BlanketGen-3DPW could be published in the future to aid further research in this area.

Chapter 8

Conclusions and Future Work

Blanket occlusions reduce the performance of vision-based HPE systems whenever they are present; this is an issue for HPE in epilepsy monitoring units, where patients are often lying under blankets.

The work presented in this dissertation explored the potential of augmenting HPE datasets with synthetic blanket occlusions as a solution to this problem. A pipeline to augment the 3DPW dataset was implemented and an augmented dataset called BlanketGen-3DPW was generated; BlanketGen-3DPW was then used to fine-tune a DL HPE model that was originally trained on several datasets without blanket occlusions. This fine-tuned model and the original model were evaluated on both 3DPW and BlanketGen-3DPW and the results were compared.

The fine-tuned model performed better on BlanketGen-3DPW, but worse on 3DPW. The improvement on BlanketGen-3DPW was expected and matches the results in the literature, but the reduction in accuracy on 3DPW was unexpected. This loss in accuracy could be caused by the fine-tuning not including the original datasets as well as by the low number of epochs in the training; the pre-trained model was trained for 200 epochs with several datasets combined, with the setup used for fine-tuning on BlanketGen-3DPW this would have taken several months.

The qualitative results on BlanketSet also suggest that the fine-tuned model improves particularly in HPE of humans under blankets, since it outperformed the pre-trained model only in cases where the entire body of the participant was occluded. This improvement is also promising as it shows that the performance boost translates to real-world scenarios, rather than only existing on BlanketGen-3DPW.

A new dataset called BlanketSet was acquired which contains recordings of 14 subjects lying down in beds performing 8 different movement sequences with different blankets covering both their lower body as well as the entire body excluding the face. The same sequences were also recorded without blanket occlusions.

BlanketSet was then used for qualitative evaluation of the impact of fine-tuning the model on BlanketGen-3DPW; for this, a survey was done where participants were asked to evaluate on a scale from 0 to 10 the quality of joint position estimations of sequences from BlanketSet. Each sequence had estimations generated both with the fine-tuned model and with the original model.

8.1 Future Work

While this dissertation couldn't fully explore whether synthetic blanket occlusions are a viable approach to improve DL HPE systems, it did have promising results; it reaffirms what was already established in [47] and [48], and brings up new questions about the effect of training on augmented datasets in HPE of unoccluded subjects, namely does it necessarily reduce the quality of HPE on unoccluded subjects or can that be overcome with further training? And is there an ideal proportion of occluded and unoccluded subjects in the training dataset? Future research is needed to answer these questions.

Besides that, this dissertation couldn't explore the value of temporal information provided by synthetic blanket occlusions: in [47] recurrent neural networks saw the most improvement from the augmentation, which implies that this temporal information is particularly important. To test this, the methodology used in this dissertation could be reused with an HPE system that uses temporal information rather than HybrIK.

Annotation of BlanketSet remains a challenge to be tackled as well. Maybe a semi-automatic approach making use of optical flow like the one proposed in [11] could be used. The initial idea of using a DL HPE to annotate the sequences with no blanket occlusions and then manually adjusting those annotations to fit the sequences with blankets remains a possibility as well, but it would require immense effort considering the size of the dataset. If good enough results can be achieved with the approach proposed in this dissertation then a prior for the ground truth could be acquired that way, although it would still almost certainly require some manual adjustments. Perhaps a mix of these approaches could be used, with a DL HPE system providing a prior that is then improved with optical flow and finally manually adjusted when needed.

An approach at tackling the task of HPE for subjects in beds that is less related to this dissertation but still somewhat similar to it would be to record a fully synthetic dataset, similar to AGORA [34], of subject in beds. Soft body simulation for the mattress, rigid SMPL body models executing movements recorded with IMUs, and cloth simulation for the blankets could provide some level of realism. As it would be synthetic absolute control over the scenarios recorded could be attained, as well as perfectly accurate ground truth.

Appendix A

Appendix

A.1 Informed Consent Form

INESCTEC

INFORMED CONSENT TO PARTICIPATE IN INVESTIGATION PROJECT

Please read the following information. If you think something is incorrect or unclear, don't hesitate to ask for more information through the emails: joao.neves.carmona@gmail.com, tamas.karacsony@inesctec.pt

If you wish to participate in the study we request your consent, which you can give by signing the document.

You will be given a copy of this consent form so you can read it at any time and keep the necessary contact information.

Participation in the study is voluntary, the participant can quit it at any time without any consequence, needing only to contact the person responsible through the email specified above.

This document will remain in the possession of the responsible entity, with the only information shared with the project group being the approval of this consent form through a numerical code. This code will be the pseudo-anonymity mechanism used in which only this code, and not your name, is divulged. The connection between the code and your name will remain confidential with only the responsible entity having access to it.

1.PROJECT DESCRIPTION

Title: BlanketGen: Synthetic Blanket Occlusion Generation For The Augmentation Of Motion Capture Datasets

Responsible Entity: INESC TEC

General: The BlanketGen project aims to develop a pipeline to augment datasets used for motion capture with synthetic blanket occlusions. This augmentation could lead to the improvement of image processing motion capture systems that make use of deep learning, particularly in scenarios where the subject is significantly occluded by blankets.
If successful, the augmentation pipeline could be used to aid with seizure detection and classification in Epilepsy Monitoring Units (EMUs).
In order to test the pipeline, a dataset will be created that includes data of movement with and without real blankets.

Procedure: You will be asked to perform a sequence of motions in the bed of the EMU while being recorded by an Azure Kinect camera, which records RGB-IR-D data (this means it records visible colors as well as infra-red and depth data). This data will be saved according to the data protection rules (General Data Protection Regulation - GDPR) as described in this document. All instructions will be explained to you by a member of the project.

Hospital: The University Hospital Center of São João has provided the location for the data acquisition as well as clinical opinion on the data acquisition protocol. However, it will not be involved in the processing of data after it is acquired.

Risks: There are no risks associated with participation in this study.

Benefits: Your participation in this study will help evaluate the data augmentation pipeline, which could improve motion capture systems for patients in beds, and, in turn, the classification and detection of epileptic seizures in EMUs. This would lead to a quality of life improvement for the patients that get evaluated in the EMUs.

Financial compensations and/or costs: There is no financial compensation or cost associated with participation in this study.

2.PROCESSING OF PERSONAL DATA

Contact information: You will be requested information to allow the responsible entity to contact you should the need arise, namely an email and/or phone number. You have the right to deny this request.

Finality of the processing: The data collected will be treated according to the applicable national legislation and of the EU and will be used by the investigators for the purpose of scientific investigation and publications. Specifically, it will be used to judge the impact of blanket occlusion augmentations in deep learning motion capture systems.

Responsible for the processing: INESC TEC – Instituto de Engenharia de Sistemas e Computadores, Tecnologia e Ciência, Campus da FEUP, Rua Dr. Roberto Frias, 4200 - 465 Porto.

Confidentiality: All the data acquired will be analyzed for scientific purposes in the scope of the project in question; all those involved in this project have a confidentiality and non-disclosure agreement of the data and information of personal character acquired from it; some steps will be taken to protect your individual identity, namely: a) each participant will be assigned a code; b) all data acquired will be saved pseudo-anonymously. However, since the dataset consists of RGB-IR-D video of your body and face and will be saved as-is without any blurring or any other similar anonymization techniques, it will still be possible to identify you from the data.

Retention period: The dataset itself will be made public, so it doesn't have a retention period (it will exist indefinitely). All the personal information, including the contact information and the association between your code and your person, will only be kept for 3 months after the conclusion of the project.

Data Accessibility: The RGB-D-IR video data collected will be made publicly available as part of the dataset, your personal information will not be disclosed, however the videos of the dataset will be published as-is.

Personal Data Sharing: The project results will be divulged/published in the scope of scientific publications. Besides publications, the dataset itself (which isn't anonymized in terms of video data) will be publicly available. So it may be accessed by entities located in other countries outside the European Union.

Rights of the Data Subject: As the subject of the data, the law grants you the following rights: Information, Access, Rectification, Erasure, Opposition, and Portability. In the case of quitting, the data acquired prior to the quitting will still be utilized, if not requested otherwise.

The exercise of these rights could be limited in respect to the terms and conditions provided by the applicable national legislation and of the EU, insofar as the exercise would be susceptible to impossibilitating or seriously hindering the achievements of the objectives of the processing for investigation purposes, and only within necessary for the pursuit of these ends.

Data Protection Officer: For any questions, exercise of rights over personal data, requests, or complaints about the data processing, please contact our data protection officer at the address: dpo@inesctec.pt

3.INFORMED CONSENT FORM	
1. I read and understand the information about the project, including the identity of the responsible person, the type of data collected, the purpose of the collection and the respective processing. 2. I read and understand the information about how the data is stored and for how long, including what will happen to my data in the case of ceasing my participation in the project. 3. I have been given the opportunity to ask questions and to clarify any doubts about the project. 4. I understand I can quit participating in the project at any point, without needing to provide any justification and without suffering any penalty or having my motives questioned. 5. I understand how to communicate my decision to quit, as well as how to exercise my rights as the data subject.	☐ ☐ ☐ ☐ ☐

The Participant:

I declare I have read and understood this document, as well as the verbal information that has been given to me previously. As such, I accept to participate in this study and allow the use of the data I offer voluntarily.

Code: __ __ __ __

(Format: 2 letters of the name of the responsible entity and a number with 2 digits; example for INESC TEC: "IT01")

Name: ____

Signature: ______________________________ Date:____/____/_______

(DD/MM/AAAA)

References

[1] Eric Haines and Tomas Akenine-Möller, editors. *Ray Tracing Gems*. Apress, 2019. http://raytracinggems.com.

[2] Blender 3.2 reference manual — blender manual. URL: https://docs.blender.org/manual/en/latest/index.html.

[3] Michael A. Nielsen. *Neural Networks and Deep Learning*. Determination Press, 2015. URL: http://neuralnetworksanddeeplearning.com.

[4] Jianxin Wu. Introduction to convolutional neural networks. 2017.

[5] Ian Goodfellow, Yoshua Bengio, and Aaron Courville. *Deep Learning*. MIT Press, 2016. http://www.deeplearningbook.org.

[6] Yong Du, Wei Wang, and Liang Wang. Hierarchical recurrent neural network for skeleton based action recognition.

[7] Azure kinect dk hardware specifications | microsoft docs. URL: https://docs.microsoft.com/en-us/azure/kinect-dk/hardware-specification.

[8] Christian Peter Klingenberg and John H Graham. Analyzing fluctuating asymmetry with geometric morphometrics: Concepts, methods, and applications. *Symmetry 2015, Vol. 7, Pages 843-934*, 7:843–934, 6 2015. URL: https://www.mdpi.com/2073-8994/7/2/843/htmhttps://www.mdpi.com/2073-8994/7/2/843, doi:10.3390/SYM7020843.

[9] Eline van der Kruk and Marco M. Reijne. Accuracy of human motion capture systems for sport applications; state-of-the-art review. *European journal of sport science*, 18:806–819, 7 2018. URL: https://pubmed.ncbi.nlm.nih.gov/29741985/, doi:10.1080/17461391.2018.1463397.

[10] Tamás Karácsony, Anna Mira Loesch-Biffar, Christian Vollmar, Soheyl Noachtar, and João Paulo Silva Cunha. A deep learning architecture for epileptic seizure classification based on object and action recognition. In *ICASSP 2020 - 2020 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 4117–4121, 2020. doi:10.1109/ICASSP40776.2020.9054649.

[11] Joao Paulo Silva Cunha, Hugo Miguel Pereira Choupina, Ana Patrícia Rocha, José Maria Fernandes, Felix Achilles, Anna Mira Loesch, Christian Vollmar, Elisabeth Hartl, and Soheyl Noachtar. Neurokinect: A novel low-cost 3dvideo-eeg system for epileptic seizure motion quantification. *PLoS ONE*, 11, 1 2016. URL: /pmc/articles/PMC4723069//pmc/articles/PMC4723069/?report=abstracthttps://www.ncbi.nlm.nih.gov/pmc/articles/PMC4723069/, doi:10.1371/JOURNAL.PONE.0145669.

[12] Jinbao Wang, Shujie Tan, Xiantong Zhen, Shuo Xu, Feng Zheng, Zhenyu He, and Ling Shao. Deep 3d human pose estimation: A review. *Computer Vision and Image Understanding*, 210:103225, 2021. URL: https://www.sciencedirect.com/science/article/pii/S1077314221000692, doi:https://doi.org/10.1016/j.cviu.2021.103225.

[13] Timo von Marcard, Roberto Henschel, Michael Black, Bodo Rosenhahn, and Gerard Pons-Moll. Recovering accurate 3d human pose in the wild using imus and a moving camera. In *European Conference on Computer Vision (ECCV)*, sep 2018.

[14] Jiefeng Li, Chao Xu, Zhicun Chen, Siyuan Bian, Lixin Yang, and Cewu Lu. Hybrik: A hybrid analytical-neural inverse kinematics solution for 3d human pose and shape estimation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 3383–3393, June 2021.

[15] Blender python api documentation — blender python api. URL: https://docs.blender.org/api/current/index.html.

[16] Glossary - blender manual. URL: https://docs.blender.org/manual/en/latest/glossary/index.html.

[17] Cyrill Stachniss. Photogrammetry & robotics lab camera parameters: Extrinsics and intrinsics. URL: https://www.ipb.uni-bonn.de/5min/.

[18] Saad Albawi, Tareq Abed Mohammed, and Saad Al-Zawi. Understanding of a convolutional neural network. In *2017 International Conference on Engineering and Technology (ICET)*, pages 1–6, 2017. doi:10.1109/ICEngTechnol.2017.8308186.

[19] Sepp Hochreiter and Jürgen Schmidhuber. Long short-term memory. *Neural Computation*, 9(8):1735–1780, 1997. doi:10.1162/neco.1997.9.8.1735.

[20] Qian Chen, Xiaodan Zhu, Zhenhua Ling, Si Wei, Hui Jiang, and Diana Inkpen. Enhanced lstm for natural language inference. *ACL 2017 - 55th Annual Meeting of the Association for Computational Linguistics, Proceedings of the Conference (Long Papers)*, 1:1657–1668, 9 2016. URL: http://arxiv.org/abs/1609.06038http://dx.doi.org/10.18653/v1/P17-1152, doi:10.18653/v1/P17-1152.

[21] A. Stelzer, K. Pourvoyeur, and A. Fischer. Concept and application of lpm - a novel 3-d local position measurement system. *IEEE Transactions on Microwave Theory and Techniques*, 52(12):2664–2669, 2004. doi:10.1109/TMTT.2004.838281.

[22] Judd S. Day, Genevieve A. Dumas, and Duncan J. Murdoch. Evaluation of a long-range transmitter for use with a magnetic tracking device in motion analysis. *Journal of Biomechanics*, 31:957–961, 10 1998. doi:10.1016/S0021-9290(98)00089-X.

[23] Soshi Shimada, Vladislav Golyanik, Weipeng Xu, and Christian Theobalt. Physcap. *ACM Transactions on Graphics (TOG)*, 39:16, 11 2020. URL: https://dl.acm.org/doi/abs/10.1145/3414685.3417877, doi:10.1145/3414685.3417877.

[24] Jörg Spörri, Christian Schiefermüller, and Erich Müller. Collecting kinematic data on a ski track with optoelectronic stereophotogrammetry: A methodological study assessing the feasibility of bringing the biomechanics lab to the field. *PLOS ONE*, 11:e0161757, 8 2016. URL: https://journals.plos.org/plosone/article?id=10.1371/journal.pone.0161757, doi:10.1371/JOURNAL.PONE.0161757.

[25] Ante Panjkota. Outline of a qualitative analysis for the human motion in case of ergometer rowing.

[26] Markus Windolf, Nils Götzen, and Michael Morlock. Systematic accuracy and precision analysis of video motion capturing systems–exemplified on the vicon-460 system. *Journal of biomechanics*, 41:2776–2780, 8 2008. URL: https://pubmed.ncbi.nlm.nih.gov/18672241/, doi:10.1016/J.JBIOMECH.2008.06.024.

[27] M. A. Brodie, A. Walmsley, and W. Page. Dynamic accuracy of inertial measurement units during simple pendulum motion. *http://dx.doi.org/10.1080/10255840802125526*, 11:235–242, 6 2008. URL: https://www.tandfonline.com/doi/abs/10.1080/10255840802125526, doi:10.1080/10255840802125526.

[28] Xsens. URL: https://www.xsens.com/motion-capture.

[29] Neuron. URL: https://neuronmocap.com/.

[30] Neurogait: Mobile recording of gait in patients with neurological diseases. URL: https://repositorio-aberto.up.pt/handle/10216/76686?mode=full.

[31] Catalin Ionescu, Dragos Papava, Vlad Olaru, and Cristian Sminchisescu. Human3.6m: Large scale datasets and predictive methods for 3d human sensing in natural environments. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 36(7):1325–1339, jul 2014.

[32] Dushyant Mehta, Helge Rhodin, Dan Casas, Pascal Fua, Oleksandr Sotnychenko, Weipeng Xu, and Christian Theobalt. Monocular 3d human pose estimation in the wild using improved cnn supervision. In *3D Vision (3DV), 2017 Fifth International Conference on*. IEEE, 2017. URL: http://gvv.mpi-inf.mpg.de/3dhp_dataset, doi:10.1109/3dv.2017.00064.

[33] 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. *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, 8693 LNCS:740–755, 5 2014. URL: https://arxiv.org/abs/1405.0312v3, doi:10.48550/arxiv.1405.0312.

[34] Priyanka Patel, Chun-Hao P. Huang, Joachim Tesch, David T. Hoffmann, Shashank Tripathi, and Michael J. Black. AGORA: Avatars in geography optimized for regression analysis. In *Proceedings IEEE/CVF Conf. on Computer Vision and Pattern Recognition (CVPR)*, June 2021.

[35] Kathleen M Robinette, Sherri Blackwell, Hein Daanen, Mark Boehmer, Scott Fleming, Tina Brill, David Hoeferlin, and Dennis Burnsides. United states air force research laboratory civilian american and european surface anthropometry resource (caesar) final report, volume i: Summary human effectiveness directorate crew system interface division 2255 h street wright-patterson afb oh 45433-7022. 1997.

[36] Size usa. URL: https://www.scan2fit.com/sizeusa/about.php.

[37] Matthew Loper, Naureen Mahmood, Javier Romero, Gerard Pons-Moll, and Michael J. Black. Smpl. *ACM Transactions on Graphics (TOG)*, 34, 10 2015. URL: https://dl.acm.org/doi/abs/10.1145/2816795.2818013, doi:10.1145/2816795.2818013.

[38] Javier Romero, Dimitrios Tzionas, and Michael J. Black. Embodied hands: Modeling and capturing hands and bodies together. *ACM Transactions on Graphics, (Proc. SIGGRAPH Asia)*, 36(6), November 2017.

[39] Georgios Pavlakos, Vasileios Choutas, Nima Ghorbani, Timo Bolkart, Ahmed A. A. Osman, Dimitrios Tzionas, and Michael J. Black. Expressive body capture: 3D hands, face, and body from a single image. In *Proceedings IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*, pages 10975–10985, 2019.

[40] Ahmed A A Osman, Timo Bolkart, and Michael J. Black. STAR: A sparse trained articulated human body regressor. In *European Conference on Computer Vision (ECCV)*, pages 598–613, 2020. URL: https://star.is.tue.mpg.de.

[41] Federica Bogo, Angjoo Kanazawa, Christoph Lassner, Peter Gehler, Javier Romero, and Michael J. Black. Keep it smpl: Automatic estimation of 3d human pose and shape from a single image. *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, 9909 LNCS:561–578, 7 2016. URL: https://arxiv.org/abs/1607.08128v1, doi:10.1007/978-3-319-46454-1_34.

[42] Leonid Pishchulin, Eldar Insafutdinov, Siyu Tang, Bjoern Andres, Mykhaylo Andriluka, Peter Gehler, and Bernt Schiele. Deepcut: Joint subset partition and labeling for multi person pose estimation. *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, 2016-December:4929–4937, 11 2015. URL: https://arxiv.org/abs/1511.06645v2, doi:10.1109/CVPR.2016.533.

[43] Shanyan Guan, Jingwei Xu, Michelle Z. He, Yunbo Wang, Bingbing Ni, and Xiaokang Yang. Out-of-domain human mesh reconstruction via dynamic bilevel online adaptation. *IEEE TRANSACTIONS ON PATTERN ANALYSIS AND MACHINE INTELLIGENCE*, XX, 11 2021. URL: https://arxiv.org/abs/2111.04017v1, doi:10.48550/arxiv.2111.04017.

[44] Muhammed Kocabas, Chun Hao P. Huang, Otmar Hilliges, and Michael J. Black. Pare: Part attention regressor for 3d human body estimation. *Proceedings of the IEEE International Conference on Computer Vision*, pages 11107–11117, 4 2021. URL: https://arxiv.org/abs/2104.08527v2, doi:10.48550/arxiv.2104.08527.

[45] Zhihao Li, Jianzhuang Liu, Zhensong Zhang, Songcen Xu, and Youliang Yan. Cliff: Carrying location information in full frames into human pose and shape estimation. 8 2022. URL: https://arxiv.org/abs/2208.00571v1, doi:10.48550/arxiv.2208.00571.

[46] Ailing Zeng, Xuan Ju, Lei Yang, Ruiyuan Gao, Xizhou Zhu, Bo Dai, and Qiang Xu. Deciwatch: A simple baseline for 10x efficient 2d and 3d pose estimation. 3 2022. URL: https://arxiv.org/abs/2203.08713v2, doi:10.48550/arxiv.2203.08713.

[47] Felix Achilles, Alexandru-Eugen Ichim, Huseyin Coskun, Federico Tombari, Soheyl Noachtar, and Nassir Navab. Patient mocap: Human pose estimation under blanket occlusion for hospital monitoring applications. pages 491–499, 10 2016. doi:10.1007/978-3-319-46720-7_57.

[48] Henry M. Clever, Patrick Grady, Greg Turk, and Charles C. Kemp. Bodypressure – inferring body pose and contact pressure from a depth image. 5 2021. URL: https://arxiv.org/abs/2105.09936v1.

[49] Coco - common objects in context. URL: https://cocodataset.org/#format-data.