MODELING HUMANS AT REST WITH APPLICATIONS TO ROBOT ASSISTANCE

A Dissertation
Presented to
The Academic Faculty

By

Henry M. Clever

In Partial Fulfillment
of the Requirements for the Degree
Doctor of Philosophy in the
School of Biomedical Engineering

Georgia Institute of Technology and Emory University

December 2021

MODELING HUMANS AT REST WITH APPLICATIONS TO ROBOT ASSISTANCE

Thesis committee:

Dr. Charles C. Kemp
Dept. of Biomedical Engineering
Georgia Institute of Technology

Dr. C. Karen Liu
Dept. of Computer Science
Stanford University

Dr. James Hays
School of Interactive Computing, College of Computing
Georgia Institute of Technology

Dr. Greg Turk
School of Interactive Computing, College of Computing
Georgia Institute of Technology

Dr. Ayanna Howard
College of Engineering
Ohio State University

Date approved: December 10, 2021

One must imagine Sisyphus happy.

Albert Camus

For Robert Hershon

ACKNOWLEDGMENTS

I first thank Liuyu Ivy Chen for following me to Atlanta for 5 years to complete my Ph.D. Without her support, I am not sure if I would have completed this journey.

I thank my those in my family who have championed my work: my parents, grandparents, siblings, aunts, uncles, cousins, and the Chen family. I am grateful to Henry and Jane Evans for believing in me and for introducing me to the world of robotics.

I thank my many friends who have touched my life in a remarkable way (in alphabetical order by last name): Catrina Cooper, the Diepenbrocks, Tobin Hagler, Zech Harjo, Stephen Henry, the Johnsons, Austin Kirby, Chudan Li, Jordan Miller, Lisa Nadeau, Carson Whitelemons, and my ITP friends. I thank my colleagues: Naveen Balaji, Tapo Bhattacharjee, Samarth Brahmbhatt, Alex Clegg, Yash Chitalia, Zackory Erickson, Patrick Grady, Phillip Grice, Ari Kapusta, Matt Lamsey, Carlotta Mummolo, Mark Pacey, William Peng, Dustyn Roberts, Josh Wade, Wenhao Yu. I thank my mentors: Dr. Congcong Chai, Ben Esner, Dr. Ankur Handa, Dr. Vikram Kapila, Dr. Sarah Kieweg, Dr. Joo H. Kim, Dr. Carey Johnson, Andy Lee, Pat Lombard, Rob Roberson, Chris Tacklind, Dr. Peter TenPas, Jane Walsh. I also thank Dr. Miles Macklin, Dr. Sarah Ostadabbas, and Dr. Shuangjun Liu for creating tools and datasets that were important to this work.

I acknowledge the financial support offered by the National Science Foundation, and to the reviewers who had confidence in my fellowship applications.

I thank the members of my thesis committee for their continued support, feedback, and advice: Dr. James Hays, Dr. Ayanna Howard, Dr. C. Karen Liu, and Dr. Greg Turk. Finally, I thank my Ph.D. advisor Charlie Kemp for showing me how to think critically and to do great research in healthcare robotics.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

SUMMARY

The physical characteristics of the human body at rest have important implications. For example, they play a role in comfort, safety, posture, and health. Resting human poses belong to a class of poses distinct from active poses, but by comparison have been studied little in prior research. Machine perception of this class of poses would be beneficial to numerous applications, but it is complicated by line-of-sight occlusion from bedding. Pressure sensing mats are a promising alternative, but data is challenging to collect at scale. To overcome this, we use modern physics engines to simulate bodies resting on a soft bed with a pressure sensing mat. This method can efficiently generate data at scale for training deep neural networks. We present a deep model trained on this data that infers 3D human pose and body shape from a pressure image, and show that it transfers well to real world data. We also present a model that infers pose, shape and contact pressure from a depth image facing the person in bed, and it does so in the presence of blankets. This model similarly benefits from synthetic data, which is created by simulating blankets on the bodies in bed. We evaluate this model on real world data and compare it to an existing method that requires RGB, depth, thermal and pressure imagery in the input. Our model only requires an input depth image, yet it is 12% more accurate. Our methods are relevant to applications in healthcare, including patient acuity monitoring and pressure injury prevention. We demonstrate this work in the context of robotic caregiving assistance, by using it to control a robot to move to locations on a person's body in bed.

CHAPTER 1
INTRODUCTION AND BACKGROUND

1.1 Overview

People spend a substantial part of their lives at rest in bed. Machine perception for this class of human activity would be beneficial to numerous applications, but the complexity of it presents challenges. The human body at rest is characterized by a lack of physical exertion, which leads to substantial contact with the surrounding environment, as well as self-contact between parts of the body. Furthermore, bedding materials and self-occlusion that are common in environments where people rest, which impedes the ability of sensors to gather information about the body. This dissertation presents mechanical models of humans at rest and their surroundings, and provides ways of inferring human pose and body shape using unconventional sensing modalities.

Prior research has largely focused on active human poses. Examples of this include datasets for markerless human pose estimation [1, 2, 3], arm dynamics in the context of exoskeletons [4, 5], and walking stability [6, 7]. While the fundamental human structure and body composition remains the same, the distribution of pose and contact differs. Generally, there is little motion in the pose of a person resting, so time-series sensor data is uninformative for real-time inference. However, a dynamical model of how human bodies move to assume a resting pose could offer insight. We map random poses to a distribution of resting poses by dropping randomly posed articulated ragdoll bodies to rest on a soft mattress in a dynamics simulator (Figure 1.1 (a)).

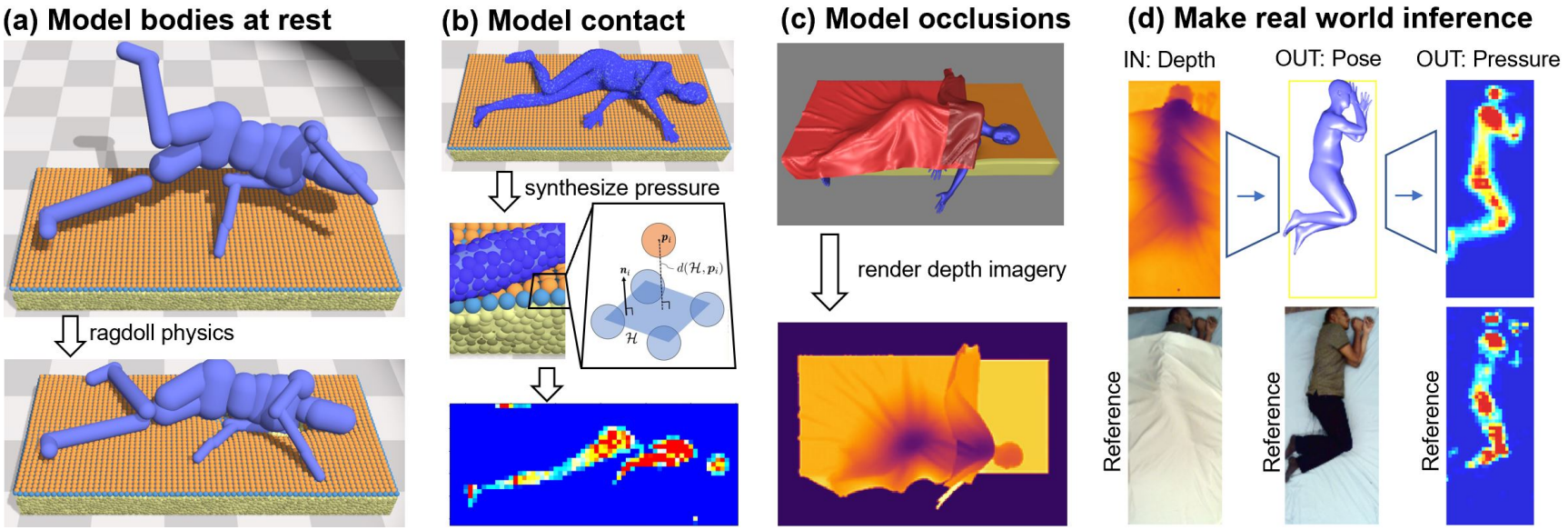

Figure 1.1: Overview of dissertation work. (a) Modeling bodies at rest with dynamic ragdoll physics. (b) Modeling contact with a synthetic pressure sensing mat underneath the body. (c) Modeling occlusions covering the body. (d) Using a deep network to infer body pose, shape and contact pressure of a human at rest from sensor data. In the example shown, the network inputs a depth image of a person underneath a blanket, and outputs human pose, body shape, and contact pressure underneath the person.

Human pose estimation is a well-known perception problem that explores how information from sensors can infer where the body is in space and how the limbs are configured. While early works in this area inferred a limited set of joint angles and/or positions in space, improved human representations allow for additional inference of the body shape, height, and weight. This is usually formulated as a supervised machine learning problem, where some input (e.g. an image) is fed into a model that extracts features and produces an estimate of the pose. State-of-the-art machine learning models for such a problem use convolutional neural networks (CNNs), which can require a substantial amount of data to train – typically tens of thousands of samples. Collecting real data at this scale would be challenging. Instead, our human body simulation can provide this: it can generate a large quantity of resting poses quickly, and it can also generate synthetic sensor images necessary for training.

This dissertation explores pressure and depth imagery for sensing a person. Pressure mats are a promising alternative to sensing a person at rest because they are not subject to line-of-sight occlusion. When placed under a person in bed, a pressure mat produces pressure images with features that spatially resemble a human figure. We treat pressure images like camera images in the context of deep learning and use them to infer a person's pose in bed. To generate synthetic pressure images, we introduce a method of simulating a pressure mat with a particle simulator [8] and a method to calibrate the mat (Figure 1.1 (b)). In comparison, simulating depth imagery is simple – a mere collection of distances between the camera and objects of interest in a scene. However, it requires realistic models of objects occluding the person. We accomplish this by simulating blankets covering the body at rest (Figure 1.1 (c)).

We present a series of deep models for perceiving humans at rest. The first model has an embedded kinematic skeleton model and infers joint angles, skeleton link lengths, and 3D joint positions [9] from a pressure image. It also includes a simple mechanical model that provides insight into the ambiguity associated with limbs raised off of the pressure mat. Our

next iteration infers a Skinned Multi-Person Linear (SMPL) 3D human parametric mesh model [10], which provides a detailed profile of body shape [11]. The third deep model also uses SMPL, but it inputs a depth image of a person underneath blankets and is able to infer the pressure distribution from the underlying surface onto the body [12] (Figure 1.1 (d)). Our deep models include a novel method of modeling pressure and depth image generation within the network using differentiable analytic geometry, which we show to be beneficial.

We describe two potential applications for this work in healthcare: (1) in-bed monitoring of patients and care receivers, and (2) robot assistance to people in bed. For example, the pose and body shape inferred by our system could be used to discern a patient's status and provide information to a clinician. We also suggest applying our method of inferring localized contact pressure from a depth camera to the problem of pressure injury prevention. For robot assistance, our methods may be useful for automating in-bed tasks including bathing assistance, itch scratching, dressing, and toileting. We provide a demonstration of our work being used to command a mobile manipulator robot to move to specific locations on a person's body in bed (see Figure 1.2). In the demonstration, a person in bed moves to different poses and a researcher commands a Hello Robot Stretch RE-1 robot [13] to move to joint positions on the person.

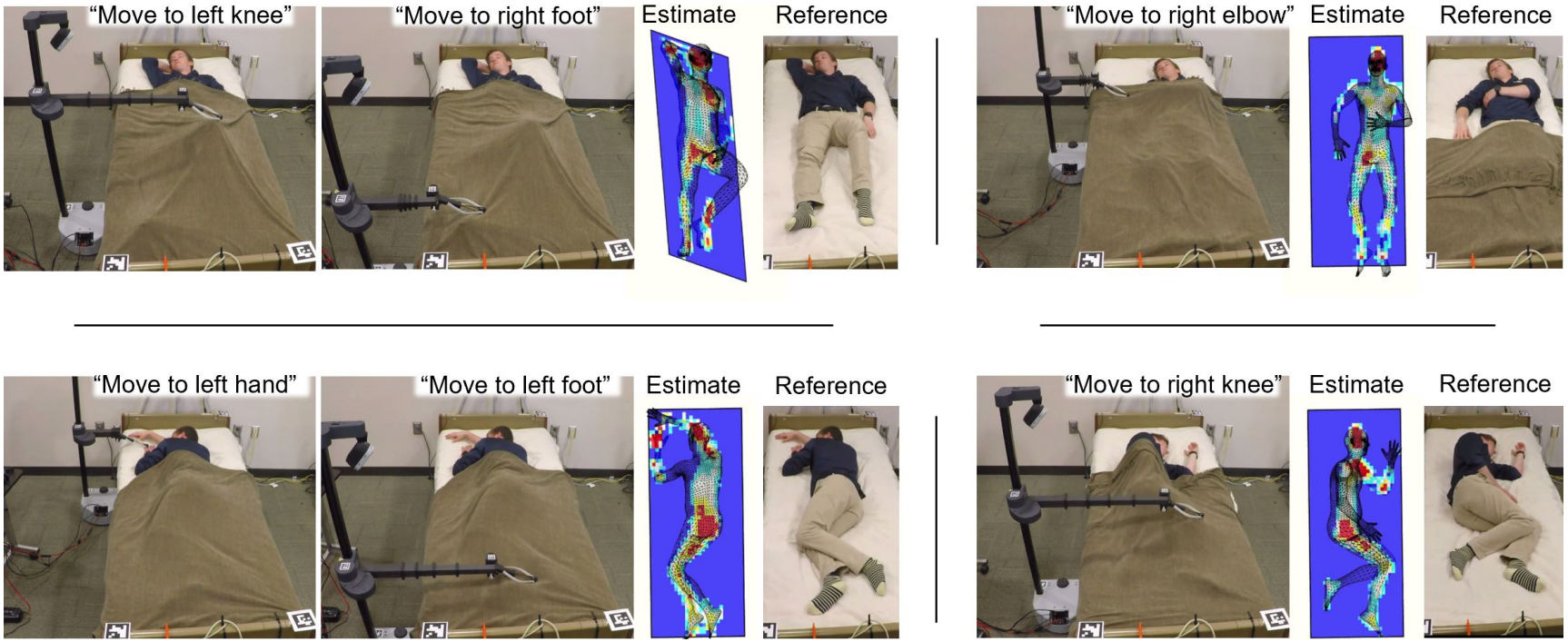

Figure 1.2: Demonstration of application to robot assistance. A researcher commands a Hello Robot Stretch RE-1 robot to move to locations on the person's body based on an estimate of their pose in bed. In this demonstration, the model inputs a depth image from an overhead camera and outputs human pose, body shape, and contact pressure. Contact pressure is shown underneath the estimate of pose, and a reference without blankets is provided to qualitatively verify the pose estimate.

1.2 Contributions

This dissertation includes contributions that cross the fields of computer vision, machine learning, optimization, haptics, robot and human mechanics, and physics simulation. An early part of this work compares methods for inferring human pose from a pressure image, and provides evidence that modern deep learning methods, specifically convolutional neural networks (CNNs), are suitable for the problem and outperform traditional machine learning algorithms. We introduce a differentiable articulated human skeleton model that can be embedded into the deep network to learn a set of kinematically constrained 3D human joint positions. The skeleton is then used to create a mechanical model of pressure image ambiguity to analyze human limbs that are out of contact with the pressure sensing mat. We apply the method of Monte Carlo dropout to human pose estimation for measuring uncertainty, and show that uncertainty is higher for limbs out of contact with the sensor.

This thesis makes a number of contributions in modeling the physics of humans at rest and creating synthetic data that is suitable for training deep models. We show that these deep models transfer well to real data for inferring body pose and shape of people resting from diverse sensing modalities. We introduce a physics-based method to simulate human bodies at rest on a soft mattress, which converts randomly sampled human poses to resting poses. We also introduce a method to simulate pressure sensing arrays using a particle simulator with position-based dynamics [8], and cover the simulated soft mattress with the pressure sensing array to generate synthetic pressure images. This simulator is used to create the publicly released PressurePose dataset[1], which includes 206K synthetic pressure images (184K train / 22K test) with associated 3D human poses and shapes. The dataset also includes 1,051 real pressure images and RGB-D images from 20 human participants that are used for testing[2]. We present a deep learning model, PressureNet[3], that is trained on the PressurePose synthetic data and estimates 3D human pose and body shape given a

[1] PressurePose synthetic data: `doi.org/10.7910/DVN/IAPI0X`

[2] PressurePose real data: `doi.org/10.7910/DVN/KOA4ML`

[3] PressurePose and PressureNet code: `github.com/Healthcare-Robotics/bodies-at-rest`

pressure image and gender.

Besides inferring pose from pressure, we present BodyPressure[4], a method that takes as input depth images of a person in bed from an overhead camera and infers the body pose, body shape, contact pressure, and localized regions of high pressure density. BodyPressure similarly leverages synthetic data for model training and has the addition of depth imagery, which is created by simulating blankets on the body at rest and rendering depth from a pinhole camera. This dissertation provides evidence that BodyPressure is robust to these occlusions, and shows that it performs best when training with a mixed bag of both real and synthetic data. We publicly release the accompanying synthetic dataset, BodyPressureSD, consisting of 97,495 bodies at rest with pressure images and depth images rendered with and without simulated blankets. We also present an optimization method to fit SMPL bodies to a real dataset, and use it to create SLP-3Dfits[5], a dataset consisting of 4,545 SMPL bodies fit to the SLP dataset [14] that resolves depth perspective ambiguity by fitting human mesh models to the point clouds from the dataset.

1.3 Organization of Dissertation

This dissertation is organized as follows. Chapter 2 describes a method of inferring human pose in bed from contact pressure [9]. It is our initial work modeling humans at rest, which uses a human skeleton model of pose at rest and compares deep networks to traditional machine learning models. Chapter 3 describes our method of simulating bodies at rest, which is used to generate data for training deep networks [11]. It presents an improved method of pose and body shape estimation that uses only synthetic data for training. Chapter 4 describes a method of inferring human pose, body shape, and contact pressure from a depth image of a person under blankets [12]. It uses both real and synthetic data for training a deep network, which involves simulating blankets on bodies at rest.

[4]BodyPressure code and data: `github.com/Healthcare-Robotics/BodyPressure`
[5]SLP-3Dfits data: `github.com/pgrady3/SLP-3Dfits`

CHAPTER 2

INFERRING HUMAN POSE IN BED FROM CONTACT PRESSURE

Various circumstances, such as illness, injury, or longterm disabilities can result in people receiving assistance in bed. Previous work has shown how robots can provide assistance with ADLs [15, 16, 17], but providing assistance to a person in bed can be challenging. Estimating the pose of a person's body could enable robots to provide better assistance. Typical methods of body pose estimation use line-of-sight sensors, such as RGB cameras, which can have difficulties when the body is occluded by blankets, loose clothing, medical equipment, over-bed trays and other common items in healthcare settings, such as hospitals. A pressure-sensing mat on the bed can allow for estimation of the body's pose in a manner that is less sensitive to bedding materials and surrounding objects [16, 18, 19, 20]. Prior work with pressure images has not addressed a number of concerns key to the success of robot assistance in bed, namely (1) pose estimation in 3D for either flat or non-flat beds and (2) appropriately dealing with uncertainty when the pose estimate may be inaccurate.

This chapter describes two methods to infer human pose on a configurable bed from a pressure image: The first method infers the overall body position using the center-of-mass on a pressure mat (Figure 2.1(a), section 2.2, [21]), and the improved second method infers a 3D human skeleton on both flat and non-flat beds using deep learning (Figure 2.1(b), section 2.3, [9]). Both of the methods operate in real time, and we additionally propose a method to measure confidence in each estimated joint position for the person in bed. We provide evidence that the improved method works for some challenging scenarios, such as when the bed and human are configured in a seated posture, and when limbs are raised off of the pressure sensing mat. Further, we release a motion capture labeled dataset of over 28,000 pressure images across 17 human participants, in addition to our open-source code.

For the deep learning method, we propose and compare two convolutional neural net-

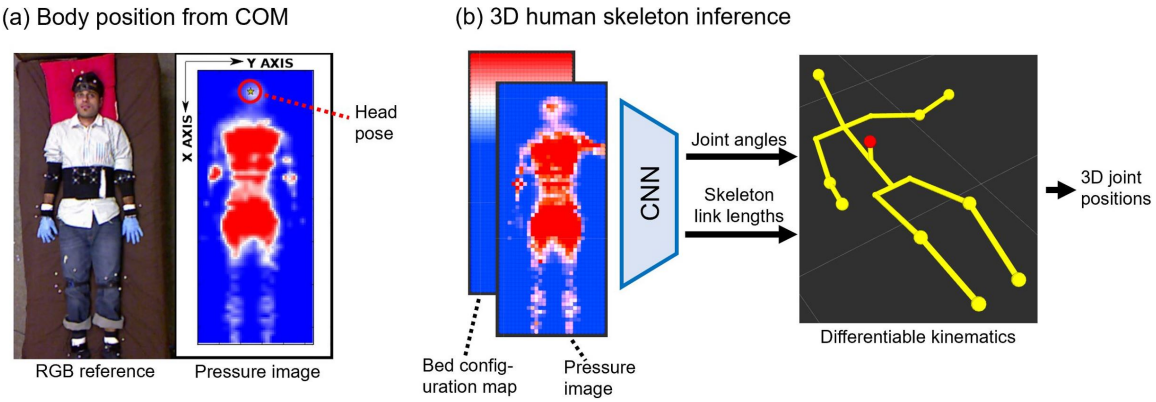

Figure 2.1: Two methods to infer human pose in bed from contact pressure: (a) Human body position measured by the center of mass on the pressure image. (b) Deep learning model that infers a 3D human skeleton on a configurable bed.

work (CNN) architectures to learn a mapping from a pressure image and bed configuration to 3D human joint positions: the head, neck, shoulders, elbows, wrists, chest, hips, knees, and ankles. The first method directly regresses to the 3D ground truth labels (x, y, z). The second method embeds a 17-joint kinematics skeleton model of the human to the last layer of the ConvNet to enforce constraints on bone lengths and joint angles. This provides a more complete pose representation with additional unlabeled joints and latent space joint angle estimates. We introduce a new architecture that adjusts the skeleton model for differently sized people, while providing comparison to a constant link length skeleton model. We train the kinematics CNNs end-to-end, backpropagating from 3D joint Euclidean error through the kinematics model. We compare our CNN methods to baseline data-driven algorithms, including ridge regression and k-nearest neighbors.

Our configurable bed, introduced previously as Autobed [16, 22], features adjustable height, leg rest angle, and head rest angle. Autobed can sense its own state, sense the pressure distribution of the person on the bed, and communicate with other devices. In this work, we estimate the human joint positions in two Autobed configurations by adjusting the bed's head rest: supine (0° flat) and seated (60° incline).

Lack of contact by limbs or other body parts presents a challenging issue for the pressure image modality. In this work, we consider common poses where this issue arises, such

as an arm raised in the air resembling a double inverted pendulum. Among other poses, this case demonstrates where the pressure image can be similar for different configurations of the arm. In such a case, the pressure data may be insufficient to confidently estimate the pose of the arm. An estimate of confidence or model uncertainty can be valuable, for example to allow an assistive robot to reject low-confidence estimates by removing them from a list of potential goals in task plan execution. To estimate model uncertainty, we use Monte Carlo dropout, a method proposed by Gal and Ghahmarani [23]. With this method, we perform a number of stochastic forward passes through the CNN during test time and compute the joint position and joint position confidence from the moments of the output distribution.

2.1 Related Works

Markerless human pose estimation is a challenging problem complicated by environment factors, the human pose configurations of interest, and data type. Relatively few researchers have used pressure images for human pose estimation in bed [18, 19, 20], while many used cameras in myriad environments and poses[24, 25, 26, 27, 28, 29, 30, 31, 32, 33, 34, 35, 36]. Researchers have increasingly explored data-driven methods such as ConvNets, from model-free networks to inclusion of models in various architectures. Here we discuss research with pressure images, data-driven methods, and measuring network uncertainty.

2.1.1 Pressure-Image-based Work

Prior pressure-image-based pose estimation work has fit 2D kinematic models to pressure image features. In a series of papers including [18], Harada et al. used a kinematic model to create a synthetic database for comparison with ground truth pressure images. Grimm et al.[19] identified human orientation and pose using a prior skeleton model. Similar to our motivation and findings, they used a pressure mat to compensate for bedding occlusion and observed higher error for lighter joints (e.g. the elbow), which had a relatively low

pressure. Liu et al. [20] generated a pictorial structures model to localize body parts on a flat bed.

A few researchers have also looked at human posture classification from pressure images [19, 37, 38]. Posture classification is a different problem from 3D body pose estimation, but it can be used in a complementary way, e.g. by providing a prior on the model used for pose estimation.

2.1.2 Data-driven Human Pose Estimation

Like the pressure-image-based research, we use a human body model, but take a data-driven approach more common in vision-based work. While infrared and depth images have seen recent attention in human pose estimation [24], a large body of vision-based work uses monocular RGB cameras [25]. Our method builds upon research with monocular RGB image input; we note the following similarities and differences:

- *Single image.* Both monocular RGB-image-based work and our pressure-image-based work has a single input array.
- *Under-constrained.* For 3D human pose, both monocular RGB images and single pressure images are under-constrained.
- *Data content.* The data encoding is fundamentally different. For example, in the context of pose estimation, light intensity in an RGB image is highly disconnected from pressure intensity.
- *Dimensionality.* RGB images typically have more features and a higher resolution than pressure images.
- *Warped spatial representation.* Calibrating RGB cameras to alleviate distortion is straightforward. In contrast, it is challenging to determine the configuration of a cloth pressure sensing mat from its pressure image in the case of folds or bends.

While the differences may limit the transferability of methods across data types, we find that some data-driven methods previously used for vision modalities are applicable to pressure images. Researchers have performed 3D human pose estimation with monocular RGB images using standard machine learning algorithms such as ridge regressors [26, 27, 28]. In particular, Okada and Soatto [26] use kernel ridge regression (KRR) as well as linear ridge regression (LRR). Further, Ionescu et al. [27] compare K-Nearest Neighbors (KNN) and ridge regression on the Human3.6M dataset. We use these classical approaches to provide a baseline comparison for our proposed method.

Recently, with the advent of high quality, labeled synchronous datasets such as Human3.6M [27], many researchers have explored deep learning methods such as end-to-end training of ConvNets [29, 30, 31, 32]. Two common ConvNet approaches include direct regression to joint labels [29, 36] and regression to discretized confidence maps [30, 33, 31]. Within 3D human pose estimation research, confidence map approaches include Pavlakos et al. [31], who train a ConvNet end-to-end on a 3D confidence voxel space, and Zhou, Zhu et al. [34] and Bogo et al. [35] who fit a 3D model to 2D confidence maps. However, the high dimensional output space of confidence maps can make real-time pose estimation difficult. Li and Chan [36] used rapid direct regression to 3D Cartesian joint positions; we implement a similar architecture because real-time estimation is important to our planned application. Zhou, Sun et al. [32] take a hybrid approach, by training a ConvNet end-to-end and enforcing anthropomorphic constraints with an embedded human skeleton kinematics model with constant link lengths. We implement a method of this form for comparison and introduce a new architecture with variable skeleton link lengths to allow the model to adapt to differently sized people.

2.2 Inferring Body Position from Center-of-Mass

Here we describe how to estimate overall body position using a center-of-mass (COM) calculation from data collected by a pressure-sensing mat. This method was developed in

2015 and published in 2019 [21]. The COM model sums the pressure values to estimate the total weight on the pressure-sensing mat.

2.2.1 Method

We used a center of mass estimator from Scikit-image [39] to estimate the position of the body on the pressure-sensing mat. On average, the human pose estimation algorithm takes < 1 millisecond to run on an external machine (Intel Core i7-3770, 3.40GHz). Figure 2.1(a) shows the estimated head position for a person lying down. The COM model sets the Y-coordinate (the Y-axis is along the width of bed) of the body model's pose equal to the Y-coordinate of the estimated center of mass estimate. It positions the head of the body model in the X-direction to be 25 cm from the head of the bed and orients the body model such that the midline of the body model is parallel to the centerline of the bed.

2.2.2 Evaluation and Results

For this evaluation, there were 8 able-bodied participants who took part in the study between August and December 2015. The participants gave written informed consent. Participants' weights ranged from 52 to 95 kg and heights from 1.60 to 1.87 m.

For the experiment, we placed the bed with the mat in a motion capture room. We asked the participants to lie on the bed in a supine configuration comfortable to them, keeping their faces pointed straight up from the bed, while wearing infrared reflective marker arrays on their bodies and heads (see Figure 2.1(a)). The bed was in a flat configuration for this evaluation. We designated the projection of the center of the forehead marker array onto the plane of the bed as the ground truth head position. We selected 50 pressure distribution images from each participant while they were lying on the bed with their faces pointed straight up from the bed to form our test dataset of 400 pressure distribution images. The images for each participant were collected over a 28 second period on average. We compared the estimated head position with ground truth, but only considered error

along the width of the bed (along the Y axis), which the system uses to position the model of the person's body. The head pose estimation error along the width of the bed (i.e., the Y direction) was 5.00 cm ± 2.54 cm (mean ± std).

2.3 Inferring a 3D Human Skeleton with Deep Learning

In this section, we present a method for a inferring 3D human skeleton model in real time with a measure of confidence in each estimated joint position for a person in a configurable bed using a pressure-sensing mat. This work was previously published in IROS 2018 [9].

2.3.1 Method

Our CNNs learn a function $f(\mathcal{P}, \theta_B)$ that estimates pose parameters of a person lying in a robotic bed, given a specified bed configuration θ_B and a 2D pressure image $\mathcal{P}$ from a pressure mat.

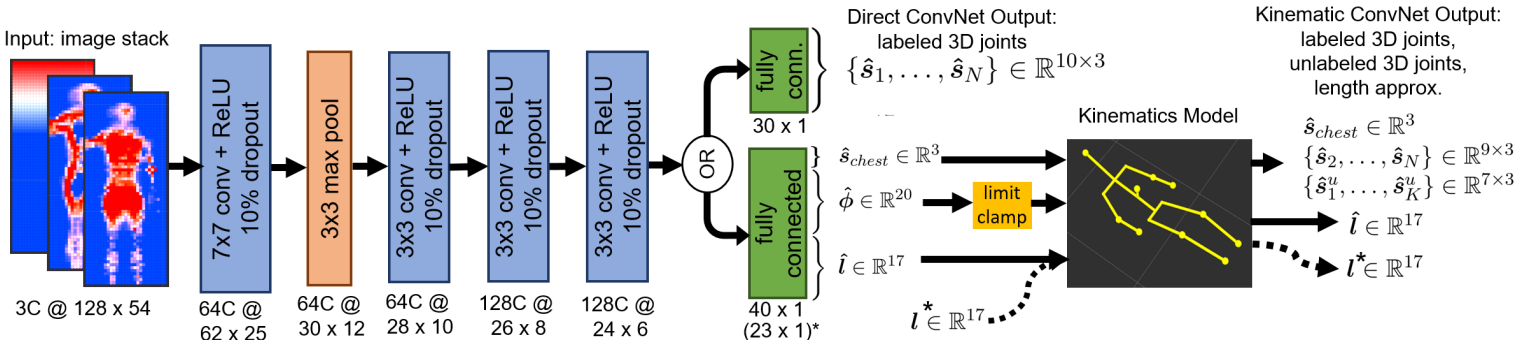

Figure 2.2: Two CNN architectures. The *Direct CNN* directly regresses to ten motion capture labeled global joint positions. The *Kinematic CNN* embeds a kinematics skeleton model into the last fully connected network layer, parameterized in the latent space by a root joint global position, joint angles, and skeleton link lengths. The *Kinematic CNN* also outputs unlabeled joint position estimates. We explore architectures with both variable link length (shown) and constant link length (dashed arrows, defined by $*$.) [9]

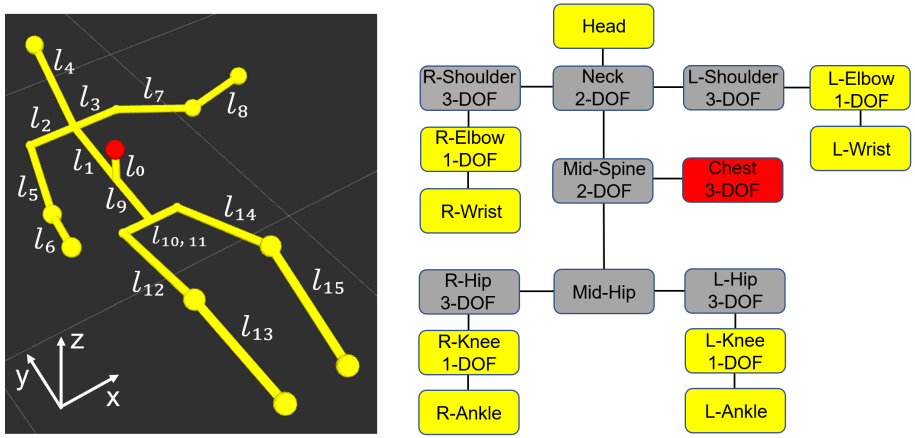

Figure 2.3: Kinematics Model Parameters. The root joint s_{chest} is specified in red. Yellow boxes represent the remaining motion capture labeled joints $\{\hat{s}_2, \ldots, \hat{s}_N\}$. Grey boxes represent unlabeled joints $\{\hat{s}^u_1, \ldots, \hat{s}^u_K\}$, where the kinematic model adjusts to an approximate fit. [9]

CNN Architecture

We explore two CNN architectures shown in Figure 2.2. Our network includes four 2D convolutional layers with 64 output channels for the first two layers and 128 channels for the last two layers. The layers mostly have 3×3 filters, with a ReLU activation and a dropout of 10% applied after each layer. We apply max pooling, and the network ends with a linear fully connected layer.

To estimate a person's joint pose, we construct an input tensor for the CNN comprised of three channels, i.e. $\{\mathcal{P}, E, B\} \in \mathbb{R}^{128\times54\times3}$. Raw data from the pressure sensing mat is recorded as a (64×27)-dimensional image, which we upsample by a factor of two, i.e. $\mathcal{P} \in \mathbb{R}^{128\times54}$. We use first order interpolation for upsampling. In addition to a pressure image, we also provide the CNN with an edge detection channel, $E \in \mathbb{R}^{128\times54}$, which is computed as a Sobel filter over both the horizontal and vertical directions of the upsampled image. Empirically, we found this edge detection input channel improved pose estimation performance. In order to estimate human pose at different configurations of the bed (e.g. sitting versus lying down), we compute a third input channel, $B \in \mathbb{R}^{128\times54}$, which depicts the bed configuration. Specifically, each element in the matrix B depicts the vertical height of the corresponding taxel on the pressure mat. When the bed frame is flat, i.e. $\theta_B = 0$, then B is simply the zero matrix.

Direct Joint Regression

The first proposed CNN architecture outputs an estimate of the motion capture labeled global 3D joint positions $\{\hat{\boldsymbol{s}}_1, \ldots, \hat{\boldsymbol{s}}_N\}$, where each $\hat{\boldsymbol{s}}_j \in \mathbb{R}^3$ represents a 3D position estimate for joint j. This direct CNN regresses directly to 3D ground truth label positions in the last fully connected layer of the network. We compute the loss from the absolute

value of Euclidean error on each joint:

$$\text{Loss}_{\text{direct}} = \sum_{j=1}^{N} ||\boldsymbol{s}_j - \hat{\boldsymbol{s}}_j|| \tag{2.1}$$

Deep Kinematic Embedding

We embed a human skeleton kinematics model into the last fully connected network layer to enforce geometric and anthropomorphic constraints. This creates an extra network layer where labeled joint estimates $\{\hat{\boldsymbol{s}}_1, \ldots, \hat{\boldsymbol{s}}_N\}$ and unlabeled joint estimates $\{\hat{\boldsymbol{s}}^u_1, \ldots, \hat{\boldsymbol{s}}^u_K\}$ are solved through forward kinematics equations depending on root joint position $\hat{\boldsymbol{s}}_1$, latent space joint angles $\hat{\boldsymbol{\phi}}$ and an estimate of skeleton link length approximations $\hat{\boldsymbol{l}}$. We use $\hat{\boldsymbol{s}}_1 = \hat{\boldsymbol{s}}_{chest}$ as the root joint. We incorporate skeleton link lengths $\boldsymbol{l}$ into the loss function by pre-computing an approximation to the ground truth. While the kinematics functions are relative to a root joint, we learn root joint global position $\boldsymbol{s}_{chest}$ to put our output in global space. We compute a weighted loss from the absolute value of Euclidean error on each joint and error on each link length:

$$\text{Loss}_{\text{kin.}} = ||\boldsymbol{s}_1 - \hat{\boldsymbol{s}}_1|| + \alpha \sum_{j=2}^{N} ||\boldsymbol{s}_j - \hat{\boldsymbol{s}}_j|| + \beta|\boldsymbol{l} - \hat{\boldsymbol{l}}| \tag{2.2}$$

where α and β are weighting factors. We compare two variants of this loss function: The *variable link length* CNN is as described, while for the *constant link length* CNN we set $\beta = 0$ and use a constant $\boldsymbol{l}^*$ input to the kinematics model. We compute $\boldsymbol{l}^*$ as the average of the approximations $\boldsymbol{l}$.

Human kinematics model. We represent the human body with a model similar to that used in other work [27, 32, 36, 40], with 17 joints to cover major links down to the wrists and ankles. We ignore minor links and joints. To train the networks that have link lengths as an output, we require ground truths for comparison. Some ground truth link lengths may

be calculated directly from the dataset's labels, for example when motion capture gives the location of both ends of the link. We approximate the link lengths for links that are under-constrained in the dataset for the skeleton model. The link lengths are an output of our network as a $\boldsymbol{l} \in \mathbb{R}^{17}$ vector. We ignore unlabeled joints in the loss function.

The mid-spine is found by a vertical offset from the chest marker to compensate for the distance between marker placement atop the chest and the modeled bending point of the spine. We do not make offset corrections with other joints; these are more challenging than the chest and the effects are less noticeable.

Angular latent space. We define 20 angular degrees of freedom consisting of 3-DOF shoulders and hips, 1-DOF elbow and knee joints, a 2-DOF spine joint, and a 2-DOF neck joint. Fig Figure 2.3 (b) shows this parameterization corresponding to labeled and unlabeled joints. For the spine of the model to better match the spine of a person seated in bed, we used two revolute joints about the x-axis. To account for head movement, we placed a neck joint at the midpoint of the shoulders with pitch and yaw rotation. We use PyTorch [41], a deep learning library with tensor algebra and automatic differentiation. We manually encoded the forward kinematics for the skeleton kinematic model. The network uses stochastic gradient descent during backpropagation to find inverse kinematics (IK) solutions.

Pressure Image Ambiguity

Raising a limb off of a bed with pressure sensors can lead to a loss of information as the sensors can only sense pressure during contact. A similar loss of information can be seen when the limbs extend off the edge of the bed. Consider the movement shown in Figure 2.4 (c) and an example of the pressure images associated with such a movement in Figure 2.5 (a). Here, the pressure images appear nearly identical while the elbow and wrist positions change substantially. To better understand what is physically causing this phenomena, we can model the arm as a double inverted pendulum, shown in Figure 2.5

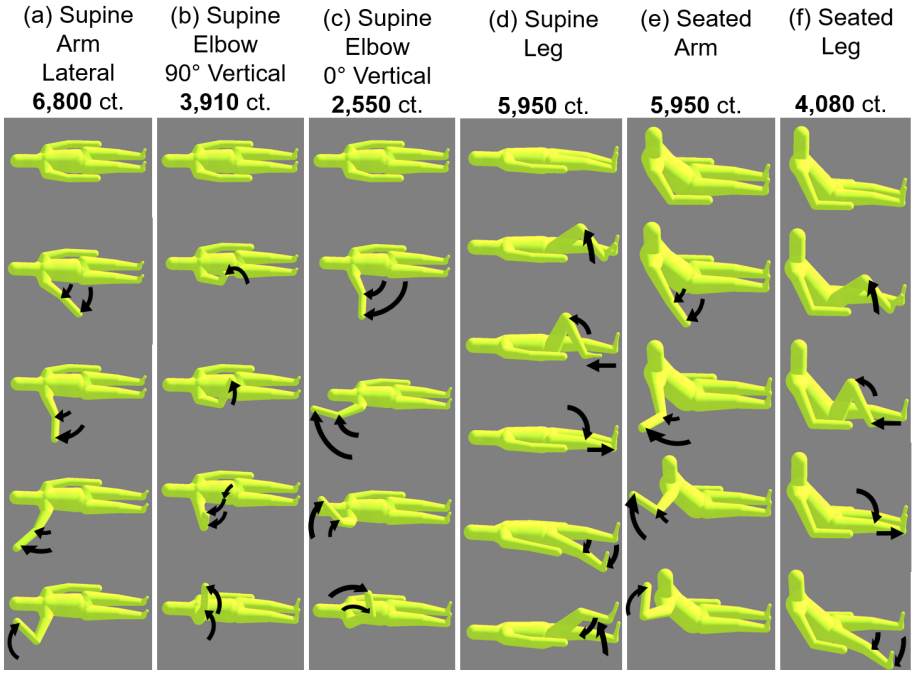

Figure 2.4: Range of paths traversed by participants during training. Equivalent paths were traversed by the left limbs. Count represents both left and right data across 17 participants.

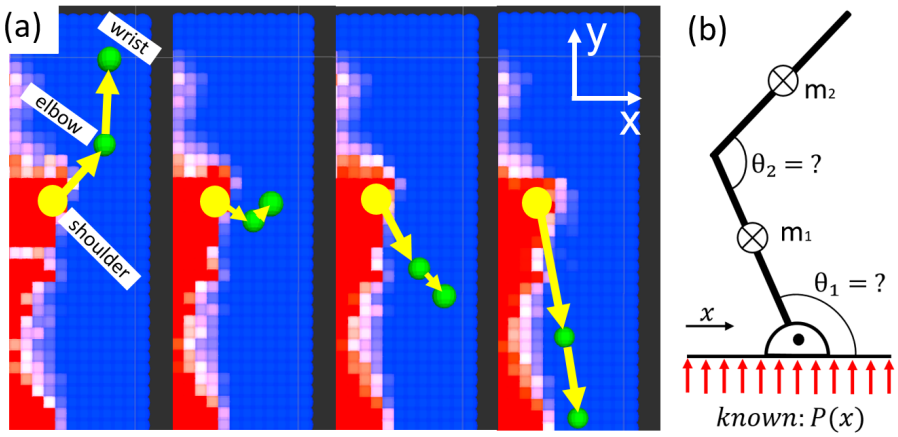

Figure 2.5: (a) Crossover of the shoulder from the *supine elbow 0 vertical* traversal, both joints envisioned as 2-DOF inverted pendulum. Green dots represent left elbow and wrist ground truth markers projected in 2D, yellow dot and arrows indicate approximate shoulder and limb positions. (b) Model of 2-DOF inverted pendulum showing static indeterminacy. Known pressure distribution $P(x)$ is insufficient to solve θ_1, θ_2.

(b). Here, the pendulum angles θ_1 and θ_2 are statically indeterminate given an underlying pressure distribution, until part of the arm touches the sensors.

Another challenge is pressure sensor resolution. Depending on the type of sensor, saturation can occur. A pressure image with higher spatial resolution, accuracy, and pressure range may result in less ambiguity. We note that a model of body shape (e.g. 3D limb capsules) might provide information that helps to resolve the ambiguity.

Uncertainty: Monte Carlo Dropout

To estimate both joint position and model uncertainty simultaneously, we apply Monte Carlo dropout from Gal and Ghahramani [23]. Monte Carlo dropout is the process of performing V forward passes through the network with dropout enabled. This results in V output vectors which may differ slightly due to the stochastic dropout of data during each forward pass. We can compute an estimated output as the average of all V outputs, corresponding to the first moment of the predictive distribution within the network. Similarly, the model's uncertainty corresponds to the second moment of the distribution, which we can compute as the variance of all V forward passes.

2.3.2 Evaluation

We recorded a motion capture labeled dataset with over 28,000 pressure images from 17 different human participants.[1] We conducted this study with approval from the Georgia Institute of Technology Institutional Review Board (IRB), and obtained informed consent from all participants. We recruited 11 male and 6 female participants aged 19-32, who ranged 1.57-1.83 m in height and 45-94 kg in weight. We fitted participants with motion capture markers at the wrists, elbows, knees, ankles, head, and chest. We asked each participant to move their limbs in 6 patterns, 4 while supine and 2 while seated, to represent some common poses in a configurable bed. The movement paths are shown in Figure 2.4.

[1] Dataset: ftp://ftp-hrl.bme.gatech.edu/pressure_mat_pose_data

We instructed participants to keep their torso static during limb movements.

We trained six data-driven models: three baseline supervised learning algorithms and the three proposed CNN architectures.[2] We designed the network using 7 participants (5M, 2F); we performed leave-one-participant-out cross validation using the remaining 10 participants (6M, 4F).

Data Augmentation

At each training epoch for the CNNs, we selected images such that each participant would be equally represented in both the training and test sets. We augmented the original dataset in the following ways to increase training data diversity:

- *Flipping.* Flipped across the longitudinal axis with probability $P = 0.5$.
- *Shifting.* Shifted by an additive factor $sh \sim \mathcal{N}(\mu = 0cm, \sigma = 2.86cm)$.
- *Scaling.* Scaled by a multiplicative factor $sc \sim \mathcal{N}(\mu = 1, \sigma = 0.06)$.
- *Noise.* Added taxel-by-taxel noise to images by an additive factor $\mathcal{N}(\mu = 0, \sigma = 1)$. Clipped the noise at min pressure (0) and at the saturated pressure value (100).

We chose these to improve the network's ability to generalize to new people and positions of the person in bed. We did not shift the seated data longitudinally or scale it because of the warped spatial representation.

Baseline Comparisons

We implemented three baseline methods to compare our CNNs against: K-nearest neighbors (KNN), linear ridge regression (LRR), and kernel ridge regression (KRR). For all baseline methods, we used histogram of oriented gradients (HOG) features [42] on $2\times$ upsampled pressure images. We applied flipping, shifting, and noise augmentation methods. We did not use scaling, as it worsened performance.

[2]Code release: https://github.com/gt-ros-pkg/hrl-assistive/tree/indigo-devel/hrl_pose_estimation

K-nearest neighbors. We implemented a K-nearest neighbors (KNN) regression baseline using Euclidean distance on the HOG features to select neighbors, as [27] did. We selected $k = 10$ for improved performance.

Ridge regression. We implemented two ridge-regression-based baselines. We trained linear ridge regression (LRR) models with a regularization factor of $\alpha = 0.7$. We also train kernel ridge regression (KRR) models with a radial basis function (RBF) kernel and $\alpha = 0.4$. We manually selected these values of α for both LRR and KRR. We also tried linear and polynomial kernels for KRR, but found the RBF kernel produced better results in our dataset.

Implementation Details of Proposed CNNs

During testing, we estimated the joint positions with $V = 25$ forward passes on the trained network with Monte Carlo dropout for each test image. For each joint, we report the mean of the forward passes as the estimated joint position. We use PyTorch and ADAM from [43] for gradient descent.

Pre-trained CNN. We created a pre-trained CNN that we use to initialize both Kinematic CNNs, with regressed and constant link length. The pre-trained network used the kinematically embedded CNN and the loss function in Equation 2.2, with $\alpha = 0.5$ and $\beta = 0.5$. This network was trained for 10 epochs on the 7 network-design participants with a learning rate of 0.00002 and weight decay of 0.0005.

Direct CNN. We trained the network for 300 epochs directly on motion capture ground truth, using the sum of Euclidean error as the loss function. We used a learning rate of 0.00002 and a weight decay of 0.0005.

Kinematic CNN, constant link length. We trained the network through the kinematically embedded CNN, used the loss function in Equation 2.2, with $\alpha = 0.5$ and $\beta = 0$. This value for β means the network would not regress to link length, leaving it constant. We initialized the network with the pre-trained CNN, but we separately initialized each

link length as the average across all images in the fold's training set for each fold of cross validation.

Kinematic CNN, regress link length. We trained the network through the kinematically embedded CNN, used the loss function in Equation 2.2, with $\alpha = 0.5$ and $\beta = 0.5$. Joint Cartesian positions and link lengths in the ground truth are represented on the same scale. We initialized with the pre-trained CNN.

Measure of Uncertainty

Here we show an example where ambiguous pressure mat data has a high model uncertainty. We compare two leg abduction movements from the *supine leg* motion, shown in the bottom two columns of Figure 2.4 (d). We sample 100 images per participant: half feature leg abduction contacting the pressure mat, and half with elevated leg abduction. For each pose, we use $V = 25$ stochastic forward passes and compute the standard deviation of the Euclidean distance from the mean for abducting joints, including knees and feet. We compare this metric between elevated and in-contact motions.

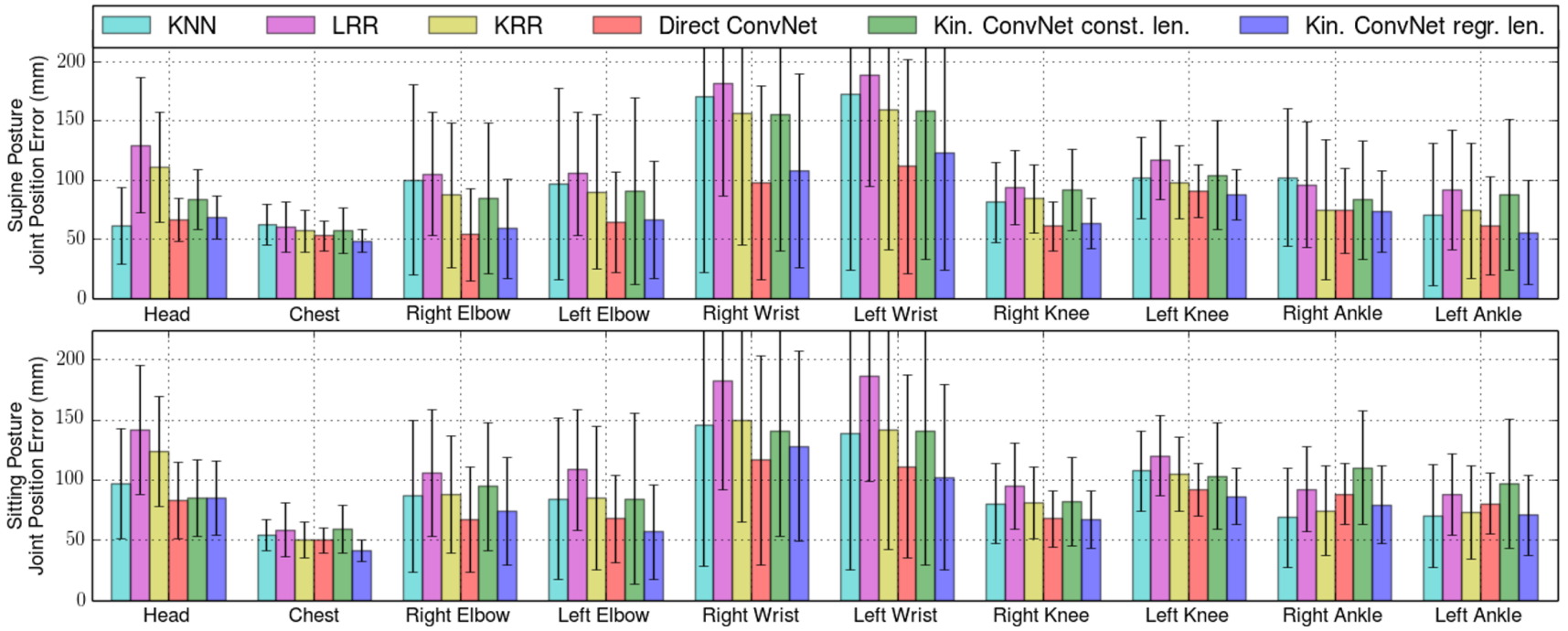

Figure 2.6: Per-joint error mean and standard deviation for leave-one-participant-out cross validation over 10 participants for a sitting and a supine posture in bed. Lower is better. Our methods outperformed baseline methods. [9]

Table 2.1: Mean Per Joint Position Error.

Method	MPJPE Supine (mm)	MPJPE Seated (mm)	MPJPE Overall (mm)
K-Nearest Neighbors	102.01	93.42	99.06
Linear Ridge Regression	117.01	114.76	116.24
Kernel Ridge Regression	99.25	97.10	98.51
Direct CNN	73.49	82.44	**76.56**
Kinematic CNN, avg. $\boldsymbol{l}$	99.74	99.70	99.72
Kinematic CNN, regr. $\boldsymbol{l}$	75.43	79.19	**76.72**

2.3.3 Results

In Table 2.1, we present the mean per joint position error (MPJPE), a metric from literature to represent overall accuracy [36, 27]. Figure 2.6 shows the per-joint position error across all trained models, separated into supine and seated postures. The error for the direct CNN and the kinematic CNN with length regression is lower than the other methods. The results of knees and legs show that more distal limbs on the kinematic chain do not necessarily result in higher error. The wrists are both distal and light, and have higher error than the other joints. Figure 2.7 shows the kinematics CNN with length regression adjusting for humans of different sizes and in different poses. Furthermore, we can perform a pose estimate with uncertainty using $V = 25$ stochastic forward passes in less than a half second.

Measure of Uncertainty

We performed a t-test to compare uncertainty in elevated leg abduction and in-contact leg abduction. We compared each knee and ankle separately. We found that the standard deviation of the Euclidean distance from the mean of $V = 25$ forward passes with Monte Carlo dropout is significantly higher for joints in the elevated position. Further, we note that variance in the latent angle parameters $\boldsymbol{\theta}$ compounds through the kinematic model, causing more distal joints in the kinematic chain to have higher uncertainty. This phenomena is

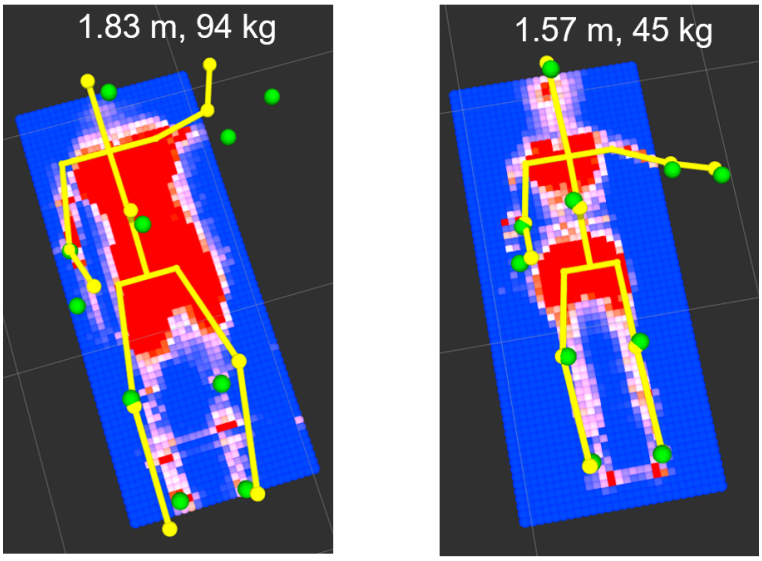

Figure 2.7: Comparison of the heaviest and tallest participant with the lightest and shortest participant. Our kinematics CNN with link length regression appears to adjust for both sizes.

further described in Figure 2.8, which shows limbs removed from the mat that have a high variance.

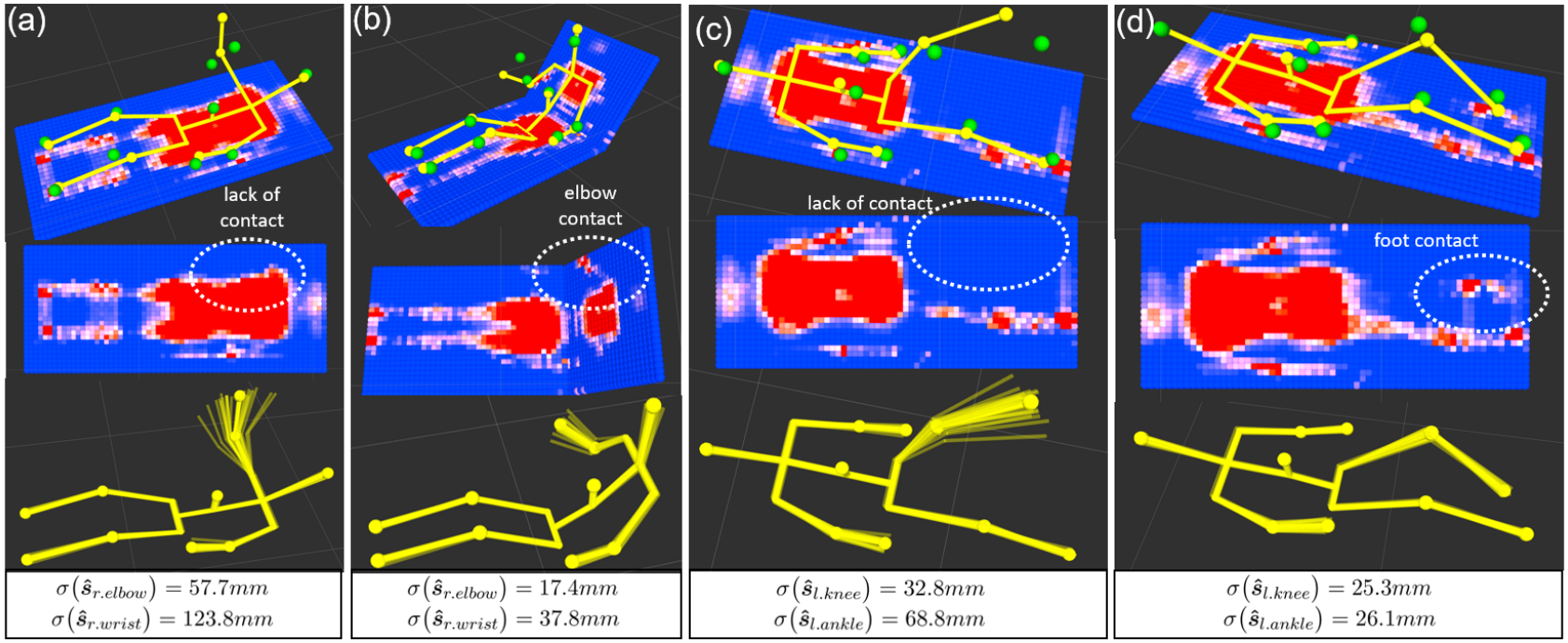

Figure 2.8: Illustration of mean and standard deviation of kinematic CNN ($\boldsymbol{l}$ regression) output. $V = 25$ forward passes with Monte Carlo dropout shown with thin translucent skeleton links. Spheres indicate joint position estimates. (a) Right arm thrust into the air. (b) Right forearm extending off the side of the pressure mat. (c) Left leg extended in the air and abducted. (d) Left knee extended in the air, left foot contacting pressure mat. Note: (c) and (d) show the same participant, others are different. [9]

2.3.4 Discussion and Limitations

Network Architecture Considerations

While the results for the direct CNN architecture were marginally better than the kinematic CNN with variable lengths, the latter has other advantages. First, there can be value in getting a skeletal model from the network, providing a more complete set of parameters including 20 angular DOFs and a total of 17 joint positions. Second, unsurprisingly, requiring that outputs from the CNN satisfy kinematic constraints means that outputs will be constrained to plausible looking body poses. Without those constraints the CNN could produce unrealistic outputs.

While limiting our skeleton model to 20 angular DOFs promotes simplicity, it has some hindrance to generalizability. For example, the mid-spine joint lacks rotational DOFs about the y- and z-axes. Adding these DOFs would allow the model to account for rolling to a different posture and laying sideways in bed.

Data Augmentation Challenges

Data augmentation cannot easily account for a person sliding up and down in a bed that is not flat. Vertical shifting augmentation for non-flat beds would not match the physical effects of shifting a person on the pressure mat. Augmentation by scaling also has problematic implications, because a much smaller or larger person may have a weight distribution that would not scale linearly at the bending point of the bed. Simulation might resolve these issues by simulating placing a weighted human model of variable shape and size placed anywhere on a simulated pressure mat, with many possibilities of bed configurations.

Dataset Considerations

The posture and range of paths traversed by participants may not be representative of other common poses, and we expect our method to have limited success in generalizing to body

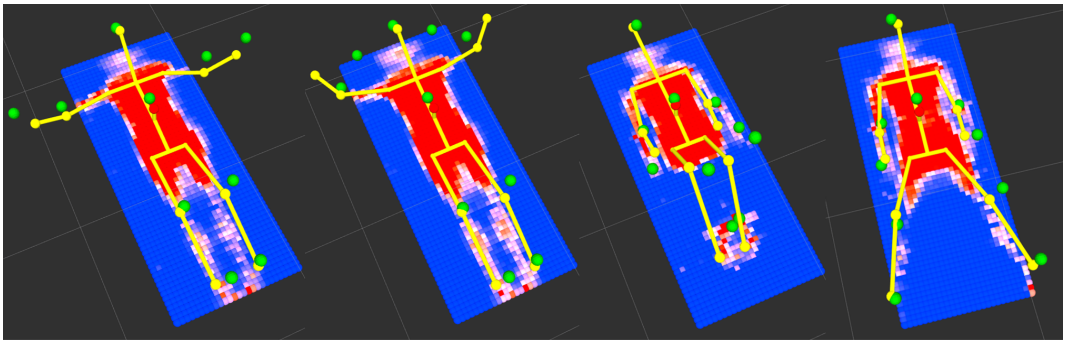

Figure 2.9: Dual arm and dual leg traversals. While these are excluded from the training set, our methods can provide a reasonable pose estimate.

poses not seen or rarely seen in the dataset. While a participant is moving their arm across a specified path, other joints remain nearly static, which over-represents poses with the arms adjacent to the chest and legs straight. We found over-represented poses to generally have a lower uncertainty. Interestingly, with our current sampling and training strategy, over-representation and under-representation is based on the percentage of images in the dataset a joint or set of joints is in a particular configuration. Additional epochs of training or directly scaling the size of the dataset does not change these effects on uncertainty of pose representation. Our method could be improved by using weighting factors or sampling strategies to compensate for this effect.

In our evaluation, some limb poses occur in separate training images, but do not occur in the same training image. For example, we recorded one participant moving both arms and both legs simultaneously. Figure 2.9 shows that our method has some ability to estimate these poses.

The skeleton model has offset error in addition to the ground truth error reported. While we attempted to compensate for the chest marker offset, the other markers were more challenging. This may have caused some inaccuracy in the link length approximations.

Removal of High Variance Joints

Figure 2.10 shows that joints with high uncertainty have a higher average error. Discarding these joint estimates can decrease the average error of the model. As an example applica-

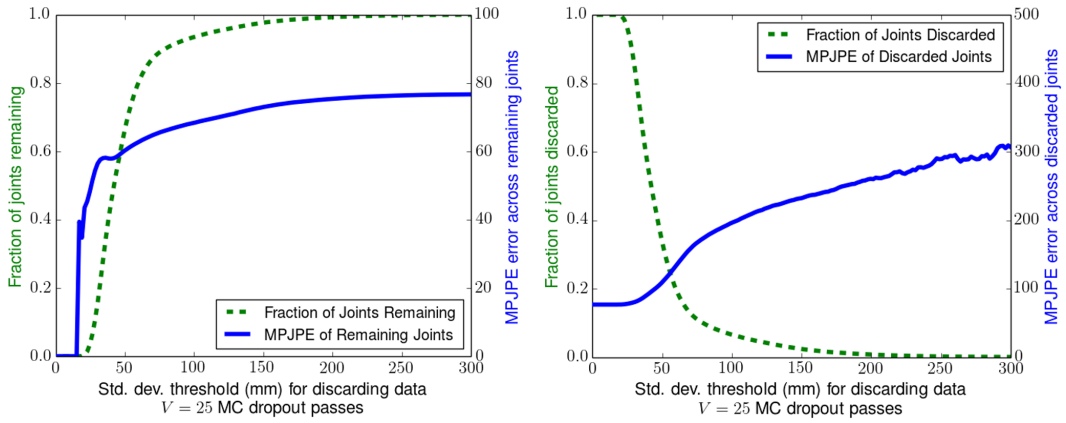

Figure 2.10: Discarding joints with a higher uncertainty can decrease error, with a tradeoff to the number of joints remaining.

tion, a robot using our method's estimated poses for task and motion planning might want to require low uncertainty before plan execution.

2.3.5 Demonstration with PR2 robot

We conducted a demonstration of how our method could inform an assistive robot trying to reach part a person's body. We conducted this study with approval from the Georgia Institute of Technology Institutional Review Board (IRB), and obtained informed consent from an participant. We recruited a single able-bodied participant who used a laptop computer running a web interface from [17] to command a PR2 robot to move its end effector to their left knee and to their left shoulder. The robot's goal was based on the estimated pose of the person's body from our ConvNet with kinematic model regressing to link lengths. For the knee position, the participant raised her knee to the configuration shown in the 2nd image of Figure 2.4 (f), and the participant was in the seated posture for both tasks. Using our 3D pose estimation method, the robot was able to autonomously reach near both locations. Figure 2.11 shows the robot reaching a shoulder goal while the participant is occluded by bedding and an over-bed table.

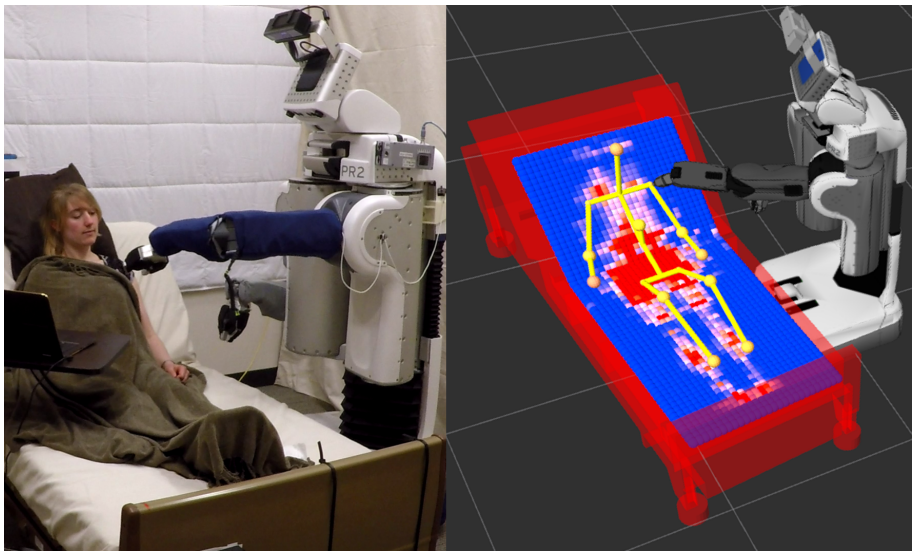

Figure 2.11: Demonstration of how an assistive robot could use our 3D human pose estimation method with a skeleton human representation. A PR2 robot uses our method's body pose estimation to reach to the person's shoulder.

2.3.6 Conclusion

In this work, we have shown that a pressure sensing mat can be used to estimate the 3D pose of a human in different postures of a configurable bed. We explored two CNN architectures and found that both outperformed data-driven baseline algorithms. Our kinematically embedded CNN with link length regression provided a more complete representation of a 17-joint skeleton model, adhered to anthropomorphic constraints, and was able to adjust to participants of varying anatomy. We provided an example where joints on limbs raised from the pressure mat had a higher uncertainty than those in contact.

CHAPTER 3

SIMULATING BODIES AT REST TO GENERATE DATA FOR DEEP LEARNING

Humans spend a large part of their lives resting. While resting, humans select poses that can be sustained with little physical exertion. Our primary insight is that human bodies at rest can be modeled sufficiently well to generate synthetic data for machine learning. The lack of physical exertion and absence of motion makes this class of human activities amenable to relatively simple biomechanical models similar to the ragdoll models used in video games [44].

We apply this insight to the problem of using a pressure image to estimate the 3D human pose and shape of a person resting in bed. This capability would be useful for a variety of healthcare applications such as bed sore management [37], tomographic patient imaging [19], sleep studies [45], patient monitoring [46], and assistive robotics [9]. To this end, we present the PressurePose dataset, a large-scale synthetic dataset consisting of 3D human body poses and shapes with pressure images (Figure 3.1, top). We also present PressureNet, a deep learning model that estimates 3D human body pose and shape from a low-resolution pressure image (Figure 3.1, bottom).

Prior work on the problem of human pose estimation from pressure images [45, 9, 19, 18, 20] has primarily used real data that is challenging to collect. Our PressurePose dataset has an unprecedented diversity of body shapes, joint angles, and postures with more thorough and precise annotations than previous datasets (Table 3.1). While recent prior work has estimated 3D human pose from pressure images, [45, 9], to the best of our knowledge PressureNet is the first system to also estimate 3D body shape.

Our synthetic data generation method first generates diverse samples from an 85 dimensional human pose and shape space. After rejecting samples based on self-collisions and Cartesian constraints, our method uses each remaining sample to define the initial con-

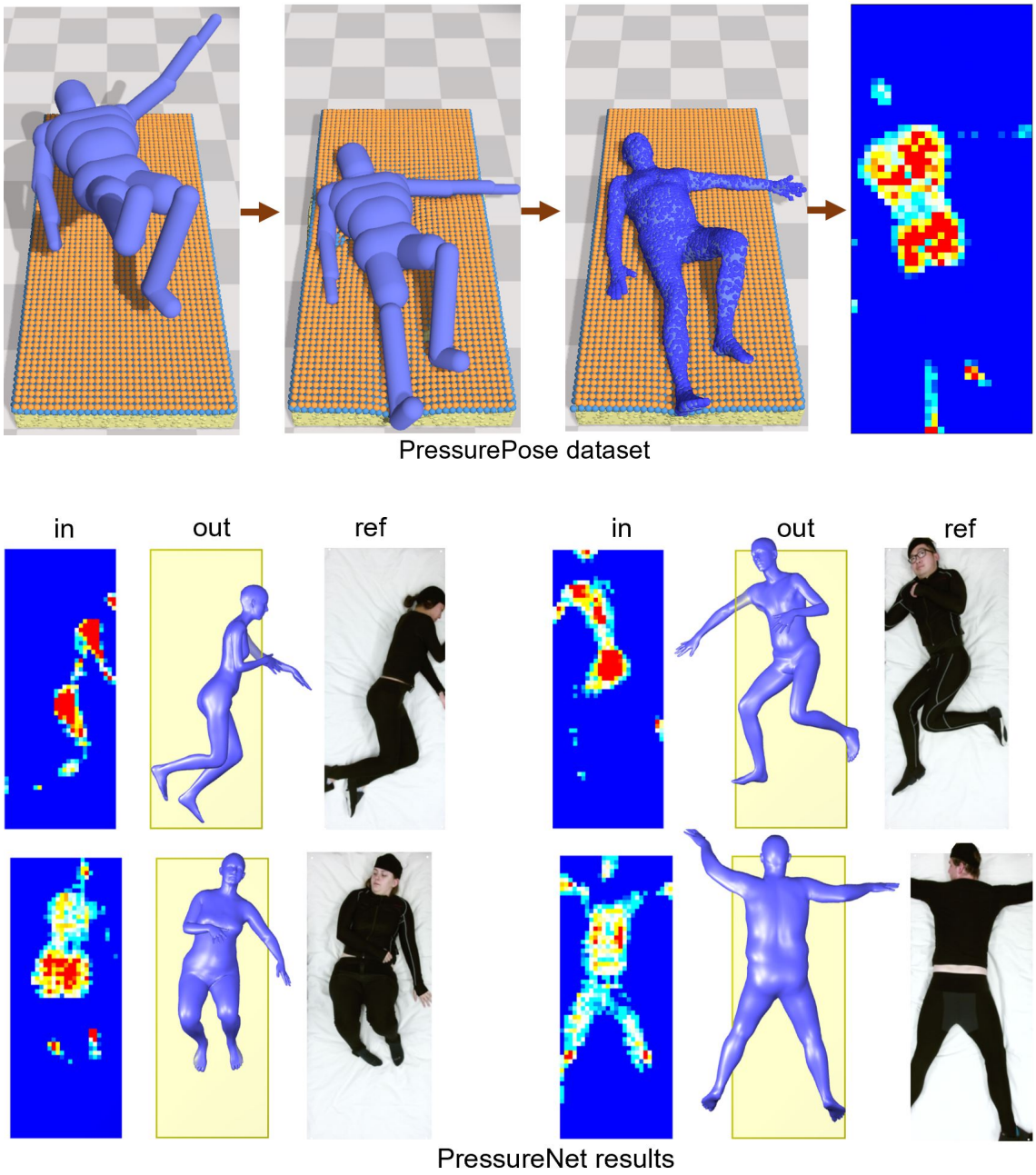

Figure 3.1: Top: The PressurePose dataset has 206K 3D human poses and shapes with pressure images generated by physics simulations that drop articulated rigid body models and soft body models on a soft body model of a bed and pressure sensing mat. Bottom: PressureNet is a deep learning model trained on synthetic data that performs well on real data: pressure image input with gender (in), 3D human mesh output (out), RGB image for reference (ref).

ditions for a series of two physics simulations. The first finds a body pose that is at rest on a simulated bed. Given this pose, the second physics simulation generates a synthetic pressure image.

Our method uses SMPL [10] to generate human mesh models and a capsulized approximation of SMPL [35] to generate articulated rigid-body models. The first physics simulation drops a capsulized articulated rigid-body model with low-stiffness, damped joints on a soft-body model of a bed and pressure-sensing mat. Once the articulated body has settled into a statically stable configuration, our method converts the settled capsulized model into a particle-based soft body without articulation. This soft body model represents the shape of the body, which is important for pressure image synthesis. The second physics simulation drops this soft-body model from a small height onto the soft-body bed and sensor model. Once settled, the simulated sensor produces a pressure image, which is stored along with the settled body parameters.

Our deep learning model, PressureNet, uses a series of two networks modules. Each consists of a convolutional neural network (CNN) based on [9], a kinematic embedding model from [47] that produces a SMPL mesh [10], and a pressure map reconstruction (PMR) network. The PMR network serves as a model of pressure image generation. It is a novel component that encourages consistency between the mesh model and the pressure image input. Without it, we found that our deep learning models would often make mistakes that neglected the role of contact between the body and the bed, such as placing the heel of a foot at a location some distance away from an isolated high pressure region.

When given a mesh model of the human body, the PMR network outputs an approximate pressure image that the network can more directly compare to the pressure image input. These approximate pressure images are used in the loss function and as input to a second residual network trained after the first network to correct these types of errors and generally improve performance.

In our evaluation, we used a commercially available pressure sensing mat (BodiTrak

BT-3510 [48]) placed under the fitted sheet of an Invacare Homecare Bed [49]. This sensing method has potential advantages to line-of-sight sensors due to occlusion of the body from bedding and other sources, such as medical equipment. However, the mat we used provides low-resolution pressure images (64×27) with limited sensitivity and dynamic range that make the estimation problem more challenging.

We only trained PressureNet using synthetic data, yet it performed well in our evaluation with real data from 20 people, including successfully estimating poses that have not previously been reported in the literature, such as supine poses with hands behind the head. To improve the performance of the model with real data, we used custom calibration objects and an optimization procedure to match the physics simulation to the real world prior to synthesizing the training data. We also created a noise model in order to apply noise to the synthetic pressure images when training PressureNet.

3.1 Related Works

There is long history of human pose estimation from camera images [28, 20, 26, 25, 54] and the more recent use of CNNs [30, 29]. The field has been moving rapidly with the estimation of 3D skeleton models [31, 32], and human pose and shape estimation as a 3D mesh [35, 47, 55] using human body models such as SCAPE [56] and SMPL [10]. These latter methods enforce physical constraints to provide kinematically feasible pose estimates, some via optimization [35] and others using learned embedded kinematics models [9, 47, 32]. Our approach builds on these works both directly through the use of available neural networks (e.g, SMPL embedding) and conceptually.

While pressure image formation differs from conventional cameras, the images are visually interpretable and methods developed in the vision community are well suited to pressure imagery [57, 47, 29]. PressureNet's model of pressure image generation relates to recent work on physical contact between people and objects [58, 59, 60]. It also relates to approaches that fine-tune estimates based on spatial differences between maps at distinct

Table 3.1: Comparison of Literature: Human Pose in Bed.

work	data: (R)eal, (S)ynth	modality: (P)ressure, (D)epth (T)thermal, IRS - infrared selective	3D: (Y)es, (N)o	human representation: (S)keleson, (M)esh	postures	# joints	# identities	# images
Harada, 2001 [18]	R	P	Y	M	SP+	18	1	?
Grimm, 2012 [19]	R	D, P	N	S	SP, L, P	10	16	1.1 K
J.J. Liu, 2014 [20]	R	P	N	S	SP, L	8*	12	1.4 K
Ours, 2015 [21]	R	P	N	S	SP	1	1	-
Achilles, 2016 [50]	R	D	Y	S	I/O, SP, L	14	10	180 K
Chen, 2018 [46]	R	RGB	N	S	SP, UNK	7	3	13 K
Ours, 2018 [9]	R	P	Y	S	SP, ST	14	17	28 K
Casas, 2019 [45]	R	P	Y	S	SP+, L+, ST	14	6	60
S. Liu, 2019 [51]	R	IRS	N	S	SP+, L+	14	2	419
S. Liu, 2019 [52]	R	T	N	S	SP+, L+	14	109	14 K
Ours, 2020 [11]	S/ R	P	Y	M	SP+, L+, P+	24	200K/ 20	200K/ 1K
S. Liu, 2020 [14]	R	RGB, D, T, P	N	S	SP+, L+	14	109	14 K
Yin, 2020 [53]	R	RGB, D, T, P	Y	M	SP+, L+	24	109	14 K
Ours, 2021	S/ R	D	Y	M	SP+, L+	24	97K/ 109	97K/ 14K

posture key: SP - supine. L - lateral. P - prone. I/O - getting in/out of bed. ST - sitting. + indicates a continuum between postures. * indicates limbs.

stages of estimation [61, 57, 62, 29].

3.1.1 Human Pose at Rest.

Human pose estimation has tended to focus on active poses. Poses in bed have attracted special attention due to their relevance to healthcare. Table 3.1 provides an overview of work on the estimation of human pose for people in bed. These efforts have used a variety of sensors including RGB cameras [46], infrared lighting and cameras for darkened rooms [51], depth cameras to estimate pose underneath a blanket profile [50], thermal cameras to see through a blanket [52], and pressure mats underneath a person [45, 9, 63, 19, 18, 20].

Researchers have investigated posture classification for people in bed [37, 19, 38]. There has been a lack of consensus on body poses to consider, as illustrated by Table Table 3.1. Some works focus on task-related poses, such as eating [50], and stretching [45]. Poses can increase ambiguity for particular modalities, such as lack of contact on a pressure mat (e.g. knee in the air) [9, 64] or overlapping body parts facing a thermal camera [52].

Large datasets would be valuable for deep learning and evaluation. While some bed pose work has used thousands of images they have either had few participants [46] or poses highly concentrated in some areas due to many frames being captured when there is little motion [50, 45, 9]. An exception is recent work by Liu et al. [52], which has 109 participants.

3.1.2 Generating Data in Simulation.

Approaches for generating synthetic data that model humans in the context of deep learning include physics-based simulators such as DART [65] and PyBullet [66] and position-based dynamics simulators such as PhysX [67] and FleX [8]. Some have used these tools to simulate deformable objects like cloth [68, 67]. For vision, creating synthetic depth images is relatively straightforward (e.g. [50]) while RGB image synthesis relies on more complex graphics approaches [69, 70, 71].

Some past works have simulated pressure sensors. One approach is to model the array as a deformable volume that penetrates the sensed object, where force is a function of distance penetrated [72]. Others model pressure sensing skin as a mass-spring-damper array [73, 74]; the former considers separate layers for the skin and the sensor, a key attribute of pressure arrays covering deformable objects.

3.2 PressurePose Synthetic Dataset Generation

Our data generation process consists of three main stages, as depicted in Figure 3.2: sampling of the body pose and shape; a physics simulation to find a body pose at rest; and a physics simulation to generate a pressure image. We use two simulation tools, FleX (subsection 3.2.1) for simulating soft body dynamics, and DART (subsection 3.2.2) for articulated rigid body dynamics.

Sample initial pose and shape. We sample initial pose (i.e. joint angles) and body shape parameters from the SMPL human model [10]. The pose consists of 69 joint angles, $\mathbf{\Theta} \in \mathbb{R}^{69}$, which we sample from a uniform distribution, $\mathcal{U}$, bounded by joint angle limits defined for the hips, knees, shoulders, and elbows in [75, 76, 77]. We initialize the human body above the bed with a uniformly sampled roll $\theta_{r,1}$, yaw $\theta_{r,3}$, and 2D translation $\{s_{r,1}, s_{r,2}\}$ across the surface of the bed. The pitch $\theta_{r,2}$ is set to 0 and the distance normal to the bed $s_{r,3}$ is based on the position of the lowest initial joint position. This determines the global transform, $\{\boldsymbol{\theta}_r,\ \boldsymbol{s}_r\} \in \mathbb{R}^6$. The shape of a SMPL human is determined from a set of 10 PCA parameters, $\boldsymbol{\beta} \in \mathbb{R}^{10}$, which we also sample uniformly, bounded by $[-3, 3]$ following [78]. We use rejection sampling in three ways for generating initial poses: to more uniformly distribute overall pose about the Cartesian space (rather than the uniformly sampled joint space), to create a variety of data partitions representing specific common postures (e.g. hands behind the head), and to reject pose samples when there are self-collisions. See Appendix A.1. This step outputs pose and shape parameters $\{\boldsymbol{\beta}, \mathbf{\Theta}_C, \boldsymbol{\theta}_r, \boldsymbol{s}_r\}$, where $\mathbf{\Theta}_C$ is a set of joint angles conditioned on $\boldsymbol{\beta}$ that has passed these criteria.

Physics Simulation #1: Resting Pose. We use FleX [8] to simulate a human model resting on a soft bed, which includes a mattress and a synthetic pressure mat on the surface of the mattress (Figure 3.2). The human is modelled as an articulated rigid body system made with capsule primitives, which is a dynamic variant of the SMPL model. Once the

simulation nears static equilibrium, we record the resting pose $\{\widetilde{\boldsymbol{\Theta}}_C, \tilde{\boldsymbol{\theta}}_r, \tilde{\boldsymbol{s}}_r\}$.

FleX is a position-based dynamics simulator with a unified particle representation that can efficiently simulate rigid and deformable objects. However, FleX does not currently provide a way for particles to influence the motions of rigid capsules. To overcome this limitation, we use DART [65] to model the rigid body dynamics of the capsulized human model. We combine FleX and DART through the following loop: 1) DART moves the capsulized articulated rigid body based on applied forces and moments. 2) FleX moves the soft body particles in response to the motions of the rigid body. 3) We compute new forces and moments to apply in DART based on the state of the FleX particles and the capsulized articulated rigid body. 4) Repeat. We call the combination of the two simulators DartFleX and subsection 3.2.2 provides further details.

Physics Simulation #2: Pressure Image. The settled, capsulized body is insufficient for producing a realistic pressure image: it approximates the human shape too roughly. Instead, we create a weighted, particlized, soft human body in FleX (Figure 3.2 and Figure 3.3) from the SMPL [10] mesh using body shape and resting pose $\{\boldsymbol{\beta}, \widetilde{\boldsymbol{\Theta}}_C, \tilde{\boldsymbol{\theta}}_r\}$. We initialize the particlized human with 2D translation over the surface of the mattress $\{\tilde{s}_{r,1}, \tilde{s}_{r,2}\} \in \tilde{\boldsymbol{s}}_r$. We set $s_{r,3}$, the position normal to gravity, so the body is just above the surface of the bed. We then start the simulation, resting the particlized body on the soft bed, and record the pressure image $\mathcal{P}$ once the simulation has neared static equilibrium. We note that this particlized representation has no kinematics and cannot be used to adjust a body to a resting configuration; thus our use of two separate dynamic simulations.

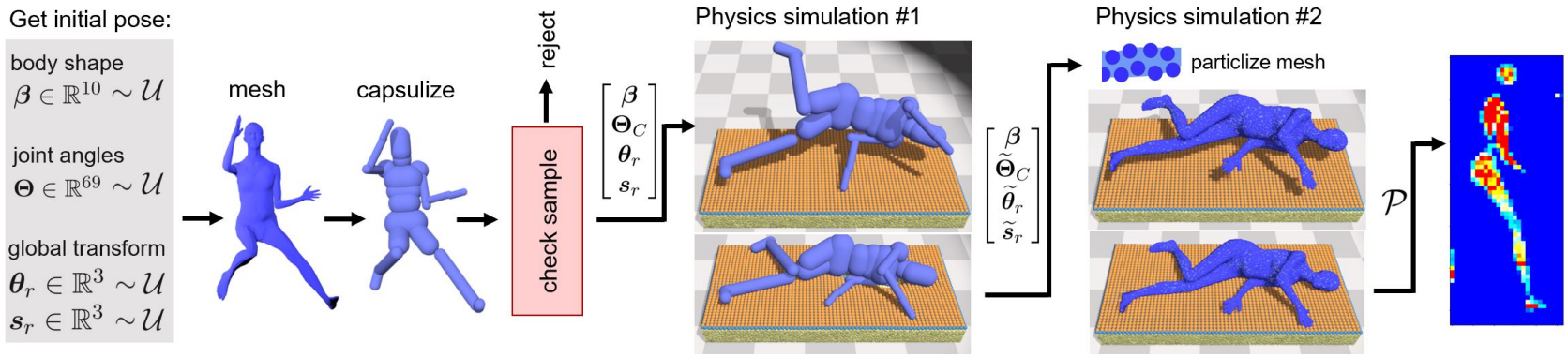

Figure 3.2: We generate the initial pose from scratch, using random sampling of the body shape, joint angles, and global transform on the bed. We use rejection sampling to distribute the poses and remove self-collisions. Then, we rest a dynamic capsulized human model onto a soft bed using DartFleX, a fusion of DART and FleX simulators, to get an updated resting pose. Because this model is a rather rough approximation of human shape, we then use FleX to particlize a finer body representation to get the pressure image.) [11]

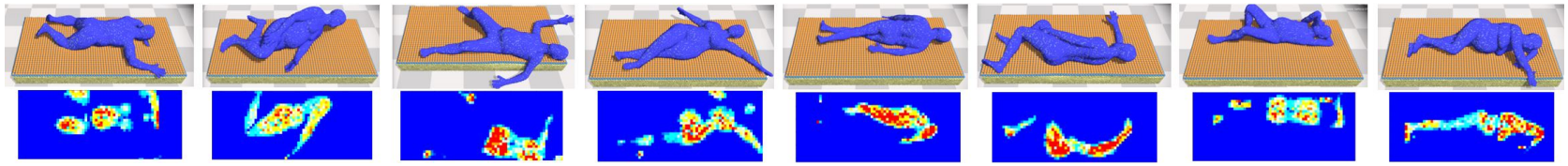

Figure 3.3: Physics simulation #2 output: PressurePose synthetic dataset examples. [11]

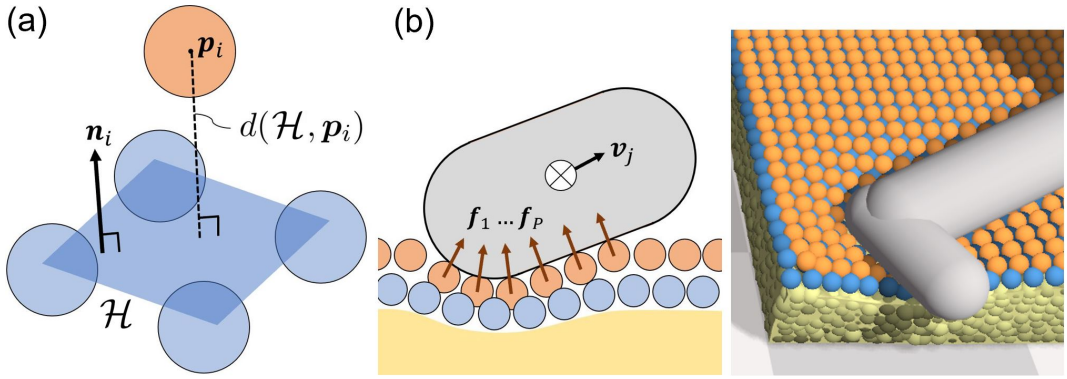

Figure 3.4: (a) Synthetic pressure mat structure. Pressure is a function of the penetration of the top layer array particle into the four underlying particles. (b) DartFleX collision between a capsulized limb and the simulated bed and pressure-sensing mat. [11]

3.2.1 Softbody Simulation with FleX

We simulate the sensing array by connecting FleX particles in a way that mimics real pressure sensing fabric, and model the mattress with a soft FleX object.

Soft Mattress and Pressure Sensing Mat. Here we describe the soft mattress and pressure sensing array within the FleX environment, as shown in Figure 3.4 and further described in Appendix A.3. The mattress is created in a common twin XL size with clusters of particles defined by their spacing, D_M, radius, R_M, stiffness, K_M, and particle mass, m_M, parameters. We then create a simulated pressure sensing mat on top of the mattress that is used to both generate pressure images and to help the human model reach a resting pose by computing the force vectors applied to the various segments of the human body. The mat consists of two layers of staggered quad FleX cloth meshes in a square pyramid structure, where each layer is defined by its stretching, K_σ, bending, K_B, and shear, K_τ, stiffnesses, which are spring constraints on particles that hold the mat together. A compression stiffness, K_C, determines the bond strength between the two layers, and its mass is defined by m_L.

We model force applied to the mat as a function of the particle penetration vector $\boldsymbol{x}_i$ based on the pyramid structure in Figure 3.4 (a). Force increases as the i^{th} particle on the

top layer, $\boldsymbol{p}_i$, moves closer to the four particles underneath.

$$\boldsymbol{x}_i = \big(d_0 - d(\mathcal{H}, \boldsymbol{p}_i)\big)\boldsymbol{n}_i \tag{3.1}$$

where d is the distance between particle $\boldsymbol{p}_i$ and an approximate underlying plane $\mathcal{H}$, d_0 is the initial distance at rest prior to contact, and $\boldsymbol{n}_i$ is the normal vector of the approximate underlying plane.

Sensor Model. The BodiTrak pressure-sensing mat has an array of pressure-sensing taxels (tactile pixels). The four particles at the base of the pyramid structure in Figure 3.4 (a) model the 1" square geometry of a single pressure-sensing taxel. We model the pressure output, u_i, of a single taxel, i, using a quadratic function of the magnitude of the penetration vector $\boldsymbol{x}_i$.

$$u_i = \big(C_2|\boldsymbol{x}_i|^2 + C_1|\boldsymbol{x}_i| + C_0\big) \tag{3.2}$$

where C_2, C_1, and C_0 are constants optimized to fit calibration data, as described in subsection 3.2.3.

3.2.2 DartFleX: Resting a Dynamic Ragdoll Body

The purpose of DartFleX is to allow rigid body kinematic chains to interact with soft objects by coupling the rigid body dynamics solver in DART to the unified particle solver in FleX as shown in Figure 3.4 (b).

Dynamic rigid body chain. Our rigid human body model relies on a capsulized approximation to the SMPL model, following [35]. To use this model in a dynamics context, we calculate the per-capsule mass and mass inertia matrix for each capsule in the kinematic chain. We weight the capsules based on volume ratios from a person with average body shape $\bar{\boldsymbol{\beta}} = \mathbf{0}$, average body mass, and mass percentage distributions between body parts as defined by Tozeren [79]. For joint stiffnesses $\boldsymbol{k}_\theta \in \mathbb{R}^{69}$, we tune parameters to achieve the

low stiffness characteristics of a ragdoll model that can settle into a resting pose on a bed due to gravity. We set torso and head stiffness high so that they are effectively immobile, and joint damping $\boldsymbol{b}_\theta = 15\boldsymbol{k}_\theta$ to reduce jitter.

DartFleX Physics. We initialize the same capsulized model in both DART and FleX. We apply gravity in DART, and take a step in the DART simulator. We get a set of updated dynamic capsule positions and orientations, and move the static geometry counterparts in FleX accordingly. In order to transfer force data from FleX to DART, we first check if any top layer pressure mat particles are in contact. Each particle i in contact has a penetration vector $\boldsymbol{x}_i(t)$ (see equation Equation 3.1) at time t, which we convert to normal force vector $\boldsymbol{f}_{N,i} \in \mathbb{R}^3$ using a mass-spring-damper model [80]:

$$\boldsymbol{f}_{N,i} = k\boldsymbol{x}_i(t) + b\dot{\boldsymbol{x}}_i(t), \tag{3.3}$$

where k is a spring constant, b is a damping constant, and $\boldsymbol{f}_{N,i} \perp \mathcal{H}$. We then assign each force to its nearest corresponding capsule j. Given the velocity, $\boldsymbol{v}_j$, of capsule j and a friction coefficient, μ_k, we compute the frictional force $\boldsymbol{f}_{T,i}$ for the i^{th} particle in contact:

$$\boldsymbol{f}_{T,i} = -\mu_k|\boldsymbol{f}_{N,i}|\frac{\boldsymbol{v}_j - proj_{\boldsymbol{f}_{N,i}}\boldsymbol{v}_j}{|\boldsymbol{v}_j - proj_{\boldsymbol{f}_{N,i}}\boldsymbol{v}_j|} \tag{3.4}$$

where $proj$ is an operator that projects $\boldsymbol{v}_j$ orthogonally onto a straight line parallel to $\boldsymbol{f}_{N,i}$. In our simulation, we set $b = 4k$ and $u_k = 0.5$, and we find k through a calibration sequence described in subsection 3.2.3. We can then compute the total particle force, $\boldsymbol{f}_i$:

$$\boldsymbol{f}_i = \boldsymbol{f}_{N,i} + \boldsymbol{f}_{T,i} \tag{3.5}$$

We then compute a resultant force $\boldsymbol{F}_j$ in FleX for the j^{th} body capsule, based on the

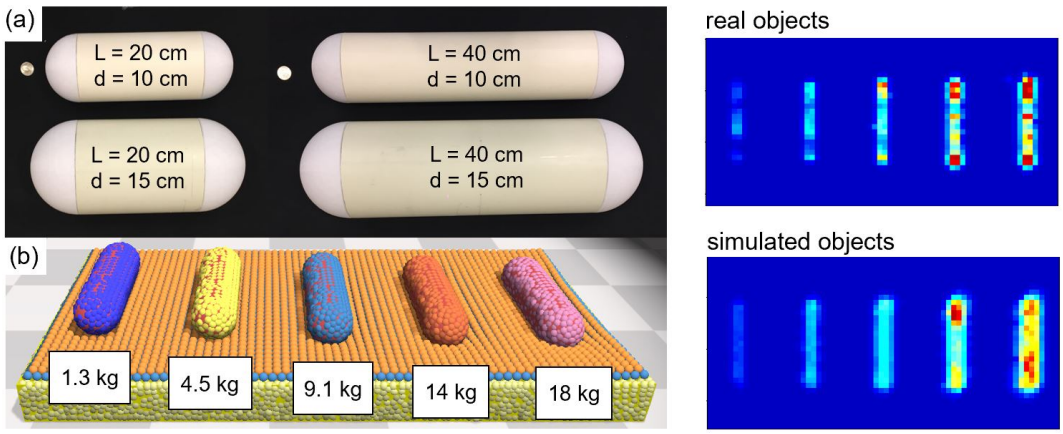

Figure 3.5: (a) Rigid calibration capsules with quarters (U.S. coins) shown for size. (b) Simulated capsules. (right) Real and simulated pressure images prior to calibration. [11]

sum of forces from P particles in contact with the capsule plus gravity, $\boldsymbol{F}_g$:

$$\boldsymbol{F}_j = \sum_{i=1}^{P} \boldsymbol{f}_i + \boldsymbol{F}_g \tag{3.6}$$

Moment $\boldsymbol{M}_j$ is computed on each capsule j from P particles in contact, where $\boldsymbol{r}_i$ is the moment arm between a particle and the capsule center of mass:

$$\boldsymbol{M}_j = \sum_{i=1}^{P} \boldsymbol{r}_i \times \boldsymbol{f}_i \tag{3.7}$$

The resultant forces and moments are applied in DART, a step is taken with the forces and gravity applied to each body part, and the DartFleX cycle repeats. We continue until the capsulized model settles and then record resting pose $\widetilde{\boldsymbol{\Theta}}_C$, root position $\tilde{\boldsymbol{s}}_r$, and root orientation $\tilde{\boldsymbol{\theta}}_r$.

3.2.3 Calibration

We calibrated our simulation using the rigid capsule shapes in Figure 3.6 (a). We placed varying weights on them on the real pressure-sensing mat and recorded data, and then created matching shapes in simulation. We first calibrated the FleX environment using the particlized capsules shown in Figure 3.6 (b) using the covariance matrix adaptation

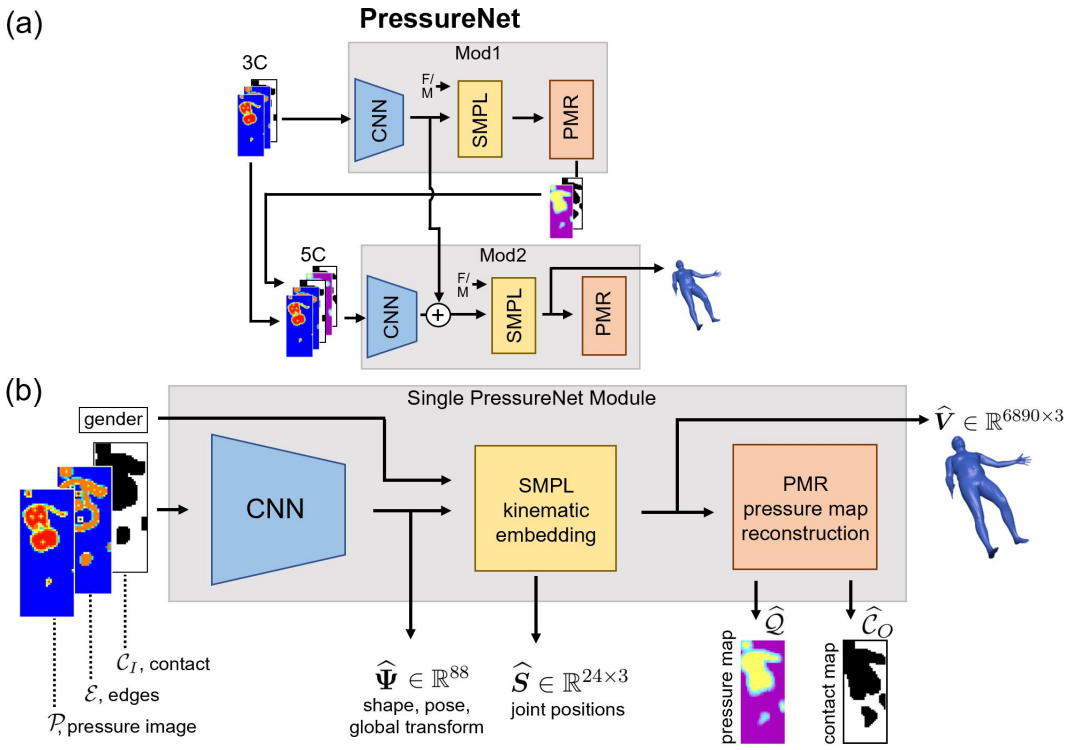

Figure 3.6: (a) PressureNet: We combine two network modules ("Mod1" and "Mod2") in series. Mod1 learns a coarse estimate and Mod2 fine-tunes, by learning a residual that takes as input the two maps reconstructed by Mod1 combined with the input to Mod1. (b) Detailed description of a single PressureNet module showing the novel PMR network that reconstructs pressure and contact maps. [11]

evolution strategy (CMA-ES) [81] to match synthetic pressure images and real pressures images of the calibrated objects by optimizing D_M, R_M, K_M, m_M, d_0, K_σ, K_B, K_τ, K_C, m_L, C_2, C_1, and C_0.

We also measure how much the real capsules sink within the mattress. We use these measurements to calibrate the mass-spring-damper model in equation Equation 3.3. We fit the simulated capsule displacement to the real capsule displacement to solve for the spring constant k and then set $b = 4k$ and $\mu_k = 0.5$. See Appendix A.4 and Appendix A.5 for details.

3.3 PressureNet: Pose Estimation with Residual Learning

Given a pressure image of a person resting in bed and a gender, PressureNet produces a posed 3D body model. PressureNet (Figure 3.6 (a)) consists of two network modules trained in sequence ("Mod1" and "Mod2"). Each takes as input a tensor consisting of three channels: pressure, edges, and contact $\{\mathcal{P}, \mathcal{E}, \mathcal{C}_I\} \in \mathbb{R}^{128\times54\times3}$, which are shown in Figure 3.6 (b), as well as a binary flag for gender. $\mathcal{P}$ is the pressure image from a pressure sensing mat, $\mathcal{E}$ results from an edge detection channel consisting of a sobel filter applied to $\mathcal{P}$, and $\mathcal{C}_I$ is a binary contact map calculated from all non-zero elements of $\mathcal{P}$. Given this input, each module outputs both an SMPL mesh body and two reconstructed maps produced by the PMR network, $\{\widehat{\mathcal{Q}}, \widehat{\mathcal{C}}_O\}$, that estimate the pressure image that would be generated by the mesh body. Mod2 has the same structure as Mod1, except that it takes in two additional channels: the maps produced by PMR in Mod1 $\{\widehat{\mathcal{Q}}_1, \widehat{\mathcal{C}}_{O,1}\}$. We train PressureNet by training Mod1 to produce a coarse estimate, freezing the learned model weights, and then training Mod2 to fine-tune the estimate.

CNN. The first component of each network module is a CNN with an architecture similar to the one proposed by Clever et al [9]. Notably, we tripled the number of channels in each convolutional layer. See Appendix B.1 for details. During training, only the weights of the CNNs are allowed to change. All other parts of the networks are held

constant. The convolutional model outputs the estimated body shape, pose, and global transform, $\widehat{\boldsymbol{\Psi}} = \{\widehat{\boldsymbol{\Theta}}, \hat{\boldsymbol{\beta}}, \hat{\boldsymbol{s}}_r, \hat{\boldsymbol{x}}_r, \hat{\boldsymbol{y}}_r\}$, with the estimated joint angles $\widehat{\boldsymbol{\Theta}} \in \mathbb{R}^{69}$, body shape parameters $\hat{\boldsymbol{\beta}} \in \mathbb{R}^{10}$, global translation of the root joint with respect to the bed $\hat{\boldsymbol{s}}_r \in \mathbb{R}^3$, and parameters $\hat{\boldsymbol{x}}_r, \hat{\boldsymbol{y}}_r$ which define a continuous orientation for the root joint of the body with $\{x_u, x_v, x_w\} \in \boldsymbol{x}_r$, $\{y_u, y_v, y_w\} \in \boldsymbol{y}_r$ for 3 DOF, i.e. $\theta_{r,u} = atan2(y_u, x_u)$ and $\{\theta_{r,u}, \theta_{r,v}, \theta_{r,w}\} \in \boldsymbol{\theta}_r \in \mathbb{R}^3$.

SMPL kinematic embedding. $\widehat{\boldsymbol{\Psi}}$ feeds into a kinematic embedding layer (see Figure 3.6), which uses the SMPL differentiable kinematics model from [47] to learn to estimate the shape, pose, and global transform. This embedding outputs joint positions for the human body, $\widehat{\boldsymbol{S}}$, and a SMPL mesh consisting of vertices $\widehat{\boldsymbol{V}}$; and relies on forward kinematics to ensure body proportions and joint angles match real humans.

PMR. The final component of each module, the PMR network, reconstructs two maps based on the relationship between the SMPL mesh $\widehat{\boldsymbol{V}}$ and the surface of the bed. The reconstructed pressure map ($\widehat{\mathcal{Q}}$) corresponds with the input pressure image, $\mathcal{P}$, and is computed for each pressure image taxel based on the distance that the human mesh sinks into the bed. The reconstructed contact map ($\widehat{\mathcal{C}_O}$) corresponds with the input contact map, $\widehat{\mathcal{C}_I}$, and is a binary contact map of $\widehat{\mathcal{Q}}$. See Appendix B for details.

Loss function. We train Mod1 in PressureNet with the following loss function, given $N = 24$ Cartesian joint positions and $S = 10$ body parameters:

$$\mathbb{L}_1 = \frac{1}{N\sigma_s} \sum_{j=1}^{N} \left|\left|\boldsymbol{s}_j - \hat{\boldsymbol{s}}_{j,1}\right|\right|_2 + \frac{1}{S\sigma_\beta} \left|\left|\boldsymbol{\beta} - \hat{\boldsymbol{\beta}}_1\right|\right|_1 \tag{3.8}$$

where $\boldsymbol{s}_j \in \boldsymbol{S}$ represents the 3D position of a single joint, and σ_s and σ_β are standard deviations computed over the whole dataset to normalize the terms. We have 24 Cartesian joint positions $\boldsymbol{S} \in \mathbb{R}^{24\times3}$ because $\{\boldsymbol{\Theta}, \boldsymbol{\theta}_r\} \in \mathbb{R}^{72}$. We compute a loss on joint error rather than vertex error because the vertices are highly concentrated in some areas like the face and hands for aesthetic reasons, rather than for representing overall pose. Moreover, training the first network module ("Mod1") with reconstruction of 24 joint positions rather

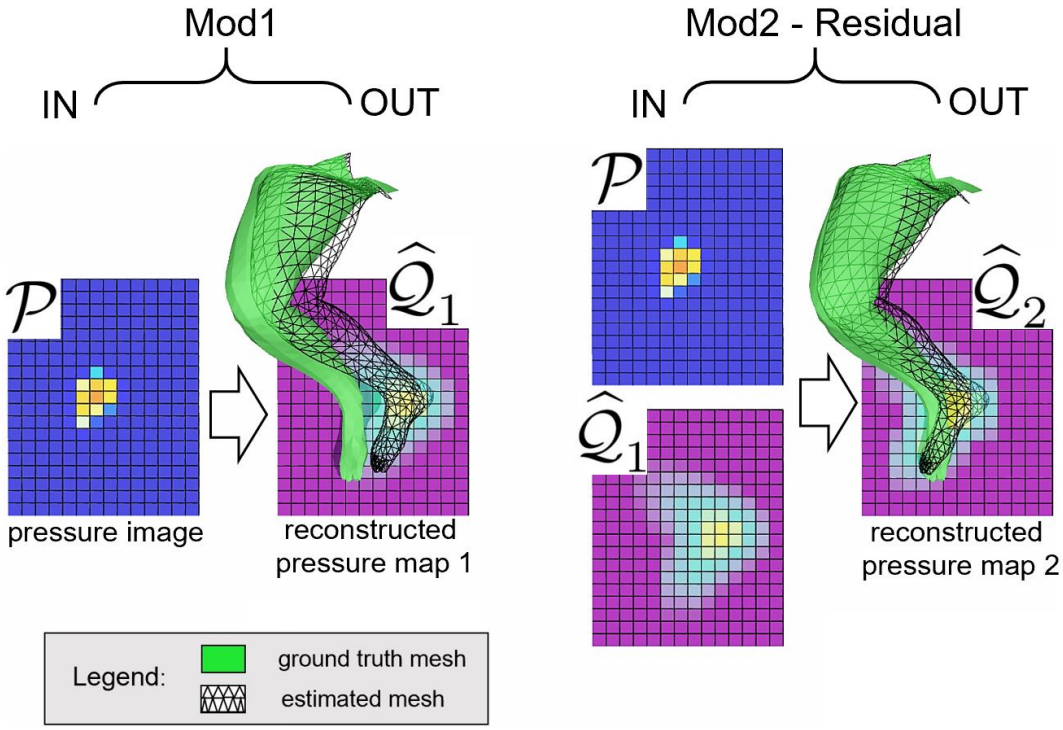

Figure 3.7: PressureNet deep learning in action, showing an example from our synthetic test set. The first network module ("Mod1") outputs an initial coarse pose estimate (right leg shown) and a reconstructed pressure map $\widehat{\mathcal{Q}}_1$. The second network module ("Mod2") corrects the estimated black mesh by a small angle difference based on the spatial residual between $\mathcal{P}$ and $\widehat{\mathcal{Q}}_1$. [11]

than a full set of vertices is much faster.

In our evaluations (section 3.5), sequentially training two separate network modules improved model performance and the resulting human mesh and pose predictions. For a pressure array of T taxels, we compute a loss for Mod2 by adding the error between the reconstructed pressure maps and the ground truth maps from simulation.

$$\mathbb{L}_2 = \mathbb{L}_1 + \frac{1}{T\sigma_{\mathcal{Q}}}\left|\left|\mathcal{Q} - \widehat{\mathcal{Q}}_2\right|\right|_2^2 + \frac{1}{T\sigma_{\mathcal{C}_O}}\left|\left|\mathcal{C}_O - \widehat{\mathcal{C}}_{O,2}\right|\right|_1 \tag{3.9}$$

where $\mathbb{L}_1$ uses Mod2 estimates (i.e. $\widehat{\boldsymbol{S}}_2$, $\hat{\boldsymbol{\beta}}_2$), $\mathcal{Q}$ and $\mathcal{C}_O$ are ground truth maps precomputed by setting $\widehat{\boldsymbol{\Psi}} = \boldsymbol{\Psi}$, and $\sigma_{\mathcal{Q}}$, $\sigma_{\mathcal{C}_O}$ are computed over the dataset.

The purpose of the second network module ("Mod2") is to fine-tune an initial estimate

from Mod1 using both reconstructed pressure maps as input and a loss function with spatial map awareness. Figure 3.7 shows a real example of how Mod2 corrects the initial mesh estimate from Mod1 using PMR. Note the spatial difference in the input images for Mod2, where the reconstructed map of the foot pressure in $\widehat{\mathcal{Q}}_1$ is shifted further right than the information on pressure image $\mathcal{P}$.

3.4 Evaluation

To evaluate our methods, we trained our CNN on synthetic data and tested it on both synthetic and real data. We generated 206K synthetic bodies at rest with corresponding pressure images (184K train / 22K test), which we partitioned to represent both a uniformly sampled space and common resting postures. By posture, we mean common recognized categories of overall body pose, such as sitting, prone, and supine. We tested 4 network types and 2 training data sets of different size.

3.4.1 PressurePose Data Partitions

We used the rejection sampling method described in section 3.2 and Appendix A.1 to generate initial poses and create dataset partitions. Our main partition, the *general* partition, consists of 116K image and label pairs. In it, we evenly distributed limb poses about the Cartesian space and randomly sampled over body roll and yaw. This partition includes supine, left/right lateral and prone postures, as well as postures in between, and has the greatest diversity of poses. We also created a *general supine* partition (58K) featuring only supine postures and evenly distributed limb poses. Finally, we generated smaller partitions representing other common postures: *hands behind the head* (5K), *prone with hands up* (9K), *supine crossed legs* (9K), and *supine straight limbs* (9K). See Appendix A.7 for details.

3.4.2 PressureNet Evaluation

We build PressureNet in PyTorch [41]. We normalized all input data by a per-image sum of taxels. We blurred synthetic and real images with a Gaussian of $\sigma = 0.5$. We trained for 100 epochs on Mod1 with loss function $\mathbb{L}_1$. Then, we pre-computed the reconstruction maps $\{\widehat{\mathcal{Q}}_1, \widehat{\mathcal{C}}_{O,1}\}$ from Mod1 for input to Mod2, and trained Mod2 for 100 epochs using loss function $\mathbb{L}_2$. For both Mod1 and Mod2, we used a learning rate of 0.00002 and a weight decay of 0.0005, which are the same used in [9]. We used the Adam optimizer for gradient descent [43]. Training Mod2 for 100 epochs using 184K images took 3 days on a Nvidia Tesla K80 GPU. Training Mod2 took 8 days due to increased computation from PMR.

We investigated 5 variants of PressureNet, which are all trained entirely with synthetic data in order to compare the effect of (1) ablating PMR, (2) adding noise to the synthetic training data, (3) ablating the contact and edge input ($\mathcal{C}_I$ and $\mathcal{E}$), and (4) reducing the training data size. Ablating PMR consists of removing the 2 reconstructed maps from the input to Mod2 and using $\mathbb{L}_1$ for training both Mod1 and Mod2. We compared the effect of adding noise to the training data to account for real-world variation, such as sensor noise. Our noise model includes per-pixel white noise, additive noise, multiplicative noise, and blur variation, all with $\sigma = 0.2$. We compared networks trained on 46K vs. 184K images.

3.4.3 Human Participant Study

We mounted a Microsoft Kinect 2 $1.6m$ above our Invacare Homecare bed to capture RGB images and point clouds synchronized with our pressure image data. See details in Appendix A.6. We recruited 20 (10F/10M) human participants with approval from an Institutional Review Board. We conducted the study in two parts to capture (1) participant-selected poses and (2) prescribed poses from the synthetic test set. We began by capturing five participant-selected poses. For the first pose, participants were instructed to get into the bed and get comfortable. For the remaining four, participants were told to get comfort-

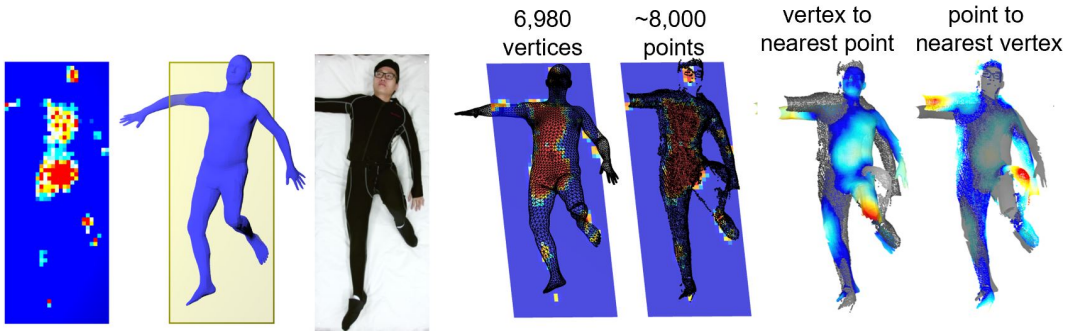

Figure 3.8: 3D error analysis between a human mesh (6,980 vertices) and a point cloud (∼8,000 downsampled points). [11]

able in supine, right lateral, left lateral, and prone postures. Next, for the prescribed poses, we displayed a pose rendering on a monitor, and instructed the participants to get into the pose shown. We captured 48 prescribed poses per participant, which were sampled without replacement from the synthetic testing set: 24 general partition poses, 8 supine-only poses, and 4 from each of the remaining partitions.

3.4.4 Data Analysis

We performed an error analysis as depicted in Figure 3.8. For this analysis, we compute the closest point cloud point to each mesh vertex, and the closest mesh vertex to each point cloud point. We introduce 3DVPE (3D vertex-point-error), which is the average of these numbers. We downsample the point cloud to a resolution of 1cm so the number of points is roughly equal to the number of mesh vertices. We clip the mesh vertices and the point cloud at the edges of the pressure mat. The point cloud only contains information from the top surface of the body facing the camera, so we clip the mesh vertices that do not have at least one adjacent face facing the camera. Finally, we normalize the mesh by vertex density: while the density of the point cloud is uniform from downsampling, the mesh vertices are highly concentrated in some areas like the face. We normalize each per-vertex error by the average of its adjacent face surface areas.

We evaluated PressureNet on the synthetic test set and compared the results to the real test set. We clip the estimated and ground truth mesh vertices and normalize per-vertex

error in the same way as the real data. Additionally, we evaluated per-joint error (24 joints) using mean-per-joint-position error (MPJPE), and per-vertex error (6890 vertices) using vertex-to-vertex error (v2v) for the synthetic data. We evaluated the network's ability to infer posture using the participant-selected pose dataset by manually labeling the inferred posture (4 labels: supine, prone, left/right lateral). We also compared to a baseline human, *BL*, where we put a body of mean shape in a supine position in the center of the bed and compare it to all ground truth poses. We positioned the legs and arms to be straight and aligned with the length of the body.

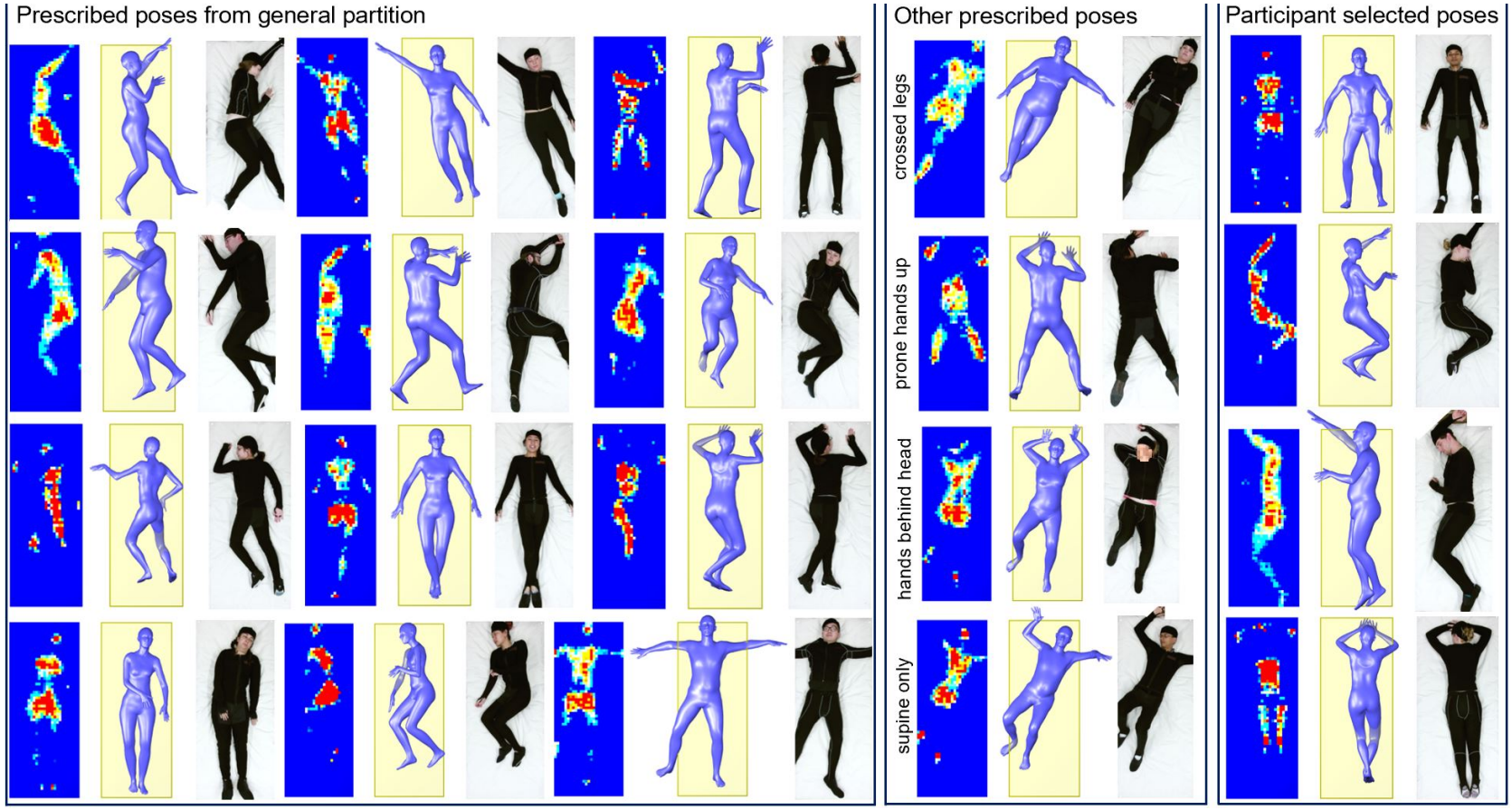

Figure 3.9: PressureNet results on real data with the best performing network (trained with 184K samples). [11]

Table 3.2: Results comparing testing data and network type.

Network Description	Training data ct.	12K synth MPJPE (cm)	12K synth v2v (cm)	12K synth 3DVPE (cm)	1K real 3DVPE (cm)	99 real 3DVPE (cm)
Best	184K	**11.18**	**13.50**	**3.94**	**4.99**	**4.76**
Noise σ ablated	184K	**11.18**	13.52	3.97	5.05	4.79
Input $\mathcal{E}, \mathcal{C}_I$ ablated	184K	11.39	13.73	4.03	5.07	4.85
Best - small data	46K	12.65	15.28	4.35	5.17	4.89
PMR ablated	184K	12.28	14.65	4.38	5.33	4.94
Baseline - mean pose	-	33.30	38.70	8.43	6.65	5.22

Table 3.3: Results - participant selected poses, real data. *See Figure 3.10-top left.

posture partition	test ct.	3DVPE (cm)	posture match
no instruction	19	3.93	100%
supine	20	4.02	100%
right lateral	20	5.45	100%
left lateral	20	5.37	100%
prone	20	4.96	95%*

3.5 Results and Discussion

Overall, we found that using more synthetic data resulted in higher performance in all tests, as shown in Table 3.2. As expected, ablating the PMR network and ablating noise reduced performance. Ablating contact and edge inputs also reduced performance. We expect that comparable performance could be achieved without them, possibly by changing the details of the CNN. Figure 3.9 shows results from the best performing network with 184K training images, noise, and the PMR network.

We compared the error on a set of 99 participant selected poses, shown in Table 3.3, using the best performing PressureNet. Results show a higher error for lateral postures where the body center of mass is further from the mat and limbs are more often resting on other limbs or the body rather than the mat. Table 3.4 shows the results of PressureNet

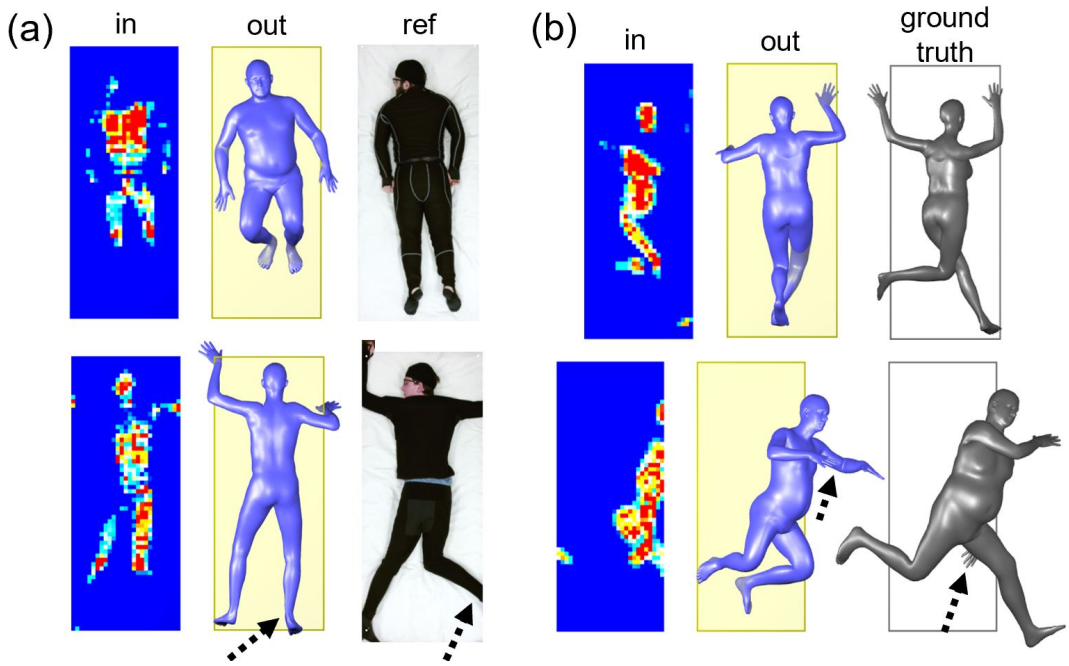

Figure 3.10: Some failure cases. (a) Real data. (b) Testing on synthetic *training* data. [11]

evaluated between different postures of prescribed resting poses from participants in bed, and a per-gender comparison. Figure 3.10 shows four failure cases.

3.6 Conclusion

With our physics-based simulation pipeline, we generated a dataset, PressurePose, consisting of 200K synthetic pressure images with an unprecedented variety of body shapes and poses. Then, we trained a deep learning model, PressureNet, entirely on synthetic data. With our best performing model, we achieve an average pose estimation error of $< 5\ cm$, as measured by 3DVPE, resulting in accurate 3D pose and body shape estimation with real people on a pressure sensing bed.

Table 3.4: Partitioned results for prescribed poses with the best network for each real and synthetic.

pose partition	test ct. real	test ct. synth	3DVPE real (cm)	3DVPE synth (cm)
supine straight limbs	76	1000	3.71	2.68
supine general	159	2000	4.51	3.40
supine crossed legs	78	1000	4.49	3.41
prone hands up	80	1000	5.12	4.24
general, roll $\sim \mathcal{U}[-\pi, \pi]$	479	6000	5.39	4.30
supine hands behind head	80	1000	5.09	4.40
gender partition				
F	480	6000	4.88	3.85
M	472	6000	5.10	4.04

CHAPTER 4

INFERRING HUMAN POSE AND CONTACT PRESSURE FROM A DEPTH IMAGE

Pressure injuries are an extremely common and longstanding ailment for bedridden individuals, yet technologies to reliably detect them, such as pressure mats, remain expensive and rare in practice. Using a camera for this task could enable the widespread proliferation of pressure injury detection systems, reducing the 2.5 million of such injuries which occur in the U.S. every year [82]. However, sensing pressure from camera imagery faces substantial challenges: not only is the contact interface visually occluded by the human body itself, but the person is frequently covered with blankets, which makes it challenging to even sense where the person is in bed.

We propose a method, BodyPressure, that can accurately infer body pose and contact pressure from a single image captured by a depth camera. With these constituents, BodyPressure can localize regions of high pressure underneath a person in bed by projecting the pressure onto a human model. We represent body pose with the Skinned Multi-Person Linear (SMPL) human model [10], which consists of a 3D volumetric mesh parameterized by 72 joint angles and 10 body shape coefficients. Our approach takes as input a depth image taken by a camera looking down at a person underneath blankets, and infers a SMPL human model, as well as the contact pressure between the body and the mattress.

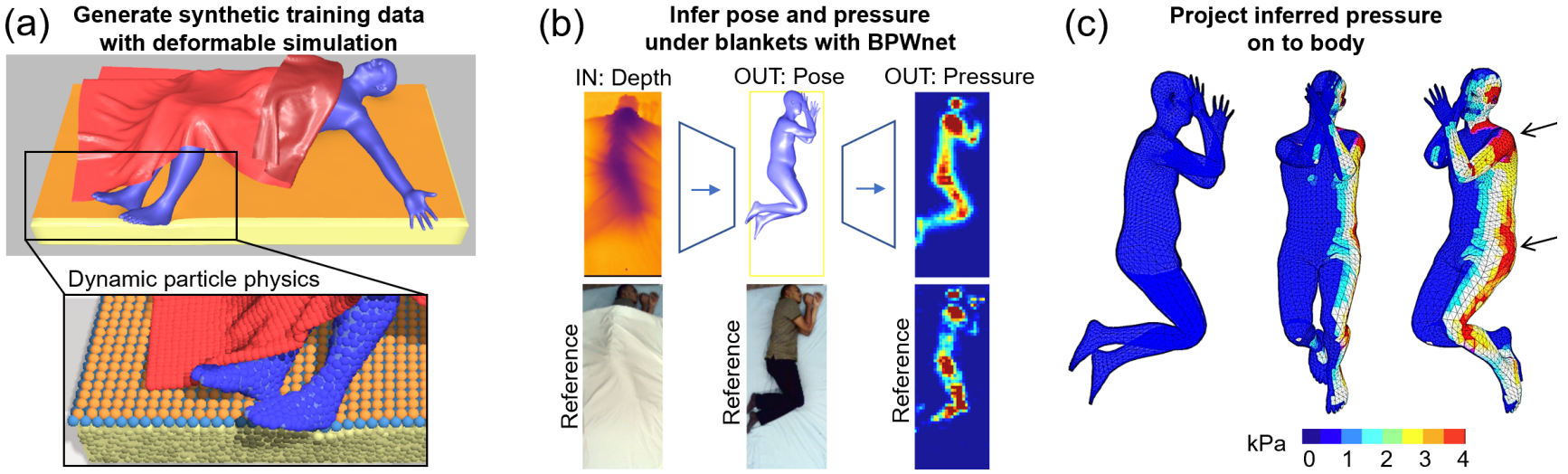

Figure 4.1: We use fast physics simulations to generate BodyPressureSD, a large synthetic human resting pose dataset, and then train a deep model, BodyPressureWnet, to infer pose and pressure from a depth image. (a) Our simulation method rests bodies on a soft bed and covers them with blankets. We then render depth images from the perspective of an overhead camera, and generate pressure images from a pressure mat underneath the person. (b) Using an augmented dataset with a mix of this synthetic data combined with real data captured by a depth camera, we learn a mapping from depth and gender to pose and contact pressure. (c) This enables a camera to infer the pressure distribution of the person and potentially detect pressure injuries in the real world [12].

To address the challenge of heavy occlusion when inferring human pose at rest, prior work has required multiple modalities as input, including RGB, depth, thermal, and pressure imagery [14, 53]. Our deep network, BodyPressureWnet, outperforms this prior work while using only depth images. This is made possible by adding a large synthetic dataset to a smaller real dataset. The depth modality has a number of benefits: it can be generated easily in simulation by rendering object geometry; deep learning models trained with synthetic depth images transfer well to the real world [83]; and depth imagery preserves patient privacy better than RGB imagery [84].

To create such a large collection of training samples, we employ fast physics simulations to generate BodyPressureSD, a synthetic dataset consisting of depth images, body poses, and pressure sensing mat data. We extend our previous work from chapter 3 [11], which simulates human bodies resting on a soft mattress with a pressure-sensing mat. We extend this method by generating blankets to cover the resting bodies, producing a set of meshes representing the bed, the person in bed, and the blanket covering the person (Figure 4.1(a)). We then render these meshes to generate images similar to those captured by a real depth camera. This process can quickly generate data for training data-hungry deep models. We find that the synthetic depth images body poses, and pressure images resemble their real counterparts with enough fidelity to greatly boost performance of the deep model.

We combine the synthetic data with real data from the Simultaneously-collected Lying Pose (SLP) dataset [14] to achieve a 9:1 mixed training data ratio. SLP offers well matched attributes for training and testing our methods. It includes co-registered depth and pressure images capturing diverse human resting poses in bed with varying scenarios of blanket occlusion, as well as 2D human pose annotations. To improve the ground truth pose labels, we present an optimization method to fit 3D SMPL bodies to the SLP dataset and publicly release the fits. We use these to supervise deep model learning, to test the model, and also to initialize the aforementioned synthetic data generator. Initializing the simulator with the appropriate pose distribution is important for generating realistic resting poses.

Inferring the pressure distribution underneath a person from a depth image involves three main steps: (1) inferring human pose as a mesh model, (2) inferring the contact pressure on the top surface of the bed, and (3) projecting the pressure from the bed surface onto the body mesh. We introduce a network architecture to address this challenge, which uses a convolutional neural network (CNN) encoder to estimate parameters for the SMPL model. Our network uses depth and pressure map reconstruction components to improve accuracy. However, in contrast to other works which use black-box (learned neural network) reconstruction models [85, 61, 86, 87], we use a white-box (analytic) reconstruction technique. This reconstruction is computationally efficient, differentiable, and has no learned parameters. We refer to the reconstructions as *maps* instead of *images* to distinguish them from sensor data.

Section 4.1 covers related literature. Section 4.2 presents a method for annotating real data to create SLP-3Dfits. These annotations are used for initializing the synthetic data generator, for training the deep model, and for testing it. Section 4.3 presents our physics simulation pipeline for generating the synthetic training data. Section 4.4 presents our deep network architectures, which are trained using real and synthetic data. Section 4.5 explains how we evaluate our method, followed by the results and discussion in section 4.6.

4.1 Related Work

Sensor-based pressure injury monitoring. Commercial pressure mapping systems are among the most common methods of monitoring pressure injury risk, and have been used to more effectively reposition patients and reduce high pressure areas [88]. Researchers have made progress to improve monitoring through automatic bodypart localization [37, 20, 89] and posture detection [90]. An alternative to this is wearable pressure sensors that can adhere to at-risk areas [91]. The cost of these devices can deter widespread use, so others have studied how inertial measurement units (IMUs) can be used [92], among other methods [88]. Yet, peak pressure localization remains a challenge. The sacrum and heels

have been noted as the most common areas, but also occur on the hips, elbows, ischium, shoulders, spinous process, ankles, toes, and head [93, 94].

Humans at rest. While many works in computer vision model humans in *active* poses such as pedestrians crossing the street [95], *resting* belongs to a different class of human activity. Resting is characterized by a low degree of physical exertion, substantial contact with surrounding surfaces such as a bed or chair, and the fact that people spend an overwhelming portion of life resting. With the ability to learn complex mappings between images and labels using CNNs, researchers have inferred human resting pose from diverse human configurations, postures, and sensing modalities [50, 45, 9, 46, 51, 52, 14].

Maintaining awareness of scene constraints and dynamics can enable more physically plausible models of humans at rest. Chao *et al.* [96] used reinforcement learning to teach dynamic agents how to sit on a chair in a virtual environment. Hassan *et al.* [59] used optimization to infer pose in a way that the human model is consistent with its surroundings, i.e. not floating above a chair or sunk into it unrealistically. Our previous work modeled humans in bed using ragdoll physics [11], and another work synthesized human poses in arbitrary environments with objects that could be contacted or rested upon [97].

Simulating human environments. Approaches for generating synthetic data that model humans in the context of deep learning use physics simulators such as DART [65] and PyBullet [98, 66] and position-based dynamics simulators such as PhysX [67] and FleX [8]. In a recent work, we combined DART and FleX to rest kinematic human bodies on a soft mattress [11], and randomized the human pose and body shape to increase variability. Others have explored cloth with physics simulations [68, 99, 67, 100], which could be used to create a diverse set of blanket configurations and profiles on a person resting in bed. Human environment models can also benefit from understanding object motion landscapes [101] to synthesize better interactions [102].

Simulating pressure and depth images. We refer the reader to our previous work on simulating pressure imagery [11], which includes a pressure image generation method

that we use. For vision, RGB image synthesis relies on relatively complex graphics approaches [69, 70, 71] while creating synthetic depth images is more straightforward [54, 103]. Achilles *et al.* [50] generated depth data for a bed environment by simulating a blanket covering the person, and trained a deep network using this data. While this work is close in concept to ours, the human is represented with a skeleton, which has limitations, and the code and dataset are unavailable. To improve generalization performance, researchers have used noise models with pixel dropout, spot noise, and synthetic occlusion [14, 54, 104, 105]. Others have denoised real data during test time [106], at the cost of real-time inference speed.

Annotating datasets with 3D human mesh models. Standard human pose datasets contain 2D keypoint annotations, which are pixel-wise joint position coordinate labels on images. Researchers have fit 3D human mesh models to these 2D keypoints by projecting a 3D body into image coordinates and optimizing over the human model parameters (e.g. kinematic joint angles) [35, 107]. Yin *et al.* [53] use the SPIN method [107] for fitting SMPL bodies to the SLP resting pose dataset. However, these methods suffer from depth perspective ambiguity. When additional information is present, such as 3D point clouds or scene geometry, it is possible to resolve these ambiguities, which Hassan *et al.* [59] showed. This can provide highly accurate annotations, but such optimizations require careful data preprocessing and are too slow to use during inference time.

Deep learning for 3D human pose estimation. Inferring 3D human pose is a significant branch of research in computer vision [53, 11, 86, 59, 69, 70, 35, 47, 107, 31]. In recent years, deep learning has seen widespread use for inferring human pose, by using convolutional neural networks (CNNs) to encode features in an image and output some representation of human pose. We refer the reader to literature surveys for more comprehensive coverage [25, 108]. Here we discuss approaches that are relevant to the particular black-box and white-box architectures we use. Differentiable kinematic models embedded into image encoders have gained traction in research due to their ability to produce phys-

ically plausible human models [32, 9, 47, 60, 109]. In contrast to differentiable skeleton models, parametric human mesh models such as SMPL [10] offer a better representation of body shape and size. Both of our architectures use this.

Many pose estimation methods incorporate black-box image reconstruction for purposes including heatmap regression [110], geometry awareness [86], and spatial residual error correction [87]. One such architecture that may be used for this is U-Net [85], which learns an image-to-image mapping with a latent space in the middle that can encode image classes [111] or physically meaningful features [112]. Fewer deep learning works have used white-box methods for image reconstruction, i.e. differentiable image generation methods with no learnable parameters. However, our previous work introduced a model of pressure map reconstruction [11] and was trained only with synthetic data, which we build on.

4.2 Annotating Real Data with SMPL Bodies

Here we describe an optimization method for annotating an existing human pose dataset with 3D SMPL bodies, as shown in Figure 4.2. This method finds body shape and pose parameters to fit the SMPL bodies to depth images and existing 2D keypoint annotations. The optimization includes terms for scene constraints, body mass, and height, if they are available in the dataset. By leveraging depth information, the method can resolve the ambiguities in pose which are inherent with 2D keypoint annotations alone. We annotate the SLP dataset [14] to create SLP-3Dfits, a dataset consisting of 3D fits to $4,545$ unique poses as described in subsection 4.2.2. SLP-3Dfits is used for initializing the synthetic data generation method described in section 4.3, and evaluating the deep learning methods described in section 4.4.

Our optimization requires a depth image, $\mathcal{D}$, capturing a real person who is not occluded by objects in the environment (e.g. blankets). A reprojection loss is computed between K 2D keypoints S $\in \mathbb{R}^{K\times 2}$ and 3D SMPL joint positions. Scene constraints, $\mathbb{C}$, are used to re-

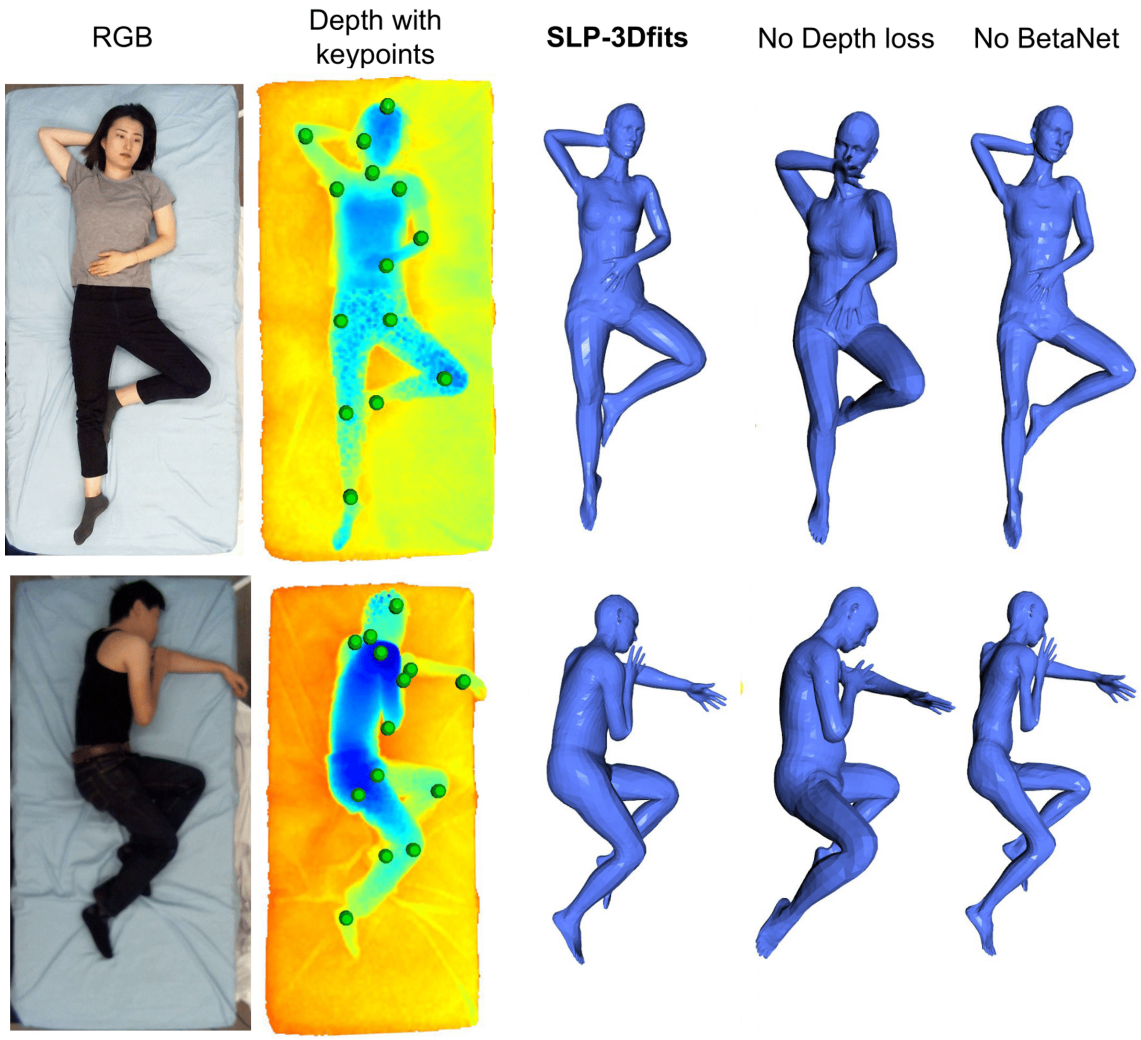

Figure 4.2: We fit 3D SMPL bodies to the SLP dataset [14], which we use for initializing the physics simulator and for training and testing our deep models. Our method resolves depth ambiguity using a loss between the SMPL mesh and 3D points from the depth image. Examples are shown without the depth loss term, resulting in poses with depth error. Examples are also shown without BetaNet, resulting in bodies with unreasonable shapes.

duce interpenetration between the body and the bed. Height and body mass measurements, h, m, are used to enforce physical consistency in the SMPL body shape, by modeling height and body mass as a function of SMPL body shape parameters. This is described in subsection 4.2.1. We define the SMPL body parameters with $\mathbf{\Psi}_R = \begin{bmatrix}\boldsymbol{\beta}_R & \boldsymbol{\Theta}_R & \boldsymbol{s}_R & \boldsymbol{\phi}_R\end{bmatrix}^\top$, which contains the body shape parameters $\boldsymbol{\beta}_R$, the joint angles $\boldsymbol{\Theta}_R$, and the global translation and rotation $\boldsymbol{s}_R$, $\boldsymbol{\phi}_R$. Subscript R distinguishes the real data annotations from parameters in later sections. The optimization seeks $\mathbf{\Psi}_R$ that minimizes the objective function E as follows:

$$E(\mathbf{\Psi}_R, \mathcal{D}, \mathrm{S}, \mathbb{C}, h, m) = E_J + \lambda_{\mathcal{D}} E_{\mathcal{D}} + \lambda_P E_P + \lambda_M E_M + \lambda_\beta E_\beta \tag{4.1}$$

The error function contains the following terms:

- $E_J(\mathbf{\Psi}_R, \mathrm{S})$ penalizes the distance between 2D keypoint annotations and SMPL joints projected into camera space.
- $E_{\mathcal{D}}(\mathbf{\Psi}_R, \mathcal{D})$ encourages a match between depth points and SMPL vertices visible from the camera's perspective. This is implemented using a robust version [59] of the Chamfer Distance [113].
- $E_M(\mathbf{\Psi}_R, \mathbb{C})$ enforces scene constraints by penalizing interpenetration between the body and the scene (bed, mattress).
- $E_P(\mathbf{\Psi}_R)$ penalizes body self-penetration, e.g., a hand interpenetrating through the chest. Collisions are detected using Bounding Volume Hierarchies [114].
- $E_\beta(\mathbf{\Psi}_R, h, m)$ encourages the SMPL shape parameters $\boldsymbol{\beta}_R$ to express a body with a height and mass that match the participant's measurements h, m. This requires a mapping from body shape to body height and mass, which is described in subsection 4.2.1.

During optimization, the joint angles $\boldsymbol{\Theta}_R$ are constrained to values based on textbook joint angle limits [75, 76, 77]. The exact joint limits are provided in the code repository. Many datasets such as the SLP dataset [14] contain a variety of different poses for a particular person. Accordingly, all such posed bodies should have the same shape. To ensure that the SMPL shape parameters for each participant are identical across all data samples, the optimization was implemented as a batched optimization, allowing the joint optimization of body pose and shape across multiple samples.

As the optimization of SMPL parameters $\boldsymbol{\Psi}_R$ provides a non-convex objective, the optimization may fall into local minima. To avoid this, each sample is initialized and optimized multiple times with different starting poses and orientations. The result with the lowest loss is selected. The optimization problem is solved using the ADAM differentiable optimizer. The method is similar to the approach used by Hassan *et al.* [59]; however, notable additions include our method of enforcing physical consistency on body height and mass, and our batched optimization that fits the same body shape across multiple pose samples.

4.2.1 BetaNet

To calculate E_β, we model body height and mass as a function of SMPL body shape parameters and gender, i.e. $\{h, m\} = f_\beta(\boldsymbol{\beta}, \boldsymbol{g})$, where h and m are values in units of meters and kilograms, respectively. Unlike $\boldsymbol{\beta}$, height and weight are directly measurable physical values that can better constrain the network. Gender is modeled with two flags, i.e. $\boldsymbol{g} \in \mathbb{R}^2$, which may account for female ($[0, 1]$), male ($[1, 0]$), or gender-neutral ($[1, 1]$) body models. In this work however, only female and male models are used. We represent the function f_β with a 2-layer fully connected network, where the input consists of body shape parameters and gender, and the output is height and body mass. We train BetaNet on a large synthetic dataset consisting of randomly shaped SMPL bodies with known height and mass, and

mass is modeled as a function of SMPL body mesh volume:

$$m = \bar{m}_{\boldsymbol{g}} \frac{\mathcal{V}_{mesh,\boldsymbol{g}}}{\bar{\mathcal{V}}_{mesh,\boldsymbol{g}}} \tag{4.2}$$

where $\bar{m}_{\boldsymbol{g}}$ is a gender-specific average body mass value we take from Tozeren [79], $\mathcal{V}_{mesh,\boldsymbol{g}}$ is the volume of a gendered body model of interest, and $\bar{\mathcal{V}}_{mesh,\boldsymbol{g}}$ is the volume of a gendered body model with average shape $\boldsymbol{\beta} = \mathbf{0}$. We train BetaNet with the following loss function:

$$\mathcal{L}_{\text{BetaNet}} := \frac{1}{\sigma_h}||h - \hat{h}||_1 + \frac{1}{\sigma_m}||m - \hat{m}||_1 \tag{4.3}$$

where $\hat{h}$ and $\hat{m}$ are the height and weight estimated by the network. Each term is normalized by standard deviations σ_h and σ_m, which are computed from the entire synthetic training dataset. The trained BetaNet is also used for the separate problem of learning a mapping from depth to pose and pressure, described in section 4.4.

4.2.2 SLP-3Dfits: SMPL fitting to the SLP dataset

Here we describe how our method of fitting parametric SMPL bodies to depth images and keypoints is applied to the SLP [14] dataset, which contains $4,590$ poses across 102 participants. We use these fits for initializing the physics simulator to generate synthetic data, for training our deep network, and for testing it.

We annotate $4,545$ poses across 101 participants (one subject is excluded due to a calibration issue). The SLP dataset contains calibrated, occlusion-free depth images (the 'uncovered' case) for every pose. 'Calibrated' in this context means that the depth images are spatially co-registered to other modalities and annotations using the calibration method in Liu *et al.* [14]. After converting these to point clouds, points from the bed surface are filtered out with a height threshold, so that all points used in the objective function are from the surface of the human body. The SLP dataset also contains 2D keypoint annotations,

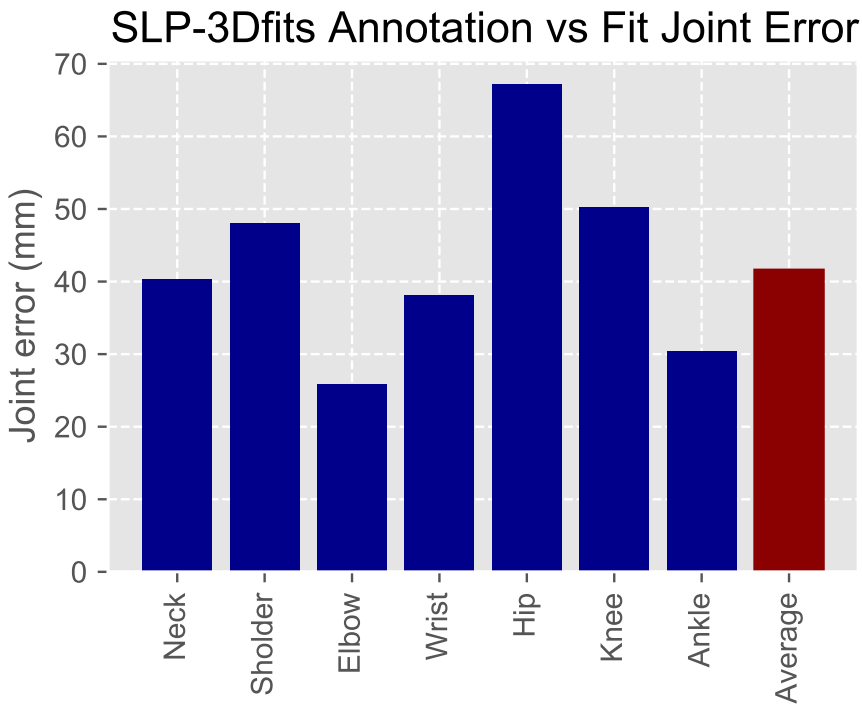

Figure 4.3: Per-joint error of SLP-3Dfits. The joint location of the SMPL model is compared to the 2D SLP annotation projected into 3D space.

body height, and body mass for all captured poses, which are factored into the objective function. A plane representing the height of the bed surface is used as a scene constraint to limit penetration into the mattress.

Generally the fits from the automatic optimization are of high quality. However, in some cases the result converges to an incorrect local minimum, usually when the participant is lying on their side and the hands are posed on the wrong side of the head. Thus, each result of the fitting process is manually checked by a human annotator for agreement with the original pose in the image data. For failure cases, the optimization is restarted with a different initialization and then re-checked. Roughly 9% of the fits required these restarts. The fits are of sufficient accuracy to use as "ground-truth" for evaluation of neural network predictions, and we have made them available publicly. The mean error between the SMPL joint locations and the 2D skeleton annotations is 41.6 mm; per-joint error is provided in Figure 4.3. Note that the SLP dataset joint annotations have some offset when compared to the SMPL model joints, inflating this error metric. The average distance from the non-bed

point cloud to the surface of the human model is 12.0 mm.

4.3 Synthetic Data Generation

We present a synthetic data generation pipeline using physics simulations that is capable of creating a large dataset of humans resting. It can generate bodies at rest on a soft mattress with depth and pressure images. In practice, this approach is much more efficient than collecting comparable real-world data. The sole purpose of this pipeline is to create a large dataset, BodyPressureSD, for training the deep model described in section 4.4. BodyPressureSD contains $97,495$ unique body shapes, poses, and image samples; data partitions are described in the evaluation (subsection 4.5.1).

The data generation pipeline consists of two processes, as depicted in Figure 4.4. The first process (I. - VI.) is similar to that of our previous work [11]; it generates bodies resting on a soft bed with synthetic pressure images. The second process (V. - IX.) generates blanket occlusions on top of the bodies in bed with synthetic depth images. It uses three simulation tools: DART [65] for simulating articulated human dynamics; FleX [8] for simulating soft materials that include the human body, bed mattress, pressure sensing mat, and blanket dynamics; and PyRender [115] for depth image rendering. Some synthetic data examples are shown in Figure 4.5.

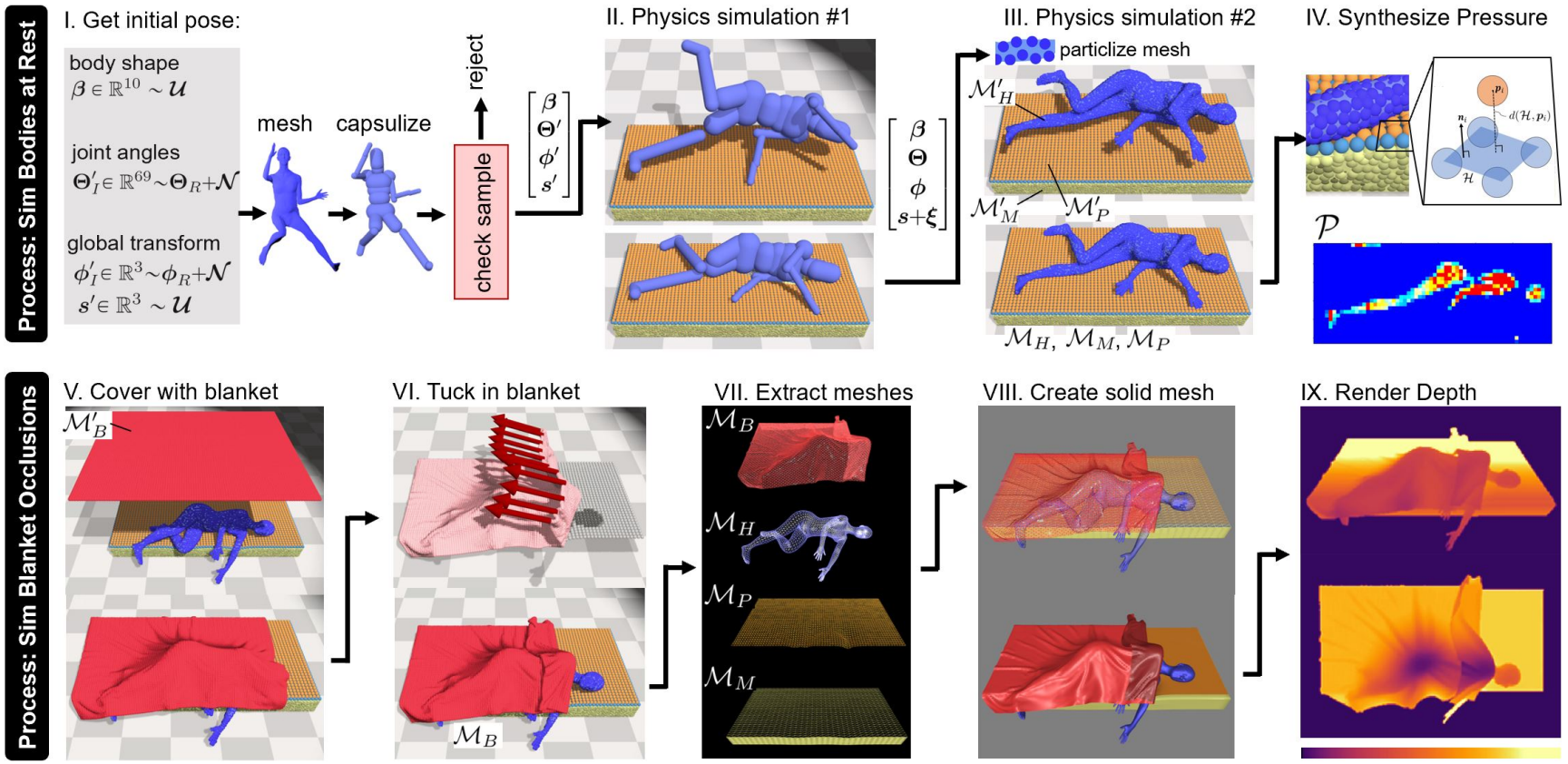

Figure 4.4: Our synthetic data generation method involves two processes: The first process is similar to our previous work [11]; it involves (I.) sampling random human poses, (II.) resting dynamic capsulized bodies on a soft bed to find a resting pose, (III.) resting a finer body representation on the bed to improve human body shape detail, and (IV.) using a simulated pressure sensing mat underneath the person to compute a pressure image. The second process involves (V.) covering the body with a blanket, (VI.) pulling the top of the blanket down to uncover the person's head, (VII.) extracting deformed meshes, (VIII.) creating a solid mesh, and (IX.) simulating a depth image from a pinhole camera positioned above the bed.

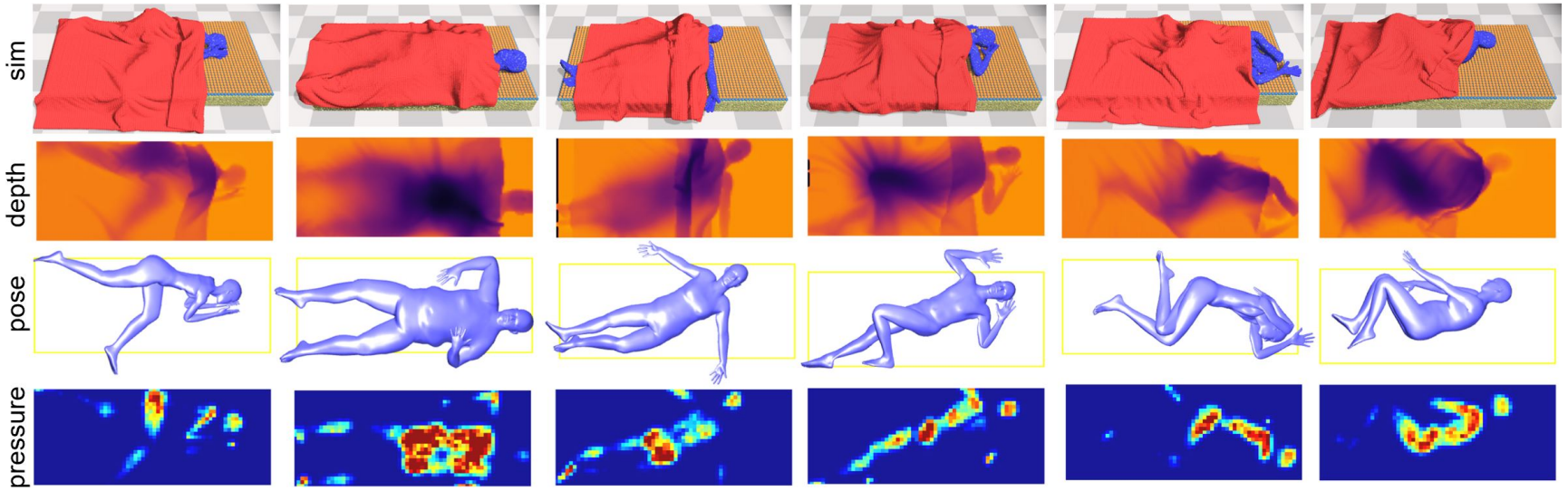

Figure 4.5: BodyPressureSD synthetic data samples created by resting bodies on a mattress and covering them with a blanket.

4.3.1 Simulating Bodies at Rest

The process begins by sampling a large set of initial synthetic body poses, where each pose sample contains joint angles $\{\boldsymbol{\Theta}'_I, \boldsymbol{\phi}'_I\}$, that are close to the real poses $\{\boldsymbol{\Theta}_R, \boldsymbol{\phi}_R\}$ fit in section 4.2. This is depicted in Figure 4.4-I. Normally distributed noise is added to the hip, knee, inner shoulder, outer shoulder, and elbow joints. Each angle j in these joints receives noise with the following equation:

$$\theta'_{I,j} = \theta_{R,j} + \mathcal{N}(0, \pi/12) \tag{4.4}$$

where $\{\theta'_{I,1}, \theta'_{I,2}, ...\} = \boldsymbol{\Theta}'_I$ and $\{\theta_{R,1}, \theta_{R,2}, ...\} = \boldsymbol{\Theta}_R$. Other joints are set equal to the real fit angles. The same amount of noise is added to two of the root angle joints representing the rotation of the body along its longitudinal axis ($\phi_{R,2}$) and the rotation of the body normal to gravity ($\phi_{R,3}$). No noise is added to $\phi_{R,1}$, which represents rotation around the sagittal axis. For each pose, the body shape is sampled from a uniform distribution, following [116]: $\boldsymbol{\beta} \sim \mathcal{U}[-3, 3]$. The 2D translation of the human body over the surface of the bed is also sampled from a uniform distribution: $s'_1, s'_2 \sim \mathcal{U}[-0.2, 0.2]$. The height of the body normal to gravity, s'_3, is set according to the lowest initial point on the body so that every part of the body is initially above the bed. Accordingly, $\{s'_1, s'_2, s'_3\} = \boldsymbol{s}'$. With the fully parameterized body of initial shape and pose $\{\boldsymbol{\beta}, \boldsymbol{\Theta}'_I, \boldsymbol{\phi}'_I, \boldsymbol{s}'\}$, body self-collisions are checked using the capsulized body from SMLPify [35]. If there is a collision, the sample is rejected, otherwise, the set of parameters becomes $\{\boldsymbol{\beta}, \boldsymbol{\Theta}', \boldsymbol{\phi}', \boldsymbol{s}'\}$.

Two physics simulations. The first process uses physics simulations to convert a body with the initial collision-free pose $\{\boldsymbol{\beta}, \boldsymbol{\Theta}', \boldsymbol{\phi}', \boldsymbol{s}'\}$ to a resting pose $\{\boldsymbol{\beta}, \boldsymbol{\Theta}, \boldsymbol{\phi}, \boldsymbol{s}\}$ with human body mesh $\mathcal{M}_H$ and pressure image $\mathcal{P}$. It is the same process as our previous work [11], with the exception of the weighting method for the simulated body, which we updated. See details in Appendix C.1. Figure 4.6 shows examples of resting poses and body shapes that

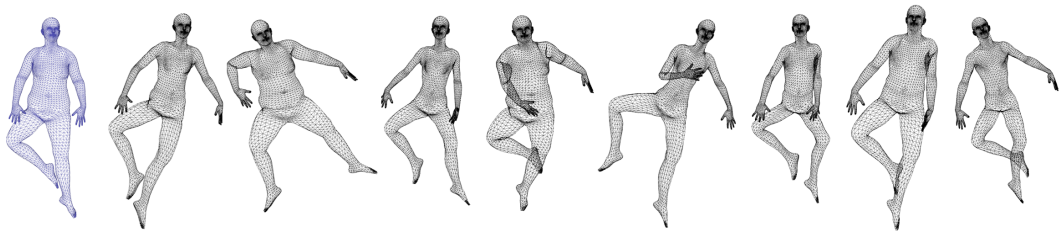

Figure 4.6: Example of resting pose diversity. Left blue pose shows a SLP-3Dfits example. Right black examples shows BodyPressureSD resting poses and body shapes that were initialized in the simulator by adding noise to the left blue pose (Equation 4.3.1) and dropped on the mattress.

are generated from a single SLP-3Dfits pose. The remainder of subsection 4.3.1 provides a high-level summary of the methods from [11].

In Physics Simulation #1 (Figure 4.4-II.), the human is modeled as an articulated rigid body made with capsule primitives. DART [65] is used to model this capsular human and simulate its dynamics. The articulated body uses the same joint angles and body shape parameters as the SMPL mesh, but unlike the mesh, the joint angles can change due to applied torques and forces (e.g. due to gravity and contact with the bed). At the same time, a different simulator, FleX [8], is used to model a mattress and a pressure-sensing mat underneath the body. FleX uses a unified particle representation to efficiently model deformable objects. These are combined in a loop to allow a dynamic articulated system (i.e., the body) to interact with soft materials (i.e., the pressure sensing mat).

While the capsulized articulated rigid body from Physics Simulation #1 is well suited for modeling ragdoll physics and finding a resting pose, it does not represent the surface geometry of the human body with sufficient fidelity for pressure image generation. The process assigns the resting pose and body shape to a SMPL mesh and fills it with deformable FleX particles, which creates a non-articulated body with a finer profile of human features. This is depicted as Physics Simulation #2 in Figure 4.4-III. This 'particlized' body is positioned above the bed with parameters $\{\boldsymbol{\beta}, \boldsymbol{\Theta}, \boldsymbol{\phi}, \boldsymbol{s} + \boldsymbol{\xi}\}$, where the term $\boldsymbol{\xi}$ represents a vertical adjustment in the root joint translation so the body can be allowed to fall a short distance to settle on the bed a second time.

The process uses polygon meshes to record the simulation state in Physics Simula-

tion #2. Initially, the meshes include the undeformed human body, mattress, and pressure sensing mat meshes. The human body mesh is a function of the resting pose: $\mathcal{M}'_H = f\big(\boldsymbol{\beta}, \boldsymbol{\Theta}, \boldsymbol{\phi}, \boldsymbol{s} + \boldsymbol{\xi}\big)$. The mattress underneath the person is set to a twin size, consisting of a rectangular prism of particles in an undeformed mesh $\mathcal{M}'_M$. The mattress is constructed using the same particlizing method as the human body [11]. The padded mat on the bed surface represents a layer of bedding underneath the person, and is constructed from a two-layer lattice of particles laced together by a grid of springs constraints, represented by mesh $\mathcal{M}'_P$. Physics Simulation #2 is run until the particlized human body reaches static equilibrium, and then outputs mesh data for the resting human body $\mathcal{M}_H$, the deformed mattress $\mathcal{M}_M$, and the deformed pressure sensing mat $\mathcal{M}_P$.

Synthesizing Pressure Imagery

Besides interacting physically with the capsulized body in Physics Simulation #1, the pressure sensing mat on the surface of the bed is also used to generate pressure images, based on particle penetration between the top (orange) and bottom (blue) layers of particles on the surface of the bed, shown in Figure 4.4-IV. The simulated sensor measures pressure as a function of how far a top layer particle penetrates the underlying particles. Each penetration distance across the mat is converted into a value on pressure image $\mathcal{P}$, which is also shown in Figure 4.4-IV. We refer the reader to a prior work for further details about this process [11].

4.3.2 Simulating Blanket Occlusions

Before simulating the cloth blanket, the process freezes the ending equilibrium state of Physics Simulation #2, which involves freezing particles representing the resting human, deformed mattress, and deformed pressure sensing mat. A blanket is created in FleX with a grid of particles shown in Figure 4.4-V. The blanket is parameterized by two sets of terms: the blanket geometry terms, which determine blanket size, particle location, particle

connection points and initial world transform; and the dynamic simulation terms, which determine the stiffness holding particles together.

The blanket has an undeformed height and width h_B and w_B, which are chosen to represent a twin size of 1.68×2.29 meters, and are created with a 102×102 particle grid. The global translation is parameterized by $\{s_{B,1}, s_{B,2}, s_{B,3}\} = \boldsymbol{s}_B \in \mathbb{R}_3$ and rotation by $\{\phi_{B,1}, \phi_{B,2}, \phi_{B,3}\} = \boldsymbol{\phi}_B \in \mathbb{R}_3$. These can be used to incorporate domain randomization (e.g. by sampling random initial blanket translations) or to better match blanket configurations in a particular dataset. The blanket is initially set to a height $s_{B,3}$ above the body so that the blanket does not initially collide with the body. The initial blanket configuration may be described by mesh $\mathcal{M}'_B = f(h_B, w_B, \boldsymbol{s}_B, \boldsymbol{\phi}_B)$. The blanket is weighted with the same density as the mattress in the previous simulation. The blanket dynamics are also influenced by the blanket stiffness K_B.

The process then runs the simulation, dropping the soft blanket on the body in bed. Depending on the initial position over the surface of the body, the blanket may cover the human's head, which is undesirable because it would likely not be a common occurrence in the real world. Thus, the blanket is adjusted by pulling on a set of particles on the top edge of the blanket - see Figure 4.4-VI. To determine if the blanket should be pulled to uncover the person's head, the process checks if the blanket is above the person's neckline. In other cases, the top edge of the blanket may be initialized at a location very far from the person's head, which could lead to the person being only partially covered. In this case, the algorithm checks if the blanket is below the person's neckline, and if it is, the same set of particles is pulled upward to better cover the person. Once the blanket reaches static equilibrium, the simulator halts and outputs the deformed blanket as mesh $\mathcal{M}_B$.

Rendering Synthetic Depth Imagery

The process extracts the deformed meshes $\{\mathcal{M}_H, \mathcal{M}_M, \mathcal{M}_P, \mathcal{M}_B\}$ as depicted in Figure 4.4-VII., and records them as part of the dataset. Then, they are assembled using Pyren-

der [115], an open source python library for rendering and visualization (Figure 4.4-VIII). Finally, the process renders $\mathcal{D}$, a depth image from a camera facing the bed. Figure 4.4-IX depicts two viewing angle perspectives, which include an observation from the side of the bed and an observation from mounting the camera directly above the bed facing downward.

4.4 Learning Pose and Pressure from Depth

We train an algorithm that learns a mapping f from a depth image $\mathcal{D}$ captured over a person at rest with gender $\boldsymbol{g}$, to a human mesh $\mathcal{M}_H$ that models pose and body shape, and a 2D array $\mathcal{P}$ that encodes the contact pressure on the surface of the mattress underneath the person:

$$\{\mathcal{M}_H, \mathcal{P}\} = f(\mathcal{D}, \boldsymbol{g}) \tag{4.5}$$

We represent f in the form of a deep network. The body mesh $\mathcal{M}_H$ and the pressure array $\mathcal{P}$ can be used to calculate the pressure distribution on the surface of the body, $\mathcal{P}_b$ (recall Figure 4.1 (c)), which contains localized pressure on specific body parts. We define $\mathcal{P}_b$ as a collection of human body mesh vertices that are each assigned a pressure value due to contact from the underlying surface. Because the mesh $\mathcal{M}_H$ is learned with a 6-DOF pose in the world reference frame and the contact pressure array $\mathcal{P}$ is collected by a sensor mounted at a known position in the world, they are implicitly co-registered. Assuming that the pressure mat exists within a flat plane normal to gravity on the top surface of the bed, each contact pressure element p_{xy} may be projected normal to gravity onto the human mesh, where p_{xy} is assigned to mesh vertex v_j if its taxel area contains the x, y position of v_j. A *taxel* is defined as a tactile pixel on a force-sensing array [11]. This mapping approximates the complex phenomena that occur when the mat is deformed due to contact

by neglecting stretching and folding. As such,

$$\mathcal{P}_b : \mathcal{P} \mapsto \mathcal{M}_H \tag{4.6}$$

This projection is sufficient to localize pressure on specific body parts, because vertex indices on the SMPL model are independent of body pose and shape, i.e. a heel vertex is always on the heel. Vertices above the undeformed bed height are set to zero because they are not in contact, and vertices with non-zero pressure are required to be unoccluded from the pressure mat by other body parts or surfaces of the body. The latter is ensured by casting a ray downwards from each vertex toward the mat and checking if it passes through any triangular faces. If it passes through a face, the pressure is set to zero. This will ignore pressure due to self contact between parts of the body.

4.4.1 BodyPressureWnet

Here we describe BodyPressureWnet (BPWnet), a deep network with a white-box model of depth and pressure image generation, shown in Figure 4.7. BPWnet uses a traditional CNN for encoding depth images, but uses a white-box model for reconstructing depth and pressure images from an embedded SMPL human model. These white-box models are analytic and fully observable with no learned parameters, and are also differentiable. BPWnet contains two modules: 'Mod1', which produces an initial estimate, and 'Mod2', which refines the estimate through residual error in a similar way to previous works [87, 57, 11].

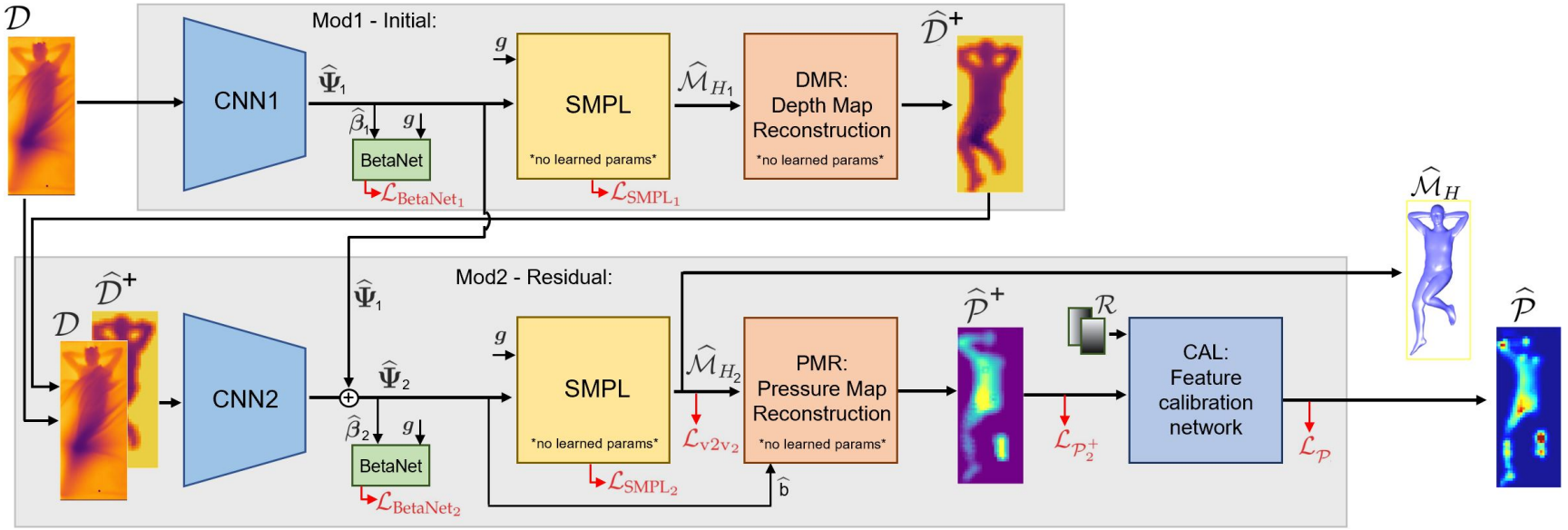

Figure 4.7: BodyPressureWnet (BPWnet), a deep network that learns a mapping from depth and gender to pose and contact pressure. Depth, $\mathcal{D}$ is encoded with a black-box CNN and outputs SMPL [10] human model parameters $\widehat{\mathbf{\Psi}}$, which is used to reconstruct a SMPL human mesh $\widehat{\mathcal{M}}_H$. Using white-box image reconstruction components DMR and PMR, it refines the pose estimate and outputs a pressure map. The pressure map features are calibrated with CAL to produce a contact pressure estimate $\widehat{\mathcal{P}}$. The two module design refine estimates with initial and residual stages; subscripts 1 and 2 indicate estimates from each module, respectively.

First, BPWnet maps a depth image $\mathcal{D}$ to SMPL parameters $\widehat{\mathbf{\Psi}}$ using a CNN. From there, it uses the differentiable SMPL embedding from Kanazawa *et al.* [47] to produce a SMPL mesh estimate, $\widehat{\mathcal{M}}_H$. A loss is applied using height and weight estimates from BetaNet. The first module, Mod1, contains a white-box model of depth image generation, which reconstructs depth maps $\widehat{\mathcal{D}}^+$ from a SMPL mesh. The spatial residual between these maps and the input depth images are used to learn a correction and refine initial human pose estimates in the second module, Mod2. In contrast to Mod1, Mod2 contains a white-box model of pressure image generation, which differentiably reconstructs pressure maps $\widehat{\mathcal{P}}^+$ from an improved SMPL mesh estimate. Finally, the CAL component in Figure 4.7 adjusts pressure maps to achieve a similar calibration to real pressure images.

Depth image encoding. Both modules of BPWnet encode depth imagery with ResNet34 convolutional neural networks (CNNs). Each CNN outputs estimated SMPL parameters $\widehat{\mathbf{\Psi}} = \left[\hat{\boldsymbol{\beta}} \;\; \widehat{\mathbf{\Theta}} \;\; \hat{\boldsymbol{s}} \;\; \hat{\boldsymbol{x}} \;\; \hat{\boldsymbol{y}} \;\; \hat{b}\right]^\top \in \mathbb{R}^{89}$, which contains body shape, joint angles, and root translation and rotation. It also contains an estimated distance between the camera and the bed, $\hat{b}$, which is described later in this section. The SMPL parameters are used to compute a SMPL human mesh model with no learnable weights. The SMPL block takes as additional input a set of gender flags $\boldsymbol{g} \in \mathbb{R}^2$ (recall subsection 4.2.1) and outputs a human mesh $\widehat{\mathcal{M}}_H$. We define a loss on the SMPL model, $\mathcal{L}_{\text{SMPL}}$, which minimizes error from the SMPL parameters and 3D SMPL joint positions. We also define a loss on the SMPL vertex positions, $\mathcal{L}_{\text{v2v}}$, which can be used in conjunction with $\mathcal{L}_{\text{SMPL}}$ to provide more supervision at a marginally higher computational cost. Appendix D.1 contains details on $\mathcal{L}_{\text{SMPL}}$ and $\mathcal{L}_{\text{v2v}}$. BPWnet also provides supervision on human mass and height using the BetaNet described in subsection 4.2.1.

White-box decoding. In Mod1, the depth map reconstruction (DMR) module computes a depth map $\widehat{\mathcal{D}}^+ \in \mathbb{R}^{64\times27}$ from the body mesh $\widehat{\mathcal{M}}_{H,1}$. This is computed by calculating the distance between the camera plane and the inferred human mesh (Figure 4.8 - left). The reconstructed depth map is used for residual error refinement in Mod2: The dif-

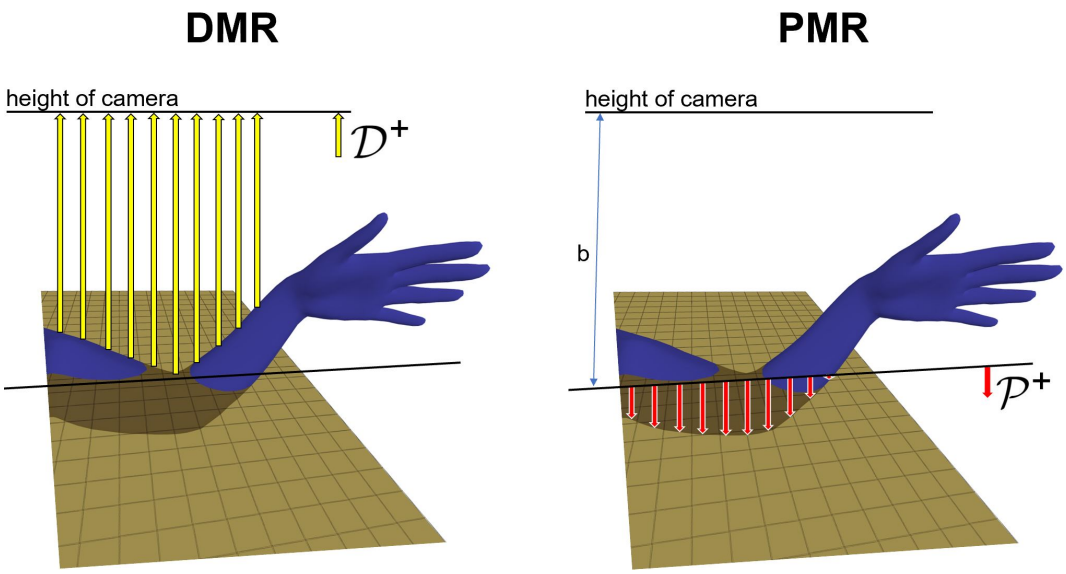

Figure 4.8: Differentiable white-box depth and pressure map reconstruction. DMR computes a linear depth map $\mathcal{D}^+$ between the height of the camera and the top surface of the human mesh (left). PMR computes a linear pressure map $\mathcal{P}^+$ between the undeformed height of the surface of the bed and the human mesh (right). Variable b is the distance between the camera and the bed.

ference between the real pose in the input depth image $\mathcal{D}$ and the pose in the reconstructed depth map $\widehat{\mathcal{D}}^+$ contains information that can be used to improve the initial pose and body shape estimate. This white-box model of depth image generation is similar to the pressure image generation introduced in our previous work [11]. Unlike the input depth image $\mathcal{D}$, the DMR reconstruction $\widehat{\mathcal{D}}^+$ only contains human mesh information and is occlusion-free; specifically, it does not contain blanket or mattress information because its sole purpose is to improve the initial pose estimate.

Mod2 uses pressure map reconstruction (PMR) from [11] to compute a pressure map $\widehat{\mathcal{P}}^+ \in \mathbb{R}^{64\times27}$ from the refined pose estimate $\widehat{\mathcal{M}}_{H,2}$. This is the distance the body mesh sinks into the underlying mattress (Figure 4.8 - right). In contrast to the previous work that reconstructs a pressure map from a *pressure* image, in BPWnet, PMR reconstructs a pressure map from a *depth* image. The position of the mattress must be known with high accuracy relative to the depth camera, because the pressure is sensitive to small changes in the vertical distance between the camera and bed. Small camera movements or the weight of a large person on the bed can change the perceived distance enough to alter the

reconstructed pressure map. Thus, it requires an additional variable to represent changes in the vertical distance between the camera and the surface of the bed, which can correct for changes in the camera's position. We define this parameter as b, and learn it from depth images at varying distance from the bed. We also define a loss on the PMR component, $\mathcal{L}_{\mathcal{P}^+}$, which can be used to train the depth image encoder. See details in Appendix D.1.

Calibrating the inferred pressure map. While $\widehat{\mathcal{P}}^+$ is spatially similar to a pressure image, it has some qualitative differences: it contains more dilated features (i.e. the spread of a given pressure point is wider), it has less noise, and the magnitude of each pixel is a distance rather than a pressure. Thus, our approach uses a small convolutional network, CAL, to calibrate $\widehat{\mathcal{P}}^+$, converting it to $\widehat{\mathcal{P}} \in \mathbb{R}^{64\times27}$. CAL takes as input a stack of 3 images including $\widehat{\mathcal{P}}^+$ and constant CoordConv maps $\mathcal{R} \in \mathbb{R}^{2\times64\times27}$, which allow the network to model non-translation invariant aspects and can improve trainability and generalization [117]. CAL contains 4 layers of convolution, with $< 0.4\%$ as many parameters as the encoder. We define a loss based on the output pressure image, $\mathcal{L}_{\mathcal{P}}$, which can be used to train CAL. See Appendix D.1.

Training Strategy

We train the encoders for Mod1 and Mod2 separately. The loss for Mod1 is computed as:

$$\mathbb{L}_{\text{BPW}_1} = \mathcal{L}_{\text{BetaNet}_1} + \mathcal{L}_{\text{SMPL}_1} \tag{4.7}$$

where BetaNet is separately pretrained (subsection 4.2.1) and contains frozen network weights. Then, the entire dataset is passed forward through the network to compute a set of Mod1 estimates with each sample containing $\{\widehat{\boldsymbol{\Psi}}_1, \widehat{\mathcal{D}}^+, \widehat{\mathcal{C}}_{\text{d}^+}\}$, where $\widehat{\mathcal{C}}_{\text{d}^+}$ are the binary maps of $\widehat{\mathcal{D}}^+$, created by setting background depth values on the bed surface to zero and all higher values corresponding to the human surface to one. The purpose of binary maps is to help the network learn from small values on the human surface that are important, yet distinct from the zero values on the bed surface. In this forward pass, noise is added to the

SMPL parameters to increase the variation in the types of error that Mod2 corrects. With a dataset consisting of inputs $\{\boldsymbol{\mathcal{D}}, \widehat{\boldsymbol{\mathcal{D}}}^{+}, \widehat{\boldsymbol{\mathcal{C}}}_{\mathrm{d}^+}\}$, we train Mod2 to learn a residual correction $(\widehat{\boldsymbol{\Psi}}_2 - \widehat{\boldsymbol{\Psi}}_1)$ with the following loss:

$$\mathbb{L}_{\mathrm{BPW}_2} = \mathcal{L}_{\mathrm{BetaNet}_2} + \mathcal{L}_{\mathrm{SMPL}_2} + \mathcal{L}_{\mathrm{v2v}_2} + \mathcal{L}_{\mathcal{P}_2^+} \tag{4.8}$$

After training the depth image encoders, we train the CAL network. CAL learns to refine the features of $\widehat{\mathcal{P}}^{+}$ and calibrate pressure values at individual taxels rather than to spatially adjust pressure for a change in limb or body movement. Ground truth human meshes from the dataset are used to compute ground truth reconstructed pressure maps $\mathcal{P}^{+}$, which are fed into CAL during training. CAL outputs the estimate $\widehat{\mathcal{P}}$, which is compared to ground truth $\mathcal{P}$ during training, using loss $\mathcal{L}_{\mathcal{P}}$.

4.5 Evaluation

We evaluate our method using the SLP multimodal dataset [14], which is a human pose dataset consisting of $4,590$ unique resting poses in bed across 102 human participants. Each pose is captured with three different situations of varying visual occlusion: (1) thin sheet covering the person, (2) thicker blanket covering, and (3) no covering. The dataset contains RGB, depth, point cloud, thermal, and pressure imagery, as well as 2D human pose keypoints; the bottom row of Figure 4.1 (b) and Figure 4.2 provide a couple examples. In subsection 4.5.1, we describe the data partitions generated using the method in section 4.3. Finally, in subsection 4.5.2, we describe the evaluation of the deep network from section 4.4 on the SLP dataset.

4.5.1 BodyPressureSD Synthetic Dataset Partitions

The synthetic data generation method from section 4.3 is used to generate BodyPressureSD: a large collection of samples, each of which includes a resting pose, a unique

Table 4.1: Human Pose and Body Shape Dataset Partitions

Description	synthetic train ct.	real train ct.	real test ct.
Supine - Unique Images w Blankets	34619	2370	660
L. Lateral - Unique Images w Blankets	32050	2370	660
R. Lateral - Unique Images w Blankets	30826	2370	660
Supine - Unique Images w/o Blankets	34619	1185	330
L. Lateral - Unique Images w/o Blankets	32050	1185	330
R. Lateral - Unique Images w/o Blankets	30826	1185	330
Total Unique Images	194990	10665	2970
Supine - Unique Poses	34619	1185	330
Right Lateral - Unique Poses	32050	1185	330
Left Lateral - Unique Poses	30826	1185	330
Total Unique Poses	97495	3555	990
Total Unique Body Shapes	97495	79	22
Num Samples for Training and Testing	97495	10665	2970

body shape, a gender, a depth image, a pressure image, and four meshes from the scene for the person, mattress, pressure sensing mat, and blanket. The pressure images for both this data and the real SLP data are normalized by body mass. The depth and pressure images are spatially co-registered with the calibrated images in the real SLP dataset. Details on these procedures are provided in Appendix C.2 and Appendix C.3.

Human body pose partitions. To create the synthetic training dataset, our process samples initial poses and body shapes close to the real poses as depicted in Figure 4.4-I. For each unique real pose among the 80 training subjects in the SLP dataset, it attempts to generate 30 initial synthetic poses split evenly between female and male SMPL bodies. Each of these poses have a unique body shape. Given 80 participants in the training dataset with 45 unique poses per participant, this would ideally generate $30 \times 80 \times 45 = 108,000$ initial poses and body shapes. However, in some cases it is challenging to find a collision-free pose for a particular body shape that is close to a particular real pose, so some samples are aborted when a limit is reached. This reduces the initial pose/body shape count to $103,966$.

Next, the process runs this set of bodies with initial poses through Physics Simulations #1 and #2, of which some more are rejected due to the simulation becoming unstable.

The process is designed to automatically detect when simulation instability is imminent, in which case it aborts the simulation and rejects the pose. This can happen due to situations such as a limb poking a hole in the pressure mat, which are described in our previous work [11]. None of the blanket covering simulations resulted in instability. This resulted in a total of $97,495$ unique data samples, which are used to train the network. The data partitions are broken down in Table 4.1.

Blanket configuration partitions. The SLP dataset contains both thick and thin covers, which are placed to cover most of the body. The simulator is only equipped to generate blankets with a single fixed thickness, which we assume is close enough to represent both real coverings. The real blankets often contain many wrinkle features that may be caused by pulling or adjusting the blanket so that it covers the body appropriately. We attempt to mimic these situations. The process incorporates randomization in the initial position of the blanket over the surface of the bed $\boldsymbol{s}_B$ (recall the top half of Figure 4.4-V). Each resting body is associated with a single initial blanket position and the covering variation that results from it. The initial blanket configurations are split into two partitions. In the first, the blanket is centered over the person with the upper edge coinciding with the person's neckline, such that no pulling is required to uncover the head or cover the body (recall Figure 4.4-VI.). In the second partition, the initial blanket position is randomly sampled across the person in bed. These two blanket configuration scenarios are split 50/50 among the synthetic data pose partitions described in Table 4.1. In all cases, the initial blanket rotation, $\boldsymbol{\phi}_B$, is set to a constant value of $\mathbf{0}$. Appendix C.4 provides details on the specific sampling bounds.

4.5.2 Network Evaluation

For human pose inference, we compare our method to the Pyramid fusion scheme by Yin *et al.* [53], which uses 4 modalities (RGB, depth, thermal, and pressure imagery) to infer a SMPL mesh. Our method uses only depth imagery, which would have significant advan-

tages for real-world deployment. Pose accuracy is evaluated with 3D mean-per-joint position error (MPJPE), using the same 22-subject test set proposed in Yin *et al.* For each pose sample, the inferred positions of 24 joints on the SMPL model are compared to ground truth using 3D Euclidean error.

For both contact pressure inference and the inferrence of pressure on the human body, we did not find any existing work for comparison. We designed a second deep architecture to compare BPWnet with, which uses a more traditional black-box method of image reconstruction. We refer to this alternative as BodyPressureBnet (BPBnet). BPBnet replaces the white-box DMR and PMR components in BPWnet with black-box decoders. These learned image decoders are designed symmetrically to the encoders, expanding the SMPL parameters back into images. Appendix D.2 contains details about BPBnet. We train both network architectures with the same hyperparameters, and compare the inferred pressure image to ground truth using mean-squared error (MSE), which is computed on a per-taxel basis. We compare the ability to localize regions of high pressure density using vertex-to-vertex pressure (v2vP) mapped to the SMPL model, where MSE is computed on a per-vertex basis. Because vertices are not evenly distributed over the surface of the body and pressure is in units of $force/area$, the pressure on each vertex is normalized by the average area of the adjacent triangles. We also compare human pose estimation error between BPWnet and BPBnet.

Dataset splits. We trained on datasets consisting of real, synthetic, and combined synthetic and real data. For the synthetic training data, we selected depth images with blanket occlusions on $2/3$ of the poses, and depth images without blankets for $1/3$ of the poses. This matches the real data, of which $2/3$ of images are occluded by blankets $1/3$ are not. We performed validation and testing on real data. We created both training/validation and training/testing splits based on the 102 subjects in the SLP dataset. For training/validation, we trained on data from the first 70 subjects (i.e. subjects 1 - 70, with $9,315$ real and $85,114$ synthetic samples), and validated on the next 10 subjects (i.e. subjects 71 - 80, with $1,350$

real samples). We used this split for tuning network hyper parameters. For training/testing, we used the same split as Yin *et al.* [53], and trained on data from the first 80 subjects (i.e. subjects 1 - 80, with $10,665$ real and $97,495$ synthetic samples), and tested on the last 22 subjects (i.e. subjects 81 - 102, with $2,970$ real samples). We did not use subject 7 data due to errors in calibration.

Depth image noise model. Our camera noise model includes white noise, dropout, and synthetic occlusion on sections of the input image using the code provided by Liu *et al.* [14]. Because the inferred contact pressure is highly sensitive to the distance between the camera and the bed, a single distance is uniformly sampled between -5 and 5 cm for each image, which is added to all depth pixels to make the inference robust to vertical movements of the camera or bed. No rotational noise or translational noise in the plane normal to gravity are added, since they appeared to be unnecessary for the SLP dataset. Noise is also added to account for the physics of the bed springs underneath the soft mattress. The simulated mattress is set on a rigid plane, which differs from the flexible springs in Invacare Homecare bed used to collected real data in [11] as well the bed used to collect the SLP dataset [14]. In practice, we observe a substantial drop in the middle of the bed when a person rests on it. This is modeled with a 2D parabolic map added to depth images during training, which is equal to zero at the edges of the bed and increases to a max in the center. A parameter that alters this max value is uniformly sampled between 0 and 10 cm.

Network hyper-parameters. For all networks, we shuffled the training data, used a batch size of 128, used the ADAM optimizer [43] for gradient computation, and used a learning rate of 0.0001 and weight decay of 0.0005. For BetaNet, we trained for 500 epochs on real and synthetic data. The BetaNet used in the optimization of section 4.2 was only trained with synthetic data because the body shape parameters were not available prior to annotation. For the CAL network in BPWnet, we trained for 500 epochs on real and synthetic data. For both BPBnet and BPWnet, we trained on 100 epochs on the first module. Then, we pre-computed their estimates and used it for training the second module,

which we trained for 40 epochs. We trained for 40 epochs because the network began to overfit the training data at this point regardless of the training dataset used. Our machine has a AMD Ryzen Threadripper 1950X 16-Core processor with 64 GB of CPU RAM and a NVIDIA RTX-3090 GPU. Training CNN1 and CNN2 network modules each took ~ 12 hrs, while BetaNet and CAL took < 3 hrs.

Network computation performance. Overall, BPWnet had better computational performance than BPBnet. First, it is a more parsimonious model. Using white-box image reconstruction with DMR and PMR in Figure D.1 instead of the learned ResNet decoder BPBnet reduces the learnable network parameters by 12%. Second, it is more memory efficient. Our BPBnet implementation uses almost 23 GB of GPU RAM, which will not fit on many consumer grade GPUs. The white-box model can be tuned to sacrifice speed for improved memory while keeping an encoder training batch size of 128 – at a cost of doubling the training time, the memory footprint reduces from 17 GB to 11.5 GB.

4.6 Results and Discussion

In this section, we present and discuss the results.

Our method for estimating pose outperforms the state-of-the-art. Using only a depth image in the input, BPWnet is able to infer pose with 12% lower error than the state-of-the-art method from [53], which uses a combination of RGB, depth, thermal, and pressure imagery to infer pose. We note that Yin *et al.* [53] use a different ground truth fitting method from [107], but their fits and code are not released so we compare to their reported error. BPWnet also outperformed the alternative BPBnet. Table 4.2 presents these results, with comparisons between the type of covering on the person. Figure 4.9 presents a visual overview of BPWnet performance.

Table 4.2: Human pose estimation (Pose), contact pressure (P. Img.), and pressure distribution (v2vP) error results when evaluating on the 22 subject test set.

Network Description	Training data ct.	Real	Synthetic	Modalities	Uncovered - Pose MPJPE (mm)	Cover 1 - Pose MPJPE (mm)	Cover 2 - Pose MPJPE (mm)	Overall - Pose MPJPE (mm)	Uncovered - P. Img. MSE (kPa^2)	Cover 1 - P. Img. MSE (kPa^2)	Cover 2 - P. Img. MSE (kPa^2)	Overall - P. Img. MSE (kPa^2)	Uncovered - v2vP MSE (kPa^2)	Cover 1 - v2vP MSE (kPa^2)	Cover 2 - v2vP MSE (kPa^2)	Overall - v2vP MSE (kPa^2)
BPWnet	97K		×	D	95.79	112.08	109.83	105.90	1.665	1.764	1.738	1.772	3.403	3.378	3.349	3.376
BPWnet	11K	×		D	103.22	104.56	104.83	104.20	1.470	1.455	1.444	1.456	2.764	2.723	2.685	2.724
Pyramid Fusion [53]	11K	×		RGB-D-T-P	78.80	79.92	80.21	79.64	-	-	-	-	-	-	-	-
BPBnet, No SMPL	108K	×	×	D	-	-	-	-	0.825	0.959	0.932	0.905	-	-	-	-
BPBnet	108K	×	×	D	70.16	76.99	76.49	74.54	**0.772**	**0.884**	**0.858**	**0.838**	2.497	2.491	2.486	2.492
BPWnet	108K†	×	×	D	**63.64**	**72.40**	**72.04**	**69.36**	1.155	1.209	1.190	1.184	**2.449**	**2.448**	**2.420**	**2.439**

† *indicates the encoder was trained with 108K mixed, while the CAL component was trained using only 11K real.*

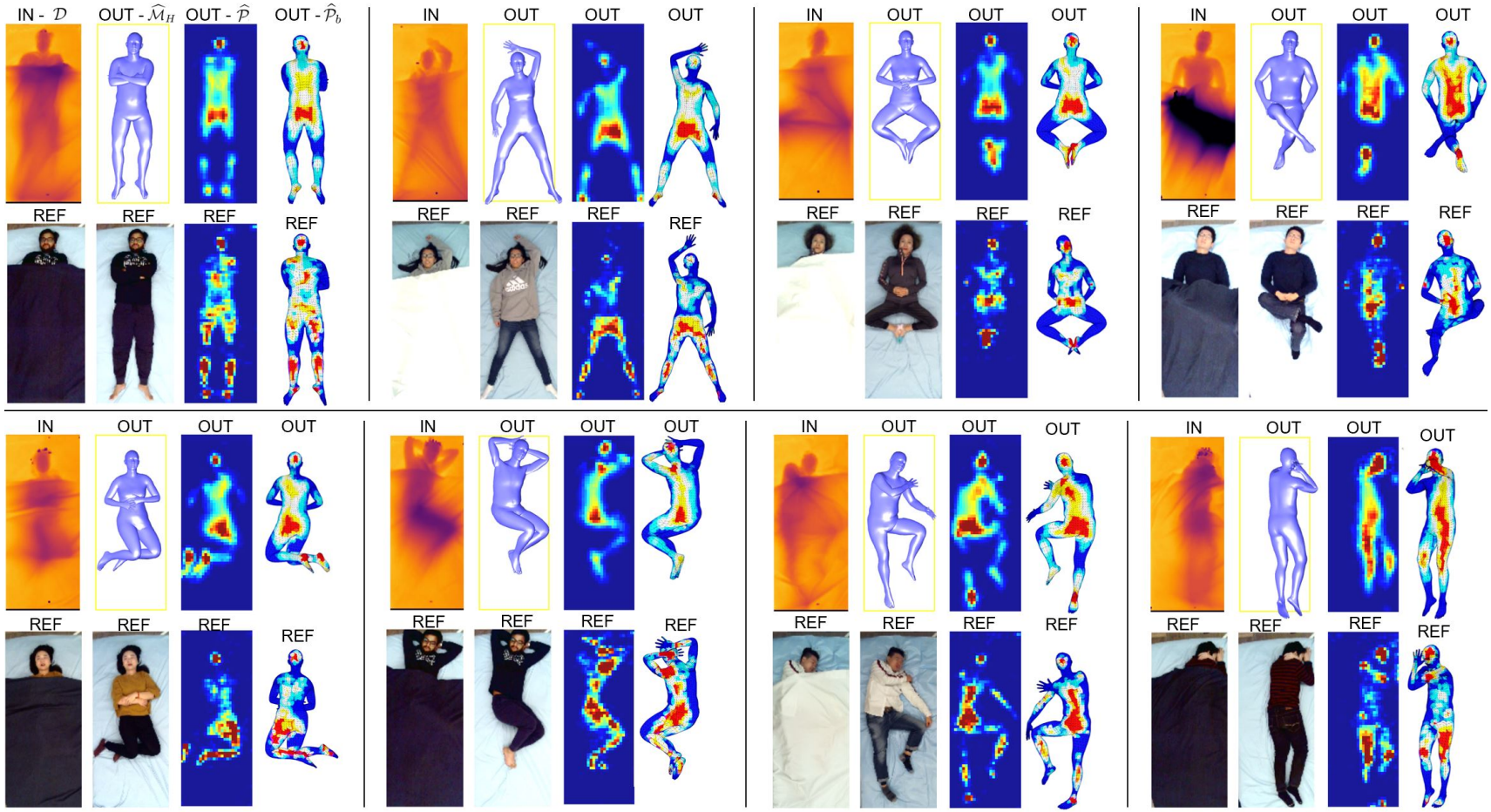

Figure 4.9: Results: Inferring pose, contact pressure, and localized pressure distribution from depth using BPWnet. Showing real data with people occluded by blankets in the depth images. All cases are from the last 22 test subjects in the SLP dataset; white coverings indicate 'cover 1' and black indicate 'cover 2.' The far right renderings in each group are a mirror flip because they show the pressure distribution *underneath* the body; the top shows an inferred pose while the bottom shows a pose from the SLP-3Dfits annotations.

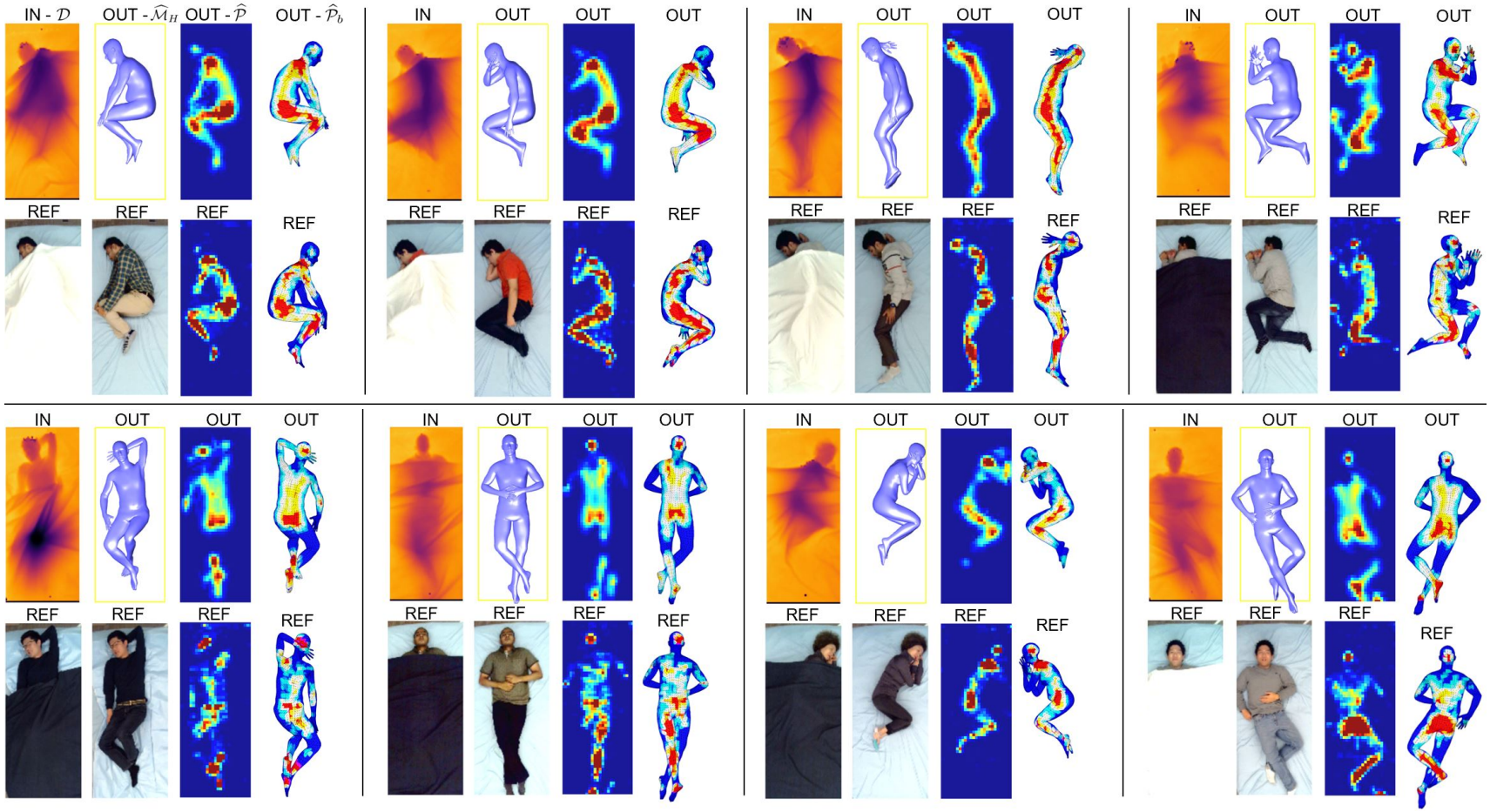

Figure 4.10: Results - continued.

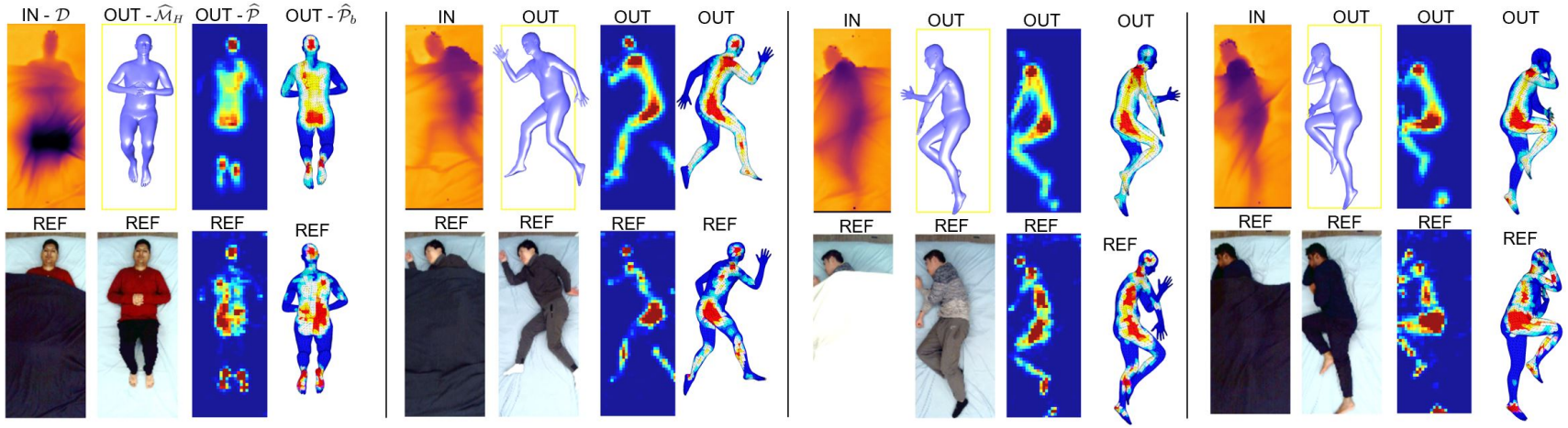

Figure 4.11: Results - continued.

Table 4.3: Pressure distribution - v2vP, MSE (kPa^2) error comparison across common pressure injury risk regions [93, 94].

Network	Head	L. heel	R. heel	R. hip	L. hip	L. shoulder	Sacrum	R. shoulder	L. elbow	L. toes	R. toes	R. elbow	Ischium	Spine
BPWnet, 108K†	3.609	3.217	2.946	2.524	2.371	1.774	1.704	1.624	1.196	1.188	1.180	1.105	0.650	0.441

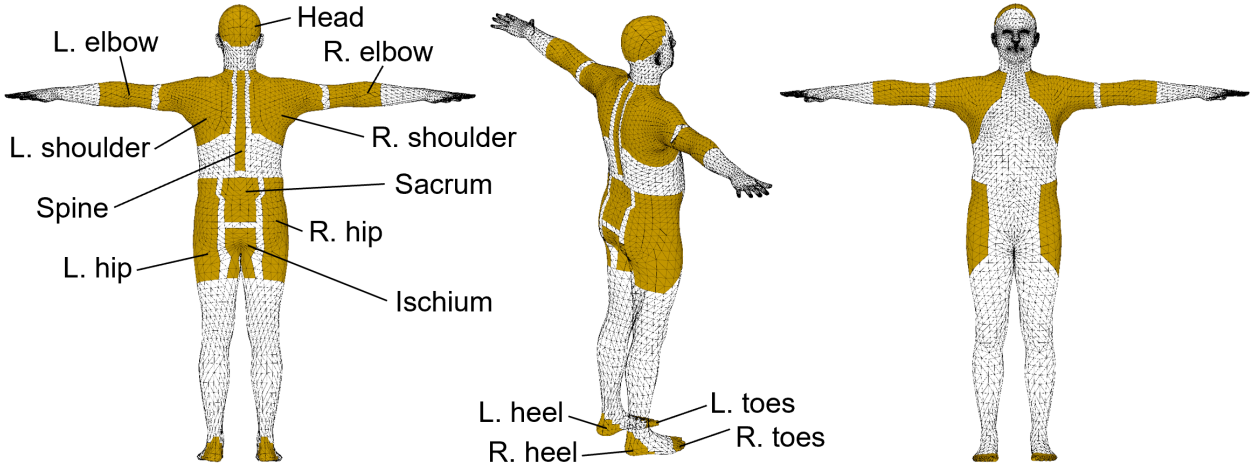

Figure 4.12: SMPL segmentation into pressure injury risk areas.

Mixing synthetic and real data boosts performance. When training the network only on real depth images from the SLP dataset or only on synthetic depth images from BodyPressureSD, performance lags. However, when the real and synthetic datasets are naively mixed into a single bag of training data, pose error drops by more than 30%.

Our method can infer contact pressure from depth. Our method can infer a pressure image ($\mathcal{P}$) from overhead depth imagery, which is also depicted visually in Figure 4.9. Like pose inference, using a mixed bag of synthetic and real data boosts performance, as shown in Table 4.2. The error for BPBnet is lower than BPWnet. When the SMPL model is ablated from BPBnet, it is still able to infer pressure but performs worse. This indicates that joint learning of the SMPL parameters helps BPBnet learn contact pressure from depth. We did not ablate the SMPL model from BPWnet because it requires SMPL to infer a pressure image.

BPWnet can infer pressure on the body. While black-box image reconstruction is far more common in computer vision and our black-box model (BPBnet) has a lower pressure image inference error, our white-box model (BPWnet) has a lower pressure distribution ($\mathcal{P}_b$) error as measured by vertex-to-vertex pressure (v2vP) in Table 4.2. We provide additional v2vP results on common pressure injury risk regions in Table 4.3 and depict the segmentation of the risk regions on the SMPL mesh in Figure 4.12.

Generally, BPWnet can more reliably localize pressure than BPBnet. This is because the inferred pressure image in BPWnet can be reduced to a function of only the SMPL parameters (i.e. $\widehat{\mathcal{P}} = f(\widehat{\mathbf{\Psi}})$) where the spatial mapping between the human mesh $\widehat{\mathcal{M}}_H$ and the pressure map $\widehat{\mathcal{P}}^+$ is a known geometric function with no learnable parameters, so the reconstructed pressure image reliably projects onto the surface of the inferred human mesh. In contrast, there is little in the black-box model to ensure the inferred pressure map spatially co-registers with the inferred mesh. We present a visual example of this phenomena in Figure 4.13. In it, the person lays supine on the bed with the left foot tucked under the right upper leg. This produces a peak pressure on the left foot, which carries

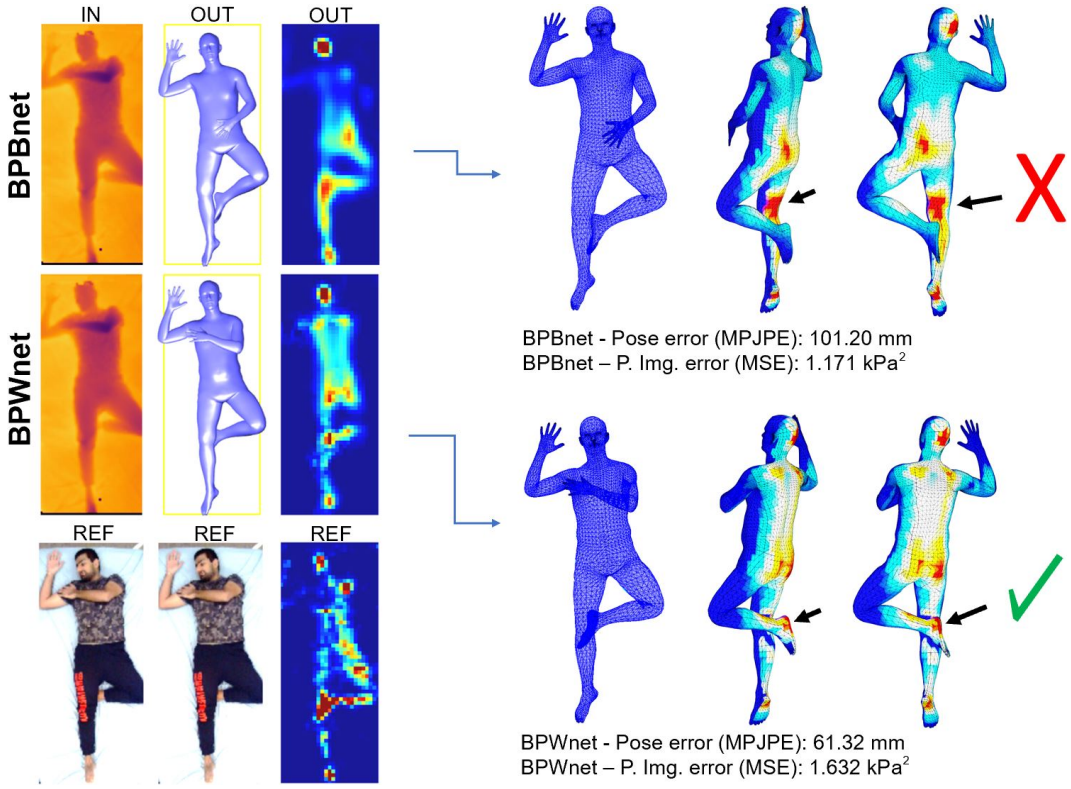

Figure 4.13: Behavior comparison of BPBnet and BPWnet. BPBnet has lower contact pressure error, but its projection onto the inferred mesh contains an artefact. BPWnet plausibly infers a high pressure on the foot, while BPBnet incorrectly assigns a high pressure to the underside of the upper leg.

extra weight from the right leg. BPWnet appropriately projects the peak pressure onto the left foot, while the BPBnet projection contains a discrepancy in the peak pressure location.

Pose vs. contact pressure accuracy tradeoff. A number of factors may account for the tradeoff in pose vs. contact pressure accuracy between BPWnet and BPBnet. For pose estimation, the reconstructed depth map estimate $\widehat{\mathcal{D}}^{+}$ in BPWnet is a constrained function of SMPL parameters and thus provides more consistent spatial residual feedback to learn the pose correction in Mod2. If the input depth image is highly occluded or contains noise, BPWnet may produce a poor initial pose estimate but $\widehat{\mathcal{D}}^{+}$ will contain no less pose information. In contrast, the black-box reconstructed depth map in BPBnet has no lower bound on the amount of information it contains, so in some cases it may not contain any useful information for the correction.

For the pressure image inference, Table 4.2 indicates that the SMPL model is not nec-

Table 4.4: Ablation study - evaluated on the 22 subject test set. Pose and pressure error shown with same metrics as previous tables. Body mass and height are evaluated with mean absolute error.

Network	Residual w/ DMR	BetaNet	CAL feature calibration	Overall - Pose MPJPE (mm)	Overall - P. Img. MSE (kPa^2)	Overall - v2vP MSE (kPa^2)	Body mass MAE (kg)	Body height MAE (mm)
BPWnet, 108K†		×	×	72.84	1.215	2.470	7.42	44.35
BPWnet, 108K†	×		×	**68.62**	1.296	2.494	7.62	65.97
BPWnet, 108K†	×	×		69.36	16.326	31.806	**5.64**	**39.45**
BPWnet, 108K†	×	×	×	69.36	**1.184**	**2.439**	**5.64**	**39.45**

essary for inferring the pressure image with BPBnet. Thus, a pixel-to-pixel black-box network is sufficient for inferring a pressure image from an occluded depth image. BPWnet likely decreases the performance of this inference because it has fewer free parameters and a tighter grasp on the pressure image formation process: if the pose changes, the pressure necessarily changes. Some poses may be more challenging to learn than some instances of contact pressure, so an inaccurate pose would adversely affect the pressure image inference.

Ablating BPWnet components reduces the performance of body pressure inference. We conducted an ablation study to test the importance of components in BPWnet, shown in Table 4.4. We ablated BetaNet, which improved pose accuracy, marginally reduced pressure accuracy, and substantially reduced accuracy of body height and weight. Coincidentally, the BetaNet model of height and mass also seems to cause a tradeoff in pose vs. pressure accuracy. However, the reduction in overall body shape accuracy as measured by height and weight casts some doubt on the merit of omitting BetaNet. We ablated the CAL feature calibration component, which affects only the contact pressure inference. Without CAL, the pressure inference performs poorly because the overall scale of the PMR output, $\mathcal{P}^+$ is different than $\mathcal{P}$. This indicates that CAL is able to scale the pressure from an arbitrary range to an appropriate range. We also ablated the residual learning by removing the DMR component in Mod1 and used PMR instead to compute pressure maps. For this,

Table 4.5: CAL feature calibration test - evaluated on the 22 subject test set. Pressure error shown with same metrics as previous tables.

Network	Normalize by body mass	CAL feature calibration	Overall - P. Img. MSE (kPa^2)	Overall - v2vP MSE (kPa^2)
BPWnet, 108K†			16.326	31.806
BPWnet, 108K†	×		1.393	2.713
BPWnet, 108K†	×	×	1.195	2.485
BPWnet, 108K†		×	**1.184**	**2.439**

we trained only the initial CNN for 100 epochs but used the $\mathbb{L}_{\text{BPW}_2}$ loss. This marginally changed pose, pressure, and body height error but substantially reduced the accuracy of body weight.

CAL succeeds at both scaling and locally calibrating pressure map features. Recall that the purpose of CAL is to calibrate $\mathcal{P}^+$ both by scaling it and by adjusting local features to better resemble features in the real pressure image $\mathcal{P}$. We tested the ability to achieve each of these purposes by adding a body mass normalization component to the output, which scales $\mathcal{P}^+$ to the correct pressure range. The estimated body mass from BetaNet was used for this and the results are shown in Table 4.5. Without CAL, mass normalization greatly improves the pressure inference, but not to the extent that CAL does. This indicates that CAL does more to improve the features in $\mathcal{P}^+$ than only scaling it. We also compared a network that both uses CAL and normalizes by mass, and observed a slight dip in performance.

Improvements to blanket simulation may improve performance. When training using only synthetic data (with and without blanket occlusions), pose estimation is substantially better when testing on real depth images that do not have blanket occlusions than testing on those that do. See Table 4.2 for reference. The same holds true when training on mixed synthetic and real data. On the other hand, when training using only real data (with and without blanket occlusions), pose estimation accuracy is comparable when testing on

real depth images with and without blanket occlusions. This seems to indicate that the quality of synthetic blanket occlusions could be improved.

We manually adjusted simulation parameters to achieve realistic blanket folding characteristics. Besides blanket stiffness, we found that other FleX parameters such as the number of simulation substeps had an impact on the cloth behavior. Optimizing the synthetic blanket parameters to make them behave more like real coverings may improve the quality of synthetic data and boost performance when testing on real depth images with blanket occlusions; for example, the methods from Runia *et al.* [118] might merit future exploration.

Body pressure loss may improve performance. The loss functions in BPWnet and BPBnet include terms computed at many different locations to provide better supervision. A loss directly computed based on the error in the inferred body pressure $\mathcal{P}_b$ merits future investigation. Recent differentiable geometry tools may enable such a loss computation and improve performance.

Released materials can support future work. In addition to the SLP-3Dfits human body annotations and the BodyPressureSD synthetic dataset used to train our model, we publicly release 3D synthetic mesh data for the resting human, mattress, pressure mat, and blanket. This may be useful for future work in the area of geometric learning.

4.7 Opportunities for Future Work

While BPWnet exhibits promising performance and demonstrates the feasibility of using a depth sensor to infer body pressure, further research will be required to establish its clinical effectiveness. For example, the occurrence of false positives or false negatives may limit the system's ability to detect when a pressure injury is imminent. One example of a false negative is in the top left of case Figure 4.14 (a), where there is a peak pressure region on the person's right shoulder, but the system indicates there is no pressure on the right shoulder. The ability of the current network to generalize to clinical settings is also unclear. For

example, pillows, nearby furniture, different types of bedding, medical instrumentation, and objects in bed, such as a mobile phone and other devices, would likely result in errors.

Some types of errors might be difficult or impossible to overcome with a single frame from a depth camera. For example, in the top left error example in Figure 4.14 (a), the person's elbow and knee are elevated such that they push the blanket up into a tent like shape that reduces contact between the blanket and the person's body. This reduces depth information about the body surface and the system makes errors, including neglecting pressure on the hidden leg and one side of the body.

For other types of errors, future work might achieve better performance. Figure 4.14 (a) shows examples of errors. On the top right, the sheet covering the bed folds upwards and the system mistakes it for the person's right leg. On the bottom left, the person crosses their right foot on top of their left knee and the system incorrectly estimates that the knee is on top of the foot. This leads the system to infer a peak pressure on the heel, rather than the calf. On the bottom right, the person assumes a pose that would be unusual when sleeping. The person rests their head on their left hand. The system misestimates the arm poses. Additionally, the annotation method incorrectly labels the left hand as being behind the head rather than supporting it.

Other errors relate to the body contacting itself. The network can output unnatural body part interpenetration. Figure 4.14 (b) shows an example of this, where the left hand penetrates the head and the lower legs penetrate one another. Our network uses a fixed open hand pose, which may contribute to unnatural hand penetration errors. Self penetration of the 3D mesh body models does not occur frequently in the data because a mesh interpenetration term was used to create SLP-3Dfits, nor in the synthetic data because the physics simulations prevent it. Real human bodies have soft tissues that deform when in contact, which can be approximated as 3D model interpenetration, but the network outputs interpenetration that poorly matches soft tissue deformation. The problem is worsened by the frequent self-contact of limbs and body parts when a person rests. Our system also neglects

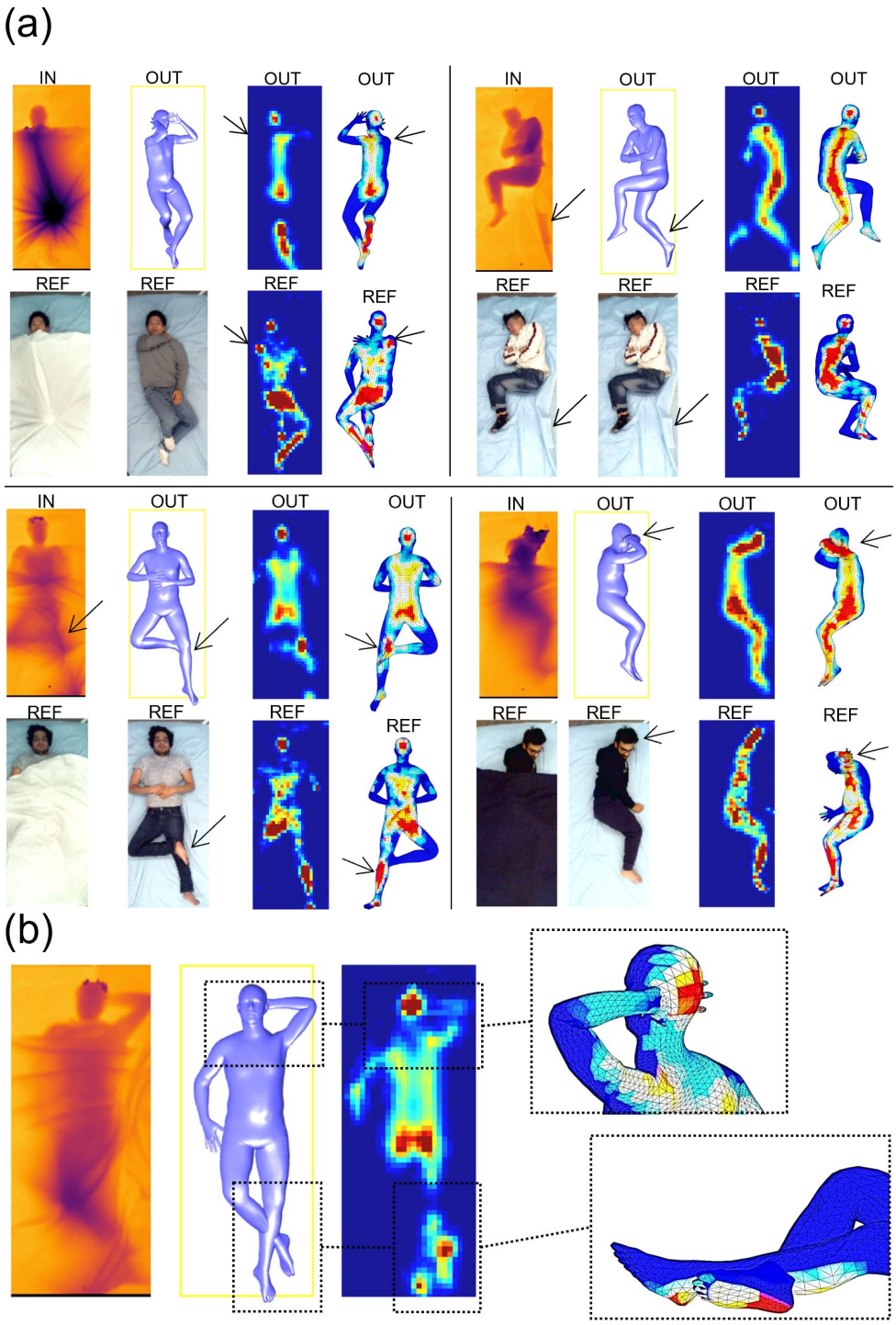

Figure 4.14: (a) Examples of errors when testing with BPWnet. (b) Limb interpenetration scenarios, also with BPWnet.

pressure due to self contact and pressure differences due to the mass of one limb resting on another limb. Better accounting for the mechanics of self-contact might improve performance [59, 119, 120, 121], and reported height and weight data from Müller *et al.* [121] may also be used to improve BetaNet.

The GPU memory footprint of BPWnet is substantial. Reducing it may allow both Mod1 and Mod2 to be trained end-to-end, simplifying learning. Recent works in human body reconstruction provide other insights for boosting performance, including body-driven attention [122], structured prediction to explicitly model joint dependencies [123], joint occupancy estimation [124], and a method to combine keypoint- and parametric model- based human pose estimation methods [125].

4.8 Conclusion

In summary, we presented a method to infer body pose and contact pressure from a depth image, which has the potential to automatically localize pressure injury risk areas using a consumer-grade depth camera. We described a method for annotating an existing human resting pose dataset with 3D body models, which we use for initializing a fast physics simulator and training and testing deep models. We generated a large synthetic resting pose dataset using physics simulations, which substantially boosts performance of our deep models. We introduced two deep learning models and compared their performance. The models were able to to accurately infer pose and contact pressure and outperform state-of-the-art methods for pose inference, even in the presence of visual occlusion from blankets.

CHAPTER 5
CONCLUSION

5.1 Key Takeaways

The previous sections discuss some important findings (subsection 2.3.4, section 3.5, and section 4.6). Here, we state the most important takeaways from the dissertation as a whole.

Pressure mats are robust to visual occlusion and can be used to infer 3D human pose and body shape, despite their relatively low spatial resolution. Pressure mats are robust to visual occlusion because they sense haptic information instead of visual information. The threshold for measuring pressure on the mat is above the amount that standard bedding materials exert when placed on the bed, so complex blanket geometries and lightweight objects placed on the bed would not affect inference. Pressure mats are also unaffected by overbed tables such as the one in Figure 2.11, which are common in a healthcare setting. We presented a series of two methods to infer 3D human pose from a pressure image. Although the input image has a relatively low spatial resolution of 64×27, it can be used to infer detailed human representations in 3D. The first of these pose estimation methods, presented in chapter 2, represents human pose using a kinematic skeleton with variable link lengths that adjust for people of different sizes. The second of these methods, presented in chapter 3, infers pose and body shape represented with a SMPL human body model. We provided compelling qualitative evidence of our method's accuracy in Figure 2.7, Figure 2.9, Figure 3.1, and Figure 3.9. We further support these claims through quantitative analyses of data in Table 2.1, Table 3.2, and Table 3.3.

Body parts that are not in contact reduce performance, and this is supported by our double inverted pendulum model. Inference suffers on parts of the body that are not in contact with the mat. Specifically, the overall accuracy decreases as the threshold

for discarding higher variance joints increases (Figure 2.10), and high variance measured by Monte Carlo Dropout is indicative of body parts being out of contact. We identified a scenario where substantial movement of a limb out of contact had little effect on the pressure measured by the mat. A change in pressure would be necessary to infer a change in body pose. This type of ambiguity can be explained by the double inverted pendulum model in Figure 2.5, where distinct configurations of the pendulum arm exert a moment and a pressure distribution that are not unique.

Random sampling with ragdoll physics generates bodies at rest with enough fidelity to train deep models that perform well with real data. Resting poses are characterized by a lack of physical exertion and an absence of motion, which we modeled with a ragdoll human body model dropped onto a soft bed in simulation (section 3.2 and section 4.3). Because the ragdoll model is articulated, the joint angles adjust as the body moves into a statically stable resting pose. We used this to collect large synthetic human resting pose datasets, and then trained deep models to infer human pose at rest. These deep models infer human resting pose using pressure imagery (section 3.3) and depth imagery (section 4.4), and are evaluated on the PressurePose real dataset [11] and on the SLP real dataset [14], respectively. In both cases, models trained using synthetic ragdoll poses generalized well to data capturing poses with real people resting.

Computationally efficient physics engines can model pressure sensing arrays on a soft bed that resemble their real counterparts closely enough to train models entirely with synthetic data. We used FleX [8], a computationally efficient particle-based simulator, to model a soft mattress and a pressure sensing mat on the surface of the mattress. We modeled the mattress as an rectangular prism filled with soft particle clusters, and the pressure sensing mat as an organized array of two stacked meshes of cloth connected by spring constraints (subsection 3.2.1). The mattress has parameters to control softness and the pressure mat has parameters to both control softness and adjust our model of pressure as a function of particle configuration. We introduced an optimization method to calibrate

these parameters so that the simulated mattress and pressure mat behave as closely as possible to the real mattress and pressure mat (subsection 3.2.3). Then, we generated a set of synthetic pressure images complementing the synthetic ragdoll poses, forming a collection of over 200,000 input-output pairs. We trained PressureNet (section 3.3) entirely with this synthetic data, and showed that it generalized well to real pressure image data.

Depth imagery can be used to infer human pose, body shape, and contact pressure underneath the person – even when the person is occluded by a blanket. An overhead depth camera can provide detailed information about the person resting in bed, even when the person is occluded by a blanket. The profile of features across the blanket surface is a good indicator of where limbs are in many cases, so the key challenge is to extract the features and convert them into a useful representation of the body. Our networks are able to do this, and they are also able to infer the contact pressure underneath the person on the surface of the bed mattress (chapter 4). The contact pressure may be projected on the body to determine its magnitude at specific locations on the body. We compare our pose inference accuracy to a state-of-the-art method that uses four modalities (RGB, depth, thermal and pressure imagery) to infer pose in the presence of blankets. Ours only requires a depth image, yet it is 12% more accurate.

Embedded geometric models of pressure and depth map reconstruction improve performance of our deep models. Our work introduces methods to reconstruct pressure and depth maps through differentiable geometry, using no learned parameters. These are more parsimonious and interpretable than learned reconstructions, and they are also modular. The modularity allows them to be interchanged within neural network architectures or other systems that would benefit from gradient computation. In our method of inferring pose and body shape from pressure (chapter 3), we introduced PMR, a method to differentiably reconstruct pressure maps from the inferred SMPL mesh. PMR measures the distance that the SMPL mesh sinks into the underlying bed and is used to promote consistency between the input and the output. When we ablated PMR, performance degraded

(Table 3.2). In our method of inferring pose, body shape, and contact pressure from depth (chapter 4), we introduced DMR, which differentiably reconstructs depth maps. DMR computes depth maps based on the distance between the overhead camera and the top surface of the SMPL mesh, and is similarly used to promote consistency on the inference. Our network that uses DMR also uses PMR so it can learn contact pressure from depth imagery. We compared this to a variant with learned depth and pressure map reconstruction modules (i.e. no DMR or PMR) and found that performance degraded overall (Table 4.2 and Figure 4.13).

Computationally efficient physics engines can model blankets and depth imagery with enough fidelity to generate data that greatly boosts performance of deep models. In chapter 4, we added a feature to simulate cloth blankets covering the body at rest. We covered the simulated bodies with many randomized blanket configurations in effort to mimic real world variety. Then we simulated an overhead pinhole depth camera facing the bed from the same perspective as a camera mounted on a real bed, and generated a large synthetic dataset. When inferring pose and body shape from depth imagery, models trained only on the large synthetic dataset (97K samples) perform comparably to models trained only on a smaller real dataset (11K samples). However, when the real data is mixed in randomly with the synthetic data, performance improves by more than 30% (chapter 4). This implies there is some reality gap in our synthetic data, but it still has great benefit.

5.2 Demographic Limitations

Body shape is influenced by a wide variety of factors, including gender, age, ethnicity, and medical conditions. For example, body fat distribution changes with age [126], bone geometry differs based on ethnicity [127], and an amputee could be missing a limb. The generalization of our methods to diverse body shapes is limited by the human representations we used and the demographics represented in the real datasets we evaluated our methods with. In this section we present demographic information that could lead to bias

in our methods.

SMPL. Most of this dissertation represents the body using SMPL [10], a skinned vertex-based model that represents a variety of body shapes in natural human poses. The SMPL body shape was learned from the CAESAR dataset [128], which consists of 3D scans covering 2000 unique body shapes for each male and female gender. Besides the CAESAR scans, the SMPL mesh is also a function of pose-dependent deformations, which were captured by scanning an additional set of 40 people (20F / 20M).

The CAESAR scans reflect the body shape of people aged 18-65 in the United States, the Netherlands, and Italy. The mean weight for females was 68.9 ± 17.6 kg, and for males was 86.2 ± 17.9 kg. The mean height for females was 1.640 ± 0.073 m, and for males was 1.778 ± 0.079 m. For the United States, 1824 people identified as white, 263 as black, and 288 as other. For the Netherlands, 1024 people identified as Dutch and 231 as other. For Italy, 764 people identified as Italian and 32 as other. Further information can be found in the CAESAR technical report. The SMPL model approximates these differences in shape using 10 shape component parameters, which do not represent the full population in CAESAR. This work in this thesis used this 10 component model, but a more comprehensive set of 300 SMPL shape components are available that better represent the CAESAR dataset. Finally, the CAESAR dataset and SMPL model are normative. They do not capture children, people who are very old, or people who are pregnant. They also do not capture people with medical conditions, such as dwarfism, Down's, cerebral palsy, amputations, etc [129].

IROS 2018 data. We collected motion capture marker and pressure image data to evaluate our methods in chapter 2. This data includes poses with 17 able-bodied people (6F / 11M) in a lab setting, aged 19-32, who ranged 1.57-1.83 m in height and 45-94 kg in weight. The subjects were recruited from a pool of undergraduate and graduate students at Georgia Institute of Technology. We validated the network using 7 subjects (2F / 5M) and tested the network with leave-one-subject-out cross validation on the remaining 10 subjects (4F / 6M).

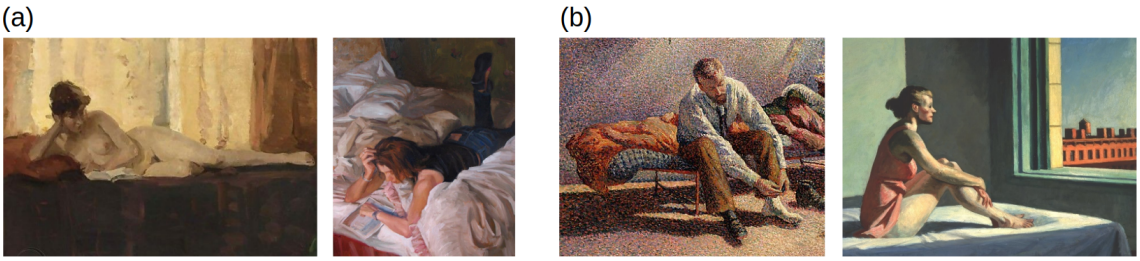

Figure 5.1: Paintings depicting people in resting poses that are not represented in our synthetic datasets. (a) Resting poses where the head is supported by the arm. (b) Examples of sitting in bed.

PressurePose data. We collected and used the real PressurePose dataset to evaluate our methods in chapter 3, which consists of RGB images, depth images, point clouds, and pressure images. This data includes poses with 20 able-bodied people (10F/ 10M) in a lab setting, aged 18+, who ranged 1.52-1.98 m in height and 42-102 kg in weight. The subjects were recruited from a pool of undergraduate and graduate students at Georgia Institute of Technology. 11 participants self-identified as white, 8 as Asian, and 2 as mixed ethnicity. We used all 20 subjects for testing.

SLP data. We used the SLP dataset to evaluate our methods in chapter 4. The SLP dataset includes poses with 102 people (28F / 74F) in a home setting. The subjects were recruited from a pool of undergraduate and graduate students at Northeastern University. Following Yin *et al.* [53], we split the data into 80 training (25F / 55M) and 22 testing (3F / 19M) subjects. The training subjects range 1.48-1.84 m in height and 44.6-105.1 kg in weight and the testing subjects range 1.51-1.81 m in height and 45.2-81.5 kg in weight.

5.3 Guide to Future Work

The previous sections discuss failure cases to show shortcomings of our existing work and provide some guidance to future work. We refer the reader to Appendix A.8, Appendix B.2 and section 4.7, and provide additional suggestions here.

Sampling resting poses. Generalization of a trained network to unseen data depends on how well the training data distribution represents the unseen data. While our ragdoll model is good for sampling many resting poses, there are some that it misses. For example,

people commonly rest their head on their hand to support the head (Figure 5.1(a)). Our synthetic data generation method did not produce such 'supporting' poses, either from random uniform initial pose sampling (section 3.2) or by initial pose sampling from the SLP dataset (section 4.3). This phenomena is also largely absent from our testing data, so an evaluation of it would require a new testing dataset. Future work may explore how to tune the existing model or use a different kind of model to produce these poses.

We only considered laying poses in bed, and it is unclear how well the ragdoll body model could be used for seated postures in bed (Figure 5.1 (b)), or sitting in general. Further, the simulation environment is only constructed to model poses on a flat bed. Many beds in healthcare such as Autobed [22] from chapter 2 are configurable, so developing the simulator to account for this would allow the methods to generalize better. We encourage work in these directions.

Demographic representation. Our models contain bias that is influenced by the demographics of people in the data used for this work. When deploying methods like ours in a real world setting, it will be important to evaluate whether they generalize to the demographics of the specific population of interest. We encourage future researchers to develop human models like SMPL that represent a wider range of ages, ethnicities, and body geometries. Similarly, we encourage researchers to evaluate related methods on larger datasets where the sample size is sufficient to test for bias between demographic groups represented in the data.

Disability-specific human model. Adapting the human body representation to include people with non-normative body shapes is a promising area of future work. It may be possible to build a statistical human model like SMPL that generalizes within specific conditions. Other body anatomies may be too specific for this (e.g. a limb amputated at a precise location), and would instead benefit from rapid 3D scanning of the individual and shape fitting prior to body pose estimation.

Modeling deformable human tissue. Humans at rest have substantial human-environment

contact, and also often have self-contact between body parts. Our simulation of bodies at rest captures this contact, but only to the extent that the capsulized body and meshed SMPL body are capable of modeling. The capsulized body has a relatively coarse profile of features that model resting pose with rigid collisions, and the meshed SMPL body in the FleX simulation has a homogeneous composition of soft particles. A model such that captures the heterogeneous properties of both deformable human tissues could enable more accurate and/or more expressive machine perception methods. A number of relevant works have published in this area, such as SoftSMPL [130] and others that model human tissue deformation across the surface of the body [131, 132].

Complex bedding materials. In the real world, people receiving care in bed are often surrounded with more materials than just a mattress, blanket, and thin sheets that our work models. For example, thicker blankets and cushions may be placed under the person to improve comfort. These can distort the pressure measured by a pressure mat. They also may be placed on top of the person or extend from underneath the person and be visible from a depth camera. A standard pillow is perhaps the most common example of this. Modeling these objects would be important for transferring models to complex real healthcare settings.

Multimodal sensing and inference. We encourage continued work on fusing multiple modalities in a deep network [53]. Some applications may require highly accurate estimates of the body pose that approach the limit of what is possible using an overhead camera or pressure mat. In the context of robotic assistance in bed, an eye-in-hand camera on the robot end effector could enable a final stage of body pose refinement. In contrast to the overhead view (~ 5 feet away), it would capture a substantially higher resolution on a local area of interest on the body.

We suggest combining a camera-based (e.g. RGB) estimate of the person's head pose using existing methods [133, 134, 135] with depth or pressure based inference for the remainder of the body. In bed, the head is usually visible with an overhead camera and its

pose could be estimated with high accuracy to boost overall performance. We also suggest exploring synthetic data generation of other modalities. This is an active research area, with many recent works focused on generating synthetic RGB imagery [69, 70, 136, 137, 138, 139, 140] and one work on thermal imagery, which introduces a heat transfer model for human limbs under blankets in bed [52].

Robot assistance. Our demonstration in section 1.1 showed the robot moving to diverse locations on the human body, but the methods in this dissertation may be useful for more complex interactions around the person in bed. For example, commanding the robot to move between two coordinates on the surface of the body may be used for wiping motions to give a person a bed bath or apply lotion on the skin.

Projecting the pressure from a real pressure mat (chapter 2 and chapter 3) or from the inferred pressure (chapter 4) onto the body may provide information to the robot about the feasibility of reaching to specific locations. If the pressure is high at a location on the skin, the robot would not be able to reach it successfully unless it physically moved the person's body or limb.

5.4 Final Remarks

Overall, this dissertation explored models of humans at rest and how the body shape and pose of people resting could be inferred from sensor data. We found that modeling people at rest with an articulated ragdoll body could map randomized human poses into a smaller space of resting poses. This model, when combined with soft body simulations of surrounding materials such as a mattress with blankets, was useful for generating synthetic data that resembled real data. We used this data to train deep networks to infer human attributes such as body pose, and found that they transferred well to real world data. We suggested applications for our methods in healthcare and provided a demonstration with a robot that reached to diverse positions on a person in bed. Finally, we hope that this work will encourage future research to understand the physical characteristics of people

in a healthcare setting and to enable robots to provide meaningful assistance to caregivers, people with motor impairments, and older adults.

Appendices

APPENDIX A

PRESSUREPOSE DATA GENERATION

A.1 Initial Pose Sampling

We use rejection sampling to generate initial pose dataset partitions. Our criteria are as follows.

Uniform Cartesian space distribution - Figure A.1 (a). We use rejection sampling to uniformly sample poses with respect to the Cartesian space, by discretizing the space and ensuring that a given limb is equally represented in each unit. We define a Cartesian space $\boldsymbol{\mathcal{Y}}$ as a cuboid for checking for presence of the most distal limb. First, we constrain $\boldsymbol{\mathcal{Y}}$ in the (x, y) directions to how far the distal joint (e.g. right foot, $\boldsymbol{s}_{r.foot}$) can extend from the promixal joint (e.g. right hip, $\boldsymbol{s}_{r.hip}$) in a limb. For the legs, we assume that the foot cannot move above the hip. For the right leg, these constraints can be summarized as: $s_{r.foot,x} \in [s_{r.hip,x} - l_{leg}, s_{r.hip,x} + l_{leg}]$ and $s_{r.foot,y} \in [s_{r.hip,y}, s_{r.hip,y} + l_{leg}]$. We also constrain the z direction to ensure that the distal joint is initially positioned at a height close to where the proximal joint is: For laying poses, the distal joints (feet and hands) are more likely to end up close to the surface of the bed than very high in the air, for example. This constraint promotes simulation stability and decreases the time it takes for physics simulation #1 (Figure 3.2) to reach an equilibrium state. We constrain $s_{r.foot,z} \in [s_{r.hip,z} - 10cm, s_{r.hip,z} + 10cm]$.

Next, we break up $\boldsymbol{\mathcal{Y}}$ into a set of smaller cuboids as shown in Figure A.1-top middle. For each limb we uniformly sample a cuboid from $\{\mathcal{Y}_1, ...\}$ and then use rejection sampling on the limb joint angles — in the case of Figure A.1 (a), the right leg — until $\boldsymbol{s}_{r.foot} \in \mathcal{Y}_4$.

Generate common posture partitions - Figure A.1 (b). Some common postures, such as resting with the hands behind the head, are unlikely to be generated when the joint

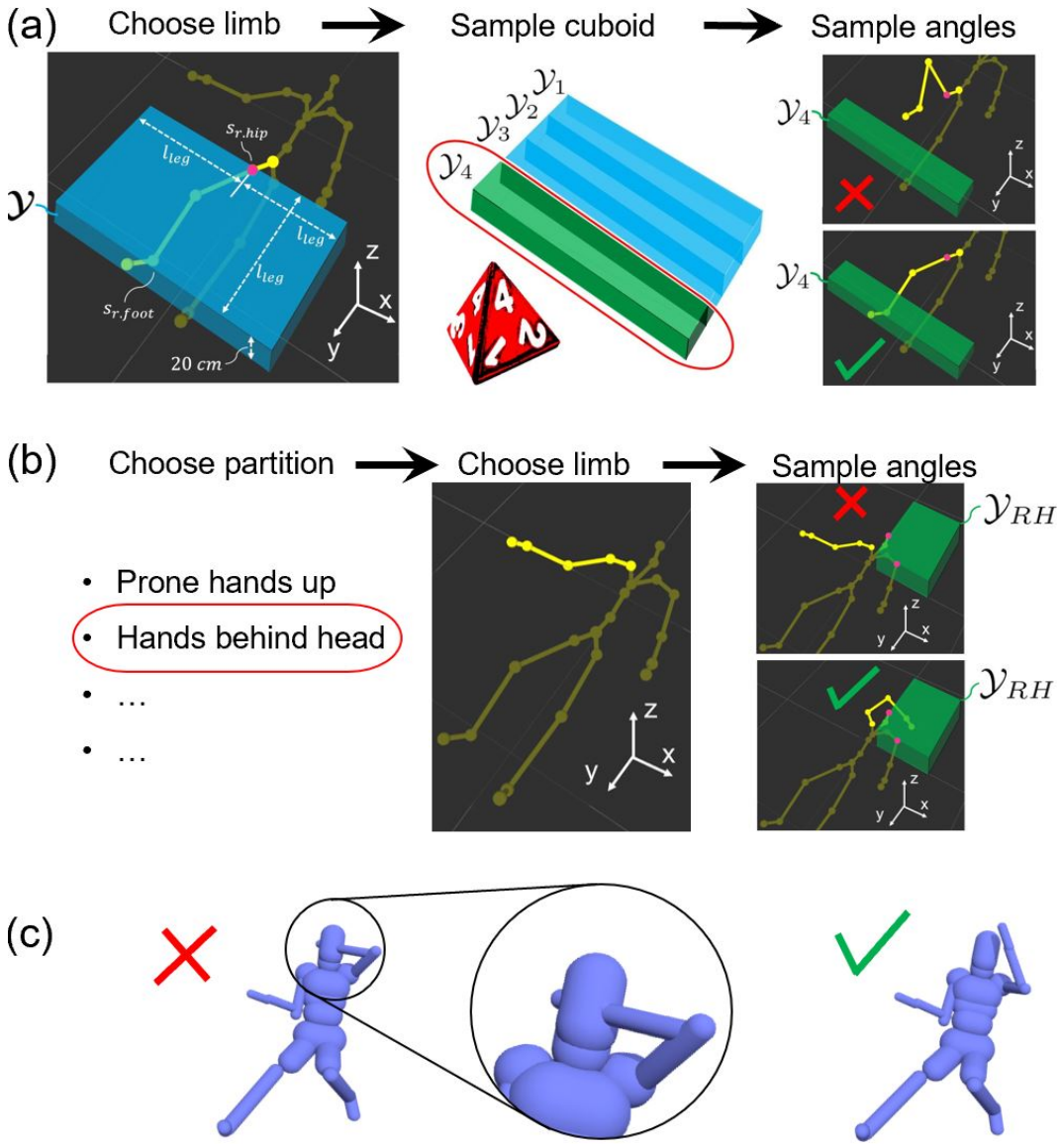

Figure A.1: Rejection sampling criteria. (a) Evenly distributing right leg poses across Cartesian space by sampling from four non-overlapping Cartesian cuboids, $\{\mathcal{Y}_1, \mathcal{Y}_2, \mathcal{Y}_3, \mathcal{Y}_4, \} \in \boldsymbol{\mathcal{Y}}$. Reject pose angles if $\boldsymbol{s}_{r.ankle} \notin \mathcal{Y}_4$ (b) For sampling right arm in the hands-behind-head partition, we reject the right arm pose angles if $\boldsymbol{s}_{r.hand} \notin \mathcal{Y}_{RH}$. (c) Pose feasibility checking via collision detection. [11]

angles are sampled from a uniform distribution. For example, there is a $< 1\%$ probability of generating a pose with the hands-behind-the-head when sampling joint angles uniformly, so a network trained with such little hands-behind-the-head data has difficulty learning such a pose. We mitigate this issue by checking for presence of the most distal joint in a cuboid representing where it would be located in such as pose. If the joint is within the cuboid, e.g. $\boldsymbol{s}_{r.hand} \in \mathcal{Y}_{RH}$, the joint passes the criteria and we add the limb pose to the set of checked initial poses.

Prevent self-collision - Figure A.1 (c). We reject poses that result in self collision by capsulizing the mesh and using the DART collision detector. We check the hands, forearms, feet, and lower leg capsules for collision with any other capsules except their adjacent capsules (e.g. forearm and upper arm should overlap).

A.2 Dynamic Simulation Details

Weighting particles in FleX. We directly calculate particle mass for the particlized human in physics simulation #2, as well as for the particlized calibration objects depicted in Figure 3.6 (b). Since FleX is a position-based dynamics simulator and the mass is defined by units of inverse mass $1/m$ on an arbitrary scale, we begin by defining the inverse mass scale for particles in the particlized human.

For this, we assume that the volume each particle in the human takes up, as well as the density of particles, is the same for that of water. Because volume and density are equal, we also can set inverse mass equal, so $1/m_H = 1$, thus $1/m_{H2O} = 1$.

We calculate the inverse mass for particles in calibration objects by a density ratio to that of water, given a known weight of the object w_k and the object volume V_o:

$$\frac{1}{m_{o,k}} = \frac{1}{m_{H2O}} \frac{\rho_{H2O}}{\rho_o} = \frac{\rho_{H2O}}{m_{H2O}} \frac{V_o}{w_k/g} \tag{A.1}$$

where ρ_{H2O} is the density of water and g is gravity. In contrast to the humans and objects

rested on the bed, the the soft mattress and synthetic pressure mat particle inverse mass are determined from an optimization described in Appendix A.4.

Weighting the capsulized human chain. We compute a per-capsule weight for the articulated capsulized chain in DartFleX based on the weight distribution for an average person and capsule volume ratios. First, we describe how we assign capsule mass for the average person. We use average body mass and mass distribution values from Tozeren [79], and calculate capsule volumes from body shape. We assume the average human of gender $g \in \{M, F\}$ has a mass of $\bar{m}_g$, mass percentage distribution for body part R of $\bar{X}_{R,g} \in \bar{\boldsymbol{X}}_g$, and SMPL body shape parameters $\bar{\boldsymbol{\beta}}_g = \mathbf{0}$. We define the mass of each capsule c in an average person to be:

$$\bar{m}_c = \bar{m}_g \bar{X}_{R,g} \frac{\bar{V}_{c,g}}{\bar{V}_{R,g}} \tag{A.2}$$

where $\bar{V}_{c,g}$ is the volume of capsule c for a mean body shape $\bar{\boldsymbol{\beta}}_g$, and $\bar{V}_{R,g}$ is the sum of volumes for all capsules in body part R. Now, we describe how this capsule mass can be converted into masses for people of other shapes. To find the mass of some capsule c for a body of particular shape $\boldsymbol{\beta}$, we use a capsule volume ratio between the particular person and an average person:

$$m_c = \bar{m}_c \frac{V_c}{\bar{V}_{c,g}} \tag{A.3}$$

where V_c is the volume of some arbitrary capsule. Computing capsule volume analytically is simple given radius and length, but this is complicated by capsule overlap, which is often substantial in the SMPLIFY capsulized model [35] we use. Instead, we use discretization to compute capsule volume and correct for overlap. First, we use the SMPLIFY regressor to calculate capsule radius and length from body shape $\boldsymbol{\beta}$. Besides shape, overlap is dependent on the particular pose of the capsulized model. We assume that pose dependent differences in overlap are very small, and set the pose constant at $\boldsymbol{\Theta} = \mathbf{0}$. We then

compute the global transform for each capsule using this shape and pose. From capsule radii, lengths, and global transforms, we place all capsules in 3D space and voxelize them with a resolution of $2mm$. This produces a set of 3D masks, which are tagged to their corresponding capsules. Voxels belonging to a unique capsule are allocated directly, while voxels belonging to multiple capsules are allocated fractionally based on the number of capsules sharing the voxel. We compute capsule mass inertia matrices analytically from capsule radius and length.

Capsulized body joint stiffness. For an average person, we set the following joint stiffnesses for the shoulders, elbows, hands, hips, knees and feet to low stiffness: $\bar{k}_{\theta,shd} = 4$ Nm, $\bar{k}_{\theta,elb} = 2$ Nm, $\bar{k}_{\theta,hnd} = 4$ Nm, $\bar{k}_{\theta,hip} = 6$ Nm, $\bar{k}_{\theta,knee} = 3$ Nm and $\bar{k}_{\theta,feet} = 6$ Nm. We set torso and head stiffness very stiff $\bar{k}_{\theta,trs}, \bar{k}_{\theta,hd} = 200$ Nm. For a person of particular body shape, we weight joint stiffnesses $\boldsymbol{k}_\theta$ by the body mass ratio, where $\boldsymbol{k}_\theta = (m/\bar{m})\bar{\boldsymbol{k}}_\theta$. We set joint damping $\boldsymbol{b}_\theta = 15\boldsymbol{k}_\theta$. The direction and magnitude of stiffness force on each joint is dependent on joint equilibrium position, i.e. the joint angle where force is 0. We set the equilibrium position of the joints to be the *home pose*, where the arms are at the sides and the legs are straight. In the SMPLIFY model, *home pose* consists of equilibrium joint positions $\boldsymbol{\Theta}_{eq}$ set to 0, except the shoulders, which are bent downward at 90 degrees. Rather than set $\boldsymbol{\Theta}_{eq}$ to initial joint angles $\boldsymbol{\Theta}_C$, we do this to guide the pose away from extreme angles at a modest force.

Because we set the joint stiffness low, our dataset does not capture non-resting postures, such when a person is getting in/out of bed (recall Table 3.1). However, we have been able to generate resting sitting poses by bending the mattress and pressure mat into a sitting configuration and then resting a person on it, like the sitting postures in [9].

Settling criteria - Physics simulation #1. For physics simulation #1, the goal is to slowly allow the body to fall on the bed and settle into a resting pose. We start the capsulized body at a height based on the lowest point on the body. For many randomly sampled poses, the lowest joint is initially much lower than the center of mass, which causes the cen-

ter of mass to build significant momentum by the time it reaches the bed. We found that this caused bouncing and instability, and was qualitatively different from the motion one might take to assume a resting pose in bed. We alleviate this issue by zeroing the velocity of the capsulized model every 4 iterations in the simulation ($\sim 0.04s$) until a capsule that better represents the center of mass contacts the surface of the bed. For this, we use the capsule approximating the buttocks.

Finding a resting pose in static equilibrium is hampered by the stability of DartFleX: DART uses a more traditional physics solver and FleX uses position-based dynamics, which are challenging to connect in a stable loop. Rather than run the simulation until static equilibrium, we use a cutoff threshold that takes velocity and acceleration of all capsules into account. We define a resting body as that when the maximum velocity of all capsules has reached $v_{max} < 0.05m/s$ and maximum acceleration has reached $a_{max} < 0.5m/s^2$. In the event the model does not settle within 2000 iterations or the pressure array becomes unstable (defined by separation of particles in the pressure mat, e.g. limb poking into mat), the simulation is terminated and the particular $\mathbf{\Theta}_C$ is rejected. Across the whole dataset, we found roughly a 10% rejection rate for both of these criteria.

Settling criteria - Physics simulation #2. We use the same approach as simulation #1 to determine the height to drop particlized humans. We found it to always be stable for our purpose, and it took roughly 150 iterations to reach the same resting velocity and acceleration previously stated. Because it only uses FleX and the limbs do not move kinematically, it is an order of magnitude faster to run and provides greater flexibility to determine settling criteria. We ran simulation #2 for a minimum of 200 iterations and terminated it once the velocity and acceleration thresholds of the particlized human, $v_{ptcl} < 0.05m/s$ and $a_{ptcl} < 0.5m/s^2$, were reached. In almost all cases, 200 iterations was sufficient.

Computation time. For both physics simulations, we ran 10 parallel simulation environments on a computer with 32 cores and a NVIDIA 1070-Ti GPU. This allowed us to generate roughly 35,000 labeled synthetic pressure images per day.

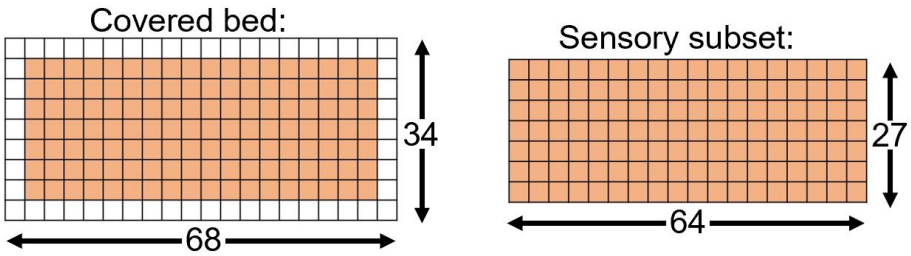

Figure A.2: Size of synthetic pressure mat. Physics simulation #1 uses forces from particles on the entire covered bed. The pressure mat calculated in physics simulation #2 uses a smaller subset representing the size of the real pressure mat. [11]

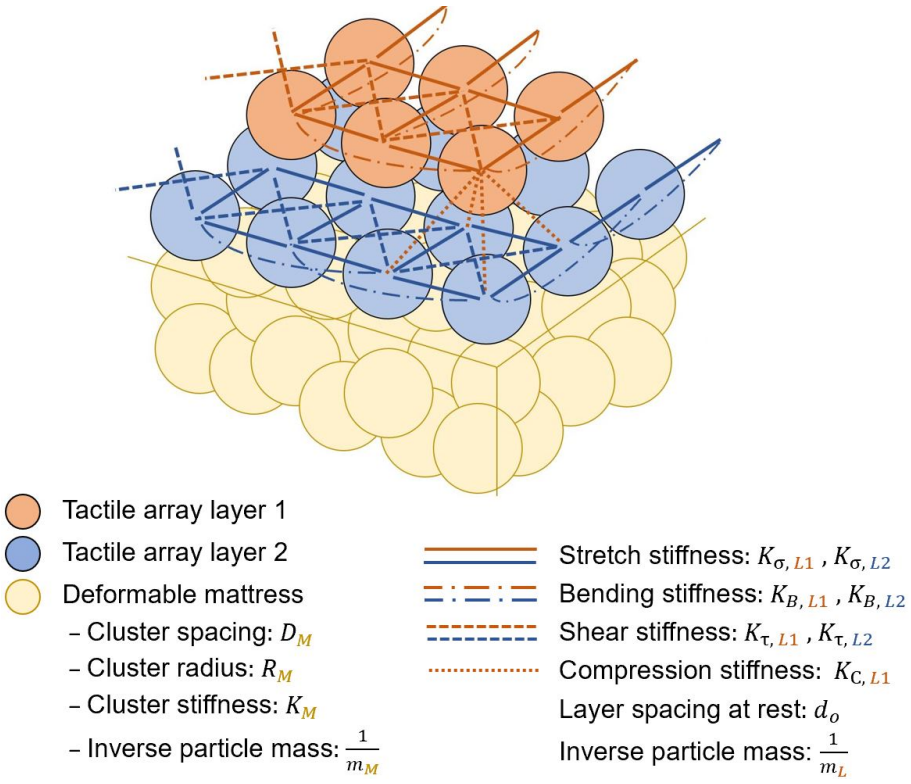

Figure A.3: Pressure mat pyramidal structure showing FleX parameters that we optimized using CMA-ES. [11]

A.3 Pressure Mat Structure Details

Limited pressure sensing area. The sensing portion of the real pressure mat does not cover the entire mattress. We measured a non-sensing border of 6 cm on the sides of the bed and 9 cm at the top and bottom. We built the simulator in the same way: the synthetic pressure mat covers the entire bed (68 x 33), but only an inner subset (64 x 27) representing the sensing area of the pressure image array is recorded, as depicted in Figure A.2.

FleX spring constraints. FleX particles in the synthetic pressure mat are bound together by stiffnesses shown in Figure A.3.

Pressure mat adhesion. For the real pressure mat, velcro and tape are used to prevent sliding across the bed. For the synthetic pressure mat, particles are fixed in horizontal directions across the bed.

A.4 FleX Calibration

Although FleX is able to simulate soft bodies, FleX is not optimized to model real-world physics or to calculate realistic pressures. To optimize our FleX simulation to match the real-world mattress and pressure mat, we place a set of static objects on the real mattress, and record the resulting pressure images from the pressure mat. We then build a similar environment in FleX, and we optimize FleX parameters such that the simulated and real-world measurements closely align.

We jointly optimize 16 deformable bed and pressure sensing array parameters $\mathcal{S}$ using CMA-ES [141]. These include the 13 FleX parameters in Figure A.3, including 4 soft mattress parameters, 7 pressure array stiffnesses, spacing between the pressure mat layers and particle inverse mass, as well as quadratic taxel force constants C_1, C_2, and C_3. To optimize, we first place a set of real rigid objects $\{\mathbb{O}_1, \ldots \mathbb{O}_J\}$ each with weights $\{w_1, \ldots w_M\}$ on the real bed. Figure 3.6 (a) depicts $\{\mathbb{O}_1, \ldots \mathbb{O}_J\}$, where $J = 4$ and we use capsular objects with 5 weights for each: 1.3, 2.3, 4.5, 9.1 and 14 kg on the shorter capsules (L=20

cm), and 1.3, 4.5, 9.1, 14 and 18 kg on the longer capsules (L=40 cm). We then collect real pressure mat images $\{\mathbb{P}_{1,1}, \ldots \mathbb{P}_{J,M}\}$ and measure the distance that the mattress compresses normal to the bed surface in centimeters, $\{\mathbb{q}_{1,1}, \ldots \mathbb{q}_{J,M}\}$.

Next, we build a matching set of simulated capsules $\{o_1, \ldots o_J\}$ in FleX with the same weights, where one of these objects is shown in Figure 3.6 (b). At each iteration of the optimization, we drop J simulated capsules of each M weights onto the FleX mattress, re-compute the synthetic pressure images, and compare them to the real ones. The loss function for our optimization takes as input simulated and real pressure images and is computed as:

$$\arg\min_{\mathcal{S}} \sum_{o=1}^{J} \sum_{k=1}^{M} \left(\mathcal{L}_{k,o}^{F} + \mathcal{L}_{k,o}^{C} + \mathcal{L}_{k,o}^{Q} \right) \tag{A.4}$$

with terms for force error in the pressure mat, $\mathcal{L}_{k,o}^{F}$, contact locations on the pressure mat, $\mathcal{L}_{k,o}^{C}$, and amount of mattress compression by the object, $\mathcal{L}_{k,o}^{Q}$. For some real object $\mathbb{o}$ with weight k resting on a soft bed at depth $\mathbb{q}$ from the unweighted height of the soft bed, a pressure image $\mathbb{P}$ measures forces on individual taxels $\{\mathbb{u}_1 \ldots \mathbb{u}_T\}$, where contact is a binary vector $\{\mathbb{c}_1 \ldots \mathbb{c}_T\}$ indicating which taxels are measuring non-zero forces. The upper limit T is a spatial index indicating the number of taxels on the pressure image. We note that the value of T for these calibration images is roughly equal to a fraction of the pressure mat size, $(64 \times 27)/5$, because we drop multiple objects simultaneously to speed up the optimization. Similar to the real mat, the values for the simulated environment are computed as u_i, c_i, and q. The loss terms are computed as:

$$\mathcal{L}_{k,o}^{F} = \frac{1}{2} \frac{\sum_{i=1}^{T} |u_i - \mathbb{u}_i|}{\sum_{i=1}^{T} (u_i + \mathbb{u}_i)} + \frac{1}{2} \frac{|\sum_{i=1}^{T} (u_i - \mathbb{u}_i)|}{\sum_{i=1}^{T} (u_i + \mathbb{u}_i)} \tag{A.5}$$

$$\mathcal{L}_{k,o}^{C} = \frac{1}{2} \frac{\sum_{i=1}^{T} |c_i - \mathbb{c}_i|}{\sum_{i=1}^{T} (c_i + \mathbb{c}_i)} + \frac{1}{2} \frac{|\sum_{i=1}^{T} (c_i - \mathbb{c}_i)|}{\sum_{i=1}^{T} (c_i + \mathbb{c}_i)} \tag{A.6}$$

$$\mathcal{L}^{Q}_{k,o} = \frac{|q - \mathbb{q}|}{|q| + |\mathbb{q}|} \tag{A.7}$$

The first term for both $\mathcal{L}^{F}_{k,o}$ and $\mathcal{L}^{C}_{k,o}$ account for errors in pressure measurements between individual taxels between the real and simulated pressure mats. The second term accounts for errors in the total measured pressure under an object. All terms are normalized. Since the distances q and $\mathbb{q}$ are signed, we take the absolute value in the denominator of $\mathcal{L}^{Q}_{k,o}$ for normalization.

CMA-ES implementation. To optimize the FleX environment with CMA-ES [141], we used a population size of 50, max iterations of 3000, max function evaluations of $1e+8$, mean learning rate of 0.25, function tolerance of $1e{-}3$, function history tolerance of $1e{-}12$, x-change tolerance of $5e-4$, max standard deviation of 4.0, and stagnation tolerance of 100. We used a machine with 8 cores and a Nvidia 1070-Ti GPU, and the optimization took 6 days.

Various combinations of parameters result in simulation instability. We perform a constrained optimization by placing a high cost on the evaluation function, `f_eval`, when a parameter is suspected of causing instability.

- Negative FleX parameters can cause instability. If any negative FleX parameter is proposed, a high `f_eval` is assigned.
- Large differences between K_σ, K_B, K_τ (see Figure A.3) causes knotting in the simulated array. If any stretch, bending, or shear stiffness value is outside of the range $0.5 < K < 2.5$, we add $10x$ the deviation from this range to the `f_eval`.
- An unusually long simulation time step indicates instability in the parameters. In this event, the particular rollout is terminated and a high `f_eval` is assigned.
- If an object takes too long to settle, the rollout is terminated and a high `f_eval` is assigned.

A.5 DartFleX Calibration

The purpose of this calibration is to calibrate the force that should be applied to a DART capsule from particle penetration on the FleX pressure mat. This enables the two simulators to be connected through a mass-spring-damper model, which we described in subsection 3.2.2 in the main paper.

We begin with an optimized FleX environment (Appendix A.4) and calibrate the spring coefficient k, from the mass-spring-damper model. We calibrate k so that the *dynamic* collision geometries displace the FleX mattress in the same way that real objects would. We take the same set of real objects from the FleX calibration of various shapes $\{\mathbb{o}_1, \ldots \mathbb{o}_D\}$ and weights $\{w_1, \ldots w_D\}$, where $D = 20$, place them on the real mattress, and measure the mattress displacement $\{\mathbb{q}_1, \ldots \mathbb{q}_D\}$. Then, we recreate the objects as collision geometries $\{\tilde{o}_1, \ldots \tilde{o}_D\}$ in FleX, displace the FleX mattress by $\{\tilde{q}_1, \ldots \tilde{q}_D, \} = \{\mathbb{q}_1, \ldots \mathbb{q}_D\}$, and record the sum of particle penetration distances of underlying taxels $\big\{ \sum_{i=1}^{P} \boldsymbol{x}_{i,1}, \ldots \sum_{i=1}^{P} \boldsymbol{x}_{i,D} \big\}$. We compute k as the average k across D objects:

$$k = \left(\frac{w_1}{\sum_{i=1}^{P} \boldsymbol{x}_{i,1}} \bigg|_{\tilde{q}_1} + \ldots + \frac{w_D}{\sum_{i=1}^{P} \boldsymbol{x}_{i,D}} \bigg|_{\tilde{q}_D} \right) / D \tag{A.8}$$

where the vertical bar indicates the amount that object $\tilde{o}$ of weight w is displaced by distance $\tilde{q}$, which results in particle penetration distances $\sum_{i=1}^{P} \boldsymbol{x}_i$. The length of a timestep is uncontrollable in FleX. Thus, the timestep in DART is calculated by dropping objects in both environments from a matching height and equating the time to contact the ground, where both simulators have $g = 9.81 m/s^2$. This resulted in a DART timestep of $0.0103 s$.

A.6 Real Dataset Collection Details

Participants donned an Optitrak motion capture suit with high contrast to the bed sheets to facilitate analysis of the pose and body shape. We provided S, M, L and XL sizes, and instructed participants to use a form fitting size.

We used the IAI Kinect2 package to calibrate the Kinect [142]. Our released dataset consists of RGB images and depth/point cloud data from the Kinect that are synchronized and spatially co-registered to the pressure images. We manually synchronized the modalities; only static poses are captured so the time discrepancy is insignificant. We spatially co-registered the Kinect to the pressure mat by putting 1" tungsten cubes on the corners of the pressure mat, which could be seen with all modalities. We captured a co-registration snapshot for each participant, which was taken after they were finished. We created an interface to click on the tungsten block locations on the images and used CMA-ES to find the 6DOF camera pose and co-register it with the mat.

A.7 Dataset Partitions

Table A.1 presents a detailed description of the data partitions. We split the data for gender. We also split for requiring initial limb positions to be over the surface of the bed, meaning that the Cartesian cuboids used for initial pose sampling (recall Figure A.1) are clipped in the x and y directions at the edge of the mattress.

A.8 Dataset Limitations

Domain gap. The real pressure mat has a larger force range. Additionally, as a result of putting a blanket on the bed during the real study, the overall pressure magnitude was reduced $\sim 3\times$, which was not reflected in synthetic data calibration. To correct for this, we normalize as described in Appendix B.1.

Synthetic body joint limits. We observed that roughly 2% of the synthetic poses appear uncomfortable or infeasible for a real person (Figure A.4). This work could be improved by using pose-conditioned joint angle limits such as [143] instead of constant limits. Figure A.4-right shows an impossible pose where the thighs are in collision. We were not able to check collisions between the thighs using the capsulized model because the thigh capsules are often in collision for valid poses.

Table A.1: Partitions for synthetic data and prescribed poses. For evening the leg space, see Figure A.1(a). For evening the arm space, an additional four subspaces $\{\mathcal{Y}_5, \ldots \mathcal{Y}_8\}$ are chosen because the most distal joint (hand) is allowed to extend all the way below and above the limb root joint (shoulder), measured in the y direction.

pose partition, limb distribution	gender	limbs on bed	train ct. synth	test ct. synth	test ct. real
general*	F	N	26000	3000	120
even leg space: $\{\mathcal{Y}_1, ...\mathcal{Y}_4\} \in \boldsymbol{\mathcal{Y}}_L$	M	N	26000	3000	119
even arm space: $\{\mathcal{Y}_1, ...\mathcal{Y}_8\} \in \boldsymbol{\mathcal{Y}}_A$	F	Y	26000	3000	120
	M	Y	26000	3000	120
supine general**	F	N	13000	1500	40
even leg space: $\{\mathcal{Y}_1, ...\mathcal{Y}_4\} \in \boldsymbol{\mathcal{Y}}_L$	M	N	13000	1500	39
even arm space: $\{\mathcal{Y}_1, ...\mathcal{Y}_8\} \in \boldsymbol{\mathcal{Y}}_A$	F	Y	13000	1500	40
	M	Y	13000	1500	40
supine hands behind head**	F	Y	2000	500	40
even leg space, arms Figure A.1(b)	M	Y	2000	500	40
prone hands up†	F	Y	4000	500	40
even leg space, hnds above shldrs	M	Y	4000	500	40
supine crossed legs**	F	N	2000	-	-
even leg space, even arm space,	M	N	2000	-	-
feet must cross according to	F	Y	2000	500	40
x direction in Figure A.1(a)	M	Y	2000	500	38
supine straight limbs**	F	N	2000	-	-
even leg space, even arm space,	M	N	2000	-	-
elbows and knees straight	F	Y	2000	500	40
	M	Y	2000	500	36
TOTAL	-	-	184000	22000	952

* $\theta_{r,3} \sim \mathcal{U}[-\frac{\pi}{3}, \frac{\pi}{3}]$, $\theta_{r,1} \sim \mathcal{U}[-\pi, \pi]$
** $\theta_{r,3} \sim \mathcal{U}[-\frac{\pi}{3}, \frac{\pi}{3}]$, $\theta_{r,1} = 0$
† $\theta_{r,3} \sim \mathcal{U}[-\frac{\pi}{3}, \frac{\pi}{3}]$, $\theta_{r,1} = \pi$

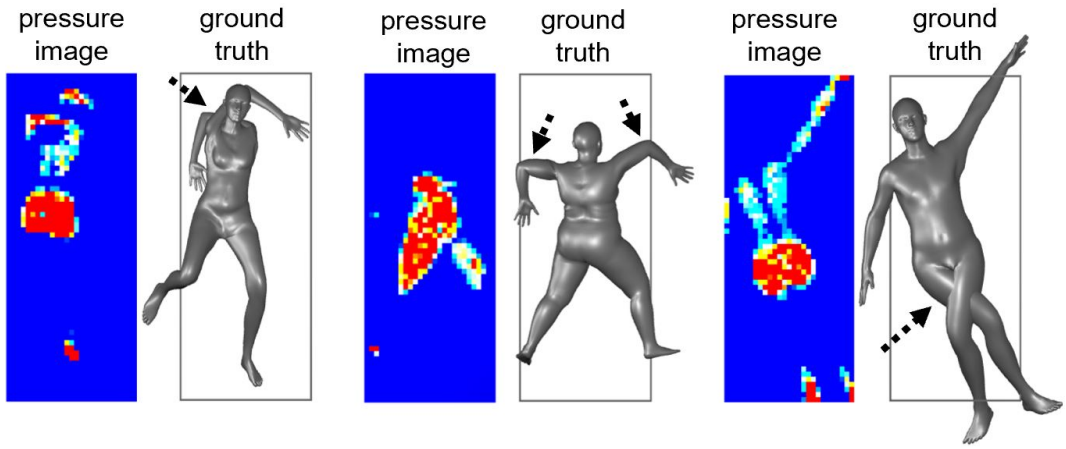

Figure A.4: Uncomfortable or infeasible poses outside of typical human movement range (left, middle). Impossible pose where the thighs are in collision (right). [11]

APPENDIX B

PRESSURENET DETAILS

B.1 PressureNet Architecture Details

CNN - Convolutional Neural Network. Our CNN architecture, depicted in Figure B.1, is similar to that of section 2.3, and uses the same kernel sizes, layers, and dropout. The first layer is a convolutional layer with a 7x7 kernel, and uses a stride of 2 and zero padding of size 3 on the sides of input images. The max pooling layer has a stride of 2 and padding of 0. All other convolutional layers are 3x3 with a stride of 1 and padding of 0. We use 192 channels in the first two convolutional layers and in the max pooling layer, and 384 channels in the last two convolutional layers. This CNN also differs from [9] in that we use tanh activation functions instead of ReLU. Through informal testing on smaller data sizes (e.g. 46K images), we observed that networks with tanh activations had less overfitting. We normalize the input and output of the network. To normalize the input channels, divide by the sum of taxels for each input image, $\mathbf{\Sigma}_I$. To normalize the output, we multiply it by the range of shape, pose, and posture parameters from the synthetic training dataset. We compute the range from the lower and upper limits,$\mathbf{\Psi}_L$ and $\mathbf{\Psi}_U$, of all parameters in the training dataset. For joint angle limits (i.e. pose), we use values from [77, 75, 76]. For body shape, we use sampling bounds $[-3, 3]$ from [78]. For global rotation, we use our sampling bounds for roll and yaw of $[-\pi, \pi]$ and $[-\frac{\pi}{6}, \frac{\pi}{6}]$, and for global translation, we use the size of the bed.

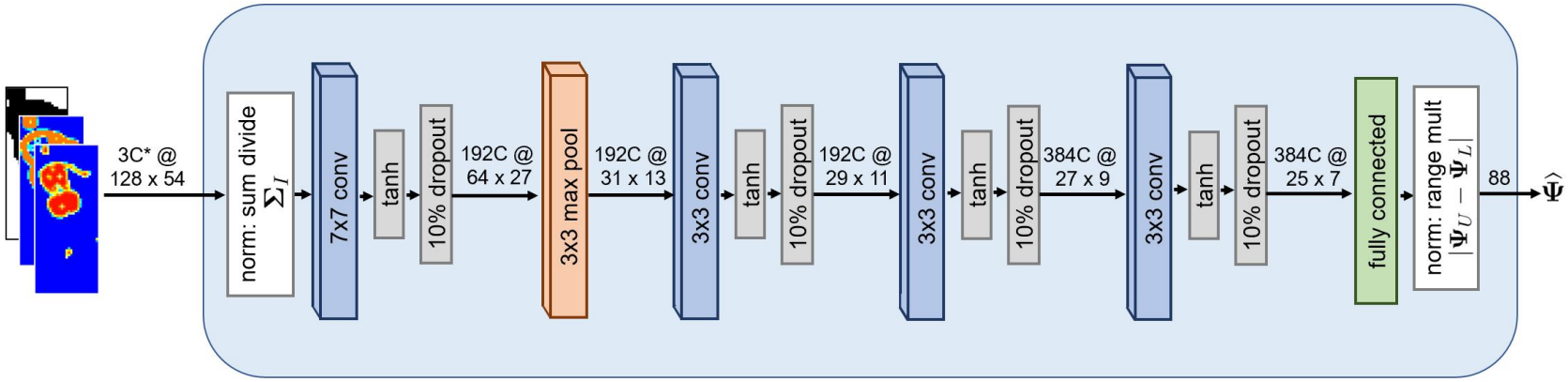

Figure B.1: PressureNet: Convolutional Neural Network (CNN) with five convolutional layers, one max pooling layer, and one fully connected layer. Input images are normalized by per-image division by the sum of taxels. * indicates that the number of channels shown (3) represents Mod1 in Figure 3.6 (a), whereas Mod2 in Figure 3.6 (a) uses 5 input channels. [11]

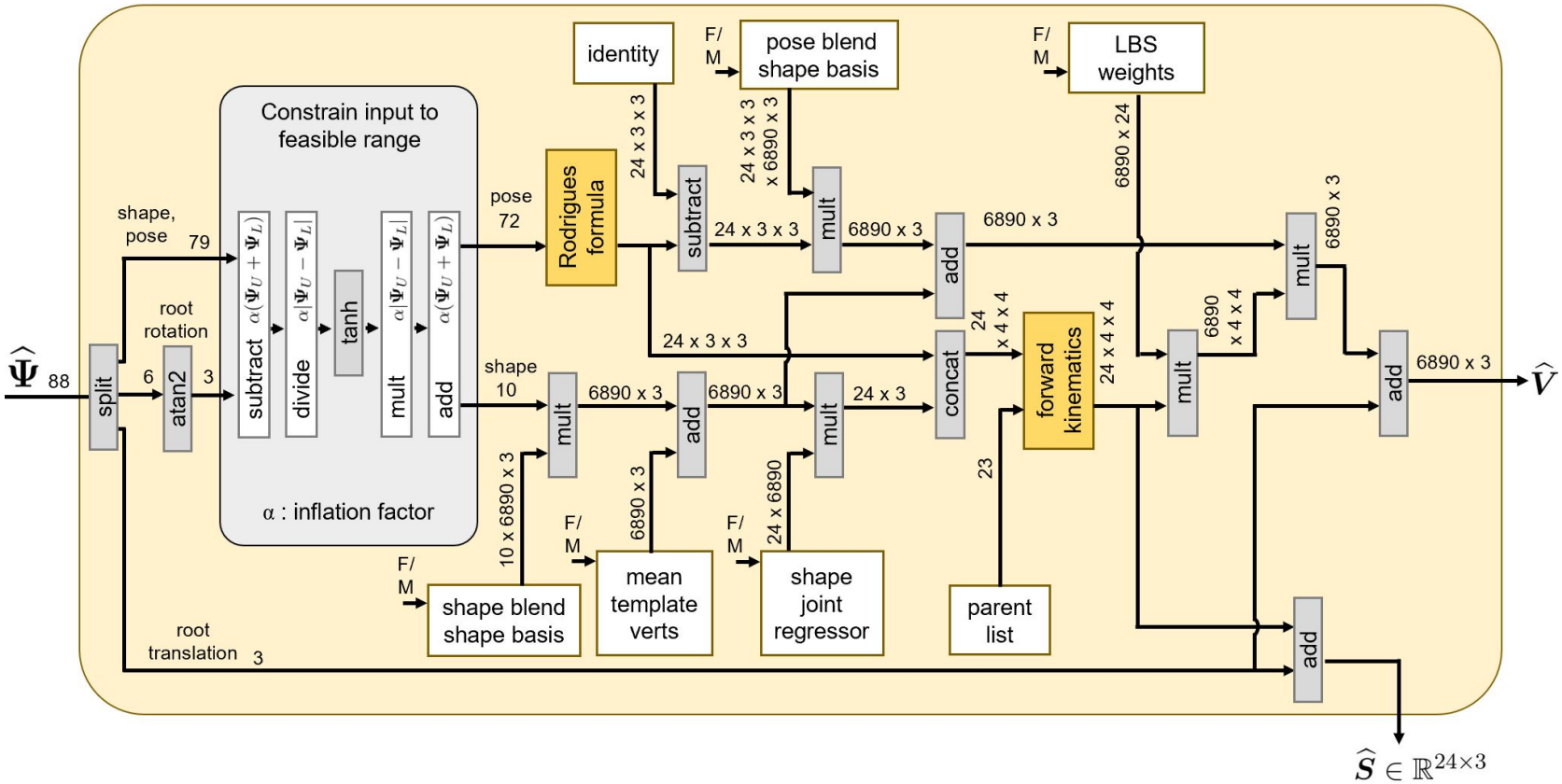

Figure B.2: PressureNet: Differentiable SMPL human mesh reconstruction from Kanazawa et al. [47]. Our additions to [47] include input constraints (shown in the light grey box) and the root joint rotation and translation. [11]

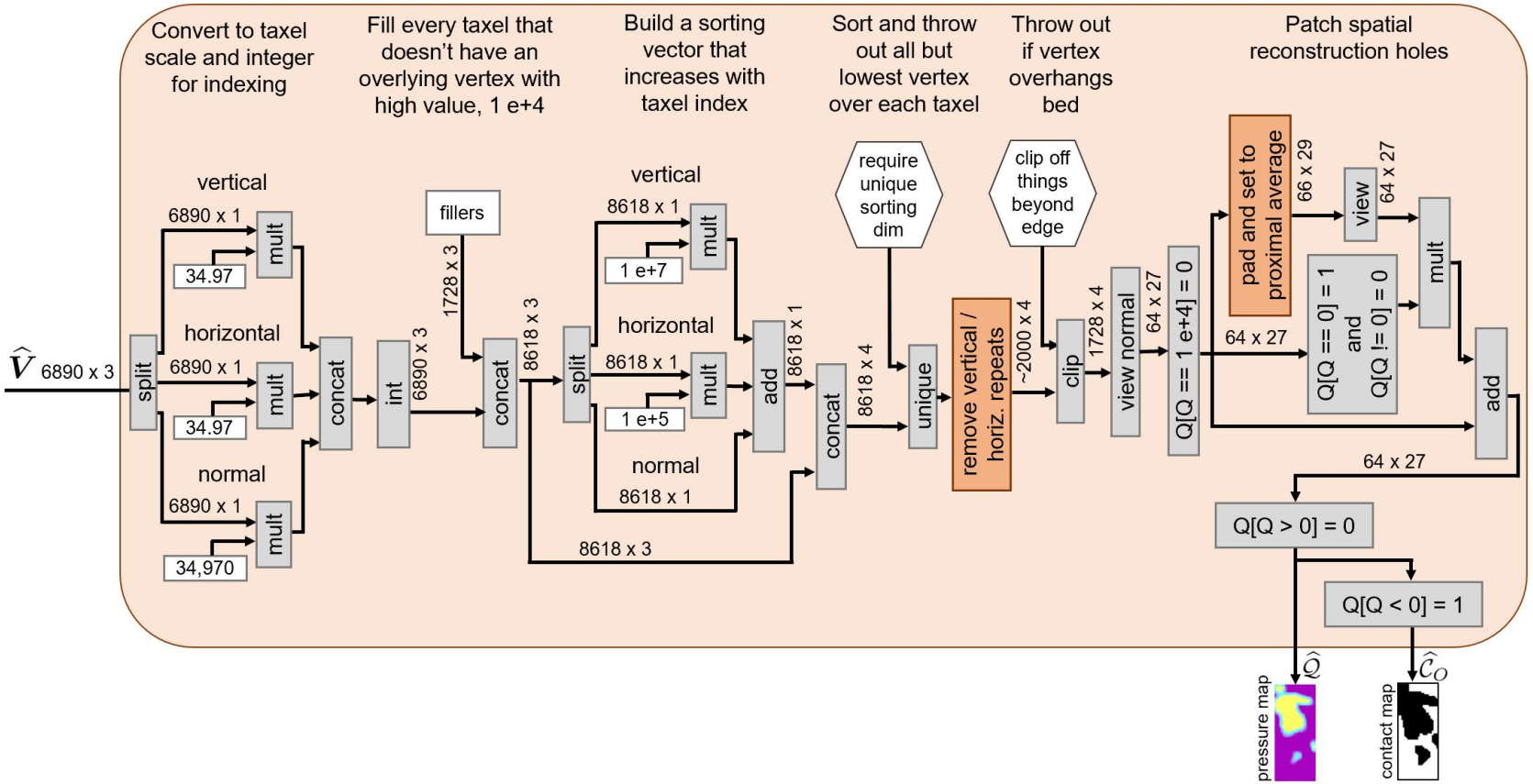

Figure B.3: PressureNet: Pressure Map Reconstruction (PMR). PMR is fully differentiable, and performs sorting, filtering and patching to reconstruct spatial maps from the human mesh. [11]

SMPL - Human Mesh Reconstruction. Following the CNN, we use the human model generative part of the HMR network [47], which inputs estimated shape, pose, and posture $\hat{\boldsymbol{\Psi}}$, and outputs a differentiable human mesh reconstruction $\hat{\boldsymbol{V}}$, as well as a set of $N = 24$ Cartesian joint positions $\hat{\boldsymbol{S}}$. This generative SMPL model, implemented in PyTorch [144], along with our modifications, is presented in Figure B.2.

In addition to using the generative kinematic SMPL embedding part of the full HMR network, our implementation constrains the input parameters to keep angles within human limits and body shape parameters inside our initial sampling range. To constrain the input parameters, we normalize the parameters to a range $[-1, 1]$ based on the limits $\boldsymbol{\Psi}_L$, $\boldsymbol{\Psi}_U$, and use a tanh function for a soft limit that is more amenable to gradient descent. Then, we perform a reverse normalization to scale back up. To prevent the tanh from clipping feasible values at the angle limits, for example a straight knee that is at 0 degrees, we inflate the angle range by a factor $\alpha = 1.2$ as shown in the figure.

PMR - Pressure Map Reconstruction. PMR, a novel component of PressureNet, takes as input a human mesh in global space $\hat{\boldsymbol{V}}$, and outputs a set of reconstructed spatial maps $\{\widehat{\mathcal{Q}}, \widehat{\mathcal{C}_O}\}$, which resemble a real pressure image and indicate where contact occurs between the estimated mesh and the bed. We reconstruct these maps differentiably as depicted in Figure B.3, meaning that we can backpropagate gradients through PMR to train the CNN. The PMR loss is based on the error between estimated spatial maps $\{\widehat{\mathcal{Q}}, \widehat{\mathcal{C}_O}\}$ and ground truth spatial maps $\{\mathcal{Q}, \mathcal{C}_O\}$. PMR works by projecting the mesh onto the surface of the bed and computing the distance that it sinks into the bed over each taxel. This amounts to finding the distance between the lowest vertex within the 2.9×2.9 cm area of each taxel and the undeformed height of the bed.

The PMR input $\hat{\boldsymbol{V}}$ is in units of meters, which we convert to units of taxels (1 m $\sim$ 35 taxels), so it can be indexed on the scale of the pressure image. We then use a process involving sorting, filtering, and patching to recreate the spatial maps, which is detailed in Figure B.3.

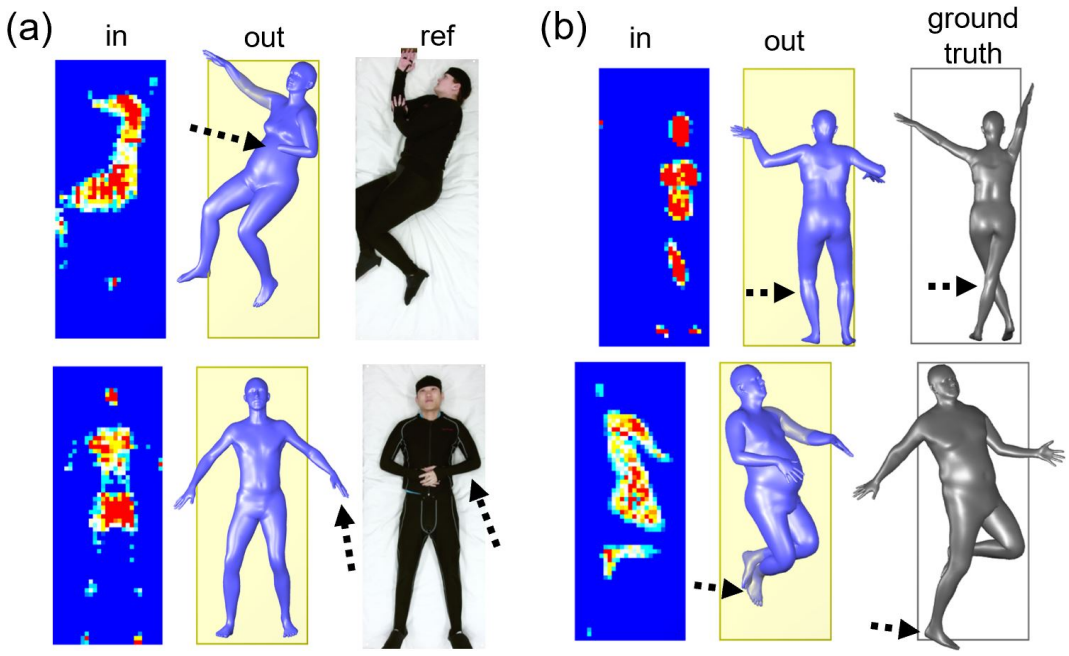

Figure B.4: (a) Real data failure cases. Self penetration of inferred left hand into chest (top), lack of information on mat leading to inaccurate pose (bottom). (b) Synthetic data failure cases: testing on *training* data, various inaccuracies.

B.2 Additional Failure Cases

We present additional failure cases in Figure B.4. One limitation is that our network does not have an interpenetration error, so the limbs sometimes intersect, e.g. the left hand in Figure B.4(a)-top left. Our network also failed for some limbs when there was little or no contact information, and for non-resting poses. This issue is related to the limitations of the sensor, which were explored in [9]. Our network failed for non-resting poses, such those in [9]; however these are not part of the training or testing PressurePose dataset. We observed some inaccuracies when testing on training data (Figure 3.10 and Figure B.4), which suggests that there is a performance limitation on the network's ability to extract pressure image features in some scenarios.

APPENDIX C

BODYPRESSURESD DATA GENERATION

C.1 Updated Synthetic Body Weighting

We develop an updated method for assigning body part weights, based on Clever *et al.* [11], for the physics simulation and synthetic dataset generation.

A per-capsule weight for the articulated capsulized chain in DartFleX is computed based on the weight distribution for an average person and capsule volume ratios. First, we describe how capsule mass for the average person is assigned. We use average body mass and mass distribution values from Tozeren [79], and calculate capsule volumes from body shape. We assume the average human of gender $g \in \{M, F\}$ has a mass of $\bar{m}_g$, mass percentage distribution for body part R of $\bar{X}_{R,g} \in \bar{\boldsymbol{X}}_g$, and SMPL body shape parameters $\bar{\boldsymbol{\beta}}_g = \mathbf{0}$. We define the mass of each capsule c in an average person to be:

$$\bar{m}_c = \bar{m}_g \bar{X}_{R,g} \frac{\bar{\mathcal{V}}_{c,g}}{\bar{\mathcal{V}}_{R,g}} \tag{C.1}$$

where $\bar{\mathcal{V}}_{c,g}$ is the volume of capsule c for a mean body shape $\bar{\boldsymbol{\beta}}_g$, and $\bar{\mathcal{V}}_{R,g}$ is the sum of volumes for all capsules in body part R. Now, we describe how this capsule mass can be converted into masses for people of other shapes. To find the mass of some capsule c for a body of particular shape $\boldsymbol{\beta}$, a capsule volume ratio between the particular person and an average person is used:

$$m_c = \bar{m}_c \frac{\mathcal{V}_c}{\bar{\mathcal{V}}_{c,g}} \tag{C.2}$$

where $\mathcal{V}_c$ is the volume of some arbitrary capsule. Note that when summing these capsules for the person, it substantially over-estimates the body density for heavy bodies and

under-estimates for light bodies. Capsule mass is corrected by dividing by the total capsule volume ratio and multiplying by the SMPL mesh volume ratio. This latter provides a better mass-density ratio model.

$$\widetilde{m}_c = m_c \frac{\sum_{j=1}^{N} \bar{\mathcal{V}}_{j,g}}{\sum_{j=1}^{N} \mathcal{V}_j} \cdot \frac{\mathcal{V}_{mesh,g}}{\bar{\mathcal{V}}_{mesh,g}} \tag{C.3}$$

where $N = 20$ capsules, $\mathcal{V}_{mesh,g}$ is the SMPL mesh volume of a person with arbitrary shape in the home pose ($\boldsymbol{\Theta} = \mathbf{0}$), and $\bar{\mathcal{V}}_{mesh,g}$ is the SMPL mesh volume of a person with average shape ($\bar{\boldsymbol{\beta}}_g = \mathbf{0}$) in the home pose.

C.2 Normalizing Pressure by Body Mass

In theory, the sum of pressure image values times the area should equal the weight of a person. However, in practice, seemingly innocuous changes such as placing an extra blanket between an object and the pressure mat can alter recorded data on some mats [11]. We normalize all ground truth pressure images in the SLP dataset and in the synthetic dataset by the body mass of each person, using the following equation:

$$\mathcal{P} = \mathcal{P}' \frac{mg}{\sum p'_{xy} A_t} \tag{C.4}$$

where m is the body mass, g is gravitational acceleration, p'_{xy} is the pressure measured by a single taxel (tactile pixel) in $\mathcal{P}'$, and A_t is the surface area of a single taxel on the pressure mat. We assume the blanket has negligible mass. As such, normalized ground truth pressure $\mathcal{P}$ projects onto the human mesh $\mathcal{M}_H$ following Eq. 6.

C.3 Co-registering Real and Synthetic Images

We position the simulated pinhole depth camera to match the camera in the SLP dataset using the given camera intrinsics. In the SLP dataset, the various visual modalities (depth, RGB, IR) are aligned to the pressure mat via a calibration process, such that the 3D world

origin in point cloud space is located at the top left corner of the pressure mat. However, the twin bed mattress and pressure mat used in the SLP dataset are of a different resolution and size than those in the simulation. Using manufacturer specifications and real measurements, the real SLP pressure images are converted to a lower resolution and slightly smaller size so they match the synthetic images. We perform similar conversions of depth imagery. It is ambiguous how far the corner of the pressure mat is from the corner of the bed. Our simulated bed and pressure mat are based on precise measurements from a real bed and mat in our previous work [11]. While both environments are square and the rotational discrepancy is negligible, we precisely align the reference frame origin translation between the real SLP dataset and our synthetic dataset. We use a grid search to do this.

C.4 Blanket Configuration Partition Details

The two blanket partitions are represented by translations $\boldsymbol{s}_B^*$ and $\boldsymbol{s}_B^{**}$. In the first, we set the initial blanket position to:

$$\boldsymbol{s}_B^* = \left\{ \begin{array}{ccc} \frac{1}{2}s_{M,1}, & s_{neck,2} - \frac{1}{2}s_{M,2}, & \xi_B \end{array} \right\} \tag{C.5}$$

where $\boldsymbol{s}_M$ contains the mattress dimensions, $s_{neck,2}$ is the distance to the person's neck-line from the origin, and the translation of the blanket is measured from its center and the world reference frame is located at the bottom left corner of the mattress. The constant ξ_B is a distance above the resting human body, where ξ_B is always above the highest joint position. In the second partition, we randomly sample across the person in bed, using:

$$\boldsymbol{s}_B^{**} = \left\{ \begin{array}{c} \mathcal{U}(-s_{j,1,min}, s_{j,1,max}) \\ s_\alpha + \mathcal{U}\big(-\frac{s_{neck,2}-s_{j,2,min}}{2}, \frac{s_{neck,2}-s_{j,2,min}}{2} \big) \\ \xi_B \end{array} \right\}^\top \tag{C.6}$$

where $s_\alpha = s_{neck,2} - \frac{1}{2}s_{M,2} - 0.4$, so the distribution of the longitudinal distance $s_{B,2}^{**}$

is centered around a location 0.4 meters below the neckline. We shift down by 0.4 so that the bottom edge of the blanket does not leave the legs too uncovered, which is uncommon in the SLP dataset. The range of the distribution is set to be equal to the longitudinal distance between the neck, $s_{neck,2}$, and the joint furthest toward the bottom of the bed, $s_{j,2,min}$ (typically a foot). The uniform distribution of the lateral shift $s^{**}_{B,1}$ across the surface of the bed is set to a range between the furthest extend of a human joint in either direction (i.e. $-s_{j,1,min}, s_{j,1,max}$).

APPENDIX D

BODYPRESSURE NETWORK DETAILS

D.1 BPWnet Loss Computation Details

This section provides explicit definitions for the loss function components introduced in section 4.4. Figure D.1(a) shows BPWnet in more detail for reference to the loss variables.

Human pose loss computation. Recall that the encoder outputs estimated SMPL parameters $\widehat{\boldsymbol{\Psi}} = \begin{bmatrix}\hat{\boldsymbol{\beta}} & \widehat{\boldsymbol{\Theta}} & \hat{\boldsymbol{s}} & \hat{\boldsymbol{x}} & \hat{\boldsymbol{y}} & \hat{b}\end{bmatrix}^\top \in \mathbb{R}^{89}$. The first two terms, $\hat{\boldsymbol{\beta}} \in \mathbb{R}^{10}$ and $\widehat{\boldsymbol{\Theta}} \in \mathbb{R}^{69}$, contain the body shape and and joint angles, respectively, for the SMPL human model [10]. The terms $\hat{\boldsymbol{s}} \in \mathbb{R}^3$, $\hat{\boldsymbol{x}} \in \mathbb{R}^3$, and $\hat{\boldsymbol{y}} \in \mathbb{R}^3$ define the global transform of the SMPL model, with translation $\hat{\boldsymbol{s}}$ and continuous rotation parameters $\{x_u, x_v, x_w\} = \hat{\boldsymbol{x}}$, $\{y_u, y_v, y_w\} = \hat{\boldsymbol{y}}$ for 3 DOF, i.e. $\phi_u = atan2(y_u, x_u)$ and $\{\phi_u, \phi_v, \phi_w\} = \boldsymbol{\phi} \in \mathbb{R}^3$. The term $\hat{b} \in \mathbb{R}^1$ defines the distance between the camera and the bed. The encoder output $\widehat{\boldsymbol{\Psi}}$ is used to differentiably reconstruct a SMPL mesh with an embedded human kinematics model. As such, it also outputs 3D Cartesian joint positions $\widehat{\boldsymbol{S}} \in \mathbb{R}^{24\times3}$. The SMPL block outputs a set of vertices for the human, $\widehat{\boldsymbol{V}}_H \in \mathbb{R}^{6890\times3}$. The estimated human mesh $\widehat{\mathcal{M}}_H$ can be assembled from $\widehat{\boldsymbol{V}}_H$, as well as the faces $\boldsymbol{F}_H$, which are constant.

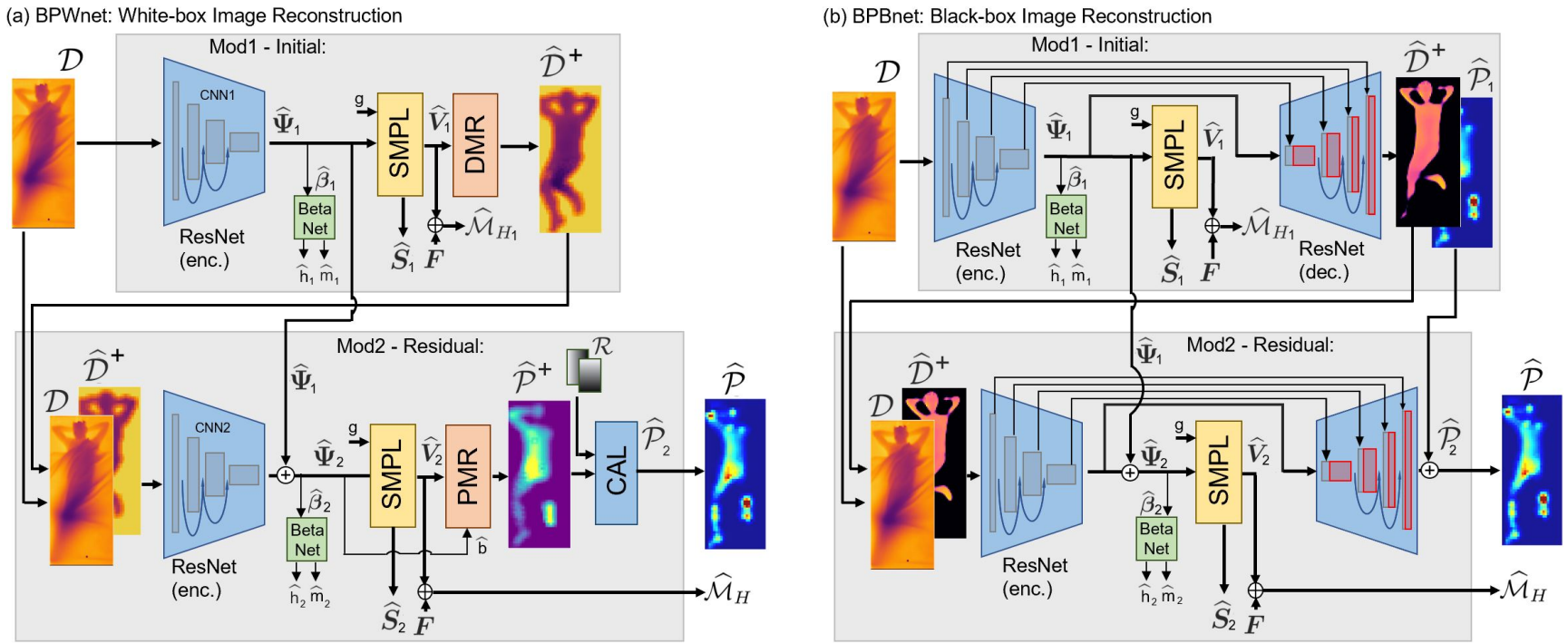

Figure D.1: A comparison between BPWnet and its black-box variant, BPBnet. The BPWnet shown above is the same as Fig. 5 but contains more detail. BPBnet replaces the white-box DMR and PMR components with a learned ResNet34 decoder that shares weights with the encoder.

Supervision is provided at multiple locations throughout the network to better utilize the fully observable nature of the synthetic data. Previous work has provided supervision on body shape [11] and on 3D Cartesian joint positions [47]. We observed that adding global body rotation, SMPL joint angles, and camera-to-bed distance enables quicker and better learning at little computational cost, and define the following loss function:

$$\begin{aligned}\mathcal{L}_{\text{SMPL}} := &\frac{1}{N_\beta \sigma_\beta}\left|\left|\boldsymbol{\beta} - \hat{\boldsymbol{\beta}}\right|\right|_1 + \frac{1}{N_\Theta \sigma_\Theta}\left|\left|\boldsymbol{\Theta} - \hat{\boldsymbol{\Theta}}\right|\right|_1 \\ &+\frac{1}{6\sigma_{yx}}\left(\left|\left|\boldsymbol{x} - \hat{\boldsymbol{x}}\right|\right|_1 + \left|\left|\boldsymbol{y} - \hat{\boldsymbol{y}}\right|\right|_1\right) \\ &+\frac{1}{N_s \sigma_s}\sum_{j=1}^{N_s}\left|\left|\boldsymbol{s}_j - \hat{\boldsymbol{s}}_j\right|\right|_2 + \frac{1}{\sigma_b}\left|\left|b - \hat{b}\right|\right|_1\end{aligned} \tag{D.1}$$

where $N_\beta = 10$ body shape parameters, $N_\Theta = 69$ joint angles, there are 6 parameters of continuous global rotation, $N_s = 24$ Cartesian joint positions, and $\boldsymbol{s}_j \subset \boldsymbol{S}$ represents the Cartesian position of a single joint. Each term is normalized by standard deviations σ_β, σ_Θ, σ_{yx}, σ_s, and σ_b, which are computed from the entire synthetic training dataset. Note that $\mathcal{L}_{\text{SMPL}}$ is sufficient to reconstruct the entire SMPL mesh, but does not require it during differentiation, so it can be computed efficiently. If the mesh is differentiably reconstructed, we may also define a loss on the SMPL mesh vertices:

$$\mathcal{L}_{\text{v2v}} := \frac{1}{N_{V_H}\sigma_{V_H}}\sum_{j=1}^{N_{V_H}}\left|\left|\boldsymbol{v}_j - \widehat{\boldsymbol{v}}_j\right|\right|_2 \tag{D.2}$$

where $\boldsymbol{v}_j \subset \boldsymbol{V}_H$ represents the Cartesian position of a single human mesh vertex, $N_{V_H} = 6890$ vertices, and the loss term is normalized by σ_{V_H}.

Image reconstruction loss computation. Because the initial module is only necessary for producing a rough estimate and there are no learned parameters in DMR, no loss is computed on the depth maps $\widehat{\mathcal{D}}^+$. However, we do train the residual module with a loss on the reconstructed pressure maps $\widehat{\mathcal{P}}^+$. This minimizes the difference between the reconstructed pressure map for the estimated pose $\widehat{\mathcal{P}}^+$ and the ground truth pose $\mathcal{P}^+$. A binary contact map $\widehat{\mathcal{C}}_{\text{p}^+}$ is computed directly from $\widehat{\mathcal{P}}^+$, and we define the following loss function

for PMR:

$$\mathcal{L}_{\mathcal{P}^+} := \frac{1}{T\sigma_{\mathcal{P}^+}}\left|\left|\mathcal{P}^+ - \widehat{\mathcal{P}}^+\right|\right|_2^2 + \frac{1}{T\sigma_{\mathcal{C}_{\mathrm{p}^+}}}\left|\left|\mathcal{C}_{\mathrm{p}^+} - \widehat{\mathcal{C}}_{\mathrm{p}^+}\right|\right|_1 \tag{D.3}$$

where $T = 1728$ pressure map taxels and standard deviations $\sigma_{\mathcal{Q}^-}$ and $\sigma_{\mathcal{C}_{q^-}}$ are similarly computed over the entire training dataset. Similarly, to train the feature calibration network CAL, we define a loss between the estimated and ground truth reconstructed pressure images $\mathcal{P}$ and their contact maps $\mathcal{C}_{\mathrm{p}}$:

$$\mathcal{L}_{\mathcal{P}} := \frac{1}{T\sigma_{\mathcal{P}}}\left|\left|\mathcal{P} - \widehat{\mathcal{P}}\right|\right|_1 + \frac{1}{T\sigma_{\mathcal{C}_{\mathrm{p}}}}\left|\left|\mathcal{C}_{\mathrm{p}} - \widehat{\mathcal{C}}_{\mathrm{p}}\right|\right|_1 \tag{D.4}$$

with standard deviations $\sigma_{\mathcal{P}}$ and $\sigma_{\mathcal{C}_p}$.

D.2 Black-box Reconstruction Net (BPBnet)

Here we describe BPBnet, a deep network with a black-box model of image reconstruction, shown in Figure D.1 (b). BPBnet uses a traditional black-box CNN for encoding depth images, as well as a black-box CNN for reconstructing depth and pressure images. Like BPWnet, it also produces an initial estimate with the first module, and uses the second module, to refine it through residual error [87, 57, 11].

Each module has an encoder and decoder which extract features into a latent space and upsamples them back into images in a reverse fashion. While the middle latent space in such a network is often left unconstrained, it may encode specific targets, such as class labels [111]. Here it is used to encode the SMPL model parameters $\widehat{\mathbf{\Psi}}$, which are a sufficient representation to decode a fixed-frame depth image of the human body. For BPBnet, $\widehat{\mathbf{\Psi}} \in \mathbb{R}^{88}$, since it drops the camera to bed distance term present in BPWnet. $\mathcal{L}_{\text{SMPL}}$ for BPBnet similarly drops the last term on Equation D.1. The black-box ResNet decoder outputs occlusion-free depth images, pressure images, and their associated binary contact maps.

Black-box decoding. Vector $\widehat{\mathbf{\Psi}}$ is additionally fed into a decoder, which uses learnable weights to reconstruct an occlusion-free depth image of the human body, $\widehat{\mathcal{D}}^{+}$, and a pressure image $\widehat{\mathcal{P}}$. BPBnet employs a U-Net [85] style architecture, where the encoder and decoder share feature maps at each stage of convolution. We define a loss between the estimated and ground truth reconstructed depth images $\mathcal{D}^{+}$, as well as the contact maps $\mathcal{C}_{d^{+}}$, which are a binary function of $\mathcal{D}^{+}$:

$$\mathcal{L}_{\mathcal{D}^{+}} := \frac{1}{T\sigma_{\mathcal{D}^{+}}}\left|\left|\mathcal{D}^{+} - \widehat{\mathcal{D}}^{+}\right|\right|_2^2 + \frac{1}{T\sigma_{\mathcal{C}_{\mathrm{d}^{+}}}}\left|\left|\mathcal{C}_{\mathrm{d}^{+}} - \widehat{\mathcal{C}}_{\mathrm{d}^{+}}\right|\right|_1 \tag{D.5}$$

BPBnet training strategy. We train Mod1 and Mod2 separately, and use the same pre-trained BetaNet from subsection 4.2.1 with locked weights for both modules. We first train Mod1, with the following loss function:

$$\mathbb{L}_{\mathrm{BPB}_1} = \mathcal{L}_{\mathrm{BetaNet}_1} + \mathcal{L}_{\mathrm{SMPL}_1} + \mathcal{L}_{\mathcal{D}_1^{+}} + \mathcal{L}_{\mathcal{P}_1} \tag{D.6}$$

where subscripts 1 on the terms indicate a loss computed from Mod1 estimates. This differs from the loss in Equation 4.7 in that it contains image reconstruction terms for training the decoder. Next, we precompute a set of reconstructed depth and contact maps $\{\widehat{\boldsymbol{\mathcal{D}}}^{+}, \widehat{\boldsymbol{\mathcal{C}}}_{\mathrm{d}^{+}}\}$ by pushing the entire depth image dataset $\boldsymbol{\mathcal{D}}$ through Mod1. We train Mod2 with a dataset consisting of inputs $\{\boldsymbol{\mathcal{D}}, \widehat{\boldsymbol{\mathcal{D}}}^{+}, \widehat{\boldsymbol{\mathcal{C}}}_{\mathrm{d}^{+}}\}$. Mod2 learns a spatial residual correction to improve estimates $\widehat{\mathbf{\Psi}}_1$ and $\widehat{\mathcal{P}}_1$, which are shown on the bottom of Figure D.1 (a). As such, the Mod2 encoder outputs a correction $(\widehat{\mathbf{\Psi}}_2 - \widehat{\mathbf{\Psi}}_1)$ and the decoder outputs $(\widehat{\mathcal{P}}_2 - \widehat{\mathcal{P}}_1)$. The following loss is computed to train Mod2 in BPBnet:

$$\mathbb{L}_{\mathrm{BPB}_2} = \mathcal{L}_{\mathrm{BetaNet}_2} + \mathcal{L}_{\mathrm{SMPL}_2} + \mathcal{L}_{\mathrm{v2v}_2} + \mathcal{L}_{\mathcal{P}_2} \tag{D.7}$$

In contrast to Mod1 in BPWnet, the depth reconstruction term is omitted and a term is added for the SMPL mesh vertices, which slows training but refines estimation quality. The Mod2 encoder in BPBnet outputs $\widehat{\mathcal{M}}_{H,2}$ and the decoder outputs $\widehat{\mathcal{P}}_2$.

REFERENCES

[1] S. Johnson and M. Everingham, "Clustered pose and nonlinear appearance models for human pose estimation.," in *British Machine Vision Conference (BMVC)*, British Machine Vision Association, vol. 2, 2010, p. 5.

[2] B. Sapp and B. Taskar, "Modec: Multimodal decomposable models for human pose estimation," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2013, pp. 3674–3681.

[3] D. Mehta, H. Rhodin, D. Casas, P. Fua, O. Sotnychenko, W. Xu, and C. Theobalt, "Monocular 3d human pose estimation in the wild using improved cnn supervision," in *2017 international conference on 3D vision (3DV)*, IEEE, 2017, pp. 506–516.

[4] J. Rosen, J. C. Perry, N. Manning, S. Burns, and B. Hannaford, "The human arm kinematics and dynamics during daily activities-toward a 7 dof upper limb powered exoskeleton," in *ICAR'05. Proceedings., 12th International Conference on Advanced Robotics, 2005.*, IEEE, 2005, pp. 532–539.

[5] Y. Zimmermann, A. Forino, R. Riener, and M. Hutter, "Anyexo: A versatile and dynamic upper-limb rehabilitation robot," *IEEE Robotics and Automation Letters*, vol. 4, no. 4, pp. 3649–3656, 2019.

[6] R. Chalodhorn, D. B. Grimes, K. Grochow, and R. P. Rao, "Learning to walk through imitation.," in *International Joint Conferences on Artificial Intelligence (IJCAI)*, vol. 7, 2007, pp. 2084–2090.

[7] W. Z. Peng, C. Mummolo, and J. H. Kim, "Stability criteria of balanced and steppable unbalanced states for full-body systems with implications in robotic and human gait," in *2020 IEEE International Conference on Robotics and Automation (ICRA)*, IEEE, 2020, pp. 9789–9795.

[8] M. Macklin, M. Müller, N. Chentanez, and T.-Y. Kim, "Unified particle physics for real-time applications," *ACM Transactions on Graphics*, vol. 33, no. 4, 2014.

[9] H. M. Clever, A. Kapusta, D. Park, Z. Erickson, Y. Chitalia, and C. C. Kemp, "3D human pose estimation on a configurable bed from a pressure image," in *IROS*, IEEE, 2018, pp. 54–61.

[10] M. Loper, N. Mahmood, J. Romero, G. Pons-Moll, and M. J. Black, "Smpl: A skinned multi-person linear model," *ACM Transactions on Graphics*, vol. 34, no. 6, p. 248, 2015.

[11] H. M. Clever, Z. Erickson, A. Kapusta, G. Turk, C. K. Liu, and C. C. Kemp, "Bodies at rest: 3d human pose and shape estimation from a pressure image using synthetic data," in *CVPR*, IEEE, 2020.

[12] H. M. Clever, P. Grady, G. Turk, and C. C. Kemp, "Bodypressure–inferring body pose and contact pressure from a depth image," *Under minor revision at IEEE Transactions on Pattern Analysis and Machine Intelligence (PAMI)*, 2021.

[13] C. C. Kemp, A. Edsinger, H. M. Clever, and B. Matulevich, "The design of stretch: A compact, lightweight mobile manipulator for indoor human environments," *arXiv preprint arXiv:2109.10892*, 2021.

[14] S. Liu, X. Huang, N. Fu, C. Li, Z. Su, and S. Ostadabbas, "Simultaneously-collected multimodal lying pose dataset: Towards in-bed human pose monitoring under adverse vision conditions," *arXiv preprint arXiv:2008.08735*, 2020.

[15] K. Hawkins, P. Grice, T. Chen, C.-H. King, and C. C. Kemp, "Assistive mobile manipulation for self-care tasks around the head," in *2014 IEEE Symposium on Computational Intelligence in Robotic Rehabilitation and Assistive Technologies*, IEEE, 2014.

[16] A. Kapusta, Y. Chitalia, D. Park, and C. C. Kemp, "Collaboration between a robotic bed and a mobile manipulator may improve physical assistance for people with disabilities," in *RO-MAN*, IEEE, 2016.

[17] P. M. Grice and C. C. Kemp, "Assistive mobile manipulation: Designing for operators with motor impairments," in *RSS 2016 Workshop on Socially and Physically Assistive Robotics for Humanity*, 2016.

[18] T. Harada, T. Sato, and T. Mori, "Pressure distribution image based human motion tracking system using skeleton and surface integration model," in *ICRA*, IEEE, vol. 4, 2001, pp. 3201–3207.

[19] R. Grimm, S. Bauer, J. Sukkau, J. Hornegger, and G. Greiner, "Markerless estimation of patient orientation, posture and pose using range and pressure imaging," *International journal of computer assisted radiology and surgery*, vol. 7, no. 6, pp. 921–929, 2012.

[20] J. J. Liu, M.-C. Huang, W. Xu, and M. Sarrafzadeh, "Bodypart localization for pressure ulcer prevention," in *EMBC*, IEEE, 2014, pp. 766–769.

[21] A. S. Kapusta, P. M. Grice, H. M. Clever, Y. Chitalia, D. Park, and C. C. Kemp, "A system for bedside assistance that integrates a robotic bed and a mobile manipulator," *PloS one*, vol. 14, no. 10, e0221854, 2019.

[22] P. M. Grice, Y. Chitalia, M. Rich, H. M. Clever, and C. C. Kemp, "Autobed: Open hardware for accessible web-based control of an electric bed," in *RESNA*, 2016.

[23] Y. Gal and Z. Ghahramanim, "Dropout as a bayesian approximation: Representing model uncertainty in deep learning," in *Conference on Machine Learning*, 2016, pp. 1050–1059.

[24] W. Gong, X. Zhang, J. Gonzàlez, A. Sobral, T. Bouwmans, C. Tu, and E.-h. Zahzah, "Human pose estimation from monocular images: A comprehensive survey," *Sensors*, vol. 12, no. 16, p. 1966, 2016.

[25] N. Sarafianos, B. Boteanu, C. Ionescu, and I. A. Kakadiaris, "3d human pose estimation: A review of the literature and analysis of covariates," *Computer Vision and Image Understanding*, no. 152, pp. 1–20, 2016.

[26] R. Okada and S. Soatto, "Relevant feature selection for human pose estimation and localization in cluttered images," in *ECCV*, Springer, 2008, pp. 434–445.

[27] C. Ionescu, D. Papava, V. Olaru, and C. Sminchisescu, "Human3.6m: Large scale datasets and predictive methods for 3d human sensing in natural environments," *Trans. on Pattern Analysis and Machine Intelligence*, vol. 36, no. 7, pp. 1325–1339, 2014.

[28] A. Agarwal and B. Triggs, "Recovering 3d human pose from monocular images," *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, vol. 28, no. 1, pp. 44–58, 2006.

[29] A. Toshev and C. Szegedy, "Deeppose: Human pose estimation via deep neural networks," in *CVPR*, 2014, pp. 1653–1660.

[30] J. J. Tompson, A. Jain, Y. LeCun, and C. Bregler, "Joint training of a convolutional network and a graphical model for human pose estimation," in *Advances in neural information processing systems*, 2014, pp. 1799–1807.

[31] G. Pavlakos, X. Zhou, K. G. Derpanis, and K. Daniilidis, "Coarse-to-fine volumetric prediction for single-image 3d human pose," in *CVPR*, IEEE, 2017.

[32] X. Zhou, X. Sun, W. Zhang, S. Liang, and Y. Wei, "Deep kinematic pose regression," in *ECCV 2016 Workshops*, 2016, pp. 186–201.

[33] S.-E. Wei, V. Ramakrishna, T. Kanade, and Y. Sheikh, "Convolutional pose machines," in *CVPR*, 2016, pp. 4724–4732.

[34] X. Zhou, M. Zhu, S. Leonardos, K. G. Derpanis, and K. Daniilidis, "Sparseness meets deepness: 3d human pose estimation from monocular video," in *CVPR*, IEEE, 2016, pp. 4966–4975.

[35] F. Bogo, A. Kanazawa, C. Lassner, P. Gehler, J. Romero, and M. J. Black, "Keep it SMPL: Automatic estimation of 3D human pose and shape from a single image," in *ECCV*, Springer, 2016, pp. 561–578.

[36] S. Li and A. B. Chan, "3d human pose estimation from monocular images with deep convolutional neural network," in *Asian Conference on Computer Vision*, Springer, pp. 332–347.

[37] M. Farshbaf, R. Yousefi, M. B. Pouyan, S. Ostadabbas, M. Nourani, and M. Pompeo, "Detecting high-risk regions for pressure ulcer risk assessment," in *BIBM*, IEEE, 2013, pp. 255–260.

[38] S. Ostadabbas, M. B. Pouyan, M. Nourani, and N. Kehtarnavaz, "In-bed posture classification and limb identification," in *BioCAS*, IEEE, 2014, pp. 133–136.

[39] S. Van der Walt, J. L. Schönberger, J. Nunez-Iglesias, F. Boulogne, J. D. Warner, N. Yager, E. Gouillart, and T. Yu, "Scikit-image: Image processing in python," *PeerJ*, vol. 2, e453, 2014.

[40] D. Mehta, S. Sridhar, O. Sotnychenko, H. Rhodin, M. Shafiei, H.-P. Seidel, W. Xu, D. Casas, and C. Theobalt, "Vnect: Real-time 3d human pose estimation with a single rgb camera," *ACM Transactions on Graphics*, vol. 36, no. 4, 44:1–44:14, 2017.

[41] A. Paszke, S. Gross, S. Chintala, G. Chanan, E. Yang, Z. DeVito, Z. Lin, A. Desmaison, L. Antiga, and A. Lerer, "Automatic differentiation in pytorch," 2017.

[42] N. Dalal and B. Triggs, "Histograms of oriented gradients for human detection," in *CVPR*, IEEE, vol. 1, 2005, pp. 886–893.

[43] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," in *arXiv preprint arXiv:1412.6980*, 2014.

[44] I. Millington, *Game physics engine development: how to build a robust commercial-grade physics engine for your game*. CRC Press, 2010.

[45] L. Casas, N. Navab, and S. Demirci, "Patient 3d body pose estimation from pressure imaging," *International Journal of Computer Assisted Radiology and Surgery*, pp. 1–8, 2019.

[46] K. Chen, P. Gabriel, A. Alasfour, C. Gong, W. K. Doyle, O. Devinsky, D. Friedman, P. Dugan, L. Melloni, T. Thesen, D. Gonda, S. Sattar, S. Wong, and V. Gilja, "Patient-specific pose estimation in clinical environments," *IEEE Journal of Translational Engineering in Health and Medicine*, vol. 6, pp. 1–11, 2018.

[47] A. Kanazawa, M. J. Black, D. W. Jacobs, and J. Malik, "End-to-end recovery of human shape and pose," in *CVPR*, IEEE, 2018, pp. 7122–7131.

[48] V. Medical, *BodiTrak pressure mapping system solutions*, www.boditrak.com/products/medical.php, Last accessed on 2020-02-27.

[49] Invacare, *5410IVC Full Electric Homecare Bed*, www.invacare.com/cgi-bin/imhqprd/inv_catalog/prod_cat_detail.jsp?s=0&prodID=5410IVC, Last accessed on 2020-02-27.

[50] F. Achilles, A.-E. Ichim, H. Coskun, F. Tombari, S. Noachtar, and N. Navab, "Patient mocap: Human pose estimation under blanket occlusion for hospital monitoring applications," in *International Conference on Medical Image Computing and Computer-Assisted Intervention*, Springer, 2016, pp. 491–499.

[51] S. Liu, Y. Yin, and S. Ostadabbas, "In-bed pose estimation: Deep learning with shallow dataset," *IEEE Journal of Translational Engineering in Health and Medicine*, vol. 7, pp. 1–12, 2019.

[52] S. Liu and S. Ostadabbas, "Seeing under the cover: A physics guided learning approach for in-bed pose estimation," in *Medical Image Computing and Computer Assisted Intervention*, Springer, 2019, pp. 236–245.

[53] Y. Yin, J. P. Robinson, and Y. Fu, "Multimodal in-bed pose and shape estimation under the blankets," *arXiv preprint arXiv:2012.06735*, 2020.

[54] J. Shotton, A. Fitzgibbon, M. Cook, T. Sharp, M. Finocchio, R. Moore, A. Kipman, and A. Blake, "Real-time human pose recognition in parts from single depth images," in *CVPR*, 2011, pp. 1297–1304.

[55] G. Pavlakos, V. Choutas, N. Ghorbani, T. Bolkart, A. A. Osman, D. Tzionas, and M. J. Black, "Expressive body capture: 3d hands, face, and body from a single image," in *CVPR*, IEEE, 2019, pp. 10 975–10 985.

[56] D. Anguelov, P. Srinivasan, D. Koller, S. Thrun, J. Rodgers, and J. Davis, "SCAPE: Shape completion and animation of people," *ACM Transactions on Graphics*, vol. 24, no. 3, pp. 408–416, 2005.

[57] J. Carreira, P. Agrawal, K. Fragkiadaki, and J. Malik, "Human pose estimation with iterative error feedback," in *CVPR*, IEEE, 2016, pp. 4733–4742.

[58] S. Brahmbhatt, C. Ham, C. C. Kemp, and J. Hays, "ContactDB: Analyzing and predicting grasp contact via thermal imaging," in *CVPR*, IEEE, 2019, pp. 8709–8719.

[59] M. Hassan, V. Choutas, D. Tzionas, and M. J. Black, "Resolving 3d human pose ambiguities with 3d scene constraints," in *ICCV*, IEEE, 2019.

[60] Y. Hasson, G. Varol, D. Tzionas, I. Kalevatykh, M. J. Black, I. Laptev, and C. Schmid, "Learning joint reconstruction of hands and manipulated objects," in *CVPR*, 2019, pp. 11 807–11 816.

[61] A. Bulat and G. Tzimiropoulos, "Human pose estimation via convolutional part heatmap regression," in *ECCV*, 2016, pp. 717–732.

[62] A. Newell, K. Yang, and J. Deng, "Stacked hourglass networks for human pose estimation," in *ECCV*, Springer, 2016, pp. 483–499.

[63] V. Davoodnia, S. Ghorbani, and A. Etemad, "In-bed pressure-based pose estimation using image space representation learning," *arXiv preprint arXiv:1908.08919*, 2019.

[64] T. Harada, T. Mori, Y. Nishida, T. Yoshimi, and T. Sato, "Body parts positions and posture estimation system based on pressure distribution image," in *ICRA*, IEEE, vol. 2, 1999, pp. 968–975.

[65] J. Lee, M. X. Grey, S. Ha, T. Kunz, S. Jain, Y. Ye, S. S. Srinivasa, M. Stilman, and C. K. Liu, "Dart: Dynamic animation and robotics toolkit," *Journal of Open Source Software*, vol. 3, no. 22, 2018.

[66] Z. Erickson, V. Gangaram, A. Kapusta, C. K. Liu, and C. C. Kemp, "Assistive gym: A physics simulation framework for assistive robotics," in *International Conference on Robotics and Automation (ICRA)*, IEEE, 2020.

[67] Z. Erickson, H. M. Clever, G. Turk, C. K. Liu, and C. C. Kemp, "Deep haptic model predictive control for robot-assisted dressing," in *International Conference on Robotics and Automation (ICRA)*, IEEE, 2018, pp. 1–8.

[68] A. Clegg, W. Yu, Z. Erickson, J. Tan, C. K. Liu, and G. Turk, "Learning to navigate cloth using haptics," in *International Conference on Intelligent Robots and Systems*, IEEE/RSJ, 2017, pp. 2799–2805.

[69] W. Chen, H. Wang, Y. Li, H. Su, Z. Wang, C. Tu, D. Lischinski, D. Cohen-Or, and B. Chen, "Synthesizing training images for boosting human 3d pose estimation," in *Conference in 3D Vision*, IEEE, 2016, pp. 479–488.

[70] G. Varol, J. Romero, X. Martin, N. Mahmood, M. J. Black, I. Laptev, and C. Schmid, "Learning from synthetic humans," in *CVPR*, 2017, pp. 109–117.

[71] T. Yu, Z. Zheng, Y. Zhong, J. Zhao, Q. Dai, G. Pons-Moll, and Y. Liu, "Simulcap: Single-view human performance capture with cloth simulation," in *CVPR*, IEEE, 2019, pp. 5499–5509.

[72] Z. Pezzementi, E. Jantho, L. Estrade, and G. D. Hager, "Characterization and simulation of tactile sensors," in *IEEE Haptics Symposium*, 2017, pp. 199–205.

[73] A. Habib, I. Ranatunga, K. Shook, and D. O. Popa, "Skinsim: A simulation environment for multimodal robot skin," in *International Conference on Automation Science and Engineering*, 2014, pp. 1226–1231.

[74] B. Hollis, S. Patterson, J. Cui, and J. Trinkle, "Bubbletouch: A quasi-static tactile skin simulator," in *Artificial Intelligence for Human-Robot Interaction*, AAAI, 2018.

[75] D. C. Boone and S. P. Azen, "Normal range of motion of joints in male subjects," *Journal of Bone and Joint Surgery*, vol. 61, no. 5, pp. 756–759, 1979.

[76] A. Roaas and G. B. Andersson., "Normal range of motion of the hip, knee and ankle joints in male subjects, 30–40 years of age," *Acta Orthopaedica Scandinavica*, vol. 53, no. 2, pp. 205–208, 1982.

[77] J. M. Soucie, C. Wang, A. Forsyth, S. Funk, M. Denny, K. E. Roach, D. Boone, and H. T. C. Network, "Range of motion measurements: Reference values and a database for comparison studies," *Haemophilia*, vol. 17, no. 3, pp. 500–507, 2011.

[78] A. Ranjan, J. Romero, and M. J. Black, "Learning human optical flow," in *British Machine Vision Conference*, BMVA Press, 2018.

[79] A. Tözeren, *Human Body Dynamics: Classical Mechanics and Human Movement*. Springer, 1999.

[80] S. Payandeh and N. Azouz, "Finite elements, mass-spring-damper systems and haptic rendering," in *ICRA*, IEEE, pp. 224–229.

[81] N. Hansen, Y. Akimoto, and P. Baudis, *CMA-ES/pycma on Github*, Zenodo, DOI: 10.5281/zenodo.2559634, Feb. 2019.

[82] D. Berlowitz, C. VanDeusen Lukas, V. Parker, A. Niederhauser, J. Silver, C. Logan, E. Ayello, and K. Zulkowski, "Preventing pressure ulcers in hospitals: A toolkit for improving quality of care," Department of Health, Human Services, Agency for Healthcare Research, and Quality, Toolkit, Accessed 2021.

[83] S. Song, F. Yu, A. Zeng, A. X. Chang, M. Savva, and T. Funkhouser, "Semantic scene completion from a single depth image," in *CVPR*, IEEE/CVF, 2017, pp. 1746–1754.

[84] Z. Luo, J.-T. Hsieh, N. Balachandar, S. Yeung, G. Pusiol, J. Luxenberg, G. Li, L.-J. Li, N. L. Downing, A. Milstein, *et al.*, "Computer vision-based descriptive analytics of seniors' daily activities for long-term health monitoring," *Machine Learning for Healthcare (MLHC)*, vol. 2, 2018.

[85] O. Ronneberger, P. Fischer, and T. Brox, "U-net: Convolutional networks for biomedical image segmentation," in *International Conference on Medical image computing and computer-assisted intervention*, Springer, 2015, pp. 234–241.

[86] H. Rhodin, M. Salzmann, and P. Fua, "Unsupervised geometry-aware representation for 3d human pose estimation," in *Proceedings of the European Conference on Computer Vision (ECCV)*, 2018, pp. 750–767.

[87] M. Oberweger, P. Wohlhart, and V. Lepetit, "Training a feedback loop for hand pose estimation," in *ICCV*, IEEE, 2015, pp. 3316–3324.

[88] S. Mansfield, K. Obraczka, and S. Roy, "Pressure injury prevention: A survey," *IEEE reviews in biomedical engineering*, vol. 13, pp. 352–368, 2019.

[89] M. B. Pouyan, J. Birjandtalab, M. Nourani, and M. M. Pompeo, "Automatic limb identification and sleeping parameters assessment for pressure ulcer prevention," *Computers in biology and medicine*, vol. 75, pp. 98–108, 2016.

[90] R. Yousefi, S. Ostadabbas, M. Faezipour, M. Farshbaf, M. Nourani, L. Tamil, and M. Pompeo, "Bed posture classification for pressure ulcer prevention," in *2011 Annual International Conference of the IEEE Engineering in Medicine and Biology Society*, IEEE, 2011, pp. 7175–7178.

[91] S. Mansfield, S. Rangarajan, K. Obraczka, H. Lee, D. Young, and S. Roy, "Objective pressure injury risk assessment using a wearable pressure sensor," in *2019 IEEE International Conference on Bioinformatics and Biomedicine (BIBM)*, IEEE, 2019, pp. 1561–1568.

[92] D. Pickham, N. Berte, M. Pihulic, A. Valdez, B. Mayer, and M. Desai, "Effect of a wearable patient sensor on care delivery for preventing pressure injuries in acutely ill adults: A pragmatic randomized clinical trial (ls-hapi study)," *International journal of nursing studies*, vol. 80, pp. 12–19, 2018.

[93] K. Vanderwee, M. Clark, C. Dealey, L. Gunningberg, and T. Defloor, "Pressure ulcer prevalence in europe: A pilot study," *Journal of evaluation in clinical practice*, vol. 13, no. 2, pp. 227–235, 2007.

[94] N. A. Lahmann, R. J. Halfens, and T. Dassen, "Pressure ulcers in german nursing homes and acute care hospitals: Prevalence, frequency, and ulcer characteristics," *Ostomy Wound Management*, vol. 52, no. 2, p. 20, 2006.

[95] O. Ghori, R. Mackowiak, M. Bautista, N. Beuter, L. Drumond, F. Diego, and B. Ommer, "Learning to forecast pedestrian intention from pose dynamics," in *2018 IEEE Intelligent Vehicles Symposium (IV)*, IEEE, 2018, pp. 1277–1284.

[96] Y.-W. Chao, J. Yang, W. Chen, and J. Deng, "Learning to sit: Synthesizing human-chair interactions via hierarchical control," *arXiv preprint arXiv:1908.07423*, 2019.

[97] Y. Zhang, M. Hassan, H. Neumann, M. J. Black, and S. Tang, "Generating 3d people in scenes without people," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2020, pp. 6194–6204.

[98] J. Tan, T. Zhang, E. Coumans, A. Iscen, Y. Bai, D. Hafner, S. Bohez, and V. Vanhoucke., "Sim-to-real: Learning agile locomotion for quadruped robots," in *Robotics: Science and Systems*, IEEE, 2018.

[99] M. Cusumano-Towner, A. Singh, S. Miller, J. F. O'Brien, and P. Abbeel, "Bringing clothing into desired configurations with limited perception," in *ICRA*, 2011, pp. 3983–3900.

[100] J. Matas, S. James, and A. J. Davison, "Sim-to-real reinforcement learning for deformable object manipulation," in *arXiv preprint arXiv:1806.07851*, 2018.

[101] S. Pirk, V. Krs, K. Hu, S. D. Rajasekaran, H. Kang, Y. Yoshiyasu, B. Benes, and L. J. Guibas, "Understanding and exploiting object interaction landscapes," *ACM Transactions on Graphics (TOG)*, vol. 36, no. 3, pp. 1–14, 2017.

[102] H. Wang, S. Pirk, E. Yumer, V. G. Kim, O. Sener, S. Sridhar, and L. J. Guibas, "Learning a generative model for multi-step human-object interactions from videos," in *Computer Graphics Forum*, Wiley Online Library, vol. 38, 2019, pp. 367–378.

[103] A. Martınez-González, M. Villamizar, O. Canévet, and J.-M. Odobez, "Real-time convolutional networks for depth-based human pose estimation," in *2018 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, IEEE, 2018, pp. 41–47.

[104] Z. Zhong, L. Zheng, G. Kang, S. Li, and Y. Yang, "Random erasing data augmentation.," in *arXiv preprint arXiv:1412.6980*, 2017.

[105] B. Planche, Z. Wu, K. Ma, S. Sun, S. Kluckner, O. Lehmann, T. Chen, A. Hutter, S. Zakharov, H. Kosch, *et al.*, "Depthsynth: Real-time realistic synthetic data gen-

eration from cad models for 2.5 d recognition," in *2017 International Conference on 3D Vision (3DV)*, IEEE, 2017, pp. 1–10.

[106] S. Zakharov, B. Planche, Z. Wu, A. Hutter, H. Kosch, and S. Ilic, "Keep it unreal: Bridging the realism gap for 2.5 d recognition with geometry priors only," in *2018 International Conference on 3D Vision (3DV)*, IEEE, 2018, pp. 1–11.

[107] N. Kolotouros, G. Pavlakos, M. J. Black, and K. Daniilidis, "Learning to reconstruct 3d human pose and shape via model-fitting in the loop," in *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 2019, pp. 2252–2261.

[108] C. Zheng, W. Wu, T. Yang, S. Zhu, C. Chen, R. Liu, J. Shen, N. Kehtarnavaz, and M. Shah, "Deep learning-based human pose estimation: A survey," *arXiv preprint arXiv:2012.13392*, 2020.

[109] P. Grady, C. Tang, C. D. Twigg, M. Vo, S. Brahmbhatt, and C. C. Kemp, "ContactOpt: Optimizing contact to improve grasps," in *CVPR*, IEEE/CVF, 2021.

[110] X. Zhou, M. Zhu, G. Pavlakos, S. Leonardos, K. G. Derpanis, and K. Daniilidis, "Monocap: Monocular human motion capture using a cnn coupled with a geometric prior," *IEEE transactions on pattern analysis and machine intelligence*, vol. 41, no. 4, pp. 901–914, 2018.

[111] A. Harouni, A. Karargyris, M. Negahdar, D. Beymer, and T. Syeda-Mahmood, "Universal multi-modal deep network for classification and segmentation of medical images," in *2018 IEEE 15th International Symposium on Biomedical Imaging (ISBI 2018)*, IEEE, 2018, pp. 872–876.

[112] L. He, G. Wang, and Z. Hu, "Learning depth from single images with deep neural network embedding focal length," *IEEE Transactions on Image Processing*, vol. 27, no. 9, pp. 4676–4689, 2018.

[113] H. Fan, H. Su, and L. J. Guibas, "A point set generation network for 3d object reconstruction from a single image," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2017, pp. 605–613.

[114] L. Ballan, A. Taneja, J. Gall, L. Van Gool, and M. Pollefeys, "Motion capture of hands in action using discriminative salient points," in *European Conference on Computer Vision*, Springer, 2012, pp. 640–653.

[115] M. Matl, *Easy-to-use glTF 2.0-compliant OpenGL renderer for visualization of 3D scenes*, 2020.

[116] A. Ranjan, D. T. Hoffmann, D. Tzionas, S. Tang, J. Romero, and M. J. Black, "Learning multi-human optical flow," *International Journal of Computer Vision*, vol. 128, no. 4, pp. 873–890, 2020.

[117] R. Liu, J. Lehman, P. Molino, F. P. Such, E. Frank, A. Sergeev, and J. Yosinski, "An intriguing failing of convolutional neural networks and the coordconv solution," in *Advances in Neural Information Processing Systems*, 2018, pp. 9605–9616.

[118] T. F. Runia, K. Gavrilyuk, C. G. Snoek, and A. W. Smeulders, "Cloth in the wind: A case study of physical measurement through simulation," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2020, pp. 10 498–10 507.

[119] M. Fieraru, M. Zanfir, E. Oneata, A.-I. Popa, V. Olaru, and C. Sminchisescu, "Three-dimensional reconstruction of human interactions," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2020, pp. 7214–7223.

[120] M. Fieraru, M. Zanfir, E. Oneata, A.-I. Popa, V. Olaru, and C. Sminchisescu, "Learning complex 3d human self-contact," in *AAAI Conference on Artificial Intelligence (AAAI)*, vol. 3, 2021.

[121] L. Müller, A. A. Osman, S. Tang, C.-H. P. Huang, and M. J. Black, "On self-contact and human pose," in *CVPR*, IEEE/CVF, 2021, pp. 9990–9999.

[122] V. Choutas, G. Pavlakos, T. Bolkart, D. Tzionas, and M. J. Black, "Monocular expressive body regression through body-driven attention," in *European Conference on Computer Vision*, Springer, 2020, pp. 20–40.

[123] E. Aksan, M. Kaufmann, and O. Hilliges, "Structured prediction helps 3d human motion modelling," in *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 2019, pp. 7144–7153.

[124] M. Mihajlovic, Y. Zhang, M. J. Black, and S. Tang, "Leap: Learning articulated occupancy of people," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2021, pp. 10 461–10 471.

[125] J. Li, C. Xu, Z. Chen, S. Bian, L. Yang, and C. 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*, 2021, pp. 3383–3393.

[126] H. Shimokata, J. D. Tobin, D. C. Muller, D. Elahi, P. J. Coon, and R. Andres, "Studies in the distribution of body fat: I. effects of age, sex, and obesity," *Journal of gerontology*, vol. 44, no. 2, pp. M66–M73, 1989.

[127] A. Zengin, S. Pye, M. Cook, J. Adams, F. Wu, T. O'Neill, and K. Ward, "Ethnic differences in bone geometry between white, black and south asian men in the uk," *Bone*, vol. 91, pp. 180–185, 2016.

[128] K. M. Robinette, S. Blackwell, H. Daanen, M. Boehmer, and S. Fleming, "Civilian american and european surface anthropometry resource (caesar), final report. volume 1. summary," Sytronics Inc Dayton Oh, Tech. Rep., 2002.

[129] M. Black, private communication, 2021.

[130] I. Santesteban, E. Garces, M. A. Otaduy, and D. Casas, "Softsmpl: Data-driven modeling of nonlinear soft-tissue dynamics for parametric humans," in *Computer Graphics Forum*, Wiley Online Library, vol. 39, 2020, pp. 65–75.

[131] D. K. Pai, A. Rothwell, P. Wyder-Hodge, A. Wick, Y. Fan, E. Larionov, D. Harrison, D. R. Neog, and C. Shing, "The human touch: Measuring contact with real human soft tissues," *ACM Transactions on Graphics (TOG)*, vol. 37, no. 4, pp. 1–12, 2018.

[132] S. H. Sheen, E. Larionov, and D. K. Pai, "Volume preserving simulation of soft tissue with skin," *Proceedings of the ACM on Computer Graphics and Interactive Techniques*, vol. 4, no. 3, pp. 1–23, 2021.

[133] N. Ruiz, E. Chong, and J. M. Rehg, "Fine-grained head pose estimation without keypoints," in *Proceedings of the IEEE conference on computer vision and pattern recognition workshops*, 2018, pp. 2074–2083.

[134] T.-Y. Yang, Y.-T. Chen, Y.-Y. Lin, and Y.-Y. Chuang, "Fsa-net: Learning fine-grained structure aggregation for head pose estimation from a single image," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2019, pp. 1087–1096.

[135] R. Valle, J. M. Buenaposada, and L. Baumela, "Multi-task head pose estimation in-the-wild," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2020.

[136] J. Tobin, R. Fong, A. Ray, J. Schneider, W. Zaremba, and P. Abbeel, "Domain randomization for transferring deep neural networks from simulation to the real world," in *2017 IEEE/RSJ international conference on intelligent robots and systems (IROS)*, IEEE, 2017, pp. 23–30.

[137] J. Tremblay, A. Prakash, D. Acuna, M. Brophy, V. Jampani, C. Anil, T. To, E. Cameracci, S. Boochoon, and S. Birchfield, "Training deep networks with synthetic data: Bridging the reality gap by domain randomization," in *Proceedings of the IEEE conference on computer vision and pattern recognition workshops*, 2018, pp. 969–977.

[138] Y. Wang, W. Liang, J. Shen, Y. Jia, and L.-F. Yu, "A deep coarse-to-fine network for head pose estimation from synthetic data," *Pattern Recognition*, vol. 94, pp. 196–206, 2019.

[139] C. Doersch and A. Zisserman, "Sim2real transfer learning for 3d pose estimation: Motion to the rescue," *arXiv preprint arXiv:1907.02499*, 2019.

[140] A. Sengupta, I. Budvytis, and R. Cipolla, "Synthetic training for accurate 3d human pose and shape estimation in the wild," *arXiv preprint arXiv:2009.10013*, 2020.

[141] N. Hansen, S. D. Müller, and P. Koumoutsakos, "Reducing the time complexity of the derandomized evolution strategy with covariance matrix adaptation (cma-es)," *Evolutionary Computation*, vol. 11, no. 1, pp. 1–18, 2003.

[142] T. Wiedemeyer, *IAI Kinect2*, https://github.com/code-iai/iai_kinect2, Accessed June 12, 2015, University Bremen: Institute for Artificial Intelligence, 2014 – 2015.

[143] I. Akhter and M. J. Black, "Pose-conditioned joint angle limits for 3d human pose reconstruction," in *CVPR*, 2015.

[144] MandyMo, *PyTorch HMR - https://github.com/MandyMo/pytorch_HMR*, 2018.

VITA

Henry M. Clever was born on April 29, 1991, and resides in New York City.