Towards Comprehensive Human Representation: Sensor Modalities, Algorithms, and Applications

By

Sizhe An

A dissertation submitted in partial fulfillment of
the requirements for the degree of

Doctor of Philosophy

(Electrical and Computer Engineering)

at the

UNIVERSITY OF WISCONSIN–MADISON

2023

Date of final oral examination: 07/28/2023

The dissertation is approved by the following members of the Final Oral Committee:
Umit Y. Ogras, Associate Professor, Electrical and Computer Engineering, UW-Madison
Yu-Hen Hu, Professor, Electrical and Computer Engineering, UW-Madison
Younghyun Kim, Assistant Professor, Electrical and Computer Engineering, UW-Madison
Yin Li, Assistant Professor, Biostatistics & Medical Informatics, UW-Madison

Dedicated to my *parents Guangshe and Lili,* always my role models,
friends, who supported me all the time,
rejections, which made me stronger,
past-self, who did not give up halfway.

ACKNOWLEDGMENTS

First of all, I extend my deepest gratitude to my advisor, Prof. Umit Y. Ogras. From selecting me from the pool of ASU Ph.D. applicants to steadfastly guiding me throughout this journey, his guidance, wisdom, and encouragement have been instrumental. His insights in organizing ideas, paper-writing, time management, and fostering my ability for independent research will undoubtedly influence my future career. His willingness to allow me the flexibility to delve into my passionate research areas is something I deeply value. Our bond, cultivated over the years at the University of Wisconsin-Madison, will surely endure beyond this academic chapter.

I am immensely thankful to Prof. Yu-Hen Hu, Prof. Younghyun Kim, and Prof. Yin Li for taking the time to serve on my Ph.D. defense committee. Their invaluable feedback has been pivotal in elevating both my research and this dissertation. A special note of appreciation goes to Prof. Yin Li. Without auditing his class and seeking collaboration, the depth and caliber of my research would undoubtedly have been different.

I sincerely thank Dr. Hongyi Xu, Dr. Yichun Shi, Dr. Guoxian Song, Dr. Linjie Luo, and Dr. Suat Gumussoy for their supervision and our insightful research discussions. I genuinely appreciate our collaborative research discussions and their invaluable insights.

I also extend my sincere thanks to all mentors including Dr. Linguang Zhang, Dr. Amy Zhao, Dr. Kun He, Dr. Tomas Hodan, Dr. Zhaoyang Lv, Dr. Cem Keskin, Dr. Randi Cabezas, Dr. Robert Wang, and all other peers and friends for their input and support during my research internship at Meta Reality Lab.

Gratitude is also due to Dr. Hengyang Lu and Dr. Hua Lin, who provided constructive feedback during my very first internship.

The eLab has been a source of enriching discussions, motivation, and moments of camaraderie. To Dr. Samet Arda, Prof. Ganapati Bhat, Prof. Sumit Mandal, Dr. Yigit Tuncel, Dr. Anish Nallamur Krishnakumar, Dr. Woosoon Jung, Alper Goksoy, Toygun Basaklar, Shruti Narayana, Conrad Holt, Vishrut Pandey, Jie Tong, Nuriye

Yildirim, Xinmiao Xiong, Jiahao Lin, and to my ASU colleagues: Dr. Manqing Mao, Dr. Jingtao Li, Dr. Shunyao Wu, Yan Xiong, Jingyi Yuan, and Xing Chen—thank you.

I thoroughly enjoyed the words of encouragement and lighter moments with my friends even during the coldest and longest winter at Madison: Qiyuan Chen, Huan Liang, Changxing Shi, Jiren Sun, Nathan Su, Zhoujingpeng Wei, and of course Calvin, Yzz, Adrian, Zaixun, Ningge, Wsf, Huge, Ryan, and all other dear basketball mates! Special thanks to my middle school gaming squads and high school bros, who helped pull me through the frustration of numerous consecutive paper rejections in my early days.

My deepest gratitude goes to Prof. John S. Ho and Prof. Atif Shamim for planting the seeds of pursuing a research career during my undergraduate days.

Finally, this would not have been possible without the support of my parents, Guangshe and Lili, grandparents Kuiying, Ailian, Chengke, Tingkui, and other family members—though the U.S. visa restrictions kept me afar since 2019, your love, encouragement, and belief in me have been my bedrock.

CONTENTS

LIST OF TABLES

LIST OF FIGURES

ABSTRACT

Recent advances in emerging Internet-of-Things (IoT) devices, Artificial Intelligence (AI), and machine learning (ML) algorithms have paved the way for sophisticated human representation. Human shapes and actions can be sensed and interpreted through various sensor modalities, enabling high-impact applications. However, the choice of sensing modality directly affects the quality and accuracy of the captured human representation. Therefore, it is critical to investigate various sensing modalities and algorithm choices and adapt suitable human representation to target applications. Moreover, the complexity and diversity of human representation also necessitate exploring advanced computational models capable of generating realistic and detailed human features. By employing generative models, we can effectively simulate and manipulate human attributes, enabling us to generate novel samples and explore the vast space of possible human representations. This opens up new opportunities for studying and understanding the intricate complexities and diversities of human representation. To better study and interpret these varied human representations, we conduct research in the following main areas:

(i) We explore the potential of emerging sensing modalities such as inertial sensors, RGB-Depth cameras, and millimeter-Wave (mmWave) radar in applications including gait monitoring, human activity recognition, and human pose estimation. In this regard, we present MGait: a model-based gait monitoring technique, MARS: mmWave-based human pose estimation for rehabilitation, and mRI: a multi-modal 3D human pose estimation dataset.

(ii) We design algorithm pipelines to pre-process the raw sensing signals effectively, train and test deep learning models. Smart algorithm design choices ensure the models are lightweight and can adapt well to unseen scenarios. To this end, we propose Fuse: fast and scalable human pose estimation using mmWave point cloud, and a transfer learning algorithm for human activity recognition.

(iii) Finally, we propose Panohead: a 3D generative model that enable high-quality, view-consistent image synthesis of human heads in 360°. This work extends the 3D human head modeling capabilities of generative adversarial networks

(GANs) from a limited frontal view to a full 360° view, opening up new possibilities for novel 3D human representation and downstream tasks such as style-mixing and inferencing the full-head from a single facial image.

1 INTRODUCTION

The accelerated growth of emerging technologies, including Internet-of-Things (IoT) devices, Artificial Intelligence (AI), and machine learning (ML) algorithms, has kindled the evolution of sophisticated forms of human representation. The concept of 'human representation' refers to the capture and description of various human attributes and behaviors through sensor data. Notably, these representations can encapsulate multiple aspects of human beings, such as body posture, facial expressions, behavior patterns, and psychological states. Such characterizations enable machines to better comprehend and interpret human behavior, fostering more natural and effective human-machine interactions.

The different facets of human form and activities can be sensed and deciphered through numerous novel sensor modalities, including RGB-D cameras [7, 8, 9, 10, 11, 12], inertial sensors [13, 14, 15, 16, 17, 18], and millimeter-Wave (mmWave) radar [19, 20, 21, 22]. These sensory data generate rich human representations that fuel high-impact applications, from 3D human generation [5, 4] and human pose estimation [11, 18, 19] to early disease diagnosis and prognosis via constant monitoring of vital signals [23].

Healthcare emerges as a leading field of application for human representation, a fact underscored by the global increase in the aging population and the parallel rise in health-related issues. These trends have ignited a surge of research interest within both industrial [24, 25] and academic circles [26, 27]. Major technological firms have, in response, significantly increased their R&D investment into wearable devices intended for mobile activity and health monitoring. As an illustration,

Apple integrates health-related features into their widely used products like the Apple Watch, thereby bridging the gap between wearable technology and clinical research tools [25]. Moreover, Google's acquisition of Fitbit, a leading wearable technology company, in 2021 showcases the burgeoning race among major tech companies to establish a foothold in AI-enabled healthcare [24].

Remote activity and health monitoring applications, predicated on emerging human representation, have the potential to provide invaluable insights [23]. Consequently, these applications can enhance people's quality of life, by utilizing applications across a broad spectrum of healthcare domains, encompassing but not limited to electroencephalogram (EEG) [28, 29, 30, 31], human activity recognition [32, 33], gait monitoring [34, 35], and human pose estimation [36, 37, 21]. Advanced machine learning algorithms can analyze motion and physiological data collected from wearable sensors locally and in real-time, thus enabling applications like irregular rhythm notifications, early warning sign detections, and fall monitoring. Beyond their immediate benefits, these applications also serve as initial steps towards diagnosing, prognosing, and rehabilitating movement disorders such as Parkinson's Disease (PD) and stroke [23]. The potential of these tools in revolutionizing personalized and preventative healthcare systems is immense.

Despite the promising potential, the emerging field of human representation faces significant challenges that need to be addressed. These challenges can be categorized into two main areas: sensing modalities and algorithm designs.

The first challenge lies in the diverse range of sensing modalities available for capturing human representation. There are intrusive modalities, such as wearable

devices with inertial measurement units (IMUs), as well as non-intrusive modalities, such as RGB cameras or millimeter-wave radar. Each modality has its own precision, environmental requirements, and power consumption characteristics. Therefore, it becomes crucial to explore and identify the most suitable modalities for specific applications. The choice of sensing modality directly affects the quality and accuracy of the captured human representation, making it essential to investigate the trade-offs between different modalities and their compatibility with target use cases.

The second challenge pertains to algorithm design, particularly in scenarios where the applications need to run on edge devices. In such cases, the chosen algorithms should be lightweight to ensure efficient execution on resource-constrained devices. Additionally, these algorithms should possess the capability to generalize well to unseen user data, as training specific users' data for edge devices is often challenging. Therefore, the algorithms need to be smart and adaptable, enabling them to learn from limited training data and generalize effectively to a larger user population. Striking the right balance between lightweight processing and generalizability is crucial to ensure the successful deployment of human representation applications on edge devices.

Beside the two main challenges, another crucial aspect to consider is the exploration of generative models, which play a vital role in enhancing our understanding and capabilities in representing human representation. The applications mentioned earlier, such as human activity recognition and human pose estimation, primarily fall under the category of discriminative models. These models focus on learning the relationship between input data (sensor measurements) and corresponding

outputs (activity labels or pose estimates). They excel at classifying or predicting specific human behaviors or attributes based on observed patterns in the data. However, to gain a deeper understanding of human representation, it is equally crucial to explore generative models. Generative models aim to capture the underlying distribution of the data and generate new samples that are indistinguishable from the real data. In the context of human representation, generative models can play a significant role in uncovering the latent factors that shape human features and behaviors, and in generating realistic and detailed human representations that go beyond what can be directly observed in the data. Furthermore, generative models can enhance our ability to handle missing or incomplete data, as they can fill in gaps and generate plausible completions based on learned representations.

1.1 Contributions

Novel applications, such as human gait monitoring and pose estimation, have emerged through the utilization of emerging sensing modalities. These advancements enable innovative approaches to remote healthcare, providing valuable insights into human movements and facilitating personalized care. This dissertation explores three high-impact healthcare applications leveraging diverse sensing modalities.

Movement disorders, such as Parkinson's disease, affect more than 10 million people worldwide [38]. Gait analysis is a critical step in the diagnosis and rehabilitation of these disorders. Specifically, step and stride lengths provide valuable

insights into the gait quality and rehabilitation process. However, traditional approaches for estimating step length are not suitable for continuous daily monitoring since they rely on special mats and clinical environments. To address this limitation, this dissertation presents MGait, a novel and practical step-length estimation technique using low-power wearable bend and inertial sensors. Experimental results show that the proposed model estimates step length with 5.49% mean absolute percentage error and provides accurate real-time feedback to the user.

Rehabilitation plays a vital role in motor disorder treatment, typically performed under clinical expert supervision [39, 40]. To overcome challenges related to commuting, expert shortages, and healthcare costs, innovative solutions are required to enable patients to perform prescribed exercises at home. Human pose estimation (HPE) is a substantial component of these programs since it offers valuable visualization and feedback based on body movements. Camera-based systems have been popular for capturing joint motion. However, they have high-cost, raise serious privacy concerns, and require strict lighting and placement settings. This dissertation proposes MARS, a mmWave-based assistive rehabilitation system for motor disorders to address these challenges. MARS provides a low-cost solution with a competitive object localization and detection accuracy. It first maps the 5D time-series point cloud from mmWave to a lower dimension. Then, it uses a convolution neural network (CNN) to estimate the accurate location of human joints. MARS can reconstruct 19 human joints and their skeleton from the point cloud generated by mmWave radar. We evaluate MARS using ten specific rehabilitation movements performed by four human subjects involving all body parts and obtain

an average mean absolute error of 5.87 cm for all joint positions.

A range of sensors, including RGB-D cameras [7, 8, 9, 10, 11, 12], mmWave radars [19, 20, 21, 22], and wearable inertial sensors [13, 14, 15, 16, 17, 18], have been extensively explored for HPE. Despite previous efforts on datasets and benchmarks for HPE, few dataset exploits multiple modalities and focuses on home-based health monitoring. To bridge this gap, this dissertation presents mRI, a multi-modal 3D human pose estimation dataset with mmWave, RGB-D, and Inertial Sensors. Our dataset consists of over 160k synchronized frames from 20 subjects performing rehabilitation exercises and supports the benchmarks of HPE and action detection. We perform extensive experiments using our dataset and delineate the strength of each modality. We hope that the release of mRI can catalyze the research in pose estimation, multi-modal learning, and action understanding, and more importantly facilitate the applications of home-based health monitoring.

In the pursuit of achieving advanced human representation, algorithm designs play a crucial role in extracting meaningful insights from the available sensor data. Specifically, these algorithms need to exhibit energy efficiency to ensure their feasibility on edge devices. Additionally, they must possess the intelligence and adaptability to learn from limited training data and effectively generalize their understanding to a broader user population. This dissertation highlights two algorithm designs that pave the way for more efficient and accurate human representation in various applications.

The use of mmWave radar technology holds great promise for high-resolution human pose estimation, offering cost-effective and computationally efficient capa-

bilities [19, 20, 21, 22]. However, mmWave data point cloud, the primary input for processing algorithms, is inherently sparse and contains less information compared to alternative sources like video frames. To address these challenges, the dissertation presents FUSE, a fast and scalable human pose estimation framework. FUSE leverages a combination of multi-frame representation and meta-learning techniques, enabling efficient and accurate estimation of human joint coordinates. Experimental evaluations demonstrate that FUSE outperforms current supervised learning approaches, adapting to unseen scenarios approximately four times faster and achieving accurate human pose estimates with minimal error.

Traditionally, human activity recognition (HAR) involves training a classifier offline using known user data and applying the same classifier to new users [41]. However, this approach often results in reduced accuracy for new users with different activity patterns. To address this challenge, the dissertation introduces a transfer learning framework for HAR. The framework leverages representational analysis to identify common features that can be transferred across users, allowing for improved generalizability. By transferring the reusable portion of the offline classifier to new users and fine-tuning the user-specific features, the framework achieves significant accuracy improvements and reduces training time. Moreover, the proposed framework demonstrates reduced power and energy consumption while maintaining or surpassing the accuracy achieved by training from scratch.

Generative models serve as a valuable approach to enriching the comprehension of human representation, complementing the insights derived from discriminative methods. Synthesis and reconstruction of 3D human face and head has gained

increasing interests in computer vision and computer graphics recently [42, 5, 4]. Existing state-of-the-art 3D generative adversarial networks (GANs) for 3D human head synthesis are either limited to near-frontal views or hard to preserve 3D consistency in large view angles. This dissertation proposes PanoHead, the first 3D-aware generative model that enables high-quality view-consistent image synthesis of full heads in 360° with diverse appearance and detailed geometry using only in-the-wild unstructured images for training. At its core, we lift up the representation power of recent 3D GANs and bridge the data alignment gap when training from in-the-wild images with widely distributed views. Specifically, we propose a novel two-stage self-adaptive image alignment for robust 3D GAN training. We further introduce a tri-grid neural volume representation that effectively addresses front-face and back-head feature entanglement rooted in the widely-adopted tri-plane formulation. Our method instills prior knowledge of 2D image segmentation in adversarial learning of 3D neural scene structures, enabling compositable head synthesis in diverse backgrounds. Benefiting from these designs, our method significantly outperforms previous 3D GANs, generating high-quality 3D heads with accurate geometry and diverse appearances, even with long wavy and afro hairstyles, renderable from arbitrary poses. Furthermore, we show that our framework can reconstruct full 3D heads from single input images for personalized realistic 3D avatars.

In summary, this dissertation makes the following contributions:

- Present MGait, a model-based gait monitoring technique [34],
- Present MARS, a mmWave-based human pose estimation framework for

rehabilitation [36],

- Present mRI, an open-source multi-modal 3D human pose estimation dataset [43],
- Present FUSE, fast and scalable human pose estimation using mmWave point cloud [37],
- Present a transfer learning algorithm using representation analysis for human activity recognition [33], and
- Present Panohead, a 3D generative model for 360° human head synthesis [44]

The rest of the dissertation is organized as follows. The literature survey is discussed in Chapter 2. Chapter 3 presents MGait, the model-based gait analysis using wearable bend and inertial sensors. MARS and mRI are presented in Chapter 4 and Chapter 5, respectively. FUSE and transfer learning algorithm for HAR are presented in Chapter 6 and Chapter 7, respectively. Chapter 8 presents Panohead, a novel approach to synthesize and generate 3D human head using generative models and neural radiance field. Finally, Chapter 9 concludes this dissertation with directions for future work.

2 LITERATURE REVIEW

2.1 Research Intersection of Modalities, Algorithms, and Applications in Human Representation

To fully unlock the potential of emerging human representation, wearable technology serves as a crucial enabler, given its agility in sensing, real-time processing capabilities, and user-friendly nature. This notion is further substantiated by breakthroughs in both academia [45, 27, 26, 23] and industry [25, 24]. However, the constraints of wearable devices, such as their small size and cost limitations, significantly limit battery capacity, making energy a major hurdle for wearable technology to become ubiquitous. Consequently, low power design and effective energy harvest&management [46, 47, 48, 49, 50, 51] are significant challenges to the widespread adoption of wearable devices. Although not directly addressed in this dissertation, the role of wearable technology and its energy management are essential to realize the broader vision of comprehensive human representation.

This dissertation seeks to provide a comprehensive exploration of emerging trends in human representation, focusing on three interconnected domains: sensing modalities, algorithms, and applications. At its core, human representation encapsulates various facets of our humanity, ranging from body posture and facial expressions to behavior patterns and psychological states. To capture these rich and insightful representations, it is critical to harness and effectively process data from various innovative sensing modalities. These include RGB-Depth cameras,

inertial sensors, and wireless sensor technologies such as mmWave radar.

Once human form and activities are transformed into digital representations, they serve as the catalyst for a multitude of innovative and impactful human-centric applications. Notable among these are human activity recognition [27], gait recognition [52, 35], and pose estimation [11, 18, 53]. By offering insights into early disease diagnosis and prognosis, these applications hold the potential to significantly enhance the quality of life for individuals [23].

Powering these processes and applications necessitate the development and application of intelligent, lightweight, and generalizable algorithms. We focus on transfer learning [54] and meta-learning [55], given their demonstrated effectiveness and widespread popularity in the field.

This chapter will expand the literature reviews into three focused sections reflecting these three domains. We will begin by charting the development of emerging sensing modalities, followed by a discussion on the role of smart algorithms, with an emphasis on transfer learning and meta-learning. The final section will delve into high-impact applications, spotlighting human activity recognition, gait recognition, pose estimation, and the generation of 3D human. By doing so, we aim to illuminate the profound interconnections among sensing modalities, algorithms, and applications in the broad context of human representation.

2.2 Emerging Sensing Modalities

Vision

Computer Vision (CV) offers an accurate representation of the real world using true-color images or videos. The primary goal of using vision-based approaches in human-centered application is to help machine understand human actions and activities. RGB video camera [56, 57], depth camera [58, 59, 60], and motion capture system [57, 60] are the major devices have been used in this area. In [56], Ar et al. present Home-based Physical Therapy Exercises (HPTE) dataset which targets therapy actions. The Kinect camera is used in this study to provide video and depth streams to the user. One of the outputs of this approach is the binary image that indicates the body shape. They give eight shoulder and exercise movements but without any joint or skeleton information. In 2015, researchers developed a system and released a dataset named EmoPain that has both body joint information and face videos [57]. These systems use RGB cameras often has limitation due to environmental noise, and lens distortion [61].

Besides the RGB video-based approach, Microsoft Kinect and Kinect V2 [61] provide depth cameras to extract the human joints information. Kinect uses an RGB and infra-red camera, while Kinect V2 uses a Time of Flight (ToF) camera to capture the information. The Kinect family has become one of the popular ways to obtain the ground truth label for training due to its convenience, low cost, and accurate performance [58, 19, 22].

Inertial Sensors

Wearables like inertial sensors (IMUs) play an essential role in human sensing. IMU is relatively robust to different environmental settings since sensing is not interfered with by light conditions or visibility. Thus, it is more practical for occlusions or baggy clothing scenarios. In addition, the explainability of IMUs-based method is promising since every IMU is placed in a fixed position of a person, thus accounting for specific limbs' movements. As one of the earliest studies in this field, [62] estimate human pose using 17 IMUs, and a Kalman filter is employed for all the measurements. It comprehensively defined 17 IMUs on a person, thus achieving accurate human pose estimation. However, the large number of IMUs require long setup times and make it uncomfortable for users. Marcard et al. proposed Sparse Inertial Poser (SIP) [16]: Automatic 3D Human Pose Estimation from Sparse IMUs. This work provides a new method to estimate the human pose using only six IMUs. By exploiting a statistical body model and jointly optimizing posture over continuous time frames to fit both orientation and acceleration data, SIP achieves positional errors of 3.9 cm. A follow-up work [15] combines IMUs and a moving camera to estimate multiple human poses in challenging outdoor scenes robustly. Wang et al. propose a biomechanical model that takes knee bending into account and uses data obtained from a total of four low power IMUs placed on both legs to estimate step length and gait asymmetry [63].

Wireless Sensors

Unlike wearables, wireless sensors using Wi-Fi, Radio-Frequency, and mmWave are non-intrusive, which is more user-friendly. Early applications of wireless sensors focused on classification [64], localization [65], and obstacle detection [66] problems that do not require a high resolution. For example, a mmWave radar-based indoor human activity recognition technique is proposed in [64]. It recognizes five different activities: boxing, jumping, jumping jacks, squats, and walking with more than 90% accuracy. Sugimoto et al. [66] present an obstacle detection method consisting of occupancy-grid representation and a segmentation method that divides the radar data. Similarly, Lemic et al. [65] propose a localization system that determines a mobile node's location using the flight time and arrival angles obtained by all the mmWave devices. In summary, early wireless sensors only requires a coarse-grained representation due to applications' simplicity.

However, it requires a fine-grained representation when we aims to reveal the nature of human motion (e.g., 3D joint coordinates) for emerging applications. Zhao et al. [19] propose a technique that uses radio-frequency (RF) antenna arrays reconstructs up to 14 body parts, including head, neck, shoulders, elbows, wrists, hip, knees, and feet. They compute 4D (time and three spatial axes) RF tensors using a 64-element antenna array with 60cm $\times$ 18cm area. The massive customized antenna arrays enrich the input representation, but the large size and high cost significantly hinder practicality.

Recent mmWave radar-based pose estimation techniques use point cloud representation from commercial radar devices like Texas Instrument (TI) xWR1x43.

Sengupta et al. [22] propose mmPose, a human pose estimation technique that constructs the skeleton using mmWave point cloud and a forked-CNN architecture. They use two radar devices and sum up the point values in the feature map level to overcome the sparse representation of the point cloud. Xue et al. [21] present the mmMesh technique to construct human mesh using mmWave point cloud. It employs a human shape model to strengthen the ability of deep learning models to predict human shape with fewer points. However, none of the prior techniques address a fundamental problem: sparsity of mmWave point cloud data. Our proposed work FUSE (Chapter 6) adequately resolves the issue by temporally fusing the mmWave point cloud.

2.3 Smart Algorithms

Meta Learning

Meta-learning has recently gained momentum because it can help ML models adapt to unseen scenarios faster with a few training iterations [55, 67, 68]. The meta-learning concept was first proposed in [69]. It focuses on learning a strategy that generalizes to related yet unseen tasks from similar task distributions. It is first trained with a batch of tasks and learning rules designed to facilitate learning new tasks using only a few training iterations. In this way, the model employs the parameters sensitive to new samples, expediting generalization to new tasks. A variety of approaches focus on learning the best initialization of the network then fine-tuning it with the new task [55, 67, 68]. Finn et al. propose a popular

algorithm MAML [55], which tries to find the optimal initialization of the network directly. The idea of MAML is that model is trained on a batch of tasks, such that it can optimize model parameters sensitive to new tasks with very few data samples. MAML is model-agnostic, and it does not introduce any learned parameters for meta-learning. It is soon followed by [67, 68]. FOMAML [67] is a variant of MAML. It ignores the second derivative terms used in MAML's gradient descent algorithm. Reptile [68] is very similar to FOMAML in the sense that it also only uses first-order gradient information. But it does not require a training-test split for each task. A few studies have recently applied meta-learning to point cloud [70, 71]. Puri et al. [70] use MAML to solve the point cloud-based object classification and show it achieves similar accuracy with fewer data samples. Similarly, Li et al. [71] propose few-shot meta-learning on point cloud for indoor semantic segmentation. However, both studies focus on the lidar point cloud, and their applications only involve simple classification and semantic segmentation. Our proposed FUSE (Chapter 6) is work applying meta-learning to human pose estimation using mmWave point cloud.

Transfer Learning

Transfer learning aims to leverage the information learned in one domain to improve the accuracy in a new domain [72]. The information used for transfer includes the weights of a classifier [73, 74, 75], features [76], and instances of data [77]. The transfer can occur between different applications or between different scenarios of a single application [72]. A popular example of transfer learning between applica-

tions is medical imaging [78, 79] where CNNs trained for classifying ImageNet [80] are adapted to classify medical images. Similarly, transfer learning for new scenarios includes adaptation to a new device [81], classes [82], domains [30, 83] or users [84].

One of the fundamentals aspects of transfer learning is identifying what information to transfer. Prior work addressed parameter transfer [85], feature representation transfer [86, 87], and data instance transfer [77]. We focus on parameter transfer literature since the proposed approach uses parameter transfer as well. Oquab et al. [85] design a method to reuse layers trained with one dataset to compute mid-level image representation for images in another dataset. Yosinski et al. [88] empirically quantify the generality versus specificity of neurons in each layer of a deep convolutional neural network for the ImageNet [80] dataset. They show that the features in the initial layers are general in that they are applicable to multiple image recognition tasks. Morcos et al. [89] directly analyze the hidden representations of each layer in CNN by using canonical correlation analysis (CCA), which enables comparing learned distributed representations between different neural networks layers and architectures.

Transfer learning has been applied successfully in fields such as medical imaging classification and computer vision [79, 90, 91, 92]. Salem et al. [79] present an approach to transfer a CNN from image classification to electrocardiogram (ECG) signal classification domain. Similarly, Raghu et al. [90] explore properties of transfer learning from natural image classification networks to medical image classification.Quattoni et al. [91] show that prior knowledge from unlabeled data

is useful in learning a new visual category from few examples. The authors develop a visual-category learning algorithm called sparse prototype learning that can learn an efficient representation from a set of related tasks while taking advantage of unlabeled data.

One of the challenges in human activity recognition (HAR) algorithms is that the data available at design time may not be representative of the activity patterns of new users. As a result, the accuracy can degrade for new users [93]. Recent research has used transfer learning to address this issue [94, 54, 93]. The survey by Cook et al. [54] presents how different types of transfer learning have been used for HAR. Ding et al. [93] perform an empirical study to analyze the performance of transfer learning methods for HAR and find that maximum mean discrepancy method is most suitable for HAR. A CNN-based method to transfer learned knowledge to new users and sensor placements is presented in [95]. The authors empirically determine the number of layers to transfer based on the accuracy obtained after transferring. However, this method is not scalable since the training has to be repeated each configuration of the transfer. Rokni et al. [94] use transfer learning to personalize a CNN classifier to each user by retraining the classification layer with new users. However, the authors do not provide any insight into the number of layers that can be transferred between users and how it benefits the learning for new users. In contrast, our proposed work (Chapter 7) perform representational analysis using CCA to determine layers that need to be fine-tuned for new users.

2.4 Novel Applications

Human Gait Recognition

Gait analysis is vital for enabling a variety of cyper physical systems (CPS) applications, such as feedback for patients under treatment, prediction of possible movement disorders [96, 97], athlete assessment, fall detection [98], and numerous other augmented and virtual reality applications. For this reason recent work has focused on using wearable sensors to estimate step length, which is a very important parameter in gait analysis [99, 63, 100, 101, 102, 103].

Table 2.1 summarizes state-of-the-art step length estimation techniques that are closest to our work. Wu et al. place two IMU dev-kits on the ankles of the subjects to estimate step length, using an inverted pendulum model with 3.69% MAPE [100]. Their model does not consider knee and hip joints. Instead, it employs ankle-mounted motion sensors to calculate the step length using the leg length and the sine of the leg's orientation. However, the error reported in their study is for the total walking distance, not for individual step lengths. Since positive and negative errors in individual step lengths cancel out, this reporting choice shows lower MAPE. For example, the proposed MGait technique MAPE decreased by 1% to 2% if we consider the total walking distance. Since the model assumes a single segment for each leg, it is able to calculate the step length using only two IMUs. Consequently, the power consumption of this approach is lower, compared to other more complicated approaches. Their approach also requires a Kinect V2 setup to estimate leg lengths of the subjects in their study. We do not include it

Table 2.1: A compilation of previous related work on gait analysis ("–" means the results are not reported)

Ref	Step Length Err.		Stride Length Err.		Velocity Err.		Sensor Type	Sensor Count	User Feed.	Wearable Form-factor	Power (mW)
	RMSE	MAPE	RMSE	MAPE	RMSE	MAPE					
[63]	5.5 cm	-	-	-	-	-	MPU-6050 IMU	4	✗	✓	36.10
[100]	-	3.69%	-	-	-	-	MPU-9150 IMU	2	✗	✗	18.53
[102]	-	<10%	-	-	-	-	Smartphone accel.	1	✗	✗	-
[99]	4.1 cm	4.75%	6.3 cm	3.70%	0.07 m/s	4.23%	*High Cost* UWB, MPU-6050 IMU	2+2	✗	✓	145.20
[103]	-	<5%	-	<5%	-	-	MPU-6050 IMU, MPU-9250 IMU	3+4	✗	✓	64.13
GAITRite [104]	-	<2%	-	<1%	-	<1%	GR* Proprietary	>10k	✗	✗	-
***MGait* (Chapter 3)**	**4.3 cm**	**5.47%**	**6.24 cm**	**3.90%**	**0.03 m/s**	**2.33%**	**MPU-9250 IMU, Bendlabs 1-Axis**	**2+2**	✓	✓	**16.97**

in the sensor type since it does not contribute to the model calculation. Pepa et al. utilize accelerometers in smartphones and develop an app to collect motion data. They also use an inverted pendulum model and estimate step length with less than 10% MAPE [102]. None of these studies provide stride length and gait velocity estimates, which are two other main parameters used for gait function analysis [52].

A recent work addresses the limitations in the previous studies, by using four *high-cost ultra-wideband* (UWB) *distance sensors* and four IMUs on the back and front of both feet [99]. It employs a geometrical trapezoid distance model and estimates step length with 4.1 cm RMSE and 4.75% MAPE. However, the high processing requirements of the data from eight sensors increase the complexity and the power consumption of the system. Another recent study places two IMUs on the feet, two IMUs on the shanks, two IMUs on the thighs, and one IMU on the pelvis, using a total of seven IMUs [103]. It uses a Kalman filter framework to calculate the orientation of each sensor. Since they use seven IMUs, the proposed approach is not practical, and the power consumption of the system is high, due to a large

number of sensors and their processing cost.

In summary, none of these studies targets real-time feedback to the user and considers the system power consumption. In contrast to all of these approaches, our proposed *MGait* (Chapter 3) estimates the key gait parameters with the minimum number of IMUs and low-power bend sensors (one on each leg), and a low power micro-controller using a novel closed-form expression. It achieves accurate step length, stride length, and gait velocity. Low power consumption and processing requirements of *MGait* can enable the first low-cost, low power, and wearable CPS for gait analysis with user-feedback functionality.

Human Activity Recognition

Human activity recognition is a critical component in a range of health and activity monitoring applications [105, 106, 107]. It provides valuable insight into movement disorders by allowing health professionals to monitor their patients in a free-living environment [108, 109, 110]. HAR is also the first step towards understanding gait parameters, such as step length and gait velocity, which are also used in movement disorder analysis and rehabilitation [34, 111, 112, 113]. In addition, HAR is used for obesity management and promoting physical activity among the public. Due to these high-impact applications of HAR, it has received increased research attention in recent years [114, 115]. One of the challenges in HAR algorithms is that the data available at design time may not be representative of the activity patterns of new users. As a result, the accuracy can degrade for new users [93].

Human Pose Estimation

3D Human pose estimation (HPE) refers to detecting and tracking human body parts or key joints (e.g., wrists, shoulders, and knees) in the 3D space. It is a fundamental and crucial task in human activity understanding and movement analysis with numerous application areas, including rehabilitation [60, 116, 56, 58], professional sports [117], augmented/virtual reality, and autonomous driving [118].

Marker-based optical motion capture (MoCap) systems are often used to acquire accurate 3D body pose [9, 57, 60]. Optical MoCap systems require attaching reflective markers to the body and are quite costly, thus are limited to laboratory settings. Recently, MoCap systems based on body-worn IMUs have been developed [14, 17, 16, 13, 119]. They are considerably cheaper yet at a cost of tracking accuracy due to drifting [120].

Besides marker-based MoCap, Marker-less MoCap has received much attention. Depth cameras are often used for pose estimation [121], yet are limited by their sensing range (within 5 meters). Recent effort has focused on pose estimation using RGB cameras. With the help of machine learning, 3D joints can be estimated from a single RGB image [122], or from several RGB images from different viewing angles captured by multiple cameras [11, 12, 17], or from a sequence of RGB frames within a video [123]. However, RGB cameras are easily affected by poor light conditions, and raise privacy concerns for home-based monitoring. More recently, mmWave-based pose estimation, including radio frequency sensing, has emerged as a promising solution [20, 19, 22, 21, 36]. A mmWave-based solution has demonstrated comparable accuracy to RGB and depth cameras, yet excels at

privacy-preserving and long working range.

High-quality datasets with annotations are crucial for the advancement of pose estimation. Table 2.2 summarizes mainstream HPE datasets. Some of the early effort focuses on 2D HPE (e.g., COCO [7] and MPII [8]), or 3D HPE a single modality (e.g., 3DHP with images, mmPose [22] with mmWave, and MPI08 [17] with IMU). More recent works combines multiple modalities for 3D HPE. For example, Human3.6M [9] contains RGB images and depth maps of 11 professional actors performing 17 daily activities, coupled with ground-truth 3D poses from optical MoCap. RF-Pose3D [19] presents the first study to use radio frequency sensing for 3D HPE, together with a dataset of both RGB images and radio signals. MoVi [119] incorporates both IMU signals and RGB frames, as well as ground-truth 3D poses from MoCap, and presents a benchmark for both 3D HPE and human activity recognition.

To the best of our knowledge, our proposed mRI (Chapter 5) is the *first HPE dataset with the most comprehensive set of sensing modalities*, including RGB, depth, IMU, and mmWave. In addition, mRI *fills the vacancy of standardized mmWave-based human pose estimation*, as all current mmWave-based HPE datasets are either not open-sourced or without proper keypoints annotations and RGB references.

Moving forward, the results of 3D HPE can be used by skeleton-based action recognition [124, 10] to localize and recognize actions in time, broadening its applications in health monitoring [125, 126] and human behavior analysis [127]. Our dataset provides action annotations and we evaluate using the estimated pose for temporal action localization [128].

Table 2.2: Summarization of relevant HPE datasets

Dataset	Sensing Modalities				# of	# of	# of	# of	Annotations		
	RGB	Depth	IMU	mmWave	Subjects	Seqs	Actions	Synced Frames	Action	2DKP	3DKP
COCO [7]	✓	-	-	-	-	-	-	104k	-	✓	-
MPII [8]	✓	-	-	-	-	24k	410	25k	✓	✓	-
MPI-INF-3DHP [12]	✓	-	-	-	8	16	8	1.3M	-	✓	✓
Human3.6M [9]	✓	✓	-	-	11	839	17	3.6M	✓	✓	✓
CMU Panoptic [11]	✓	✓	-	-	8	65	5	154M	-	✓	✓
NTU RGB+D [10]	✓	✓	-	-	40	56k	60	4M	✓	✓	✓
3DPW [14]	✓	-	✓	-	7	60	-	51k	-	✓	✓
MPI08 [17]	✓	-	✓	-	4	24	24	14k	-	-	✓
TNT15 [13]	✓	-	✓	-	1	-	5	14k	✓	-	✓
MoVi [119]	✓	-	✓	-	90	1044	21	712k	✓	✓	✓
RF-Pose [20]†	✓	-	-	✓	100	-	1	-	✓	✓	-
RF-Pose3D [19]†	✓	-	-	✓	>5	-	5	-	✓	✓	✓
mmPose [22]†	-	-	-	✓	2	-	4	40k	✓	-	✓
mmMesh [21]†	✓	-	-	✓	20	-	8	3k	✓	-	✓
MARS [36]	-	-	-	✓	4	80	10	40k	✓	-	✓
Reiss et al. [116]	-	-	✓	-	9	-	18	3.6M	✓	-	-
HPTE [56]	✓	✓	-	-	5	240	8	100k	✓	-	✓
EmoPain [57]	✓	-	-	-	50	-	11	33k	✓	-	✓
AHA-3D [58]	✓	-	-	-	21	79	4	170k	✓	-	✓
UI-PRMD [60]	✓	-	-	-	10	100	10	60k	✓	-	✓
mRI (Chapter 5)	✓	✓	✓	✓	20	300	12	160k	✓	✓	✓

HPE promises to capture complex body movement naturally occurring in daily activities or prescribed by clinicians, and thus offers a promising vehicle to inform treatment and to quantify the progress of treatment. In particular, human pose estimation plays an increasingly important role in healthcare applications, such as remote rehabilitation training [129, 130]. The current mainstream rehabilitation treatment involves a physical therapist supervising the patients in person. In contrast, HPE-based health monitoring systems can help clinicians correct patients' movements or instruct them remotely. Individual sensing modality has been previously considered, including RGB camera [56, 57], depth camera [58, 59, 60], IMUs [116], and MoCap [57, 60]. Reiss et al. [116] presents a dataset monitoring

physical activities with three IMUs and a heart rate monitor. The home-based physical therapy exercises (HPTE) dataset [56] uses Kinect to record video and depth streams while users perform eight therapy actions. The EmoPain dataset [57] captures both joint information and face videos to classify the pain level based on the emotion in the rehabilitation movements. The AHA-3D [58] dataset contains 79 skeleton videos recorded by Kinect for four healthcare activities. Similarly, the UI-PRMD [60] dataset captures common physical rehabilitation exercises using the Kinect and Vicon MoCap. Similar to these works, mRI focuses on rehabilitation exercises, and provides the most comprehensive set of sensing modalities while remaining competitive in its scale. To the best of our knowledge, our proposed MARS (Chapter 4) is the first work performs home-based rehabilitation through mmWave-based human pose estimation.

3D Human Generation

All of the aforementioned applications primarily rely on discriminative models, which focus on learning the relationship between sensor measurements and corresponding outputs such as activity labels or pose estimates. However, the exploration of generative models offers an equally valuable perspective for enriching our understanding of human representation, complementing the insights derived from discriminative methods. Generative models play a crucial role in enhancing our comprehension of human representation by capturing the underlying distribution of the data and generating new samples that are virtually indistinguishable from real data. This approach provides valuable capabilities for data augmentation,

completion, and the creation of novel human representations. By filling in missing information and simulating diverse scenarios, generative models can effectively address challenges associated with limited or incomplete data.

3D-aware generative models have recently seen rapid progress, fueled by the integration of implicit neural representation in 3D scene modeling and Generative Adversarial Networks (GANs) for human face synthesis [2, 131, 132, 5, 133, 4, 3]. Given the impressive progress of GANs on 2D image generation [134, 135, 136, 137], many studies have attempted to extend them to 3D-aware generation. These GANs aim to learn a generalizable 3D representation from 2D image a collections. For face synthesis, Szabo et al. [138] first proposed using vertex position maps as the 3D representation to generate textured mesh outputs. Shi et al. [139] proposed a self-supervised framework to convert 2D StyleGANs [135] into 3D generative models, although its generalizability is bounded by its base 2D StyleGAN. GRAF [2] and pi-GAN [131] are the first to integrate NeRF into 3D GANs. However, their performance is limited by the intense computation cost of forwarding and backwarding a complete NeRF. Many recent studies [42, 132, 3, 140, 4, 141, 5, 133, 142, 143, 144] have attempted to improve the efficiency and quality of such NeRF-based GANs. Specifically, EG3D [5] introduces tri-plane representation that can leverage a 2D GAN backbone for generating efficient 3D representation and is shown outperforming other 3D representations [144]. Parallel to these works, another thread of studies [145, 146, 147, 143] have been working on controllable 3D GANs that can manipulate the generated 3D faces or bodies. In strong contrast to previous studies, our proposed PanoHead (Chapter 8) is the first 3D GAN

framework that enables view-consistent and high-fidelity full-head image synthesis with detailed geometry, renderable in 360°. It opens up new possibilities for human representation and downstream tasks such as style mixing and inferencing the full-head from a single view RGB image.

3 MGAIT: MODEL-BASED GAIT ANALYSIS USING WEARABLE BEND AND INERTIAL SENSORS

3.1 Background, Motivation and Contributions

Movement disorders are one of the leading causes of functional disability in the elderly population. More than 10 million people worldwide suffer from movement disorders, such as Parkinson's disease (PD); more than 1,200,000 patients are expected to be diagnosed with PD in the United States by the year 2030 [148, 26]. Gait impairment and instability are among the most common symptoms in movement disorder patients and stroke survivors [149]. Therefore, gait function analysis plays an important role in treatment and rehabilitation.

Gait function analysis provides valuable insight into a patient's symptoms and rehabilitation, by evaluating metrics, such as step length, stride length, and gait velocity [52, 99]. In particular, step length is used to analyze the symmetry of gait. While step length remains nearly constant for a healthy person, it varies as the left and right feet alternate for a patient with gait asymmetry. This difference is used as an important feature to evaluate the symmetry of gait and monitor the progress of rehabilitation [52].

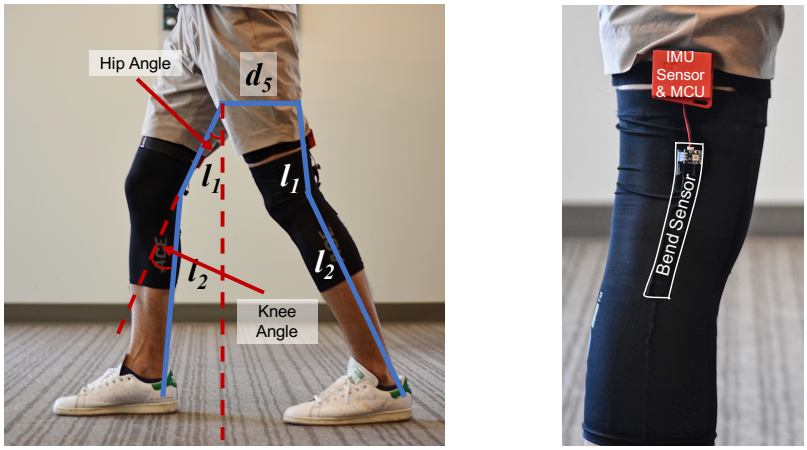
Several clinical studies use the GAITRite system [104], a pressure-sensitive walking mat, to analyze the gait parameters [150, 151, 152]. While GAITRite can provide a 98%-99% accuracy, it cannot be used to continuously monitor the patient's gait after they leave the clinic. To address this limitation, recent work employs wearable

sensors for gait analysis [63, 99, 100]. Most of these studies mount a number of inertial motion units (IMU), which typically incorporate a 3-axis accelerometer and a 3-axis gyroscope, on the leg to collect acceleration and rotation data when a person is walking. However, these approaches need to employ either a large number of sensors [99, 103] or are evaluated using a high-cost IMU (over a few thousand dollars) [101], which is not practical for daily use. Therefore, there is a critical need for simple and intuitive models to estimate gait parameters using relatively few sensors. This chapter presents a wearable cyber-physical system (CPS), called *MGait*, that combines physiological sensors, energy-efficient local processing, and real-time user feedback. We implement *MGait* on a knee sleeve, as shown in Figure 3.1 and demonstrate that it achieves the following goals:

Functionality: Enable continuous daily monitoring and *real-time feedback* with 95% average accuracy without relying on any clinical environment and experimental infrastructure.

Power-performance: Achieve real-time operation and mW-range operation using only one pair of IMUs and low-power bend sensors *for the first time in literature*.

We meet these goals with two sensors per leg and commercial-off-the-shelf components that can be integrated with a total cost of less than $160, in contrast to techniques that use sensors and equipment with over one order of magnitude higher cost. A stretchable bend sensor mounted on the back of the knee measures the knee angle, while an IMU sensor above the knee measures the swing of the hip on each leg, as shown in Figure 3.1. Data from the bend and IMU sensors are processed *locally in real-time* to obtain these angles. We propose a novel biomechanical model



(a) **(b)**

Figure 3.1: (a) Wearable setup used for gait analysis (b) Magnified view of the sensors

and derive a *closed-form expression* that computes the step length using the angle data. Using only our closed-form expression and sensor data leads to over 10% estimation error due to sensor offset and measurement noise. We reduce this error significantly using a novel two-step nonlinear regression technique. Thorough experimental evaluations with seven subjects show that the proposed approach achieves a 5% mean absolute percentage error (MAPE) with this optimization. In addition, we also present a recursive least square estimation technique that can tune the proposed model in the field. Finally, we employ the proposed model to derive other commonly used clinical metrics, including stance time, swing time, gait velocity, and stride length.

In summary, the major contributions of this chapter include:

- An energy-efficient wearable cyber-physical system for real-time gait analysis and step length estimation.

- A novel analytical modeling approach and a closed-form expression for estimating step length, stride length, stance time, swing time, and gait velocity.
- A thorough experimental evaluation, including offline and online estimation, with seven subjects and a new dataset.

3.2 Overview of Gait Cycle and MGAIT Approach

3.2.1 Gait Cycle Definition and Segmentation

Accurate step length modeling requires a clear understanding of the periodic gait cycle and two key angles (*All the angles and the length of the limbs in the following text are considered on the sagittal plane unless otherwise specified.*):

Knee Angle (β) is the angle formed at the joint of the thighs and legs, as shown in Figure 3.1(a). It ranges between 0 (when the leg is straight) and $\approx$50 (during the swing).

Hip Angle (α) is the angle formed by the inner thigh and the vertical axis, as shown in Figure 3.1(a). It is positive ($\approx$40) when the thigh is in front of the torso and negative when the thigh is behind the torso, as shown in Figure 3.2(a).

Knee and hip angles change periodically during the gait cycle. For the front limb, the hip and knee angles are denoted by α_f and β_f, respectively. Similarly, the hip and knee angles of the back limb are denoted using α_b and β_b, respectively. There are four key events and two stances [153, 154], as marked in Figure 3.2(a):

1. *Initial Contact*: At the beginning of the gait cycle, the foot just touches the

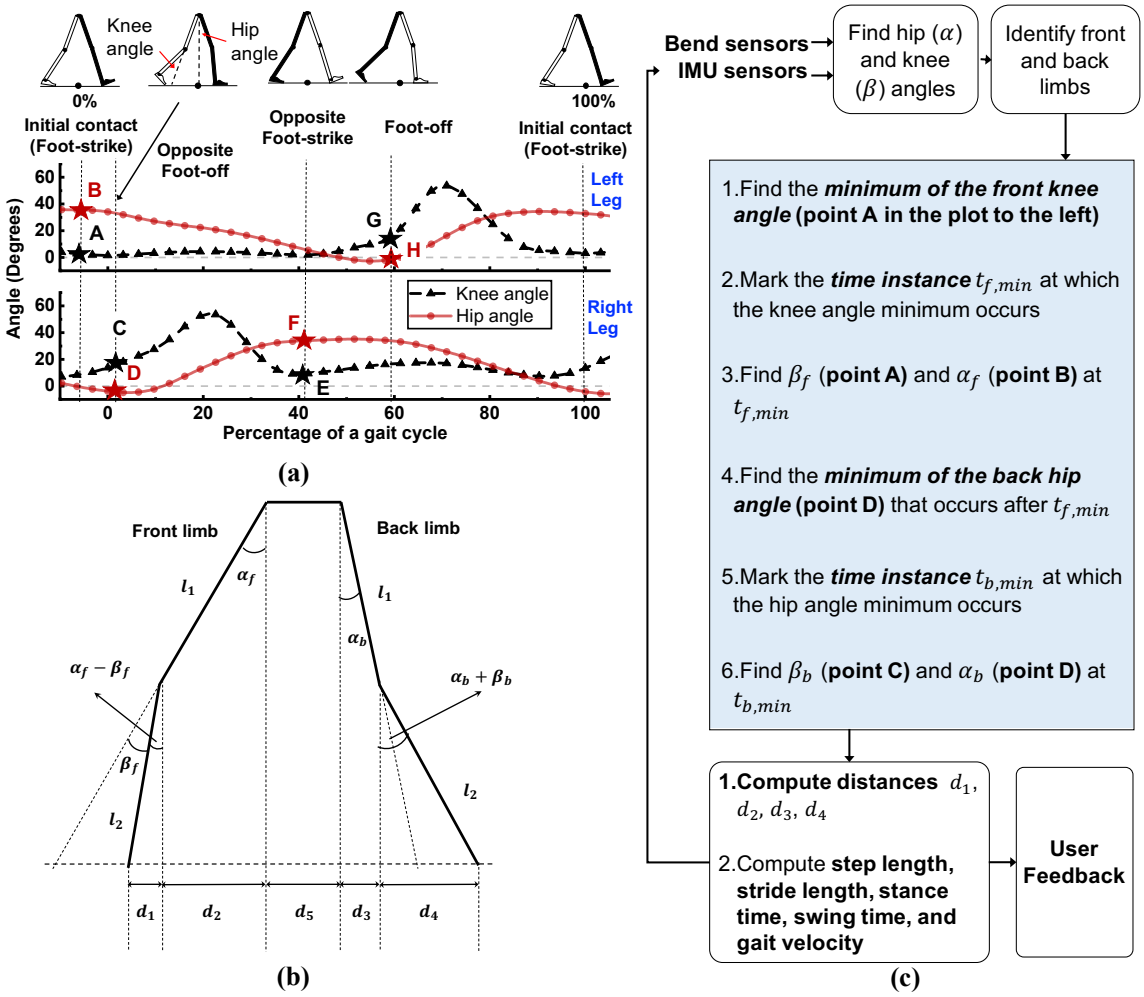


Figure 3.2: Overview of a gait cycle and MGait framework

ground. The knee angle of the front leg (the left leg shaded in Figure 3.2(a)) reaches its first minimum and the hip angle is close to its maximum, as shown with markers A and B, respectively.

2. *Opposite foot-off*: The opposite leg (the right leg in Figure 3.2(a)) starts lifting from the ground. After this point, the right leg is in a forward swing, until it

strikes the ground. The angles in this stance are denoted by markers C and D.

3. *Opposite foot-strike*: At this point, the right leg completes the swing (i.e., one complete step) and touches the ground. The right leg is in front of the torso while the left leg is behind the torso. Thus, the knee angle of the right leg is at a minimum, as shown with the marker E in Figure 3.2(a).

4. *Foot-off*: The hip angle of the left leg reaches its minimum value. The left also starts its swing forward for the next step. This stance is shown using markers G and H.

These four events of one leg belong to its stance phase. The period from foot-off to the next initial contact is the swing phase.

3.2.2 Flow of the MGait Approach

The goal of the *MGait* approach is to model and estimate the step length and relevant parameters using the setup shown in Figure 3.1(a). We place a bend sensor vertically aligned to the backside of the knee when the subject is standing. To measure the knee angle with the best accuracy, the midpoint of the bend sensor is adjusted at the knee level. The IMU is placed on the hip, perpendicular to the ground when the subject is standing. Accelerometer and gyroscope data from the IMU are used to calculate the hip angle. We process the angle data to find the major events in the gait cycle, as outlined in Figure 3.2(c). Specifically, we find the points {A, B, C, D} for the first step and {E, F, G, H} for the second step plotted in Figure 3.2(a). The angles at these points are used to calculate the step length

components d_1–d_4 shown in Figure 3.2(b). These components are used by the *MGait* approach to compute the step length, stride length, and gait velocity. The details of the analytical model to compute the step length, stride length, and gait velocity are presented in Section 3.4.

3.3 Hip and Knee Angle estimation

3.3.1 Sensor Calibration and Data Pre-processing

Due to changing environment, sensor data is noisy and drift prone. For instance, angle measured by the bend sensor for the resting state can vary with time and IMU sensors may have offsets. To calibrate the sensors, we instruct the users to stand still with a straight leg for a few seconds before the experiment. During this period, we record the sensor readings and find their median values as the offsets. We then subtract the offsets from the sensor data during actual use. For example, the self-calibration ensures that the IMU measures only the gravity when the user stands in a straight position.

After calibrating the sensors, we apply a downsampling and smoothing filter to reduce noise in the data. For each sensor, we obtain the downsampled and smoothed samples as:

$$s_s[Mk] = \frac{1}{2M} \sum_{i=M(k-1)+1}^{M(k+1)} s[i] \tag{3.1}$$

where s_s is the smoothened data stream, M is the downsampling factor, k is the sample index, and s is the data stream before filtering. The IMU and bend sensor

sampling rates are 250 Hz and 100 Hz, respectively. Since the fastest gait is less than 10 Hz, we downsample the data to 25 Hz, which is sufficient, considering the Nyquist rate. The specifications of sensors before and after filtering are shown in Table 3.1.

Table 3.1: Sensors specifications

	Sensor range	Sampling rate (Hz)	Sampling rate after median filter (Hz)
Accelerometer	$\pm$8g	250	25
Gyrometer	$\pm$250dps	250	25
Bending sensor	$\pm$180°	100	25

3.3.2 Knee Angle Computation and Visualization

The output of the bend sensor (located at the back of the knee) is the angle displacement experienced by the sensor, as illustrated in Figure 3.1(a). Sample knee angle data from our experiments is visualized in Figure 3.3(a). A comparison between the raw and filtered data shows that the averaging filter smooths out the variations in the data. The knee angles reach a maximum of about 50–55° as the leg swings. Another key observation is that the angles of the two legs exhibit approximately 50% phase shift when walking. This is expected for healthy subjects, since their steps are of similar length.

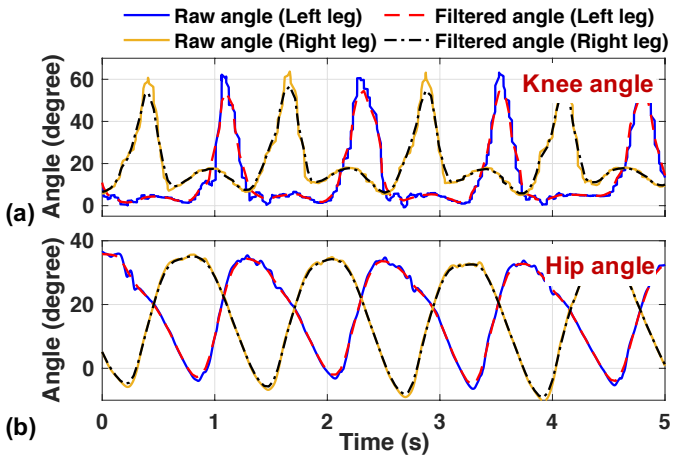


Figure 3.3: Visualization of raw and median filtered (a) knee angle and (b) hip angle of both legs during walking

3.3.3 Hip Angle Computation and Visualization

We find the hip angle using the IMU sensor located right above the knee, as shown in Figure 3.1(a). There are several approaches to filter the accelerometer and gyroscope samples from the IMU and obtain the hip angle. Among these, we compared the performance of the complementary filter [155] and the Madgwick filter [156], which are the most commonly used techniques. Since the Madgwick filter outperforms the complementary filter in terms of the ability to converge and the smoothness, we used it in the final implementation. The Madgwick filter uses a quaternion representation of the accelerometer samples and applies a gradient descent algorithm to calculate the error in the direction of the gyroscope samples. By compensating for the error, it accurately estimates the orientation of the IMU even during motion, without being affected by gyroscopic drift [156].

Figure 3.3(b) plots left and right hip angles during walking. The hip angles

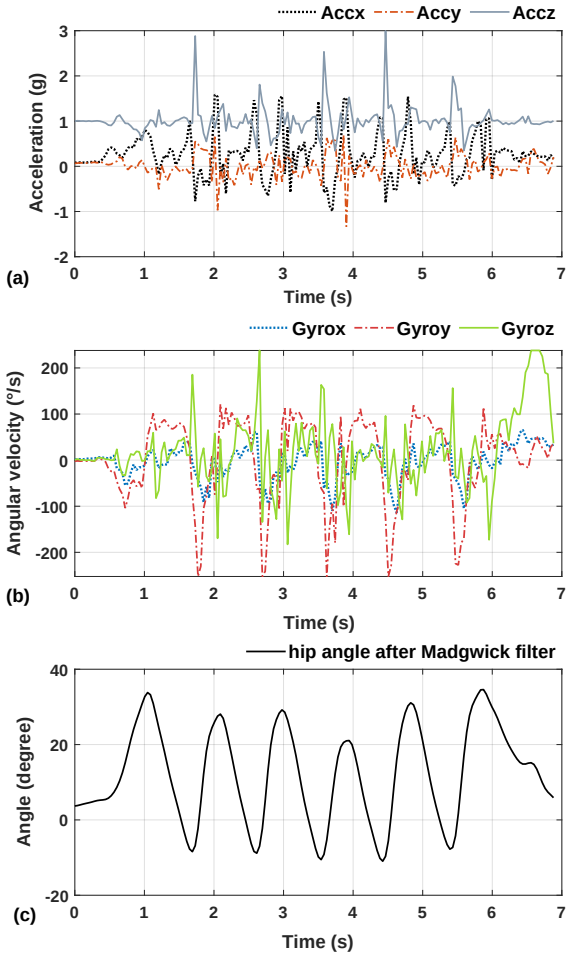


Figure 3.4: Visualization of acceleration values (a) and angular velocity values (b) forming hip angle (c) after the Madgwick filter

vary between a minimum of about -10° to a maximum of 35° and exhibit a 50% phase difference, as expected. We also show inputs and outputs of the Madgwick filter in Figure 3.4. Specifically, the Madgwick filter takes the accelerometer and gyroscope data shown in Figures 3.4(a) and 3.4(b) as inputs. Using these inputs, it generates the smoothed hip angles, as shown in 3.4(c).

3.3.4 Finding the Key Events during the Gait Cycle

We need to identify the key gait cycle events, such as initial contact and foot-off, to convert the knee and hip angles into step length. From our gait cycle analysis, we know that these events occur when the knee angle is at a minimum for one leg (point **A** in Figure 3.2(a)), while the hip angle is at a minimum for the other (point **D** in Figure 3.2(a)). To obtain the minimum values, we continuously monitor the five-point derivative of the two angles. A minimum is marked whenever the derivative changes from negative to positive or zero to positive. After detecting the minimum of the knee angle, we designate the corresponding leg as the front limb for the current step. We then call the knee angle β_f and hip angle α_f for the front limb. For example, the knee angle β_f and hip angle α_f in Figure 3.2(a) correspond to points **A** and **B**. We then wait for the minumum of the hip angle to occur in the other leg, which is referred to as the "back limb". At this point, the knee angle β_b and the hip angle α_b for the back limb corresponds to points **C** and **D** in Figure 3.2(a). The angle values are then plugged into our analytical model to estimate the step length, as detailed in Section 3.4. This process is repeated continuously, where front and back limbs alternate.

3.4 Proposes Gait Analysis Model

3.4.1 Parameter Definitions

MGait estimates the step length, stride length, and gait velocity defined below in real-time.

Step Length is the distance between the front and back feet, when the front limb is in the initial contact stance and the back limb is in the foot-off stance.

Stride Length is the distance between two foot strike stances of the same leg.

Stance Time stands for the period that from the foot touches the ground to the same foot leaves the ground.

Swing Time stands for the period that from the foot leaves the ground to the same foot touches the ground.

Gait Velocity is the ratio of the stride length and the time taken to complete one stride.

3.4.2 Gait Parameter Estimation

We represent the stances during a step by the geometric stick diagram shown in Figure 3.5. The left side of each drawing shows the front limb in the initial contact stance, while the right side shows the back limb in the foot off position. There are two cases of the stick diagram, depending on the position of the back limb. In the first case shown in Figure 3.5(a), the back limb is extended behind the torso of the subject. In contrast, the upper part of the back limb aligns with the torso of the subject in the second case shown in Figure 3.5(b). We consider these two cases separately, since they change how the various components of the step length contribute to the model.

The step length is a function of the length of the leg of the subject. The length of the leg between the gluteus (hip) and popliteus point (knee joint) is denoted by l_1, as shown in Figure 3.5(a). The length of the leg from the popliteus to the

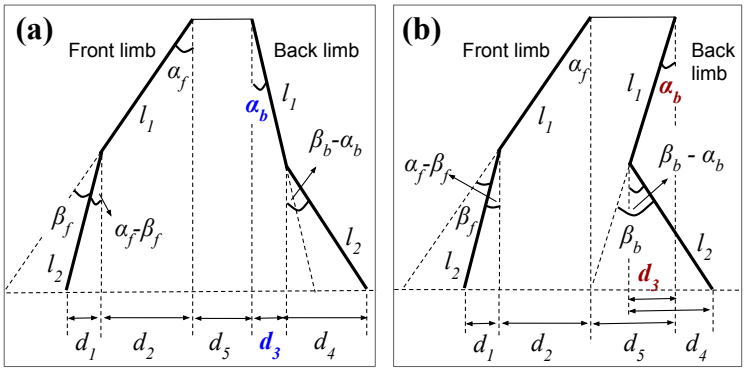


Figure 3.5: Stick diagram for step length calculation. The stick model has two cases, depending on the position of the back limb. (a) upper part of the back limb is behind the torso and (b) upper part of the back limb is parallel or slightly in front of the torso. The figures show the magnitude of all the angles in the model.

calcaneus point (ankle) is given by l_2. We then consider the knee and hip angles formed by both legs in the stick model. Using these definitions, we can express the step length as a sum of five components shown in Figure 3.5. Next, we describe the definitions of each of these components.

d_1: The d_1 component is the projection of the front leg on the ground. Figure 3.5(a) shows that the angle formed by the vertical line and the leg is $\alpha_f - \beta_f$. Using this angle, the value of d_1 is written as:

$$d_1 = l_2 \sin(\alpha_f - \beta_f) \tag{3.2}$$

d_2: d_2 is the projection of the front thigh to the ground. Since the angle formed by the thigh and the vertical axis is given by α_f, the projection is:

$$d_2 = l_1 \sin(\alpha_f) \tag{3.3}$$

d_3**:** d_3 is the projection of the back thigh to the ground. Since the angle formed by the thigh and the vertical axis is given by α_b, the projection is:

$$d_3 = l_1 \sin(-\alpha_b) \tag{3.4}$$

There is a negative sign in Equation 3.4 to account for the two cases shown in Figure 3.5(a) and Figure 3.5(b). In Figure 3.5(a), the d_3 has to be added to obtain the total step length. Therefore, we must ensure that the sign of d_3 is positive. By our definition of the hip angle, α_b is negative in Figure 3.5(a) because it is behind the torso. The inclusion of a negative sign in Equation 3.4 ensures that d_3 is positive. Conversely, in Figure 3.5(b) d_3 overlaps with d_4 and d_5. Consequently, it has to be subtracted from the step length. In this case, α_b is positive because it is in front of the torso. Hence, we obtain a negative sign for d_3, ensuring its subtraction from the total step length.

d_4**:** d_4 is the projection of the back leg to the ground. Similar to d_1, we first need to calculate the angle between the back leg and the vertical line from the knee to the ground. In case of Figure 3.5(a), the magnitude of this angle is given by the sum of angles α_b and β_b, whereas for Figure 3.5(b) the magnitude is given by the difference of β_b and α_b. The sign of β_b is always positive, while the sign of α_b is negative in Figure 3.5(a) and positive in Figure 3.5(b). Therefore, when we consider the signs of the respective angles, the resulting angle between the back leg and the vertical line becomes $(\beta_b - \alpha_b)$. Hence, the projection is expressed as:

$$d_4 = l_2 \sin(\beta_b - \alpha_b) \tag{3.5}$$

d_5**:** d_5 is given by the diameter of the subject's thigh below the gluteus. It is included, since the stick diagram does not cover the width of the user's leg. It is measured at the beginning to personalize the model to each user.
Adding d_1–d_5 the total step length is obtained as:

$$D = l_2 \sin(\alpha_f - \beta_f) + l_1 \sin(\alpha_f) + l_1 \sin(-\alpha_b) + l_2 \sin(\beta_b - \alpha_b) + d_5 \qquad (3.6)$$

After obtaining the step length, we calculate the stride length by adding the lengths of two consecutive steps. We compute the gait velocity by dividing the length of a sequence of five strides by the time. The stride time is obtained by subtracting two consecutive timestamps obtained when finding the minimums, as described in Figure 3.2(b).

3.4.3 Real-time User Feedback

Gait analysis is commonly used in patient rehabilitation and health monitoring. Therefore, providing feedback to the users about any abnormalities in gait is a crucial aspect of gait analysis. To this end, we provide the following feedback to the user in real-time. Note that real-time feedback is optional, and users can always choose offline analysis of the gait after their trial. Some users with functional disabilities might find it uncomfortable if the buzzer or the LED is always on. Therefore, we enable logging the gait data during the whole trial and provide the batch least square algorithm users can use to analyze their gait patterns offline. We

emphasize that the proposed feedback mechanism is tested only with healthy subjects mimicking the limping behavior due to difficulty in accessing actual patients. Nevertheless, these studies can pave the way for applying the proposed algorithms to real patients. As our future work, we plan to collect the users' feedback, improve the framework's robustness, and test it with actual patients, e.g., those suffering from the freezing of gait (FoG).

Gait Asymmetry Detection: Patients with movement disorders often have different left and right step lengths [157]. Therefore, we include gait asymmetry as one of the feedbacks that we provide to the user. We notify the user of gait asymmetry using a buzzer on our device. The buzzer sounds an audible alarm to the user such that they can take appropriate action to correct the gait asymmetry.

Gait Velocity Reduction Detection: Falling is among the primary causes of death in the elderly population [98].

Change in the gait velocity over a long period is considered as a significant cue to predict future falls [98, 96]. *MGait* keeps track of the gait velocity of each patient and notifies the patient or the healthcare provider, if there is a specific trend in the decrease of gait velocity that can be identified as a potential risk of fall.

Step/Stride Length Reduction Detection: Reduced stride/step length is one of the most prominent features of various movement/motor disorders in the elderly population, especially in Parkinson's Disease (PD) patients [158].

Therefore, we keep track of the user's moving average step/stride length and notify the user if there is a significant reduction in step length using an LED on our device.

Feedback Algorithm: Gait asymmetry and step/stride length reduction can be detected by observing the trend of step length over time. In a healthy walking pattern, left-to-right step length and right-to-left step length are similar in length. In contrast, when gait asymmetry and step/stride length reduction are present, one step is shorter than the other. That is, the difference between left-to-right and right-to-left step length is larger than healthy walking. Using the insights, we design the feedback algorithm using the percentage gait asymmetry in strides.

The percentage gait asymmetry is employed as a metric to determine if a stride is asymmetric. The first step of our algorithm is to calculate the percentage gait asymmetry in a stride. The percentage gait asymmetry is defined in Equation 3.7:

$$Gait^{i}_{asym} = \frac{\left|L^{i}_{left} - L^{i}_{right}\right|}{0.5 \times (L^{i}_{left} + L^{i}_{right})} \times 100\% \tag{3.7}$$

, where i is the stride count and L is the step length of a stride. L^{i}_{right} represents the i_{th} right step length, and the similar explanation applies for the left situation. The percentage gait asymmetry quantifies the difference between the left and right steps length in the same stride. If the difference exceeds a threshold, we then identify that this is an asymmetric stride. The value of the difference threshold has to be chosen carefully to avoid false positives. Since percentage gait asymmetry involves the normalization operation itself, it accurately reveals the percentage left-to-right step length difference even for subjects with different height and static parameters (l1, l2, and d5). Our dataset shows that the percentage gait asymmetry in normal step length when walking is less than 20%. To avoid extra false alarms, in our implementation, we choose the threshold as 25%. At the same time, all steps length

and step speed data are logged. By using the proposed algorithm, the users can analyze their gait patterns offline.

3.5 Experimental Evaluation

3.5.1 Experimental Setup

Wearable Device: *MGait* framework uses the wearable setup shown in Figure 3.1(a). The magnified view of the wearable setup is shown in Figure 3.1(b). It consists of a wearable bend sensor [159] and a Texas Instruments CC2650 Sensortag [160]. We sample the bend sensor and sensortag at 100 Hz and 250 Hz, respectively. The proposed gait analysis technique is implemented on the TI Sensortag to enable runtime analysis and user feedback.

User Studies: We collected data from a total of seven subjects (S1–S7), following an official protocol approved by the IRB board of our institution. The information, including the static parameters (l_1, l_2, and d_5), height, age, and gender of 7 subjects, are shown in Table 3.2. Each subject participated in six trials. Four of the six trials were regular free walking with normal pace, whereas in the remaining two the subjects were asked to imitate limping, resulting in a total of 806 steps. In addition to the empirical data, we also proposed a method to generate additional data. The synthetic data generation augments the data collected from the 7 subjects to create a richer dataset for further research. Both empirical and synthetic data will be released to the public, along with this paper.

Accurate step length reference is critical to evaluating *MGait*. For most of the

experiment, subjects walked on a 7-meter long white paper roll. We rubbed the bottom of the subject's shoes with washable ink. The marks left on the paper are then used to capture the user steps. After each trial, the distances between the marks on the paper are recorded as step lengths. For the last two subjects, we employed the GAITRite [104] system to obtain the step length reference.

Across all the experiments, the recorded ranges for step length, stride length, and gait velocity are 25–78 cm, 69–156 cm, and 0.60–1.27 m/s respectively. We also measure the stance time and the swing time of our dataset. Stance time of gait stands for the period that from the foot touches the ground to the same foot leaves the ground. Swing time of gait stands for the period that from the foot leaves the ground to the same foot touches the ground. The dataset's stance time and swing time ranges are 0.38—0.75 s and 0.23—0.54 s, respectively. Generally, 60% of one gait period is stance time, while 40% is swinging time. The step length distribution in our experimental data is as follows. 80% of the step lengths are between 60 cm and 70 cm, 9% of them fall into 50 cm to 60 cm range, 6% are shorter than 50 cm, finally and 5% of them are longer than 70 cm. Similarly, 76% of the stride lengths are from between 120 cm and 140 cm, 11% of them vary from 100 cm to 120 cm, 7 of them are shorter than 100 cm, and 6% of them are longer than 140 cm. We also perform a statistical analysis on our dataset. For each subject's step length and stride length, we calculate the mean and standard deviation. Moreover, a 95% confidence interval is computed. The detailed results are shown in Table 3.2.

Table 3.2: Overview of the dataset for each subject. std represents standard deviation and CI represents confidence interval.

	Static param. l1/l2/d5 (cm)	Height (cm)	Age	Gender	Step length (cm)		Stride length (cm)	
					Mean $\pm$ std	CI	Mean $\pm$ std	CI
S1	30/45/14	193	23	M	61.33 $\pm$ 6.65	60.31 - 62.35	122.87 $\pm$ 11.07	120.46 - 125.29
S2	25/40/15	175	26	M	61.75 $\pm$ 13.39	57.67 - 65.82	123.50 $\pm$ 25.41	112.23 - 134.76
S3	29/40/16	188	26	M	73.37 $\pm$ 3.83	72.10 - 74.65	146.66 $\pm$ 6.20	143.58 - 149.74
S4	27/33/12	170	29	M	58.21 $\pm$ 13.56	53.98 - 62.44	116.42 $\pm$ 25.33	104.89 - 127.96
S5	20/25/15	163	26	F	48.05 $\pm$ 6.58	46.42 - 49.68	96.65 $\pm$ 11.57	92.48 - 100.82
S6	28/40/16	181	25	M	60.47 $\pm$ 2.89	60.19 - 60.75	121.01 $\pm$ 5.29	120.28 - 121.74
S7	22/28/22	158	25	F	57.76 $\pm$ 5.35	56.14 - 59.39	115.53 $\pm$ 10.36	110.4 - 120.13

3.5.2 Offline and Online Estimation of Static Parameters

The proposed model has three user-specific static parameters: length of the thigh (l_1), length of the leg (l_2), and hip diameter (d_5). Since measuring them is subject to human error, we employ both offline batch processing and online least squares regression models to determine these parameters.

Batch Least Square Estimation

Batch processing is a common method used in the least square estimation. For our step length estimation problem, batch processing is an effective solution when the gait analysis is performed offline. For instance, a patient can walk for a few minutes, while measuring the golden step length reference. The collected data then can be used to reduce the measurement error by tuning the model.

We employ a nonlinear least square estimation to solve the problem. First, we measure their nominal values by hand and record them as l_1^{nom}, l_2^{nom}, d_5^{nom}. We then use our test data and these nominal values to formulate the following regression problem:

$$
\begin{aligned}
\text{minimize} \quad & \sum_{i=1}^{N} \|D_{ref,i} - D_i(l_1, l_2, d_5)\|^2 \\
\text{subject to} \quad & 0.9 \times l_1^{nom} \leqslant l_1 \leqslant 1.1 \times l_1^{nom} \\
& 0.9 \times l_2^{nom} \leqslant l_2 \leqslant 1.1 \times l_2^{nom} \\
& 0.9 \times d_5^{nom} \leqslant d_5 \leqslant 1.1 \times d_5^{nom}
\end{aligned} \tag{3.8}
$$

where N is the number of steps used for regression and $D_{ref,i}$ is the step length reference for i^{th} step. The objective in Equation 3.8 minimizes the sum of squared error between $D_{ref,i}$ and our estimate D_i obtained with Equation 3.6 using the measured angles $(\alpha_f, \alpha_b, \beta_f, \beta_b)$. We constrain the optimization variables l_1, l_2, and d_5 within 10% of their nominal values to ensure that they do not overfit to unrealistic values. As an example, the parameters of S1 are 30.0, 45.0, and 12.0 cm initially. After the regression, they are corrected as 33.0, 42.1, and 13.2 cm, respectively. *This method is used once for each subject to find their static parameters.*

Recursive Least Square Estimation

It is also useful to fine-tune the model in real-time after it is deployed, since this can help the proposed system to adapt to a particular person. Batch processing is not appropriate for this purpose as it requires offline processing. Therefore, we also introduce a Recursive Least Square (RLS) estimation framework that can calibrate the static parameters for a given user.

We can represent the step length in Equation 3.6 as a linear equation with the

input features as:

$$\mathbf{h}[n] = \begin{bmatrix} \sin(\alpha_f[n]) - \sin(\alpha_b[n]) \\ \sin(\alpha_f[n] - \beta_f[n]) + \sin(\beta_b[n] - \alpha_b[n]) \\ 1 \end{bmatrix}, \tag{3.9}$$

where n is the time index. Similarly, the model coefficients become the static parameters we aim to estimate:

$$\mathbf{w} = \begin{bmatrix} l_1 & l_2 & d_5 \end{bmatrix}^T \tag{3.10}$$

When the step length model needs to be updated, e.g., for a new user, the system operates in a calibration mode. In this mode, the user is asked to walk on a line with a constant step length. During this time, the RLS filter estimates the step length as a product of $\mathbf{h}[n]$ and $\mathbf{w}$, which are given Equation 3.10 and Equation 3.9, respectively. This estimate is subtracted from the reference step length to find the modeling error. Then, this error is then used to update the parameter estimates $\mathbf{w}$ using an RLS technique with stability guarantees [161].

Experimental results show that the RLS method improves the step length estimation accuracy by about 3% on average. It is effective in reducing the measurement error online, especially for subjects with a high initial error rate. Table 3.3 shows the error in step length estimation using offline and online method. For instance, S2 and S5 have 16.44% and 16.01% initial MAPE, respectively. Applying RLS estimation, the MAPE are reduced by 7.54% and 8.05%, respectively. Figure 3.6 shows how

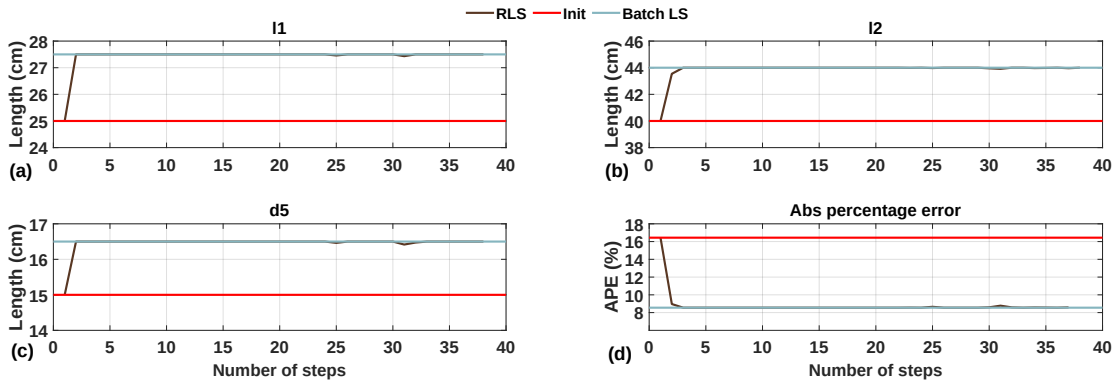


Figure 3.6: Visualization of how (a) l1, (b) l2, and (c) d5 online updating and converging to the batch LS result and the (d) absolute percentage error of step length estimation using RLS for one subject.

static parameters and the error converge online, as steps increases. The RLS model converges to its final values within five steps. The accuracy of RLS converges also to the batch LS when it is applied to all subjects. This shows that the RLS approach is effective for estimating the static parameters at runtime.

Offline vs. Online Calibration: The online calibration (Section 3.5.2) can expedite the MGait framework to adjust to new users, but it requires the user to walk in a straight line with a constant step length. However, patients with movement disabilities may be unable to walk in a straight line. In this case, the offline calibration (Section 3.5.2) may be preferred, as it does not require the user to walk in a straight line with a constant step length. The user will only walk a few trials without any restriction to do the offline calibration.

Table 3.3: Error in step length estimation using offline and online method (without angle correction). RLS represents recursive least square, MAPE represents mean absolute percentage error, and RMSE means root mean square error.

	Initial		After Batch LS		After RLS	
	MAPE (%)	RMSE (cm)	MAPE (%)	RMSE (cm)	MAPE (%)	RMSE (cm)
S1	6.38	4.03	5.80	3.65	6.28	3.99
S2	16.44	12.15	8.55	6.34	8.90	6.60
S3	7.17	4.53	6.88	4.31	6.93	4.38
S4	7.65	4.99	5.89	3.75	5.95	3.79
S5	16.01	8.03	7.94	3.99	7.96	3.99
S6	8.70	5.25	8.07	4.86	8.66	5.22
S7	7.88	4.47	7.69	4.35	7.66	4.35
Avg.	10.03	6.21	7.26	4.46	7.47	4.62

3.5.3 Analysis of Error Distribution of Angle Measurements

The angle measurements are also subject to error, due to the noisy nature of the sensors. Furthermore, sensors may experience a systematic bias, due to their positioning. This difference should be accounted for in the models to ensure an accurate estimation of the step length. To compensate for these errors, we solve the nonlinear regression problem given in Equation 3.8 again, by using the static parameters found in Section 3.5.2. This time, we let the knee and hip angles in Equation 3.6 become free variables. That is, the estimation for the i^{th} step is changed as $D_i(\alpha_f, \alpha_b, \beta_f, \beta_b)$ in the problem formulation in Equation 3.8. Similarly, these parameters are constrained within 10% of their nominal values given by the average of our observations, following the same methodology as in Section 3.5.2.

The output of the nonlinear regression gives the average knee and hip angles

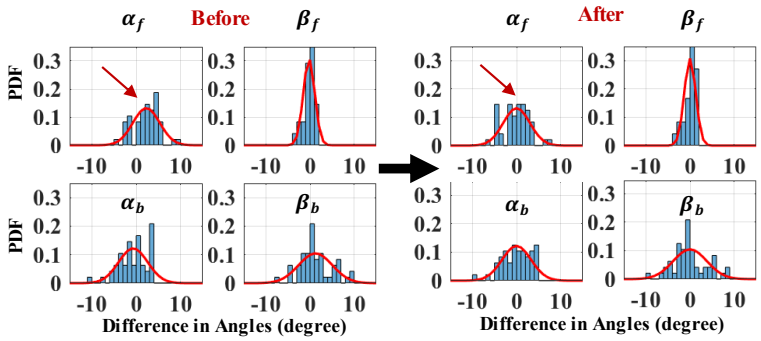


Figure 3.7: Illustration of the error distribution after removing sensor bias.

that provide the minimum estimation error. Hence, we use the difference between these values and our observations as the measurement bias. For instance, the nonlinear regression finds the hip angle of the forward leg at the initial contact point as $\alpha_f = 24.8°$ for Subject 5. The same value is 22.5° in our observation dataset. Consequently, we compute the bias as -2.3°. We rectify the sensor bias, by subtracting these empirically determined values from the knee and hip angles found in real-time. After correcting the bias, the error between the best fit and our measured values become zero-mean, as demonstrated in Figure 3.7.

3.5.4 Accuracy Evaluation

We use 70% of the steps collected from a subject for regression (Sections 3.5.2 and 3.5.3) and the remaining 30% for evaluating accuracy. The errors reported in this section use the *corrected* static parameters and angle distributions. The average MAPE in step length estimation is about 10.03%, *when we use the user measurements without any error correction*. The error is reduced to about 7.26% with the fitting of user-specific parameters, as shown in Figure 3.8. It is reduced further to about

5.49%, when we remove the offset in angle measurements.

In particular, the MAPE for Subjects 2 and 5 drops by more than 10%, since they had a higher initial error due to inaccurate limb measurements.

We then evaluate the accuracy for step length, stride length and gait velocity for each subject. Each row in Table 3.4 corresponds to a different subject, while the columns show the MAPE and RMSE in step length, stride length, and gait velocity, respectively. The MAPE and RMSE for step length range from 3.63% to 6.91% and 2.83 cm to 5.48 cm, respectively. The error rates are lower for stride length and gait velocity estimations. Specifically, the maximum MAPE in stride length and gait velocity is 6.40% and 3.59%, respectively. We note that the RMSE value for stride length seems larger, since each stride consists of two steps. In summary, the results show that the proposed model is able to accurately estimate gait parameters with lower power consumption overheads.

We also compare the proposed approach to simpler techniques, such as integrating the acceleration twice while the feet are in motion [162]. Even after passing the raw data through a low-pass filter, double integration of acceleration leads to a MAPE of 13.5%, significantly higher than MGait.

3.5.5 Evaluation of Real-Time User Feedback

User feedback on gait quality plays a critical role in patient rehabilitation and health monitoring. For instance, feedback on unequal step lengths is an important part of rehabilitation of patients with a leg injury. Similarly, gait speed is an important indicator in movement disorders.

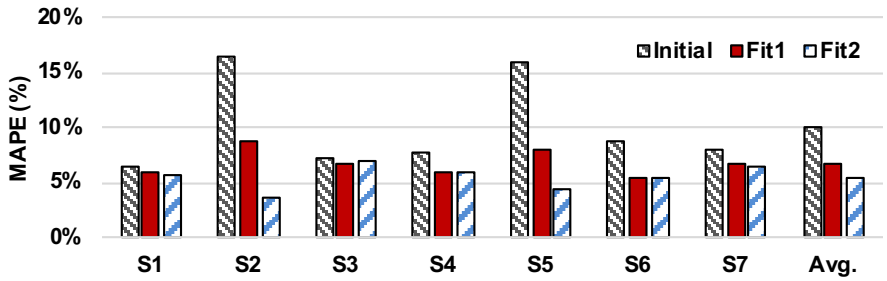


Figure 3.8: Mean absolute percentage error in step length estimates

Table 3.4: Error in step length, stride length, and velocity with angle correction. MAPE represents mean absolute percentage error, and RMSE means root mean square error.

	Step Length		Stride Length		Velocity	
	MAPE (%)	RMSE (cm)	MAPE (%)	RMSE (cm)	MAPE (%)	RMSE (m/s)
S1	5.76	4.53	4.48	6.55	3.33	0.04
S2	3.63	3.35	3.93	7.56	0.77	0.01
S3	6.91	5.48	3.89	6.54	1.64	0.02
S4	5.94	4.51	3.71	6.48	2.06	0.03
S5	4.38	2.83	1.99	2.66	0.94	0.01
S6	5.31	3.61	4.79	6.61	2.73	0.05
S7	6.51	4.24	6.40	7.72	3.59	0.06
Avg.	5.49	4.08	4.17	6.30	2.15	0.03

An example of the user feedback for one subject given by *MGait* is shown in Figure 3.9. Figure 3.9(a) shows the left step length estimation, (b) shows the right step length estimation, and (c) shows the absolute percentage error of the left-to-right difference in strides of continuous walking. *MGait* provides user feedback as soon as the step lengths are asymmetric and the percentage gait asymmetry in a stride exceeds the threshold of 25%, as shown in Figure 3.9. The subject starts

walking, aiming for walking with symmetric steps in strides. After 78 strides, the subject starts making one regular step, followed by a short step (Gait asymmetry). *MGait* keeps track of the percentage gait asymmetry in a stride. Since there are no abnormalities in the first 150 steps, *MGait* does not produce any user feedback. The percentage gait asymmetry in the stride is larger than the threshold of 25% at 78_{th} strides. After this point, *MGait* waits for five steps and compares the average step length before and after the change in variance. Since the percentage gait asymmetry in the stride exceeds 25%, *MGait* raises feedback. Specifically, the user is notified using a buzzer and an LED.

We evaluated all seven subjects using this feedback algorithm. Overall, 25 strides out of 403 strides are obtained while the user was walking with gait asymmetry. The precision, recall, and f1 score of predicting limping are 1, 0.73, and 0.85, respectively.

3.5.6 Sensors Sustainability and Power Analysis

The reliability of the sensors is crucial for *MGait*'s practicality. In this subsection, we evaluate both sensors' sustainability under continuous monitoring.

Sensor drift

This study is the first work introducing wearable bend sensors to the gait monitoring system. The readers might be interested in the sensing drift of the bend sensor since it is a common problem for all sensors. To this end, a systematic evaluation is performed using a robotic motor joint. We employ a robotic joint performing a 60 sinusoidal movement at 1Hz, attach the bending sensor to the joint, and record

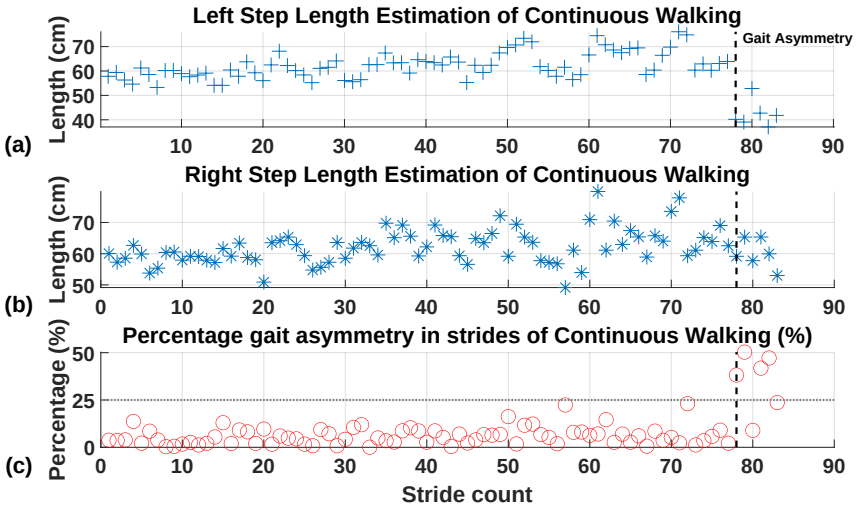


Figure 3.9: Gait asymmetry detection. (a) shows the left step length estimation, (b) shows the right step length estimation, and (c) shows the percentage gait asymmetry in strides. The grey dot line shows the threshold.

the sensing data. The experiment lasts 1.5 hours, a time interval that is enough to cover most of the single clinical therapy session. Figure 3.10 shows that the range of the sensor is initially from 1 to 52. After 45 minutes, the sensor shows a sensing range from -2 to 53. At the end of the experiments, the sensor exhibits a sensing range from -2 to 59. The result indicates that the sensor is reasonably stable in the 1.5 hours of experiments. The sensor drift might be from the friction or the sensors' poor attachment with the joint as time goes on. Also, sports sleeves may slip down after long periods of usage. Therefore, we suggest users align the sports sleeve and calibrate the sensor before each trial to get better performance.

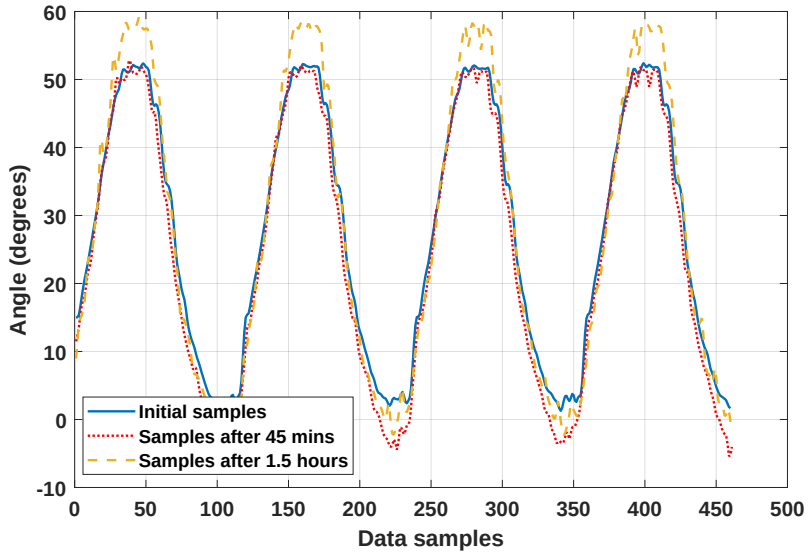


Figure 3.10: Bending sensors reading under continuous running condition

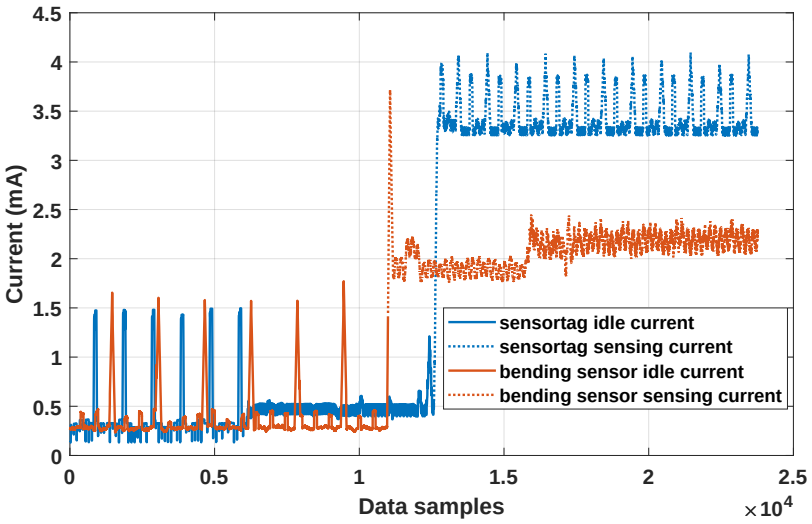


Figure 3.11: Power consumption of bending sensor and sensortag

Power evaluation

Low-power operation is crucial for *MGait* to offer a long-term gait monitoring solution [163]. To explore the power consumption of *MGait*, we measure the actual current consumption of the bending sensors and the IMUs. First, we power the bending sensor with a 150 mAh @ 3.3 V battery and connect a sensing resistor in series to it. Then we turn the sensor on and start sensing. We record the voltage on the resistor and calculate the current drawn by the sensor. Since the resistor's value is low, the drawn current is essentially the sensor's current consumption. We follow the same steps to measure the IMU's current consumption. Figure 3.11 shows the power consumption results. We observe that, for the bending sensor, the average of initial and idle current (the solid line) consumption is 0.5 mA and the sensing current (the dashed line) is about 2.3 mA. The idle current consumption of the IMU is 0.5 mA, while its sensing current is about 3.5 mA. Therefore, when connected to a 150 mAh lithium-ion battery, the bending sensor can run more than 12 days in idle mode, 65 hours in the sensing mode. Similarly, the IMU can run more than 12 days in the idle mode and 43 hours in sensing mode.

4 MARS: MMWAVE-BASED ASSISTIVE REHABILITATION SYSTEM FOR SMART HEALTHCARE

4.1 Background, Motivation and Contributions

Rehabilitation is the process of recovering a patient's health condition to its normal state after a period of illness. The sequelae of central nervous system disorders, such as Parkinson's disease (PD) and cerebrovascular diseases (e.g., stroke), afflict more than 10 million people worldwide. According to recent studies, patients can recover up to 91% functional ability if they start the rehabilitation within three months of the stroke [39]. Similarly, PD patients must follow regular rehabilitation treatments to maximize their functional ability and minimize secondary complications [40]. There is also a strong interplay between mental well-being and maintaining physical activity [164]. These examples demonstrate the importance of rehabilitation treatment to regain patients' quality of life.

The current mainstream rehabilitation treatment involves a physical therapist who supervises the patients in person. The supervision aims to guide the patients to perform specific movement exercises and give feedback to ensure their correctness. Individualized attention from an expert is certainly favorable, but it also incurs a high cost due to critical dependence on experts, dedicated infrastructure, and patients' commute. *Indeed, the Covid-19 pandemic experience has further demonstrated the growing importance of developing alternatives to in-person care*. Home-based rehabilitation systems are needed to address this urgent need. These systems must

allow patients to perform prescribed movement exercises at home while receiving feedback. In this way, they can complement the therapists by enabling daily practices between clinic visits, which can be weeks apart. During these practices, they can monitor whether the patients perform the movements correctly and adjust the intensity. For example, when a patient lifts the arm or the thigh, the system must tell if it is high enough and guide patients to improve the movements.

Home-based patient monitoring systems use predominantly two approaches: wearable sensors- and video-based systems [165, 166, 167, 27]. One of the main advantages of wearable systems is their independence from environmental factors. For instance, it does not matter whether the patients perform the rehabilitation exercise indoors or outdoors as long as they wear the sensors. Also, sensor measurements can be very accurate when appropriately placed. However, recent studies show that frequent charging requirements and discomfort hinder the users from using the wearables [168, 26]. Moreover, multiple sensors are required to capture full-body motion, e.g., account for wrist, arm, and legs simultaneously. The video-based systems tackle some of the drawbacks of wearable systems. They usually use an RGB video camera, depth camera, or other motion capture systems, such as Microsoft Kinect [121] and Intel RealSense sensors [169]. The user only needs to perform some actions in front of a camera instead of wearing multiple sensors. Besides, video-based methods can reconstruct multiple essential body parts, such as the neck, wrist, shoulder, and ankle [170]. Then, they can model real-time skeleton movement by connecting these points [61, 170]. However, video-based systems also face critical challenges that limit their practicality. First, they require a strict

environment setting, such as lighting and camera placement, which significantly affects accuracy. Second, privacy is a more complex issue since many users do not want to share their camera access and videos. Consequently, the challenges discussed in this paragraph limit the use of video-based rehabilitation assistive systems at home.

With the recent advances in mmWave technology, Radio Frequency (RF) imaging has emerged as a promising technique that can address the limitations of wearable and video-based rehabilitation systems [171]. Small form-factor and low-power mmWave radars have become commercially available [172]. These devices provide a high-resolution 3D point cloud representation, which can be processed locally using edge artificial intelligence (AI) algorithms to reconstruct human motion. Since they generate and transmit RF signals towards the target, they can maintain a robust operation under poor lighting and weather conditions. They also address privacy concerns since mmWave radar signals do not involve any video images or facial information. *However, existing mmWave radar techniques have primarily been limited to object detection and target localization since it represents the scene with a reflection point cloud instead of the true color image* [65, 35].

This chapter proposes a novel real-time mmWave-based Assistive Rehabilitation System (MARS) to monitor patient movements in home environments accurately and provide real-time feedback. MARS tracks the patient movement using a low-cost mmWave radar [172]. Unlike prior work that uses the point cloud for activity recognition and localization [173, 65, 35, 64], our novel pre-processing algorithms and convolutional neural network (CNN) design convert the radar point cloud to

3D joint coordinates. *This unique capability enables MARS to produce real-time skeleton movements without using any video images or facial information.* Hence, its output is compatible with more expensive and complex video-based systems. Furthermore, MARS supports angle and speed estimations of limbs as well as posture correction. We evaluated MARS empirically by performing experiments with *70 minutes of exercise data (40,083 frames) from ten popular rehabilitation movements*. We also collected reference data using Microsoft Kinect V2 sensor [121] during these experiments. Experimental results show that MARS accurately estimates the 3D coordinates of 19 joints with only 5.87 cm mean absolute error (MAE). For more details about comparative results, please refer to Section 4.4.2. MARS also offers joint angle and velocity estimation as the feedback of the rehabilitation movements. The average MAE of MARS in the knee and the elbow angle estimation is 6° and 12°, respectively. Finally, we implement the proposed approach on the Nvidia Jetson Xavier-NX board [174].

Our experiments show that MARS can process well over 9,000 frames per second with less than 500 µJ energy consumption per frame. Hence, it can be used reliably for home-based rehabilitation systems.

In summary, the major contributions of this chapter are as follows:

- A low-cost and low-power mmWave-based assistive rehabilitation system that accurately reconstructs 19 human joints and skeleton movements using mmWave point cloud data,
- A novel pre-processing method that transforms the raw 5D time-series point cloud with irregular length and random order to a 3D 5-channel stacked

feature map; a CNN for processing the proposed feature map to 3D spatial coordinates of human joints,

- A *first-of-its-kind* rehabilitation movement dataset using mmWave point cloud, including 70 minutes of ten distinct rehabilitation movements performed by four human subjects with 19 human joints data and 40,083 labeled frames and their video demonstrations to the public,
- Experimental evaluations show, on average, about 5 cm localization error in 3D space, and 6° error for the knee angle, and 12° error for the elbow angle.

4.2 mmWave primer

Frequency Modulated Continuous Wave (FMCW) mmWave radar has recently attracted significant attention, especially in automotive and industrial applications. The fundamental component of FMCW is a chirp signal, which is a sinusoid wave whose frequency increases linearly with time [175, 176, 177]. Due to this characterization, a chirp signal is typically displayed by a linear frequency versus time plot, as illustrated in the top left part of Figure 4.1. The chirp signal is uniquely defined by its start frequency (f_c), duration (T_c), and bandwidth (B). The bandwidth to duration ratio gives the chirp slope (S), i.e., the rate at which the signal frequency increases ($S = B/T_c$).

The FMCW radar synthesizes a sequence of chirp signals to form a frame. For instance, Figure 4.1 illustrates a frame with N back-to-back chirp signals. It transmits the chirp frame using a transmitter (TX) antenna. If any object is in the

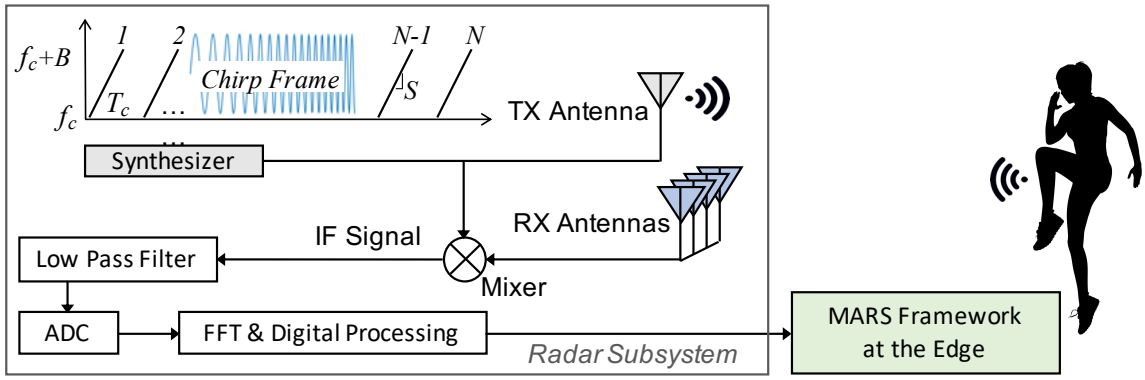


Figure 4.1: Overview of the mmWave imaging system.

vicinity, it reflects off the chirp frame. Then, the FMCW radar receives the reflected signals at the receiver (RX) antennas. Note that both TX- and RX-signals are chirps with different instantaneous frequencies and phases. A mixer module in the radar processes these signals to produce an *intermediate frequency* (*IF*) signal [175], which is another sinusoid with the following instantaneous frequency (f_{IF}) and phase (ϕ_{IF}):

$$f_{IF} = f_{TX} - f_{RX}, \quad \phi_{IF} = \phi_{TX} - \phi_{RX} \tag{4.1}$$

Suppose only one object reflects the chirp frame with distance d from the radar. The round-trip delay of the received signal can be found as $\tau = 2d/c$, where c is the speed of light. Since the received signal is a replica of the transmitted frames delayed by τ, the frequency of the IF signal will be $St - S(t - \tau) = S\tau$. That is, the IF signal has a single tone when only one object reflects the chirp signal. We can find the frequency of this tone as $S\tau = 2dS/c$. When the chirp frame is reflected from multiple objects or different parts of the body, the mixer will produce an IF signal with multiple tones, a.k.a., beat frequencies. FMCW radar chips extract the

IF signal tones by computing frequency spectrum using the fast Fourier transform (FFT), as depicted in Figure 1. As in the single object case, the frequency of each tone is proportional to the distance of the corresponding object. The IF signal is processed in the digital domain to map the tones into range bins using a *range FFT* process [177]. Note that the range resolution is inversely proportional to the chirp bandwidth:

$$d_{res} = \frac{c}{2B} \tag{4.2}$$

where c is the speed of light and B is the chirp bandwidth.

Another essential metric besides the range is the velocity of the detected objects. FMCW radars compute the velocity using the phase changes in the IF signal across multiple chirps. This process converts small displacements of the object to a phase difference in the IF signal. As in the range detection case, there may be multiple objects with equal distance from the radar but with different relative velocities. The chirps in the transmitted and received frames (N chirps in Figure 1) are processed by a second FFT, called *Doppler-FFT* [177], to resolve the velocity of different objects. After this step, the radar can produce a range-Doppler heat map to detect object velocities. The velocity resolution of the radar is inversely proportional to the frame time as:

$$v_{res} = \frac{\lambda}{2NT_c} \tag{4.3}$$

where λ is the wavelength, N is the number of chirps, and T_c is the time between two chirps. For instance, v_{max} is 39 m/s given T_c is 25 µs, which is significantly faster than human motion. Finally, the FMCW radar filters out the noise interfer-

ence using a *noise elimination* algorithm, such as the built-in constant false alarm rate (CFAR) [178], used in this work.

The last metric estimated by the radar is the angle of arrival (AoA). AoA is defined as the angle of a reflected signal with the horizontal plane [179]. Angle estimation requires at least two RX antennas and it is calculated by $\theta_{res} = \lambda/N_{RX}N_{TX}d\cos(\theta)$, where λ is the wavelength, N_{RX} is the number of receiver antennas, N_{TX} is the number of transmitter antennas, d is the distance between two consecutive receiver antenna, and $\cos(\theta)$ is the cosine of the angle between two receivers. Note that the resolution is often quoted assuming that $d = \frac{\lambda}{2}$ and $\theta = 0$, such that $\theta_{res} = \frac{2}{N_{RX}N_{TX}}$ (radians) [180]. The radar chip estimates the angle of arrival by using the phase change in the 2D-FFT peak caused by the different distances from the object to each antenna. This FFT is referred to as the *angle FFT*, which outputs the azimuth angle divided by elevation angle.

Finally, the radar receives multiple RX signals back for all the chirps the TX antennas sends. Each object (or body part) that reflects an RX signal is referred to as a point. For each point p_i in the frame, the radar chip calculates its 3D coordinates by using the result of range FFT after the noise elimination algorithm. Moreover, it computes the Doppler velocity and reflection intensity. Multiple points within each frame form the *point cloud*, which is formatted as follows:

$$p_i = \{x_i, y_i, z_i, D_i, I_i\}, i \in [0, N_P] \tag{4.4}$$

where x_i, y_i, z_i represents the spatial coordinates of the point, D_i denotes the Doppler velocity, I_i denotes the signal intensity, and N_P denotes the total number

Table 4.1: List of major parameters and variables related to mmWave and their values in this work.

Symbol	Description	Values	Symbol	Description	Values
f_c	Starting frequency	77 GHz	θ_{res}	Angle resolution	9.55°
T_c	Chirp duration	32 µs	N_{RX}	No. of RX antennas	4
B	Bandwidth	3.20 GHz	N_P	Maximum points detectable per frame	64
S	Slope of chirp	100 MHz/µs	f_{IF}	Frequency of IF signal	NA
N	No. of chirps per frame	96	ϕ_{IF}	Phase of IF signal	NA
d_{res}	Range resolution	4.69 cm	τ	RX signals time difference	NA
v_{max}	Maximum Velocity	5.69 m/s	D_i	the i^{th} point's Doppler velocity	NA
v_{res}	Velocity resolution	0.35 m/s	I_i	the i^{th} point's reflection intensity	NA
x_i, y_i, z_i	3D coordinates of the i^{th} point	NA	p_i	Point representation of the i^{th} point	NA
N_{TX}	No. of TX antennas	3			

of points in this frame. Note that N_P will be zero, i.e., there will not be any points when no object is detected. On the contrary, the number of detected points can exceed the radar chip's capacity if too many signals are reflected. Therefore, radar chips limit the maximum value N_P can take.

The radar parameters used in this study are shown in Table 4.1. In our work, the first 64 reflected points are processed, i.e., $N_P = 64$. The IWR1443 radar sensor is configured with three TX antennas and four RX antennas. The frame duration and sampling rate are set to 100 ms and 2.49 Msps, respectively. With these parameters, the radar has a maximum detection range of 3.37 m, maximum detection velocity of 5.69 m/s, a range resolution of 4.69 cm, and velocity resolution of 0.35 m/s. For more mmWave radar details, we refer the readers to recent tutorials [177, 176, 180].

4.3 Overview of MARS framework

This section presents the proposed MARS framework that processes the raw mmWave point cloud data to provide rehabilitation feedback to the user. To provide accurate and relevant feedback, MARS tracks the following fundamental attributes in real-time: The 3D position (x, y, z coordinates) and velocity (along x, y, z dimensions) of 19 joints, four key angles, as listed in Table 4.2. Furthermore, it provides correction feedback on ten commonly used postures shown in the last row of Table 4.2. MARS accomplishes these tasks by following the following steps outlined in Figure 4.2:

1. Use an FMCW radar to collect point cloud data, as described in Section 4.2,
2. Pre-process the point cloud to construct robust and delay-invariant features (Section 4.3.1),
3. Infer 3D joint positions using the new features and a CNN architecture (Section 4.3.2),
4. Produce user feedback by converting the 3D joint positions to joint velocity and angle estimations (Section 4.3.3).

4.3.1 Point Cloud Pre-Processing

Input data: Challenges and reformatting

The primary input to MARS is a point cloud arranged as five-dimensional (5D) time-series data. Each point consists of the x, y, z coordinates of the points that

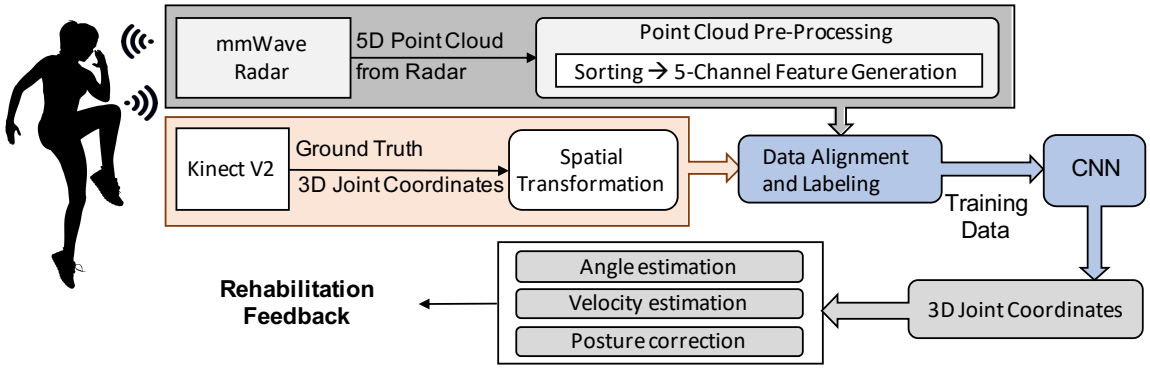


Figure 4.2: Overview of proposed MARS framework and its interaction with radar and Kinect V2.

Table 4.2: MARS provides 3D joint positions and the velocity of 19 joints listed in the first row. It also estimates the angles in the second row and provides posture correction feedback for ten movements in the last row.

3D joint position estimation	SpineBase, SpineMid, Neck, Head, SpineShoulder, ShoulderLeft, ElbowLeft, WristLeft, ShoulderRight, ElbowRight, WristRight, HipLeft, KneeLeft, AnkleLeft, FootLeft, HipRight, KneeRight, AnkleRight, FootRight
3D joint velocity estimation	
Angle estimation	Left elbow, Right elbow, Left knee, Right knee
Posture correction feedback	Left upper limb extension, Right upper limb extension, Both upper limbs extension, Left front lunge, Right front lunge, Squat, Left side lunge, Right side lunge, Left limb extension, Right limb extension.

reflected the TX-signal (x, y, z), Doppler velocity D, and the reflection intensity I, as described in Section 4.1. The FMCW radar stores the first N_P points to form a data frame (in this work N_P=64, as shown in Table 4.1). If fewer than N_P body parts reflect the chirp signals, fewer than 64 points will be received. In these cases, the rest of the frame is padded with zeros to obtain a uniform size $(N_P \times 5)$ input

frames.

Figure 4.3 depicts a sample input frame from different perspectives. The triangle marker represents the radar location, which is also set at the origin $(0, 0, 0)$. Figure 4.3(a) shows that point positions in 3D, while the other plots show their projections to 2D coordinates. Similarly, Figure 4.3(a) illustrates the Doppler velocity, which indicates the relative velocity from the detected point to the radar. Finally, the colors in the figures represent the energy intensity of the reflected signals.

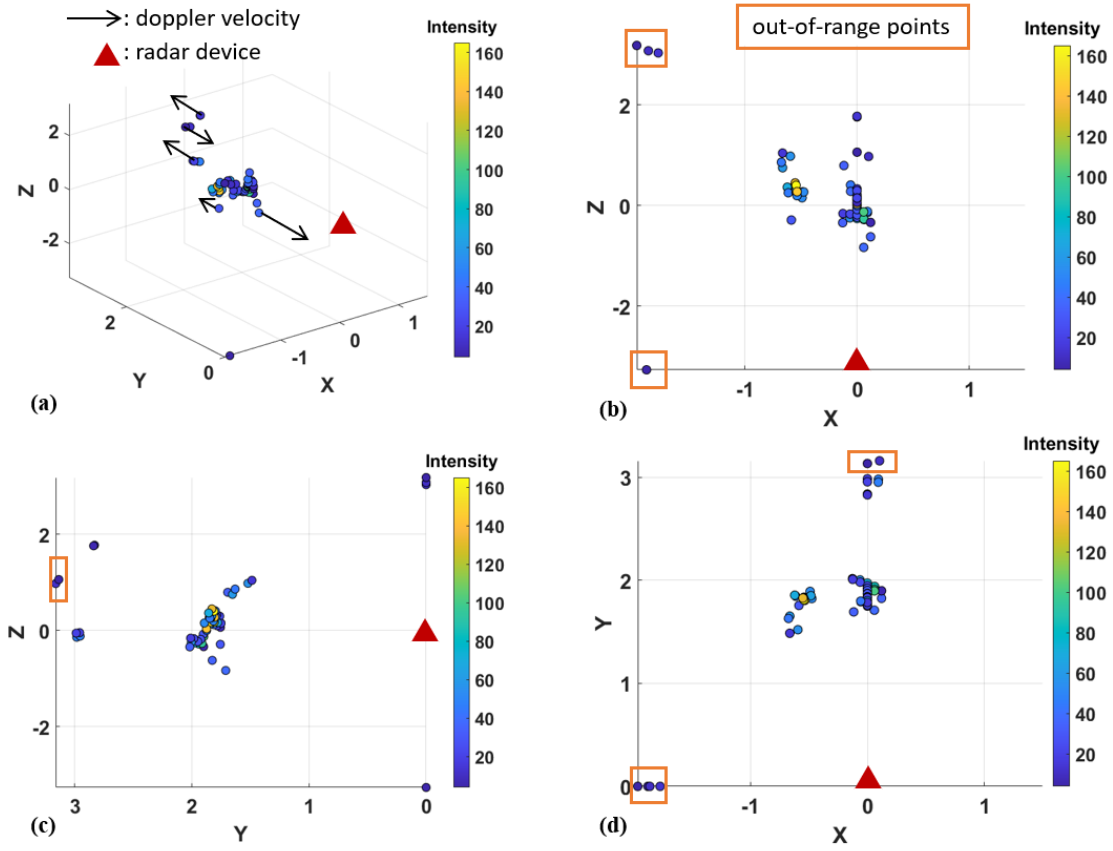


Figure 4.3: mmWave point cloud representation for one frame. (a), (b), (c), and (d) shows the 3D view, front view, side view, and top view, respectively.

Feature generation for CNN

Note that the reflected chirp signals arrive at the radar in random order due to slight variations in the body posture and round-trip delay, as illustrated in Figure 4.4(a). Therefore, the order of the points in a frame is random, i.e., the same point may be in different positions across consecutive frames. Although CNNs have a decent shift, scaling, and rotation invariance, random data ordering poses a challenge to CNN design. Furthermore, the input data must have a fixed shape. To address these issues, we propose a pre-processing algorithm that consists of sorting and matrix transformation. We sort the points in each frame in ascending order of x, y, and z coordinates. The points are sorted first based on their x coordinates in ascending order. At the second level, the points with the same x coordinate are sorted by their y coordinates. Finally, for the data points with the same x and y coordinates, we sort them by z coordinates in ascending order, as illustrated in Figure 4.4(b). Note that this sorting does not change the distances between the points since we only change the order of the inputs to the CNN.

After the sorting phase, the dimension of the input features is 64×5, i.e., the input data is arranged as a column vector with 64 rows each with 5 features. The transformation converts 64 rows into an 8x8 square matrix in the row-major order, similar to images commonly used in CNNs. Thus, the transformation reshapes the same data while preserving the values from the 64×5 matrix to an $8\times8\times5$ data structure. Since there are five dimensions (x, y, z coordinates, Doppler velocity, reflection intensity), we end up with five channels, each with an 8×8 feature map.

(a)

# Obj	X	Y	Z	Doppler	Intensity
11	-0.06	1.81	0.35	0.36	13
11	-0.12	1.86	0.22	0.36	15
11	-0.12	1.86	0.18	0.36	14
11	-0.18	1.91	0.22	0.36	10
11	0.61	1.88	0.89	0.36	10
11	0.00	1.80	0.21	-0.36	9
11	0.00	1.84	0.17	-0.36	13
11	-0.12	1.87	0.08	-0.36	16
11	0.00	1.80	0.04	-0.36	14
11	-0.12	1.93	0.05	-0.36	11

(b)

# Obj	X	Y	Z	Doppler	Intensity
11	-0.18	1.91	0.22	0.36	10
11	-0.12	1.86	0.18	0.36	14
11	-0.12	1.86	0.22	0.36	15
11	-0.12	1.87	0.08	-0.36	16
11	-0.12	1.93	0.05	-0.36	11
11	-0.06	1.81	0.35	0.36	13
11	0.00	1.80	0.04	-0.36	14
11	0.00	1.80	0.21	-0.36	9
11	0.00	1.84	0.17	-0.36	13
11	0.61	1.88	0.89	0.36	10

Figure 4.4: Input data (a) before and (b) after sorting.

Handling *out-of-range* "ghost images"

mmWave radar imaging can sometimes generate a point called "ghost image" that is outside the range of interest [181]. In assistive rehabilitation systems, the user is standing within a fixed distance away from the radar sensor. Hence, we divide the generated point cloud into two classes: *in-range* and *out-of-range*. The in-range point cloud is defined by lower and upper bounds on each dimension. In our implementation, we use the following ranges: $x \in \{-1\ m, 1\ m\}$ (horizontal width), $y \in \{0, 3\ m\}$ (depth), and $z \in \{-1\ m, 1\ m\}$ (vertical height). The points within these boundaries are considered in-range, while others are marked as out-of-range points. The out-of-range points are highlighted by rectangles in Figure 4.3 for illustration. The out-of-range points (i.e., ghost images) are inevitable in real application scenarios due to scattering. Therefore, MARS marks and includes out-of-range points in training and inference. Section 4.4.3 presents quantitative results and discusses the implications of this choice.

4.3.2 CNN Architecture Design

The next step is converting the feature maps depicted in Figure 4.3 into actual 3D joint positions. This challenging task is accomplished using a CNN architecture that outputs the x, y, and z coordinates of 19 joints, as illustrated in Figure 4.5. The input layer of the CNN takes the stacked 5-channel feature map as the input. Two consecutive convolution layers follow the input layer with 16 and 32 channels, respectively. After performing the convolutions, the data is passed to a flattening layer that generates the input vector for the fully connected (FC) layers. The first FC layer is with 512 neurons. The final output of CNN contains 57 neurons, which stand for 3D coordinates for the 19 joints. All activation functions are Relu except for the final FC layer, where we use linear activation. Finally, four dropout layers with probabilities of 0.3 and 0.4 are employed after the convolution layers and the fully connected layers to avoid excessive dependency on specific neurons.

The design choice of including Batch Normalization (BN) and max-pooling or leaving them out is vital in developing a CNN. The BN layer is commonly used to avoid significant data distribution changes after each mini-batch called "internal covariate shift." The max-pooling layer is used to maintain the feature invariance after image translation, rotation, and scaling by taking the maximum of a particular region. It also reduces model parameters while avoiding overfitting and improving the generalization ability of the model. MARS implements BN layers after the second convolution layer and the fully connected layers. We choose not to have a max-pooling layer since it loses local information, which is essential for the spatial coordinates regression task. We discuss these choices and present a comprehensive

quantitative evaluation in Section 4.4.3.

Ground truth and loss function

Training the proposed CNN architecture requires the ground truth, i.e., the reference positions of the target joint positions. In this work, we use a Kinect V2 sensor [121] to capture the reference coordinates. The Kinect sensor and the mmWave radar are placed on the same table, next to each other. This placement does not lead to spatial offsets in the y and z-axis between two sensors since their y and z coordinates are identical. However, there is a spatial offset. Firstly, the x-axis in the Kinect sensor's reference frame is inverted with respect to the mmWave radars x-axis. Thus, we take the additive inverse of all x-axis values during the pre-processing stage. Secondly, there is still a 3 cm offset in the x-axis since the sensors are placed next to each other. We do not manually calibrate this offset since the CNNs have decent shift-invariance. CNN itself can learn the spatial offset between the mmWave sensor and the Kinect sensor. The Kinect sensor's sampling rate is fixed at 30Hz, while the radar's frame duration is 100ms. Hence, we align the radar and the Kinect sensor data frame by frame. The frame alignment is achieved by connecting both devices to the same laptop and timestamping the data frames from each device. We find the closest timestamp in the Kinect sensor for each radar data frame and pair it with the radar data as its label[1].

For a given data frame, let x_i, y_i, and z_i be the *reference coordinates of joint* i, $1 \leqslant i \leqslant N_J$ from the Kinect sensor, where N_J is the number of tracked joints.

[1]The time difference between a data-label pair is less than 5ms by using the proposed time alignment method.

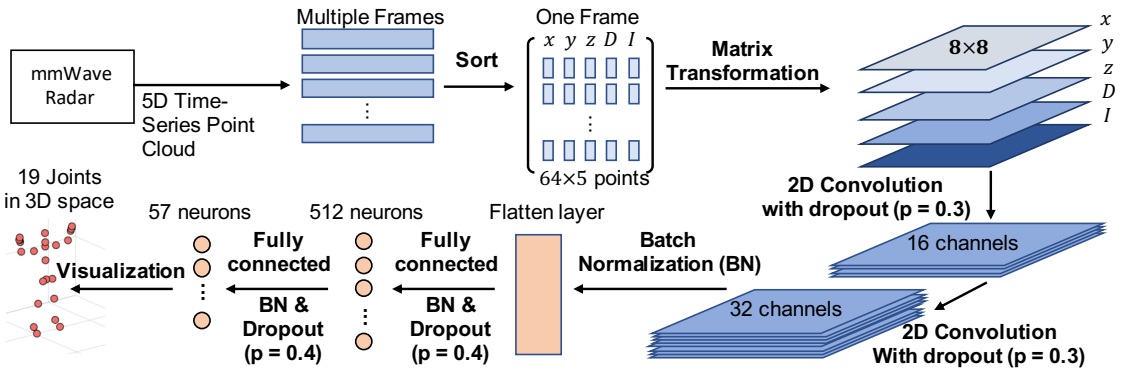


Figure 4.5: Point cloud pre-processing and CNN architecture

Similarly, let the corresponding estimates from MARS be $\hat{x}_i$, $\hat{y}_i$, and $\hat{z}_i$, respectively. We define the loss function as the mean squared error (MSE) between the reference positions and the estimations as follows:

$$\text{Loss}_{coor} = \frac{\sum_{i=1}^{N_J}(x_i - \hat{x}_i)^2 + \sum_{i=1}^{N_J}(y_i - \hat{y}_i)^2 + \sum_{i=1}^{N_J}(z_i - \hat{z}_i)^2}{3N_J} \tag{4.5}$$

An illustration of MARS estimating 19 human joints from the mmWave radar is shown in Figure 4.6. From left to right, the subfigure represents the point cloud generated by radar, estimation from MARS, and the ground truth from the Kinect V2 sensor. We observe that MARS reconstructs 19 human joints accurately. We show only five of the ten rehabilitation movements due to space limitation. The remaining five movements look very similar since they are mirrored versions of the same movements. *A live demo can be found on our GitHub page, where the dataset is released* [182].

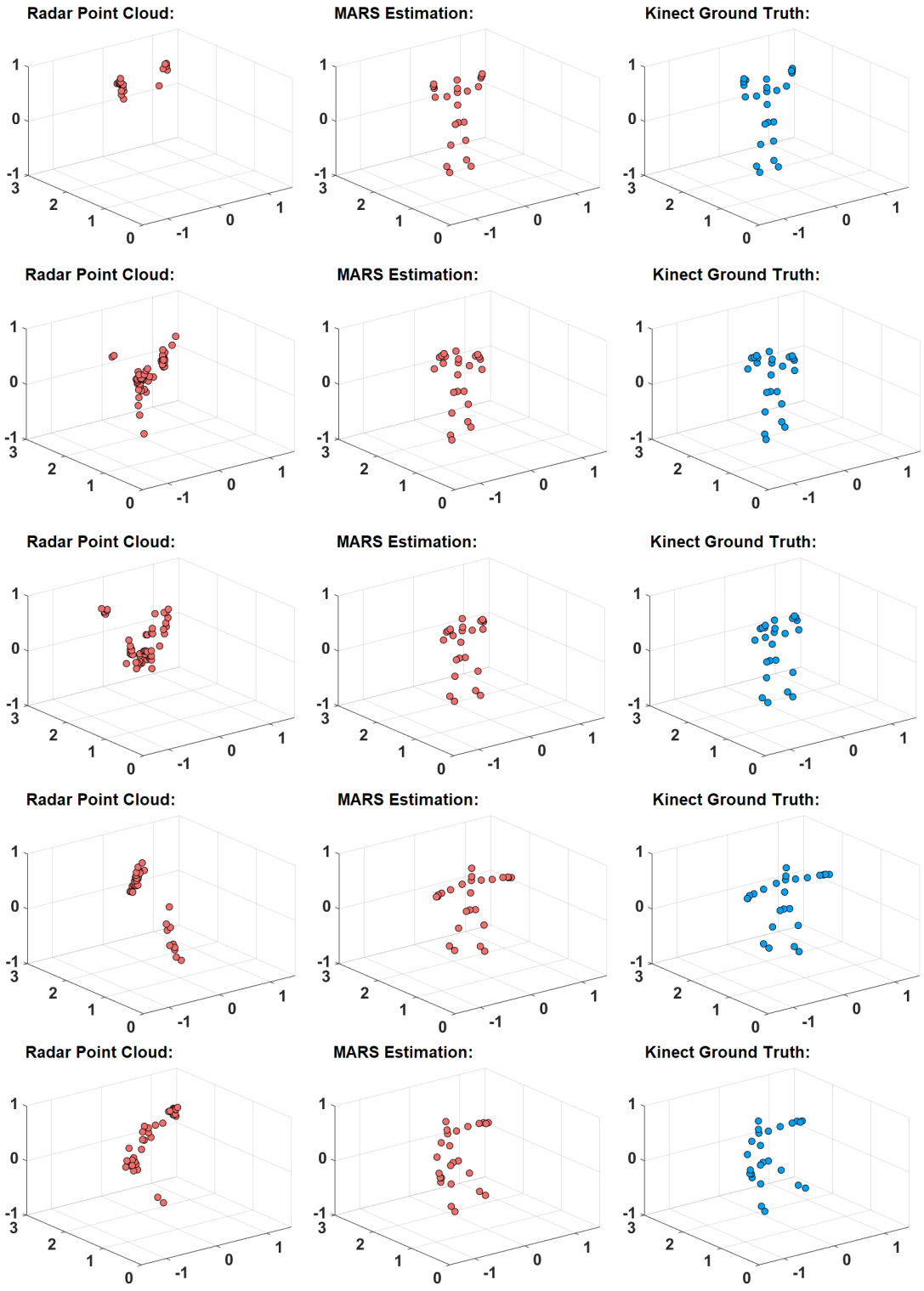


Figure 4.6: Demo of MARS reconstructing human joints from point cloud for *both upper limb extension, right front lunge, squat, left side lunge, and left limb extension*. From left to right, it shows radar point cloud, MARS estimation, and ground truth, respectively. *The accuracy of the estimations are analyzed in Section 4.4.*

4.3.3 Rehabilitation Movement Feedback to User

Velocity estimation

The CNN presented in Section 4.3.2 produces the 3D coordinates of 19 joints listed in Table 4.2. The next step is deriving the velocity of these joints. We find each joint's velocity by dividing its distance between two consecutive frames by the frame duration. One example is shown in Figure 4.7(d) for the squat movement. We first find the complete squatting frame (shown in solid line in Figure 4.7 (d)). Then, the corresponding positions in the previous frame are found. Finally, the ratio of the distance between two consecutive frames and frame duration gives the joint velocities. The ground truth velocity is derived using consecutive ground truth 3D coordinates reported by the Kinect sensor and their sampling times. For the squat example, the spinebase joint's velocity is used for evaluating the squat speed. In general, users can observe every joint's velocity when they perform different movements and adjust their pace accordingly.

Joint angle estimation

The joint coordinates found by the CNN are also used to find the angles between critical joints. This work focuses on the four most commonly used joint angles: right and left elbow angles, right and left knee angles, as listed in Table 4.2. The elbow angle is found using the shoulder, elbow, and wrist positions, as illustrated in Figure 4.7. We first calculate the skeleton length between the shoulder and elbow and the length between the elbow and wrist using their 3D coordinates. Then, the angle is obtained by using triangulation from the law of cosines. We follow the same

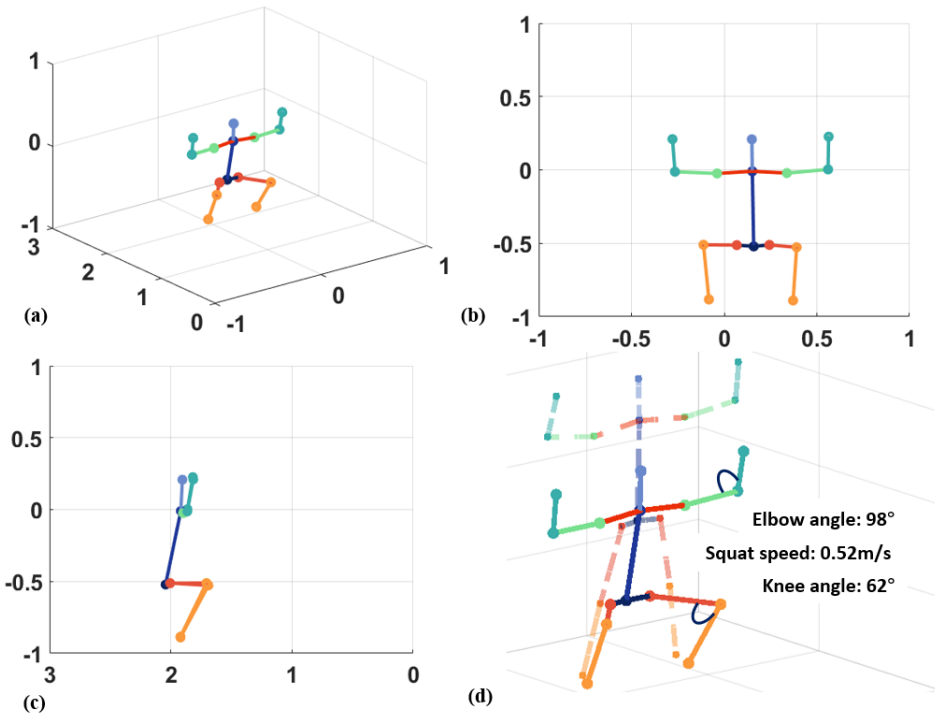


Figure 4.7: Angle estimation by MARS during *squat* movements. (a), (b), (c), (d) shows the 3D view, front view, side view, and a zoomed in version with the estimated angles and speed

procedures to calculate the knee angle using the hip, knee, and ankle positions. The ground truth angle is computed using the ground truth 3D coordinates reported by the Kinect sensor. As an example, Figure 4.7(d) illustrates a squat movement. We observe that the elbow and knee angles are 98° and 62°, respectively, for the complete squat.

Posture correction

Therapists can define *specific rehabilitation movements* for a given user, such as the squat movement illustrated in Figure 4.7. Then, the correctness of a movement

can easily be defined by setting acceptable ranges for relevant joints' velocities and angles. For example, the knee angles are essential for the squat movement. Thus, the user can set acceptable ranges for the knee angles, such as 55°–65° when the legs are stretched the most.

While the user performs the movements, MARS tracks the knee angles, as described in Section 4.3.3. Then, it compares the joint positions and angles to the acceptable ranges specified by the user (e.g., +/- 10% around ideal positions). The reconstructed skeleton is shown to the user with a transparent dash-line before the user's movement satisfies the acceptable range target, as shown in Figure 4.7 (d). The skeleton visualization becomes a solid line after the joint coordinates, angle, and velocities reach the set goals. The system can also support a sound played or a visual cue as feedback that indicates successful completion of the exercise.

4.4 Experimental Evaluation

4.4.1 Experimental Setup and Dataset

mmWave radar: The radar processing is performed on Texas Instruments (TI) IWR1443 Boost mmWave radar [172]. We use a Matlab Runtime implementation from TI [183] for the data acquisition. The detailed configuration and radar parameters are summarized in Table 4.1. The device is connected to a laptop through the UART interface. It starts acquiring the data from the Matlab Runtime using a frame duration of 100 ms. Note that the frame duration can be set to different values for different applications. Due to the bandwidth limitation, the least frame duration

we can set is 33.3 ms, equivalent to the 30 Hz sampling rate. We chose 100 ms (i.e., 10 Hz sampling rate) since it is enough for measuring human movement (the frequency of most voluntary human movements spans from 0.6 to 8 Hz [184]). The average power consumption of IWR1443 mmWave radar tested at power terminals is 2.1 W [185]

Kinect V2 sensor [121]: The ground truth reference is obtained using Microsoft Kinect V2. Both Kinect and radar are placed on a 1 m tall table while the subjects perform the instructed movements two meters away from the table. The Kinect V2 sensor is connected to a laptop through the USB port using an adaptor. It captures images with a 30 Hz sampling rate. Then, the images are processed using Matlab to identify the 3D coordinates of 19 human joints listed in Table 4.2. These positions are used as the labels during training and reference points for testing, as described in Section 4.3.2. The Kinect reference system requires a 12V 2.67A power adapter to work.

Hardware measurements: We implemented the proposed MARS framework, including all the pre-processing steps and the proposed CNN, on the Nvidia Jetson Xavier NX Development Kit [174]. The execution time and power measurements are presented in Section 4.4.5.

Open-source training and test datasets: We collected training and test data through user-subject studies, following an official protocol approved by our institution's IRB board. Each subject performed the ten movements listed in Table 4.2 (five of them are illustrated in Figure 4.6). This set of movements enables us to evaluate both the upper and lower body joints and associated angles. Each user performed each

movement for two minutes, i.e., approximately 20 minutes of data is collected in total. As a result, we obtained close to 10,000 data frames per user. Each frame contains data for 19 joints. Furthermore, the Kinect V2 reference data points have three dimensions, while the data points from the radar have five dimensions (3D coordinates, Doppler velocity, and reflection intensity). Hence, our reference data set from Kinect and radar contain close to 570K (10,000×19×3) and 950K (10,000×19×5) points for each subject, respectively. We emphasize that this is a large-scale dataset with a comparable size to other similar studies. More importantly, it is *the first rehabilitation movement dataset using mmWave point cloud with well-labeled joints.* Since home-based assistive rehabilitation systems, like MARS, are user-specific, evaluations even on one user are representative. Regardless, we repeated the evaluations with four different users to obtain a total of 2.28 million reference data points from Kinect V2 and 3.81 million data points from mmWave data. We plan to release this dataset to the public through Github [182] together with the existing demo.

CNN training details: We implemented the proposed CNN using Tensorflow 2.2.0 [186] with Keras 2.3.4 [187]. We use the Adam [188] as the optimizer with an initial learning rate of 0.001. The CNN is trained with a batch size of 128 for 150 epochs, where the validation loss converges at 0.01. Aiming for a personalized model, we split the data time-wise for training, validation, and testing. First, each movement data is divided into 60% (24,066 frames)-20% (8,033 frames)-20% (7,984 frames). Then, we take the first 60% of it for training, the next 20% for validation, and the last 20% for testing. We choose to use the 60%-20%-20% ratio instead of the fixed-length since some data is not exactly two minutes. These frames add up to

2.28 million data points from Kinect V2 and 3.81 million data points from radar, as described under the dataset. The training is performed on AMD Ryzen™ 7 3800X 8-Core 3.9 GHz and Nvidia RTX2080 with 8GB of graphics memory.

4.4.2 Accuracy of 3D Joint Position Estimation

Table 4.3 shows the detailed localization error for the 19 human joint points, illustrated in Figure 4.6. We use the mean absolute error (MAE), and root mean squared error (RMSE) metrics to evaluate MARS. To eliminate the system errors, we train ten different models and take the average. This methodology is applied to all quantitative results reported in this paper. The average MAE for all 19 joints is

Table 4.3: Average localization error for 19 human joints position.

	X (Horizontal) (cm)		Y (Depth) (cm)		Z (Vertical) (cm)		Average (cm)	
	MAE	RMSE	MAE	RMSE	MAE	RMSE	MAE	RMSE
SpineBase	5.67	8.22	3.55	4.96	5.96	7.83	5.06	7.00
SpineMid	6.16	8.90	3.07	3.97	6.80	8.94	5.34	7.27
Neck	6.78	9.80	3.39	4.32	7.58	9.97	5.92	8.03
Head	7.37	10.57	3.69	4.69	8.20	10.68	6.42	8.65
ShoulderLeft	6.92	9.91	3.39	4.39	6.88	9.00	5.73	7.77
ElbowLeft	7.52	10.23	4.37	6.00	8.19	10.74	6.69	8.99
WristLeft	10.34	13.76	5.07	6.80	13.57	18.14	9.66	12.90
ShoulderRight	6.75	9.69	3.78	5.05	7.04	9.21	5.86	7.98
ElbowRight	7.96	10.71	4.73	6.74	8.41	10.93	7.03	9.46
WristRight	10.74	14.18	5.22	7.26	14.14	18.68	10.03	13.37
HipLeft	5.63	8.13	3.56	4.99	5.84	7.67	5.01	6.93
KneeLeft	5.56	8.10	4.09	5.63	3.25	4.53	4.30	6.09
AnkleLeft	6.27	8.83	4.29	6.18	2.47	4.49	4.34	6.50
FootLeft	6.59	9.36	4.84	7.01	3.04	5.14	4.82	7.17
HipRight	5.55	8.04	3.66	5.06	5.91	7.77	5.04	6.96
KneeRight	6.11	8.53	4.38	5.83	3.60	5.33	4.70	6.57
AnkleRight	6.92	9.42	4.43	6.10	2.74	5.37	4.69	6.96
FootRight	7.38	9.99	4.57	6.51	3.22	5.81	5.05	7.44
SpineShoulder	6.62	9.57	3.22	4.10	7.40	9.72	5.75	7.80
MARS 19 points Avg.	6.99	9.79	4.07	5.56	6.54	8.94	5.87	8.10

6.99, 4.07, 6.54 cm for x-, y-, and z-axes, respectively. Similarly, the average RMSE of x-, y-, and z-axes are 9.79, 5.56, and 8.94 cm, respectively. In general, the x- and z-axes have larger errors than the y-axis since our movements involve intensive horizontal and vertical displacement of all body parts. In contrast, the error along the y- axis is minimal (3.07 cm–5.22 cm) due to the smaller displacement in depth.

The mean average absolute error of most joints is smaller than 8 cm. The most notable exceptions are the right and left wrist joints. An intuitive explanation is that joints related to hands need a higher resolution to localize. Since the mmWave radar's range resolution is 4.69 cm@3.20 GHz as mentioned in Section 4.2, it is challenging for the model to reconstruct these points. Since estimating human pose from mmWave point cloud is a relatively new research area, there are only a few studies to compare with [22]. MARS achieves 5% lower error with only half model parameters than the method proposed in [22], as explained in Section 4.4.3. By searching similar research areas, we note that the accuracy of MARS is competitive with the human pose estimation techniques [189]. Hence, MARS can provide reliable user feedback in home-based rehabilitation systems.

4.4.3 Ablation Study

We performed extensive ablation studies to demonstrate the necessity of each component in MARS and justify the design choice adopted by MARS.

Using out-of-range point clouds during training

As described in Section 4.3.1, some radar data frames may contain out-of-range points, also referred to as ghost images. It is possible to train MARS by *including* or *excluding* the out-of-range points, which constitute about 2% of all frames. To evaluate each choice's effectiveness, we first train the model with the frames that only contain the in-range point clouds (i.e., *out-of-range points are excluded*). Then, we obtain a different CNN model by using *all frames*. We observe that the model trained with all point clouds performs slightly better than the one trained with only in-range point clouds, as shown in Table 4.4. The out-of-range point clouds add noise to the inputs, making the CNN more robust. Furthermore, out-of-range points are inevitable during real use cases. Therefore, we conclude that they should be included in the training data.

Table 4.4: Comparison of average localization error for 19 human joints position between models trained with different point cloud range.

	X (Horizontal) (cm)		Y (Depth) (cm)		Z (Vertical) (cm)		Average (cm)	
	MAE	RMSE	MAE	RMSE	MAE	RMSE	MAE	RMSE
With "in-range" point cloud	7.75	10.73	4.18	5.79	7.11	9.63	6.35	8.72
With all point cloud	6.99	9.79	4.07	5.56	6.54	8.94	5.87	8.10

Different feature channels and projection for CNN

As discussed in Section 4.3.1, we have an 8×8 feature map for each channel, including x, y, z coordinates, Doppler velocity, and reflection intensity. We can combine and stack these feature maps in different ways to obtain a stacked feature map for the CNN. To find out the best option, we train different CNNs with four different

stacked feature maps and refer to the CNNs as *Configuration-1*, *Configuration-2*, *Configuration-3*, and *Configuration-4*. *Configuration-1* represents the CNN trained with feature maps only stacked with x, y, z three channels. *Configuration-2* represents the CNN trained with feature maps stacked with x, y, z, and Doppler velocity, four channels. *Configuration-3* represents the CNN trained with feature maps stacked with x, y, z, and reflection intensity, four channels. Finally, *Configuration-4* represents the CNN trained with feature maps stacked with x, y, z, Doppler velocity, and reflection intensity, five channels.

We observe that the *Configuration-1* model has the worst performance due to a lack of Doppler velocity and reflection intensity information, as shown in Table 4.5. *Configuration-2* and *Configuration-3* has slightly better performance since Doppler velocity or intensity information is introduced. *Configuration-4* performs the best since the 5-channel feature maps contain all the information, including x, y, z with both Doppler and intensity. Note that because of weight sharing in CNN, adding channels in input only increases negligible parameters in the model, as shown in Table 4.5. We then apply *Configuration-4* in MARS.

A recent prior study, mmPose [22], projects the point cloud to two different planes as features and then concatenates them. However, the projection step increases the number of parameters hence increases the computation cost. Decomposing features into different projections increases the model parameters linearly since we need to perform the convolution multiple times for each decomposed feature map and then concatenate them. To analyze the effect of projections, we implement the mmPose model [22] and compare it with MARS. To make a fair comparison, we

Table 4.5: Comparison of average localization error for 19 human joints position across models trained with different feature channels.

	X (Horizontal) (cm)		Y (Depth) (cm)		Z (Vertical) (cm)		Average (cm)		
	MAE	RMSE	MAE	RMSE	MAE	RMSE	MAE	RMSE	No. of parameters
Configuration-1	7.37	10.37	4.64	6.48	7.06	9.77	6.36	8.87	1,094,827
Configuration-2	7.33	10.20	4.37	6.02	7.00	9.52	6.23	8.58	1,094,971
Configuration-3	6.94	9.80	4.46	6.08	6.62	9.10	6.01	8.33	1,094,971
Configuration-4	6.99	9.79	4.07	5.56	6.54	8.94	5.87	8.10	1,095,115
mmPose[22]	6.80	10.21	4.79	6.67	6.94	9.86	6.18	8.91	2,281,739

reduce mmPose's feature map size from 16×16 to 8×8 since the maximum number of points per frame N_J is 64 in our dataset. As shown in Table 4.5, the CNN used in MARS has 1,095,115 parameters, which is half of 2,281,739 in mmPose. Moreover, the MAE of the 3-axis localization error of MARS is 5.87 cm, lower than 6.18 cm of mmPose. The result shows that MARS feature generation reduces the model complexity while obtaining higher performance. We also emphasize that mmPose requires two radars, while MARS uses only one radar, making it more practical and easier to use. Furthermore, MARS handles complex rehabilitation movements, whereas mmPose is developed to analyze joint movements during walking.

CNN architecture design

Section 4.3.2 presented the use of BN and max-pooling concepts in CNN architectures. This section justifies incorporating or excluding BN and max-pooling by training different models with or without them. We first train the model without BN and max-pooling as the baseline. Then, we train another model called "Baseline with BN", which adds a BN layer after each convolution layer and fully connected layer. Similarly, we train another model called "Baseline with max-pooling", which

adds a max-pooling after the convolution layers. Finally, we train a model called "Baseline with both", which adds a max-pooling layer after each BN layer except the final BN layer after the fully connected layer. We observe that the "Baseline with BN" gives the best result and "Baseline with both" gives the worst, similar to the baseline, as shown in Table 4.6. BN successfully avoids the internal covariate shift. Max-pooling is not a good option because our model maps mmWave points to the joints point such that this task is essentially a mapping regression problem. Max-pooling introduces information loss when taking the local maximum of the features such that the model cannot leverage every joint's coordinates accurately. We then decide to keep only BN in MARS.

Table 4.6: Comparison of average localization error for 19 human joints positions position across models trained using CNN architecture with different components.

	X (Horizontal) (cm)		Y (Depth) (cm)		Z (Vertical) (cm)		Average (cm)	
	MAE	RMSE	MAE	RMSE	MAE	RMSE	MAE	RMSE
Baseline	7.52	10.55	4.84	6.65	7.36	10.05	6.57	9.08
Baseline with BN	6.99	9.79	4.07	5.56	6.54	8.94	5.87	8.10
Baseline with max-pooling	7.63	10.38	4.65	6.64	7.15	9.57	6.48	8.86
Baseline with both	7.44	10.20	4.45	6.09	7.22	9.81	6.37	8.70

Training with user-specific or aggregate data

Our dataset contains close to 10,000 frames per user, which alone is sufficient to train custom user-specific models. This section moves one step forward to analyze the ability of MARS to generalize to multiple users, considering that several people in the same household can use a shared setup.

To this end, we investigate the performance between the model trained with

individual users and all users. Using the same CNN architecture, we first train the models with individual user data then train a model with all users. The first four rows in Table 4.7 summarize the MAE and RMSE for the test data when the CNN is trained for a single user. We observe that the maximum MAEs for the x-, y-, z-axes are 8.25 cm, 5.23 cm, and 6.56 cm, respectively. The fifth row shows that the MAE and RMSE average across all subjects. The corresponding average MAE and RMSE across all subjects and dimensions are 6.29 cm and 8.40 cm, respectively.

The last row in Table 4.7 shows the MAE and RMSE when a single model is trained using data from all users. The resulting modeling errors are very similar to the performances of user-specific models for each subject. We also note that the model trained for all users has a slightly higher MAE of 6.54 cm along the z-axis than x-axes. This behavior is attributed to the height differences between all our subjects (160 cm-192 cm) since the z-axis represents the vertical dimension. Overall, these results show that multiple people in the same household can easily use a shared MARS profile. We also note that multiple user profiles can also be used depending on the user preference.

Table 4.7: Comparison of average localization error for 19 human joints position between models trained with individual user and all users.

	X (Horizontal) (cm)		Y (Depth) (cm)		Z (Vertical) (cm)		Average (cm)	
	MAE	RMSE	MAE	RMSE	MAE	RMSE	MAE	RMSE
Subject 1 (M)	8.01	10.78	5.23	6.70	6.56	9.08	6.60	8.85
Subject 2 (M)	7.80	10.31	4.95	6.57	5.37	7.41	6.04	8.10
Subject 3 (M)	8.25	10.80	4.82	6.42	6.13	8.07	6.40	8.43
Subject 4 (F)	7.58	10.34	5.08	6.89	5.70	7.45	6.12	8.23
Avg. of all subjects	7.91	10.56	5.02	6.65	5.94	8.00	6.29	8.40
All subjects	6.99	9.79	4.07	5.56	6.54	8.94	5.87	8.10

4.4.4 Joint Angle Estimation

The angle estimation is essentially a nonlinear transformation of the coordinate estimations. Therefore, the estimation error in the joint angle is related to the localization error trend. The average MAE of MARS in estimating left elbow angle, right elbow angle, left knee angle, and right knee angles are 12°, 13°, 7°, 6°, respectively. We observe that the elbow angles have a higher error than the knee angles, as summarized in Table 4.8. The higher error stems from using the WristLeft and WristRight joint positions to calculate the elbow angles. Since these two joints have higher estimation errors, as discussed in Section 4.4.2, they increase the error in the elbow angle estimates. Moving the users closer to the radar might reduce this error since the radar can detect more hands-related points, but the system may also lose the entire body's aspect. Further improvement of elbow angle estimates will be considered in our future work.

Table 4.8: MAE of MARS joint angle estimation.

MAE of Left Elbow	MAE of Right Elbow	MAE of Left Knee	MAE of Right Knee
12°	13°	7°	6°

4.4.5 Power and Execution Time Analysis

MARS's ability to run on hardware within acceptable power and execution time is crucial for its practicality on low-power edge devices [190, 191]. To evaluate this ability, we implemented MARS on Nvidia Jetson Xavier NX Development Kit [174]. The board has a 6-core ARM CPU, 384 Nvidia CUDA cores, and 48 tensor

processing units.

We focus on real-time model inference since the training is usually done using more powerful computing resources, and inference is more meaningful during rehabilitation exercises. The computing power of edge devices varies widely. To do a comprehensive study considering most use-cases, we set five different hardware configurations with different numbers of active CPU cores and maximum CPU/GPU operating frequencies, as shown in the upper part of Table 4.9. We sort five configurations in descending order of computation power. For different configurations, the total inference time for all 40,083 frames ranges from 2.5 s to 4.1 s, as shown in Table 4.9. The total CPU and GPU power consumption decreases from 3921.4 mW to 1950.6 mW as we move from Config. 1 to Config. 5. The corresponding total power consumption decreases from 6865.2 mW to 4700.2 mW. We find the average inference time and energy consumption by dividing these measurements into the total number of frames (40,083). The average frame processing time ranges

Table 4.9: Power and latency results for model inference of MARS on Jetson Xavier NX.

	Config. 1	Config. 2	Config. 3	Config. 4	Config. 5
Online CPU	2	2	4	4	6
Max CPU frequency (MHz)	1900	1500	1400	1200	1400
Max GPU frequency (MHz)	1100	800	1100	800	1100
Total time (second)	2.5	3.2	3.7	3.9	4.1
CPU-GPU power (mW)	3921.4	2366.8	2075.7	1968.1	1950.6
Total power (mW)	6865.2	5211.4	4854.4	4723.1	4700.2
Total energy (J)	17.3	16.5	17.9	18.2	19.4
Time per frame (μs)	64.4	81.1	94.4	98.9	105.6
Energy per frame (μJ)	442.3	422.9	458.3	467.5	496.4

from 64.4 μs to 105.6 μs, which shows that MARS can process well over 9,000 frames per second with less than 500 μJ energy consumption per frame. These results show that MARS provides a high-performance and energy-efficient solution for reliable and privacy-preserving home-based rehabilitation.

5 *MRI*: MULTI-MODAL 3D HUMAN POSE ESTIMATION DATASET USING MMWAVE, RGB-D, AND INERTIAL SENSORS

5.1 Background, Motivation and Contributions

Many existing studies rely heavily on processing RGB frames from color cameras for human pose estimation [7, 8, 9, 10, 11, 12]. RGB image and video frames are the most common input types since they offer an non-invasive approach for HPE. However, the image quality depends heavily on the environmental setting, such as light conditions and visibility [36]. Moreover, using image and video data poses significant privacy concerns, especially in a household environment. Finally, the data-intensive nature of real-time video processing requires computationally powerful equipment with high cost and energy consumption.

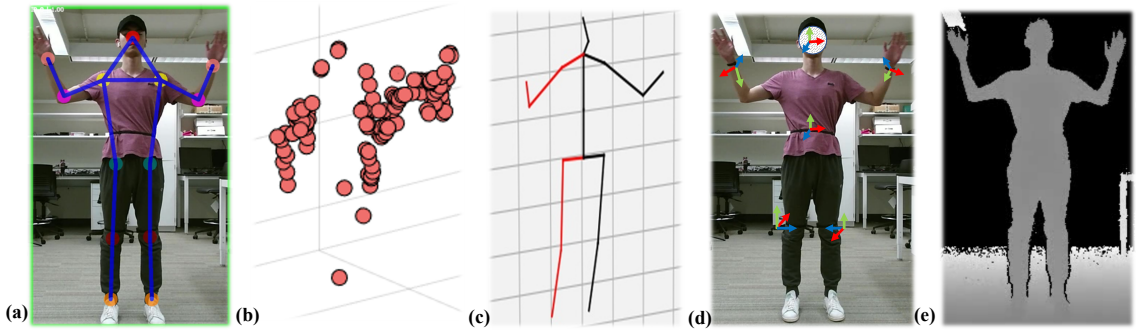


Figure 5.1: Overview of all modalities and annotations in *mRI* dataset. (a) 2D human keypoints with bounding box on RGB image, (b) 3D mmWave point cloud, (c) 3D human skeletons, (d) IMU rotations, (e) depth image.

Frame quality, privacy, and computational power drawbacks of video processing can be addressed by emerging *complementary sensor modalities*, such as

lidar, millimeter wave (mmWave) radar [36, 21, 19], and wearable inertial sensors [13, 14, 15, 16, 17, 18, 33], The point cloud from lidar overcomes frame quality and privacy challenges. However, it has a high cost and computation power requirements to process the data, making it unsuitable for indoor applications such as rehabilitation. In contrast, mmWave radar can generate high-resolution 3D point clouds of objects while maintaining low cost, privacy, and computational power advantages. Similarly, wearable inertial sensors provide accurate rotation and acceleration information regarding joints with low cost and computational power requirements [14, 15, 16, 33], yet at a price of body worn sensors.

High-quality and large-scale datasets provide a vital foundation for algorithm development. To catalyze research in HPE, this chapter (*mRI*) combines mmWave radar, RGB-Depth (RGB-D), and Inertial sensors to exploit their complementary advantages. We present a comprehensive 3D human pose estimation dataset performed by 20 human subjects, consisting of more than 160k synchronized frames from three sensing modalities. The contributions and unique aspects of *mRI* are as follows:

- **Multiple Sensing Modalities.** *mRI* consists of mmWave point cloud, RGB frames, depth frames, and inertial signals. The experimental data is captured using a commercial low-power, and a low-cost mmWave radar, two depth cameras, and six high-accuracy inertial measurement units (IMUs). All sensors are temporally synchronized and spatially calibrated. To the best of our knowledge, *mRI* is the first dataset that combines these complementary modalities.

- **Healthcare Movements Focus.** We use ten clinically-suggested rehabilitation movements that involve the upper body, lower body, and the major muscles related to human mobility, as described in Section 5.2.2. These movements are crucial for patients to recover from sequelae of central nervous system disorders, such as Parkinson's disease (PD) and cerebrovascular diseases (e.g., stroke). Hence, the *mRI* dataset can serve as a reference from healthy subjects, while the experimental methodology can enable future studies with patients.

- **Flexible Data Format and Extensive Benchmarks.** We release the raw synchronized and calibrated sensor data and a comprehensive set of benchmarks for 2D/3D human pose estimation and action detection using multiple modalities (see Section 5.3). The proposed end-to-end pipeline pre-processes the raw data into the point cloud, features, and 2D/3D keypoints. In addition, all manually-labeled actions annotations and 3D human key points ground truth are released to public, as detailed in Section 5.2.2.

- **Low-Power & Low-Cost Requirements.** Widespread use of home-based rehabilitation depends critically on the affordability and operating cost of the deployed systems (see Section 5.2.1). Our *mRI* dataset and findings pave the way to sustainable systems with low-power and low-cost sensors and edge devices. For example, only mmWave radar and IMU sensors can be used in the field after they are trained with all three modalities (including RGB-D) in a clinical environment.

5.2 Overview of *mRI* Dataset

mRI includes 3D point cloud from mmWave, RGB frames and depth maps from RGB-D cameras, joints rotations and accelerations from wearable IMU sensors, as well as annotations of 2D keypoints, 3D joints, and action labels of 12 clinically relevant movements. *mRI* consists of 300 time-series sequences with 160K synchronized frames and more than 5M total data points from all sensors, from 20 subjects. We hope that our dataset will contribute to the multi-modal machine learning community, and facilitate applications of HPE for rehabilitation and other healthcare problems. In what follows we describe the hardware system to capture the data and the data collection process.

5.2.1 Capturing Multi-Modal Signals for Human Pose Estimation

To capture multi-modal data, we designed a sensor system composed of one mmWave radar, two sets of RGB and depth cameras, and six wearable IMUs. Detailed specifications and features of all sensors are shown in Table 5.1. The mmWave radar and two Kinect V2 sensors are placed on a desk 2.4 m away from the sub-

Table 5.1: Comparison across sensors. #: Number of sensors. Freq.: Sampling frequency. Con.: Type of connection to the host PC. Privacy indicates privacy-preserving ability. Anti-interference represents how much it is affected by environmental factors like non-ideal lighting.

	#	Freq.	Con.	Power	Privacy↑	Anti-inter.↑	Intrusive	Output
mmWave [172]	1	10 Hz	Wired	2.1 W	⋆⋆⋆	⋆⋆⋆	No	Point cloud
RGB [121]	2	30 Hz	Wired	16 W	⋆	⋆	No	RGB frame
Depth [121]	2	30 Hz	Wired	16 W	⋆⋆	⋆⋆	No	Depth and infra-red frame
IMU [192]	6	50 Hz	BLE	120 mW	⋆⋆⋆	⋆⋆⋆	Yes	Accelerations and quaternions

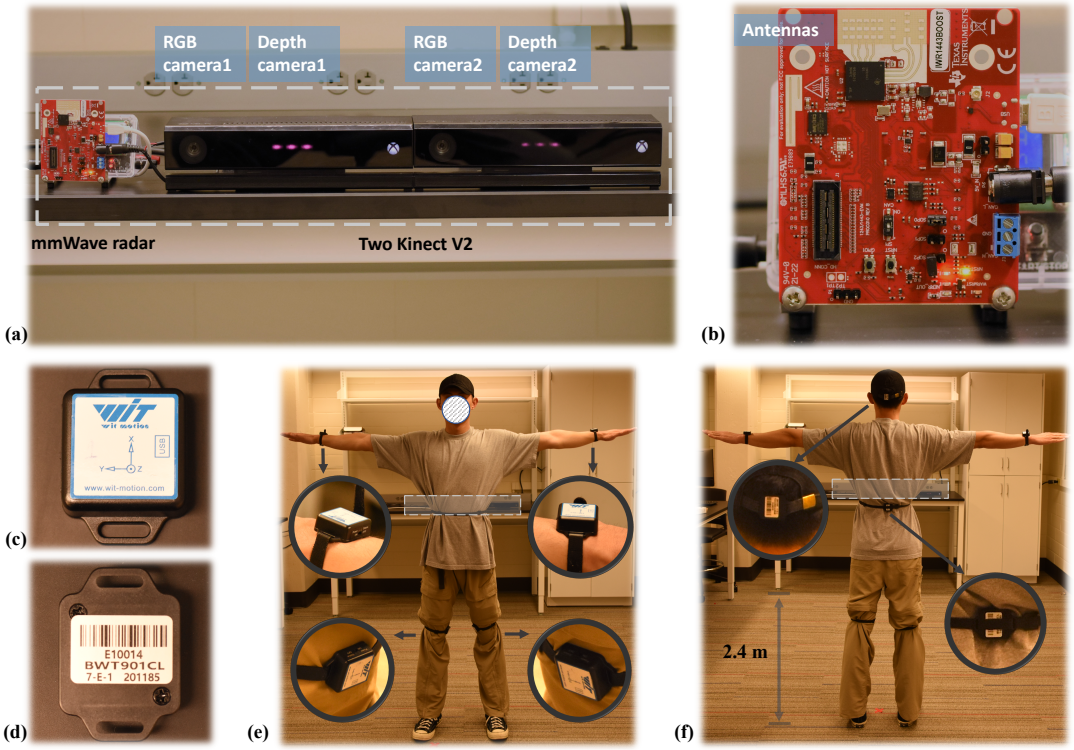


Figure 5.2: Overview of the experimental setup. (a) shows all non-intrusive sensors, including mmWave radar, two RGB, and depth cameras. (b) shows a zoom-in version of the mmWave radar and its antennas. The front and back views of the IMU are shown in (c) and (d), respectively. (e) and (f) show the front and back view of the subject standing as a "T pose" with six IMUs and zoom-in views of IMUs. The gray dash line boxes in (a), (e), and (f) represent the position of non-intrusive sensors.

ject, wearing six IMU sensors, as shown in Figure 5.2. We now describe the data capturing for each modality and the synchronization across modalities.

Point cloud from mmWave radar. A Texas Instruments (TI) IWR1443 Boost mmWave radar [172] is used to obtain the mmWave point cloud. 3D mmWave point cloud is generated by Frequency Modulated Continuous Wave (FMCW) radar using multiple transmit (Tx) and receiver antennas (Rx) configuration [36, 22, 19].

The radar emits a chirp signal, a sinusoid wave whose frequency increases linearly with time. Then the reflected signals are received at the Rx antenna side. The range, velocity and angle resolutions are computed with the received data using *range FFT*, *Doppler FFT*, and *angle estimation* algorithms, respectively. After the constant false alarm rate (CFAR) algorithm eliminates the noise, a point cloud capturing object shape and movement is constructed as

$$P_i = (x_i, y_i, z_i, d_i, I_i), i \in \mathbb{Z}, 1 \leqslant i \leqslant N \tag{5.1}$$

where x_i, y_i, z_i are the spatial coordinates of the point, d_i represents the Doppler velocity, I_i denotes the signal intensity, and $N = 64$ represents the total number of points in a given frame. To further increase the density of the point cloud, we follow [37] to fuse points from three consecutive frames, i.e., increasing the number of points per frame from 64 to 192.

The radar is connected to the host PC through the UART interface. We modify a Matlab Runtime implementation from TI [183] for the data acquisition. The sampling rate is set to 10 Hz since it is sufficient for measuring human movement (the frequency of most voluntary human movements spans from 0.6 to 8 Hz [184]).

RGB and depth frames from RGB-D cameras. Two Microsoft Kinect V2 [121] sensors are used to capture RGB and depth frames. Kinect V2 has a high precision color camera and infra-red camera, generating color and depth frame with a resolution of 1920 $\times$ 1080 and 512 $\times$ 424, respectively. We modified the software from libfreenect2[1] to generate aligned color, depth, and infra-red frames with the global

[1] https://github.com/OpenKinect/libfreenect2

timestamp from two Kinect V2 sensors. We calibrate the two cameras using the Matlab camera calibration toolbox [193]. The center of the RGB camera 1, as shown in Figure 5.2 (a) is selected as the origin of the world coordinate system.

Joints rotations and accelerations from wearable IMUs. Six Wit-motion BWT901CL IMUs [192] are used to capture the rotation and acceleration of the human joints. In our experiments, the IMUs are tightly attached to the left wrist, right wrist, left knee, right knee, head, and pelvis of the subject to capture the complete information about the human body, as shown in Figure 5.2(e) and (f). Each IMU contains a 3-axis accelerometer, 3-axis gyrometer, and 3-axis magnetometer as the sensing unit. The raw output data from the sensors are accelerations, angular velocity, Euler angle, and magnet field values. Based on these values, we extract joint quaternion and 3-axis acceleration following [194, 18] as they fully specify the body pose and movement. The IMUs connect to the host PC via a USB-HID device using the BLE protocol with a sampling rate of 50 Hz (see Table 5.1), ensuring low-latency data transmission with multiple devices.

Sensors synchronization. All sensors are connected to the same host PC, allowing global timestamps from the host attached to each data point from different sensors. We then synchronize all data points using these global timestamps. Since mmWave radar has the lowest sampling rate, we use its timestamp as the basic timestamp. For each timestamp in mmWave radar, we find the timestamp in other sensors with the minimum absolute difference between itself and the mmWave timestamp and align them. The time difference between sensors is less than 5 ms with the proposed time alignment method. Finally, the synchronized data across

all modalities have the same number of data points.

5.2.2 Data Collection, Annotation, and Visualization

In this section, we explain the details of data collection, annotation, and visualization.

Rehabilitation exercises. We consider 12 movements related to rehabilitation exercises covering the entire human body. The first ten rehabilitation movements are modified from [60, 36]. Figure 5.3 shows all movements: (a) left upper limb extension, (b) right upper limb extension, (c) both upper limb extension, (d) left front lunge, (e) right front lunge, (f) squat, (g) left side lunge, right side lunge, (h) left limb extension, and right limb extension. The 11^{th} and 12^{th} movement are stretching and relaxing in free forms (i), and walking in a straight line (j), respectively. These two movements are meant to increase the diversity of the dataset, as the 11^{th} movement is determined by each subject and the 12^{th} movement features a global displacement. The duration of each type of movement is around one minute per subject. To calibrate the IMUs, we require the subject to perform a "T Pose" at the beginning of each recording.

Participant recruitment and consent. To conduct human subject study, we obtained an approval from the IRB at the university. Our participants were recruited locally and all experiments were carried out in a laboratory setting. Before each session, a researcher introduces the research goal, experiment procedure, and potential risk via both verb communication and video tutorials. The participant is free to raise questions before he or she sign the consent form, and is free to withdraw

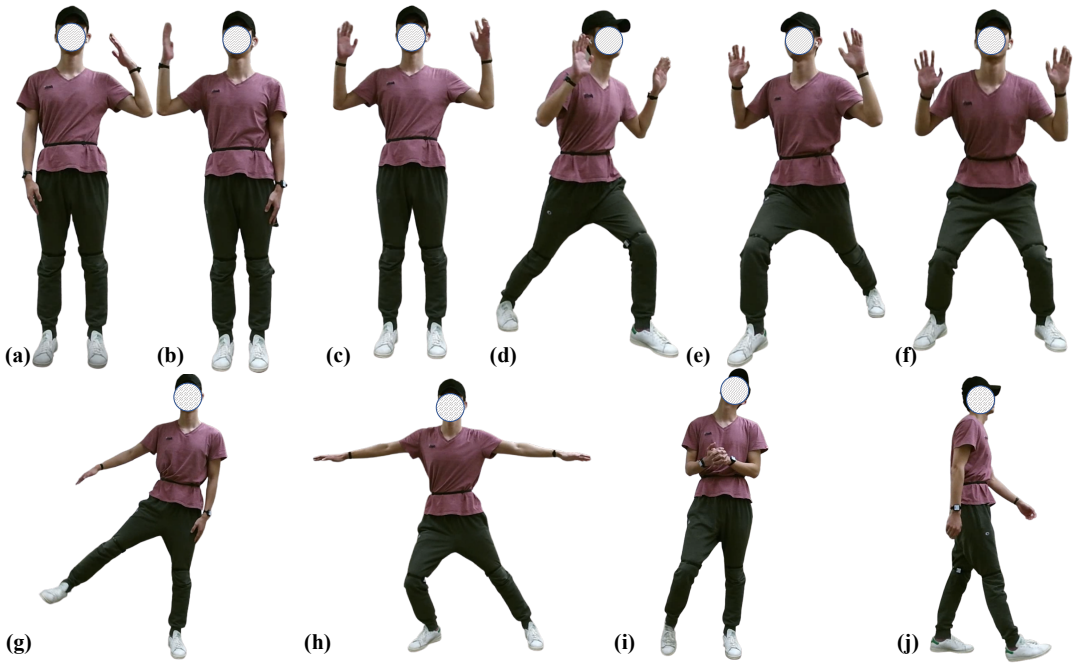


Figure 5.3: Overview of all movements in *mRI*, as described in Section 5.2.2. The mirror movements of (g) and (h) are not shown due to limited space.

from the study at any time. We refer more details to our Ethic statements.

20 healthy participants consented and managed to perform the study. There are 13 males and 7 females, with an average age of 24.1±4.4 and a height of 175.6±9.3 cm.

Obtaining human body pose. We now describe how we derive 2D keypoints and 3D joints given our sensor data. Without using MoCap, our solution is a combination of 2D keypoint detection (body parts), 3D triangulation (joints), and an optimization-based refinement.

- First, we use HRNet [195] (with bounding boxes from Mask RCNN [196]) to detect 2D keypoints of human body parts in all RGB frames from both cameras.

- Next, we triangulate two sets of 2D keypoints captured at the same time yet from different cameras, using camera parameters obtained via camera calibration. The results are a set of 3D body joints (17 in total following COCO format).

- Finally, we refine the 3D joints in each video by solving an optimization problem. Our optimization minimizes 2D reprojection error, imposes equal bone length constraint for all frames, and enforces temporal smoothness of the 3D joints.

Specifically, our refinement step solves the following optimization problem

$$\min_{\{\mathbf{p}_i\}} \sum_{i=1}^{\mathbb{Z}} \left(\left\| P^l \mathbf{p}_i - \mathbf{q}_i^l \right\| + \left\| P^r \mathbf{p}_i - \mathbf{q}_i^r \right\| \right) + \sum_{j}^{bonelist} \left\| \mathbf{B}_j - median(\mathbf{B}) \right\| + \sum_{i=1}^{\mathbb{Z}-1} \left\| \mathbf{p}_{i+1} - \mathbf{p}_i \right\|, \tag{5.2}$$

where $\{\mathbf{p}_i\}$ is the set of 3D joints of size $\mathbb{Z}$, $\mathbf{q}_i^l$ and $\mathbf{q}_i^r$ are the 2D keypoints from the left and right camera, respectively. P^l and P^r are the camera projection matrix for the left and right camera, respectively. $\{\mathbf{B}_j\}$ is a set of bone length defined by connecting a subset of the joints (e.g., wrist to elbow, elbow to shoulder). The first term represents the re-projection errors of the two cameras. The second term enforces equal bone length across all frames in the same video (i.e., the same subject). And the third term imposes temporal smoothness of the 3D joint coordinates. More details, including both quantitative and qualitative results, can be found in the supplement. After the optimization, we re-project the 3D joints to 2D and thus update the 2D keypoints.

Keypoints quality. To validate the reliability of the obtained 3D joints, we report the reprojection error of the derived 3D joints by comparing their 2D projections to human annotated 2D keypoints. Specifically, we randomly sample 50 video frames from our dataset, manually annotate the 2D keypoints for each frame, and calculate the error between the projected 3D joints and the annotated 2D keypoints, following [8]. The mean absolute percentage error (MAPE) is 1.5%, and the percentage of correct keypoints thresholded at 50% of the head segment length (PCKh) is 98.9.

Annotating actions in videos. We also provide annotations of the 12 movements for each video. The multi-media annotation tool ELAN [197] is employed to annotate the videos. For each video sequence, we manually label the start and end timestamp and the category of the 12 different movements.

5.3 Evaluation and Benchmarks

We introduce a standardized evaluation pipeline of using our dataset for 3D human pose estimation and human action detection. We use latest models to benchmark the performance of each modality and discuss their results.

5.3.1 3D Human Pose Estimation

Our main benchmark is 3D HPE. We now describe our experiment protocol, evaluation metrics, and the method we used, followed by the presentation of our results.

Experiment protocol. We consider two settings of data splits. **Setting 1 (S1**

Random Split): A random split of 80% and 20% of all data is used as the training and testing set, respectively. **Setting 2 (S2 Split by Subjects)**: A randomly selected subset (80%, i.e., 16 out of 20) of the subjects is used for training, while the rest are for testing. S1 mimics a case where personalized HPE model is possible, while S2 evaluates across-subject generalization. For each setting, we randomly sample three splits and report the averaged results.

Further, we also define two evaluation protocols based on the design of our movements, as mentioned in Section 5.2.2. **Protocol 1 (P1)** consists of all 12 movements, including stretching and relaxing in free forms and walking. While **Protocol 2 (P2)** only considers the first ten rehabilitation movements. Such protocols help us investigate the robustness of the model in terms of fixed/free form movement.

Evaluation metrics. We adopt Mean Per Joint Position Error (MPJPE) and Procrustes Analysis MPJPE (PA-MPJPE), widely used in human body pose estimation [9], as the main metrics. MPJPE represents the mean Euclidean distance between ground truth and prediction for all joints. MPJPE is calculated after aligning the root joints (the pelvis) of the estimated and ground truth 3D pose. PA-MPJPE is MPJPE after being aligned to the ground truth by the Procrustes method [198], a similarity transformation including rotation, translation, and scaling. We also report additional metrics such as joint angles provided in the supplement.

Methods. We conduct 3D human pose estimation using mmWave, RGB, and IMUs separately using latest methods. Here we briefly introduce the methods considered in our evaluation and refer to our supplement for more implementation details.

- **mmWave**: We use the data processing pipeline and model from [36] that learns a convolutional neural network on the 5D point cloud to regress the 3D joints. The model is trained from scratch on our dataset, and outputs the 3D joints in the global coordinates system.

- **RGB**: We adopt the model from [123], where 2D keypoints from a sequence of frames are "lifted" into 3D joints (in the camera coordinate system) using a convolutional neural network. We use the pre-trained model from [123]. As the pre-trained model outputs a different set of joint, we only evaluate on a subset that intersects with our set of joints.

- **IMUs**: We employ the feature processing method from [18], with a convolutional neural network trained to regress rotations relative to a root joint (e.g., pelvis) using data from IMUs. The model is trained from scratch on our dataset.

Results and discussion. Table 5.2 shows the 3D HPE results for mmWave, RGB, and IMUs. Under **S1** and **P1**, mmWave-based HPE achieves 163 and 94 mm for MPJPE and PA-MPJPE, respectively. The metrics are further reduced to 125 and 74 mm for **P2**. IMU-based HPE obtains MPJPE and PA-MPJPE of 87 and 60 mm, respectively, under **S1** and **P1**. Figure 5.4 shows visualization comparison of estimation across different modalities.

Under **S2**, mmWave-based HPE performs similarly to **S1**, while IMU-based HPE obtains worse results than **S1**. This is because the sensing data from IMU is more fine-grained on the joints while mmWave grasps more information about

Table 5.2: 3D human pose estimation results for mmWave, RGB, and IMUs. We report the mean and standard deviation of joint errors averaged across multiple splits under both our settings (**S1** & **S2**).

		Protocol 1		Protocol 2	
Modality	Setting	MPJPE (mm)↓	PA-MPJPE (mm)↓	MPJPE (mm)↓	PA-MPJPE (mm)↓
mmWave	S1	163.3±9.1	94.1±3.6	125.1±2.4	74.1±1.0
	S2	186.6±23.8	97.3±7.8	126.6±11.3	75.0±7.1
RGB	S1	116.9±0.1	66.8±0.2	115.0±0.1	64.4±0.1
	S2	120.1±3.7	67.5±1.9	118.4±3.8	64.7±1.4
IMUs	S1	**80.2**±12.6	**51.9**±1.9	**40.9**±1.0	**28.4**±0.9
	S2	147.4±18.4	74.5±5.9	94.3±13.8	54.0±4.9

body trunk, which is not too subject-specific. As a result, the IMU-based model is more sensitive to different subjects. We can observe that for all modalities **S2** yields higher standard deviations than **S1** since the difference between subjects is much more significant than random split, between train and test set. Similarly, **P1** yields higher standard deviations than **P2** since all movements in **P2** are fixed positions, which makes the model learning the keypoints distribution easier.

RGB-based HPE achieve 116 and 66 mm MPJPE and PA-MPJPE for **P1** under **S1**. Both data-split yield similar results. To compare, the same model achieves 36 mm PA-MPJPE on Human3.6M dataset. However, the model is trained and evaluated on Human3.6M while it is only evaluated on *mRI* without any fine-tuning. We leave fine-tuning the model on *mRI* as future work. In summary, all modalities perform reasonably well on our dataset.

Result visualization. We further visualize sample results of 3D pose estimation from different modalities in Figure 5.4.

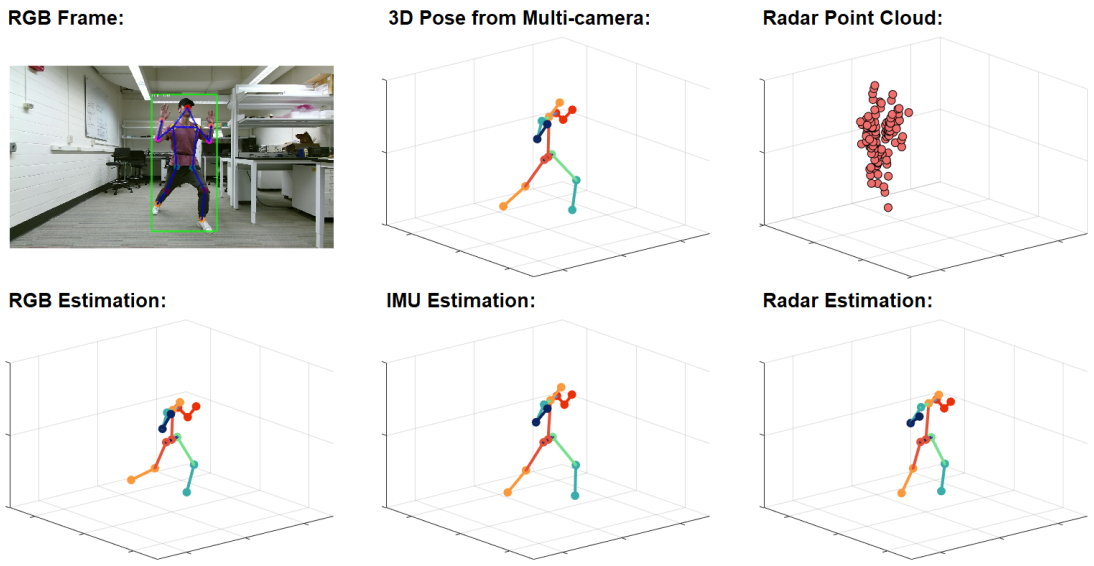


Figure 5.4: Visualization of sample pose data and results during left front lunge.

5.3.2 Skeleton-based Action Detection

Moving forward, we explore using the estimated 3D joints for temporal action detection in untrimmed videos — the simultaneous localization and recognition of action instance in time. Specifically, given an input untrimmed video, temporal action localization seeks to predict a set of action instances with varying size. Each instance is defined by its onset, offset, and action labels.

Experiment protocol. We consider the more challenging setting **S2**, where a model is tasked to detect actions performed by subjects not presented in the training set. Here each movement type defines one action category. Similar to our HPE experiments, we evaluate on both **P1** (12 categories) and **P2** (10 categories focusing on rehabilitation exercises). Importantly, we consider using individual modalities and all combinations of these modalities (e.g., RGB+IMU or RGB+mmWave). To

combine multiple modalities, 3D joint data from each modality at every time step is concatenated, and the resulting sequence is fed into the model.

Evaluation metrics. Following prior work [128], we report the mean average precision (mAP) at multiple temporal intersection over union (tIoU) thresholds ([0.5:0.05:0.95]). tIoU is defined as the intersection over union between two temporal windows, i.e., the 1D Jaccard index. Given a tIoU threshold (e.g., 0.75), mAP computes the mean of average prevision across all action categories. An average mAP is also reported by averaging the mAP scores across all tIoUs.

Method. We make use of a latest method — ActionFormer [199] for temporal action detection. ActionFormer develops a Transformer based model and achieves state-of-the-art results across action detection benchmarks. Specifically, we feed the model with a sequence of estimated 3D poses from different modalities at a sampling rate of 2 Hz, and train the model from scratch on our dataset.

Results and discussion. Table 5.3 summarizes the results from three modalities and their combinations averaged across all splits. Overall, all modalities perform fairly well, with mAP scores around 90%. Under **P1**, IMUs data have the best results with 93.4% mAP, and outperform the RGB frames and radar signals by 1.9% and 6.4%, respectively. Under **P2**, both IMUs data and RGB frames perform equally well with improved mAP (around 95%). The RGB frames achieve a major improvement when evaluated under **P2**. It is interesting to cross reference the results of HPE and action detection. While RGB frames have lower joint errors under **S2** and **P1**, they have slightly worse results on action detection. On the other hand, IMUs data perform consistently well on action detection in **P1** and **P2**.

Table 5.3: Action detection results with mmWave (**W**), RGB (**R**), IMUs (**I**), and their combinations. We report the mean and standard deviation of *m*AP averaged across 3 splits under our setting 2 (**S2**).

	*m*AP↑							
	Protocol 1				Protocol 2			
Modality	tIoU=0.50	tIoU=0.75	tIoU=0.95	average	tIoU=0.50	tIoU=0.75	tIoU=0.95	average
mmWave (**W**)	98.22±3.08	97.59±4.17	29.02±6.31	87.04±4.89	99.00±1.73	97.21±2.41	31.11±15.34	87.55±3.61
RGB (**R**)	**100.00**±0.00	99.14±0.75	44.80±10.55	91.56±2.08	**100.00**±0.00	**100.00**±0.00	59.87±8.12	**95.07**±1.46
IMUs (**I**)	**100.00**±0.00	**100.00**±0.00	**53.55**±12.39	**93.46**±2.30	**100.00**±0.00	**100.00**±0.00	**60.13**±6.82	94.89±1.39
W+R	**100.00**±0.00	**100.00**±0.00	55.71±11.20	94.17±1.58	**100.00**±0.00	**100.00**±0.00	59.89±15.18	95.09±2.14
W+I	**100.00**±0.00	**100.00**±0.00	56.53±12.23	94.38±1.70	**100.00**±0.00	**100.00**±0.00	62.42±5.65	95.26±1.08
I+R	**100.00**±0.00	99.61±0.67	60.10±11.97	94.54±1.45	**100.00**±0.00	**100.00**±0.00	61.10±8.46	94.80±1.28
W+R+I	**100.00**±0.00	**100.00**±0.00	**60.62**±8.42	**94.88**±1.75	**100.00**±0.00	**100.00**±0.00	**66.16**±10.89	**95.83**±1.50

Further combining the modalities results in a noticeable performance boost. It is probably not surprising that using all three modalities yields the best results, outperforming the best single modality by 1.4% (**P1**) and 0.8% (**P2**) in average mAP and with most gains in mAP under tIoU=0.95 (+7.1% for **P1** and +6.0% for **P2**). Fusing any of the two modalities leads to improved performance than the best of the constituting modality, except the combination of IMUs+RGB under **P2**. These results demonstrate a first step towards multi-modal learning with our dataset.

mmWave radar is less invasive than IMU sensors and offers better privacy than RGB cameras. While in its infancy for human sensing, this modality presents an emerging solution for home-based health monitoring. Part of our goal in this paper is to explore mmWave radar for human sensing by comparing its performance to other common modalities. The results indicate that mmWave radar leads to compelling performance for both human pose estimation and action localization. While its results are worse than those with RGB cameras or IMU sensors, mmWave radar might still be preferred for privacy-sensitive applications.

5.4 Ethics Statement

The human subject studies reported in this paper was reviewed and approved by the IRB committee at the University of Wisconsin-Madison. Each participant was informed about the research project and signed the consent form. The data has been de-identified with facial information blurred in all videos and participant ID anonymized, and made publicly available to facilitate future research.

To the best of the authors' knowledge, this work does not disadvantage any person directly. The authors do acknowledge that any pose estimation and activity recognition method can potentially be used with malicious intent, such as tracking user movements. If the human pose estimation/human activity understanding algorithms are directly used to make decisions for patients, potential failures in the classification would affect the users' quality of life. Therefore, the data and insights on patient activity must be verified by health professionals before making any decisions.

6 FUSE:FAST AND SCALABLE HUMAN POSE ESTIMATION USING MMWAVE POINT CLOUD

6.1 Background, Motivation and Contributions

Human pose estimation can be performed by processing image, video, lidar (light detection and ranging), or mmWave radar data. The most common input type is RGB image and video frames since they offer accurate real-world representations using true color. However, the RGB frame quality depends heavily on the environmental setting, such as light condition and visibility. The lidar point cloud is a powerful alternative that overcomes these challenges. However, it has high cost and significant processing requirements, making them unsuitable for indoor applications such as rehabilitation. In contrast, mmWave radar can generate high-resolution 3D point clouds while maintaining low-cost and power advantages [200].

Using mmWave point cloud for human pose estimation faces two major challenges. First, mmWave point cloud is significantly sparser and less informative compared to video and lidar point cloud data. For example, humans can easily recognize the object and its details from video and lidar point cloud, while it is almost impossible for people to interpret the mmWave point cloud representation accurately. Second, the amount of labeled mmWave data lags severely behind video and lidar point cloud data. However, ML algorithms need a large amount of data to learn the generalization to new scenarios. For example, human pose estimation techniques must easily generalize to new users not included during training. *Hence,*

there is a critical need for approaches that can perform well with fewer data points to harness the potential of mmWave data. This capability can enable home-based applications with significantly lower computation requirements since fewer data samples and training efforts are needed.

Meta-learning has recently gained momentum because it can help ML models adapt to unseen scenarios faster with a few training iterations. It focuses on learning a strategy that generalizes to related yet unseen tasks from similar task distributions [55, 67]. It is first trained with a batch of tasks and learning rules designed to facilitate learning new tasks using only a few training iterations. In this way, the model employs the parameters sensitive to new samples, expediting generalization to new tasks. The meta-learning concept fits the mmWave point cloud context very well since the amount of labeled training mmWave data is substantially smaller than video and lidar point cloud data. Hence, it can be a crucial enabler for mmWave radar point cloud-based human pose estimation with a few data samples and training iterations.

This chapter presents FUSE, a fast and scalable human pose estimation technique using mmWave point cloud. FUSE estimates the coordinates of 19 joints on the human body using mmWave point cloud as the input. Its first component is a novel point cloud pre-processing method that fuses sparse frames to construct multi-frame data representations. Multi-frame representation enriches the sparse point cloud representation, reducing the mean absolute error (MAE) on the human pose estimation task by 34%, as demonstrated in Section 4.2. This method can easily be integrated into existing mmWave radar-based techniques as a pre-processing step

to boost their performance without affecting other parts. The second component is a meta-learning framework that enables FUSE to adapt to the unseen data within a few epochs. Experimental results show that FUSE can converge to the optimal state in just five epochs, which is 4× faster than prior approaches.

In summary, the major contributions of this chapter are as follows:

- An effective point cloud pre-processing method that enhances mmWave point cloud representation by frame fusion,
- A meta-learning framework that significantly enhances the ability to generalize and adapt to unseen scenarios,
- Experimental evaluations that show 7 cm MAE in estimating joint coordinates with 4× fewer training iterations than prior approaches.

6.2 Overview of FUSE

This section first provides a brief background on mmWave human pose estimation. Then, it presents the proposed FUSE framework that consists of point cloud pre-processing and meta-learning steps, as described in Figure 6.1.

6.2.1 Shortcomings of state-of-the-art techniques

Prior mmWave pose estimation techniques focus on improving ML model accuracy for an existing set of users or movements. However, two critical aspects are ignored in these studies. First, they do not consider improving the sparse point cloud

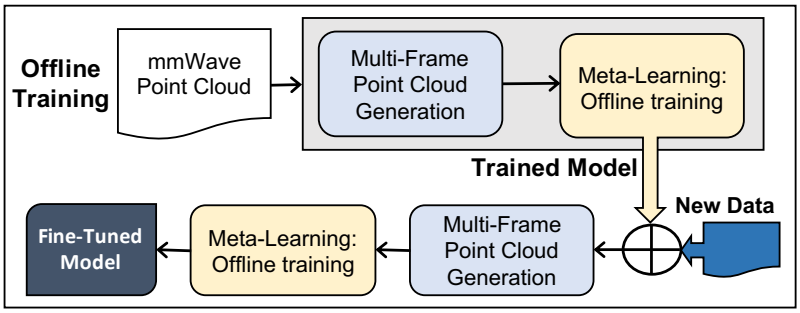


Figure 6.1: Overview of the proposed FUSE framework

representation. For example, Zhao et al. [19] designed large antenna arrays to enrich the radar data. Similarly, Sengupta et al. [22] use two mmWave radars and sum up their information. However, increasing the area or the number of antennas does not address the fundamental challenge. Instead, it increases the cost and makes deployment harder. Second, these studies rely on offline model training and testing. This choice is not practical since ML models need to quickly adapt to unseen scenarios in different application scenarios such as autonomous driving and rehabilitation. Thus, it is crucial to have an initial model that can fastly converge with any new data samples. To address these challenges, we propose a framework that enhances the point cloud representation and adapts to the unseen scenarios faster with a few training iterations.

6.2.2 Multi-Frame Point Cloud Representation

As the sole data source to pose estimation, the point cloud must contain sufficient information to enable CNNs to extract the features, thus predicting accurate joint coordinates. However, current mmWave point cloud solutions only offer up to

hundreds of points for one frame due to the limited number of antennas on the commercial mmWave radar [21]. For illustration, Figure 6.2(a) shows a video frame of a subject performing squat movement. With 512X424 frames (217K pixels), this frame can also be easily interpreted by humans. In strong contrast, the corresponding mmWave point cloud has only 64 3D points (192 data points) in a frame, i.e., almost 1000x fewer data points, as shown in Figure 6.2(b). Consequently, it is harder for ML algorithms to extract information from this representation.

The redundancy in video frames is often eliminated by using residual frames (i.e., differences between consecutive frames), which emphasize the changes due to motion [201]. Indeed, Figure 6.2(c) illustrates that residual frames preserve the relevant information, facilitating feature extraction by ML algorithm. mmWave radar faces precisely the opposite problem: we must enrich a severely sparse representation as opposed to reducing redundancy. Therefore, *for the first time in literature, we propose to fuse multiple sparse point cloud frames to synthesize a richer representation*. As illustrated by Figure 6.2(d), the proposed multi-frame approach significantly improve the interpretability compared to a single mmWave point cloud frame. Unlike a single-frame point cloud frame in Figure 6.2(b), multi-frame point cloud representation accurately captures the shape in the upper body. For example, we can see there are more points around the main body and arm area.

Let $T_S > 0$ be the sampling period of the target mmWave radar (in this work, $T_S = 100$ ms). The k^{th} frame $f[k]$ in the point cloud contains the points collected during time interval $\left[kT_s, (k+1)T_s\right)$ for $k \in \mathbb{Z}^+$. Hence, we can express The k^{th}

frame $f[k]$ as:

$$f[k] = \Big[P_1[k], P_2[k], \ldots, P_N[k]\Big] \; \forall k \in \mathbb{Z}^+ \tag{6.1}$$

where $P_i[k]$ is the i^{th} point in frame k for $1 \leqslant i \leqslant N$. Then, we fuse M consecutive frames by concatenating them as follows:

$$F[k] = \Big\{f[k-M], f[k-(M-1)], \ldots, f[k], \ldots, f[k+M-1], f[k+M]\Big\} \tag{6.2}$$

where M is a meta parameter that controls the number of fused frames. For instance, $M = 1$ implies fusing three frames as $F[k] = \{f[k-1], f[k], f[k+1]\}$. This simple yet powerful idea significantly enhances the information content as interpretability even with $M = 1$, as shown in Figure 6.2(d). Quantitative analysis in Section 6.3.2 demonstrates that our proposal can significantly improve the results of prior studies that employ mmWave point cloud data.

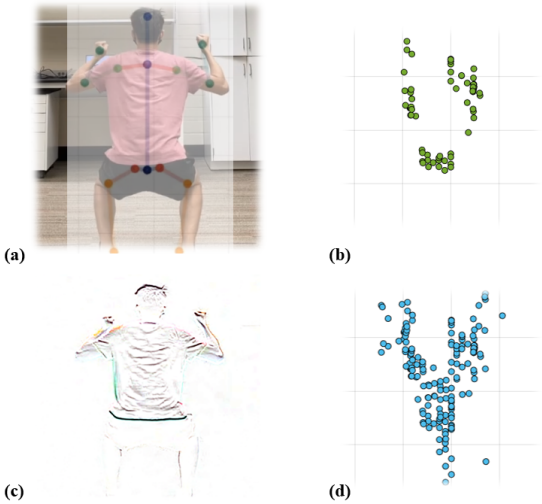


Figure 6.2: Visualization comparison of (a) one RGB frame, (b) a single-frame point cloud, (c) RGB residual frame, (d) proposed multi-frame point cloud.

6.2.3 Meta-Learning for Human Pose Estimation

Meta-learning, also known as "learning to learn", aims at training a model on a variety of tasks such that it can solve new learning tasks using only "few training samples". In our context, consider that a learning model, such as a CNN, is trained using an initial set of users and prescribed movements. If new users or movements are introduced, traditional techniques would either need to train models from scratch or adapt the model using incremental learning [33, 202]. Learning from scratch is highly inefficient, while the latter approach is an after-thought. *In strong contrast, we construct the initial model by explicitly maximizing its ability to adapt to new users and movements using only a few training samples.* This capability is achieved by choosing model parameters sensitive to new samples, thereby ensuring that the gradient-based online learning rule can rapidly progress with new data. In this way, FUSE adapts to unseen scenarios quickly and estimates joint coordinates accurately after fine-tuning with a few training iterations.

mmWave meta-learning setup

This section defines the parameters and terminology used for meta-learning.

Definition 6.1 (*Training data,* $\mathcal{D}_{\texttt{train}}$)**.** *The training data is the set of all fused frames* $F[k], k \geqslant 1$ *constructed using the point cloud frames as defined by Equation 6.2, i.e.,*

$$\mathcal{D}_{\texttt{train}} = \bigcup_{k} F[k] \tag{6.3}$$

Instead of directly using individual samples in $\mathcal{D}_{\texttt{train}}$, meta-learning generates

tasks and uses them for learning as described next.

Definition 6.2 (*Task*, $\mathcal{T}$)**.** *We define task $\mathcal{T}$ a set of fused frames uniformly sampled from the training data, i.e.,* $\mathcal{T} \sim \mathcal{D}_{train}$.

Next, we present the proposed offline meta-training and online fine-tuning techniques.

Offline Meta-Training

After constructing the training data $\mathcal{D}_{train}$, we train the initial model using meta-learning using Algorithm 1. The algorithm starts with randomly initializing the model parameters θ. Then, it starts performing the meta-training iterations (lines 2–12). Each iteration i starts with uniformly sampling a batch of tasks from the training data: $\mathcal{T}_i \sim \mathcal{D}_{train}$ (line 3). Then, the inner loop (lines 4–10) operates on these tasks. We first randomly choose a subset of tasks in the batch, and denote them as support tasks $\mathcal{T}_i^{sup}$. The support tasks are used to update the intermediate model parameters in each iteration using gradient descent:

$$\theta_i' = \theta - \alpha \nabla_\theta L_{\mathcal{T}_i^{sup}}(g_\theta) \tag{6.4}$$

where α is the $sample-level$ learning rate. Our implementation uses the mean absolute error in joint coordinates (i.e., L1 distance) as the loss function, but other functions such as L2 can also be used (line 7). Unlike traditional supervised learning techniques, meta-learning samples next a set of query tasks $\mathcal{T}_i^{qry}$ (line 8) similar to

the selection of the support tasks. Then, the loss function for the query tasks are found for the model updated on line 7 (line 9).

After going over all tasks in $\mathcal{T}_i$, the model updates its initial parameters θ using the summation of the loss on each query task T_i^{qry}, as shown in the following equation:

$$\theta = \theta - \beta \nabla_\theta \sum_{\mathcal{T}_i^{qry}} L_{\mathcal{T}_i^{qry}}(g_{\theta_i'}) \tag{6.5}$$

Note that the initial parameters θ are **only** updated once after all T_i are done for a meta-training iteration. Also, in strong contrast to traditional transfer learning, these parameters are updated using the *intermediate parameters obtained from* $\mathcal{T}^{sup}$ but the *loss evaluated from* $\mathcal{T}^{qry}$. In transfer learning, the initial parameters θ are updated using the intermediate parameters and the loss obtained from the same tasks. This crucial difference is why meta-learning can find the most sensitive parameters to the new data samples.

Online Fine-Tuning Phase

So far, we presented the construction of the initial meta-learned model using the available training data. Suppose a new user or movement, which is absent from the training data, is introduced in the field, i.e., after the trained model is deployed. Our goal is to use few training samples denoted by $\mathcal{D}_{test}$ to update the initial model. We emphasize that the cardinality of the test data is $|\mathcal{D}_{test}| << |\mathcal{D}_{train}|$. Section 6.3.3 validates this claim and shows that the size of the required test data is $4\times$ smaller than that required by supervised techniques.

Algorithm 1: Meta-training for mmWave point cloud

Input: $\mathcal{D}_{train}$, g_θ (untrained model), β (meta-learning rate)
Output: ML model that computes human joint coordinates using mmWave point cloud.

Initialize the parameters θ of the ML model g_θ
for *each meta-training iteration* **do**
  Sample a batch of tasks: $\mathcal{T}_i \sim \mathcal{D}_{train}$
  for *all* $\mathcal{T}_i$ *do* **do**
    Sample support tasks from $\mathcal{T}_i$: $\mathcal{T}_i^{sup} \subset \mathcal{T}_i$
    Compute the gradient $\nabla_\theta L_{\mathcal{T}_i^{sup}}(g_\theta)$
    Update parameters $\theta_i' = \theta - \alpha \nabla_\theta L_{\mathcal{T}_i^{sup}}(g_\theta)$
    Sample query tasks $\mathcal{T}_i^{qry} \subset \mathcal{T}_i$
    Evaluate $L_{\mathcal{T}_i^{qry}}(g_{\theta_i'})$ using parameters θ_i'
  end
  Update the initial parameters $\theta = \theta - \beta \nabla_\theta \sum_{\mathcal{T}_i^{qry}} L_{\mathcal{T}_i^{qry}}(g_{\theta_i'})$
end

We use $\mathcal{D}_{test}$ to perform the fine-tuning and testing to evaluate our meta-trained model's performance. The model takes a part of $\mathcal{D}_{test}$ to perform forward pass and back-propagation to fine-tune. Then, we use the other part of $\mathcal{D}_{test}$ only to perform inference and evaluate the model. The exact procedure to split the data is described with the implementation in Section 6.3.1. Ideally, only fine-tuning for a few iterations should be enough since the model learns the generalization of the point cloud. In summary, the fine-tuning phase does not require any extra steps, facilitating online usage.

6.3 Experimental Evaluations

6.3.1 Experimental Setup and Baseline Model

Human Pose Estimation Data: We employ an open-source mm-Wave point cloud dataset (MARS [36]) to evaluate the proposed FUSE framework. The dataset consists of 40,083 labeled frames (defined in Equation 6.1) collected using TI IWR1443 Boost mmWave radar [200]. The frames correspond to 10 distinct rehabilitation movements performed by four human subjects in front of the mmWave radar and Microsoft Kinect V2 sensor. The reference coordinates of 19 joints are found using Kinect V2 and added as labels to the mmWave data at a 10 Hz sampling rate. Then, each movement data is individually split into 60% training, 20% validation, and 20% test sets.

Baseline ML model: We implement the CNN trained with the MARS dataset [36] as the baseline model to ensure a fair comparison. It has two convolution layers with Rectified Linear Unit (ReLU) activations, followed by two FC layers, with a total model of 1,095,115 parameters. The number of neurons of two FC layers is 512 and 57, respectively. Here, the output values of the final 57 neurons represent 19 human joints coordinates values in x, y, z-axes. The proposed CNN trained using the FUSE framework has the same dimensions and model size for a fair comparison.

Implementation details: We implemented all baseline and meta-learning approaches using PyTorch 1.8.1 [203]. The training and testing are performed on a Nvidia GeForce RTX 3090 graphic card with 24GB of memory. The meta-learning approach is based on MAML-PyTorch implementation [204]. Our results can be

reproduced using the following hyper-parameter values: 20,000 meta-training iterations, 32 tasks sampled for each iteration, *sample-level* learning rate $\alpha = 0.1$, *task-level* meta-learning rate $\beta = 0.001$. During meta-training, each support task and query task $\mathcal{T}_i$ samples 1,000 frames from $\mathcal{D}_{train}$ randomly. We use 200 frames from $\mathcal{D}_{test}$ to fine-tune and the all rest frames from $\mathcal{D}_{test}$ to evaluate the performance. Finally, MAE (i.e. the L1 loss) loss function and Adam optimizer [205] are employed for calculating the loss and updating the gradients.

6.3.2 Multi-Frame Fusion of Point Cloud Data

Fusing multiple frames enriches the information content of the mmWave point. To study the impact of frame the fusion alone, this section uses the baseline CNN architecture, the default 60% - 20% - 20% data split, and training parameters (128 batch size and 150 epochs). We conduct experiments on three settings: single-frame (baseline), fuse three frames, and fuse five frames.

Table 6.1 summarizes the average MAE in predicting the joint coordinates with and without multi-frame fusion. Without changing any other parameters, fusing three frames *consistently decreases the average MAE* along $x-$, $y-$, and $z-$axis, and average MAE reduction from 5.5 cm to 3.6 cm. Fusing more frames does not continuously improve the accuracy since redundancy is introduced. Specifically, we observe that fusing three frames outperforms a single frame by 1.9 cm margin (34%). This improvement is impressive since achieving similar savings without frame fusion would require a significant increase in the model complexity.

These controlled experiments show that fusing multiple frames enhances the

point cloud representation, thus improving the performance of human pose estimation. Hence, it can boost the performance of existing mmWave radar techniques without affecting the ML models they employ. In the rest, we fuse three frames since it leads to significant savings with negligible overhead.

Table 6.1: MAE of the baseline model under different frame fusion settings.

	X (cm)	Y (cm)	Z (cm)	Average (cm)
Single-frame	6.4	3.6	6.5	5.5
Fuse 3 Frames	4.2	2.5	4.4	3.6
Fuse 5 Frames	6.9	4.1	5.5	5.5

6.3.3 Convergence Time and Accuracy Evaluation

Data splitting

To examine the ability of FUSE to adapt to new scenarios, this experiment splits the dataset to capture the worst-case scenario. The training and validation sets *exclude all data* from one particular movement (*"right limb extension"*) and one of the users (user 4). With this split, the test data ($\mathcal{D}_{\texttt{test}}$) seen only during fine-tuning has only 749 frames, justifying our claim about few samples available online. In contrast, the training data ($\mathcal{D}_{\texttt{train}}$) consists of 29,225 frames from the remaining nine movements and users. A more comprehensive leave-one-out experiment is left for future work due to space and execution time considerations.

Quantitative results

Fine-tuning is a commonly used method in transfer learning [202] to test a model's ability to adapt to new data samples. It fine-tunes all layers or part of a pre-trained model with new data samples [202]. We conduct experiments for both cases: fine-tuning all layers and only the last FC layer with its activation.

Fine-tune *all layers*: Figure 6.3 shows the MAE comparison between baseline and FUSE model fine-tuned for all layers. The baseline model achieves a remarkable MAE of 6.7 cm after the initial training with the original data available offline, as shown in Figure 6.3(a). In contrast, the proposed FUSE model starts with 12.4 cm MAE since it is optimized for generalization rather than fitting to known cases. Indeed, FUSE achieves almost 6.0 cm MAE after only 5 fine-tuning epochs with the new data. *We emphasize that both the baseline model and FUSE are fine-tuned using the new data, which is not included in the initial training.* The extra data improves FUSE's performance even on the original training dataset. After fine-tuning, the MAE of FUSE stabilizes at about 9.4 cm for the original data (Figure 6.3(a)) and 4.0 cm for the new user data (Figure 6.3(b)).

Figure 6.3(b) also shows that the baseline model can be effectively fine-tuned for the new data. In strong contrast to FUSE, the improved performance comes at a steep penalty for the original data, as shown by the solid line in Figure 6.3(b). As the baseline model adapts to new data, it forgets the original one, implying that it tends to overfit rather than learn the trends.

In summary, FUSE is able to achieve 6.0 cm MAE, ~3 cm lower than the baseline with only 5 epochs fine-tuning. With 26 epochs, as a red circle shown in

Figure 6.3(b), baseline approach is able to achieve 4.6 cm, which is comparable to 4.3 cm for FUSE model. However, it is at the expense of forgetting the original data. The MAE for original data reaches 10.6 cm, as summarized in Table 6.2 (columns labeled "All layers"), since the baseline model do not learn the generalization. After that, the baseline approach just keep memorizing the new data and forgetting the original data.

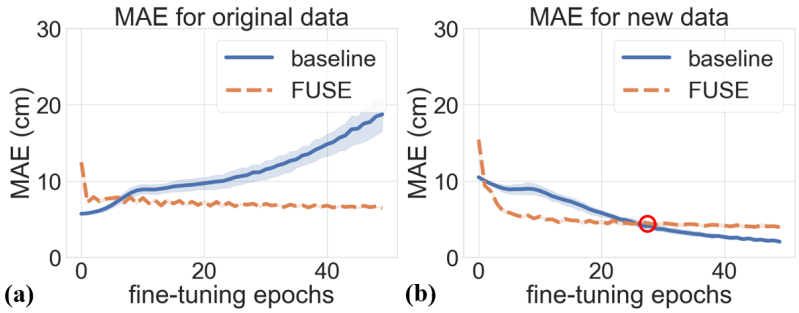


Figure 6.3: MAE comparison between baseline and FUSE model for fine-tuning all layers.

Table 6.2: MAE comparison between baseline and FUSE model. Results of both fine-tuning all layers and only the last layer are presented in the table..

		All layers		Last layer	
		baseline	FUSE	baseline	FUSE
5 epochs	Original	6.4	7.6	6.5	9.0
	New	9.0	6.0	9.6	8.3
Intersection	Original	10.6	6.6	7.2	8.2
	New	4.6	4.3	7.1	7.0
50 epochs	Original	18.7	6.4	31.0	7.8
	New	2.0	3.9	3.9	6.0

Fine-tune *the last layer*: Figure 6.4 shows the MAE comparison between the baseline and FUSE model when only the last fully connected layer is fine-tuned. It shows a very similar pattern with fine-tuning with all layers, as summarized in Table 6.2 (columns labeled "last layer"). With only 5 epochs of fine-tuning, the FUSE model achieves 8.3 cm MAE, 1.3 cm lower than the baseline. With 16 epochs, as shown with a red circle in Figure 6.4(b), the baseline approach can achieve 7.1 cm, which is comparable to 7.0 cm for FUSE model. However, it is at the expense of forgetting the original data, as the MAE for original data reaches 7.2 cm. The baseline approach memorizes the new data and forgets the original data to adapt to new scenarios. Compared to fine-tuning all layers, fine-tuning only the last layer yields a higher error for new data and significantly worse forgetting trend. The results show that fine-tuning with all layers can help the model adapt to unseen scenarios since FUSE learns the generalization.

In summary, FUSE enhances the mmWave point cloud representation, thus improving the human pose estimation performance significantly. In addition, FUSE adapts to the unseen data within five epochs. This is 4× faster than the baseline

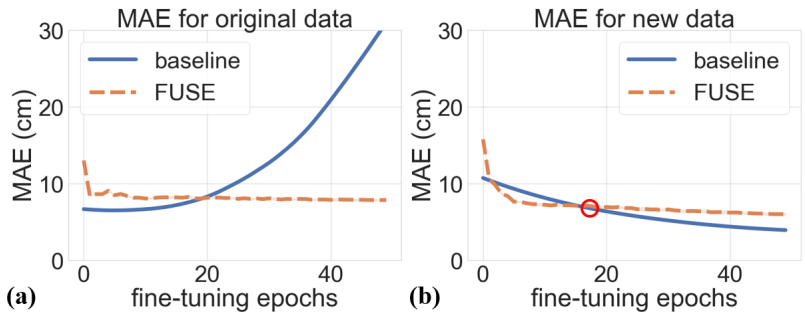


Figure 6.4: MAE comparison between baseline and FUSE model for fine-tuning the last layer.

approach that needs at least 20 epochs to achieve the same performance, at the expense of forgetting the original data.

7 TRANSFER LEARNING FOR HUMAN ACTIVITY RECOGNITION USING REPRESENTATIONAL ANALYSIS OF NEURAL NETWORKS

7.1 Background, Motivation and Contributions

Most HAR techniques start with collecting sensor data from users available at design time [41]. This data is used to train a classifier for the activities of interest. Then, the trained classifier is used by new users, whose data is not available for training. This approach assumes that the HAR classifiers can be transferred across different user sets. However, this assumption may not hold in general as activity patterns can change with age, gender, and physical condition. Hence, lack of model personalization can limit the accuracy [206]. For instance, Figure 7.1 shows the stretch sensor data during walking for four users in our experimental dataset. There is a significant variation both in the range of sensor values and the data patterns. Furthermore, the activity patterns may vary over time even for a given user due to progression of symptoms, injury, or other physical changes. These variations can significantly reduce the recognition accuracy for new users as we demonstrate in this paper. Therefore, *classifiers designed offline must adapt to changing data patterns of new and existing users to achieve high classification accuracy*.

Training classifiers from scratch for new users is expensive due to data storage and computational requirements. It is challenging especially for low-power wearable and mobile devices, which are the most common platforms for HAR [114, 41]. Moreover, training from scratch for a particular user loses generalization capability

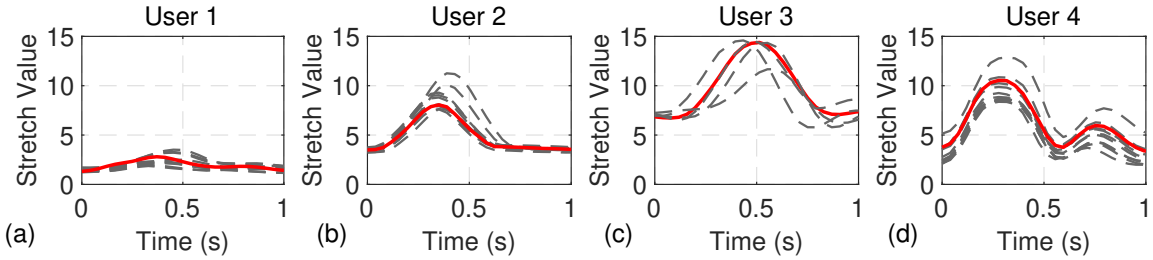


Figure 7.1: Comparison of stretch sensor [1] data of four users for a single step during walk. There is a significant change in both the range of values and data pattern. The grey dashed lines show different instances of the same activity, while the red line shows a *representative* activity window for each user.

and robustness due to overfitting [89]. In contrast, transfer learning with a common feature set can carry over generalization capabilities from offline stage and fine-tune user-specific features after deployment to improve training efficiency.

This chapter first demonstrates that HAR classifiers designed offline cannot be transferred as a whole to an arbitrary set of users. Then, it presents the first systematic study to determine how to transfer the offline knowledge and adapt HAR classifiers to individual users. We show not only the feasibility of transferring the offline classifier, but also determine which parts can be transferred to minimize the *time and energy* required for customization. Furthermore, we demonstrate these benefits on a mobile hardware board using five datasets. This study is performed using convolutional neural networks (CNNs) due to their ability to produce broadly applicable features from raw input data. We use canonical correlation analysis (CCA) [89] to evaluate the distance between the layers of CNNs trained with different sets of users. Our analysis leads to a theoretically grounded and practical HAR classifier framework validated on hardware. It achieves high accuracy and significant savings in training time by fine-tuning the deeper layers that differentiate

users while transferring the earlier layers.

The proposed approach is validated using five public datasets [27, 207, 208, 209, 210]. We first divide the users in each dataset into multiple clusters to evaluate the effect of transfer learning for unseen user clusters. In a challenging scenario, the clusters have to be as separated as possible such that the classifier can only learn the patterns from the users in the training set. For instance, the four users in Figure 7.1 belong to different clusters. To this end, we generate user clusters both randomly (for average-case) and using k-means clustering [211] (for the most challenging scenario). Next, a CNN classifier is trained for each user cluster. After analyzing the similarity between the CNNs trained with different user clusters, the weights are transferred between user clusters and the dissimilar layers are fine-tuned. Extensive evaluations for the w-HAR dataset show that in the worst case, transferring weights and fine-tuning the last layer achieves accuracy that is same as the accuracy obtained by training from scratch while reducing the computation during training by 66%. Finally, we implement the proposed approach on the Nvidia Jetson Xavier-NX board [174]. Our experiments show up to 68% energy and 43% power consumption savings. These results demonstrate the practicality of the proposed technique on low-power edge devices.

In summary, the major contributions of this chapter are:

- An empirical demonstration which shows that HAR classifiers designed offline cannot be transferred to an arbitrary set of users,
- A systematic analytical study that reveals that deeper (typically last two) layers of HAR classifiers capture user-specific information, while the first

three to four layers provide general features,

- Extensive experimental evaluations using five datasets that show up to 43% and on average 14% higher accuracy compared to the accuracy without using transfer learning for new users.
- Hardware experimental evaluations that demonstrate up to 68% energy and 43% power consumption savings.

7.2 Underlying HAR framework and datasets

Most HAR approaches start with data from wearable inertial sensors or a smartphone to record the data when the user is performing the activities of interest.[1] After collecting the sensor data, the next step is to pre-process and segment the sensor data for feature generation. The most common approach in literature for segmentation is to divide the data into one to ten-second windows with a 50% overlap between consecutive windows [207, 213, 209]. Then, the data within each window is processed to generate the features for the classifier. A variety of classifiers, such as neural networks, decision trees (DT), random forest (RF), support vector machine (SVM), and k-nearest neighbors (KNN), have been used for HAR. Since previous approaches use different datasets and activities for evaluation, a direct comparison to our approach is not possible. For a fair comparison, we use the five datasets in this paper on commonly used HAR algorithms. Table 7.1 shows

[1]Video cameras are also used for HAR [212], but we focus on HAR using wearable sensors and smartphones.

the accuracy of these approaches on the w-HAR dataset. The first row shows the accuracy when the classifier is tested on the users available during training and we refer it to the classifier accuracy with users in test set. The second row shows the accuracy for new users and we refer it to the classifier accuracy with users in cross-test set. All the classifiers achieve a high accuracy on the users available for training, however the accuracy drops significantly when tested on new users. The accuracy drop is minimum for the CNN classifier. This shows that CNN has more potential to generalize by producing broadly applicable features in the convolutional layers. Furthermore, CNNs can be easily fine-tuned at runtime for new subjects using the proposed transfer learning approach. This is much more challenging for other approaches. Indeed, the state-of-the-art sensor-based HAR techniques Convnet [214] and Iss2Image [215] employ CNN classifier similar to our choice. They report 95% and 96% accuracy for the UCI HAR [207] dataset, respectively. This level of accuracy is higher than the alternative classifier shown in Table 7.1. In summary, our own comparisons reported in Table 7.1 and state-of-the-art techniques [214, 215] justify choosing a CNN classifier for HAR.

Table 7.1: Comparison of classifiers

	KNN	SVM	DT	RF	CNN
Classifier accuracy with users in the test set (%)	90	89	97	96	98
Classifier accuracy with users in the cross-test set (%)	58	61	52	42	76

The CCA distance, which is used to measure similarity between two networks, is a function of the underlying training data. Therefore, we will use a broad range of HAR datasets for both presenting the proposed transfer learning approach (*in*

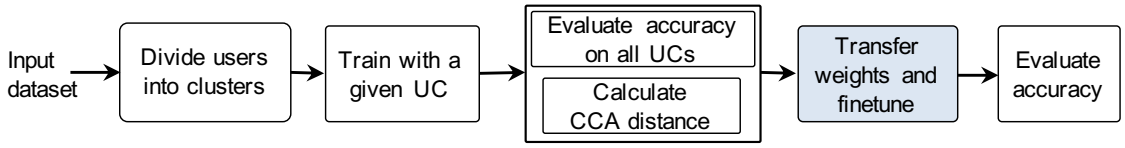


Figure 7.2: Overview of the transfer learning approach for HAR

Section 7.3) and its evaluation (*in Section 7.4*). *The rest of this section overviews these datasets since they are used in the following section.*

Wearable HAR dataset (w-HAR) [27]: The w-HAR dataset contains data of 22 users performing seven (*jump, lie down, sit, stand, stairs down, stairs up, and walk*) and transitions between them. The data is collected using an 3-axis accelerometer and a wearable stretch sensor. The raw data from the users is segmented into activity windows and labeled such that each window contains a single activity. Overall, the dataset contains 4470 windows. Using these windows, the dataset generates discrete wavelet transform (DWT) of the accelerometer data and fast Fourier transform (FFT) of the stretch sensor data. These DWT and FFT features are then supplemented with the minimum and maximum values of the stretch sensor data, and activity window duration to generate 120 features for each window. We use the default feature set provided by the dataset in our analysis.

UCI HAR dataset [207]: The UCI HAR consists of data from 30 users who performed six activities (*lie down, sit, stand, stairs down, stairs up, and walk*) while wearing a smartphone on the waist. The dataset records readings from the accelerometer and gyroscope sensors in the smartphone. The dataset provides 561 time and frequency domain features for each activity window of 2.56 s. We use the default feature set provided by the dataset in our analysis.

UCI HAPT dataset [208]: This is an updated version of the UCI HAR dataset where the authors added information about postural transitions, such as stand-to-sit and sit-to-stand. Similar to the UCI HAR dataset, the HAPT dataset provides 561 time and frequency domain features for each activity.

UniMiB dataset [210]: The UniMiB SHAR is a dataset of acceleration samples acquired with an Android smartphone. The dataset includes nine physical activities performed by 30 subjects. The dataset provides pre-processed windows for user activities. We generate DWT and FFT features, resulting in 450 features for each window.

WISDM dataset [209]: The WISDM dataset consists of data from 36 subjects who performed six activities (*Walking*, *Jogging*, *Upstairs*, *Downstairs*, *Sitting*). The dataset records readings from the accelerometer and gyroscope sensors. Following the authors' recommendation, we use 10 s windows to generate 5424 activity windows from the raw data. Then, we produce the DWT and FFT features for the accelerometer data in each window, resulting in 405 features for each window.

7.3 Transfer Learning for HAR

7.3.1 Flow of the proposed transfer learning approach

This subsection overviews the flow of the proposed approach illustrated in Figure 7.2. Then, the remaining subsections describe each step in more detail.

1. The feature data in each dataset is split into multiple clusters using k-means clustering. The clustering ensures that users across different clusters are more

dissimilar than the random partitioning.

2. Train a CNN classifier for each user cluster (UC) obtained in step 1. The accuracy obtained in this step is the baseline accuracy for each UC, since the training data is available at design time.

3. Evaluate the accuracy for unseen UCs using the classifiers trained in step 2. The accuracy obtained in this step is referred to as the *cross-UC accuracy*. This step also calculates the CCA distance between trained CNNs.

4. Transfer the CNN weights between UCs and fine-tune the layers that provide distinguishable information for each UC. Finally, evaluate the accuracy after fine-tuning the CNN layers.

The rest of this section details these steps.

7.3.2 Clustering users with distinct activity patterns

The first step in the proposed framework is partitioning the users into clusters. Clustering ensures that users in separate clusters have as distinct activity patterns as possible, as illustrated in Figure 7.1. Hence, we can analyze the benefits and efficiency of transfer learning under more challenging conditions than random partitioning. We employ the following steps to obtain the user clusters.

Representative window for each user: Each user has multiple windows for the same activity since the collected data is segmented before feature generation. For example, one-minute-long walking data is divided into 60 activity windows, assuming a one-second segment duration. Furthermore, activity data may be coming

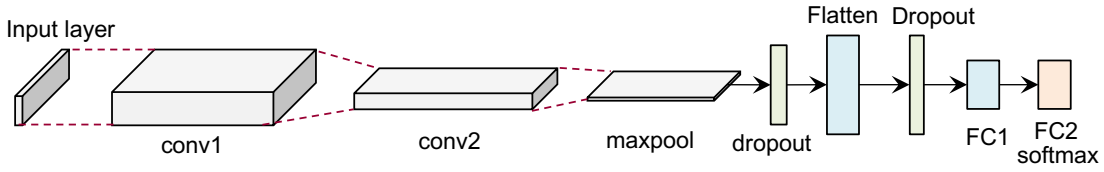


Figure 7.3: The architecture of CNN. The layers annotated at the bottom are used in CCA distance.

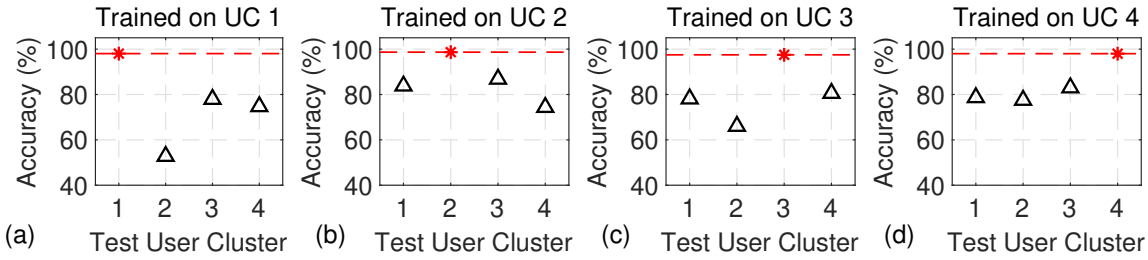


Figure 7.4: The accuracy of the CNNs tested with different UCs for the w-HAR dataset. The red star shows the accuracy of the UC used training while the triangles show cross-UC accuracy.

from different experiments with small differences in sensor locations. As a result, there are variations in the features across different windows, even for a given user and activity. To bypass the variations across windows and facilitate clustering, we identify a *representative window* for each activity of each user. For example, consider our first dataset with 22 users and 8 activities. To obtain the representative windows we first extract all the activity windows for a given user and activity (e.g., all windows labeled as "walk" for user-1). Then, we compute the mean Euclidean distance from each window to all other windows for this user-activity pair. A large distance means that the corresponding window is more likely to be an outlier. In contrast, the window with the smallest mean distance to the other windows is marked as the *representative window* for the corresponding user-activity pair, as illustrated with red lines in Figure 7.1.

k-means clustering: The previous process results in one representative window for each activity for each user. That is, each of the 22 users has 8 representative windows (one for each activity) in our dataset with 22 users and 8 activities. Next, we compute the mean correlation distance [216] between the representative windows across different users for each activity.

A small distance means that the activity pattern of the user is similar to other users. Conversely, a larger distance implies that the user's activity pattern is separated from the other users. The distance to other users for each activity is stored as a multidimensional vector whose length is equal to the number of activities in the dataset.

Finally, the distance vectors are used with the k-means algorithm [211] to generate user clusters that are as separated as possible.

Based on our empirical observations of the data, we choose four clusters each for the five datasets we use in this paper.

7.3.3 Baseline classifier training for each user cluster

CNN for HAR: We design a CNN for each input dataset to recognize the activities in the respective dataset, as shown in Figure 7.3. For each dataset, the data dimensions are different while the structure remains the same. The input layer of the CNN takes the feature vectors as the input in the form of a 2-dimensional (2D) image. This is followed by two convolutional layers, max-pooling, and flatten layers. After flattening the data, we feed it to the two fully connected (FC) layers before applying the softmax activation to classify the activity. Additional details on the structure of

the CNN and training parameters are presented in Section 7.3.6.

CNN accuracy evaluation: The CNN shown in Figure 7.3 is trained for each UC obtained in the previous section. We use 60% data of each UC for training while 20% data is used for cross-validation during training. We train 10 networks with each UC with different initialization such that we can analyze both CCA distance and accuracy on an average basis. Next, we analyze the accuracy of the CNN for the UCs not seen during training using all the data of the unseen UCs. Figure 7.4 shows the average accuracy of CNNs trained with each of the four UCs obtained from our dataset and tested on the other three. First, we see that the CNNs achieve a high accuracy on 20% test data of the UC used for training. However, there is a significant reduction in cross-UC accuracy. For example, the accuracy for UC 2 when tested on CNNs trained with UC 1 is only about 52%. Similar behavior is observed for other UCs as well, with the accuracy drop ranging from 10%–40%. This shows that the CNN is only able to learn the data pattern of the current UC and it cannot generalize other UCs with distinct activity patterns. In the next section, we analyze the distance between networks trained with different UCs to gain better insight into the representations learned by the CNN. Additional details on the accuracy and cross-UC accuracy of other datasets are presented in Appendix A.

7.3.4 Analysis of distance between trained networks

Background on CCA Distance: We employ CCA to analyze the distance between different networks and understand the representational similarity between network layers [89].

CCA has been successfully used to analyze neural network similarities for medical imaging [90], language models [217], and speech recognition [218]. It analyzes the representational similarity between networks by analyzing the ordered output activations of neurons on a set of inputs, instead of working on the network weights directly. Taking the activation vectors of neurons from two layers (trained with different UCs or with different initializations) as inputs, CCA first finds the linear combinations of the activations such that they are as correlated as possible. Once the correlations are obtained, they are used to compute the distance between the two activation vectors, i.e., between the two layers [89].

For illustration, let L1 and L2 be two matrices that represent two intermediate outputs of neural network layers. We denote the covariance matrices of L1 and L2 as $\sum_{L1,L1}$ and $\sum_{L2,L2}$, respectively. Similarly, $\sum_{L1,L2}$ denotes the cross-covariance matrix of L1 and L2. To find the canonical correlation coefficient, we need to solve a singular value decomposition shown in the following equation 7.1:

$$\sum_{L1,L1}^{-1/2} \sum_{L1,L2} \sum_{L2,L2}^{-1/2} = U\Lambda V \tag{7.1}$$

where U, V are the unitary matrices, and Λ is a rectangular diagonal matrix with non-negative real numbers on the diagonal. Then, the canonical correlation coefficient ρ corresponds to the largest singular value of Λ. To make a distance measure, we define $1 - \rho$ as the CCA distance between L1 and L2. This definition assumes that all CCA vectors are equally important to the representations at layer L1 and L2. Otherwise, ρ will have to be a weighted average. A more detailed description can be found in [89]. In this work, we employ the implementation proposed in [89] to

analyze the distance between the CNNs trained for HAR.

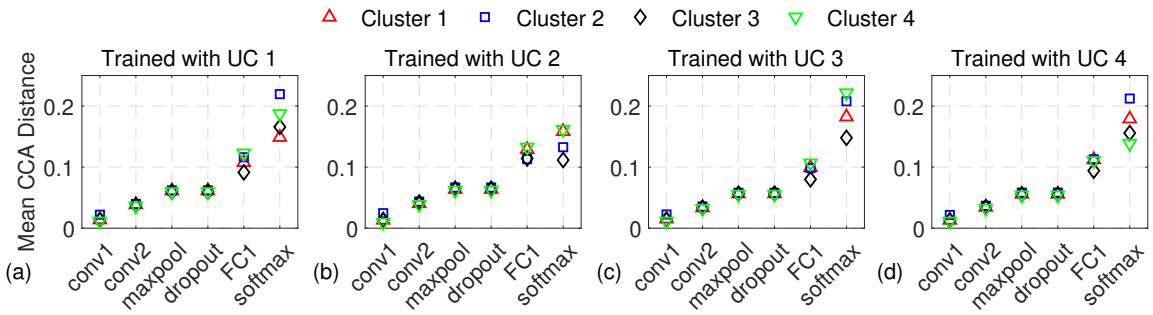


Figure 7.5: The CCA distance between CNNs trained with (a) UC 1, (b) UC 2, (c) UC 3, and (d) UC 4 from the w-HAR dataset when tested on all the four UCs. The final "softmax" is after FC2. Since FC2 and softmax have the same CCA distance, we denote it as "softmax" to keep it consistent. All the CCA figures follow the same convention.

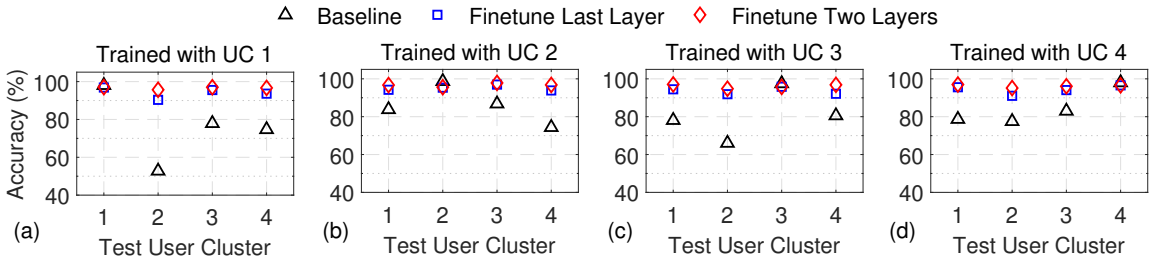


Figure 7.6: Comparison of accuracy between original and fine-tuned CNN for the w-HAR dataset.

Distance between networks trained with same UC: To analyze which layers generalize between users, we calculate the mean CCA distance between the networks trained with the same UC. To this end, 10 CNNs with different initializations are trained with the training data of each UC. Next, we find the mean pairwise CCA distance between the corresponding layers of the 10 CNNs trained with each UC.

Figure 7.5 shows the mean CCA distance among NNs trained with 4 UCs with the w-HAR dataset. Each sub-figure in Figure 7.5 shows the distance between

networks trained with a single UC when tested on all the four UCs. For example, Figure 7.5(a) shows the mean distance for the 10 networks when they are trained on UC 1 and tested on all four user clusters. The figure shows that for the first four layers of the CNN (two convolution layers, one max-pooling layer, and one dropout layer), the mean CCA distance is low regardless of the input UC. Moreover, the distances are almost equal for all the four UCs. This shows that in the first four layers learn features that are similar for all the users in the dataset. To further validate this behavior, we calculate the pairwise CCA distance with features of (UC 1 + UC 2), (UC 1 + UC 3), and (UC 1 + UC 4) on networks trained with UC1. For each of these cases, the CCA distance follows a pattern similar to Figure 7.5.

The networks start to diverge for different UCs from the first fully connected layer. The largest divergence is seen at the softmax layer where the distance is lowest for the UC used for training. This means that the fully connected layers extract information that is specific to each UC and do not generalize to other UCs. We observe a similar trend for the other datasets as well. In addition to this, we also calculate the CCA distance between networks trained with different UCs (*e.g.*, the distance between CNNs trained with UC 1 and UC 2). The analysis of the distance from this perspective helps in understanding the similarity between networks trained with users that have distinct activity patterns. The details of this analysis are presented in Appendix B.

7.3.5 Transferring the NN and fine-tuning

CCA distance analysis reveals that the convolutional layers provide general features while the deeper layers provide the most distinguishing information. We use this insight to optimize the transfer learning process and improve the training time for new UCs. Specifically, we transfer the weights of the first four layers from a trained CNN to a network targeting another UC. Then, the deeper layers are fine-tuned with the data of the new UC. The fine-tuning process uses 60% of the new UC's data for training, 20% data for validation and the remaining 20% for testing. Following this process, we are able to avoid training the convolutional layers, thus saving a significant amount of computations. We evaluate the accuracy after fine-tuning under two scenarios: 1) fine-tune the last FC layer and 2) fine-tune the last two FC layers. Figure 7.6 shows that the accuracy for new UCs improves significantly after fine-tuning either the last FC layer or last two FC layers for w-HAR dataset. With fine-tuning of one layer, we obtain on average 18% accuracy improvement. Specifically, the accuracy for UC 2 improves from 52%, 64%, and 75% to 90%, 92%, and 91%, respectively. When we fine-tune the last two layers, the accuracy improves to 95% for all UCs.

7.3.6 CNN training details

We design a convolutional neural network (CNN) for each of the input datasets to recognize the activities, as shown in Figure 4. The feature vector is first converted to an image to be used as the input to the CNN. The dimensions of the image for each dataset are shown in the second column of Table 7.2. The input layer is followed by

two convolutional layers with 32 and 64 channels, respectively. After performing the convolutions, the data is passed to a max-pooling layer to extract the most distinguishable features. It is then followed by a flattening layer that generates the input vector for the fully connected (FC) layers. One FC layer with 128 neurons and relu activation is included in the CNN. To avoid excessive dependency on certain neurons, two dropout layers with a probability of 0.25 and 0.5 are employed after the max-pooling layer and fully connected layers, respectively. Finally, an output layer with the softmax activation performs the activity classification. The dimensions of each CNN layer for the five datasets are shown in Table 7.2.

We use Tensorflow 2.2.0 [186] with Keras 2.3.4 [187] to train the CNNs. Categorical cross-entropy is employed as the loss during training. The algorithm used for training is the Adadelta [219] optimizer with an initial learning rate of 0.001 and a decay rate of 0.95. We train the CNNs for 100 epochs using a batch size of 128. The training is performed on Nvidia Tesla GPU V100-SXM2 with 32 GB of memory. The details on training time are provided in Section 7.4.2.

Table 7.2: Data dimensions for five datasets

	Input	Conv1	Conv2	Max-pooling	Flatten	FC1	FC2	Softmax
w-HAR	(4, 30, 1)	(4, 30, 32)	(2, 28, 64)	(1, 14, 64)	896	128	8	8
UCI HAR	(33, 17, 1)	(33, 17, 32)	(31, 15, 64)	(15, 7, 64)	6720	128	6	6
UCI HAPT	(33, 17, 1)	(33, 17, 32)	(31, 15, 64)	(15, 7, 64)	6720	128	12	12
UniMiB	(25, 18, 1)	(25, 18, 32)	(23, 16, 64)	(11, 8, 64)	5632	128	9	9
WISDM	(27, 15, 1)	(27, 15, 32)	(25, 13, 64)	(12, 6, 64)	4608	128	6	6

7.4 Experimental Evaluations

7.4.1 Accuracy analysis

The analysis performed in the previous sections showed results with user clusters that are designed to be as distinct as possible. In this section, we validate the transfer learning approach on randomly generated user clusters. To this end, we first generate 200 random user splits from the w-HAR dataset. Each user split contains four user clusters, in line with previous sections. We then train 5 CNNs for each cluster and test their cross-UC accuracy before applying transfer learning. The first column in Figure 7.7 shows the distribution of the cross-UC accuracy. The minimum cross-UC accuracy is 65% for all the UCs among 200 random user splits. We also show the minimum accuracy for the k-means clustering using a red dot. The minimum accuracy with k-means is 52%, which is lower than the minimum accuracy for all random user splits. This experimentally shows that our k-means clustering approximates the *worst-case scenario* well, where the users are as distinct as possible. Next, we fine-tune the last one and two layers of the CNNs to capture the information specific to each user cluster, as shown in the second and third columns of Figure 7.7, respectively. We also analyze the accuracy for 100 randomly chosen UCs and convergence for the UCI HAR and UCI HAPT datasets, respectively. Figures 7.9 and 7.11 show that the median accuracy obtained for the 100 random clusters is similar to the accuracy obtained from the k-means clustering. This is in line with the results from the w-HAR dataset. The classification accuracy improves significantly after the fine-tuning process. Specifically, the median accuracy after

fine-tuning one layer is 93%, while it further increases to about 95% by fine-tuning last two layers. These median accuracies are very close to the accuracy obtained for the k-means clustering. This shows our k-means clustering is representative of a wide range of randomly generated user clusters. We also note that some UCs achieve a higher accuracy after fine-tuning when compared to the k-means clustering. This is because the users in these clusters have similar features. Conversely, some UCs have lower accuracy after fine-tuning because the user cluster that CNN fine-tuned with does not have all the activities present in the tested user cluster.

All convolution layers have a lower CCA distance compared to the deeper layers. These experimental results prove that only fine-tuning the deeper layers is enough to obtain decent accuracy in HAR for different UCs. We also see 2% to 5% accuracy degradation for extra 3 and 6 convolution layers CNN compared to the baseline shallow CNN since simply adding convolution layers in CNN causes overfitting. This result implies that it is proper to choose shallow CNN regarding the sensor-based HAR problem.

In summary, the proposed transfer learning approach of fine-tuning the deeper layers of the network significantly improves the classification accuracy. This shows that the proposed transfer learning approach provides accuracy improvements for multiple datasets.

7.4.2 Training time, loss and convergence analysis

Transfer learning provides benefits in training speed and convergence when compared to training from scratch for new user clusters. Figure 7.8(a) shows the

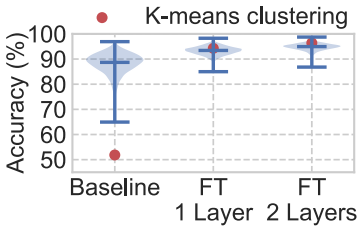


Figure 7.7: Comparison of cross-UC accuracy between original and fine-tuned NN for 200 UCs from the *w-HAR* dataset.

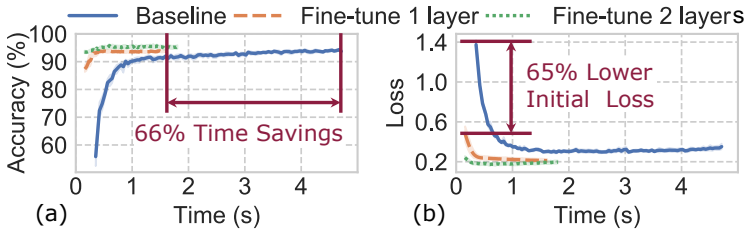


Figure 7.8: Transfer learning improvement analysis: (a) Training time, (b) Loss for the *w-HAR* dataset

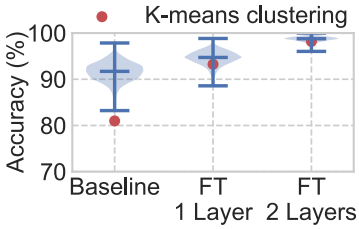


Figure 7.9: Comparison between original and fine-tuned NN for 100 UCs from the *UCI HAR* dataset.

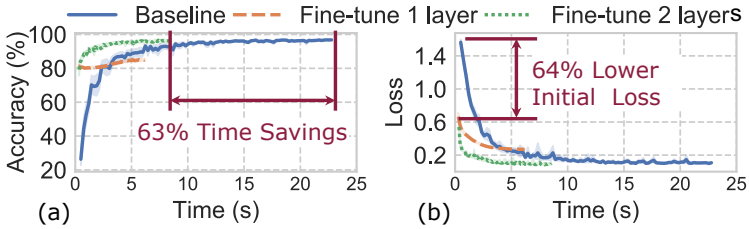


Figure 7.10: Transfer learning improvement analysis:(a) Training time, (b) Loss for the *UCI HAR* dataset.

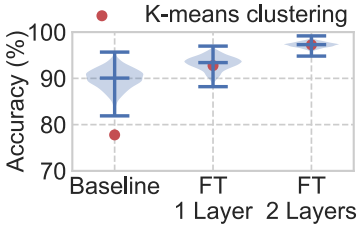


Figure 7.11: Comparison between original and fine-tuned NN for 100 UCs from the *UCI HAPT* dataset.

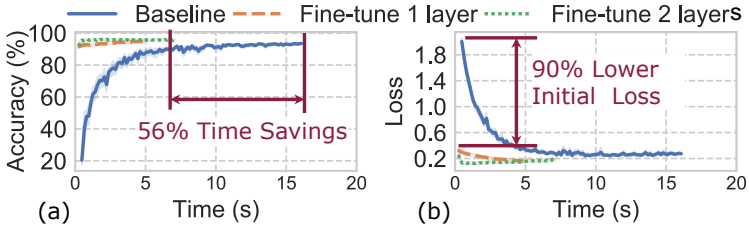


Figure 7.12: Transfer learning improvement analysis:(a) Training time, (b) Loss for the *UCI HAPT* dataset.

comparison of training time between the baseline and proposed transfer learning approach on w-HAR dataset. The transfer learning approach has both a higher starting accuracy and lower convergence time with respect to the baseline on w-HAR dataset. Specifically, fine-tuning one layer converges to 93% in about 1.6 s, which is 66% lower than the baseline approach of training from scratch. When two layers of the CNN are fine-tuned, the accuracy is higher than the baseline with a small increase in the training time. Similar results are observed for the loss in Figure 7.8(b) where the starting loss with transfer learning is 65% lower when compared to the baseline. The loss at the end of training is also lower with the transfer learning approach. Figure 7.10(a) shows the comparison of training time between the baseline and proposed transfer learning approach on UCI HAR dataset. The transfer learning approach has both a higher starting accuracy and lower convergence time compared to the baseline. Specifically, when two layers of the CNN are fine-tuned, the accuracy is the same as the baseline with 63% lower training time. Similar results are observed for the loss in Figure 7.10(b) where the starting loss with transfer learning is 64% lower than the baseline. The loss at the end of training is also lower with the transfer learning approach. Figure 7.12(a) and Figure 7.12(b) show the corresponding results for the UCI HAPT data set. In this case, we observe that the training time is 56% lower, while the starting loss is 90% lower. In summary, this shows that the general features transferred from a trained CNN aid in learning the activities of new users.

7.4.3 Energy and power consumption analysis

This section analyzes the energy and power consumption of the proposed approach since the ability to run on low-power mobile devices is critical for its practicality. To enable this analysis, we implemented it on the Nvidia Jetson Xavier NX Development Kit [174]. This board has a 6-core ARM CPU, 384 Nvidia CUDA cores, and 48 tensor processing units.

Table 7.3: Summary of power and energy consumption of the proposed approach when compared to training from scratch.

	Training from scratch			Fine-tune 1 layer			Fine-tune 2 layers		
	w-HAR	UCI-HAR	WISDM	w-HAR	UCI-HAR	WISDM	w-HAR	UCI-HAR	WISDM
Configurations	4 UCs			4 UCs * 4 FT			4 UCs * 4 FT		
CPU+GPU Power (W)	1.93	5.20	4.69	1.27	2.40	2.26	1.30	3.81	2.29
Total SoC Power (W)	3.26	7.15	6.56	2.51	4.10	3.91	2.58	5.55	3.98
Exec. Time per Case (s)	28.08	59.66	28.80	17.72	33.48	18.50	18.74	37.84	20.12
Energy per Case (J)	91.76	427.00	189.15	44.63	137.33	72.42	48.45	210.01	80.28
Power Saving (%)		N/A		23	43	40	21	22	39
Energy Saving (%)		N/A		51	68	62	47	51	58

Table 7.4: Summary of power and energy consumption of the proposed approach when compared to training from scratch.

	Training from scratch		Fine-tune 1 layer		Fine-tune 2 layers	
	UCI-HAPT	UniMiB	UCI-HAPT	UniMiB	UCI-HAPT	UniMiB
Configurations	4 UCs		4 UCs * 4 FT		4 UCs * 4 FT	
CPU+GPU Power (W)	5.47	4.41	2.52	2.00	3.21	2.16
Total SoC Power (W)	7.36	6.34	4.24	3.54	4.92	3.86
Exec. Time per Case (s)	58.67	39.97	32.79	22.39	37.00	24.72
Energy per Case (J)	431.89	253.38	139.04	79.39	182.05	95.44
Power Saving (%)		N/A	42	33	44	39
Energy Saving (%)		N/A	68	58	69	62

Methodology: We evaluate the power and energy consumption of training the CNN under the following scenarios:

1. Train the classifier on the board from scratch for all 4 UCs in each dataset. Then, find the average energy and power consumption.

2. Fine-tune only one or two layers using the proposed transfer learning approach. Repeat the experiments for all UC combinations and datasets.

In both cases, we repeat the training ten times and take the average to suppress runtime variations.

Results: Table 7.3 shows the power and energy consumption comparisons for w-HAR, UCI, and WISDM datasets, while the results for other datasets are in Table 7.4. The proposed transfer learning approach achieves 21%–43% and 47%–68% reduction in the power and energy consumption, respectively. For instance, if we fine-tune one layer for the UCI-HAR dataset instead of training from scratch, we achieve 68% savings in energy consumption. Table 7.4 shows the power and energy consumption comparisons for UCI-HAPT and UniMiB datasets. The proposed transfer learning approach achieves 33%–44% and 58%–59% reduction in the power and energy consumption, respectively. These results show that the transfer learning approach shown in this paper provides an efficient mechanism to adapt HAR classifiers for new users among all datasets.

8 PANOHEAD: GEOMETRY-AWARE 3D FULL-HEAD SYNTHESIS IN 360°

8.1 Background, Motivation and Contributions

Photo-realistic portrait image synthesis has been a continuous focus in computer vision and graphics, with a wide range of downstream applications in digital avatars, telepresence, immersive gaming, and many others. Recent advances in Generative Adversarial Networks (GANs) [134] has demonstrated strikingly high image synthesis quality, indistinguishable from real photographs [135, 136, 137]. However, contemporary generative approaches operate on 2D convolutional networks without modeling the underlying 3D scenes. Therefore 3D consistency cannot be strictly enforced when synthesizing head images under various poses.

To generate 3D heads with diverse shapes and appearances, traditional approaches require a parametric textured mesh model [220, 221] learned from large 3D scan collections. However, the rendered images lack fine details and have limited perceptual quality and expressiveness. With the advent of differentiable rendering and neural implicit representation [222, 223], conditional generative models have been developed to generate more realistic 3D-aware face images [224, 225, 226, 227]. However, those approaches typically require multi-view image or 3D scan supervision, which are hard to acquire and have limited appearance distribution as those are usually captured in controlled environments.

3D-aware generative models have recently seen rapid progress, fueled by the

integration of implicit neural representation in 3D scene modeling and Generative Adversarial Networks (GANs) for image synthesis [2, 131, 132, 5, 133, 4, 3]. Among them, the seminal 3D GAN, EG3D [5], demonstrates striking quality in view-consistent image synthesis, trained only from in-the-wild single-view image collections. However, these 3D GAN approaches are still limited to synthesis in near-frontal views.

In this chapter, we propose *PanoHead*, a novel 3D-aware GAN for high-quality full 3D head synthesis in 360° trained from only in-the-wild unstructured images. Our model can synthesize *consistent 3D heads viewable from all angles*, which is desirable by many immersive interaction scenarios such as digital avatars and telepresence. To the best of our knowledge, our method is *the first* 3D GAN approach to achieve full 3D head synthesis in 360°.

Extending 3D GAN frameworks such as EG3D [5] to full 3D head synthesis poses several significant technical challenges: Firstly, many 3D GANs [5, 4] cannot separate foreground and background, inducing 2.5D head geometry. The background, formulated typically as a wall structure, is entangled with the generated head in 3D and therefore prohibits rendering from large poses. We introduce a *foreground-aware tri-discriminator* that jointly learns the decomposition of the foreground head in 3D space by distilling the prior knowledge in 2D image segmentation.

Secondly, while being compact and efficient, current hybrid 3D scene representations, like tri-plane [5], introduce strong projection ambiguity for 360° camera poses, resulting in 'mirrored face' on the back head. To address the issue, we present a novel 3D *tri-grid volume representation* that disentangles the frontal features with

the back head while maintaining the efficiency of tri-plane representations.

Lastly, obtaining well-estimated camera extrinsics of in-the-wild back head images for 3D GANs training is extremely difficult. Moreover, an image alignment gap exists between these and frontal images with detectable facial landmarks. The alignment gap causes a noisy appearance and unappealing head geometry. Thus, we propose a novel *two-stage alignment scheme* that robustly aligns images from any view consistently. This step decreases the learning difficulty of 3D GANs significantly. In particular, we propose a camera self-adaptation module that dynamically adjusts the positions of rendering cameras to accommodate the alignment drifts in the back head images.

Our framework substantially enhances the 3D GANs' capabilities to adapt to in-the-wild full head images from arbitrary views. The resulting 3D GAN not only generates high-fidelity 360° RGB images and geometry, but also achieves better quantitative metrics than state-of-the-art methods. With our model, we showcase compelling 3D full head reconstruction from a single monocular-view image, enabling easily accessible 3D portrait creation.

In summary, our main contributions are as follows:

- The first 3D GAN framework that enables view-consistent and high-fidelity full-head image synthesis with detailed geometry, renderable in 360°. We demonstrate our approach in high-quality monocular 3D head reconstruction from in-the-wild images.
- A novel tri-grid formulation that balances efficiency and expressiveness in representing 3D 360° head scenes.

- A foreground-aware tri-discriminator that disentangles 3D foreground head modeling from 2D background synthesis.
- A novel two-stage image alignment scheme that adaptively accommodates imperfect camera poses and misaligned image cropping, enabling training of 3D GANs from in-the-wild images with wide camera pose distribution.

8.2 Methodology

8.2.1 PanoHead Overview

To synthesize realistic and view-consistent full head images, we build PanoHead upon a state-of-the-art 3D-aware GAN, EG3D [5], due to its efficiency and synthesis quality. Specifically, EG3D leverages StyleGAN2 [136] backbone to output a tri-plane representation that represents a 3D scene with three 2D feature planes. Given a desired camera pose c_{cam}, the tri-plane is decoded with a MLP network and volume rendered into a feature image, followed by a super-resolution module to synthesize a higher resolution RGB image I^+. Both the low and high resolution images are then jointly optimized by a dual discriminator $\mathbb{D}$.

In spite of EG3D's success in generating frontal faces, we found it to be a much more challenging task to adapt to 360° in-the-wild full head images for the following reasons: 1) foreground-background entanglement prohibit large pose rendering, 2) strong inductive bias from tri-plane representation causes mirroring face artifacts on the back head, and 3) noisy camera labels and inconsistent cropping of back head images. To address these problems, we introduce a background generator

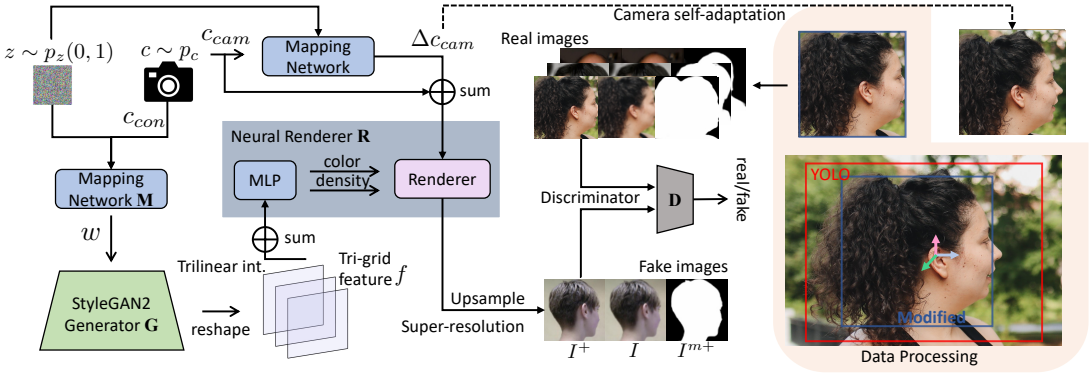


Figure 8.1: Overview of proposed PanoHead method.

and a tri-discriminator for decoupling foreground and background (Section 8.2.2), an efficient yet more expressive tri-grid representation while still being compatible with StyleGAN backbone (Section 8.2.3), and a two-stage image alignment scheme with an self-adaptation module that dynamically adjusts rendering cameras during training (Section 8.2.4). The overall pipeline for our model is illustrated in Figure 8.1.

8.2.2 Foreground-Aware Tri-Discrimination

A typical challenge of state-of-the-art 3D-aware GANs, like EG3D [5], is the entangled foreground with the background of synthesized images. Regardless of the highly detailed geometry reconstruction, directly training the 3D GAN from in-the-wild RGB image collections, such as FFHQ [135], results in a 2.5D face, as illustrated in Figure 8.2 (a). Augmenting with image supervisions from the side and back of the head helps build up the full-head geometry with reasonable back head shapes. However, it does not solve the problem because the tri-plane representation itself is not designed to represent separated foreground and background.

To disentangle the foreground from the background, we first introduce an additional StyleGAN2 network [136] to generate 2D backgrounds at the same resolution of raw feature image I^r. During volume rendering, the foreground mask I^m can be obtained by:

$$I^r(r) = \int_0^\infty w(t)f(r(t))dt,\ I^m(r) = \int_0^\infty w(t)dt, \tag{8.1}$$

$$w(t) = exp\big(-\int_0^t \sigma(r(s))ds\big)\sigma(r(t)), \tag{8.2}$$

where $r(t)$ represents a ray emitted from the rendering camera center. The foreground mask is then used to compose a new low-resolution image I^{gen}:

$$I^{gen} = (1 - I^m)I^{bg} + I^r, \tag{8.3}$$

which is fed into the super-resolution module. Note that the computation cost of background generator is insignificant since its output has a much lower resolution than the tri-plane generator and super-resolution module.

Simply adding a background generator does not fully decouple it from the foreground since the generator tends to synthesize foreground content in the background. Thus, we propose a novel foreground-aware tri-discriminator to supervise the rendered foreground mask along with the RGB images. Specifically, the input of the tri-discriminator has 7 channels, composed with a bilinearly-upsampled RGB image I , a super-resolved RGB image I^+ and single-channel upsampled foreground mask I^{m+}. The additional mask channel allows the 2D segmentation prior knowledge to be back-propagated into the density distribution of the neural radiance field.

Figure 8.2: Geometry and RGB images from dual-discrimination (a) and foreground-aware tri-discrimination (b, c). EG3D (a) fails to decouple the background. PanoHead's tri-discrimination offers both background-free geometry (b) and background-switchable full head image synthesis (c).

Our approach reduces the learning difficulty in shaping the 3D full head geometry from unstructured 2D images, enabling authentic geometry ((Figure 8.2 (b))) and appearance synthesis of a full head composable with various backgrounds (Figure 8.2 (c)). We note that in constrast from ENARF-GAN [143] that employs a single discriminator for RGB images composed of synthesized foreground and background images using a dual-generated mask, our tri-discriminator better ensures view-consistent high-resolution outputs.

8.2.3 Feature Disentanglement in Tri-Grid

The tri-plane representation, proposed in EG3D [5], offers an efficient representation for 3D generation. The neural radiance density and appearance of a volume

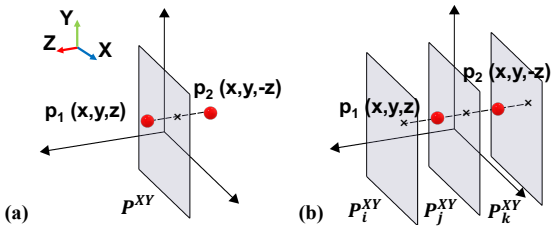


Figure 8.3: Comparison between tri-plane (a) and tri-grid (b) architecture in Z axis.

point are obtained by projecting its 3D coordinate over three axis-aligned orthogonal planes and decoding the sum of three bilinearly interpolated features with a tiny MLP. However, when synthesizing a full head in 360°, we observe tri-plane is limited in expressiveness and suffers from mirroring-face artifacts. The problem is even pronounced when the camera distribution of the training images is unbalanced. The root cause is the inductive bias originating from tri-plane projection, where one point on a 2D plane has to represent features of different 3D points. For example, a point on the front face and a point on the back hair will be projected to the same point on the XY plane P^{XY} (orthogonal to Z axis), as illustrated in Figure 8.3 (a). Although the other two planes should theoretically provide complementary information to alleviate this projection ambiguity, we found it not the case when there is less visual supervision from the back or when the structure of the back head is challenging to learn. The tri-planes are prone to borrow features from the front face to synthesize the back head, referred to as mirroring-face artifacts here (Figure 8.4(a)).

To reduce the inductive bias of the tri-plane, we lift its formulation into a higher dimension by augmenting tri-plane with an additional depth dimension. We call

Figure 8.4: Images synthesis with tri-plane and tri-grid ($D = 3$). Due to the projection ambiguity, tri-plane representation (a) can generate good-quality front face image yet with a 'mirrored face' on back head, while our tri-grid representation synthesizes high-quality back head appearance and geometry (b).

this enriched version as a tri-grid. Instead of having three planes with a shape of $H \times W \times C$ with H and W being the spatial resolution and C being the number of channel, each of our tri-grid has a shape of $D \times H \times W \times C$, where D represents the depth. For instance, to represent spatial features on the XY plane, tri-grid will have D axis-aligned feature planes $P_i^{XY}, i = 1, \ldots, D$ uniformly distributed along the Z axis. We query any 3D spatial point by projecting its coordinate onto each of the tri-grid, retrieving the corresponding feature vector by *tri-linear interpolation*. As such, for two points sharing the same projected coordinates but with different depths, the corresponding feature would be likely to be interpolated from non-shared planes (Figure 8.3 (b)). Our formulation disentangles the feature presentation of the front face and back head and therefore largely alleviates the mirroring-face artifacts (Figure 8.4).

Similar to tri-plane in EG3D [5], we can synthesize the tri-grid as $3 \times D$ feature planes using the StyleGAN2 generator [135]. That is, we increase the number of

output channels of the original EG3D backbone by D times. Thus, tri-plane can be regarded as a naïve case of our tri-grid representation with $D = 1$. The depth D of our tri-grid is tunable and larger D offers more representation power at the cost of additional computation overhead. Empirically we find a small value of D $(D = 3)$ is sufficient in feature disentanglement while still maintaining its efficiency as a 3D scene representation.

8.2.4 Self-Adaptive Camera Alignment

For adversarial training of our full head in 360°, we need in-the-wild image exemplars from a much wider range of camera distribution than the mostly frontal distribution, as in FFHQ [135]. Although our 3D-aware GAN is only trained from widely-accessible 2D images, the key to the best quality training is accurate alignment of visual observations across images labeled with well-estimated camera parameters. While a good practice has been established for frontal face images cropping and alignment based on facial landmarks, it has never been studied in preprocessing large-pose images for GAN training. Both camera estimation and image cropping are no longer straightforward due to the lack of robust facial landmarks detection for images taken from the side and back.

To resolve the aforementioned challenge, we propose a novel two-stage processing. In the first stage, for images with detectable facial landmarks, we still adopt the standard processing where the faces are scaled to a similar size and aligned at the center of the head using state-of-the-art face pose estimator 3DDFA [228]. For the rest of the images with large camera poses, we employ a head pose estimator

Figure 8.5: Image synthesized without (a) and with the camera self-adaptation scheme(b). Without it, the model generates misaligned back head images, leading to a defective dent in back head.

WHENet [229] that provides a roughly-estimated camera pose, and a human detector YOLO [230] with a bounding box centered at the detected head. To crop the images at a consistent head scale and center, we apply both YOLO and 3DDFA on a batch of front-face images, from which we adjust the scale and translation of the head center of YOLO with constant offsets. This approach enables us to pre-process all head images with labeled camera parameters and in a consistent alignment to a large extent.

Due to the presence of various hairstyles, there is still inconsistency in the alignment of back head images, inducing significant learning difficulties for our network to interpret the complete head geometry and appearance (see Figure 8.5 (a)). We, therefore, propose a self-adaptive camera alignment scheme to fine-tune the transformation of volume rendering frustum for each training image. Specifically, our 3D-aware GAN associates each image with a latent code z that embeds the 3D scene information of geometry and appearance, which can be

synthesized at a view of c_{cam}. c_{cam} might not align well with the image content for our training images; so, it is hard for the 3D GAN to figure out a reasonable full head geometry. Therefore we co-learn a residual camera transformation Δc_{cam} mapped from (z, c_{cam}) together with our adversarial training. The magnitude of Δc_{cam} is regularized with a L_2 norm. Essentially, the network dynamically self-adapts the image alignment with refined correspondence across different visual observations. We note that this is only possible credited to the nature of 3D-aware GAN that can synthesize view-consistent images at various cameras. Our two-stage alignment enables 360-degree view-consistent head synthesis with authentic shape and appearance, learnable from diverse head images with widely distributed camera poses, styles, and structures.

8.3 Experimental Evaluations

8.3.1 Datasets and Baselines

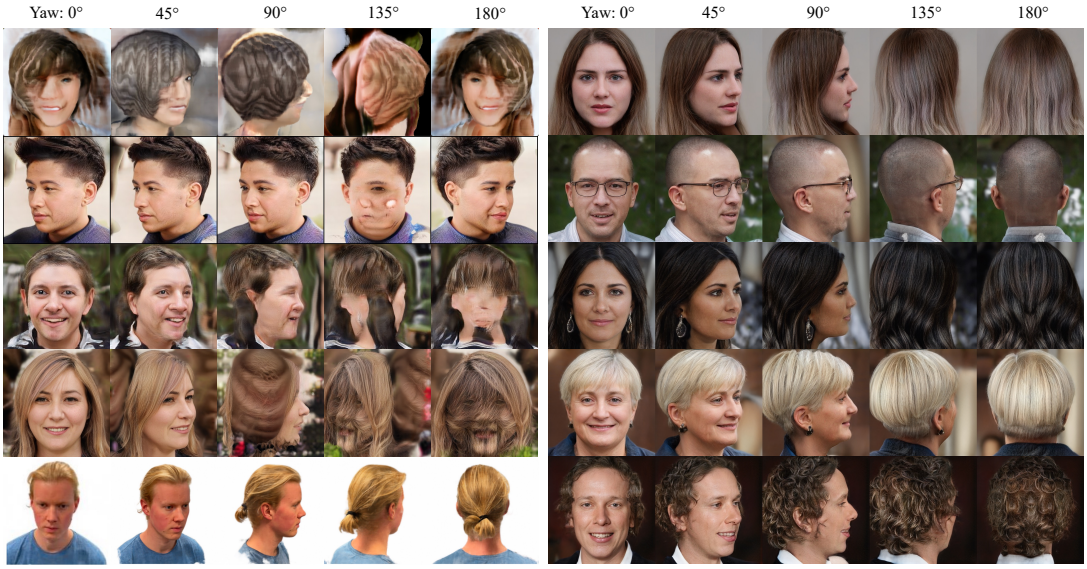
We train and evaluate our framework on a balanced combination of FFHQ [135], K-hairstyle dataset [231], and an in-house large-pose head image collection. FFHQ contains 70K diverse high-resolution face images, yet mainly fall in the absolute yaw range from 0° to 60°, assuming up-front camera pose corresponds to 0°. We augment the FFHQ dataset with 4K back-head images from K-hairstyle dataset and 15K in-house large-pose images with diverse styles, ranging from 60° to 180°. For brevity, we name this dataset combination as FFHQ-F. We refer to the supplementary paper for more dataset analysis and network training details.

We compare against state-of-the-art 3D-aware GANs including GRAF [2], EG3D [5], StyleSDF [4], and GIRAFFEHD [3]. All baselines are retrained from the same FFHQ-F dataset. We measure the quality of generated multiview images and geometry both quantitatively and qualitatively.

8.3.2 Qualitative Comparisons

360° Image Synthesis. Figure 8.6 visually compares the image quality against the baselines, all trained with FFHQ-F, by synthesizing images from five different views, ranging the yaw angle from 0 to 180°. GRAF [2] fails to synthesize compelling head images and its background is entangled with foreground head. StyleSDF [4] and GIRAFFEHD [3] are able to synthesize realistic frontal face images but in low perceptual quality when rendered from a larger camera pose. Without explicit reliance on camera labels, we suspect the above methods have difficulty in interpreting the 3D scene structures by themselves directly from images with 360° camera distribution.

We observe that EG3D [5] is able to synthesize high-quality view-consistent frontal head images before rotating the view to the side or even the back. Mirroring face artifacts are clearly observable from the back, due to the tri-plane's projection ambiguity and the entangled fore-background. The method proposed in [6] builds personalized full-head NeRF at the extra cost of multi-view supervision. Regardless of its good quality images at all views, the model itself is not a generative model. In strong contrast, our model generates superior photo-realistic head images *for all camera poses* while retaining multi-view consistency. It delivers photo-realism



(a) Baselines: GRAF, GIRAFFEHD, StyleSDF, EG3D, a full-head NeRF (b) PanoHead with different seeds

Figure 8.6: Qualitative comparison between GRAF [2], GIRAFFEHD [3], StyleSDF [4], EG3D [5], multi-view supervised NeRF [6] (different methods from top to bottom on left side), and our PanoHead (right). Except [6], all models are trained on FFHQ-F.

with fine details at diverse appearances, ranging from shaved head with glasses to long curly hairstyles. To better appreciate our multi-view full-head synthesis, please refer to our supplementary video for more comprehensive visual results.

Geometry Generation. Figure 8.7 compares the visual quality of the underlying 3D geometry extracted by running Marching Cubes algorithms [232]. While StyleSDF [4] generates decent appearances of the front face, the complete geometry of the head is noisy and broken. EG3D presents detailed geometry of front face and hair, but either with background concrete entangled (Figure 8.2(a)) or with a hollowed back head (Figure 8.7). In contrast, our model can consistently generate high-fidelity background-free 3D head geometry even with various hairstyles.

Table 8.1: Metrics comparison across all baselines. For segmentation MSE, only GIRAFFEHD and PanoHead decouple the background and foreground. For ID score, GRAF's low-quality images lead to facial detection failure.

	GRAF	GIRAFFEHD	StyleSDF	EG3D	Ours
FID-all ↓	68.2	37.3	78.5	6.2	**5.4**
MSE (10^{-2}) ↓	N/A	42.6	N/A	N/A	**9.1**
ID ↑	N/A	0.39	0.41	**0.74**	**0.74**

8.3.3 Quantitative Results

To quantify the visual quality, fidelity, and diversity of the generated images, we employ Frechet Inception Distance (FID) [233] of 50K real and fake image samples. We measure the multi-view consistency using the identity similarity score (ID) by calculating the average Adaface [234] cosine similarity score from paired synthesized face images rendered from different camera poses. Note that this metric can only be applied to those images with detected facial landmarks. We assess mean square error (MSE) to calculate the accuracy of the generated segmentation against the mask obtained with DeepLabV3 ResNet101 network [235]. Table 8.1 compares these metrics across all baselines and our method. We observe that our model outperforms other baselines consistently from all perspectives. Refer to supplemental material for metrics definition and implementation details.

To evaluate the image quality at different views, we employ FID and Inception Score (IS) [236] for synthesized images with only back poses ($|yaw| \geqslant 90°$), front poses ($|yaw| < 90°$), and all camera poses. FID measures on the similarity and diversity of real and fake image distributions while IS focuses more on the image quality itself. Our GAN model follows EG3D for the main backbone, where the

Table 8.2: Ablation studies on different components. +seg. means with foreground-aware tri-discrimination. +self-adpat. means with camera self-adaptation scheme. All are trained with FFHQ-F

	EG3D	+seg.		+seg.&self-adapt.
		tri-plane	tri-grid	tri-grid
FID-back ↓	50.4	44.1	44.0	**40.9**
FID-front ↓	6.6	**5.0**	5.5	5.4
FID-all ↓	6.2	**5.2**	**5.2**	5.4
IS-back ↑	4.3	3.9	4.2	**4.4**
IS-front ↑	3.9	**4.1**	**4.1**	**4.1**
IS-all ↑	3.8	4.0	4.0	**4.1**
Runtime ↓	**1**	1.14×	1.26×	1.28×

tri-plane generator is conditioned on a camera pose. We observe that such a design leads to biased image synthesis quality toward the conditioning camera pose. Specifically, when conditioning on the front view, our generator achieves inferior quality for synthesizing the head images from the back, and vice versa. However, when calculating FID-all, the conditioning camera is always the same as the rendering view. Therefore the generator could still achieve an excellent FID-all score even though the quality of generated heads might degenerate in unseen views. Hence, the original FID metrics (FID-all and FID-front) can hardly thoroughly reflect the overall generation quality of full heads in 360°. To alleviate this issue, we propose FID-back, where we condition on the front view but synthesize the images from the back. It leads to higher FID scores but reflects the quality in 360° image synthesis better.

We perform an ablation study on our method to quantitatively evaluate the efficacy of each individual component (Table 8.2). As shown in the second column, we

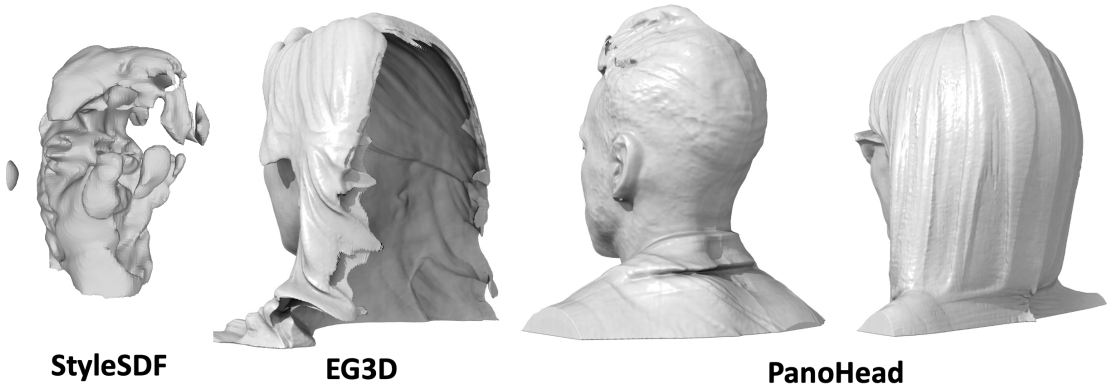


Figure 8.7: PanoHead achieves high-quality complete head geometry whereas StyleSDF [4] and EG3D [5] produce 3D noises or hallowed heads.

notice a significant quality boost after adding the foreground-aware discrimination for all cases, compared with the original EG3D. That indicates the prior segmentation knowledge largely ease the network learning difficulty of 3D heads from in-the-wild image collections. Frontal face synthesis quality is comparable among all methods given the strong supervision from the large amount of well-aligned frontal images. However, for the back head, decoupling foreground and background largely improves the synthesis quality. In addition, changing tri-plane to tri-grid representation further enhances the image quality. With tri-discrimination, tri-grid, and camera self-adaption scheme altogether, PanoHead achieves the lowest FID-back and the highest IS for back head generation. As reflected in the row of run-time analysis, our novel component only introduces minor computation overhead, but with significant image synthesis quality improvements. Note that the frontal image quality is superior to the back head, largely due to the significant learning difficulty in various hairstyles and unstructured back-head appearances.

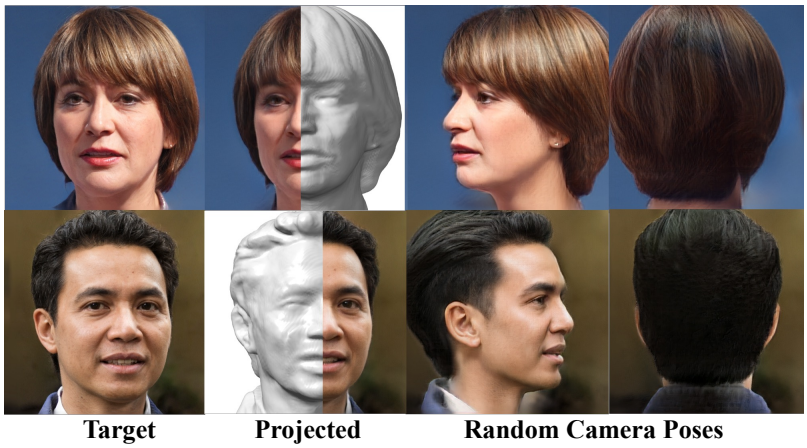


Figure 8.8: Single-view reconstruction from different camera poses. The first column shows the target images, second column projected RGB images and reconstructed 3D shapes using GAN inversion, last two columns rendered images from any given camera poses.

8.3.4 Single-view GAN Inversion

Figure 8.8 demonstrates full-head reconstruction from a single-view portrait using PanoHead's generative latent space. To achieve that, we first perform an optimization to find the corresponding latent noise z for the target image using pixel-wise L_2 loss and image-level LPIPS loss [237]. To further improve reconstruction quality, we perform pivotal tuning inversion (PTI) [238] to alter the generator parameters with a fixed optimized latent code z. From a single-view target image, PanoHead not only reconstructs photo-realistic image and high-fidelity geometry but also enables novel-view synthesis in 360°, including large pose and back head.

8.4 Discussion

Limitations. While PanoHead exhibits excellent images and shapes quality from 360°, it still contains minor artifacts, e.g. in the teeth area. Similar to the original EG3D, flickering texture issue is also noticeable in our model. Switching to StyleGAN3 [239] as the backbone would help preserve high-frequency details. In practice, we also observe more noticeable flickering artifacts with a higher swapping probability of the conditional camera pose. We set this value to 70% as opposed to 50% in EG3D since we empirically find it enhances 360° rendering quality but at the minor cost of flickering texture artifacts. Another observation is that it lacks finer high-frequency geometric details, e.g. hair tips. We leave it as future work to quantitatively evaluate our geometric quality such as using depth maps. Finally, although PanoHead is able to generate diverse images in terms of gender, races, and appearances, reliance on training with only several datasets combination still makes it suffer from data bias, to some extent. In spite of our data collection effort, large-scale full-head annotated training image dataset is one of the most critical directions to facilitate full-head synthesis research. We anticipate such datasets can resolve some of the limitations aforementioned.

Ethical considerations. PanoHead is not specifically designed for any malicious uses, yet we do realize that the single-view portrait reconstruction could be manipulated, which might pose a social threat. We do not encourage the method being used for violating others' rights in any forms.

9 CONCLUSIONS AND FUTURE DIRECTIONS

The swift advancement of emergent technologies, encompassing IoT devices, AI, and ML, has sparked the evolution of sophisticated forms of human representation. Human representation, broadly speaking, involves capturing and describing various human attributes and behaviors via sensor data. Such representations have the capacity to encapsulate manifold aspects of human existence, including body posture, facial expressions, behavior patterns, and psychological states. Insightful and transformative human-centered applications can be materialized through the comprehension and manipulation of human representation. However, this being an incipient field, we are still some distance from attaining a comprehensive and accurate understanding of human representation.

In this regard, this dissertation addresses major technical and adaptation challenges and presents three foundational aspects concerning human representation: i) the examination of emerging sensing modalities, where we undertake a comprehensive investigation of the strengths and limitations of various modalities, selecting the most appropriate one for each specific application; ii) the design of intelligent algorithms, which covers the entire process from raw sensor data preprocessing to deep learning model training, with an emphasis on lightweight designs that can readily adapt to unseen users and scenarios; iii) the exploration of innovative applications, with a particular focus on healthcare.

The subsequent sections of this thesis detail six studies aimed at these objectives. Firstly, we introduce MGait, an innovative and practical step-length estimation

technique that utilizes low-power wearable bend and inertial sensors. Experimental outcomes indicate that the proposed model estimates step length with a 5.49% mean absolute percentage error, providing precise real-time feedback to the user. Secondly, we propose MARS, the first mmWave-based assistive rehabilitation system employing human pose estimation, enabling home-based rehabilitation tasks while preserving users' privacy and meeting low-power requirements. Thirdly, we present mRI, a multi-modal 3D human pose estimation dataset that for the first time systematically explores human pose estimation and action localization tasks using both intrusive modalities (IMU) and non-intrusive modalities (mmWave and RGB-D).

Regarding algorithm design, we present FUSE, a fast and scalable human pose estimation framework that combines multi-frame representation and meta-learning techniques, enabling efficient and accurate estimation of human joint coordinates. We also propose a transfer learning framework for human activity recognition that leverages representational analysis to identify common features transferable across users, thereby improving generalizability.

Lastly, we explore generative models for human representation and introduce PanoHead, the first-ever 3D-aware generative model capable of synthesizing high-quality, view-consistent images of humans heads in 360° with varied appearances and detailed geometry.

In summary, this dissertation addressed several critical gaps in obtaining and understanding human representation by making the following contributions:

- Present MGait, a model-based gait monitoring technique [34],

- Present MARS, a mmWave-based human pose estimation framework for rehabilitation [36],
- Present mRI, an open-source multi-modal 3D human pose estimation dataset [43],
- Present FUSE, fast and scalable human pose estimation using mmWave point cloud [37],
- Present a transfer learning algorithm using representation analysis for human activity recognition [33], and
- Present Panohead, a 3D generative model for 360° human head synthesis [44]

9.1 Future Directions

Looking ahead, there are several potential avenues for future research and exploration. These range from improvements in sensor modality understanding and integration, to synergies with emerging hardware, to utilizing large models. In each of these areas, innovation and new developments could significantly impact our ability to create more comprehensive human representations.

Self-supervised learning and multi-modal sensing: The interplay between these two areas can offer invaluable insights in improving human representation models. What makes this particularly exciting is the potential for learning cross-modal semantics. For example, what does an IMU signal's pattern say about the language or emotion of the individual? How can we use visual cues to decode underlying mmWave data? By jointly learning these different modalities under a

self-supervised paradigm, we can build robust human representation models that could function efficiently even with limited labeled data. Additionally, I envision a future where one modality could augment the learning of another, allowing for robustness in environments where certain data modalities may be sparse or unavailable.

Synergizing with Emerging Hardware: In the realm of envisioning the future, it's fascinating to consider the possiblities of deploying smart algorithms on emerging hardwares that make comprehensive human representation a reality. We're on the cusp of witnessing a era of hardware innovations that could serve as viable platforms for this deployment. For instance, the advent of smart textiles – clothing embedded with technology that can interact with the wearer's environment – offers promising prospects. These devices could provide a seamless interface for capturing human representation data in an unobtrusive and user-friendly manner. Similarly, the emergence of energy-harvesting wearables, such as those harnessing power from solar sources or even the kinetic energy of human motion, provide exciting opportunities for sustainability and constant uptime, bypassing traditional limitations of battery life. While algorithmic innovation indeed forms the bedrock of advancements in human representation, it's the revolution at the hardware level that could truly democratize its benefits, making them palpable in the daily lives of individuals.

Utilizing large-scale models: The rapid evolution of large language models (LLM) and large vision models (LVM) offers an unprecedented opportunity for complex human representation tasks. We are now at a stage where these models can internal-

ize and generate a massive body of semantic, visual, and sensor-based information, providing a rich reservoir of priors for any task. Armed with large models, we could in turn deepen our understanding of human representation, even manipulating and editing various attributes to suit our needs.

As we continue to innovate and make advancements in these directions, it's essential to be mindful of potential ethical implications. As we develop better human representations, issues related to privacy, consent, and data misuse may become more critical. Developing strategies to address these concerns while pursuing more accurate and comprehensive representations will be a key challenge for future work in this field.

BIBLIOGRAPHY

[1] Ben O'Brien, Todd Gisby, and Iain A Anderson. Stretch Sensors for Human Body Motion. In *Proc. Electroact. Polym. Actuators Devices*, volume 9056, page 905618, 2014.

[2] Katja Schwarz, Yiyi Liao, Michael Niemeyer, and Andreas Geiger. Graf: Generative radiance fields for 3d-aware image synthesis. *NeurIPS*, 2020.

[3] Yang Xue, Yuheng Li, Krishna Kumar Singh, Yong Jae Lee, Krishna Kumar, Singh Yong, and Jae Lee. GIRAFFE HD: A High-Resolution 3D-aware Generative Model. In *CVPR*, pages 18440–18449, 2022.

[4] Roy Or-El, Xuan Luo, Mengyi Shan, Eli Shechtman, Jeong Joon Park, and Ira Kemelmacher-Shlizerman. Stylesdf: High-resolution 3d-consistent image and geometry generation. *arXiv preprint arXiv:2112.11427*, 2021.

[5] Eric R Chan, Connor Z Lin, Matthew A Chan, Koki Nagano, Boxiao Pan, Shalini De Mello, Orazio Gallo, Leonidas Guibas, Jonathan Tremblay, Sameh Khamis, and Others. Efficient geometry-aware 3D generative adversarial networks. *CVPR*, 2022.

[6] Stanislaw Szymanowicz, Virginia Estellers, Tadas Baltrusaitis, and Matthew Johnson. Photo-realistic 360 Head Avatars in the Wild. *arXiv Prepr. arXiv2210.11594*, 2022.

[7] Tsung-Yi Lin, Michael Maire, Serge Belongie, James Hays, Pietro Perona, Deva Ramanan, Piotr Dollár, and C Lawrence Zitnick. Microsoft coco: Common objects in context. In *Eur. Conf. Comput. Vis.*, pages 740–755. Springer, 2014.

[8] Mykhaylo Andriluka, Leonid Pishchulin, Peter Gehler, and Bernt Schiele. 2d human pose estimation: New benchmark and state of the art analysis. In *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, pages 3686–3693, 2014.

[9] Catalin Ionescu, Dragos Papava, Vlad Olaru, and Cristian Sminchisescu. Human3. 6m: Large scale datasets and predictive methods for 3d human sensing in natural environments. *IEEE Trans. Pattern Anal. Mach. Intell.*, 36(7):1325–1339, 2013.

[10] Amir Shahroudy, Jun Liu, Tian-Tsong Ng, and Gang Wang. NTU RGB+D: A large scale dataset for 3D human activity analysis. In *Proc. IEEE Conf. Comput. Vis. pattern Recognit.*, pages 1010–1019, 2016.

[11] Hanbyul Joo, Hao Liu, Lei Tan, Lin Gui, Bart Nabbe, Iain Matthews, Takeo Kanade, Shohei Nobuhara, and Yaser Sheikh. Panoptic Studio: A Massively Multiview System for Social Motion Capture. In *IEEE Int. Conf. Comput. Vis.*, 2015.

[12] Dushyant Mehta, Helge Rhodin, Dan Casas, Pascal Fua, Oleksandr Sotnychenko, Weipeng Xu, and Christian Theobalt. Monocular 3D Human Pose Estimation In The Wild Using Improved CNN Supervision. In *3D Vis. (3DV), 2017 Fifth Int. Conf.* IEEE, 2017.

[13] Timo von Marcard, Gerard Pons-Moll, and Bodo Rosenhahn. Human Pose Estimation from Video and IMUs. *Trans. Pattern Anal. Mach. Intell.*, 38(8):1533–1547, jan 2016.

[14] Timo von Marcard, Roberto Henschel, Michael J Black, Bodo Rosenhahn, and Gerard Pons-Moll. Recovering accurate 3d human pose in the wild using imus and a moving camera. In *Proc. Eur. Conf. Comput. Vis.*, pages 601–617, sep 2018.

[15] Timo von Marcard, Roberto Henschel, Michael J Black, Bodo Rosenhahn, and Gerard Pons-Moll. Recovering accurate 3d human pose in the wild using imus and a moving camera. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 601–617, 2018.

[16] Timo Von Marcard, Bodo Rosenhahn, Michael J Black, and Gerard Pons-Moll. Sparse inertial poser: Automatic 3d human pose estimation from sparse

IMUs. In *Comput. Graph. Forum*, volume 36-2, pages 349–360. Wiley Online Library, 2017.

[17] Gerard Pons-Moll, Andreas Baak, Thomas Helten, Meinard Müller, Hans-Peter Seidel, and Bodo Rosenhahn. Multisensor-Fusion for 3D Full-Body Human Motion Capture. In *IEEE Conf. Comput. Vis. Pattern Recognit.*, jun 2010.

[18] Xinyu Yi, Yuxiao Zhou, and Feng Xu. TransPose: Real-time 3D Human Translation and Pose Estimation with Six Inertial Sensors. *ACM Trans. Graph.*, 40(4), 2021.

[19] Mingmin Zhao, Others, Yonglong Tian, Hang Zhao, Mohammad Abu Alsheikh, Tianhong Li, Rumen Hristov, Zachary Kabelac, Dina Katabi, Antonio Torralba, and Others. RF-based 3D skeletons. In *Proc. Conf. ACM Spec. Interes. Gr. Data Commun.*, pages 267–281, 2018.

[20] Mingmin Zhao, Tianhong Li, Mohammad Abu Alsheikh, Yonglong Tian, Hang Zhao, Antonio Torralba, Dina Katabi, Mohammad Abu Alsheikh, Yonglong Tian, Hang Zhao, Antonio Torralba, and Dina Katabi. Through-Wall Human Pose Estimation Using Radio Signals. *Proc. IEEE Comput. Soc. Conf. Comput. Vis. Pattern Recognit.*, pages 7356–7365, 2018.

[21] Hongfei Xue, Yan Ju, Chenglin Miao, Yijiang Wang, Shiyang Wang, Aidong Zhang, and Lu Su. mmMesh: Towards 3D real-time dynamic human mesh construction using millimeter-wave. In *Proc. 19th Annu. Int. Conf. Mob. Syst. Appl. Serv.*, pages 269–282, 2021.

[22] Arindam Sengupta, Feng Jin, Renyuan Zhang, and Siyang Cao. Mm-pose: Real-time human skeletal posture estimation using mmWave radars and CNNs. *IEEE Sens. J.*, 20(17):10032–10044, 2020.

[23] Toygun Basaklar, Yigit Tuncel, Sizhe An, and Umit Ogras. Wearable devices and low-power design for smart health applications: challenges and oppor-

tunities. In *2021 IEEE/ACM Int. Symp. Low Power Electron. Des.*, page 1. IEEE, 2021.

[24] Google. Google completes Fitbit acquisition. https://blog.google/products/devices-services/fitbit-acquisition/ accessed 8 Jul. 2021, 2021.

[25] Apple. Apple Watch. Helping your patients identify early warning signs. https://www.apple.com/healthcare/apple-watch/ accessed 8 Jul. 2021, 2021.

[26] Ganapati Bhat, Ranadeep Deb, and Umit Y Ogras. OpenHealth: open-source platform for wearable health monitoring. *IEEE Des. & Test*, 36(5):27–34, 2019.

[27] Ganapati Bhat, Nicholas Tran, Holly Shill, and Umit Y Ogras. w-HAR: An activity recognition dataset and framework using low-power wearable devices. *Sensors*, 20(18):5356, 2020.

[28] Xiaodong Qu, Saran Liukasemsarn, Jingxuan Tu, Amy Higgins, Timothy J Hickey, and Mei-Hua Hall. Identifying clinically and functionally distinct groups among healthy controls and first episode psychosis patients by clustering on eeg patterns. *Frontiers in psychiatry*, 11:541659, 2020.

[29] Xiaodong Qu, Peiyan Liu, Zhaonan Li, and Timothy Hickey. Multi-class time continuity voting for eeg classification. In *Brain Function Assessment in Learning: Second International Conference, BFAL 2020, Heraklion, Crete, Greece, October 9–11, 2020, Proceedings 2*, pages 24–33. Springer, 2020.

[30] Xiaodong Qu, Qingtian Mei, Peiyan Liu, and Timothy Hickey. Using EEG to distinguish between writing and typing for the same cognitive task. In *Int. Conf. Brain Funct. Assess. Learn.*, pages 66–74. Springer, 2020.

[31] Xiaodong Qu and Timothy J Hickey. Eeg4home: A human-in-the-loop machine learning model for eeg-based bci. In *International Conference on Human-Computer Interaction*, pages 162–172. Springer, 2022.

[32] Ganapati Bhat, Yigit Tuncel, Sizhe An, and Umit Y Ogras. Wearable IoT Devices for Health Monitoring. *TechConnect Briefs*, 2019:357–360, 2019.

[33] Sizhe An, Ganapati Bhat, Suat Gumussoy, and Umit Ogras. Transfer Learning for Human Activity Recognition using Representational Analysis of Neural Networks. *arXiv Prepr. arXiv2012.04479*, 2020.

[34] Sizhe An, Others, Yigit Tuncel, Toygun Basaklar, Gokul K Krishnakumar, Ganapati Bhat, and Umit Y Ogras. Mgait: Model-based gait analysis using wearable bend and inertial sensors. *ACM Trans. Internet Things*, 3(1):1–24, 2021.

[35] Zhen Meng, Song Fu, Jie Yan, Hongyuan Liang, Anfu Zhou, Shilin Zhu, Huadong Ma, Jianhua Liu, Ning Yang, and Others. Gait recognition for co-existing multiple people using millimeter wave sensing. In *Proc. AAAI Conf. Artif. Intell.*, volume 34, pages 849–856, 2020.

[36] Sizhe An and Umit Y Ogras. MARS: mmWave-based Assistive Rehabilitation System for Smart Healthcare. *ACM Trans. Embed. Comput. Syst.*, 20(5s):1–22, 2021.

[37] Sizhe An and Umit Y Ogras. Fast and Scalable Human Pose Estimation Using MmWave Point Cloud. In *Proc. 59th ACM/IEEE Des. Autom. Conf.*, pages 889–894, 2022.

[38] E Dorsey, Todd Sherer, Michael S Okun, and Bastiaan R Bloem. The emerging evidence of the parkinson pandemic. *Journal of Parkinson's disease*, 8(s1):S3–S8, 2018.

[39] Kyoung Bo Lee and Others. Six-month functional recovery of stroke patients: a multi-time-point study. *Intl. J. Rehabil. Res. Int. Zeitschrift fur Rehabil. Rev. Int. Rech. Readapt.*, 38(2):173, 2015.

[40] Giovanni Abbruzzese, Roberta Marchese, Laura Avanzino, and Elisa Pelosin. Rehabilitation for Parkinson's disease: Current outlook and future challenges. *Park. & Relat. Disord.*, 22:S60—-S64, 2016.

[41] Oscar D Lara, Miguel A Labrador, and Others. A Survey on Human Activity Recognition Using Wearable Sensors. *IEEE Commun. Surv. & Tut.*, 15(3):1192–1209, 2013.

[42] Yu Deng, Jiaolong Yang, Jianfeng Xiang, and Xin Tong. GRAM: Generative Radiance Manifolds for 3D-Aware Image Generation. *CVPR*, i:10663–10673, 2022.

[43] Sizhe An, Yin Li, and Umit Ogras. mri: Multi-modal 3d human pose estimation dataset using mmwave, rgb-d, and inertial sensors. *Advances in Neural Information Processing Systems*, 35:27414–27426, 2022.

[44] Sizhe An, Hongyi Xu, Yichun Shi, Guoxian Song, Umit Y Ogras, and Linjie Luo. Panohead: Geometry-aware 3d full-head synthesis in 360deg. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 20950–20959, 2023.

[45] Ganapati Bhat, Yigit Tuncel, Sizhe An, Hyung Gyu Lee, and Umit Y Ogras. An ultra-low energy human activity recognition accelerator for wearable health applications. *ACM Trans. Embed. Comput. Syst.*, 18(5s):1–22, 2019.

[46] Toygun Basaklar, Yigit Tuncel, and Umit Y Ogras. tinyMAN: Lightweight Energy Manager using Reinforcement Learning for Energy Harvesting Wearable IoT Devices. *arXiv Prepr. arXiv2202.09297*, 2022.

[47] Yigit Tuncel, Shiva Bandyopadhyay, Shambhavi V Kulshrestha, Audrey Mendez, and Umit Y Ogras. Towards wearable piezoelectric energy harvesting: Modeling and experimental validation. In *Proc. ACM/IEEE Int. Symp. Low Power Electron. Des.*, pages 55–60, 2020.

[48] Yigit Tuncel, Ganapati Bhat, Jaehyun Park, and Umit Ogras. ECO: Enabling Energy-Neutral IoT Devices through Runtime Allocation of Harvested Energy. *arXiv Prepr. arXiv2102.13605*, 2021.

[49] Yigit Tuncel, Toygun Basaklar, and Umit Ogras. How much energy can we harvest daily for wearable applications? In *2021 IEEE/ACM Int. Symp. Low Power Electron. Des.*, pages 1–6. IEEE, 2021.

[50] Yigit Tuncel, Sizhe An, Ganapati Bhat, Naga Raja, Hyung Gyu Lee, and Umit Ogras. Voltage-Frequency Domain Optimization for Energy-Neutral Wearable Health Devices. *Sensors*, 20(18):5255, 2020.

[51] Yigit Tuncel, Anish Krishnakumar, Aishwarya Lekshmi Chithra, Younghyun Kim, and Umit Ogras. A domain-specific system-on-chip design for energy efficient wearable edge ai applications. In *Proceedings of the ACM/IEEE International Symposium on Low Power Electronics and Design*, pages 1–6, 2022.

[52] Walter Pirker and Regina Katzenschlager. Gait Disorders in Adults and the Elderly. *Wien. Klin. Wochenschr.*, 129(3-4):81–95, 2017.

[53] Jinrui Zhang, Deyu Zhang, Xiaohui Xu, Fucheng Jia, Yunxin Liu, Xuanzhe Liu, Ju Ren, and Yaoxue Zhang. MobiPose: Real-time multi-person pose estimation on mobile devices. In *Proc. 18th Conf. Embed. Networked Sens. Syst.*, pages 136–149, 2020.

[54] Diane Cook, Kyle D Feuz, and Narayanan C Krishnan. Transfer learning for activity recognition: A survey. *Knowl. Inf. Syst.*, 36(3):537–556, 2013.

[55] Chelsea Finn, Pieter Abbeel, and Sergey Levine. Model-agnostic meta-learning for fast adaptation of deep networks. In *Int. Conf. Mach. Learn.*, pages 1126–1135. PMLR, 2017.

[56] Ilktan Ar and Yusuf Sinan Akgul. A computerized recognition system for the home-based physiotherapy exercises using an RGBD camera. *IEEE Trans. Neural Syst. Rehabil. Eng.*, 22(6):1160–1171, 2014.

[57] Min S H Aung and Others. The automatic detection of chronic pain-related expression: requirements, challenges and the multimodal EmoPain dataset. *IEEE Trans. Affect. Comput.*, 7(4):435–451, 2015.

[58] João Antunes, Alexandre Bernardino, Asim Smailagic, and Daniel P Siewiorek. AHA-3D: A Labelled Dataset for Senior Fitness Exercise Recognition and Segmentation from 3D Skeletal Data. In *Prof. Br. Mach. Vis. Conf.*, page 332, 2018.

[59] Daniel Leightley, John Darby, Baihua Li, Jamie S McPhee, and Moi Hoon Yap. Human activity recognition for physical rehabilitation. In *2013 IEEE Int. Conf. Syst. Man, Cybern.*, pages 261–266. IEEE, 2013.

[60] Aleksandar Vakanski, Hyung-pil Jun, David Paul, and Russell Baker. A data set of human body movements for physical rehabilitation exercises. *Data*, 3(1):2, 2018.

[61] Ling Shao, Jungong Han, Dong Xu, and Jamie Shotton. Computer vision for RGB-D sensors: Kinect and its applications [special issue intro]. *IEEE Trans. Cybern.*, 43(5):1314–1317, 2013.

[62] Daniel Roetenberg, Henk Luinge, and Per Slycke. Xsens MVN: Full 6DOF human motion tracking using miniature inertial sensors. *Xsens Motion Technol. BV, Tech. Rep*, 1:1–7, 2009.

[63] Lei Wang, Yun Sun, Qingguo Li, and Tao Liu. Estimation of Step Length and Gait Asymmetry Using Wearable Inertial Sensors. *IEEE Sens. J.*, 18(9):3844–3851, jan 2018.

[64] Akash Deep Singh, Sandeep Singh Sandha, Luis Garcia, and Mani Srivastava. Radhar: Human activity recognition from point clouds generated through a millimeter-wave radar. In *Proc. 3rd ACM Work. Millimeter-wave Networks Sens. Syst.*, pages 51–56, 2019.

[65] Filip Lemic and Others. Localization as a feature of mmWave communication. In *Proc. Intl. Wirel. Commun. Mob. Comput. Conf.*, pages 1033–1038, 2016.

[66] Shigeki Sugimoto, Hayato Tateda, Hidekazu Takahashi, and Masatoshi Okutomi. Obstacle detection using millimeter-wave radar and its visualization on image sequence. In *Proc. 17th Int. Conf. Pattern Recognition, 2004. ICPR 2004.*, volume 3, pages 342–345. IEEE, 2004.

[67] Abhijat Biswas and Shubham Agrawal. First-order Meta-Learned Initialization for Faster Adaptation in Deep Reinforcement Learning. Nips, 2018.

[68] Alex Nichol, Joshua Achiam, and John Schulman. On first-order meta-learning algorithms. *arXiv Prepr. arXiv1803.02999*, 2018.

[69] Jürgen Schmidhuber. *Evolutionary principles in self-referential learning, or on learning how to learn: the meta-meta-... hook*. PhD thesis, Technische Universität München, 1987.

[70] Rishi Puri, Avideh Zakhor, and Raul Puri. Few Shot Learning For Point Cloud Data Using Model Agnostic Meta Learning. In *2020 IEEE Int. Conf. Image Process.*, pages 1906–1910. IEEE, 2020.

[71] Xudong Li, Li Feng, Lei Li, and Chen Wang. Few-Shot Meta-Learning on Point Cloud for Semantic Segmentation. *arXiv Prepr. arXiv2104.02979*, 2021.

[72] Sinno Jialin Pan and Qiang Yang. A Survey on Transfer Learning. *IEEE Trans. Knowl. Data Eng.*, 22(10):1345–1359, 2010.

[73] Anton Schwaighofer, Volker Tresp, and Kai Yu. Learning Gaussian process kernels via hierarchical Bayes. In *Adv. Neural Inf. Process. Syst.*, pages 1209–1216, 2005.

[74] Theodoros Evgeniou and Massimiliano Pontil. Regularized multi–task learning. In *Proc. tenth ACM SIGKDD Int. Conf. Knowl. Discov. data Min.*, pages 109–117. ACM, 2004.

[75] Siwei Feng and Marco F Duarte. Few-shot learning-based human activity recognition. *Expert Syst. Appl.*, 138:112782, 2019.

[76] Rajat Raina, Alexis Battle, Honglak Lee, Benjamin Packer, and Andrew Y Ng. Self-taught learning: transfer learning from unlabeled data. In *Proc. 24th Int. Conf. Mach. Learn.*, pages 759–766. ACM, 2007.

[77] Wenyuan Dai, Qiang Yang, Gui-Rong Xue, and Yong Yu. Boosting for Transfer Learning. In *Proc. 24th Int. Conf. Mach. Learn.*, ICML '07, pages 193–200, New York, NY, USA, 2007. ACM.

[78] Hoo-Chang Shin, Holger R Roth, Mingchen Gao, Le Lu, Ziyue Xu, Isabella Nogues, Jianhua Yao, Daniel Mollura, and Ronald M Summers. Deep convolutional neural networks for computer-aided detection: CNN architectures, dataset characteristics and transfer learning. *IEEE Trans. Med. Imaging*, 35(5):1285–1298, 2016.

[79] Milad Salem, Shayan Taheri, and Jiann-Shiun Yuan. ECG Arrhythmia Classification Using Transfer Learning from 2-Dimensional Deep CNN Features. In *2018 IEEE Biomed. Circuits Syst. Conf.*, pages 1–4. IEEE, 2018.

[80] Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. Imagenet: A large-scale hierarchical image database. In *2009 IEEE Conf. Comput. Vis. pattern Recognit.*, pages 248–255. Ieee, 2009.

[81] Ali Akbari and Roozbeh Jafari. Transferring activity recognition models for new wearable sensors with deep generative domain adaptation. In *Proc. 18th Int. Conf. Inf. Process. Sens. Networks*, pages 85–96. ACM, 2019.

[82] Ulf Blanke and Bernt Schiele. Remember and transfer what you have learned-recognizing composite activities based on activity spotting. In *Int. Symp. Wearable Comput. 2010*, pages 1–8. IEEE, 2010.

[83] Ziyuan Zhao, Kaixin Xu, Shumeng Li, Zeng Zeng, and Cuntai Guan. Mt-uda: Towards unsupervised cross-modality medical image segmentation with

limited source labels. In *Int. Conf. Med. Image Comput. Comput. Interv.*, pages 293–303. Springer, 2021.

[84] Hirotaka Hachiya, Masashi Sugiyama, and Naonori Ueda. Importance-weighted least-squares probabilistic classifier for covariate shift adaptation with application to human activity recognition. *Neurocomputing*, 80:93–101, 2012.

[85] Maxime Oquab, Leon Bottou, Ivan Laptev, and Josef Sivic. Learning and transferring mid-level image representations using convolutional neural networks. In *Proc. IEEE Conf. Comput. Vis. pattern Recognit.*, pages 1717–1724, 2014.

[86] Andreas Argyriou, Theodoros Evgeniou, and Massimiliano Pontil. Multi-task feature learning. In *Adv. Neural Inf. Process. Syst.*, pages 41–48, 2007.

[87] John Blitzer, Ryan McDonald, and Fernando Pereira. Domain adaptation with structural correspondence learning. In *Proc. 2006 Conf. Empir. methods Nat. Lang. Process.*, pages 120–128. Association for Computational Linguistics, 2006.

[88] Jason Yosinski, Jeff Clune, Yoshua Bengio, and Hod Lipson. How transferable are features in deep neural networks? In *Adv. Neural Inf. Process. Syst.*, pages 3320–3328, 2014.

[89] Ari Morcos, Maithra Raghu, and Samy Bengio. Insights on representational similarity in neural networks with canonical correlation. In *Adv. Neural Inf. Process. Syst.*, pages 5732–5741, 2018.

[90] Maithra Raghu, Chiyuan Zhang, Jon Kleinberg, and Samy Bengio. Transfusion: Understanding transfer learning for medical imaging. In *Adv. Neural Inf. Process. Syst.*, pages 3342–3352, 2019.

[91] Ariadna Quattoni, Michael Collins, and Trevor Darrell. Transfer learning for image classification with sparse prototype representations. In *2008 IEEE Conf. Comput. Vis. Pattern Recognit.*, pages 1–8. IEEE, 2008.

[92] Ziyuan Zhao, Jinxuan Hu, Zeng Zeng, Xulei Yang, Peisheng Qian, Bharadwaj Veeravalli, and Cuntai Guan. MMGL: Multi-Scale Multi-View Global-Local Contrastive Learning for Semi-Supervised Cardiac Image Segmentation. In *2022 IEEE Int. Conf. Image Process.*, pages 401–405. IEEE, 2022.

[93] Renjie Ding, Xue Li, Lanshun Nie, Jiazhen Li, Xiandong Si, Dianhui Chu, Guozhong Liu, and Dechen Zhan. Empirical Study and Improvement on Deep Transfer Learning for Human Activity Recognition. *Sensors*, 19(1):57, 2019.

[94] Seyed Ali Rokni, Marjan Nourollahi, and Hassan Ghasemzadeh. Personalized human activity recognition using convolutional neural networks. In *Thirty-Second AAAI Conf. Artif. Intell.*, 2018.

[95] Belkacem Chikhaoui, Frank Gouineau, and Martin Sotir. A CNN based transfer learning model for automatic activity recognition from accelerometer sensors. In *Int. Conf. Mach. Learn. Data Min. Pattern Recognit.*, pages 302–315. Springer, 2018.

[96] Lorraine J Phillips and Others. Using Embedded Sensors in Independent Living to Predict Gait Changes and Falls. *West. J. Nurs. Res.*, 39(1):78–94, 2016.

[97] Maria Laura Ferster, Sinziana Mazilu, and Gerhard Tröster. Gait parameters change prior to freezing in Parkinson's disease: a data-driven study with wearable inertial units. In *Proc. 10th EAI Int. Conf. Body Area Networks*, pages 159–166, 2015.

[98] Antoine Piau, Nora Mattek, Rachel Crissey, Zachary Beattie, Hiroko Dodge, and Jeffrey Kaye. When will my patient fall? Sensor-based in-home walking

speed identifies future falls in older adults. *Journals Gerontol. Ser. A*, 75(5):968–973, 2020.

[99] Boyd Anderson and Others. Mobile Gait Analysis Using Foot-Mounted UWB Sensors. *Proc. ACM IMWUT*, 3(3):1–22, sep 2019.

[100] Xiaoxu Wu, Yan Wang, and Gregory Pottie. A robust step length estimation system for human gait using motion sensors. In *Proc. Conf. Wirel. Heal.*, pages 1–5, 2015.

[101] Daehyun Kim, Hojin Ju, and Chan Gook Park. Comparison of Step Length Estimation Models Using Inertial Sensor on Pelvis. *ICA-SYMP*, 2019.

[102] Lucia Pepa, Giacomo Marangoni, Matteo Di Nicola, Lucio Ciabattoni, Federica Verdini, Luca Spalazzi, and Sauro Longhi. Real time step length estimation on smartphone. In *2016 IEEE Int. Conf. Consum. Electron.*, pages 315–316. IEEE, 2016.

[103] Chandra Tjhai and Kyle O'Keefe. Using step size and lower limb segment orientation from multiple low-cost wearable inertial/magnetic sensors for pedestrian navigation. *Sensors*, 19(14):3140, 2019.

[104] CIR Systems, Inc. GAITRite®, 2019.

[105] Walter Maetzler, Jochen Klucken, and Malcolm Horne. A Clinical View on the Development of Technology-Based Tools in Managing Parkinson's Disease. *Mov. Disord.*, 31(9):1263–1271, jul 2016.

[106] Dustin A Heldman and Others. Telehealth Management of Parkinson's Disease Using Wearable Sensors: An Exploratory Study. *Digit. biomarkers*, 1(1):43–51, 2017.

[107] Judit Bort-Roig, Nicholas D Gilson, Anna Puig-Ribera, Ruth S Contreras, and Stewart G Trost. Measuring and Influencing Physical Activity With Smartphone Technology: A Systematic Review. *Sport. Med.*, 44(5):671–686, 2014.

[108] Jean-Francois Daneault. Could Wearable and Mobile Technology Improve the Management of Essential Tremor? *Front. Neurol.*, 9:257:1—-257:8, 2018.

[109] Alberto J Espay and Others. Technology in Parkinson's Disease: Challenges and Opportunities. *Mov. Disord.*, 31(9):1272–1282, 2016.

[110] Ganapati Bhat, Ranadeep Deb, and Umit Y Ogras. OpenHealth: Open Source Platform for Wearable Health Monitoring. *IEEE Des. & Test*, 36(5):27–34, 2019.

[111] Johannes C M Schlachetzki and Others. Wearable sensors objectively measure gait parameters in Parkinson's disease. *PLoS One*, 12(10), 2017.

[112] Gelan Yang, Wei Tan, Huixia Jin, Tuo Zhao, and Li Tu. Review wearable sensing system for gait recognition. *Cluster Comput.*, 22(2):3021–3029, 2019.

[113] Ranadeep Deb, Ganapati Bhat, Sizhe An, Holly Shill, and Umit Y Ogras. Trends in technology usage for parkinson's disease assessment: A systematic review. *MedRxiv*, 2021.

[114] Muhammad Shoaib, Others, Stephan Bosch, Ozlem Durmaz Incel, Hans Scholten, and Paul J M Havinga. A Survey of Online Activity Recognition Using Mobile Phones. *Sensors*, 15(1):2059–2085, 2015.

[115] Jindong Wang, Yiqiang Chen, Shuji Hao, Xiaohui Peng, and Lisha Hu. Deep learning for sensor-based activity recognition: A survey. *Pattern Recognit. Lett.*, 119:3–11, 2019.

[116] Attila Reiss and Didier Stricker. Creating and benchmarking a new dataset for physical activity monitoring. In *Proc. 5th Int. Conf. PErvasive Technol. Relat. to Assist. Environ.*, pages 1–8, 2012.

[117] CAMILLE SIMON-AL-ARAJI. Bringing AI to the NBA, 2019.

[118] Elisha Odemakinde. Human pose estimation with deep learning - ultimate overview in 2021, sep 2021.

[119] Saeed Ghorbani, Kimia Mahdaviani, Anne Thaler, Konrad Kording, Douglas James Cook, Gunnar Blohm, and Nikolaus F Troje. MoVi: A large multi-purpose human motion and video dataset. *PLoS One*, 16(6):e0253157, 2021.

[120] Norhafizan Ahmad, Raja Ariffin Raja Ghazilla, Nazirah M Khairi, and Vijayabaskar Kasi. Reviews on various inertial measurement unit (IMU) sensor applications. *Int. J. Signal Process. Syst.*, 1(2):256–262, 2013.

[121] Microsoft. Kinect sensor. https://developer.microsoft.com/en-us/windows/kinect/ accessed 29 Sep. 2020, 2014.

[122] Julieta Martinez, Rayat Hossain, Javier Romero, and James J Little. A simple yet effective baseline for 3d human pose estimation. In *Proc. IEEE Int. Conf. Comput. Vis.*, pages 2640–2649, 2017.

[123] Dario Pavllo, Christoph Feichtenhofer, David Grangier, and Michael Auli. 3D human pose estimation in video with temporal convolutions and semi-supervised training. In *Conf. Comput. Vis. Pattern Recognit.*, 2019.

[124] Jun Liu, Amir Shahroudy, Mauricio Perez, Gang Wang, Ling-Yu Duan, and Alex C Kot. NTU RGB+D 120: A large-scale benchmark for 3D human activity understanding. *IEEE Trans. Pattern Anal. Mach. Intell.*, 42(10):2684–2701, 2020.

[125] Yin Ling and Heng Wang. Unsupervised human activity segmentation applying smartphone sensor for healthcare. In *2015 IEEE 12th Intl Conf Ubiquitous Intell. Comput. 2015 IEEE 12th Intl Conf Auton. Trust. Comput. 2015 IEEE 15th Intl Conf Scalable Comput. Commun. Its Assoc. Work.*, pages 1730–1734. IEEE, 2015.

[126] Henry Friday Nweke, Ying Wah Teh, Ghulam Mujtaba, and Mohammed Ali Al-Garadi. Data fusion and multiple classifier systems for human activity detection and health monitoring: Review and open research directions. *Inf. Fusion*, 46:147–170, 2019.

[127] Marcus Rohrbach, Sikandar Amin, Mykhaylo Andriluka, and Bernt Schiele. A database for fine grained activity detection of cooking activities. In *2012 IEEE Conf. Comput. Vis. pattern Recognit.*, pages 1194–1201. IEEE, 2012.

[128] Bernard Ghanem Fabian Caba Heilbron Victor Escorcia and Juan Carlos Niebles. ActivityNet: A Large-Scale Video Benchmark for Human Activity Understanding. In *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, pages 961–970, 2015.

[129] Tangfei Tao, Xingyu Yang, Jiayu Xu, Wei Wang, Sicong Zhang, Ming Li, and Guanghua Xu. Trajectory planning of upper limb rehabilitation robot based on human pose estimation. In *2020 17th Int. Conf. Ubiquitous Robot.*, pages 333–338. IEEE, 2020.

[130] Ying Li, Chenxi Wang, Yu Cao, Benyuan Liu, Joanna Tan, and Yan Luo. Human pose estimation based in-home lower body rehabilitation system. In *2020 Int. Jt. Conf. Neural Networks*, pages 1–8. IEEE, 2020.

[131] Eric R Chan, Marco Monteiro, Petr Kellnhofer, Jiajun Wu, and Gordon Wetzstein. pi-gan: Periodic implicit generative adversarial networks for 3d-aware image synthesis. In *CVPR*, 2021.

[132] Michael Niemeyer and Andreas Geiger. Giraffe: Representing scenes as compositional generative neural feature fields. In *CVPR*, 2021.

[133] Ivan Skorokhodov, Sergey Tulyakov, Yiqun Wang, and Peter Wonka. EpiGRAF: Rethinking training of 3D GANs. *arXiv Prepr. arXiv2206.10535*, 2022.

[134] Ian Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. Generative Adversarial Nets. In *Adv. Neural Inf. Process. Syst.*, pages 2672–2680, 2014.

[135] Tero Karras, Samuli Laine, and Timo Aila. A Style-Based Generator Architecture for Generative Adversarial Networks. In *CVPR*, volume 43, pages 4217–4228, 2019.

[136] Tero Karras, Samuli Laine, Miika Aittala, Janne Hellsten, Jaakko Lehtinen, and Timo Aila. Analyzing and improving the image quality of stylegan. In *CVPR*, 2020.

[137] Tero Karras, Miika Aittala, Samuli Laine, Erik Härkönen, Janne Hellsten, Jaakko Lehtinen, and Timo Aila. Alias-free generative adversarial networks. *Advances in Neural Information Processing Systems*, 34:852–863, 2021.

[138] Attila Szabo, Givi Meishvili, and Paolo Favaro. Unsupervised Generative 3D Shape Learning from Natural Images. *arXiv*, 2019.

[139] Yichun Shi, Divyansh Aggarwal, and Anil K Jain. Lifting 2d stylegan for 3d-aware face generation. In *cvpr*, pages 6258–6266, 2021.

[140] Jiatao Gu, Lingjie Liu, Peng Wang, and Christian Theobalt. Stylenerf: A style-based 3d-aware generator for high-resolution image synthesis. *CVPR*, 2022.

[141] Lior Yariv, Jiatao Gu, Yoni Kasten, and Yaron Lipman. Volume rendering of neural implicit surfaces. In *NeurIPS*, 2021.

[142] Jun Gao, Tianchang Shen, Zian Wang, Wenzheng Chen, Kangxue Yin, Daiqing Li, Or Litany, Zan Gojcic, and Sanja Fidler. GET3D: A Generative Model of High Quality 3D Textured Shapes Learned from Images. *nips*, 2022.

[143] Atsuhiro Noguchi, Xiao Sun, Stephen Lin, and Tatsuya Harada. Unsupervised learning of efficient geometry-aware neural articulated representations. In *Comput. Vision–ECCV 2022 17th Eur. Conf. Tel Aviv, Isr. Oct. 23–27, 2022, Proceedings, Part XVII*, pages 597–614. Springer, 2022.

[144] Katja Schwarz, Axel Sauer, Michael Niemeyer, Yiyi Liao, and Andreas Geiger. VoxGRAF: Fast 3D-Aware Image Synthesis with Sparse Voxel Grids. *Adv. Neural Inf. Process. Syst.*, 2022.

[145] Jianfeng Zhang, Zihang Jiang, Dingdong Yang, Hongyi Xu, Yichun Shi, Guoxian Song, Zhongcong Xu, Xinchao Wang, and Jiashi Feng. AvatarGen: a 3D Generative Model for Animatable Human Avatars. *arXiv Prepr. arXiv2208.00561*, 2022.

[146] Keqiang Sun, Shangzhe Wu, Zhaoyang Huang, Ning Zhang, Quan Wang, and HongSheng Li. Controllable 3D Face Synthesis with Conditional Generative Occupancy Fields. *arXiv Prepr. arXiv2206.08361*, pages 1–23, 2022.

[147] Yue Wu, Yu Deng, Jiaolong Yang, Fangyun Wei, Qifeng Chen, and Xin Tong. AniFaceGAN: Animatable 3D-Aware Face Image Generation for Video Avatars. *arXiv Prepr. arXiv2210.06465*, 2022.

[148] C Marras and Others. Prevalence of Parkinson's Disease Across North America. *NPJ Park. Dis.*, 4(1):21, 2018.

[149] An-Lun Hsu, Pei-Fang Tang, and Mei-Hwa Jan. Analysis of Impairments Influencing Gait Velocity and Asymmetry of Hemiplegic Patients After Mild to Moderate Stroke. *Arch. Phys. Med. Rehabil.*, 84(8):1185–1193, 2003.

[150] Iván González and Others. Comparison Between Passive Vision-Based System and a Wearable Inertial-Based System for Estimating Temporal Gait Parameters Related to the GAITRite Electronic Walkway. *J. Biomed. Inform.*, 62:210–223, 2016.

[151] Andrew L McDonough, Mitchell Batavia, Fang C Chen, Soonjung Kwon, and James Ziai. The validity and reliability of the GAITRite system's measurements: A preliminary evaluation. *Arch. Phys. Med. Rehabil.*, 82(3):419–425, 2001.

[152] Kate E Webster, Joanne E Wittwer, and Julian A Feller. Validity of the GAITRite®walkway system for the measurement of averaged and individual step parameters of gait. *Gait & posture*, 22(4):317–321, 2005.

[153] James J Carollo and Dennis Matthews. Strategies for Clinical Motion Analysis Based on Functional Decomposition of the Gait Cycle. *Phys. Med. Reh. Clin. N.*, 13(4):949–977, 2002.

[154] Liang Shi, Feng Duan, Yikang Yang, and Zhe Sun. The effect of treadmill walking on gait and upper trunk through linear and nonlinear analysis methods. *Sensors*, 19(9):2204, 2019.

[155] Walter T Higgins. A comparison of complementary and Kalman filtering. *IEEE Trans. Aerosp. Electron. Syst.*, (3):321–325, 1975.

[156] Sebastian O H Madgwick, Andrew J L Harrison, and Ravi Vaidyanathan. Estimation of IMU and MARG Orientation Using a Gradient Descent Algorithm. In *IEEE ICORR*, pages 1–7, 2011.

[157] Galit Yogev and Others. Gait Asymmetry in Patients with Parkinson's Disease and Elderly Fallers: When Does the Bilateral Coordination of Gait Require Attention? *Exp. Brain Res.*, 177(3):336–346, 2006.

[158] Lorenzo Brognara, Emmanuel Navarro-Flores, Lorenzo Iachemet, Nuria Serra-Catalá, and Omar Cauli. Beneficial effect of foot plantar stimulation in gait parameters in individuals with Parkinson's disease. *Brain Sci.*, 10(2):69, 2020.

[159] Bend Labs. Flexible Single Axis Bidirectional Sensor, 2020.

[160] Texas Instruments Inc. CC-2650 Sensortag. [Online] http://www.ti.com/tool/TIDC-CC2650STK-SENSORTAG, accessed 31 July 2019.

[161] T R Fortescue, Lester S Kershenbaum, and B Erik Ydstie. Implementation of self-tuning regulators with variable forgetting factors. *Automatica*, 17(6):831–835, 1981.

[162] Julius Hannink, Malte Ollenschläger, Felix Kluge, Nils Roth, Jochen Klucken, and Bjoern M Eskofier. Benchmarking foot trajectory estimation methods for mobile gait analysis. *Sensors*, 17(9):1940, 2017.

[163] Jaehyun Park et al. Flexible PV-cell Modeling for Energy Harvesting in Wearable IoT Applications. *ACM Trans. Embedd. Comput. Syst.*, 16(5s):156:1–156:20, 2017.

[164] K Czarnecki and Others. Functional movement disorders: successful treatment with a physical therapy rehabilitation protocol. *Parkinsonism Relat. Disord.*, 18(3):247–251, 2012.

[165] Md Atiqur Rahman Ahad, Anindya Das Antar, Omar Shahid, Md Atiqur, Rahman Ahad, Anindya Das Antar, and Omar Shahid. Vision-based Action Understanding for Assistive Healthcare: A Short Review. In *Proc. Conf. Comput. Vis. Pattern Recognit. Work.*, number June, pages 1–11, jun 2019.

[166] Shyamal Patel, Hyung Park, Paolo Bonato, Leighton Chan, and Mary Rodgers. A Review Of Wearable Sensors And Systems With Application In Rehabilitation. *J. Neuroengineering Rehabil.*, 9(1):21, 2012.

[167] Qi Wang, Panos Markopoulos, Bin Yu, Wei Chen, and Annick Timmermans. Interactive wearable systems for upper body rehabilitation: a systematic review. *J. Neuroeng. Rehabil.*, 14(1):1–21, 2017.

[168] Ana Ligia Silva de Lima and Others. Feasibility of Large-Scale Deployment of Multiple Wearable Sensors in Parkinson's Disease. *PLoS One*, 12(12):e0189161, 2017.

[169] Intel. Realsense sensor. https://www.intelrealsense.com/ accessed 29 Sep. 2020, 2014.

[170] Zhe Cao, Tomas Simon, Shih-En Wei, and Yaser Sheikh. Realtime multi-person 2d pose estimation using part affinity fields. In *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, pages 7291–7299, 2017.

[171] Zhicheng Yang, Parth H Pathak, Yunze Zeng, Xixi Liran, and Prasant Mohapatra. Monitoring vital signs using millimeter wave. In *Proc. Intl. Symp. Mob. Ad hoc Netw. Comput.*, pages 211–220, 2016.

[172] Texas Instruments. IWR1443BOOST. https://www.ti.com/tool/IWR1443BOOST accessed 29 Sep. 2020, 2014.

[173] Haipeng Liu, Others, Yuheng Wang, Anfu Zhou, Hanyue He, Wei Wang, Kunpeng Wang, Peilin Pan, Yixuan Lu, Liang Liu, Huadong Ma, and Others. Real-time Arm Gesture Recognition in Smart Home Scenarios via Millimeter Wave Sensing. *Proc. ACM Interactive, Mobile, Wearable Ubiquitous Technol.*, 4(4):1–28, 2020.

[174] Nvidia. Jetson Xavier NX Developer Kit. https://developer.nvidia.com/embedded/jetson-xavier-nx-devkit accessed 29 Sep. 2020, 2014.

[175] Texas Instruments. mmWavefundamentals. https://www.ti.com/lit/spyy005 accessed 8 Apr. 2021, 2020.

[176] Texas Instruments. mmWavetutorial. https://www.ti.com/lit/pdf/swra553 accessed 29 Sep. 2020, 2014.

[177] Sandeep Rao. Introduction to mmWave sensing: FMCW radars. *Texas Instruments mmWave Train. Ser.*, 2017.

[178] Gary Minkler and Jing Minkler. CFAR: the principles of automatic radar detection in clutter. *NASA STI/Recon Tech. Rep. A*, 90:23371, 1990.

[179] M Bondarenko and Vadym I Slyusar. Influence of jitter in ADC on precision of direction-finding by digital antenna arrays. *Radioelectron. Commun. Syst.*, 54(8):436–445, 2011.

[180] Vivek Dham. Programming chirp parameters in ti radar devices. *Appl. Rep. SWRA553, Texas Instruments*, 2017.

[181] Abdi T Abdalla, Mohammad T Alkhodary, and Ali H Muqaibel. Multipath ghosts in through-the-wall radar imaging: challenges and solutions. *ETRI J.*, 40(3):376–388, 2018.

[182] Sizhe An. MARS. https://github.com/SizheAn/MARS accessed 8 Jul. 2021, 2021.

[183] Texas Instruments. Zone Occupancy. https://www.ti.com/lit/pdf/tiduea7 accessed 8 Apr. 2021, 2018.

[184] ACRMDOG Godfrey, Richard Conway, David Meagher, and Gearoid ÓLaighin. Direct measurement of human movement by accelerometry. *Med. Eng. & Phys.*, 30(10):1364–1386, 2008.

[185] Texas Instruments. Datasheet. https://www.ti.com/lit/ds/symlink/iwr1443.pdf accessed 8 Apr. 2021, 2014.

[186] Mart Abadi and Others. TensorFlow: Large-Scale Machine Learning on Heterogeneous Systems. [Online] http://tensorflow.org/, accessed 31 July 2019, 2015.

[187] François Chollet and Others. Keras. https://keras.io, 2015.

[188] Diederik P Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *arXiv Prepr. arXiv1412.6980*, pages 1–13, 2014.

[189] Piotr Szczuko. Deep neural networks for human pose estimation from a very low resolution depth image. *Multimed. Tools Appl.*, 78(20):29357–29377, 2019.

[190] Ganapati Bhat, Jaehyun Park, and Umit Y Ogras. Near-Optimal Energy Allocation for Self-Powered Wearable Systems. In *Proc. Intl. Conf. Comput. Des.*, pages 368–375, 2017.

[191] Radu Marculescu, Diana Marculescu, and Umit Ogras. Edge AI: Systems Design and ML for IoT Data Analytics. In *Proc. ACM SIGKDD Intl. Conf. Knowl. Discov. & Data Min.*, pages 3565–3566, 2020.

[192] Wit-motions. BWT901CL. https://www.wit-motion.com/9-axis/witmotion-bluetooth-2-0-mult.html accessed 8 Apr. 2021, 2021.

[193] Mathworks. Using the Stereo Camera Calibrator App. https://www.mathworks.com/help/vision/ug/using-the-stereo-camera-calibrator-app.html, 2018.

[194] Yinghao Huang, Manuel Kaufmann, Emre Aksan, Michael J Black, Otmar Hilliges, and Gerard Pons-Moll. Deep Inertial Poser: Learning to Reconstruct Human Pose from Sparse Inertial Measurements in Real Time. *ACM Trans. Graph. (Proc. SIGGRAPH Asia)*, 37:185:1–185:15, nov 2018.

[195] Ke Sun, Bin Xiao, Dong Liu, and Jingdong Wang. Deep high-resolution representation learning for human pose estimation. In *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, pages 5693–5703, 2019.

[196] Kaiming He, Georgia Gkioxari, Piotr Dollár, and Ross Girshick. Mask R-CNN. In *Proc. IEEE Intl. Conf. Comput. Vis.*, pages 2961–2969, 2017.

[197] Hennie Brugman, Albert Russel, and Xd Nijmegen. Annotating Multi-media/Multi-modal Resources with ELAN. In *LREC*, pages 2065–2068, 2004.

[198] Kostas Daniilidis. Pose from 3D point correspondences: The procrustes problem - pose estimation, 2022.

[199] Chenlin Zhang, Jianxin Wu, and Yin Li. ActionFormer: Localizing Moments of Actions with Transformers. In *Eur. Conf. Comput. Vis.*, 2022.

[200] TI. IWR1443, 2020.

[201] Li Tao, Xueting Wang, and Toshihiko Yamasaki. Rethinking motion representation: Residual frames with 3d convnets for better action recognition. *arXiv Prepr. arXiv2001.05661*, 2020.

[202] Fuzhen Zhuang, Zhiyuan Qi, Keyu Duan, Dongbo Xi, Yongchun Zhu, Hengshu Zhu, Hui Xiong, and Qing He. A comprehensive survey on transfer learning. *Proc. IEEE*, 109(1):43–76, 2020.

[203] Adam Paszke. Pytorch: Tensors and dynamic neural networks in python with strong GPU acceleration, dec 2019.

[204] Liangqu Long. MAML-Pytorch Implementation. https://github.com/dragen1860/MAML-Pytorch, 2018.

[205] Diederik P Kingma and Jimmy Ba. Adam: A Method for Stochastic Optimization. In Yoshua Bengio and Yann LeCun, editors, *3rd Int. Conf. Learn. Represent. ICLR 2015, San Diego, CA, USA, May 7-9, 2015, Conf. Track Proc.*, 2015.

[206] Ching-Yi Lin and Radu Marculescu. Model personalization for human activity recognition. In *Intl. Conf. Pervasive Comput. Commun. Work. (PerCom Work.*, pages 1–7, 2020.

[207] Davide Anguita, Alessandro Ghio, Luca Oneto, Xavier Parra, and Jorge Luis Reyes-Ortiz. A public domain dataset for human activity recognition using smartphones. In *Esann*, 2013.

[208] Jorge-L Reyes-Ortiz, Luca Oneto, Albert Samà, Xavier Parra, and Davide Anguita. Transition-aware human activity recognition using smartphones. *Neurocomputing*, 171:754–767, 2016.

[209] Jennifer R Kwapisz, Gary M Weiss, and Samuel A Moore. Activity Recognition Using Cell Phone Accelerometers. *ACM SigKDD Explor. Newsl.*, 12(2):74–82, 2011.

[210] Daniela Micucci, Marco Mobilio, and Paolo Napoletano. Unimib shar: A dataset for human activity recognition using acceleration data from smartphones. *Appl. Sci.*, 7(10):1101, 2017.

[211] Jerome Friedman, Trevor Hastie, and Robert Tibshirani. *The Elements of Statistical Learning*, volume 1. Springer, 2001.

[212] Allah Bux Sargano, Xiaofeng Wang, Plamen Angelov, and Zulfiqar Habib. Human action recognition using transfer learning with deep representations. In *2017 Int. Jt. Conf. neural networks*, pages 463–469. IEEE, 2017.

[213] Aiguo Wang, Guilin Chen, Jing Yang, Shenghui Zhao, and Chih-Yung Chang. A Comparative Study on Human Activity Recognition Using Inertial Sensors in a Smartphone. *IEEE Sensors J.*, 16(11):4566–4578, 2016.

[214] Charissa Ann Ronao and Sung-Bae Cho. Human activity recognition with smartphone sensors using deep learning neural networks. *Expert Syst. Appl.*, 59:235–244, 2016.

[215] Taeho Hur, Jaehun Bang, Jongwon Lee, Jee-In Kim, Sungyoung Lee, and Others. Iss2Image: A novel signal-encoding technique for CNN-based human activity recognition. *Sensors*, 18(11):3910, 2018.

[216] Stijn Van Dongen and Anton J Enright. Metric distances derived from cosine similarity and Pearson and Spearman correlations. *arXiv Prepr. arXiv1208.3145*, 2012.

[217] Naomi Saphra and Adam Lopez. Understanding Learning Dynamics Of Language Models with SVCCA. In *Proc. 2019 Conf. North Am. Chapter Assoc. Comput. Linguist. Hum. Lang. Technol. Vol. 1 (Long Short Pap.*, pages 3257–3267, 2019.

[218] Chu-Xiong Qin and Dan Qu. Towards Understanding Attention-Based Speech Recognition Models. *IEEE Access*, 8:24358–24369, 2020.

[219] Matthew D Zeiler. Adadelta: an adaptive learning rate method. *arXiv Prepr. arXiv1212.5701*, 2012.

[220] Volker Blanz and Thomas Vetter. A morphable model for the synthesis of 3D faces. In *Proc. 26th Annu. Conf. Comput. Graph. Interact. Tech.*, pages 187–194, 1999.

[221] Tianye Li, Timo Bolkart, Michael. J Black, Hao Li, and Javier Romero. Learning a model of facial shape and expression from 4D scans. 36(6):194:1—-194:17, 2017.

[222] Ben Mildenhall, Pratul P. Srinivasan, Matthew Tancik, Jonathan T. Barron, Ravi Ramamoorthi, and Ren Ng. Nerf: Representing scenes as neural radiance fields for view synthesis. In *ECCV*, volume 12346 LNCS, pages 405–421, 2020.

[223] Yiheng Xie, Towaki Takikawa, Shunsuke Saito, Or Litany, Shiqin Yan, Numair Khan, Federico Tombari, James Tompkin, Vincent Sitzmann, and Srinath Sridhar. Neural fields in visual computing and beyond. In *Comput. Graph. Forum*, volume 41, pages 641–676. Wiley Online Library, 2022.

[224] Ayush Tewari, Michael Zollhöfer, Pablo Garrido, Florian Bernard, Hyeongwoo Kim, Patrick Pérez, and Christian Theobalt. Self-supervised multi-level face model learning for monocular reconstruction at over 250 hz. In *CVPR*, 2018.

[225] Luan Tran, Feng Liu, and Xiaoming Liu. Towards high-fidelity nonlinear 3D face morphable model. In *CVPR*, 2019.

[226] Yang Hong, Bo Peng, Haiyao Xiao, Ligang Liu, and Juyong Zhang. Headnerf: A real-time nerf-based parametric head model. In *cvpr*, pages 20374–20384, 2022.

[227] Yiyu Zhuang, Hao Zhu, Xusen Sun, and Xun Cao. Mofanerf: Morphable facial neural radiance field. *arXiv Prepr. arXiv2112.02308*, 2021.

[228] Jianzhu Guo, Xiangyu Zhu, Yang Yang, Fan Yang, Zhen Lei, and Stan Z Li. Towards fast, accurate and stable 3d dense face alignment. In *ECCV*, 2020.

[229] Yijun Zhou and James Gregson. Whenet: Real-time fine-grained estimation for wide range head pose. In *BMVC*, 2020.

[230] Marina Ivašić-Kos, Mate Krišto, and Miran Pobar. Human detection in thermal imaging using YOLO. In *Proc. 2019 5th Int. Conf. Comput. Technol. Appl.*, pages 20–24, 2019.

[231] Taewoo Kim, Chaeyeon Chung, Sunghyun Park, Gyojung Gu, Keonmin Nam, Wonzo Choe, Jaesung Lee, and Jaegul Choo. K-hairstyle: A large-scale korean hairstyle dataset for virtual hair editing and hairstyle classification. In *2021 IEEE Int. Conf. Image Process.*, pages 1299–1303. IEEE, 2021.

[232] William E Lorensen and Harvey E Cline. Marching cubes: A high resolution 3D surface construction algorithm. *ACM siggraph Comput. Graph.*, 21(4):163–169, 1987.

[233] Martin Heusel, Hubert Ramsauer, Thomas Unterthiner, Bernhard Nessler, and Sepp Hochreiter. Gans trained by a two time-scale update rule converge to a local nash equilibrium. *NeurIPS*, 2017.

[234] Minchul Kim, Anil K Jain, and Xiaoming Liu. AdaFace: Quality Adaptive Margin for Face Recognition. In *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, pages 18750–18759, 2022.

[235] Liang-Chieh Chen, George Papandreou, Florian Schroff, and Hartwig Adam. Rethinking atrous convolution for semantic image segmentation. *arXiv Prepr. arXiv1706.05587*, 2017.

[236] Tim Salimans, Ian Goodfellow, Wojciech Zaremba, Vicki Cheung, Alec Radford, and Xi Chen. Improved techniques for training gans. *NeurIPS*, 2016.

[237] Richard Zhang, Phillip Isola, Alexei A Efros, Eli Shechtman, and Oliver Wang. The unreasonable effectiveness of deep features as a perceptual metric. In *cvpr*, pages 586–595, 2018.

[238] Daniel Roich, Ron Mokady, Amit H Bermano, and Daniel Cohen-Or. Pivotal Tuning for Latent-based Editing of Real Images. *ACM TOG*, 2021.

[239] Tero Karras, Miika Aittala, Samuli Laine, Erik Härkönen, Janne Hellsten, Jaakko Lehtinen, and Timo Aila. Alias-Free Generative Adversarial Networks. *NeurIPS*, 2021.