SuDA: Support-based Domain Adaptation for Sim2Real Hinge Joint Tracking with Flexible Sensors

Jiawei Fang [1]   **Haishan Song** [1]   **Chengxu Zuo** [1]   **Xiaoxia Gao** [2]   **Xiaowei Chen** [1]   **Shihui Guo** [1]   **Yipeng Qin** [3]

Abstract

Flexible sensors hold promise for human motion capture (MoCap), offering advantages such as wearability, privacy preservation, and minimal constraints on natural movement. However, existing flexible sensor based MoCap methods rely on deep learning and necessitate large and diverse labeled datasets for training. These data typically need to be collected in MoCap studios with specialized equipment and substantial manual labor, making them difficult and expensive to obtain at scale. Thanks to the high-linearity of flexible sensors, we address this challenge by proposing a novel *Sim2Real* solution for hinge joint tracking based on domain adaptation, eliminating the need for labeled data yet achieving comparable accuracy to supervised learning. Our solution relies on a novel *Support-based* Domain Adaptation method, namely **SuDA**, which aligns the *supports* of the predictive functions rather than the instance-dependent distributions between the source and target domains. Extensive experimental results show the effectiveness of our method and its superiority over state-of-the-art distribution-based domain adaptation methods in our task.

1. Introduction

Human motion capture (MoCap) has garnered significant attention for its vast potential in comprehending human intentions and its capacity to control physical devices and systems effectively (Yi et al., 2022; Shen et al., 2023). As an alternative to vision-based and inertial-based solutions, flexible sensors are emerging as they offer the benefits of bio-compatibility, high stretchability, lightweight, and easy
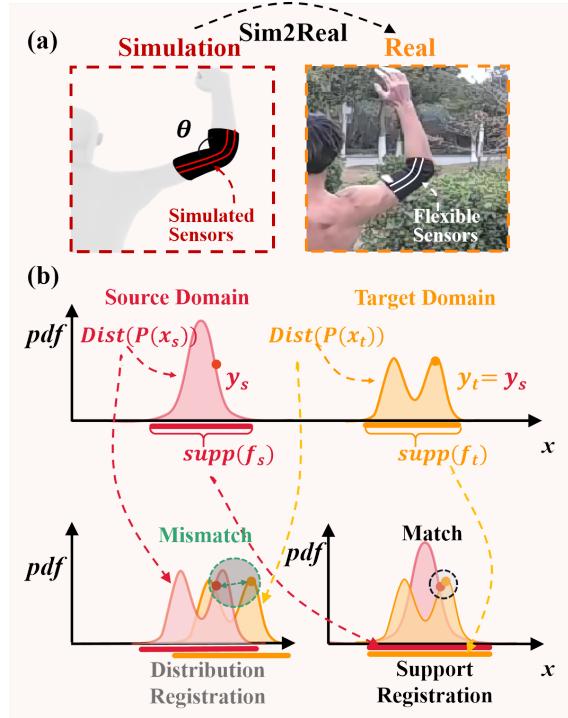

Figure 1. (a) Our Sim2Real approach simulates flexible sensors to predict joint angle in the real world. (b) **Distribution Registration** registers the feature distributions $Dist(P(x_s))$ and $Dist(P(x_t))$ of the source and target domains, which fails to match points of the same label $y_s = y_t$ when the two distributions are inherently different. In contrast, the proposed **Support Registration** registers the function supports $supp(f_s)$ and $supp(f_t)$ that are independent of the specific distributions, ensuring successful domain adaptation.

integration into clothing, enabling long-term monitoring of human physical status (Zazoum et al., 2022; Chen et al., 2023a). Hence, there has been a growing demand to incorporate wearable flexible sensors into MoCap (Luo et al., 2021; Zhang et al., 2022b; Luo, 2023).

Deep neural networks have become the leading methodology for flexible sensor-based MoCap (Mathis et al., 2020; Zuo et al., 2023) due to their strengths in modeling the complex relationships between sensor signals and motions. However, the conflict between data-hungry deep neural networks and the costly, labor-intensive process of collecting paired motion and sensor data poses a challenge in the real-

world application of flexible sensor-based Mocap. Specifically, acquiring such paired data involves setting up motion capture studios equipped with both sensor reading collection equipment and experienced staff, which is expensive and labor-intensive to conduct at scale. A full-fledged optical MoCap setup typically costs between $25K and $500k.

Thanks to the high-linearity of flexible sensors, our paper addresses the conflict mentioned above by proposing a novel *Sim2Real* solution, which eliminates the need for any paired motion and sensor data yet achieves comparable accuracy. Specifically, a *support-based* domain adaptation (SuDA) method is proposed to align the mapping between sensor and motion from the simulated (source) domain to the real-world (target) domain in a simple yet effective way. In contrast to state-of-the-art distribution-based domain adaptation methods (*i.e.*, DiDA) that register the instance-dependent distributions between the source and target domains (Fig. 1), SuDA registers the *supports* of the predictive functions instead, making it independent of distribution variations resulting from body shapes, motions, sensor positions, etc.. Without loss of generality, we empirically evaluate our solution by predicting the hinge joint angle using a smart pad equipped with two flexible sensors and worn on users' joints. Extensive experimental results demonstrate the effectiveness of our method and its superiority over state-of-the-art distribution-based domain adaptation methods in human motion capture.

Our contributions are three-fold:

- We propose a *Sim2Real* solution for human hinge joint tracking using flexible sensors, eliminating the need for an expensive real-world data collection process yet achieving comparable accuracy to supervised learning.

- We propose a novel *support-based* domain adaptation method (SuDA) that registers predictive function supports rather than data distributions, leading to more accurate and generalizable results.

- We conduct extensive experiments covering various application scenarios (different users, motions, wearing positions and joints) to show the effectiveness of our method.

2. Related Work

2.1. Motion Capture with Flexible Sensors

Human motion capture has been actively explored due to its competence in learning profound knowledge about human motion from raw sensor inputs (Wang et al., 2022). In addition to established motion capture systems using the multi-camera matrix (such as Vicon), alternative methods mainly include vision-based (Tian et al., 2023) and IMU-based approaches (Yi et al., 2022; Mollyn et al., 2023).

Flexible sensors have been used for long-term monitoring of human physical status, precisely motion capture (Zhang et al., 2022b; Chen et al., 2023a), human-computer interfaces (Liu et al., 2022; Zhang et al., 2020; Fang et al., 2023), soft robotics (Guo et al., 2023), etc., for their advantages of bio-compatibility, high stretch-ability, lightweight, and ease of integration within clothing (Zazoum et al., 2022).

For MoCap, existing methods have explored the use of flexible sensors in tracking the motion of the full body (Chen et al., 2023b), upper body (Jin et al., 2020a; Zhou et al., 2023), fingers (Glauser et al., 2019), lower limbs (Kwak et al., 2020), elbow joints (Abro et al., 2019; Chen et al., 2023a), and knee joints (Yu et al., 2021).

However, all these methods face critical data scarcity challenges, requiring collection across different users, motion types, and wearing sessions (Chen et al., 2023a; Kang et al., 2022). This typically involves MoCap studios, specialized equipment, and significant manual labor, making large-scale data acquisition difficult and expensive.

2.2. Sim2Real Paradigm

The Sim2Real paradigm trains models on simulated data and then applies them to solve real-world problems, thus successfully alleviating the data scarcity problem in machine learning (Höfer et al., 2021; Nikolenko, 2021). Researchers use this approach to develop cost-effective solutions for MoCap (Huang et al., 2018; Yi et al., 2021; 2022), pose estimation (Lin et al., 2021), pedestrian detection (Fabbri et al., 2021; Stauner et al., 2022), action recognition (Winkler et al., 2022), autonomous driving (Müller et al., 2018; Zhou et al., 2019) and robotics (Abeyruwan et al., 2023; Kataoka et al., 2023; Wu et al., 2023). These works use generative models that make simulation look more like reality (Bousmalis et al., 2017), domain adaptation to mitigate domain discrepancy (Doersch and Zisserman, 2019), or randomize the simulator to increase the distribution overlap (Horváth et al., 2022).

To our knowledge, Park et al. (2020) is the only work that adopted a Sim2Real approach for flexible sensor based Mo-Cap, which simulates the interplay between flexible capacitive sensors and human motions using OpenSim (Delp et al., 2007). However, unlike their use of a distribution-based method to fine-tune models pre-trained with synthetic data using real *labeled* data, our SuDA achieves a high accuracy comparable to supervised learning without the need for any real labeled data and is therefore more cost-effective.

2.3. Domain Adaptation

Domain adaptation (DA) aims to mitigate the gap between the source and target domains so that models trained in the source domain(s) can be applied to the target domain(s).
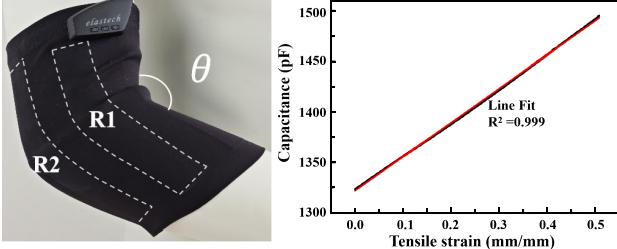
Figure 2. (a) Our hardware. R_1 and R_2 denote the two sensors on our smart pad, and θ denotes the joint angle. (b) High linearity characteristics of capacitive strain sensors.

To date, distribution-based domain adaptation (*DiDA*) has dominated DA, which aims to reduce the distribution discrepancy between two domains (Pan and Yang, 2009). With advent of deep learning, pioneer works showed that deep neural networks could learn more transferable features for domain adaptation (Yosinski et al., 2014). Since then, various solutions have been proposed to align the distributions (or their statistics) of features extracted by deep neural networks in the source and target domains. For example, Deep Domain Confusion (DDC) (Tzeng et al., 2014) first proposed the use of Maximum Mean Discrepancy (MMD) loss to align the feature distributions; Deep Adaptation Networks (DAN) (Gretton et al., 2012) extends the idea to the use of multiple-kernel MMD; Deep CORAL (Sun and Saenko, 2016) proposed CORAL loss to align statistics of feature distributions for domain adaptation. This distribution-based formulation has become a "standard" for state-of-the-art domain adaptation solutions (Ganin and Lempitsky, 2015; Yu et al., 2019; Liu et al., 2021; Deng et al., 2021; Yang et al., 2022; Lee et al., 2022; Garg et al., 2023).

Despite their effectiveness, DiDA methods struggle when there is a significant distribution disparity between the two domains (*a.k.a., Negative Transfer*), which is a long-standing and challenging issue in DA (Rosenstein et al., 2005; Wang et al., 2019; Jin et al., 2020b; Zhang et al., 2022a). Addressing this challenge, we propose a novel *support-based* domain adaption (SuDA) paradigm that aligns the supports rather than distributions between two domains. Thanks to its independence from distribution disparity, SuDA produces more accurate and generalizable motion capture results.

3. Real and Simulation Data Preparation

3.1. Real Data with Flexible Sensors

We develop a prototype by augmenting a standard smart pad with two capacitive strain sensors with high-linearity placed on the elbow's olecranon side. Our method aims to estimate the bending angle, θ, of human joints, denoted as y_t, from the two sensor readings, denoted as $x_t = \{R_1, R_2\}$ (Fig. 2). The high-linearity nature of capacitive strain sensors guaran-

tees their capacitance (*i.e.*, sensor readings) increases with the stretch caused by joint bending (Atalay et al., 2017; Totaro et al., 2017; White et al., 2017; Wang et al., 2017). Please see Appendix A about our hardware.

3.2. Simulation Data with Body-Fabric-Sensor Model

The ultimate goal of our simulation approach is to mimic characteristics of the capacitive strain sensors.

To achieve this, we here construct a body-fabric-sensor model to generate the paired sample between the simulated joint rotation and sensor readings. We used the SMPL model to create fbx files of simulated human 3D models. The entire process of synthesizing digital mannequins through SMPL models can be divided into four major stages as follows.

3.2.1. BODY MODELING

We employ the Skinned Multi-Person Linear (SMPL) (Loper et al., 2015) model to simulate human body and movement. It comprises the following four steps.

- *Template construction*: we first defined a mean template as the foundational pose of the human body and then represent each human pose using parameters that describe its offsets from the fundamental pose.

- *Shape variation*: we specified the shape of the simulated human body using the SMPL shape parameter and obtained a mesh with specific height and weight.

- *Mesh skinning*: We only calculated the mesh in the static pose in the previous two stages. When human skeleton joints move, the "skin" composed of vertexes will change with the movement of the skeleton joints; this process is called skinning.

- *Motion simulation and binding*: To address the effect of different motions, we collected approximately 30,000 frames of MoCap data from the free online animation library Mixamo containing eleven expressive motions (e.g., *Chincken Dance, Robot Dance*).

3.2.2. FABRIC DEFORMATION

To reduce the Sim2Real domain gaps arising from creases of real fabric, we used the cloth database built into the clothing design software Marvelous Designer. Considering the basic parameters such as elasticity, friction, and thickness of real fabric pads and visual comparison inspection of simulation results, we selected hard-jeanet from the database.

3.2.3. SENSOR STRETCHING

The flexible sensors we used exhibit high linearity and repeatability ($R^2 = 0.999$) with low hysteresis (Fig. 2). We

can approximate its readings by its stretched length. Specifically, we compute the changes in the geodesic distance between sensor endpoints in the elbow pad in different frames to measure sensor stretching and its readings. The simulated sensor readings are represented as x_s.

3.2.4. ELBOW BENDING ANGLE

We calculate the elbow angle of the SMPL model created above using the 3D coordinates of its elbow joint in different frames. This is the y_s in the source domain.

4. Support-based Domain Adaptation (SuDA)

The overall objective of SuDA is to minimize the support divergence between the two domains.

4.1. Preliminaries

Definition of Support For a real-valued function $f : X \mapsto \mathbb{R}$ defined on a domain X, we denote the **set-theoretic support** of f as $supp(f)$, *i.e.*, the set of points in X where f is non-zero:

$$supp(f) = \{x \in X : f(x) \neq 0\} \qquad (1)$$

Definition 4.1 (SuDA). Let $\mathcal{D}_s = \{(x_s, y_s) \in (X_s \times Y_s)\}_{s=1}^{N_s}$ be the source domain, $\mathcal{D}_t = \{x_t \in X_t\}_{t=1}^{N_t}$ be the unlabeled target domain, and let f_s and f_t be the motion prediction functions defined on X_s and X_t respectively, SuDA aims to learn an optimized f_t^* by minimizing a given loss function L (*e.g.*, MAE) computed between the prediction $f_t(\hat{x}_t)$, where $\hat{x}_t$ is estimated from x_s, and its corresponding label $\hat{y}_t = y_s$:

$$
\begin{aligned}
f_t^* &= \arg\min_{f_t} \mathbb{E}_{\mathcal{D}_s} L(f_t(\hat{x}_t), \hat{y}_t) \\
&= \arg\min_{f_t} \mathbb{E}_{\mathcal{D}_s} L(f_t(R(x_s)), y_s)
\end{aligned}
\qquad (2)
$$

where the **Support Registration** function R is defined as:

$$R : supp(f_s) \mapsto supp(f_t) \qquad (3)$$

4.2. The Rationale Underpinning SuDA

The effectiveness of SuDA hinges on the assumption that $supp(f_s)$ and $supp(f_t)$ are bijective (*i.e.*, one-to-one correspondence), *i.e.*, $\forall x_s^a, x_s^b \in supp(f_s), \forall x_t^a, x_t^b \in supp(f_t)$:

$$x_s^a \neq x_t^a, \ x_s^b \neq x_t^b$$
$$y^b - y^a = \int_{x_s^a}^{x_s^b} \frac{df_s}{dx_s} dx_s = \int_{x_t^a}^{x_t^b} \frac{df_t}{dx_t} dx_t \qquad (4)$$

Eq. 4 guarantees the the correctness of SuDA, which ensures that y^a and y^b are the labels corresponding to x_s^a, x_t^a and

x_s^b, x_t^b respectively through Support Registration:

$$R : x_s^a \mapsto x_t^a, x_s^b \mapsto x_t^b \qquad (5)$$

Consequently, the success of SuDA is contingent upon the effectiveness of the Support Registration function R.

4.3. Support Registration in Flexible Sensor-Based Hinge Joint Tracking

In our task, y denotes joint angles, $supp(f_s)$ and $supp(f_t)$ denote geodesic distances on simulated motion meshes and flexible sensor readings, respectively. The overall object of Support Registration is to map x_s and x_t with the same label (*i.e.*, $y_s = y_t$) properly.

The intrinsic degree of freedom of the elbow joint $Dim = 1$, therefore we parameterize the curves of $supp(f_s)$ and $supp(f_t)$ with parameter $l \in [0, 1]$ using geometric curve fitting and have:

$$
\begin{aligned}
supp(f_s) &\leftrightarrow g_s(l) \\
supp(f_t) &\leftrightarrow g_t(l)
\end{aligned}
\qquad (6)
$$

where $g_s(l)$ and $g_t(l)$ denote the parametric equations of $supp(f_s)$ and $supp(f_t)$, respectively. After parameterization, we find data with equal parameter change (*i.e.*, $l - 0$) have the same label change (*i.e.*, $y^b - y^a$):

$$\int_{x_s^a}^{x_s^b} \frac{df_s}{dx_s} dx_s = \int_{x_t^a}^{x_t^b} \frac{df_t}{dx_t} dx_t = \int_0^l \frac{df_l}{dl} dl \qquad (7)$$

which ensures the success of SuDA in our flexible sensor-based MoCap task. Please see evidence of Eq. 7 in Appendix C.

To find a proper mapping between corresponding x_s and x_t with the same label, we can define the Support Registration function R for flexible sensor-based motion capture as:

$$R : x_s \mapsto l \mapsto g_t(l) = \hat{x}_t \qquad (8)$$

Then we simplify subsequence calculations by quantizing each of the support curves into $n + 1$ evenly-distributed proxy points:

$$
\begin{aligned}
g_s^Q(l) &: \{p_s^0, p_s^1, ..., p_s^{n-1}, p_s^n\}, \ \text{where } p_s^i = g_s(l * i/n) \\
g_t^Q(l) &: \{p_t^0, p_t^1, ..., p_t^{n-1}, p_t^n\}, \ \text{where } p_t^i = g_t(l * i/n)
\end{aligned}
\qquad (9)
$$

This allows us to define the quantized version of our Support Registration function, denoted as R^Q, by mapping x_s to its nearest p_s^i:

$$R^Q : x_s \mapsto p_s^i \mapsto i \mapsto p_t^i = \hat{x}_t \qquad (10)$$
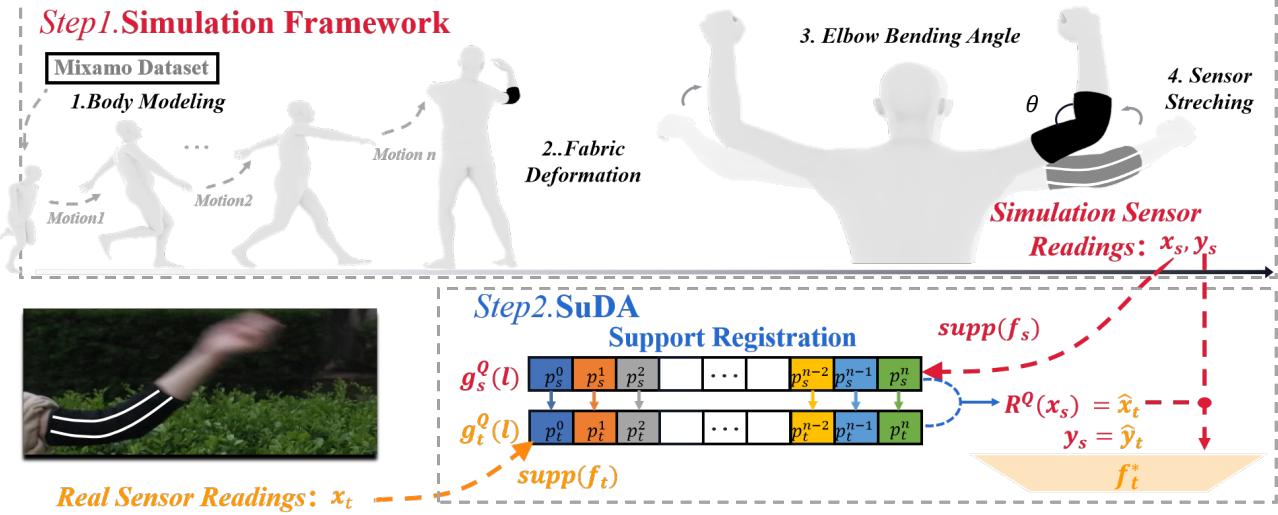
Figure 3. Pipeline Overview. Top row: the generation of simulation data with our Body-Fabric-Sensor model, which consists of four main parts: *1. Body Modeling 2. Fabric Deformation 3. Elbow Bending Angle 4. Sensor Streching.* Bottom row: the proposed *support-based* domain adaptation method (**SuDA**). SuDA first parameterizes the supports of source and target domain, denoted as $supp(f_s)$ and $supp(f_t)$, into $n+1$ evenly-distributed proxy points respectively, denoted as $g_s^Q(l)$ and $g_t^Q(l)$. Then, it applies a novel Support Registration R^Q to map x_s to its nearest p_s^i. Finally, we train the predictive function f_t^* on the target domain using $(\hat{x}_t, \hat{y}_t = y_s)$ and finally to $\hat{x}_t$.

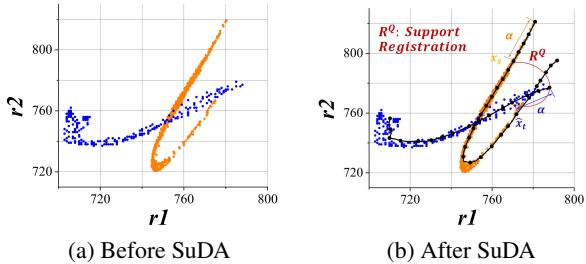We use R^Q in all our experiments. Alg. 1 shows the pseudo-code of our SuDA method.

(a) Before SuDA   (b) After SuDA

Figure 4. **Orange** and **Blue**: source and target data samples. **Black**: parameterized and quantized supports. (a) Two scatter curve tracks are our data samples and represent function support of the two domains. (b) Then, we use curve parameterization to quantize function support into several segments. According to Eq. 7, points with the same parameters have the same labels and can be registered together by R^Q.

5. Experimental Results

5.1. Experimental Setup

5.1.1. IMPLEMENTATION DETAILS

Following common practice, we implement the predictive function in the target domain (f_t^* in Step 4 of Alg. 1) with a neural network consisting of LSTM and MLP layers and train it using an SGD optimizer with a learning rate of $1e^{-3}$ and a weight decay of $5e^{-4}$, momentum= 0.9, batch size= 32, and training epoch $e = 50$. We use a LRscheduler

---

Algorithm 1 Support Based Domain Adaptation (SuDA)

Input: Source domain data $\mathcal{D}_s = \{(x_s, y_s)\}_{s=1}^{N_s}$, target domain data $\mathcal{D}_t = \{x_t \in X_t\}_{t=1}^{N_t}$, loss function L
Output: Prediction function f_t^* on the target domain;
  1: Parameterize $supp(f_s)$ and $supp(f_t)$ with parameter $l \in [0, 1]$ using $\{x_s\}_{s=1}^{N_s}$ and $\{x_t\}_{t=1}^{N_t}$, respectively;
  2: Quantize the parameterized supports (Eq. 9)
  3: Conduct (quantized) support registration R^Q to map x_s to $\hat{x}_t$ with $\hat{y}_t = y_s$ (Eq. 10);
  4: Optimize $f_t^* = \arg\min_{f_t} \mathbb{E}_{\mathcal{D}_s} L(f_t(\hat{x}_t), \hat{y}_t)$ (Eq. 2) using gradient descent (*e.g.*, SGD);
  5: **Return** f_t^*

---

with $\gamma = 0.0003$ and decay = 0.8. We employ an MAE loss as the L used in Alg. 1: $\mathcal{L}_{mae} = \sum_{i=1}^n |y_i - f(x_i)|/n$.

To make a fair comparison, we use the same network architecture, optimization strategy, and loss function when training our Sim2Real method and the two baselines: supervised learning and Real2Real. All experiments were conducted on a desktop PC with an Intel i7-12700KF CPU and an NVIDIA RTX 3090 GPU. The appendix includes details on network architectures(including implementations of methods in the following comparison experiment) and the dataset used in our experiments.

5.1.2. REAL DATASET $\mathcal{D}_{t^*}$

After obtaining the ethical approval, we collected our real dataset $\mathcal{D}_{t^*}^{(id,m,wp)}$ consisting of paired motion and sensor readings, where id denotes user ID, m denotes motion, and wp denotes wearing position, as follows. Specifically, we recruited 11 participants ($id = 1, 2, ..., 11$) and asked them to perform four basic motions ($m = run, walk, jump, clap$) at 12 wearing positions ($wp = 30°, 60°, 90°, ..., 360°$), and collected 371,661 frames of data in total. For each (id, m, wp) instance, we split its data into a training set $\mathcal{D}_{t,train}^{(id,m,wp)}$ and a test set $\mathcal{D}_{t^*,test}^{(id,m,wp)}$ by $0.7 : 0.3$, where we remove the motion labels in the training set to satisfy the unsupervised domain adaptation conditions in real-world application scenarios. Note that we use $\mathcal{D}_{t^*}$ rather than $\mathcal{D}_t$ as we need to collect ground truth motion labels to evaluate our method.

We report the mean and standard deviation of the test accuracy across all $\mathcal{D}_{t^*,test}^{(id,m,wp)}$ in a set of instances specified by different (id, m, wp) over four runs in all our experiments. Unless specified, we evaluated all combinations of (id, m, wp).

5.1.3. SIMULATION DATASET $\mathcal{D}_s$

With the simulation platform introduced above, we built our simulation dataset using around 30,000 motion capture frames involving eleven complex semantic motions (e.g., swing-dancing, chicken-dancing, hybrid motions) gathered from the Mixamo online animation library (Mixamo Animation, 2022).

5.2. Comparison with Supervised Learning

As Table 1 shows, although using *no* real data, the proposed method, denoted as Ours (Sim2Real), achieves a high accuracy comparable to that of supervised learning, demonstrating its effectiveness in flexible sensor-based human motion capture. To further justify our choice of using simulated rather than real data, we compared it to an additional baseline, denoted as Ours (Real2Real), which applies the proposed SuDA to a piece of real data collected with a random user, motion, and wearing position. Experimental results show that Ours (Sim2Real) achieves much higher accuracy than Ours (Real2Real), suggesting that our simulation creates data with more diverse motion features and less noise at a lower cost, which is more desirable in practice.

5.3. Comparison with State-of-the-Art Methods

As Table 2 shows, we compared our method with state-of-the-art (SOTA) distribution-based domain adaptation methods. For a fair comparison, we have adapted the official code provided by the authors to share the same input and output format as ours. Please see the Appendix for more

Table 1. Results of our method (Sim2Real) and two baselines: supervised learning and ours (Real2Real).

Method	MAE (deg)
Ours (Real2Real)	11.91 ± 4.87
Ours (Sim2Real)	7.60 ± 2.58
Supervised	6.42 ± 2.73

Table 2. Comparison of our method (SuDA) with SOTA distribution-based domain adaptation methods.

Method	MAE (deg)
Source Only	21.93 ± 8.83
DAN (Long et al., 2015)	22.38 ± 8.34
DANN (Ganin and Lempitsky, 2015)	21.51 ± 8.03
D-CORAL(Sun and Saenko, 2016)	21.89 ± 8.85
DAAN (Yu et al., 2019)	21.80 ± 8.28
BNM (Cui et al., 2020)	21.70 ± 8.61
Ours (SuDA)	7.60 ± 2.58
Supervised	6.42 ± 2.73

details on implementations. It can be observed that SuDA outperforms all SOTA DiDA methods, which demonstrates its superiority in flexible sensor-based hinge joint tracking.

5.4. Robustness against Different Users, Motions & Wearing Positions

5.4.1. ROBUSTNESS AGAINST DIFFERENT USERS

As Table 3 shows, our method consistently achieves high accuracy comparable to supervised learning and demonstrates significant improvements over competing methods for all 11 users, demonstrating the robustness of our method across different users.

5.4.2. ROBUSTNESS AGAINST DIFFERENT WEARING POSITIONS

Similar to the above, our method achieves high accuracy comparable to supervised learning and demonstrates its superiority over competing methods across different wearing positions (Table 4).

5.4.3. ROBUSTNESS AGAINST DIFFERENT MOTIONS

The same conclusions hold for different motions: Ours is comparable to supervised learning while significantly outperforming competing methods (Table 5).

In addition, we observed that: i) the accuracy of both our SuDA method and supervised learning change with different users, wearing positions and motions, showing that the inherent variations among instances still account for

Table 3. Performance comparison over different users. Source Only: the same predictive neural network trained on the source domain (simulated) data only. We enumerate on id while averaging over all possible values of m and wp of $D_{t*,test}^{id,m,wp}$.

Method	id =1	2	3	4	5	6	7	8	9	10	11
Source Only	17.94±0.34	32.14±1.14	12.94±2.28	24.34±0.67	15.34±0.98	16.62±1.63	21.26±0.69	33.28±5.58	18.34±1.12	14.90±3.37	26.15±1.93
DAN	21.17±3.62	34.43±1.50	12.81±0.18	24.96±1.05	17.43±0.67	18.02±0.73	20.36±0.33	33.95±4.14	16.80±2.48	14.37±3.03	24.37±0.82
DANN	17.62±1.48	31.60±1.24	12.02±0.70	24.37±0.98	14.72±2.30	16.31±1.22	19.65±0.67	33.43±5.69	18.96±1.59	15.33±3.53	24.75±0.39
D-CORAL	17.46±0.87	31.82±1.25	12.23±0.77	25.20±1.12	13.29±2.18	16.21±2.24	21.09±0.69	33.56±3.73	17.11±2.20	15.13±3.16	26.18±1.17
DAAN	18.43±0.35	33.16±1.39	12.53±0.68	25.36±0.71	14.12±1.25	16.40±2.48	21.03±0.20	31.67±5.01	16.46±3.24	15.51±2.62	25.36±0.77
BNM	18.41±0.44	32.14±1.14	12.15±0.53	24.34±0.68	13.20±2.00	16.11±2.10	21.08±0.77	33.53±4.02	16.99±2.02	14.99±3.29	26.53±2.17
SuDA (Ours)	**2.57±0.13**	**2.55±0.09**	**6.72±0.36**	**7.37±0.75**	**5.76±0.08**	**9.65±0.16**	**7.59±0.16**	**7.43±0.29**	**10.16±0.72**	**12.87±3.16**	**14.11±1.31**
Supervised	1.41±0.04	1.96±0.07	3.06±0.13	2.54±0.16	6.51±0.23	7.58±0.31	9.18±0.98	3.33±0.36	7.67±0.38	15.12±1.00	17.01±0.27

Table 4. Performance comparison over different wearing positions. Source Only: the same predictive neural network trained on the source domain (simulated) data only. We enumerate on wp while average over all possible values of id and m of $D_{t*,test}^{id,m,wp}$.

Method	wp = 30°	60°	90°	120°	150°	180°	210°	240°	270°	300°	330°	360°
Source Only	17.94±0.34	13.57±0.67	12.11±1.38	17.21±3.90	21.75±1.98	27.04±1.76	46.70±1.64	24.07±1.45	16.60±0.57	12.96±0.49	19.78±2.15	11.34±0.53
DAN	21.17±3.62	16.24±0.68	12.44±1.08	19.59±2.79	25.44±1.62	25.41±0.49	45.01±3.26	24.04±3.53	16.28±0.67	13.02±0.57	20.34±0.69	19.85±0.91
DANN	17.62±1.48	13.96±0.44	12.30±1.00	19.20±2.10	22.00±2.13	28.15±2.60	42.92±3.64	24.03±0.62	17.04±1.02	12.77±0.22	19.37±1.62	20.24±0.31
D-Coral	17.46±0.87	14.04±1.00	12.51±1.08	19.74±4.73	24.34±2.53	28.50±1.49	46.75±1.38	24.52±0.58	16.65±0.22	13.09±0.25	19.50±1.39	19.95±0.87
DAAN	18.43±0.35	13.76±0.28	12.81±1.45	20.92±2.70	24.76±2.41	26.92±2.19	44.70±3.09	24.85±0.66	15.83±0.93	12.67±0.18	19.52±1.69	19.65±0.44
BNM	18.41±0.44	13.88±0.59	12.10±1.38	17.21±3.90	24.34±2.53	27.47±1.76	46.70±1.64	25.04±1.22	16.37±0.57	12.96±0.49	19.50±1.39	19.89±0.88
SuDA (Ours)	**2.57±0.13**	**5.55±0.36**	**5.41±0.23**	**4.05±0.77**	**9.39±0.14**	**6.79±0.14**	**8.18±0.25**	**7.99±0.31**	**10.59±0.42**	**11.69±0.24**	**13.45±0.19**	**5.83±0.23**
Supervised	1.41±0.04	6.52±0.45	8.05±0.86	2.55±0.11	6.88±0.29	6.25±0.30	8.49±0.39	4.83±0.19	6.61±0.09	7.61±0.13	10.86±0.23	6.59±0.24

Table 5. Performance comparison over different motions. Source Only: the same predictive neural network trained on the source domain (simulated) data only. We enumerate on m while average over all possible values of id and wp of $D_{t*,test}^{id,m,wp}$.

Method	m =Jump	m =Run	m =Clap	m =Walk
Source Only	27.49±0.49	13.76±0.62	36.19±2.40	31.34±3.08
DAN	27.29±1.51	12.79±0.63	37.62±0.70	17.15±1.87
DANN	26.04±0.44	13.50±0.27	37.29±1.53	17.86±0.90
D-Coral	27.79±0.64	13.78±0.63	35.85±1.55	17.72±1.78
DAAN	28.04±0.30	13.86±0.79	36.78±1.91	17.45±2.05
BNM	27.62±0.50	13.63±0.64	36.19±2.40	17.54±1.65
SuDA (Ours)	**7.63±0.07**	**11.00±0.10**	**3.98±0.17**	**8.59±0.23**
Supervised	6.51±0.05	10.55±0.23	4.30±0.13	7.87±0.08

Table 6. Applications in real-world scenarios. The results are averaged over 5 new participants with varying body shapes who wore the elbow pad in three arbitrary positions.

Method	**Table Tennis**	**Basketball**	**Boxing**	**Average**
Source Only	21.53±6.40	19.39±4.52	20.89±8.71	20.61±0.90
DAN	21.60±5.74	18.80±3.81	19.42±6.41	19.94±1.20
DANN	21.73±6.39	18.79±3.82	20.10±8.41	20.47±1.24
D-Coral	22.00±6.04	18.90±4.81	21.14±8.31	20.68±1.31
DAAN	21.40±6.07	17.78±4.34	20.85±8.33	20.01±1.60
BNM	21.74±6.06	19.26±5.08	21.26±8.73	20.76±1.08
SuDA (Ours)	**10.56±2.30**	**11.62±3.53**	**10.02±3.18**	**10.74±0.67**
Supervised	12.28±3.50	11.90±3.06	11.96±3.75	12.05±0.17

Table 7. Applications in four different joints. The results are averaged over 3 new participants with varying body shapes who wore the elbow pad and knee pad.

Method	**Left Elbow**	**Right Elbow**	**Left Knee**	**Right Knee**
Source Only	39.29±3.78	40.85±5.13	43.45±6.28	42.88±5.80
DAN	38.01±4.75	37.98±5.80	41.92±5.81	40.32±6.28
DANN	40.22±3.50	40.89±4.83	42.92±5.35	42.01±6.05
D-Coral	39.23±5.26	41.25±5.91	43.18±5.27	40.14±5.10
DAAN	39.65±3.58	40.59±7.92	42.92±5.35	42.01±6.05
BNM	38.78±4.60	40.76±5.54	42.01±6.46	42.28±5.82
SuDA (Ours)	**9.09±1.89**	**9.86±2.55**	**12.13±2.02**	**10.84±1.84**
Supervised	6.07±0.84	7.50±2.84	8.20±0.40	9.12±3.06

significant differences in performance; ii) in most cases, most state-of-the-art (SOTA) domain adaptation methods work comparably to Source Only, that is, applying models trained on the source domain (simulated) data only, indicating that they failed at our task and did not perform much meaningful adaptation at all. We ascribe their failure to the inherently different distributions between the two domains and extremely low dimensionality of our sensor data.

6. Applications

Performance in real-world scenarios. We recruited five new participants of varying body shapes to investigate how our proposed method works in real-world applications. We asked them to wear our elbow pads in three different arbitrary positions and perform three distinct real-world activities respectively: Table Tennis, Basketball, and Boxing. In total, we collected 45 (5 participants × 3 wearing positions × 3 real-world activities) unique data segments, comprising 81, 848 frames. As Table 6 shows, our method achieves a high accuracy comparable to supervised learning. It significantly outperforms SOTA distribution-based

domain adaptation methods, demonstrating that our SuDA method generalizes well to real-world scenarios.

Generalization across joints. Besides, we recruited another three participants to investigate how SuDA works in different joints. In total, we collected 12 (3 participants × 4 joints) unique data segments, comprising 22, 577 frames. As Table 7 shows, our method achieves a high accuracy comparable to supervised learning. It significantly outperforms SOTA distribution-based domain adaptation methods, demonstrating that our SuDA method generalizes well to different joints. Notice that we only use right elbow simulation data as our source domain.
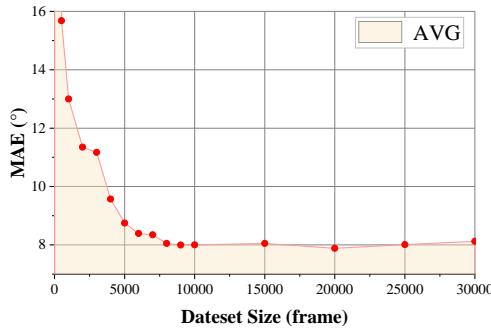
Figure 5. MAE vs. size of simulation dataset. MAE drops quickly between 0 and 5,000 frames with the increase in the size of the simulation dataset. Between 5,000 and 10,000 frames, the MAE drops slightly towards convergence. With more than 10,000 frames, MAE becomes stable against the increase in dataset size.

7. Discussions

7.1. MAE vs. Size of Simulation Dataset

As Fig. 5 shows, MAE decreases quickly when the size of the simulation dataset increases from 0 to 5,000 frames, followed by a slower decline, and eventually converges around 10,000 frames. This indicates that the support of the source domain (simulated) can be well approximated with more than 10,000 motion frames. In all our experiments, we use a safer option of 30,000 frames as the size of our simulation dataset.

7.2. Distribution-based vs. Support-based Domain Adaptation Methods (DiDA vs. SuDA)

Objective   DiDA methods have the same goal as SuDA - aligning the supports of the predictive functions between the source and target domains, *i.e.*, $x_s \leftrightarrow x_t$ so that the predictive function that works on the source domain f_s can also be applied to x_t. However, they take an indirect approach to achieve this goal which aligns the feature distributions of the source and target domains.

Data Distribution   SuDA generally relies on function support yet DiDA relies more on data distribution. In our Sim2Real task, the distribution gap between the two domains (*e.g., real running and simulated dancing, real knee and simulated elbow*) varies remarkably, leading to failure on all DiDA methods. Please see Appendix D about Distribution Variations of our data.

Data Dimension   SuDA is more applicable to low-dimensional data yet DiDA is suitable to high-dimensional one. Low-dimensional data provides significant advantages for directly understanding the function supports. In our task, the extremely low (readings from two flexible sensors) dimension of our data makes us easy to find support registration methods, which motivates this work and leads to success. Despite the success of DiDA methods (*e.g.*, in computer vision), they have a hidden limiting assumption

- that the data being handled must be complex and high-dimensional to provide enough space for extracting lower-dimensional features. This reliance on high-dimensional data restricts the applicability of distribution-based methods to low-dimensional datasets like the one used in this paper.

Hence, the impact of diverse data distributions and low dimensionality of sensor data makes DiDA methods have no advantage over naive source-only methods. Empirically, we verify this in Fig. 6, where the "transfer loss" measuring distributional distances does not reduce during training, yielding similar performance as Source-only.
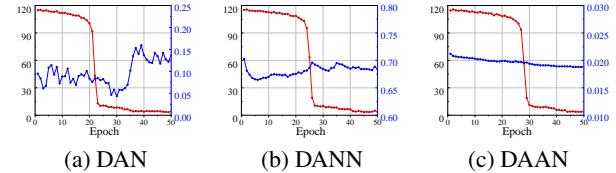

| (a) DAN | (b) DANN | (c) DAAN |

Figure 6. Training loss of distribution-based methods. Red Line: Supervised Loss; Blue Line: Transfer Loss.

7.3. Limitations

Although effective, the proposed method is based on the assumption that function support in the source and target domain can be registered properly. Specifically, for flexible sensor-based MoCap, the source and target data must have similar ranges of joint angles during motions. Observing that the effective range of joint angles is similar for most people performing typical motions (around $40°$ to $160°$), leading to matching to the support scope, so we find that our method works well in most real-world scenarios after calibration. However, this assumption becomes problematic in some extreme cases when the effective range of elbow joint angles differs between domains (please see failure cases in the Appendix F).

8. Conclusion

To address the costly real data collection process in motion capture using flexible sensors, we propose a novel *Sim2Real* solution based on domain adaptation, eliminating the need for labeled data yet achieving comparable accuracy. Our solution relies on a novel *support-based* domain adaptation method that aligns the *supports* of the predictive functions rather than the instance-dependent distributions between the source and target domains. This makes our adaptation method applicable to unpaired simulated and real data with inherently different body shapes, motions, sensor positions, etc., as long as they share similar ranges of joint motion. Experimental results demonstrate that the effectiveness of our unsupervised support-based method is comparable with supervised learning and its superiority over state-of-the-art distribution-based domain adaptation methods. In future work, we plan to explore applying the proposed method to other applications, especially those involving low-dimensional data.

Acknowledgements

Impact Statement

This paper presents work whose goal is to advance the field of Sensor-based Mocap and Domain Adaptation. There are many potential societal consequences of our work, none which we feel must be specifically highlighted here.

References

Saminda Wishwajith Abeyruwan, Laura Graesser, David B D'Ambrosio, Avi Singh, Anish Shankar, Alex Bewley, Deepali Jain, Krzysztof Marcin Choromanski, and Pannag R Sanketi. 2023. i-sim2real: Reinforcement learning of robotic policies in tight human-robot interaction loops. In *Conference on Robot Learning*. PMLR, 212–224.

Zamir Ahmed Abro, Zhang Yi-Fan, Chen Nan-Liang, Hong Cheng-Yu, Rafique Ahmed Lakho, and Habiba Halepoto. 2019. A novel flex sensor-based flexible smart garment for monitoring body postures. *Journal of Industrial Textiles* 49, 2 (2019), 262–274.

Asli Atalay, Vanessa Sanchez, Ozgur Atalay, Daniel M Vogt, Florian Haufe, Robert J Wood, and Conor J Walsh. 2017. Batch fabrication of customizable silicone-textile composite capacitive strain sensors for human motion tracking. *Advanced Materials Technologies* 2, 9 (2017), 1700136.

Konstantinos Bousmalis, Nathan Silberman, David Dohan, Dumitru Erhan, and Dilip Krishnan. 2017. Unsupervised pixel-level domain adaptation with generative adversarial networks. In *Proceedings of the IEEE conference on computer vision and pattern recognition*. 3722–3731.

Xiaowei Chen, Xiao Jiang, Jiawei Fang, Shihui Guo, Juncong Lin, Minghong Liao, Guoliang Luo, and Hongbo Fu. 2023a. DisPad: Flexible On-Body Displacement of Fabric Sensors for Robust Joint-Motion Tracking. *arXiv preprint arXiv:2301.06249* (2023).

Xiaowei Chen, Xiao Jiang, Lishuang Zhan, Shihui Guo, Qunsheng Ruan, Guoliang Luo, Minghong Liao, and Yipeng Qin. 2023b. Full-body human motion reconstruction with sparse joint tracking using flexible sensors. *ACM Transactions on Multimedia Computing, Communications and Applications* 20, 2 (2023), 1–19.

Shuhao Cui, Shuhui Wang, Junbao Zhuo, Liang Li, Qingming Huang, and Qi Tian. 2020. Towards discriminability and diversity: Batch nuclear-norm maximization under label insufficient situations. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 3941–3950.

Scott L Delp, Frank C Anderson, Allison S Arnold, Peter Loan, Ayman Habib, Chand T John, Eran Guendelman, and Darryl G Thelen. 2007. OpenSim: open-source software to create and analyze dynamic simulations of movement. *IEEE transactions on biomedical engineering* 54, 11 (2007), 1940–1950.

Zhun Deng, Linjun Zhang, Kailas Vodrahalli, Kenji Kawaguchi, and James Y Zou. 2021. Adversarial training helps transfer learning via better representations. *Advances in Neural Information Processing Systems* 34 (2021), 25179–25191.

Carl Doersch and Andrew Zisserman. 2019. Sim2real transfer learning for 3d human pose estimation: motion to the rescue. *Advances in Neural Information Processing Systems* 32 (2019).

Matteo Fabbri, Guillem Brasó, Gianluca Maugeri, Orcun Cetintas, Riccardo Gasparini, Aljoša Ošep, Simone Calderara, Laura Leal-Taixé, and Rita Cucchiara. 2021. Motsynth: How can synthetic data help pedestrian detection and tracking?. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*. 10849–10859.

Fengyi Fang, Hongwei Zhang, Lishuang Zhan, Shihui Guo, Minying Zhang, Juncong Lin, Yipeng Qin, and Hongbo Fu. 2023. Handwriting velcro: Endowing AR glasses with personalized and posture-adaptive text input using flexible touch sensor. *Proceedings of the ACM on Interactive, Mobile, Wearable and Ubiquitous Technologies* 6, 4 (2023), 1–31.

Yaroslav Ganin and Victor Lempitsky. 2015. Unsupervised domain adaptation by backpropagation. In *International conference on machine learning*. PMLR, 1180–1189.

Saurabh Garg, Nick Erickson, James Sharpnack, Alex Smola, Sivaraman Balakrishnan, and Zachary Chase Lipton. 2023. Rlsbench: Domain adaptation under relaxed label shift. In *International Conference on Machine Learning*. PMLR, 10879–10928.

Oliver Glauser, Shihao Wu, Daniele Panozzo, Otmar Hilliges, and Olga Sorkine-Hornung. 2019. Interactive hand pose estimation using a stretch-sensing soft glove. *ACM Transactions on Graphics (ToG)* 38, 4 (2019), 1–15.

Arthur Gretton, Dino Sejdinovic, Heiko Strathmann, Sivaraman Balakrishnan, Massimiliano Pontil, Kenji Fukumizu, and Bharath K Sriperumbudur. 2012. Optimal kernel choice for large-scale two-sample tests. *Advances in neural information processing systems* 25 (2012).

Shihui Guo, Lishuang Zhan, Yancheng Cao, Chen Zheng, Guyue Zhou, and Jiangtao Gong. 2023. Touch-and-Heal: Data-driven Affective Computing in Tactile Interaction with Robotic Dog. *Proceedings of the ACM on Interactive, Mobile, Wearable and Ubiquitous Technologies* 7, 2 (2023), 1–33.

Sebastian Höfer, Kostas Bekris, Ankur Handa, Juan Camilo Gamboa, Melissa Mozifian, Florian Golemo, Chris Atkeson, Dieter Fox, Ken Goldberg, John Leonard, et al. 2021. Sim2Real in robotics and automation: Applications and challenges. *IEEE transactions on automation science and engineering* 18, 2 (2021), 398–400.

Dániel Horváth, Gábor Erdős, Zoltán Istenes, Tomáš Horváth, and Sándor Földi. 2022. Object detection using sim2real domain randomization for robotic applications. *IEEE Transactions on Robotics* (2022).

Yinghao Huang, Manuel Kaufmann, Emre Aksan, Michael J Black, Otmar Hilliges, and Gerard Pons-Moll. 2018. Deep inertial poser: Learning to reconstruct human pose from sparse inertial measurements in real time. *ACM Transactions on Graphics (TOG)* 37, 6 (2018), 1–15.

Ying Jin, Guoning Chen, Kete Lao, Songhui Li, Yong Lu, Yufeng Gan, Zhundong Li, Jiajia Hu, Jingcheng Huang, Jinxiu Wen, et al. 2020a. Identifying human body states by using a flexible integrated sensor. *npj Flexible Electronics* 4, 1 (2020), 1–8.

Ying Jin, Ximei Wang, Mingsheng Long, and Jianmin Wang. 2020b. Minimum class confusion for versatile domain adaptation. In *Computer Vision–ECCV 2020: 16th European Conference, Glasgow, UK, August 23–28, 2020, Proceedings, Part XXI 16*. Springer, 464–480.

Beibei Kang, Xiangrui Yan, Zengdian Zhao, and Shasha Song. 2022. Dual-sensing, stretchable, fatigue-resistant, adhesive, and conductive hydrogels used as flexible sensors for human motion monitoring. *Langmuir* 38, 22 (2022), 7013–7023.

Satoshi Kataoka, Youngseog Chung, Seyed Kamyar Seyed Ghasemipour, Pannag Sanketi, Shixiang Shane Gu, and Igor Mordatch. 2023. Bi-Manual Block Assembly via Sim-to-Real Reinforcement Learning. *arXiv preprint arXiv:2303.14870* (2023).

Jean Won Kwak, Mengdi Han, Zhaoqian Xie, Ha Uk Chung, Jong Yoon Lee, Raudel Avila, Jessica Yohay, Xuexian

Chen, Cunman Liang, Manish Patel, et al. 2020. Wireless sensors for continuous, multimodal measurements at the skin interface with lower limb prostheses. *Science translational medicine* 12, 574 (2020), eabc4327.

Jonghyun Lee, Dahuin Jung, Junho Yim, and Sungroh Yoon. 2022. Confidence score for source-free unsupervised domain adaptation. In *International Conference on Machine Learning*. PMLR, 12365–12377.

Kevin Lin, Lijuan Wang, and Zicheng Liu. 2021. End-to-end human pose and mesh reconstruction with transformers. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*. 1954–1963.

Hong Liu, Jianmin Wang, and Mingsheng Long. 2021. Cycle self-training for domain adaptation. *Advances in Neural Information Processing Systems* 34 (2021), 22968–22981.

Zekun Liu, Tianxue Zhu, Junru Wang, Zijian Zheng, Yi Li, Jiashen Li, and Yuekun Lai. 2022. Functionalized Fiber-Based Strain Sensors: Pathway to Next-Generation Wearable Electronics. *Nano-Micro Letters* 14, 1 (2022), 1–39.

Mingsheng Long, Yue Cao, Jianmin Wang, and Michael Jordan. 2015. Learning transferable features with deep adaptation networks. In *International conference on machine learning*. PMLR, 97–105.

Matthew Loper, Naureen Mahmood, Javier Romero, Gerard Pons-Moll, and Michael J. Black. 2015. SMPL: A Skinned Multi-Person Linear Model. *ACM Trans. Graphics (Proc. SIGGRAPH Asia)* 34, 6 (Oct. 2015), 248:1–248:16. https://doi.org/10.1145/2816795.2818013

Yiyue Luo. 2023. Intelligent Textiles for Physical Human-Environment Interactions. In *Adjunct Proceedings of the 36th Annual ACM Symposium on User Interface Software and Technology*. 1–5.

Yiyue Luo, Yunzhu Li, Pratyusha Sharma, Wan Shou, Kui Wu, Michael Foshey, Beichen Li, Tomás Palacios, Antonio Torralba, and Wojciech Matusik. 2021. Learning human–environment interactions using conformal tactile textiles. *Nature Electronics* 4, 3 (2021), 193–201.

Naureen Mahmood, Nima Ghorbani, Nikolaus F Troje, Gerard Pons-Moll, and Michael J Black. 2019. AMASS: Archive of motion capture as surface shapes. In *Proceedings of the IEEE/CVF international conference on computer vision*. 5442–5451.

Alexander Mathis, Steffen Schneider, Jessy Lauer, and Mackenzie Weygandt Mathis. 2020. A primer on motion capture with deep learning: principles, pitfalls, and perspectives. *Neuron* 108, 1 (2020), 44–65.

Alex Kaszynski Michael Hogg. 2021. Pygeodesic. Retrieved mar, 2021 from https://github.com/mhogg/pygeodesic

Mixamo Animation. 2022. https://www.mixamo.com/.

Vimal Mollyn, Riku Arakawa, Mayank Goel, Chris Harrison, and Karan Ahuja. 2023. IMUPoser: Full-Body Pose Estimation using IMUs in Phones, Watches, and Earbuds. In *Proceedings of the 2023 CHI Conference on Human Factors in Computing Systems*. 1–12.

Matthias Müller, Alexey Dosovitskiy, Bernard Ghanem, and Vladlen Koltun. 2018. Driving policy transfer via modularity and abstraction. *arXiv preprint arXiv:1804.09364* (2018).

Sergey I Nikolenko. 2021. *Synthetic data for deep learning*. Vol. 174. Springer.

Sinno Jialin Pan and Qiang Yang. 2009. A survey on transfer learning. *IEEE Transactions on knowledge and data engineering* 22, 10 (2009), 1345–1359.

Hyunkyu Park, Junhwi Cho, Junghoon Park, Youngjin Na, and Jung Kim. 2020. Sim-to-real transfer learning approach for tracking multi-DOF ankle motions using soft strain sensors. *IEEE Robotics and Automation Letters* 5, 2 (2020), 3525–3532.

Michael T Rosenstein, Zvika Marx, Leslie Pack Kaelbling, and Thomas G Dietterich. 2005. To transfer or not to transfer. In *NIPS 2005 workshop on transfer learning*, Vol. 898.

Kaiyue Shen, Chen Guo, Manuel Kaufmann, Juan Jose Zarate, Julien Valentin, Jie Song, and Otmar Hilliges. 2023. X-avatar: Expressive human avatars. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 16911–16921.

Thomas Stauner, Frederik Blank, Michael Fürst, Johannes Günther, Korbinian Hagn, Philipp Heidenreich, Markus Huber, Bastian Knerr, Thomas Schulik, and Karl-Ferdinand Leiß. 2022. SynPeDS: A Synthetic Dataset for Pedestrian Detection in Urban Traffic Scenes. In *Proceedings of the 6th ACM Computer Science in Cars Symposium*. 1–10.

Baochen Sun and Kate Saenko. 2016. Deep coral: Correlation alignment for deep domain adaptation. In *European conference on computer vision*. Springer, 443–450.

Yating Tian, Hongwen Zhang, Yebin Liu, and Limin Wang. 2023. Recovering 3d human mesh from monocular images: A survey. *IEEE Transactions on Pattern Analysis and Machine Intelligence* (2023).

Massimo Totaro, Tommaso Poliero, Alessio Mondini, Chiara Lucarotti, Giovanni Cairoli, Jesùs Ortiz, and Lucia Beccai. 2017. Soft smart garments for lower limb joint position analysis. *Sensors* 17, 10 (2017), 2314.

Eric Tzeng, Judy Hoffman, Ning Zhang, Kate Saenko, and Trevor Darrell. 2014. Deep domain confusion: Maximizing for domain invariance. *arXiv preprint arXiv:1412.3474* (2014).

Jindong Wang, Yiqiang Chen, Shuji Hao, Wenjie Feng, and Zhiqi Shen. 2017. Balanced distribution adaptation for transfer learning. In *2017 IEEE international conference on data mining (ICDM)*. IEEE, 1129–1134.

Zirui Wang, Zihang Dai, Barnabás Póczos, and Jaime Carbonell. 2019. Characterizing and avoiding negative transfer. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*. 11293–11302.

Zitian Wang, Xuecheng Nie, Xiaochao Qu, Yunpeng Chen, and Si Liu. 2022. Distribution-Aware Single-Stage Models for Multi-Person 3D Pose Estimation. arXiv:2203.07697 [cs.CV]

Edward L White, Michelle C Yuen, Jennifer C Case, and Rebecca K Kramer. 2017. Low-Cost, facile, and scalable manufacturing of capacitive sensors for soft systems. *Advanced Materials Technologies* 2, 9 (2017), 1700072.

Alexander Winkler, Jungdam Won, and Yuting Ye. 2022. QuestSim: Human motion tracking from sparse sensors with simulated avatars. In *SIGGRAPH Asia 2022 Conference Papers*. 1–8.

Chengzhi Wu, Xuelei Bi, Julius Pfrommer, Alexander Cebulla, Simon Mangold, and Jürgen Beyerer. 2023. Sim2real Transfer Learning for Point Cloud Segmentation: An Industrial Application Case on Autonomous Disassembly. In *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision*. 4531–4540.

Jianfei Yang, Xiangyu Peng, Kai Wang, Zheng Zhu, Jiashi Feng, Lihua Xie, and Yang You. 2022. Divide to adapt: Mitigating confirmation bias for domain adaptation of black-box predictors. *arXiv preprint arXiv:2205.14467* (2022).

Xinyu Yi, Yuxiao Zhou, Marc Habermann, Soshi Shimada, Vladislav Golyanik, Christian Theobalt, and Feng Xu. 2022. Physical inertial poser (pip): Physics-aware real-time human motion tracking from sparse inertial sensors. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 13167–13178.

Xinyu Yi, Yuxiao Zhou, and Feng Xu. 2021. Transpose: Real-time 3d human translation and pose estimation with six inertial sensors. *ACM Transactions on Graphics (TOG)* 40, 4 (2021), 1–13.

Jason Yosinski, Jeff Clune, Yoshua Bengio, and Hod Lipson. 2014. How transferable are features in deep neural networks? *Advances in neural information processing systems* 27 (2014).

Chaohui Yu, Jindong Wang, Yiqiang Chen, and Meiyu Huang. 2019. Transfer learning with dynamic adversarial adaptation network. In *2019 IEEE International Conference on Data Mining (ICDM)*. IEEE, 778–786.

Miao Yu, Jiaqin Jin, Xia Wang, Xu Yu, Dingjia Zhan, and Jia Gao. 2021. Development and design of flexible sensors used in pressure-monitoring sports pants for human knee joints. *IEEE Sensors Journal* 21, 22 (2021), 25400–25408.

Bouchaib Zazoum, Khalid Mujasam Batoo, and Muhammad Azhar Ali Khan. 2022. Recent advances in flexible sensors and their applications. *Sensors* 22, 12 (2022), 4653.

Hechuan Zhang, Zhiyong Chen, Shihui Guo, Juncong Lin, Yating Shi, Xiangyang Liu, and Yong Ma. 2020. Sensock: 3D foot reconstruction with flexible sensors. In *Proceedings of the 2020 CHI Conference on Human Factors in Computing Systems*. 1–13.

Quan Zhang, Tao Jin, Jianguo Cai, Liang Xu, Tianyiyi He, Tianhong Wang, Yingzhong Tian, Long Li, Yan Peng, and Chengkuo Lee. 2022b. Wearable Triboelectric Sensors Enabled Gait Analysis and Waist Motion Capture for IoT-Based Smart Healthcare Applications. *Advanced Science* 9, 4 (2022), 2103694.

Wen Zhang, Lingfei Deng, Lei Zhang, and Dongrui Wu. 2022a. A survey on negative transfer. *IEEE/CAA Journal of Automatica Sinica* 10, 2 (2022), 305–329.

Bo Zhou, Daniel Geissler, Marc Faulhaber, Clara Elisabeth Gleiss, Esther Friederike Zahn, Lala Shakti Swarup Ray, David Gamarra, Vitor Fortes Rey, Sungho Suh, Sizhen Bian, et al. 2023. MoCaPose: Motion Capturing with Textile-integrated Capacitive Sensors in Loose-fitting Smart Garments. *Proceedings of the ACM on Interactive, Mobile, Wearable and Ubiquitous Technologies* 7, 1 (2023), 1–40.

Brady Zhou, Philipp Krähenbühl, and Vladlen Koltun. 2019. Does computer vision matter for action? *Science Robotics* 4, 30 (2019), eaaw6661.

Chengxu Zuo, Jiawei Fang, Shihui Guo, and Yipeng Qin. 2023. Self-adaptive motion tracking against on-body displacement of flexible sensors. (2023).

A. Hardware

We design and develop a prototype by adding two capacitive strain sensors to the standard elbow and knee pads, which are placed on the ulnar side (elbow) and tibial side (knee).

The length and circumference of the pad are 20 and 25 cm (elbow pad), and 20 and 40cm (knee pad). The two sensors are placed 2 cm apart. Our method aims to estimate the bending angle, θ, of an elbow joint (Fig. 7) from the sensor readings of the two flexible sensors. The elbow bending angle is defined as the angle in the sagittal plane between the humerus and the central line between the radius and the ulna. In contrast, the knee bending angle is defined as the angle in the sagittal plane between the femur and the central line between the tibia and the fibula.

Fabric sensors are purchased as off-shelf products from ElasTech. They are capacitive, *i.e.*, and their capacitance increase with the stretch caused by the bending of the arm. The sensor readings are digitized to values in the range [0, 1023] and transmitted wirelessly via Bluetooth Low Energy at a frame rate of 50Hz. The collected sensor readings are x in the target domain.

Figure 7. Our smart elbow and knee pad (hardware).

For capacitive strain sensors, due to their high stretchability, lightness, and ease of attachment near joints, are suitable for monitoring human body movements in unrestricted wearable devices. As flexible materials inherently possess stickiness and elasticity, and joint movements involve twisting, bending, and other deformations, hysteresis and low repeatability may occur. To address these issues, we have taken the following measures:

1. We selected the Elas Tech ESSB-01 commercial capacitive sensor, which features high reliability (over 300,000 bending cycles), thinness (as thin as 1mm), and high linearity.

2. In order to improve repeatability and suppress hysteresis, we used a blended fabric with high spandex content and stretchable wires to assemble the sensor and the entire system. We pre-stretched the assembled system to eliminate its internal stress, and measured a linearity of $R^2 = 0.999$ after pre-stretching.
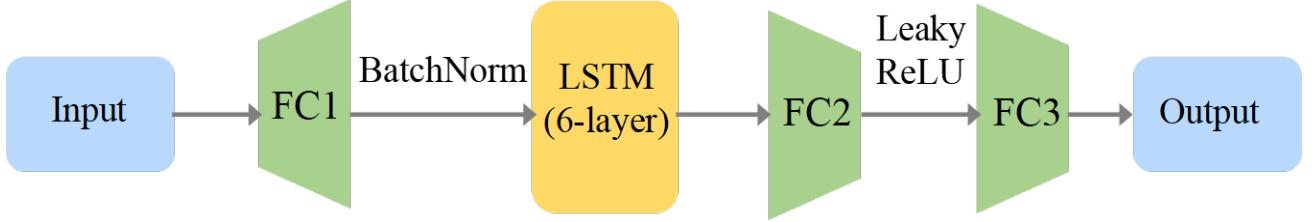
Figure 8. Detailed network architecture. LSTM: Long Short-Term Memory. FC: Fully-Connected.

B. Implementation Details

Fig. 8 shows the proposed network's architecture details. Specifically, we use a 3-layer LSTM network with a hidden layer size of 256 for sequence feature extraction and feed both its hidden state and cell state into FC2 for subsequent computation. The output sizes of network layers are listed in Table 8.

Table 8. Details of network layers.

Layer	Output Size
FC1	256
LSTM	$6 \times 256 \times 2$
FC2	128
FC3	1

To make a fair comparison, we used the same architecture shown in Table 8 and the same hyper-parameters.

In addition, we used the same loss $\mathcal{L}_{mae}$ (please see the main paper) for the supervised pre-training but made the following modifications for different competitors:

- **MMD** (Long et al., 2015): We applied a Max Mean Discrepancy (MMD) loss at the output of FC2 for the adaptation.

- **D-CORAL** (Sun and Saenko, 2016): We applied a CORAL loss at the output of FC2 for the adaptation.

- **DANN** (Ganin and Lempitsky, 2015): We applied a gradient reverse layer and a domain classifier (a 2-layer fully connected network) at the output of FC2 for the adaptation.

- **DAAN** (Yu et al., 2019): We added a softmax layer after FC2 and applied a DAAN loss at its output for the adaptation. The dynamic factor was updated every epoch.

- **BNM** (Cui et al., 2020): We applied a Batch Nuclear-norm Maximization (BNM) loss at the output of FC2 for the adaptation.

C. Evidence of Eq. 7

$$\int_{x_s^a}^{x_s^b} \frac{df_s}{dx_s} dx_s = \int_{x_t^a}^{x_t^b} \frac{df_t}{dx_t} dx_t = \int_0^l \frac{df_l}{dl} dl \tag{11}$$

suggests that data with equal parameter change (*i.e.,* $l - 0$) have the same label change (*i.e.,* $y^b - y^a$), we empirically evaluate this by conducting parameterization to both simulation (source) and real (target) function supports (Fig. 9).

D. Illustration of Distribution Variations

From (a) to (e), we observe commonalities in simulation data: they all share similar label ranges. However, the distinction lies in the distribution of labels. Simulation data exhibits a relatively uniform distribution across labels, with a higher
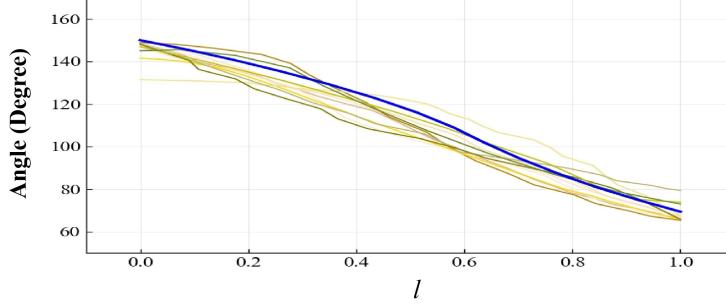
Figure 9. The relationship between parameters l and label angle. The blue one is the simulation data and the others are real data samples.

(a) $\mathcal{D}(Simulation)$ (b) $\mathcal{D}(id = 1)$ (c) $\mathcal{D}(wp = 180°)$ (d) $\mathcal{D}(m = walk)$ (e) $\mathcal{D}(RightKnee)$

(f) $\mathcal{D}(Simulation)$ (g) $\mathcal{D}(id = 1)$ (h) $\mathcal{D}(wp = 180°)$ (i) $\mathcal{D}(m = walk)$ (j) $\mathcal{D}(RightKnee)$

(k) $\mathcal{D}(Simulation)$ (l) $\mathcal{D}(id = 1)$ (m) $\mathcal{D}(wp = 180°)$ (n) $\mathcal{D}(m = walk)$ (o) $\mathcal{D}(RightKnee)$

(p) $\mathcal{D}(Simulation)$ (q) $\mathcal{D}(id = 1)$ (r) $\mathcal{D}(wp = 180°)$ (s) $\mathcal{D}(m = walk)$ (t) $\mathcal{D}(RightKnee)$
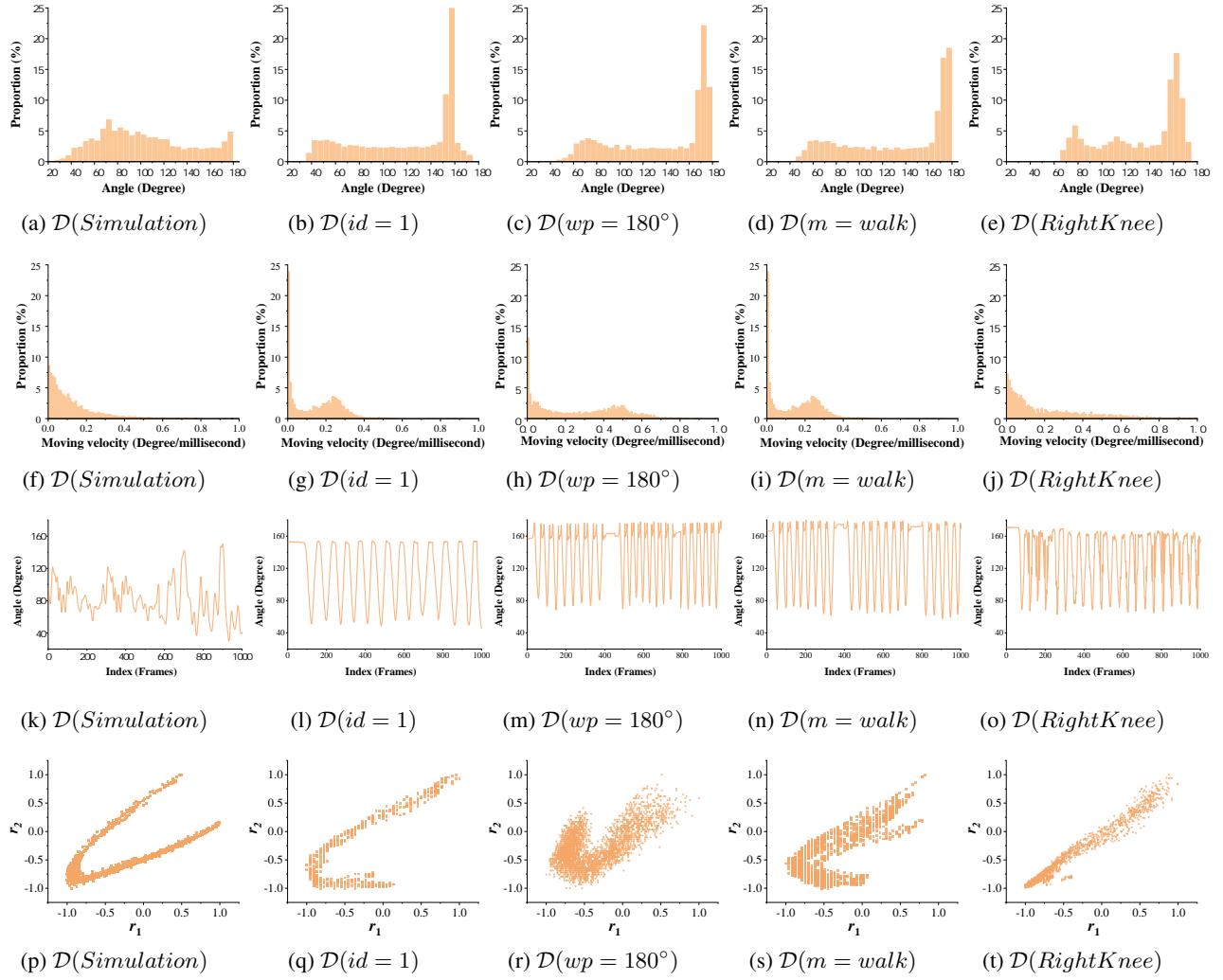
Figure 10. (a) (b) (c) (d) (e) *Label (Angle) proportion* of Simulation, User, Position, Motion, and Joint data Samples; (f) (g) (h) (i) (j) *Moving Velocity* of Simulation, User, Position, Motion, and Joint data Samples; (k) (l) (m) (n) (o) *Time Sequence* of Simulation, User, Position, Motion, and Joint data Samples; (p) (q) (r) (s) (t) Function Supports (sensor readings) of Simulation, User, Position, Motion, and Joint data Samples;

concentration within a smaller label range (60 degrees to 100 degrees). In contrast, real users, positional motions, and joint movements tend to be more prevalent in a broader label range (160 degrees to 180 degrees), constituting over 30%. This discrepancy may be attributed to the increased occurrence of elbow extension movements in real-life scenarios.

When examining the moving velocity (from (f) to (j)), it is evident that the proportion of simulation data velocities gradually decreases from (0-0.4) (from 10 % to 0 %). In contrast, real data for users, positions, and motions experiences a decline followed by a sudden increase. This phenomenon may be attributed to the fact that the range of 0.2 to 0.6 (degrees/microsecond) corresponds to the natural movement speed of the human body, thus constituting a higher proportion.

The time-sequence plots from (k) to (o) reveal a notable mismatch between simulation data and all real data. In fact, in practical usage scenarios, it is challenging to identify two action sequences with entirely matching time sequences between virtual and real data. This substantial disparity in distribution between virtual and real data is a crucial factor contributing to this discrepancy.

In (p) through (t), the support distribution plots are presented. 1. Real data exhibits a more pronounced dispersion compared to simulation data, attributed to inherent biases or noise introduced during the sensor-based real data acquisition process. 2. The shapes of the support distributions for simulation and real data differ. SuDA assumes identical $\frac{df}{dx}$, aligning supports to achieve successful migration, offering a mechanism to address the dissimilarity between the support distributions of simulation and real data.

In conclusion, the data presented above underscores significant disparities in the distribution between simulation data and real data. Moreover, the failure of the DiDA method can be attributed to the low-dimensionality of the data.

E. Evaluations of SuDA

Figure. 11 shows the Performance Visualization of our method SuDA. Figure 12, 13, and 14 show that **SuDA** achieves comparable motion capture predictive performance in different Users, Wearing Positions and motions, although the source data (Simulation) is quite different from target data (Real world).
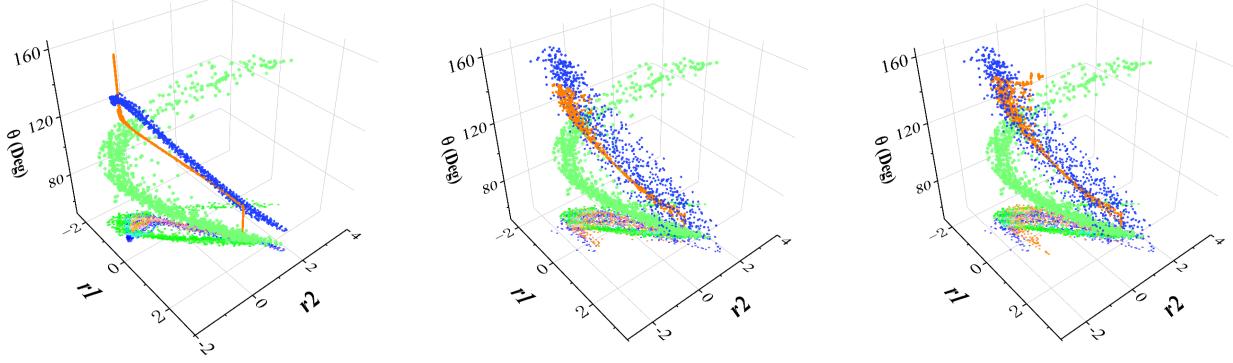

(a) $\mathcal{D}_s$ and $\mathcal{D}(id = 1)$ Visualization. (b) $\mathcal{D}_s$ and $\mathcal{D}(wp = 180°)$ Visualization. (c) $\mathcal{D}_s$ and $\mathcal{D}(m = run)$ Visualization.

Figure 11. Performance Visualization of our method **SuDA**. Note that the green points represent source data (Simulation), the blue points represent target data (Real World), and the orange points represent **SuDA** registered simulation data. It is evident that **SuDA** is capable of aligning source and target data in the feature space by aligning the *supports*.
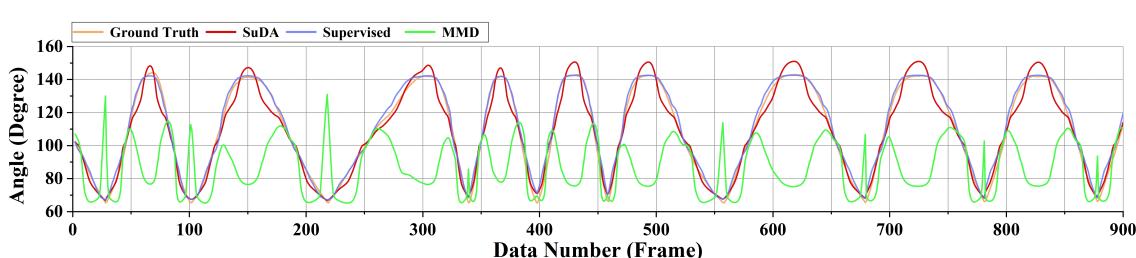

Figure 12. A plot of the detailed predict results on $\mathcal{D}(id = 1)$.
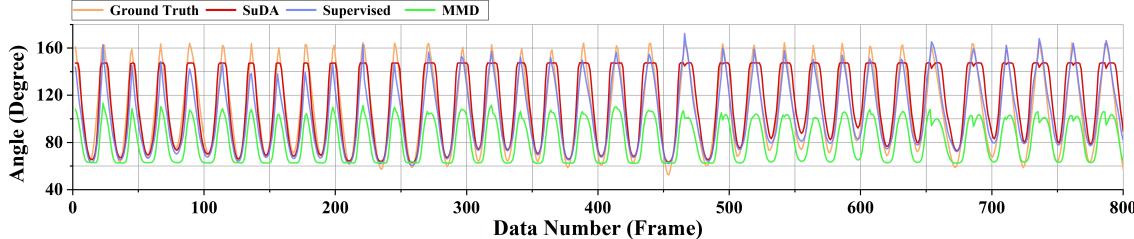
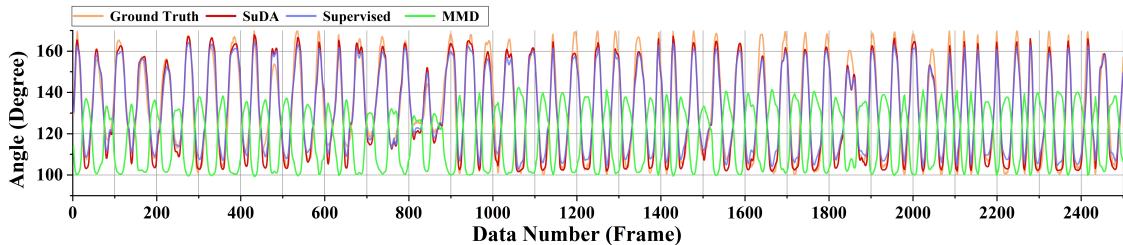*Figure 13.* A plot of the detailed predict results on $\mathcal{D}(wp = 180°)$.

Figure 14. A plot of the detailed predict results on $\mathcal{D}(m = run)$.

F. Potential Applications

Table 9 shows that it is possible to predict full-body movement from hand and leg motion signals using our method. To demonstrate this, we conducted a pilot experiment on full-body movements and use the AMASS (Mahmood et al., 2019) dataset for training and testing. Specifically, we used SuDA to predict the angles of four joints, and then used an auto-regressive model to predict full-body joint angles based on them. Experimental results show that SuDA could achieve effective full-body motion capture from only four flexible sensor pads, with a more comfortable user experience.

G. Failure Case of SuDA

SuDA may fail when the supports of two domains are inherently different. Specifically, we assume that both the simulated and real movements cover the full elbow bending angles (around 30° to 180°). Therefore, SuDA will degenerate when real elbow movements cover fewer angles (Table 10).

H. Real World Dataset Description

The real-world data collected was calculated using the three-dimensional positions of markers worn on the volunteers' elbow joints to determine real-time angle labels. The MARS 2H motion capture system, produced by NOKOV 3 company, tracked and calculated the three-dimensional positions of the markers at a rate of 60 frames per second. The motion capture system's three-dimensional tracking error is ±15mm, and the estimated rotation tracking error is a maximum of 0.88.

Users We collected 11 volunteers as our User data, which can be denoted as $D(id = 1, 2, ..., 11)$. Their body shape details can be seen in Table 11. Our User dataset consists of a total of 219, 932 frames.

Positions Considering that users of elbow pads may have different wearing habits, we took into account different wearing positions when collecting data. We explore the effect caused by circular types of offsets. We moved the elbow pad in 30-degree increments for circular offset to collect data, denoted as $D(wp = 30°, 60°, ..., 360°)$. Our position dataset consists of a total of 155, 296 frames.

Motions Motion data of four basic motions, which can be denoted as $D(m = run, walk, jump, clap)$. Our Motion dataset consists of a total of 143, 575 frames.

Sum After data cleansing and calibration, we finally collected 371, 611 frames as our aggregate data, which can be denoted as $\mathcal{D}_{t^*}^{(id,m,wp)}$.

Table 9. Results of pilot experiment on full-body movement prediction from hand and leg motion signals.

Motion	MPJRE	MPJPE	MPJVE
Pick	9.56	9.39	64.07
Run	8.27	7.00	48.67
Walk	8.33	7.61	70.72
Combo-motions	5.16	4.54	38.59

Table 10. SuDA will degenerate if real movements cover fewer angles than simulated ones (around $30°$ to $180°$). To demonstrate this, we tested SuDA on modified real motions where we manually filtered out data outside a given angle range. The angle range was determined from the average of the real data, i.e., filter symmetrically from both sides of the mean.

Angle Range (%)	MAE
100	5.94
90	10.44
80	13.16
70	14.47
60	17.16
50	19.38
40	21.43
30	22.59
20	24.76
10	27.77

Applications of Real-world scenarios The real-world scenarios ((*table tennis, basketball, football, boxing*)) participants can be seen in Table 12. Our Real world dataset consists of a total of 81, 848 frames. Our different joints applications ((*Left elbow, right elbow, left knee, right knee*)) datasets comprises 22, 577 frames.

I. Simulation Framework

We used the SMPL model to create fbx files of simulated human 3D models. Note that the skeleton model discussed in the following steps is a standard human skeleton obtained via MotionBuilder binding, with 68 joints instead of the 24 joints native to SMPL. The entire process of synthesizing digital mannequins through SMPL models can be divided into three major stages as follows.

I.1. SMPL Establishment

Shape Blend Shapes Firstly, we established a mean template $\overline{T}$ as the basic pose of the whole human body, which was obtained statistically by using N=6,890 vertexes to represent the whole mesh. Then we described the pose of the human body and the offset of the basic pose through the parameters and superimposed them to form the final desired human pose. The process described above is linear, which contains a matrix multiplication procedure for parametric linear matrices, and we will continue to discuss it next. The pose of the human mesh obtained here is called the T-pose because it does not consider the influence of the pose parameters.

Pose Blend Shapes When we specified the shape of the human mesh according to the specified β parameter, we got a mesh with a specific height and weight. Nevertheless, we knew that a specific action might affect the specific shape of the local human body changes; in other words, the pose parameter θ will also affect the silent pose mesh shape to some extent.

Skinning We only calculated the mesh in the static pose in the previous two stages. When human skeleton joints move, the "skin" composed of vertexes will change with the movement of the skeleton joints; this process is called skinning. The skinning process can be considered a weighted linear combination of skin nodes with changes in skeletal joints. However, the closer an endpoint is to a specific bone joint, the stronger the effect of changes, such as rotation and translation, with that bone joint.

Table 11. Body profile details of the eleven experiment participants.

User ID	upperc(cm)	lowerc (cm)	height (cm)	weight (kg)	Gender
ID1	24.6	24	186	78.3	Female
ID2	24	23.5	174	61	Female
ID3	25.8	24	171	69	Male
ID4	24.3	23.8	164	54.6	Famale
ID5	25	24.5	170	60.2	Male
ID6	28	27	160	72.2	Male
ID7	23	22	158	50.2	Female
ID8	25	24.2	164	61	Male
ID9	26.3	27.5	170	65.4	Male
ID10	30	28.5	175	86	Male
ID11	29	27.6	177	75	Male

Table 12. Body profile details of the five real world scenarios experiment participants

User ID	Gender	Height (cm)	Weight (kg)	Arm circumference (cm)
1	Male	179	84	30.2
2	Male	175	86	28.5
3	Male	177	75	27.6
4	Female	172	67	27.0
5	Female	168	65	26.6

I.2. Motion Binding

To address the effect of different motions on sensor parameters, we collected approximately 30,000 frames (some samples are increased to balance the weight) of motion capture data from the free online animation library Mixamo, which contains motion capture files for walking, running, clapping, and hybrid motions. Furthermore, we used the online binding function to bind the motions to the 3D human model generated based on SMPL.

I.3. Fabric Simulation

The creases produced by the flexible sensors when worn and the differences in wearing position are also sources of Sim2Real domain gaps. In order to solve these problems, we used the cloth database built into the clothing design software Marvelous Designer. Considering the basic parameters such as elasticity, friction, and thickness of real elbow pads and visual comparison inspection of simulation results, we selected hard-jeanet from the database. The simulation can produce similar wrinkles, pulling, and other phenomena as real motion scenes.

I.4. Elbow Bending Angle and Sensor Readings Simulation

Our task is to estimate the bending angle of the elbow joint based on two flexible sensors surrounding the arm. Previously, it usually required professional motion capture (mocap) equipment to complete, *i.e.*, the wearer had to wear the mark points of the mocap equipment, and the data needed to be collected in a specific professional environment (as shown in Figure (b)). As soon as the data collection was completed, professional staff had to preprocess the collected data (Figure (c)). There is no doubt that data collection is highly labor-intensive and time-consuming. To address this issue, our SuDA method employs a Sim2Real paradigm and no longer requires real data, thus significantly reducing the costs for flexible sensor-based motion capture.

Elbow Bending Angle Simulation We use the Biovision (BVH) file to calculate the elbow angle of SMPL. BVH file is a general human feature animation file format containing the rotation data of the character's bone and limb files. Various popular animation software, such as Maya and 3DMax, are widely supported by it. The file mainly consists of two parts: skeleton information and data block:

Table 13. Motion types of simulation datasets.

Class	Dataset Size
Chicken Dance	571
House Dance	2376
House Dance2	3142
Macarena Dance	988
Robot Hip Hop Dance	1852
Robot Hip Hop Dance	2798
Swing Dancing2	297
Swing Dancing3	624
Swing Dancing4	2965
Swing Dancing5	2510
Tut Hip Hop Dance	2032
Sum	20155

- Skeleton information: According to the hierarchical relationship, the required three-dimensional coordinate positions, such as the forearm, arm, hand, and Euler angles of rotation components, are defined to form a complete skeleton.

- Data block: the data information of each frame part is marked.

Since the 3D position of each part of the skeleton in the original BVH file is not easy to extract, hampering the subsequent input of neural network data, we use a script called bvh-converter to preprocess the original BVH file and obtain the file worldpose.csv that records the 3D coordinate position information of each frame of motion joint. The forearm, arm vector, and arm, hand vector are obtained through the 3D position calculation and then utilize formula $\cos\theta = \frac{\mathbf{V}_1 \times \mathbf{V}_2}{|\mathbf{V}_1 \times \mathbf{V}_2|}$ to obtain the elbow bending angle.

Sensor Readings Simulation The Geodesic Distance of Fabric was used to calculate simulated sensor readings. First, we split the processed FBX file into a bunch of OBJ files, which contain the vertex data free-form curves, rendering attributes, and other information of each frame. Then we used **pygeodesic** (Michael Hogg, 2021) to calculate the geodesic distance for simulative fabric.