MODA: Motion-Drift Augmentation for Inertial Human Motion Analysis

Supplementary Material

7. Discussion

7.1. MODA Compatibility with IMU Signals

MODA is a specialized data augmentation technique designed explicitly for IMU signals. Its strength lies in its ability to simulate the natural drift of body-worn IMUs during human movement. This approach not only enhances the realism of augmented data by mirroring complex, real-world scenarios but also adheres seamlessly to the three fundamental properties intrinsic to IMU signals.

Time-Series. As discussed in the main paper, MODA simulates IMU drift, a time-dependent variable, making it inherently compatible with the time-series nature of IMU signals. In other words, the temporal variations in drift ensure that the augmented IMU signals preserve their intrinsic time-series properties, maintaining coherence with real-world dynamics.

Multimodality. MODA simulates two distinct types of drift commonly observed in IMU-based human motion analysis: self-rotation and relative sliding. These are used to augment rotation and acceleration measurements, effectively capturing the multimodal nature of IMU signals. Furthermore, it can be formally demonstrated that this augmentation preserves the orthogonality of rotation readings, ensuring their physical validity:

Proposition 4 *The measurement $\hat{R}^G_{\text{sensor}}(t)$ after MODA augmentation still maintains its orthogonality.*

According to Eq. (7) in the main paper, $\hat{R}^G_{\text{sensor}}(t)$ is defined as:

$$\hat{R}^G_{\text{sensor}}(t) = P^{GM} R^M_{\text{bone}}(t)(R^G_{\text{offset}}(t))^{-1} \tag{14}$$

And we have:

$$\begin{aligned}(\hat{R}^G_{\text{sensor}}(t))^\intercal &= (P^{GM} R^M_{\text{bone}}(t)(R^G_{\text{offset}}(t))^{-1})^\intercal \\ &= ((P^{GM} R^M_{\text{bone}}(t))\ \ ((R^G_{\text{offset}}(t))^\intercal))^\intercal \\ &= R^G_{\text{offset}}(t)\ \ (P^{GM} R^M_{\text{bone}}(t))^\intercal \\ &= R^G_{\text{offset}}(t)(R^M_{\text{bone}}(t))^\intercal (P^{GM})^\intercal\end{aligned} \tag{15}$$

$$\begin{aligned}\hat{R}^G_{\text{sensor}}(t)(\hat{R}^G_{\text{sensor}}(t))^\intercal &= P^{GM} R^M_{\text{bone}}(t)(R^G_{\text{offset}}(t))^{-1} \\ &\quad R^G_{\text{offset}}(t)(R^M_{\text{bone}}(t))^\intercal (P^{GM})^\intercal \\ &= P^{GM} R^M_{\text{bone}}(t)(R^G_{\text{offset}}(t))^\intercal \\ &\quad R^G_{\text{offset}}(t)(R^M_{\text{bone}}(t))^\intercal (P^{GM})^\intercal \\ &= I\end{aligned} \tag{16}$$

Therefore, $\hat{R}^G_{\text{sensor}}(t)$ maintains orthogonality.

Motion-Consistency. In Eqs. (7) and (8) of the main paper, the augmented IMU measurements $\hat{R}^G_{\text{sensor}}(t)$ and $\hat{a}^S_{\text{sensor}}(t)$ are explicitly derived from the bone movements R^M_{bone} and a^M_{bone}. This ensures that the augmented IMU signals closely and accurately encapsulate the underlying physical human motion.

7.2. MODA's Generalization

IMUs inside Smart Glasses. In this section, we have discussed MODA's performance with strictly aligned device positions and poses, such as the IMUs inside smart glasses. We believe that this falls outside the scope of our paper, as we focus on IMUs attached to the human body, covered by "soft skin", rather than rigid metal surfaces on robots, where strict alignment of IMUs is achievable. Specifically, while the relative positions of IMUs may be aligned with smart glasses, the frames of the glasses inevitably shift along the ears, especially during prolonged and comfortable wear.

Six-axis IMUs. For IMU data containing only angular velocity and acceleration, a fusion algorithm (*e.g.*, Kalman filter) can be employed to synthesize the rotation matrix, allowing us to apply our MODA framework accordingly. As shown in Table 9, our conclusions remain consistent on such IMU data.

Table 9. Comparison of Baseline and MODA on PAMAP2 [40] dataset in the **Fully-Supervised** and **Few-Shot** scenarios.

	Fully-Supervised		Few-Shot	
	top-1	top-5	top-1	top-5
Baseline (w/o aug.)	59.63%	90.06%	48.92%	84.77%
MODA	**63.21%**	**92.64%**	**49.88%**	**86.27%**

8. Implementation Details

We begin by providing a detailed description of the experimental setup for the HPE (Sec. 8.1) and HAR (Sec. 8.2) tasks. Next, we elaborate on the *six* data augmentation techniques along with their implementation details (Sec. 8.3). We also detail the training settings for all models (Sec. 8.4).

8.1. Experimental Setup for HPE

Datasets and Settings. We utilize the following public datasets: AMASS [25] (*synthetic*), DIP [15], TotalCapture [46], CIP [35] and ANDY [26]. And we also define four data settings. Specifically, i) for **Fully-Supervised**

Learning setting, we utilize all samples from the training sets ($ratio = 1$) of CIP and AnDy datasets to train the models, subsequently evaluating them on the corresponding test sets. ii) for **Semi-Supervised Learning** setting, we leverage a limited number of labeled samples ($ratio = 1/r, r = \{8, 16, 32\}$ from CIP and ANDY) for model training and evaluate it on the corresponding test sets. iii) for **Domain Adaptation** setting, we train the model using the AMASS (*synthetic*) dataset along with a few samples from real datasets (CIP, ANDY, DIP and TotalCapture), and evaluate its performance on their respective test sets. iv) for **Domain Generalization** setting, we exclusively utilize the AMASS (*synthetic*) for training the model without incorporating any real data, and we then test it on real datasets (CIP, ANDY, DIP and TotalCapture). Note that we do not have a Few-Shot Learning setting for HPE as its input motion sequences are very limited (HPE has only one class) and do not provide enough data to train a meaningful pose estimator, leading to extremely high errors.

Evaluation Metrics. Following the previous works [53, 59, 60, 65], we utilize the five metrics to assess our MODA for HPE task: i) *SIP error*, which measures the average global rotation error of upper arms and upper legs in degrees; ii) *Angular error*, which measures the average global rotation error of all body joints in degrees; iii) *Positional error*, which measures the average Euclidean distance error of all estimated joints in centimeters (cm) with the root joint (Spine) aligned; iv) *Mesh error*, which measures the average Euclidean distance error of all vertices of the estimated body mesh (SMPL [22]) in centimeters (cm) with the root joint (Spine) aligned. Noted that the errors in twisting (rotations around the bone) can not be measured by positional error, but will be reflected in mesh error; and v) *Jitter*, measuring the mean jerk (time derivative of acceleration) of all body joints in the global space, which reflects the smoothness [9] of the motion sequence.

8.2. Experimental Setup for HAR

Datasets. In the field of IMU-based HAR, there exist multiple real datasets such as Human Activities and Postural Transitions (**HAPT**) [41], RealWorld HAR (**RW**) [44], **GesHome** [31] and Physical Activity Monitoring Data Set (**PAMAP2**) [40]. The reason we did not use these datasets is that their action categories are very limited (at most 19), and current methods have already achieved high accuracy (over 90%). In contrast, we utilize the BABEL [38], a large dataset with language labels describing the actions being performed in mocap sequences [25]. Specifically, we first synthesize the IMU measurements (following Transpose [59]), by placing *six* virtual IMUs on the human body. Next, we use the action categories from the BABEL as labels for the synthetic data to train and evaluate the model. The dataset split is consistent with BABEL.

Settings. We define two data settings. Specifically, i) for **Fully-Supervised Learning** setting, we leverage all available training samples from the BABEL dataset for model training and then we test it on its test sets. ii) for **Few-Shot Learning** setting, we manually reduce the number of samples in the training set so that each category contains no more than 10 training samples (denoted as BABEL*), considering that certain specific categories of actions are difficult to collect in real-world situations [20, 50]. Note that we do not have a Semi-Supervised Learning setting for HAR as BABEL contains only around 2,000 samples, averaging fewer than 20 samples per category. This limited data volume is insufficient for semi-supervised learning but aligns well with a few-shot learning setting. Also, we do not include Domain Adaptation and Domain Generalization settings due to the lack of publicly available datasets.

8.3. Implementations of DA Methods

Each data augmentation (DA) technique generates **four** additional augmented versions based on the original training data. Given the time-series (Property 1) and multi-modal (Property 2) nature of IMU signals, *i.e.*, $x(t) = \{R(t), a(t)\}$, these techniques are applied separately to the rotation measurements $R(t)$ and the acceleration measurements $a(t)$. This results in the augmented measurements $\hat{R}_i(t)$ and $\hat{a}_i(t)$, where $i = \{0, 1, 2, 3\}$ denotes a specific augmented version. Detailed parameter configurations for these augmentations are provided below.

Jittering. The augmented measurements $\hat{R}_i(t)$ and $\hat{a}_i(t)$ can be calculated as follows:

$$\hat{R}_i(t) = R(t) + \Delta R_i(t), \Delta R_i(t) \sim \mathcal{N}(\mu_i^{rot}, (\sigma_i^{rot})^2) \tag{17}$$

$$\hat{a}_i(t) = a(t) + \Delta a_i(t), \Delta a_i(t) \sim \mathcal{N}(\mu_i^{acc}, (\sigma_i^{acc})^2) \tag{18}$$

where μ_i^{rot}, σ_i^{rot}, μ_i^{acc} and σ_i^{acc} are hyperparameters for the i-th *jittering* agumentation and we set them as follows: $\mu_i^{rot} = \{0, 0, 0, 0\}$, $\sigma_i^{rot} = \{1, 0.1, 0.5, 0.01\}$ and $\mu_i^{acc} = \{0.5, -0.5, 0.1, -0.1\}$, $\sigma_i^{acc} = \{1, 1, 1, 1\}$. In practice, we first sample from a standard Gaussian distribution (*e.g.*, $z_i^{rot} \sim \mathcal{N}(0, 1)$) and use the reparameterization trick to map it to $\Delta R_i(t)$ like:

$$\Delta R_i(t) = z_i^{rot} * \sigma_i^{rot} + \mu_i^{rot} \tag{19}$$

Moving Average. The augmented measurements $\hat{R}_i(t)$ and $\hat{a}_i(t)$ can be calculated as follows:

$$\hat{R}_i(t) = \frac{1}{N_i} \sum_{j=t-k_i}^{t-k_i+N_i-1} R(j), \quad 0 \le k_i \le N_i \tag{20}$$

$$\hat{a}_i(t) = \frac{1}{N_i} \sum_{j=t-k_i}^{t-k_i+N_i-1} a(j), \quad 0 \le k_i \le N_i \tag{21}$$

where k_i and N_i are hyperparameters for the i-th *moving average* augmentation, denoting the window size before moment t and the total window size, respectively. We set them as follows: $k_i = \{7, 15, 15, 31\}$ and $N_i = \{15, 15, 31, 31\}$. In practice, we utilize different 1D convolution kernels to implement these operations.

Random Masking. The augmented measurements $\hat{R}_i(t)$ and $\hat{a}_i(t)$ can be calculated as follows:

$$\hat{R}_i(t) = R(t) \odot M_i^{rot}, \quad M_i^{rot}[i][j] \in \{0, 1\} \tag{22}$$

$$\hat{a}_i(t) = a(t) \odot M_i^{acc}, \quad M_i^{acc}[i][j] \in \{0, 1\} \tag{23}$$

where $\odot$ denotes element-wise multiplication, $M_i^{rot} \in \mathbb{R}^{3\times3}$ and $M_i^{acc} \in \mathbb{R}^3$ denote the mask matrices. We set the mask ratios to $r_i = \{0.1, 0.25, 0.5, 0.75\}$, which correspond to masking 1, 3, 6, and 9 elements, respectively.

To ensure that the augmented rotation $\hat{R}_i(t)$ and acceleration $\hat{a}_i(t)$ at the same time remain consistent with the original sequence (*i.e.*, $\hat{R}_i(t)$ and $\hat{a}_i(t)$ still come from the same timestamp before augmentation), we first concatenate R and a along the feature dimension before applying *flipping* and *slicing* operations:

$$x = \text{concatenate}(\text{reshape}(R), a) \tag{24}$$

where $x \in \mathbb{R}^{T\times(3\times3+3)}$ denotes the concatenated vector.

Flipping. We define *flipping* as a more general operation of shuffling the order of the input sequence, where reversal is a special case. Given a concatenated IMU signal vector $x \in \mathbb{R}^{T\times12}$ with T frames, the augmented measurements $\hat{x}_i$ can be calculated as follows:

$$\hat{x}_i = x[o_i], \quad o_i[j] \in \{0, 1, ..., T-2, T-1\} \tag{25}$$

where $o_i \in \mathbb{R}^T$ denotes the order of the i-th augmented sequence and each element $o_i[j]$ denotes the frame index. We set o_0 as the *reversal* case, *i.e.*, $o_0 = [T-1, T-2, ..., 1, 0]$, and the other three o_1, o_2, o_3 are arranged in a random order.

Slicing. The augmented measurements $\hat{x}_i$ can be calculated as follows:

$$\hat{x}_i = x[s_i], \quad s_i = [\text{begin}_i : \text{end}_i] \tag{26}$$

where s_i denotes the segment of i-th augmentation, begin_i and end_i denote the indexes. We set them as follows: $\text{begin}_i = \{0, \frac{T}{4}, \frac{T}{2}, \frac{3T}{4}\}$ and $\text{end}_i = \{\frac{T}{4}, \frac{T}{2}, \frac{3T}{4}, T\}$.

MODA. As mentioned in the main paper, we first sample two random and time-varying perturbations $\Delta\hat{\theta}_i(t)$ and $\Delta\hat{d}_i(t)$, and generating the IMU signals with Motion-Drift augmentation according to Eqs. (7), (8), (9), (10) and (11). The perturbations $\Delta\hat{\theta}_i(t)$ and $\Delta\hat{d}_i(t)$ follow that:

$$\Delta\hat{\theta}_i(t) \sim \mathcal{N}(\mu_i^{rot}, (\sigma_i^{rot})^2), \Delta\hat{d}_i(t) \sim \mathcal{N}(\mu_i^{acc}, (\sigma_i^{acc})^2) \tag{27}$$

where μ_i^{rot}, σ_i^{rot}, μ_i^{acc} and σ_i^{acc} are hyperparameters for the i-th MODA agumentation and we set them as follows: $\mu_i^{rot} = \{\frac{\pi}{64}, -\frac{\pi}{64}, \frac{\pi}{64}, -\frac{\pi}{64}\}$, $\sigma_i^{rot} = \{\frac{\pi}{32}, \frac{\pi}{32}, -\frac{\pi}{32}, -\frac{\pi}{32}\}$ and $\mu_i^{acc} = \{0.5, -0.5, 0.1, -0.1\}$, $\sigma_i^{acc} = \{1, 1, 1, 1\}$. Similar to Eq. (19), we first sample from standard Gaussian distributions and then use the reparameterization trick to map them to $\Delta\hat{\theta}_i(t)$ and $\Delta\hat{d}_i(t)$.

8.4. Train Settings

We implement all models using the PyTorch [36] framework on one NVIDIA GeForce RTX 4090 GPU. PyTorch version is 2.0.0, and CUDA version is 11.8. During the training stage for both HPE and HAR tasks, we use the AdamW [23] optimizer to train all models with a batch size of 4096. The learning rate is initialized to 0.0001 and decayed by 0.99 per epoch.

9. Additional Experimental Results

9.1. Quantitative Results

9.1.1. Semi-Supervised IMU-based HPE

Tab. 10 and Tab. 11 respectively showcase the results of training models on two real datasets (CIP [35] and ANDY [26]), using different ratio of data. It can be observed that, regardless of the ratio, traditional DA techniques underperformed, especially under severe data scarcity (ratio $= 1/32$), where they yield only minimal improvements or even adverse effects. We believe the reason behind this is that these DA methods violate the properties of IMU signals, resulting in the generation of unreliable training samples. In contrast, MODA's augmentation strategy is more aligned with the nature of IMU signals and closely matches their characteristics, thereby outperforming other methods and achieving state-of-the-art results on both benchmarks. Impressively, after applying MODA, training with just $1/8$ of the dataset leads to performance on par with using the entire training set on both two datasets (10.01 *vs.* 11.58 and 5.25 *vs.* 5.20 for Ang Err).

9.1.2. Domain Adaptation IMU-based HPE

The experimental results presented in Tab. 12 and Tab. 13 illustrate that, given the overwhelming presence of synthetic data in the training set, most other DA methods suffer significant performance degradation to varying degrees. Again, we believe this decline stems from a lack of consideration for the inherent characteristics of IMU signals, along with an inability to bridge the domain gap between real and synthetic data. On the contrary, our proposed MODA aligns closely with the properties of IMU signals enhances the *realism* of synthetic data by simulating drift, seamlessly resolving these dual challenges and resulting in a remarkable performance improvement. Specifically, compared to the baseline, positional error decreased by 0.63 cm

Table 10. Comparison with other DA methods in the **Semi-Supervised Learning** scenario, on the CIP [35] dataset.

	CIP (1/8)					CIP (1/16)					CIP (1/32)				
	SIP Err	Ang Err	Pos Err	Mesh Err	Jitter	SIP Err	Ang Err	Pos Err	Mesh Err	Jitter	SIP Err	Ang Err	Pos Err	Mesh Err	Jitter
Baseline	25.35	16.42	9.31	11.58	0.12	25.77	17.45	9.83	11.42	0.14	35.21	37.83	16.41	20.57	0.29
Jittering	23.62	15.50	9.01	10.41	0.12	23.87	16.25	9.33	10.91	0.13	33.93	38.51	16.77	21.78	0.37
Moving Average	24.58	16.53	9.33	11.06	0.21	25.92	17.83	9.95	11.67	0.20	34.77	38.42	16.90	21.70	0.44
Random Mask	23.94	16.23	9.22	10.50	0.11	25.44	16.68	9.62	11.24	0.14	33.28	35.43	15.68	20.33	0.25
Flipping	22.99	13.75	8.44	9.65	0.11	23.95	16.37	8.61	9.83	0.11	33.71	37.78	16.57	22.31	0.24
Slicing	22.40	13.53	8.15	9.43	0.11	23.77	16.09	9.06	10.57	0.12	31.22	36.34	15.07	19.90	0.27
MODA (ours)	**22.14**	**11.58**	**8.08**	**9.07**	**0.06**	**23.28**	**11.97**	**8.31**	**9.28**	**0.06**	**30.74**	**29.29**	**13.49**	**17.57**	**0.13**

Table 11. Comparison with other DA methods in the **Semi-Supervised Learning** scenario, on the ANDY [26] dataset.

	ANDY (1/8)					ANDY (1/16)					ANDY (1/32)				
	SIP Err	Ang Err	Pos Err	Mesh Err	Jitter	SIP Err	Ang Err	Pos Err	Mesh Err	Jitter	SIP Err	Ang Err	Pos Err	Mesh Err	Jitter
Baseline	10.67	7.27	3.90	4.56	0.04	11.85	8.36	4.67	5.56	0.04	11.96	8.87	4.80	5.57	0.05
Jittering	10.95	8.08	4.27	5.00	0.06	13.10	9.99	5.58	6.61	0.06	13.18	11.72	5.76	6.67	0.06
Moving Average	10.96	7.35	4.02	4.67	0.07	12.23	7.85	4.63	5.43	0.06	12.76	9.89	5.27	6.24	0.08
Random Mask	11.75	7.80	4.37	5.09	0.04	13.39	10.73	5.87	7.03	0.06	12.74	11.08	5.97	7.34	0.05
Flipping	10.51	5.79	3.56	4.00	0.03	11.43	7.35	4.22	4.90	0.03	12.06	9.07	4.67	5.47	0.05
Slicing	10.19	5.91	3.53	4.06	0.03	11.45	7.61	4.40	5.07	0.03	12.53	8.72	4.86	5.78	0.04
MODA (ours)	**10.07**	**5.25**	**3.38**	**3.83**	**0.02**	**11.24**	**7.17**	**4.12**	**4.88**	**0.03**	**12.01**	**7.82**	**4.46**	**5.13**	**0.03**

(9%), 1.5 cm (24%), 0.91 cm (11%), and 3.23 cm (34%) across the four real datasets further proving MODA's superiority over other DA methods.

9.1.3. Domain Generalization IMU-based HPE

As shown in Tab. 14 and Tab. 15, *Jitter* soars by roughly 50% across most datasets compared to the Domain Adaptation scenario, with a sharp decline in prediction accuracy (SIP and Angular errors increase by 5-10 degrees). We believe this is due to the fact that models trained exclusively on synthetic data struggle to generalize effectively when applied to real-world datasets, thereby generating highly jittery and inaccurate motion sequences. Similar to the Domain Adaptation scenario, MODA aligns well with the IMU properties and thus does not amplify the domain gap, resulting in superior performance. Specifically, MODA simulates IMU drift to ensure the generated data better reflect real-world data distribution, allowing it to predict accurate and stable motion sequences, outperforming other DA methods by a large margin (The angular error decreases by 3.9, 4.64, 6.05 and 1.15 degrees on the four datasets, respectively).

9.2. Qualitative Results

We also present a **video** that compares motion sequences predicted by the model with various data augmentation methods to highlight MODA's superiority. The visualization results clearly illustrate that MODA consistently outperforms others, keeping prediction accuracy and motion smoothness for both simple and vigorous actions. Please see the **video** for more details.

10. Limitations and Future Work

Magnetic Fields Interference. MODA simulates IMU drifts to generate variations in IMU readings for identical motion sequences. However, another source of variation, external interference from surrounding magnetic fields, is not addressed in this paper. We will explore DA methods for it in future work.

Bias in Motion Data. Since MODA is applied to motion data, it is also affected by their biases. Specifically, while extensive human motion datasets are available, certain actions, such as *slipping* and *falling*, are inherently challenging to capture. We will collect datasets covering more rare and critical motion types in future work.

Table 12. Comparison with other DA methods in the **Domain Adaptation** scenario, on the CIP [35] and ANDY [26] datasets.

	CIP [35]					ANDY [26]				
	SIP Err	Ang Err	Pos Err	Mesh Err	Jitter	SIP Err	Ang Err	Pos Err	Mesh Err	Jitter
Baseline	17.02	12.06	6.99	8.31	0.12	15.99	9.89	6.15	7.17	0.05
Jittering	19.39	14.37	7.98	9.42	0.11	16.88	12.07	7.45	8.90	0.04
Moving Average	20.61	16.09	8.98	10.84	0.15	16.63	11.95	7.15	8.66	0.06
Random Mask	19.71	14.50	8.13	9.51	0.10	17.19	12.88	8.02	9.57	0.04
Flipping	19.31	14.18	8.08	9.46	0.10	16.51	11.45	7.52	8.86	0.04
Slicing	19.29	14.22	8.07	9.47	0.10	16.53	11.48	7.50	8.02	0.04
MODA (ours)	**15.27**	**11.04**	**6.36**	**7.54**	**0.10**	**12.22**	**8.17**	**4.65**	**5.51**	**0.04**

Table 13. Comparison with other DA methods in the **Domain Adaptation** scenario, on the DIP [15] and TotalCapture [46] datasets.

	DIP [15]					TotalCapture [46]				
	SIP Err	Ang Err	Pos Err	Mesh Err	Jitter	SIP Err	Ang Err	Pos Err	Mesh Err	Jitter
Baseline	20.54	14.95	8.58	10.24	0.10	20.48	16.69	9.40	10.76	0.12
Jittering	21.43	16.27	8.91	10.77	0.09	20.49	16.87	9.43	10.79	0.11
Moving Average	24.36	18.70	10.36	12.51	0.11	22.98	18.17	10.43	11.81	0.14
Random Mask	22.65	17.13	9.39	11.23	0.08	20.49	16.52	9.45	10.74	0.11
Flipping	21.82	16.24	9.14	10.78	0.07	19.51	16.15	9.00	10.34	0.10
Slicing	21.99	16.40	9.22	10.87	0.08	19.51	16.16	9.01	10.34	0.10
MODA (ours)	**19.43**	**13.54**	**7.67**	**9.19**	**0.07**	**14.24**	**12.54**	**6.17**	**7.23**	**0.10**

Table 14. Comparison with other DA methods in the **Domain Generalization** scenario, on the CIP [35] and ANDY [26] datasets.

	CIP [35]					ANDY [26]				
	SIP Err	Ang Err	Pos Err	Mesh Err	Jitter	SIP Err	Ang Err	Pos Err	Mesh Err	Jitter
Baseline	29.05	18.39	9.78	11.35	0.25	25.39	16.43	9.64	11.36	0.10
Jittering	29.57	18.53	9.99	11.58	0.22	27.48	16.93	10.13	11.48	0.10
Moving Average	31.59	20.06	10.82	12.56	0.35	28.35	17.00	10.11	11.47	0.13
Random Mask	30.29	18.78	10.08	11.65	0.23	26.36	16.55	9.68	11.05	0.09
Flipping	30.29	18.42	9.88	11.51	0.26	26.12	16.83	9.66	11.50	0.10
Slicing	30.30	18.49	9.25	11.53	0.26	26.12	16.86	9.70	11.50	0.10
MODA (ours)	**27.78**	**14.59**	**9.17**	**10.12**	**0.14**	**22.26**	**11.79**	**7.72**	**7.20**	**0.05**

Table 15. Comparison with other DA methods in the **Domain Generalization** scenario, on the DIP [15] and TotalCapture [46] datasets.

	DIP [15]					TotalCapture [46]				
	SIP Err	Ang Err	Pos Err	Mesh Err	Jitter	SIP Err	Ang Err	Pos Err	Mesh Err	Jitter
Baseline	29.24	19.46	10.30	12.52	0.17	26.28	17.58	8.74	10.02	0.25
Jittering	28.72	19.51	10.46	12.55	0.17	25.29	18.04	9.40	10.62	0.22
Moving Average	30.97	21.34	11.39	13.68	0.25	28.62	19.49	10.15	11.48	0.30
Random Mask	28.77	19.86	10.36	12.52	0.17	28.65	18.87	9.75	11.23	0.22
Flipping	29.17	19.03	9.84	12.07	0.18	25.80	17.99	9.33	10.50	0.25
Slicing	29.17	19.19	9.90	12.07	0.18	25.90	18.15	9.40	10.57	0.25
MODA (ours)	**28.37**	**13.41**	**8.25**	**9.44**	**0.10**	**25.42**	**16.43**	**8.23**	**9.38**	**0.12**