MetaCap: Meta-learning Priors from Multi-View Imagery for Sparse-view Human Performance Capture and Rendering
– Supplemental Document –

Guoxing Sun[1], Rishabh Dabral[1], Pascal Fua[2], Christian Theobalt[1],
Marc Habermann[1]

[1] Max Planck Institute for Informatics, Saarland Informatics Campus
[2] EPFL
{gsun,rdabral,theobalt,mhaberma}@mpi-inf.mpg.de, pascal.fua@epfl.ch
https://vcai.mpi-inf.mpg.de/projects/MetaCap/

1 Overview

To facilitate a more comprehensive analysis and understanding of MetaCap and experiment configurations, we offer additional results (Sec. 2), method details (Sec. 3, 4), implementation details (Sec. 5), method costs (Sec. 6), comparisons (Sec. 7, 8), ablations (Sec. 9), applications (Sec. 10), and temporal results (Sec. 11).

2 More Results on Different Poses and Subjects

Fig. 1 presents additional qualitative results showcasing the performance of our method across diverse motions and subjects. Since our method learns the meta prior in the canonical pose space, it is robust to various testing poses.

3 Template Model

We revisit two types of human template, SMPL [7] and DDC [4] and demonstrate how to compute the transformation matrix and deformed position for each vertex, which are crucial for the canonicaliztion step. Here, each template has vertices $\bar{\mathbf{X}} \in \mathbb{R}^{V\times 3}$ in canonical pose $\bar{\mathcal{M}} = \{\bar{\boldsymbol{\theta}}, \bar{\boldsymbol{\alpha}}, \bar{\boldsymbol{z}}\}$. $\bar{\boldsymbol{\theta}}, \bar{\boldsymbol{\alpha}}, \bar{\boldsymbol{z}}$ represents the joint rotations, the root rotation, and the root translation of canonical pose, respectively. We define a window size W skeletal motion at time f as $\mathcal{M}_{f,W} = \{\boldsymbol{\theta}_{f-W}, \boldsymbol{\alpha}_{f-W}, \boldsymbol{z}_{f-W}, ..., \boldsymbol{\theta}_f, \boldsymbol{\alpha}_f, \boldsymbol{z}_f\}$. If W is not explicitly set, it defaults to 1.

3.1 Parametric Template–SMPL

SMPL [7] is a parametric human body model. It characterizes V vertices and J joint positions of the human mesh using shape parameters, β, canonical

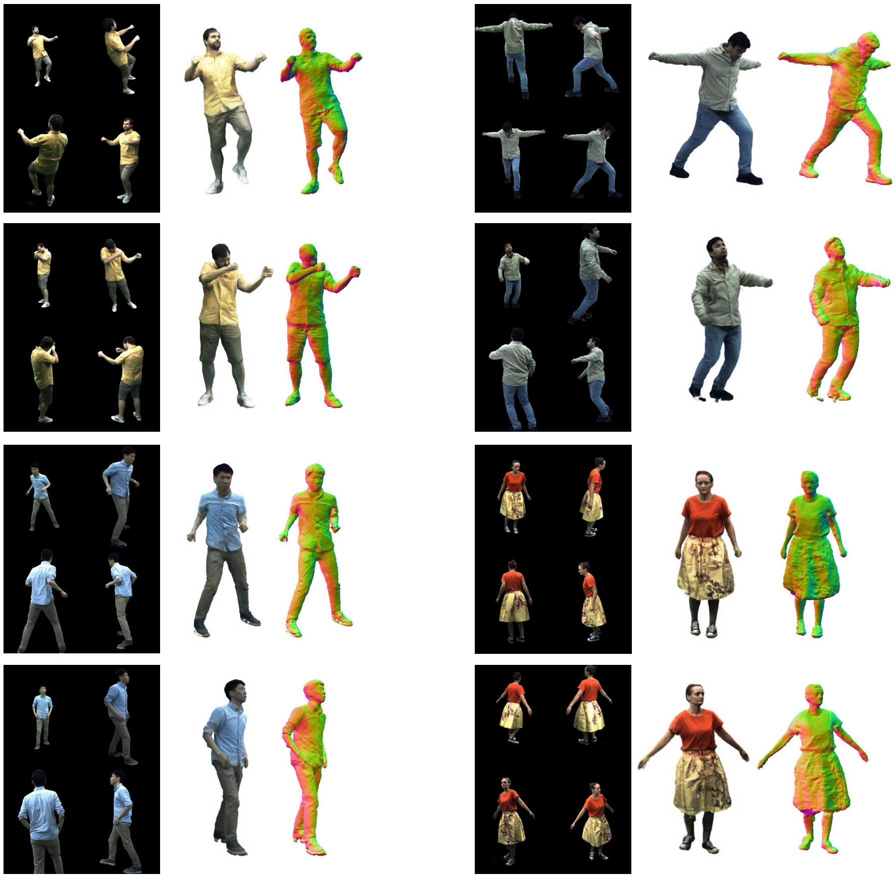

Fig. 1: Qualitative Results. Here, we showcase additional qualitative results of MetaCap utilizing four-view images as inputs, demonstrating its robustness across diverse poses and different subjects.

pose parameters, $\bar{\mathcal{M}}$, and pose parameters, $\mathcal{M}$. The overall mesh deformation, $\mathbf{T}_{\mathrm{def}}(\beta, \mathcal{M})$, is determined by the sum of shape dependent displacements and pose dependent displacements. Linear Blend Skinning (LBS) is employed for animating the deformed mesh:

$$\mathbf{T}_{\mathrm{FK},i}(\mathcal{M}, \bar{\mathcal{M}}) = \sum_{j=1}^{J} w_{j,i} T_j(\mathcal{M}) (\sum_{j=1}^{J} w_{j,i} T_j(\bar{\mathcal{M}}))^{-1} \tag{1}$$

$$\mathbf{T} = \mathbf{T}_{\mathrm{FK}}(\mathcal{M}, \bar{\mathcal{M}}) \mathbf{T}_{\mathrm{def}}(\beta, \mathcal{M}) \tag{2}$$

$$\mathbf{X} = \mathbf{T}\bar{\mathbf{X}} \tag{3}$$

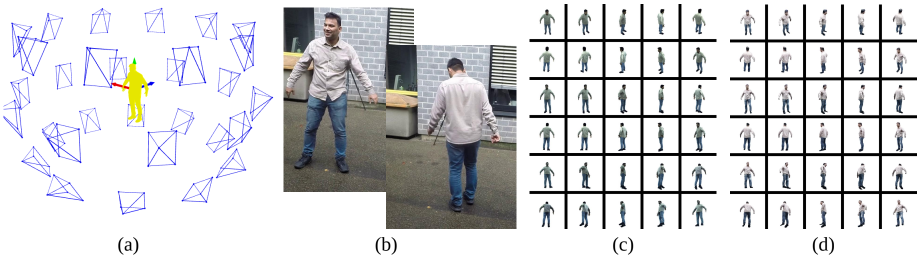

Fig. 2: Illustrations of image proxy. (a) Visualization of the camera distribution for rendering the proxy images. (b) Monocular frames used to generate in-the-wild image proxy. (c) Proxy images in the dome. (d) Proxy images in the wild. Best viewed with zoom.

where $w_{j,i}$ is the blend weight from joint j to vertex i, $T_j(\mathcal{M})$ denotes the joint j's local transformation, $\mathbf{T}_{\mathrm{FK},i}(\mathcal{M},\bar{\mathcal{M}})$ represents the global transformation of the deformed vertex i from canonical pose $\bar{\mathcal{M}}$ to the pose $\mathcal{M}$.

3.2 Deformable Template–DDC

DDC [4] is a personalized deformable body model. It models motion-dependent body deformation with embedded graph deformation and vertex displacements. Given a window size W skeletal motion $\mathcal{M}_{f,W}$ at time f, it employs Graph Convolutional Networks (GCN) [21] to estimate the embedded-graph's deformation parameters $\mathbf{A},\mathbf{T} \in \mathbb{R}^{K\times 3}$ and the per-vertex displacements d. The final geometry is obtained through Dual Quaternion Skinning (DQS):

$$\mathbf{T}_{\mathrm{def},i}(\mathcal{M}_{f,W}) = \sum_k \mathrm{w}_{k,i} \left[\begin{array}{c|c} R(\mathbf{a}_k) & d_i + (I - R(\mathbf{a}_k))\mathbf{g}_k + \mathbf{t}_k \\ \hline \overrightarrow{0} & 1 \end{array}\right] \tag{4}$$

$$\mathbf{T} = \mathbf{T}_{\mathrm{FK}}(\mathcal{M}_f,\bar{\mathcal{M}})\mathbf{T}_{\mathrm{def}}(\mathcal{M}_{f,W}) \tag{5}$$

$$\mathbf{X} = \mathbf{T}\bar{\mathbf{X}} \tag{6}$$

where $\mathrm{w}_{k,i}$ represents the weight from graph node k to vertex i, $R(\cdot)$ transforms Euler representations into matrix representations, $\mathbf{a}_k$ and $\mathbf{t}_k$ are node k's local rotation and translation, $\mathbf{g}_k$ denotes the position of node k. $\mathbf{T}_{\mathrm{FK}}(\mathcal{M}_f,\bar{\mathcal{M}})$ represents global transformation from initial pose $\bar{\mathcal{M}}$ to pose $\mathcal{M}_f$. In the implementation of DDC, the window size is set to 3. Note that, $\mathcal{M}_f$ represents the pose at frame f, and the motion window of it is 1.

4 Proxy Image Generation for Occlusion Handling

We propose the occlusion handling to address missing information when occlusion happens. To offer additional information for the occluded areas, we render proxy images of the human in canonical pose space. Depending on the lighting condition and camera setup, we have two configurations: in-the-dome and in-the-wild. Here, we demonstrate the details of proxy image generation (see Fig. 2).

4.1 Proxy Image Generation in the Dome

We have dense-view cameras for the in-the-dome case. Consequently, we select one frame to reconstruct its geometry and texture with space canonicalization. This enables us to render novel-view in-the-dome proxy images in the canonical space (see Fig. 2 (c)).

4.2 Proxy Image Generation in the Wild

Under the in-the-wild scenarios, where sparse-view or monocular cameras are predominant, occlusion handling becomes particularly crucial, especially in monocular scenarios. Consider the most challenging scenario, namely the monocular camera setup. In such situations, it is not feasible to directly reconstruct geometry and appearance like in-the-dome scenario. Instead, leveraging the capabilities of the meta prior, we fine-tune multiple frames (see Fig. 2 (b)) into a unified canonical space simultaneously to construct a complete human representation. We then render it into novel-view in-the-wild proxy images in the canonical space (see Fig. 2 (d)).

5 Implementation Details

In this section, we present implementation details of MetaCap (Sec. 5.1), implementation details of methods that we compare with (Sec. 5.2), ablations (Sec. 5.3), and comparison on in-the-wild sequences (Sec. 5.4). Fig. 3 illustrates the camera distribution when conducting the prior learning, fine-tuning and evaluation in the comparisons. These three sets of cameras do not overlap. Additionally, the motions used for prior learning and fine-tuning are distinct.

5.1 MetaCap

Space Canonicalization. During comparison, we utilize the deformable template DDC [4] as the default template for space canonicalization. To acquire the template and the deformation parameters, we first create a character with smoothed template, embedded graph, skeleton and default motion. Subsequently, we follow the methodology outlined in [4] to implement multi-view silhouette supervision using a differentiable renderer [1] and distance transformation [3]. For the loose-cloth subject 'S5', we apply additional Chamfer loss supervision. For parametric models SMPL [7] and SMPL-X [11] used in the ablations and comparison methods, we first obtain 3D marker positions by animating the character's skeleton with the same motions used in the deformable template. We then utilize EasyMocap [2] to estimate the shape and pose parameters.

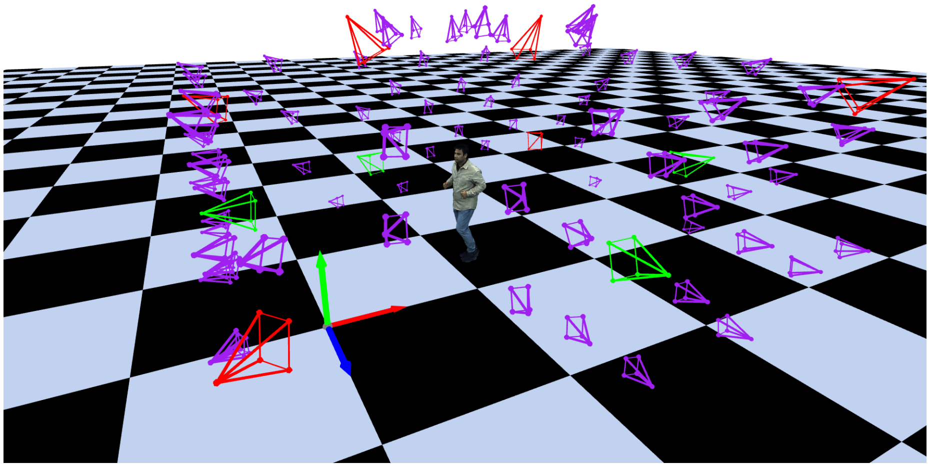

Fig. 3: Visualization of the camera distribution for prior learning, inference and evaluation in the comparison with state-of-the-art methods. Dense-view cameras with purple color are the training views used for prior learning. Four-view cameras with green color are the input views during inference. Six-view cameras with red color are the evaluation views.

Meta-learning and Fine-tuning. During the meta-learning stage, we use approximately 100-frame images captured by 100-view cameras, paired with human template at each frame. We apply SGD [13] with $l^{out} = 1.0$ to the outer loop and Adam [6] with $l^{in} = 1e{-}4$ to the inner loop. In each outer loop sampling step, we randomly sample $M = 24$ camera views and rays on each image simultaneously. After $M = 24$ unrolled gradient steps in the inner loop, we follow Reptile [8] to update the outer loop weights. The inner loop is warmed up by linearly updating learning rate from 1% to 100% for the first 50 outer loop steps. The template threshold η is set to 0.05 for 'S2' and 'S5', and to 0.01 for 'S3' and 'S27', with a threshold decay to 50% applied after 300 outer loop steps. The total number of meta-learning outer loop steps is 2000. The input images are resized to 50% and applied Gaussian blur with a $5 * 5$ kernel. The weights of loss functions are set as $\lambda_{\mathrm{c}} = 10.0, \lambda_{\mathrm{e}} = 0.1, \lambda_{\mathrm{m}} = 0.1, \lambda_{\mathrm{s}} = 0.01$.

During the fine-tuning stage, we load the meta-learned weights and apply the Adam optimizer [6] with learning rate $lr = 1e - 4$, $\beta_1 = 0.9$, $\beta_1 = 0.99$, $\epsilon = 1e - 15$ to fine-tune weights for 3000 steps. In each step, we randomly sample 8192 rays from all input observations. The template threshold η is set 0.05. There's no warm-up in this stage. The weights of loss functions are set as $\lambda_{\mathrm{c}} = 10.0, \lambda_{\mathrm{e}} = 0.1, \lambda_{\mathrm{m}} = 0.1, \lambda_{\mathrm{s}} = 0.01$.

5.2 Comparison Methods

DeepMultiCap. As DeepMultiCap [22] is trained on a large scale human scan dataset and exhibits generalization ability, we utilize the official checkpoint without additional fine-tuning. It relies on SMPL-X as the template model to provide geometry and global normal maps. Following the template procedure outlined earlier, we fit SMPL-X and subsequently render it to produce normal maps.

DiffuStereo. Official DiffuStereo [16] utilizes geometry results from DoubleField [15] for initializing the disparity maps. Since DoubleField [15] is not open source, we employ the deformable template [4] as a substitute for initializing the disparity maps, as it contains rough geometry information. Subsequently, we use the official checkpoint trained with 20-degree angle images to refine the disparity maps. Due to the large camera baseline from 4-view cameras, the output point-clouds are often incomplete. Therefore, we incorporate additional template point-clouds to complete the mesh when applying the Possion surface reconstruction [5].

Drivable Volumetric Avatars (DVA). The dataset division of DVA [12] is the same as ours, including human template, training multi(dense)-view images and testing sparse-view images. We utilize the official code with our estimated SMPL-X parameters. We train the personalized DVA model using images from dense-view training set. At the testing stage, we adhere to the original paper's manner that no fine-tuning added, and employ sparse-view images and template to render novel view images. Given that DVA does not focus on geometry reconstruction, we extract their estimated primitive parameters, convert them to box meshes, and use Possion surface reconstruction [5] to reconstruct the final watertight mesh.

TransHuman. TransHuman [9] is trained on multi-view videos with multiple subjects. However, We found that directly applying the official pre-trained checkpoint on our data yields low-quality results. Therefore, for each subject, we fine-tune them individually on our training set, and generate testing set results without additional fine-tuning.

ARAH. ARAH [18] incorporates a meta prior [17] trained from a large scale scan dataset. In our implementation, we use the official checkpoint as initialization and further fine-tune it with all the frames in the testing set. It's worth noting that other methods only utilize 1-frame sparse-view images as input during inference.

5.3 Ablations

Weight Initialization and Space Canonicalization. During this ablation study, we maintain the camera setup consistent with the comparison section.

Specifically, we utilize dense-view cameras for prior learning and four-view cameras for fine-tuning.

We have three types of network initialization consisting of two baseline methods random weights, pre-trained weights and our meta weights. Random weight initialization utilizes the default weight initialization from PyTorch [10]. Pre-trained weights are trained on the same views and frames as meta-learning. It's implemented by setting $M = 1$ in meta-learning process and training for 1000 steps. When performing the fine-tuning with random initialization and pre-trained initialization, we reserve 500 warm-up steps. Meta weights are obtained following the procedure outlined in Sec. 5.1.

In terms of space canonicalization, we employ three types: root canonicalization, SMPL canonicalization, and DDC canonicalization. Root canonicalization is implemented by transforming world-space points to canonical space with the transformation of root joint of human from motions. With SMPL template and SMPL motion parameters, we perform a more fine-grained canonicalization by determining the transformation of each world-space point to the nearest points. When using DDC as the template, the canonicalization procedure is similar to SMPL template, but the transformation computation is adjusted for DDC.

Number of Camera Views and Occlusion Handling. In this ablation study, we investigate the impact of different camera views and occlusion handling. Specifically, we utilize DDC as the template for space canonicalization and meta weights for weight initialization.

We first evaluate the effect of varying camera numbers in the meta-learning phase. We utilize 4-view cameras for fine-tuning, while we experiment with different camera numbers in meta prior learning: 1, 2, 4, 8, and dense.

Next, we evaluate the influence of camera numbers in the fine-tuning phase. Here, we utilize dense-view cameras for prior-learning but different camera numbers in the fine-tuning: 1, 2, 4, 8. Additionally, we examine the influence of occlusion handling (OH) by employing this strategy during monocular fine-tuning.

Convergence Speed and Quality We aim to investigate the the performance of convergence when using different weight initializations. The camera setup remains consistent with the comparison section and we utilize DDC as the template for space canonicalization. For dynamic evaluation during fine-tuning, we select a single frame from testing data. The corresponding qualitative results of the curves are presented in the supplementary video.

5.4 Comparison on In-the-wild Sequences

During the comparison on the in-the-wild sequences, we have four views to provide inputs and an additional view to offer ground truth images. We follow the Sec. 5.1 and 5.2 to implement our method and ARAH.

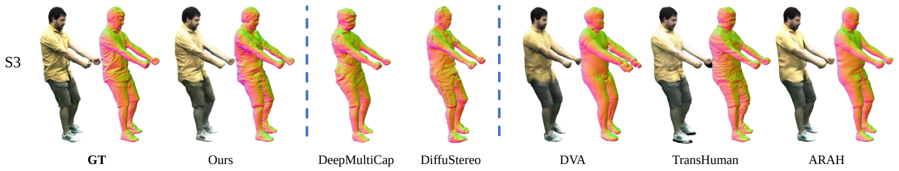

Fig. 4: Qualitative Comparison. We additionally compare our method with other approaches on S3 of DynaCap dataset. Our method demonstrates superior performance in geometry capturing and rendering quality.

Table 1: Quantitative Comparison. For the S3 from DynaCap dataset, our method still achieves state-of-the-art results for novel-view synthesis and geometry reconstruction. *Note, that ARAH requires 4D scans for meta learning and videos for the fine-tuning whereas other methods solely require static images.

Method	Subject	Appearance			Geometry				
		PSNR ↑	SSIM ↑	LPIPS ↓	NC-Cos ↓	NC-L2 ↓	Chamfer ↓	P2S ↓	IOU ↑
DeepMultiCap [22]	S3	-	-	-	0.131	0.425	1.137	1.158	0.717
DiffStereo [16]	S3	-	-	-	0.143	0.441	1.169	1.269	0.818
DVA [12]	S3	24.862	0.824	0.284	0.109	0.378	1.593	2.119	0.465
TransHuman [9]	S3	25.136	0.826	0.277	0.118	0.393	1.477	2.006	0.797
ARAH* [18]	S3	25.093	0.842	0.278	0.069	0.294	0.780	0.836	0.866
Ours	S3	25.528	0.839	0.251	0.106	0.382	0.671	0.792	0.908

6 Learning and Testing Cost Against ARAH

We use 100 frames for the meta prior learning, it takes around 5-6 hours on a single GPU. When conducting fine-tuning, our method takes between 40 seconds and 3 minutes for one frame with one GPU. The prior used in ARAH is from MetaAvatar [17], which uses 10-48 hours for the prior learning in a single GPU. During fine-tuning, ARAH [18] takes around 16 hours for 90 frames with 4 GPUs, i.e. around 10 minutes per frame. Thus, our method is significantly faster than ARAH during meta-learning and fine-tuning.

7 Additional Comparisons on S3

Fig. 4 and Tab. 1 present additional qualitative and quantitative results on the 'S3' from DynaCap Dataset [4]. The implementations of the methods are consistent with those in the 'Results' section. Our approach continues to outperform other methods in both rendering and reconstruction.

8 Additional Comparisons on Monocular Methods

Fig. 5 shows additional qualitative comparisons between our method and monocular reconstruction methods. Here, we initialize our network with the meta prior and fine-tune it using monocular input images, with occlusion handling

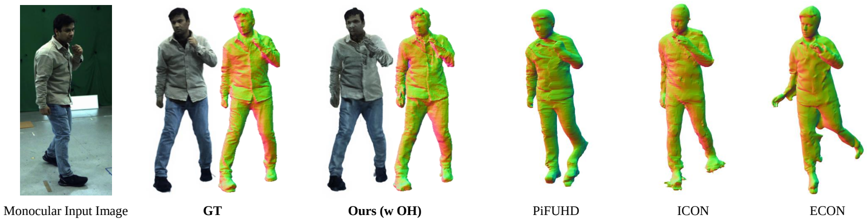

Fig. 5: Qualitative Comparison. In this comparison, we compare our method, which involves monocular fine-tuning with occlusion handling, against other monocular reconstruction approaches, namely PiFUHD [14], ICON [20], and ECON [19]. Our method exhibits robustness to the human pose and camera pose, and produces superior geometry and appearance capture.

Table 2: Quantitative Ablation. Here, we study the influence of motion tracking quality on our method. Comparing to dense mocap, our method with sparse mocap exhibits a slight decrease in performance.

Method	Motion	Subject	Appearance			Geometry				
			PSNR ↑	SSIM ↑	LPIPS ↓	NC-Cos ↓	NC-L2 ↓	Chamfer ↓	P2S ↓	IOU ↑
ARAH*	Dense	S2	26.279	0.833	0.302	0.079	0.315	0.839	0.913	0.859
Ours	Sparse	S2	26.240	0.836	0.253	0.102	0.362	0.712	0.840	0.883
Ours	Dense	S2	26.529	0.841	0.249	0.096	0.351	0.679	0.814	0.887

applied. Our approach employs perspective camera projection, enabling human reconstruction in real-world scale and coordinates. In contrast, PiFUHD [14], ICON [20], and ECON [19] utilize orthogonal camera projection. PiFUHD [14] exhibits sensitivity to both human and camera poses. ICON [20] demonstrates limited generalization ability. ECON [19] predicts normal maps for the front and back sides, and integrates them onto SMPL template. The predicted normal maps may lack accuracy or fail easily. Our method yields reasonable results by fine-tuning the canonical space human fields. In the 'Comparison' section, our method outperforms the multi-view method DeepMultiCap [22], which presents superior results to multi-view PiFUHD.

9 Ablation on Motion Capture Quality

To evaluate the influence of motion capture quality to our method, we replace motions from dense mocap with sparse mocap and generate rendering and reconstruction results. The sparse motions come from the same four-view camera setup used for fine-tuning, while the dense motions are estimated from 34 cameras in the dome. Tab. 2 demonstrates that, though the perfomance drops a bit, our method with sparse mocap still produces comparable rendering quality and better geometry compared to ARAH [18] with dense mocap.

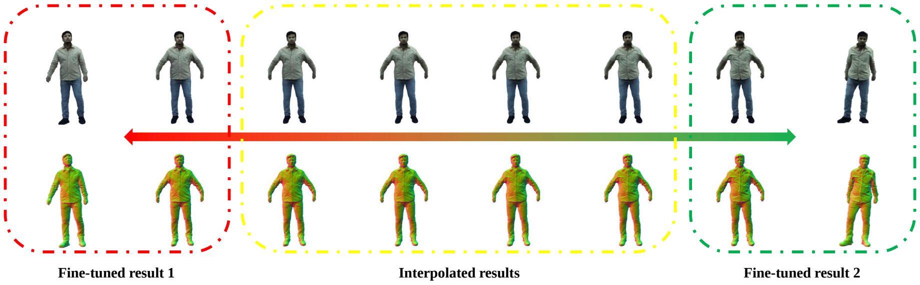

Fig. 6: Appearance and geometry interpolation on two fine-tuned results. The red and green boxes represent the appearance and geometry of different frames' fine-tuned results displayed in both world space and canonical space.

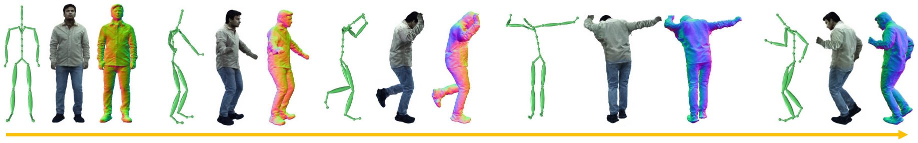

Fig. 7: Animating our four-view fine-tuned results over time. Our hybrid representation can be easily animated with motion and corresponding template.

10 Applications

10.1 Interpolation in Weight Space

Thanks to the space canonicalization and meta initialization, we are able to linearly interpolate results from different frames in the weight (hyper) space and produce meaningful novel interpolated appearance and geometry results, as shown in Fig. 6. This experiment further validates our hypothesis that space canonicalization narrows the range of spatial features and facilitates meta prior learning.

10.2 Animating the Fine-tuned Results

After fine-tuning our meta prior with four-view images, we obtain a canonicalized hybrid human avatar. This avatar can be easily animated with novel motions and corresponding deformable template, like Fig. 7. The animated results maintain photorealistic appearance and high-quality geometry.

11 Temporal Fine-tuned Results

Fig. 8 shows fine-tuned results on a temporal sequence. As our method is not designed for temporal inputs, we generate these results by a frame-by-frame fine-tuning. Please refer to the project page for the video display.

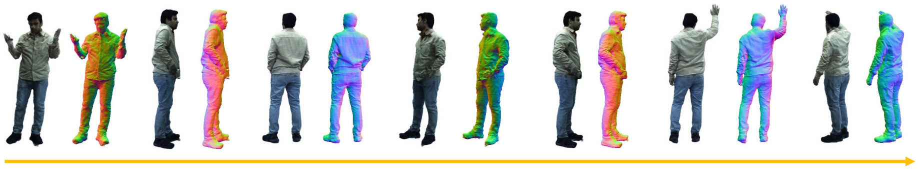

Fig. 8: Fine-tuned results on a temporal sequence using a frame-by-frame manner.

References

1. https://github.com/ayushtewari/GVV-Differentiable-CUDA-Renderer
2. https://github.com/zju3dv/EasyMocap
3. Fabbri, R., Costa, L.D.F., Torelli, J.C., Bruno, O.M.: 2d euclidean distance transform algorithms: A comparative survey. ACM Computing Surveys (CSUR) **40**(1), 1–44 (2008)
4. Habermann, M., Liu, L., Xu, W., Zollhoefer, M., Pons-Moll, G., Theobalt, C.: Real-time deep dynamic characters. ACM Transactions on Graphics **40**(4) (aug 2021)
5. Kazhdan, M., Bolitho, M., Hoppe, H.: Poisson surface reconstruction. In: Proceedings of the fourth Eurographics symposium on Geometry processing. vol. 7, p. 0 (2006)
6. Kingma, D.P., Ba, J.: Adam: A method for stochastic optimization (2017)
7. Loper, M., Mahmood, N., Romero, J., Pons-Moll, G., Black, M.J.: SMPL: A skinned multi-person linear model. ACM Trans. Graphics (Proc. SIGGRAPH Asia) **34**(6), 248:1–248:16 (Oct 2015)
8. Nichol, A., Achiam, J., Schulman, J.: On first-order meta-learning algorithms. arXiv preprint arXiv:1803.02999 (2018)
9. Pan, X., Yang, Z., Ma, J., Zhou, C., Yang, Y.: Transhuman: A transformer-based human representation for generalizable neural human rendering. In: Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV). pp. 3544–3555 (October 2023)
10. Paszke, A., Gross, S., Chintala, S., Chanan, G., Yang, E., DeVito, Z., Lin, Z., Desmaison, A., Antiga, L., Lerer, A.: Automatic differentiation in pytorch (2017)
11. Pavlakos, G., Choutas, V., Ghorbani, N., Bolkart, T., Osman, A.A.A., Tzionas, D., Black, M.J.: Expressive body capture: 3D hands, face, and body from a single image. In: Proceedings IEEE Conf. on Computer Vision and Pattern Recognition (CVPR). pp. 10975–10985 (2019)
12. Remelli, E., Bagautdinov, T., Saito, S., Wu, C., Simon, T., Wei, S.E., Guo, K., Cao, Z., Prada, F., Saragih, J., et al.: Drivable volumetric avatars using texel-aligned features. In: ACM SIGGRAPH 2022 Conference Proceedings (2022)
13. Ruder, S.: An overview of gradient descent optimization algorithms. arXiv preprint arXiv:1609.04747 (2016)
14. Saito, S., Simon, T., Saragih, J., Joo, H.: Pifuhd: Multi-level pixel-aligned implicit function for high-resolution 3d human digitization. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 84–93 (2020)
15. Shao, R., Zhang, H., Zhang, H., Chen, M., Cao, Y., Yu, T., Liu, Y.: Doublefield: Bridging the neural surface and radiance fields for high-fidelity human reconstruction and rendering. In: CVPR (2022)

16. Shao, R., Zheng, Z., Zhang, H., Sun, J., Liu, Y.: Diffustereo: High quality human reconstruction via diffusion-based stereo using sparse cameras. In: European Conference on Computer Vision. pp. 702–720. Springer (2022)
17. Wang, S., Mihajlovic, M., Ma, Q., Geiger, A., Tang, S.: Metaavatar: Learning animatable clothed human models from few depth images. In: Advances in Neural Information Processing Systems (2021)
18. Wang, S., Schwarz, K., Geiger, A., Tang, S.: Arah: Animatable volume rendering of articulated human sdfs. In: European Conference on Computer Vision (2022)
19. Xiu, Y., Yang, J., Cao, X., Tzionas, D., Black, M.J.: Econ: Explicit clothed humans optimized via normal integration. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 512–523 (2023)
20. Xiu, Y., Yang, J., Tzionas, D., Black, M.J.: ICON: Implicit Clothed humans Obtained from Normals. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR). pp. 13296–13306 (June 2022)
21. Zhang, S., Tong, H., Xu, J., Maciejewski, R.: Graph convolutional networks: a comprehensive review. Computational Social Networks **6**(1), 1–23 (2019)
22. Zheng, Y., Shao, R., Zhang, Y., Yu, T., Zheng, Z., Dai, Q., Liu, Y.: Deepmulticap: Performance capture of multiple characters using sparse multiview cameras. In: Proceedings of the IEEE/CVF International Conference on Computer Vision. pp. 6239–6249 (2021)