DeepCap: Monocular Human Performance Capture Using Weak Supervision – Supplemental Document –

Marc Habermann[1,2] Weipeng Xu[1,2] Michael Zollhoefer[3] Gerard Pons-Moll[1,2] Christian Theobalt[1,2]

[1]Max Planck Institute for Informatics, [2]Saarland Informatics Campus, [3]Stanford University

In the following, we provide more details on the creation of the character model (Sec. 1) and the training data (Sec. 2). Further, we explain our two networks, *PoseNet* (Sec. 3) and *DefNet* (Sec. 4) in more detail and provide implementation details (Sec. 5). Finally, we show more qualitative and quantitative results and comparisons (Sec. 6).

1. Character Model

Template Mesh. The template acquisition can be fully automated. To create the textured mesh (see Fig. 1), we capture the person in a static T-pose with an RGB-based scanner[1] which has 134 RGB cameras resulting in 134 images $\mathcal{I}_{\text{rec}} = \{I_{\text{rec}_1}, \cdots, I_{\text{rec}_{134}}\}$. The textured 3D geometry is obtained by leveraging a commercial 3D reconstruction software, called Agisoft Metashape[2], that takes as input the images $\mathcal{I}_{\text{rec}}$ and reconstructs a textured 3D mesh of the person (see Fig. 1). We apply Metashape's mesh simplification to reduce the number of vertices N and Meshmixer's[3] remeshing to obtain roughly uniform shaped triangular surfaces. Similar to [4], we segment the mesh into different non-rigidity classes resulting in per-vertex rigidity weights s_i for each vertex i.

Embedded Deformation Graph To obtain the embedded graph $\mathcal{G} = \{\mathbf{A}, \mathbf{T}\}$ where $\mathbf{A}, \mathbf{T} \in \mathbb{R}^{K\times 3}$ with K graph nodes, we further decimate the template mesh to around 500 vertices (see Fig. 1). The connections of a node k to neighboring nodes are given by the vertex connections of the decimated mesh and are denoted as the set $\mathcal{N}_{\text{n}}(k)$. For each vertex of the decimated mesh we search for the closest vertex on the template mesh in terms of Euclidean distance. These points then define the position of the graph nodes $\mathbf{G} \in \mathbb{R}^{K\times 3}$ where $\mathbf{G}_k$ is the position of node k. To compute the vertex-to-node weights $w_{i,k}$, we measure the geodesic distance between the graph node k and the template vertex i and denote the set of nodes that influence vertex i as $\mathcal{N}_{\text{vn}}(i)$.

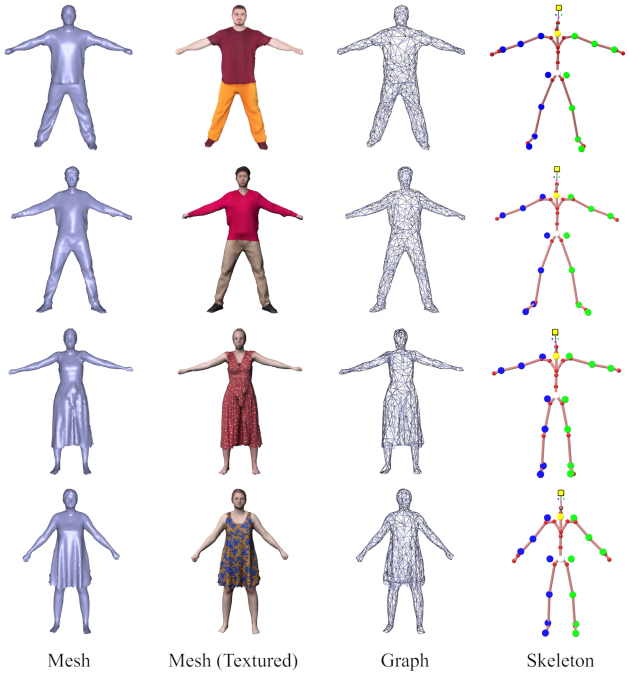

Figure 1. Character models. Here, we show the character model of $S1$ to $S4$ (top to bottom) of our new dataset. It consists of the textured mesh, the underlying embedded deformation graph as well as the attached skeleton.

Skeleton. Next, we automatically fit the skeleton (see Fig. 1) to the 3D mesh by fitting the SMPL model [8]. To this end, we first optimize the pose by performing a sparse non-rigid ICP where we use the head, hands and feet as feature points since they can be easily detected in a T-pose. Then, we perform a dense non-rigid ICP on vertex level to obtain the final pose and shape parameters. For clothing types that roughly follow the human body shape, e.g., pants and shirt, we propagate the per-vertex skinning weights of the naked SMPL model to the template vertices.

[1] https://www.treedys.com/
[2] http://www.agisoft.com
[3] http://www.meshmixer.com/

For other types of clothing, like skirts and dresses, we leverage Blenders's[4] automated skinning weight computation. Our skeleton consists of 27 joint angle parameters, 23 joints and 21 attached landmarks (17 body and 4 face landmarks). The landmark placement follows the convention of OpenPose [3, 2, 11, 12].

2. Training Data Acquisition

To acquire the training data for our method, we ask the person to perform a large set of different and challenging motions in a green screen studio. The number of frames per subject varies between 26000 and 38000 depending on how fast the person performed all the motions. We used C calibrated and synchronized cameras with a resolution of 1024×1024 for capturing where for all subjects we used between 11 and 14 cameras. The original image resolution is too large to transfer all the distance transform images to the GPU during training. Fortunately, most of the image information is anyways redundant since we are only interested in the image region where the person is. Therefore, we crop the distance transform images using the bounding box that contains the segmentation mask with a conservative margin. Finally, we resize it to a resolution of 350×350 without loosing important information.

3. PoseNet – Additional Explanations

Sparse Keypoint Loss. λ_m is a hierarchical re-weighting factor that varies during training for better convergence. More precisely, for the first one third of the training iterations per training stage (see Sec. 5) for *PoseNet*, we multiply the keypoint loss with a factor of $\lambda_m = 3$ for all torso markers and with a factor of $\lambda_m = 2$ for elbow and knee markers. For all other markers, we set $\lambda_m = 1$. For the remaining iterations, we set $\lambda_m = 3$ for all markers. This re-weighting allows us to let the model first focus on the global rotation (by weighting torso markers higher than others). We found that this gives better convergence during training and joint angles overshoot less often, especially at the beginning of training.

4. DefNet – Additional Explanations

Non-rigid Silhouette Loss. $\mathcal{B}_c$ is the set of vertices that lie at the boundary when the deformed 3D mesh is projected into the camera frame c. Those vertices are computed by rendering a depth map using a custom CUDA-based rasterizer that can be easily integrated into deep learning architectures as a separate layer. The vertices that project onto a depth discontinuity (background vs. foreground) in the depth map are treated as boundary vertices.

[4] https://www.blender.org/

5. Implementation Details

Both network architectures as well as the GPU-based custom layers are implemented in the Tensorflow framework [1]. We use the Adam optimizer [7] in all our experiments.

Training Strategy for PoseNet. As we are interested in joint angle regression, one has to note that multiple solutions for the joint angles exist due the fact that every correct solution can be multiplied by 2π leading to the same loss value. To this end, training has to be carefully designed. In general, our strategy first focuses on the torso markers by giving them more weight (see Sec. 3). By that, the global rotation will be roughly correct and joint angles are slowly trained to avoid overshooting of angular values. This is further ensured by our limits term. After several epochs, when the network already learned to fit the poses roughly, we turn off the regularization and let it refine the angles further. More precisely, the training of *PoseNet* proceeds in three stages. First, we train *PoseNet* for $120k$ iterations with a learning rate of 10^{-5} and weight $\mathcal{L}_{\text{kp}}$ with 0.01. $\mathcal{L}_{\text{limit}}$ has a weight of 1.0 for the first $40k$ iterations. Between $40k$ and $60k$ iterations we re-weight $\mathcal{L}_{\text{limit}}$ with a factor of 0.1. Finally, we set $\mathcal{L}_{\text{limit}}$ to zero for the remaining training steps. Second, we train *PoseNet* for another $120k$ iterations with a learning rate of 10^{-6} and $\mathcal{L}_{\text{kp}}$ is weighted with a factor of 10^{-4}. Third, we train *PoseNet* again $120k$ iterations with a learning rate of 10^{-6} and $\mathcal{L}_{\text{kp}}$ is weighted with a factor of 10^{-5}. We always use a batch size of 90.

Training Strategy for DefNet. We train *DefNet* for $120k$ iterations with a batch size of 50. We used a learning rate of 10^{-5} and weight $\mathcal{L}_{\text{sil}}$, $\mathcal{L}_{\text{kpg}}$, and $\mathcal{L}_{\text{arap}}$ with $1k$, 0.05, and $1.5k$ respectively.

Training Strategy for the Domain Adaptation. To fine-tune the network for in-the-wild monocular test sequences, we train the pre-trained *PoseNet* and *DefNet* for 250 iteration, respectively. To this end, we replace the multi-view losses with a single view loss which can be trivially achieved. For *PoseNet*, we disable $\mathcal{L}_{\text{limit}}$ and weight $\mathcal{L}_{\text{kp}}$ with 10^{-6}. For *DefNet*, we weight $\mathcal{L}_{\text{sil}}$, $\mathcal{L}_{\text{kpg}}$, and $\mathcal{L}_{\text{arap}}$ with $1k$, 0.05, and $1.5k$ respectively. Further, we use a learning rate of 10^{-6} and use the same batch sizes as before. This fine-tuning in total takes around 5 minutes.

6. More Results

Qualitative Results. In Fig. 2, we show our reconstruction without applying the non-rigid deformation regressed by our *DefNet*, referred to as *Pose-only*, and our final result. Our deformed template also looks plausible from a reference view that was not used for tracking. Further, *DefNet* can correctly regress deformations that are along the camera viewing direction of the input camera (see reference view in second column) and surface parts that are even occluded

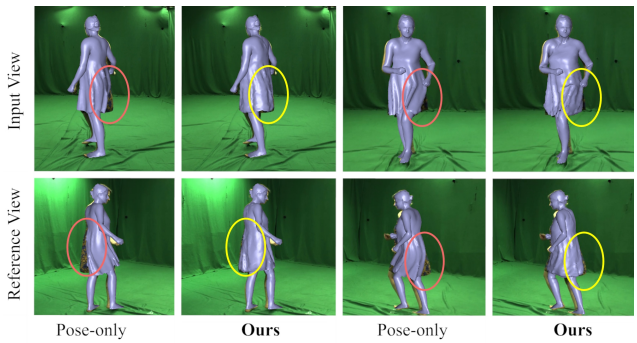

Figure 2. Our result from the input view and a reference view that was not used for tracking. Note that our *DefNet* can even regress deformations along the camera viewing axis of the input camera (second column) and it can correctly deform surface parts that are occluded (fourth column).

Figure 3. Comparisons to related work [5, 4, 10, 13] on our in-the-wild sequences showing *S1*. Our approach can recover the deformations of clothing in contrast to [5] and gives more stable and accurate results in 3D compared to [4]. Moreover, note that in contrast to previous work [10, 13], our method regresses space-time coherent geometry, which follows the structure of the human body.

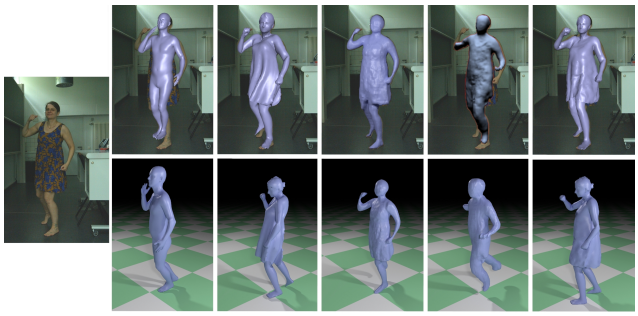

Figure 4. Comparisons to related work [5, 4, 10, 13] on our in-the-wild sequences showing *S4*.

(see reference view in fourth column). This implies that our weak multi-view supervision during training let the network learn the entire 3D surface deformation of the human body. For more results, we refer to the supplemental video.

Qualitative Comparison. In Fig. 3 and Fig. 4, we show more comparisons to related work [5, 4, 10, 13] on our in-the-wild test sequences. Note that our reconstruction not only most accurately overlays to the input but also looks more plausible in 3D compared to previous methods. The implicit functions cannot preserve the skeletal structure, e.g., produce missing arms or noise in the reconstruction, whereas our method recovers space-time coherent geometry without missing body parts. Moreover, in contrast to HMR [5], our methods accounts for clothing deformations. Finally, we are more robust in 3D compared to LiveCap [4] which suffers from the monocular setting. Also note that except LiveCap[4] and our method none of the others can recover results in a consistent global space, as they recover geometry or the human body model only up to scale.

Quantitative Evaluations. In Tab. 1, we report the quantitative comparison for the skeletal pose accuracy on the remaining subjects, *S2* and *S3*. Again note, that we outperform other approaches in all metrics. Further note that even for *S3* we achieve accurate results even though she wears a long dress such that legs are mostly occluded. On *S2*, we found that our results are more accurate than *MVBL* since the classical frame-to-frame optimization can get stuck in local minima, leading to wrong poses.

In Tab. 2, we provide more quantitative comparisons to other related methods in terms of surface reconstruction accuracy. As in the main document, we report the proposed intersection over union metrics (IoU) of subjects *S2* and *S3* on our evaluation sequences. Note that we consistently outperform other approaches for the multi-view evaluation. This is due to our *DefNet* that can account for non-rigid surface deformations that cannot be captured by skinning alone. This is in strong contrast to other work that can only capture the naked body shape and pose [5]. The other reason for the improvement is that our multi-view supervision during training helps to disambiguate the monocular setting whereas other approaches [4] suffer from these inherent ambiguities. Further, we are again close to our multi-view baseline, which uses all views whereas our method is monocular.

Ablation Study. In the following, we provide more visual results to study the impact of *1)* the number of cameras used during training, *2)* the number of training frames, and *3)* the domain adaptation step. *1)* we evaluate the influence of the number of cameras used during training. Most of the improvement can be obtained by going from the monocular setting to two cameras. This is not surprising as the additional camera helps to resolve depth ambiguity. But also adding more cameras consistently improves the result, both, quantitatively as well as qualitatively. An example is shown in Fig. 5 where the pose and deformations are consistently improving the more cameras are added to the training. *2)* in Fig. 6, we visualize the influence of different number of

MPJPE/GLE (in mm) and 3DPCK/AUC (in %) on S2				
Method	**GLE↓**	**PCK↑**	**AUC↑**	**MPJPE↓**
VNect [9]	-	80.50	39.98	66.96
HMR [5]	-	80.02	39.24	71.87
HMMR [6]	-	82.08	41.00	74.69
LiveCap [4]	142.39	79.17	42.59	69.18
Ours	**75.79**	**94.72**	**54.61**	**52.71**
MVBL	64.12	89.91	45.58	57.52

MPJPE/GLE (in mm) and 3DPCK/AUC (in %) on S3				
Method	**GLE↓**	**PCK↑**	**AUC↑**	**MPJPE↓**
VNect [9]	-	78.03	41.95	88.14
HMR [5]	-	83.37	42.37	79.02
HMMR [6]	-	79.93	36.27	91.62
LiveCap [4]	281.27	66.30	31.44	98.76
Ours	**89.54**	**95.09**	**54.00**	**58.77**
MVBL	67.82	96.37	54.99	56.08

Table 1. Skeletal pose accuracy. Note that we are consistently better than other monocular approaches and are even close to the multi-view baseline which takes all camera views as input whereas our method only needs a single view.

AMVIoU, RVIoU and SVIoU (in %) on S2			
Method	**AMVIoU↑**	**RVIoU↑**	**SVIoU↑**
HMR [5]	59.79	59.1	66.78
HMMR [6]	62.64	62.03	68.77
LiveCap [4]	60.52	58.82	77.75
DeepHuman [13]	-	-	**91.57**
Ours	**83.73**	**83.49**	89.26
MVBL	89.62	89.67	92.02

AMVIoU, RVIoU and SVIoU (in %) on S3			
Method	**AMVIoU↑**	**RVIoU↑**	**SVIoU↑**
HMR [5]	59.05	58.73	63.12
HMMR [6]	61.73	61.32	67.14
LiveCap [4]	61.55	60.47	75.6
DeepHuman [13]	-	-	79.66
Ours	**85.75**	**85.55**	**88.27**
MVBL	90.31	90.21	91.53

Table 2. Surface deformation accuracy. Note that we again outperform all other monocular methods and are close to the multi-view baseline. Further note, that for [10, 13] an evaluation of the multi-view IoU is not possible since their output is always in local image space that cannot be brought to global space.

frames available during training. The more pose and clothing deformations were seen during training the better the network can sample the space of possible poses and deformations. Thus, at test time our results consistently improved the more frames we used during training, e.g., see the leg and arm. *3)* finally, in Fig. 7, we visually demonstrate the

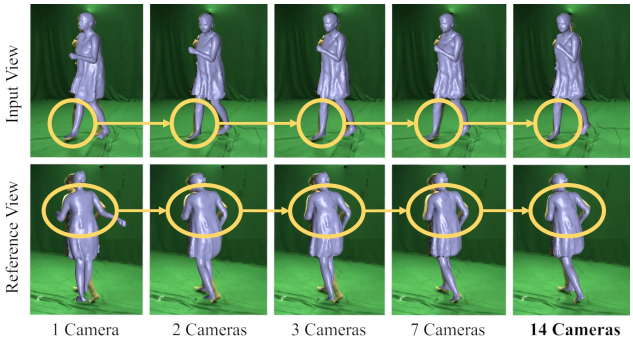

Figure 5. Ablation for number of *cameras* used during training. The most significant improvement happens when adding one additional camera to the monocular setting. But also adding further cameras consistently improves the result as the yellow circles indicate.

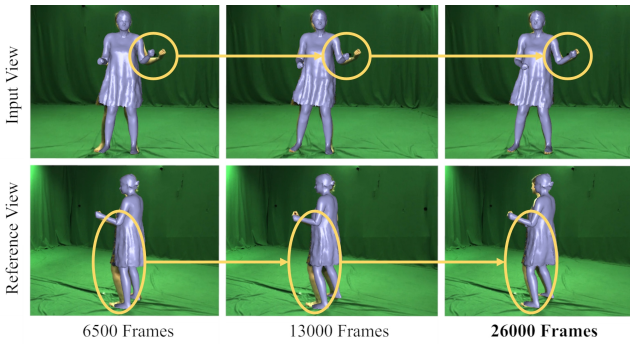

Figure 6. Ablation for number of *frames* used during training. The more frames we used during training the better the result becomes as the network can better sample the possible pose and deformation space.

impact of our domain adaptation step. It becomes obvious that the refinement drastically improves the pose as well as the non-rigid deformations so that the input can be matched at much higher accuracy. Further, we do not require any additional input for the refinement as our losses can be directly adapted to the monocular setting.

7. Further Discussion

Disentanglement of Pose and Deformation. Conceptually, both representations, pose and the non-rigid deformations, are decoupled. Nevertheless, since the predicted poses during training are not perfect, our *DefNet* also deforms the graph to account for wrong poses to a certain degree.

Further Limitations. In our supplemental video, we also tested our method on subjects that were not used for training but which wear the same clothing as the training subject. Although, our method still performs reasonable, the overall accuracy drops as the subjects appearance was never

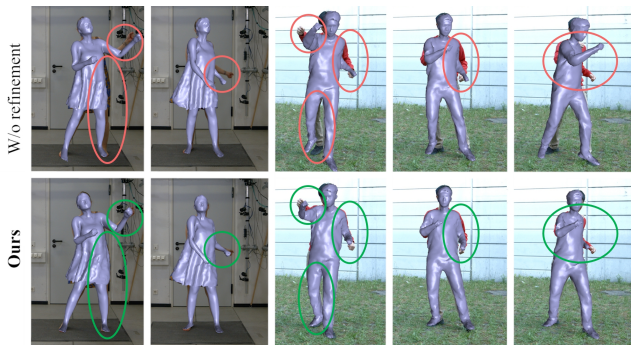

Figure 7. Impact of the in-the-wild domain adaption step. Note that after the network refinement, both, the pose as well as the deformations better match the input.

observed during training. Further, our method can fail for extreme poses, e.g. a hand stand, that were not observed during training.

Acknowledgements. This work was funded by the ERC Consolidator Grant 4DRepLy (770784) and the Deutsche Forschungsgemeinschaft (Project Nr. 409792180, Emmy Noether Programme, project: Real Virtual Humans).

References

[1] M. Abadi, A. Agarwal, P. Barham, E. Brevdo, Z. Chen, C. Citro, G. S. Corrado, A. Davis, J. Dean, M. Devin, S. Ghemawat, I. Goodfellow, A. Harp, G. Irving, M. Isard, Y. Jia, R. Jozefowicz, L. Kaiser, M. Kudlur, J. Levenberg, D. Mané, R. Monga, S. Moore, D. Murray, C. Olah, M. Schuster, J. Shlens, B. Steiner, I. Sutskever, K. Talwar, P. Tucker, V. Vanhoucke, V. Vasudevan, F. Viégas, O. Vinyals, P. Warden, M. Wattenberg, M. Wicke, Y. Yu, and X. Zheng. TensorFlow: Large-scale machine learning on heterogeneous systems, 2015. Software available from tensorflow.org. 2

[2] Z. Cao, G. Hidalgo, T. Simon, S.-E. Wei, and Y. Sheikh. OpenPose: realtime multi-person 2D pose estimation using Part Affinity Fields. In *arXiv preprint arXiv:1812.08008*, 2018. 2

[3] Z. Cao, T. Simon, S.-E. Wei, and Y. Sheikh. Realtime multi-person 2d pose estimation using part affinity fields. In *CVPR*, 2017. 2

[4] M. Habermann, W. Xu, M. Zollhoefer, G. Pons-Moll, and C. Theobalt. Livecap: Real-time human performance capture from monocular video. *ACM Trans. Graph.*, 2019. 1, 3, 4

[5] A. Kanazawa, M. J. Black, D. W. Jacobs, and J. Malik. End-to-end recovery of human shape and pose. *CoRR*, abs/1712.06584, 2017. 3, 4

[6] A. Kanazawa, J. Y. Zhang, P. Felsen, and J. Malik. Learning 3d human dynamics from video. In *Computer Vision and Pattern Regognition (CVPR)*, 2019. 4

[7] D. Kingma and J. Ba. Adam: A method for stochastic optimization. *International Conference on Learning Representations*, 12 2014. 2

[8] M. Loper, N. Mahmood, J. Romero, G. Pons-Moll, and M. J. Black. SMPL: A skinned multi-person linear model. *ACM Trans. Graphics (Proc. SIGGRAPH Asia)*, 34(6):248:1–248:16, Oct. 2015. 1

[9] D. Mehta, S. Sridhar, O. Sotnychenko, H. Rhodin, M. Shafiei, H.-P. Seidel, W. Xu, D. Casas, and C. Theobalt. Vnect: Real-time 3d human pose estimation with a single rgb camera. volume 36, July 2017. 4

[10] S. Saito, Z. Huang, R. Natsume, S. Morishima, A. Kanazawa, and H. Li. Pifu: Pixel-aligned implicit function for high-resolution clothed human digitization. *CoRR*, abs/1905.05172, 2019. 3, 4

[11] T. Simon, H. Joo, I. Matthews, and Y. Sheikh. Hand keypoint detection in single images using multiview bootstrapping. In *CVPR*, 2017. 2

[12] S.-E. Wei, V. Ramakrishna, T. Kanade, and Y. Sheikh. Convolutional Pose Machines. In *Conference on Computer Vision and Pattern Recognition (CVPR)*, 2016. 2

[13] Z. Zheng, T. Yu, Y. Wei, Q. Dai, and Y. Liu. Deephuman: 3d human reconstruction from a single image. *CoRR*, abs/1903.06473, 2019. 3, 4