KBody: Towards general, robust, and aligned monocular whole-body estimation Supplementary Material

Nikolaos Zioulis[1], James F. O'Brien*[1,2]
[1] Klothed Technologies Inc., [2] UC Berkeley
https://klothed.github.io/KBody

A. Complementary Discussion

In the main paper, qualitative and quantitative comparisons are offered to help appreciate the gains of KBody. We select PyMAf-X [9] as a representative of a single-shot estimator that delivers high quality performance on pose capture, evident also by its quantitative and qualitative performance. On the other end, we also select SHAPY [3] as a representative of a single-shot estimator that delivers high quality performance on shape capture, also evident by its quantiative and qualitative performance. Further, as our approach is an iterative optimization based one, we also include the SMPLify-X [7] baseline and its data-driven successor ExPose [4] which is mostly used as a parameter initializer. All aforementioned methods are used with their original – best – hyper-parameters and pre-trained models, receiving the same inputs, and, where necessary, cropped with an extended bounding box calculated from OpenPose's [2] keypoints. Performance is first evaluated on a 3D vertex level, with V2V evaluating pose and shape jointly, PVE-T-SC evaluating shape only, and the body specific measurement metrics presented in SHAPY [3]. We also evaluate pixel alignment performance using the ground-truth rendered bodies when available (EHF & SSP3D), and by extracting segmentation masks using a background subtraction service [1] on the Lab images of HBW (val), which contain minimal clothing and plain backgrounds.

From a strict comparison perspective, PyMAF-X and SHAPY are single-shot regressors, while SMPLify-X and KBody are iterative optimizers. Still, the latter are constrained by the estimations of third-party regressors like OpenPose [2] for the 2D keypoints and MODNet [6] for the silhouette. While the *predict-and-optimize* approach can be considered as a generic scheme and be applied to different regressors like PyMAF-X and SHAPY, there are conflicting arguments made in the literature.

Both EFT [5] and Pose-NDF [8] show that simply post-processing an initial regressor estimate with iterative optimization using third-party regressed constraints does not necessarily lead to improved results. This is the case for ExPose [4] being refined via SMPLify-X [7] in the experiments conducted in Pose-NDF [8] and reported in Table 1 of the main paper, as well as the experiments reported in Table 5 of EFT [5], where better performing initializations lead to inferior results post fine-tuning.

Instead the exemplar fine-tuning [5] that includes no prior terms and only exploits the prior learned by the model is shown to always improve results, albeit less in cases where the initialization is already of high quality. However, EFT post-processing [5] was only applied to simple single-shot pose-dominant regressors, whereas the best performing models in each category, PyMAF-X [9] and SHAPY [3], are modified versions of these regressors whose interplay with EFT post-processing is to be investigated. PyMAF-X [9] already contains a feedback loop, essentially a mini iteration loop using its pyramidal features, to improve pose estimates, and its results interestingly show that this does not improve shape capture as it regresses towards a camera-scaled mean shape. SHAPY [3] on the other hand was (partly) trained with shape annotations, a fact that improved the performance of the shape coefficient regression, but crucially reduced pose performance. Original EFT post-processing would only use the OpenPose keypoints to fine-tune, with unclear consequences for the shape estimate. Still, different schemes that combine the silhouette term, the original SHAPY shape measurements and a discarding of the optimized shape coefficients are potential options, but these are expected to require extensive fine-tuning of the parameters and the process itself, opening up an entirely new problem.

Our findings show that achieving holistic human capture across its different components (shape, pose, image alignment – see Figure 1 of the main paper) requires the integration of predictions and optimization which, considering the rapid important advances in the former, needs more investigative effort on the post-processing/fine-tuning side to align with these developments.

Finally, robustness and applicability to in-the-wild images is an orthogonal goal. These include challenging poses

*Corresponding author: james@getklothed.com

and a variety of body shapes, depicted in images with plain and complex backgrounds, either in full or partially. Section C of this supplemental material includes an extensive set (over 200) of qualitative results comparison across full and partial images and an asymmetric distance field ablation.

B. Runtime Performance

The single-shot regressors (PyMAF-X [9], SHAPY [3]) exhibit significantly better runtime performance as they produce their estimates an order of magnitude faster (around $0.1s$) than the *unoptimized* iterative optimization approaches (SMPLify-X [7], KBody) that need some seconds ($\sim\geq 17s$) to converge. While KBody exploits the ExPose [4] initialization to skip the camera initialization stages of SMPLify-X [7], the use of a differentiable renderer at its later stages incurs extra computational costs, especially for gradient computations. This amounts to an increase of 40% in runtime compared to SMPLify-X [7], with all the measurements taken on the same computational infrastructure, an AWS $g4dn.2xlarge$ instance.

C. Qualitative Results

Figs. 1 to 25 present 112 qualitative result comparisons between the presented KBody method (rightmost - pink) the optimization-based SMPLify-X [7] (leftmost - light green), and the single-shot models PyMAF-X [9] (middle left - purple) and SHAPY [3] (middle right - green), focusing on pose and shape capturing respectively.

In addition, Figs. 29 to 32 present an extra 32 qualitative results that demonstrate benefits of the asymmetric distance field (ADF), compared to a symmetric variant, when considering clothing robustness gains and unnatural shape captures.

Further, Figs. 33 to 52 present 78 qualitative result comparisons between the presented KBody method (rightmost - pink), the optimization-based SMPLify-X [7] (leftmost - light green), and the single-shot models PyMAF-X [9] (middle left - purple) and SHAPY [3] (middle right - green), focusing on pose and shape capturing respectively. These examples focus on partial images with missing head and/or lower body information, and present a challenging scenario for high quality monocular body fitting. Our generative inversion-based completion approach handles them gracefully and helps produce reasonable fits even in the absence of important information. As illustrated by the examples, priors alone cannot handle this properly for optimization-based approaches like SMPLify-X [7], while single-shot estimates [3, 9] exhibit reduced performance given the lack of necessary image context.

References

[1] Remove Image Background 100% Automatically and Free. https://www.remove.bg/. 1

[2] Z. Cao, G. Hidalgo Martinez, T. Simon, S. Wei, and Y. A. Sheikh. OpenPose: Realtime Multi-Person 2D Pose Estimation using Part Affinity Fields. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2019. 1

[3] Vasileios Choutas, Lea Müller, Chun-Hao P Huang, Siyu Tang, Dimitrios Tzionas, and Michael J Black. Accurate 3D Body Shape Regression Using Metric and Semantic Attributes. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 2718–2728, 2022. 1, 2, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16, 17, 18, 19, 20, 21, 22, 23, 24, 25, 26, 27, 28, 29, 30, 31, 36, 37, 38, 39, 40, 41, 42, 43, 44, 45, 46, 47, 48, 49, 50, 51, 52, 53, 54, 55

[4] Vasileios Choutas, Georgios Pavlakos, Timo Bolkart, Dimitrios Tzionas, and Michael J Black. Monocular expressive body regression through body-driven attention. In *European Conference on Computer Vision*, pages 20–40. Springer, 2020. 1, 2

[5] Hanbyul Joo, Natalia Neverova, and Andrea Vedaldi. Exemplar fine-tuning for 3d human model fitting towards in-the-wild 3d human pose estimation. In *2021 International Conference on 3D Vision (3DV)*, pages 42–52. IEEE, 2021. 1

[6] Zhanghan Ke, Jiayu Sun, Kaican Li, Qiong Yan, and Rynson WH Lau. Modnet: Real-time trimap-free portrait matting via objective decomposition. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 36, pages 1140–1147, 2022. 1

[7] Georgios Pavlakos, Vasileios Choutas, Nima Ghorbani, Timo Bolkart, Ahmed AA Osman, Dimitrios Tzionas, and Michael J Black. Expressive body capture: 3d hands, face, and body from a single image. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 10975–10985, 2019. 1, 2, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16, 17, 18, 19, 20, 21, 22, 23, 24, 25, 26, 27, 28, 29, 30, 31, 36, 37, 38, 39, 40, 41, 42, 43, 44, 45, 46, 47, 48, 49, 50, 51, 52, 53, 54, 55

[8] Garvita Tiwari, Dimitrije Antić, Jan Eric Lenssen, Nikolaos Sarafianos, Tony Tung, and Gerard Pons-Moll. Pose-NDF: Modeling Human Pose Manifolds with Neural Distance Fields. In *European Conference on Computer Vision*, pages 572–589. Springer, 2022. 1

[9] Hongwen Zhang, Yating Tian, Yuxiang Zhang, Mengcheng Li, Liang An, Zhenan Sun, and Yebin Liu. PyMAF-X: Towards Well-aligned Full-body Model Regression from Monocular Images. *arXiv preprint arXiv:2207.06400*, 2022. 1, 2, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16, 17, 18, 19, 20, 21, 22, 23, 24, 25, 26, 27, 28, 29, 30, 31, 36, 37, 38, 39, 40, 41, 42, 43, 44, 45, 46, 47, 48, 49, 50, 51, 52, 53, 54, 55

Figure 1. Left-to-right: SMPLify-X [7] (light green), PyMAF-X [9] (purple), SHAPY [3] (green) and KBody (pink).

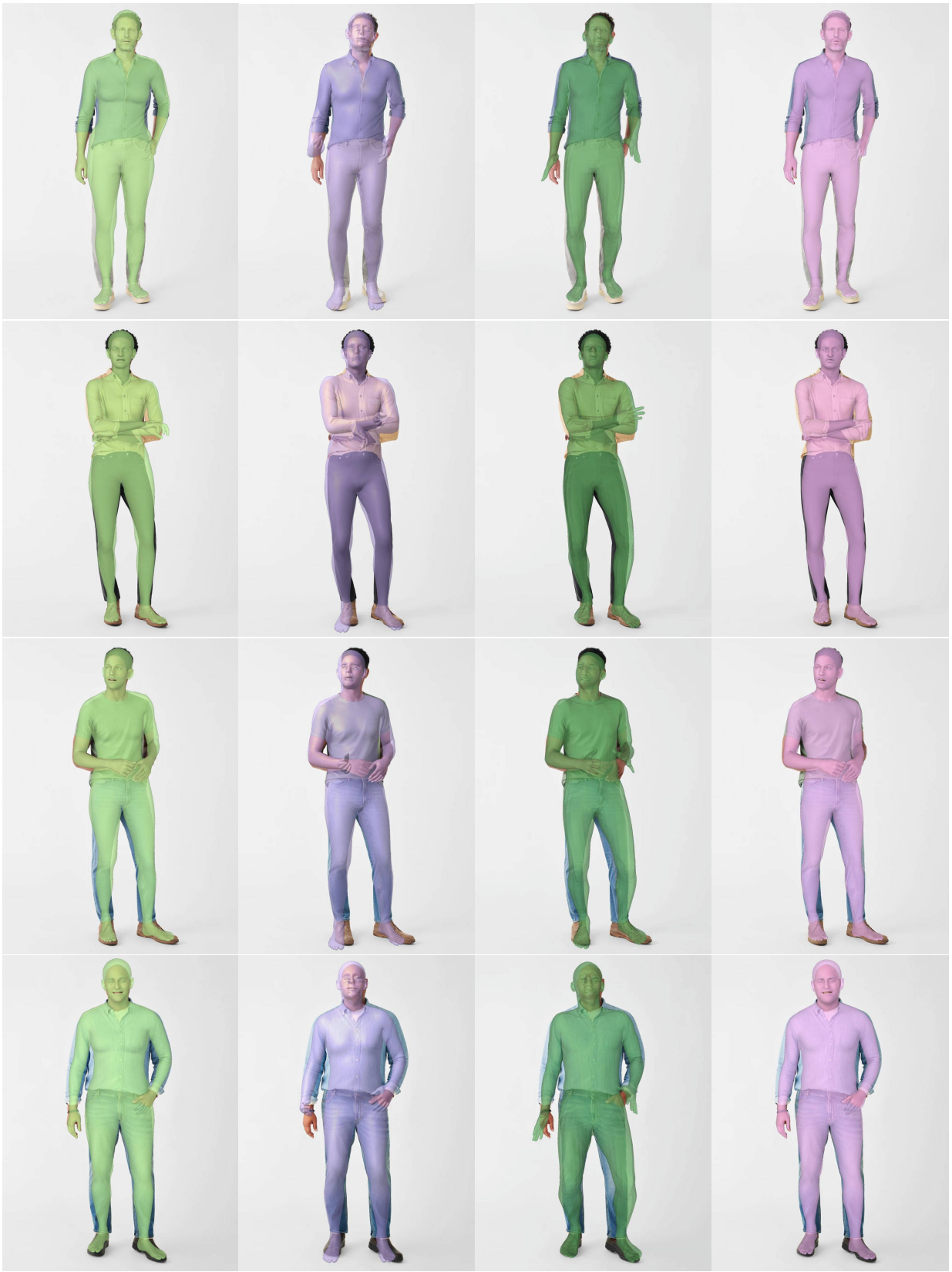
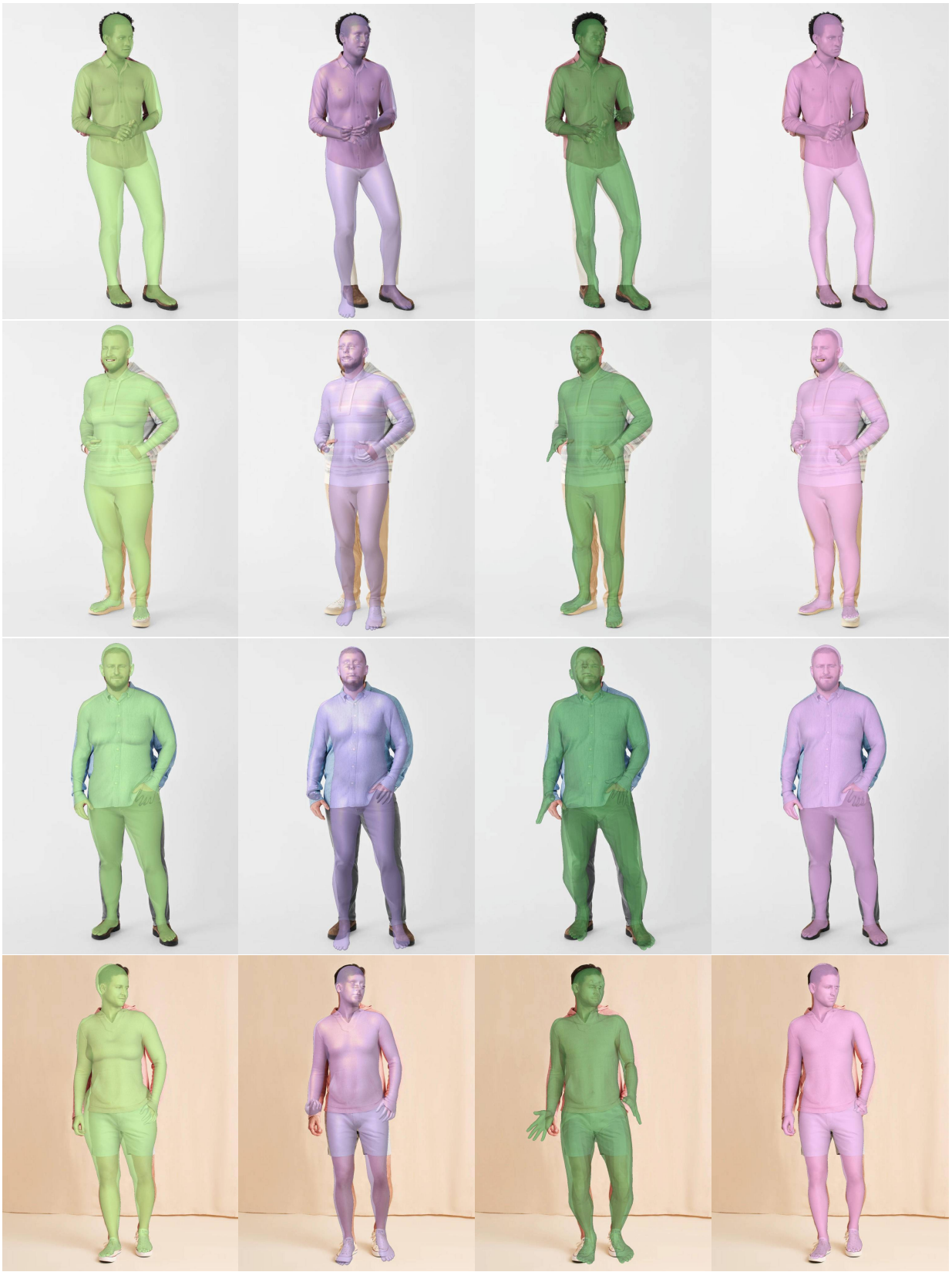


Figure 2. Left-to-right: SMPLify-X [7] (light green), PyMAF-X [9] (purple), SHAPY [3] (green) and KBody (pink).

Figure 3. Left-to-right: SMPLify-X [7] (light green), PyMAF-X [9] (purple), SHAPY [3] (green) and KBody (pink).



SMPLify-X [7] PyMAF-X [9] SHAPY [3] KBody (Ours)

Figure 4. Left-to-right: SMPLify-X [7] (light green), PyMAF-X [9] (purple), SHAPY [3] (green) and KBody (pink).

Figure 5. Left-to-right: SMPLify-X [7] (light green), PyMAF-X [9] (purple), SHAPY [3] (green) and KBody (pink).

Figure 6. Left-to-right: SMPLify-X [7] (light green), PyMAF-X [9] (purple), SHAPY [3] (green) and KBody (pink).

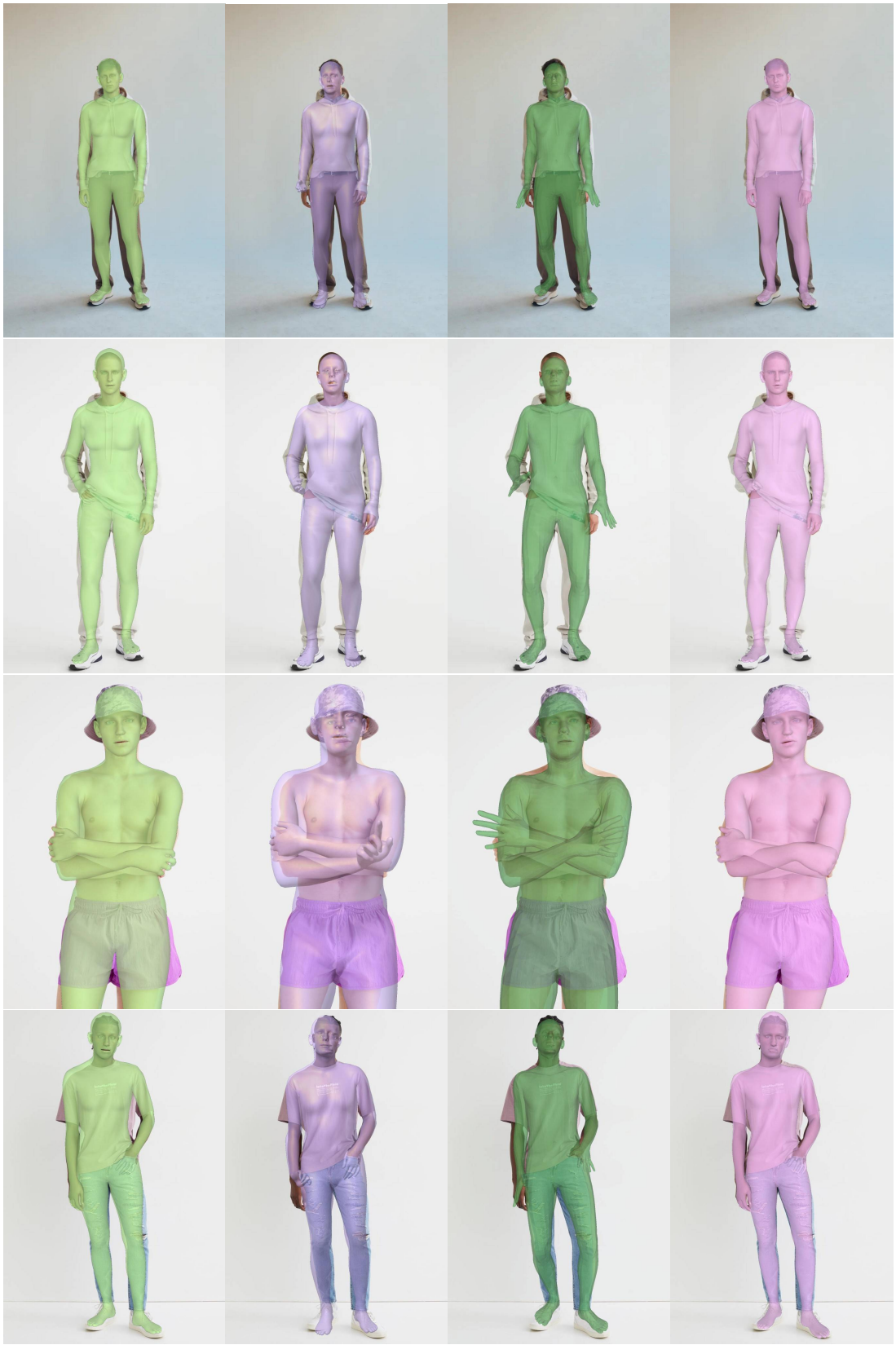


Figure 7. Left-to-right: SMPLify-X [7] (light green), PyMAF-X [9] (purple), SHAPY [3] (green) and KBody (pink).

Figure 8. Left-to-right: SMPLify-X [7] (light green), PyMAF-X [9] (purple), SHAPY [3] (green) and KBody (pink).

Figure 9. Left-to-right: SMPLify-X [7] (light green), PyMAF-X [9] (purple), SHAPY [3] (green) and KBody (pink).

Figure 10. Left-to-right: SMPLify-X [7] (light green), PyMAF-X [9] (purple), SHAPY [3] (green) and KBody (pink).

Figure 11. Left-to-right: SMPLify-X [7] (light green), PyMAF-X [9] (purple), SHAPY [3] (green) and KBody (pink).

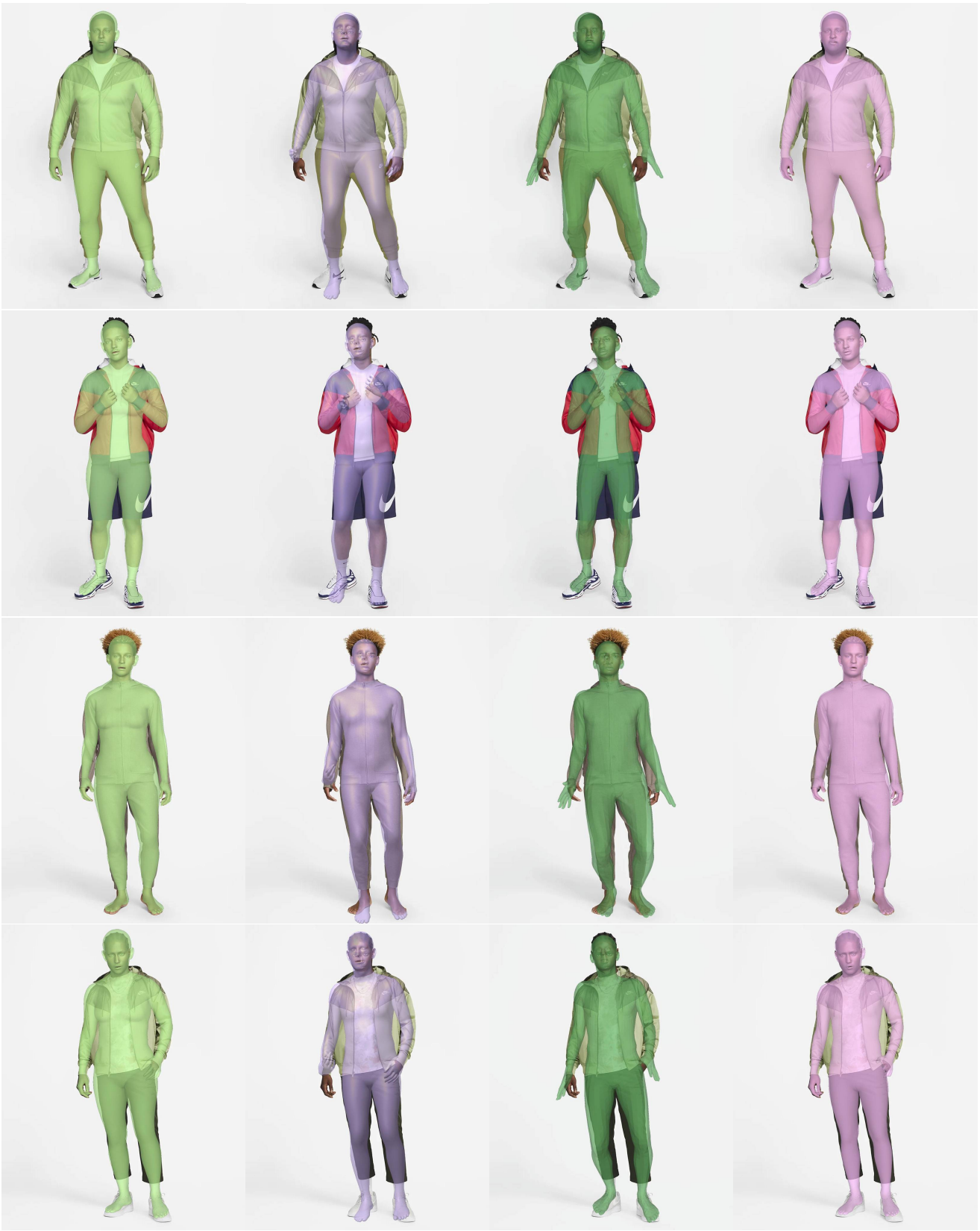


Figure 12. Left-to-right: SMPLify-X [7] (light green), PyMAF-X [9] (purple), SHAPY [3] (green) and KBody (pink).

Figure 13. Left-to-right: SMPLify-X [7] (light green), PyMAF-X [9] (purple), SHAPY [3] (green) and KBody (pink).

Figure 14. Left-to-right: SMPLify-X [7] (light green), PyMAF-X [9] (purple), SHAPY [3] (green) and KBody (pink).

Figure 15. Left-to-right: SMPLify-X [7] (light green), PyMAF-X [9] (purple), SHAPY [3] (green) and KBody (pink).

Figure 16. Left-to-right: SMPLify-X [7] (light green), PyMAF-X [9] (purple), SHAPY [3] (green) and KBody (pink).

SMPLify-X [7] PyMAF-X [9] SHAPY [3] KBody (Ours)

Figure 17. Left-to-right: SMPLify-X [7] (light green), PyMAF-X [9] (purple), SHAPY [3] (green) and KBody (pink).

SMPLify-X [7] PyMAF-X [9] SHAPY [3] KBody (Ours)

Figure 18. Left-to-right: SMPLify-X [7] (light green), PyMAF-X [9] (purple), SHAPY [3] (green) and KBody (pink).

Figure 19. Left-to-right: SMPLify-X [7] (light green), PyMAF-X [9] (purple), SHAPY [3] (green) and KBody (pink).

SMPLify-X [7] PyMAF-X [9] SHAPY [3] KBody (Ours)

Figure 20. Left-to-right: SMPLify-X [7] (light green), PyMAF-X [9] (purple), SHAPY [3] (green) and KBody (pink).

Figure 21. Left-to-right: SMPLify-X [7] (light green), PyMAF-X [9] (purple), SHAPY [3] (green) and KBody (pink).

SMPLify-X [7] PyMAF-X [9] SHAPY [3] KBody (Ours)

Figure 22. Left-to-right: SMPLify-X [7] (light green), PyMAF-X [9] (purple), SHAPY [3] (green) and KBody (pink).

Figure 23. Left-to-right: SMPLify-X [7] (light green), PyMAF-X [9] (purple), SHAPY [3] (green) and KBody (pink).

Figure 24. Left-to-right: SMPLify-X [7] (light green), PyMAF-X [9] (purple), SHAPY [3] (green) and KBody (pink).

Figure 25. Left-to-right: SMPLify-X [7] (light green), PyMAF-X [9] (purple), SHAPY [3] (green) and KBody (pink).

Figure 26. Left-to-right: SMPLify-X [7] (light green), PyMAF-X [9] (purple), SHAPY [3] (green) and KBody (pink).

Figure 27. Left-to-right: SMPLify-X [7] (light green), PyMAF-X [9] (purple), SHAPY [3] (green) and KBody (pink).

Figure 28. Left-to-right: SMPLify-X [7] (light green), PyMAF-X [9] (purple), SHAPY [3] (green) and KBody (pink).

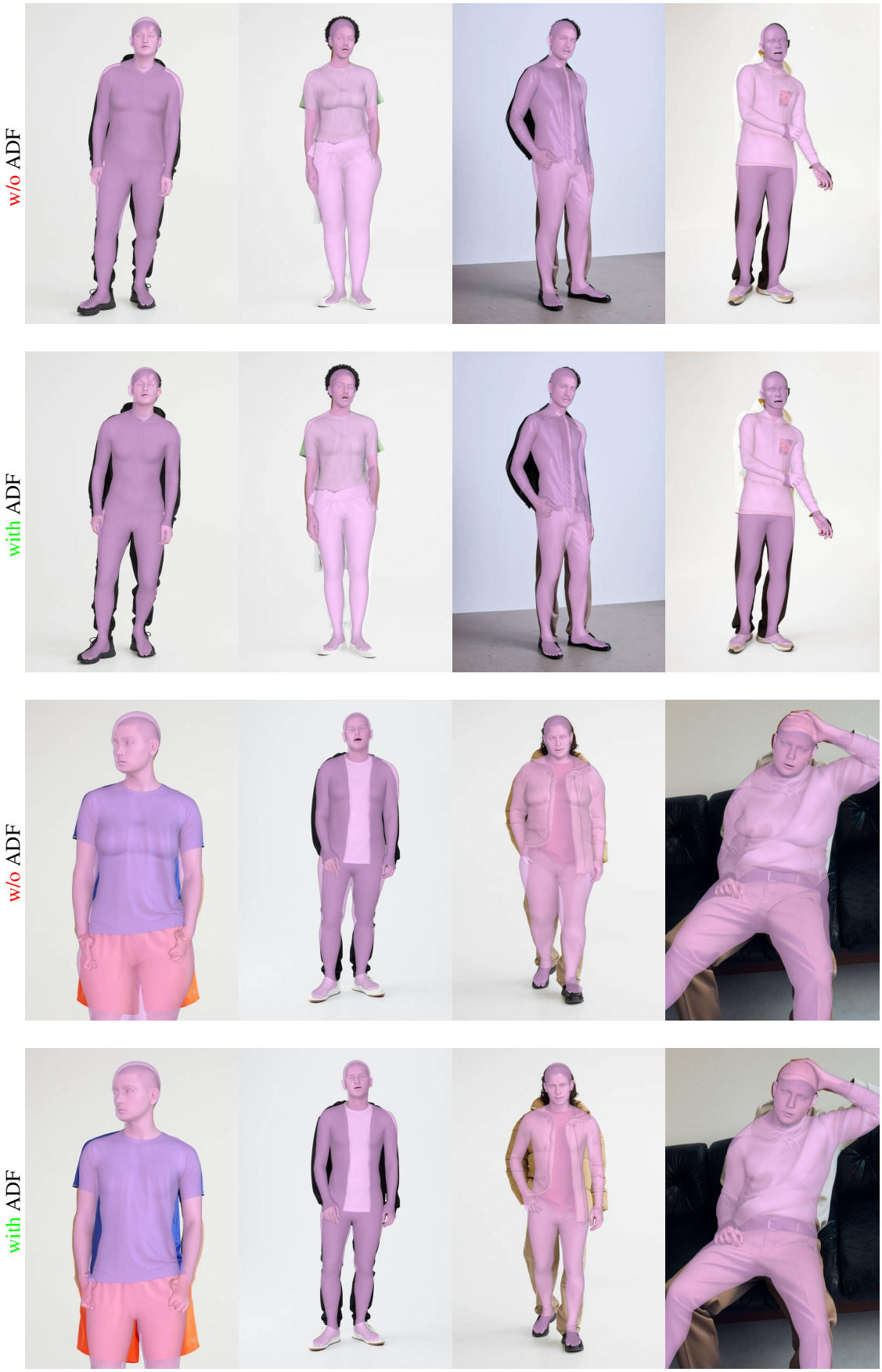


Figure 29. KBody fitting results without ADF on each even row and with ADF on each odd row.

Figure 30. KBody fitting results without ADF on each even row and with ADF on each odd row.

Figure 31. KBody fitting results without ADF on each even row and with ADF on each odd row.

Figure 32. KBody fitting results without ADF on each even row and with ADF on each odd row.

Figure 33. Left-to-right: SMPLify-X [7] (light green), PyMAF-X [9] (purple), SHAPY [3] (green) and KBody (pink).

Figure 34. Left-to-right: SMPLify-X [7] (light green), PyMAF-X [9] (purple), SHAPY [3] (green) and KBody (pink).

Figure 35. Left-to-right: SMPLify-X [7] (light green), PyMAF-X [9] (purple), SHAPY [3] (green) and KBody (pink).

Figure 36. Left-to-right: SMPLify-X [7] (light green), PyMAF-X [9] (purple), SHAPY [3] (green) and KBody (pink).

Figure 37. Left-to-right: SMPLify-X [7] (light green), PyMAF-X [9] (purple), SHAPY [3] (green) and KBody (pink).

Figure 38. Left-to-right: SMPLify-X [7] (light green), PyMAF-X [9] (purple), SHAPY [3] (green) and KBody (pink).

Figure 39. Left-to-right: SMPLify-X [7] (light green), PyMAF-X [9] (purple), SHAPY [3] (green) and KBody (pink).

Figure 40. Left-to-right: SMPLify-X [7] (light green), PyMAF-X [9] (purple), SHAPY [3] (green) and KBody (pink).

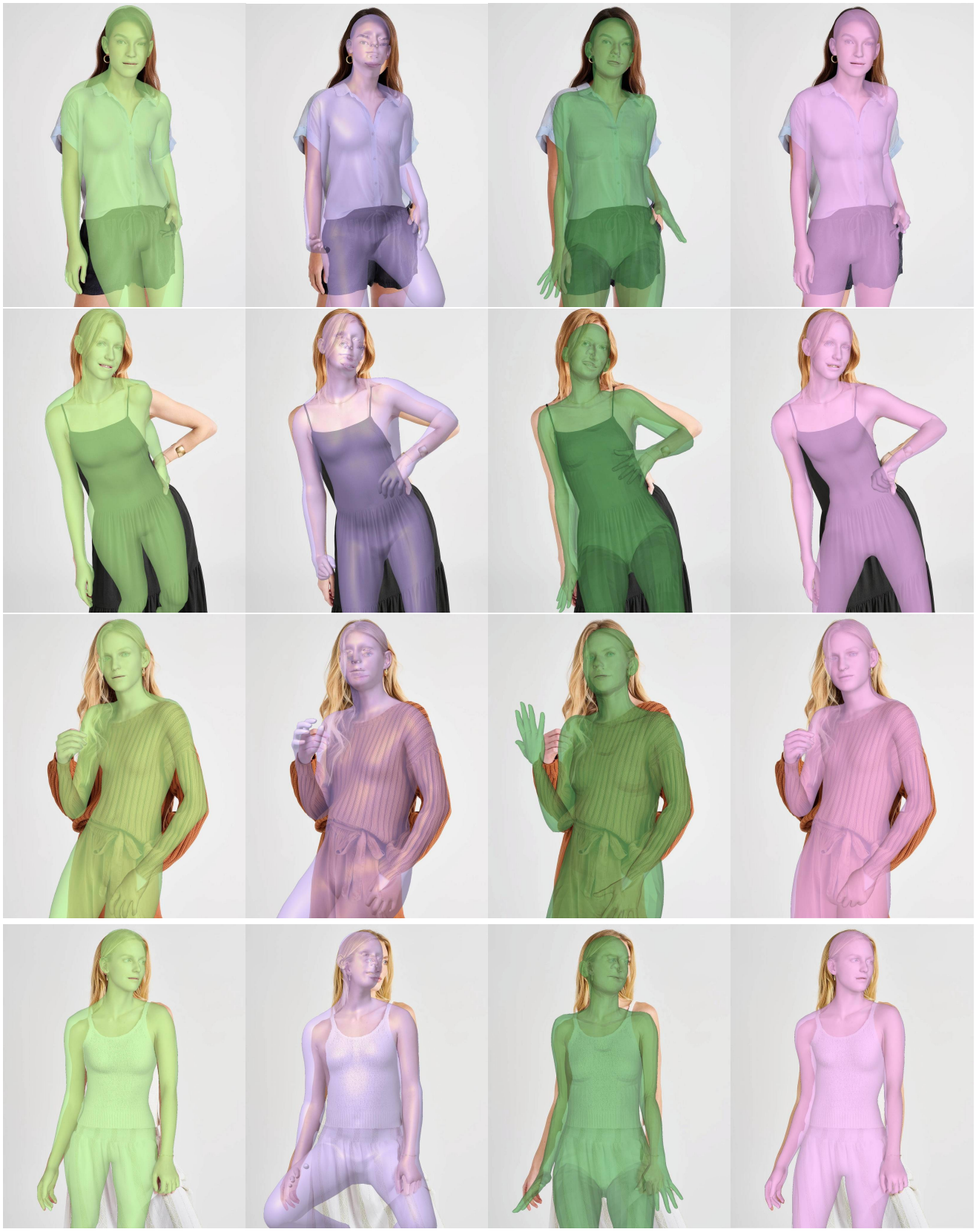


Figure 41. Left-to-right: SMPLify-X [7] (light green), PyMAF-X [9] (purple), SHAPY [3] (green) and KBody (pink).

Figure 42. Left-to-right: SMPLify-X [7] (light green), PyMAF-X [9] (purple), SHAPY [3] (green) and KBody (pink).

Figure 43. Left-to-right: SMPLify-X [7] (light green), PyMAF-X [9] (purple), SHAPY [3] (green) and KBody (pink).

Figure 44. Left-to-right: SMPLify-X [7] (light green), PyMAF-X [9] (purple), SHAPY [3] (green) and KBody (pink).

Figure 45. Left-to-right: SMPLify-X [7] (light green), PyMAF-X [9] (purple), SHAPY [3] (green) and KBody (pink).

Figure 46. Left-to-right: SMPLify-X [7] (light green), PyMAF-X [9] (purple), SHAPY [3] (green) and KBody (pink).

Figure 47. Left-to-right: SMPLify-X [7] (light green), PyMAF-X [9] (purple), SHAPY [3] (green) and KBody (pink).

Figure 48. Left-to-right: SMPLify-X [7] (light green), PyMAF-X [9] (purple), SHAPY [3] (green) and KBody (pink).

Figure 49. Left-to-right: SMPLify-X [7] (light green), PyMAF-X [9] (purple), SHAPY [3] (green) and KBody (pink).

SMPLify-X [7] PyMAF-X [9] SHAPY [3] KBody (Ours)

Figure 50. Left-to-right: SMPLify-X [7] (light green), PyMAF-X [9] (purple), SHAPY [3] (green) and KBody (pink).

Figure 51. Left-to-right: SMPLify-X [7] (light green), PyMAF-X [9] (purple), SHAPY [3] (green) and KBody (pink).

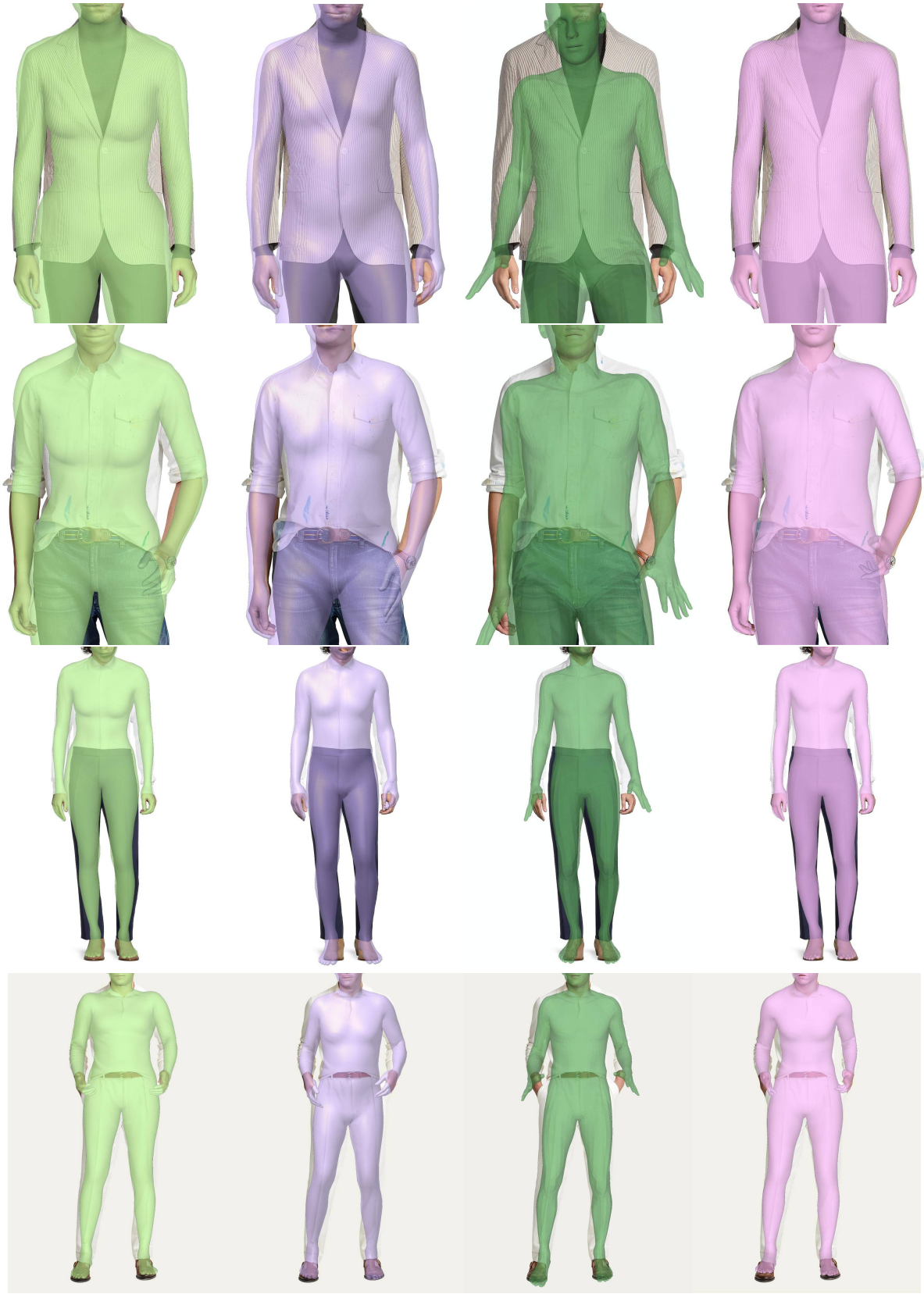


Figure 52. Left-to-right: SMPLify-X [7] (light green), PyMAF-X [9] (purple), SHAPY [3] (green) and KBody (pink).