Supplementary:Human Body Model Fitting by Learned Gradient Descent

Jie Song[1⋆], Xu Chen[1,2⋆], and Otmar Hilliges[1]

[1] ETH Zürich
[2] Max Planck ETH Center for Learning Systems

These are the supplementary materials, accompanying our main paper. Here we provide further implementation details and discuss additional quantitative and qualitative results.

1 Implementation Details

In this section we introduce the implementation details of our method to ease reproduceability.The code for training and testing will be released.

Network architecture.
We use a standard multi-layer perceptron as our Gradient Updating Network. The network consists of 4 blocks, each of which is composed by one fully connected layer, one batch normalization layer, one parametric ReLU layer and one dropout layer. Each fully connected layer has 1024 neurons, and the dropout probability is set to 0.2.

Data sampling for training.
At each iteration of training, we uniformly randomly sample a pose and a shape parameter from the motion database, processed via MOSH. The pose parameters already contain global orientation of the body. In order to simulate different camera view points, we rotate the body with yaw angles that we uniformly sample from $[-180°, 180°]$ and roll and pitch angles uniformly sampled from $[-20°, 20°]$. To simulate missing or erroneous detections from the 2D pose detector, we randomly drop simulated joints with a probability of 20%.

Training routine.
We train the network using Adam optimizer with learning rate 0f 0.001 for 20 epochs. The batch size is set to 1024. The whole training procedure takes roughly 8 hours on a Nvidia GTX 1080Ti GPU.

⋆ Equal contribution.

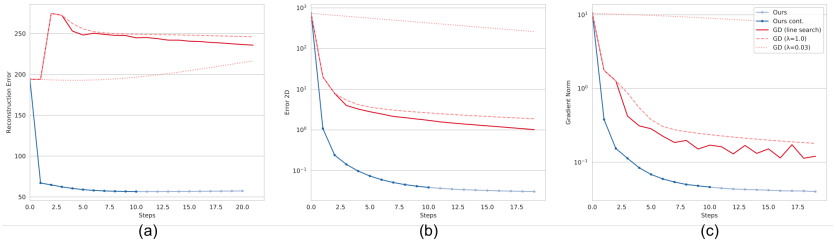

Fig. 1: **Gradient comparison with gradient descent, initialized with zero pose.** (a): reconstruction error along with iterations; (b): 2D reprojection error along with iterations; (c): Gradient norm along with iterations.

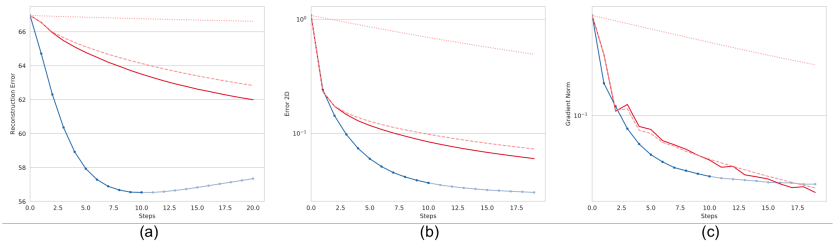

Fig. 2: **Gradient comparison with gradient descent, initialized with the pose running our algorithm for one iteration.** (a): reconstruction error along with iterations; (b): 2D reprojection error along with iterations; (c): Gradient norm along with iterations.

2 Convergence Analysis

In this experiment we shed further light on the convergence properties of the proposed algorithm and compare it to standard gradient descent.

To assess the quality of the solutions found by these two iterative algorithms we optimize the same objective (the 2D re-projection error) without any regularization or prior terms and then compare convergence rate and final solution quality. As indication for convergence we analyse the gradient norm. In order to assess the solution quality we analyse both the objective value and the true reconstruction error (in 3D).

To this end we conduct two experiments. First, we initialize both algorithms with the zero pose (Fig. 1) and let both algorithms run until convergence.

Fig. 1, c) plots the gradient-norm after each iteration, comparing ours with standard gradient descent, both with and without employing line-search to find the optimal step-size. Clearly, both algorithms make progress in terms of convergence, where ours consistently achieves lower gradient norms, and is faster. Fig. 1, (a+b) furthermore illustrate that both algorithms find a solution with respect to the optimization criterion, ours finds significantly better solutions with respect to the true objective. This suggests that our method indeed learns a

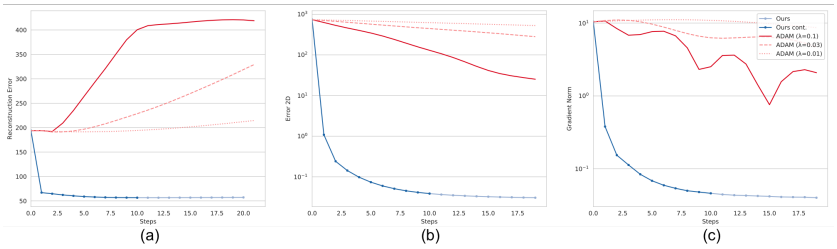

Fig. 3: **Gradient comparison with Adam, initialized with zero pose.** (a): reconstruction error along with iterations; (b): 2D reprojection error along with iterations; (c): Gradient norm along with iterations.

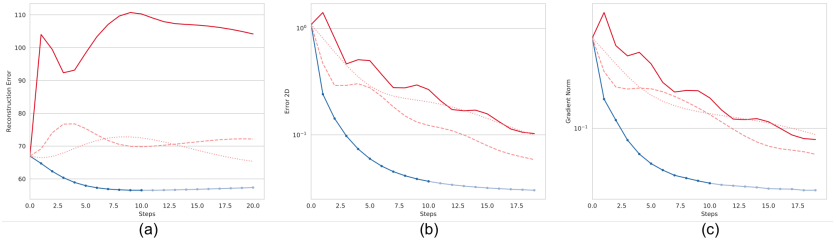

Fig. 4: **Gradient comparison with Adam, initialized with the pose running our algorithm for one iteration.** (a): reconstruction error along with iterations; (b): 2D reprojection error along with iterations; (c): Gradient norm along with iterations.

manifold of valid poses and learns additional regularizing terms that are necessary in other optimization based methods in order to converge to a good solution.

In our second experiment, illustrated in Fig. 2, we repeat the same experiment but initialize both algorithms with a pose configuration that is already close to the true solution. This set of pose parameters is attained by running our algorithm for one iteration and then using the intermediate output as starting condition. As can be seen in Fig. 2, our method still converges faster to a better solution even if gradient descent is initialized close to a good solution.

Please take note that during training we never iterate beyond $N = 10$ but at inference we iterate beyond this training window (light blue: Ours beyond the training window). Please also note that we plot the true gradient norm $\left\| \frac{\partial \mathcal{L}(\Theta_n)}{\partial \Theta_n} \right\|$ rather than that of the parameter update $\|\Delta\Theta\|$ in both Figures to allow for a direct comparison with gradient descent.

Same experiments conducted with Adam can be found in Fig. 3 and Fig. 4. We can see similar observations.

3 More quantitative results

We also compare other methods on MPI-INF-3DHP dataset [3]. It is a dataset captured with a multi-view setup mostly in indoor environments. No markers are used for the capture, so 3D pose data tend to be less accurate compared to other datasets. We use the provided training set (subjects S1 to S8) for training and we report results on the test set. The results are in Tab. 1. Ours also achieves better performance than other state-of-the-art regression methods [1,2] if compared fairly to ours. That is we directly compare the setting in which these methods do not use image-to-3D paired information since our method does not have access to this additional data.

Table 1: **Evaluation on MPI-INF.** [3]. For PCK and AUC, higher is better, while for MPJPE, lower is better. Ours achieves better performance than other state-of-the-art regression methods [1,2] if compared fairly to ours.

Method	PCK	AUC	MPJPE
VNect [4]	83.9	47.3	98.0
HMR (with additional 3D data) [1]	86.3	47.8	89.8
SPIN (with additional 3D data) [2]	92.5	55.6	67.5
HMR [1]	77.1	40.7	113.2
SPIN [2]	87.0	48.5	80.4
Ours	**92.0**	**52.9**	**73.8**

4 More qualitative results

In Fig. 5, we demonstrate more qualitative results and we show some failure cases in Fig. 6.

References

1. Kanazawa, A., Black, M.J., Jacobs, D.W., Malik, J.: End-to-end recovery of human shape and pose. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. pp. 7122–7131 (2018) 4
2. Kolotouros, N., Pavlakos, G., Black, M.J., Daniilidis, K.: Learning to reconstruct 3d human pose and shape via model-fitting in the loop. In: Proceedings of the IEEE International Conference on Computer Vision. pp. 2252–2261 (2019) 4
3. Mehta, D., Rhodin, H., Casas, D., Fua, P., Sotnychenko, O., Xu, W., Theobalt, C.: Monocular 3d human pose estimation in the wild using improved cnn supervision. In: 2017 International Conference on 3D Vision (3DV). pp. 506–516. IEEE (2017) 4
4. Mehta, D., Sridhar, S., Sotnychenko, O., Rhodin, H., Shafiei, M., Seidel, H.P., Xu, W., Casas, D., Theobalt, C.: Vnect: Real-time 3d human pose estimation with a single rgb camera. ACM Transactions on Graphics (TOG) **36**(4), 44 (2017) 4

Fig. 5: **Qualitative results from various datasets.**

Fig. 6: **Some failure cases.**