Multiple View Geometry Transformers for 3D Human Pose Estimation

Supplementary Material

We present more implementation details and experiment results in this supplementary.

A1. Network Architectures

There are several MLP networks used in our model. We present the detailed structures here. The feature dimension of the appearance term is 256.

Network 1: $g_\theta(.)$ This network is used for estimating the 2D pose residuals in the appearance module. It has three layers with their hidden dimensions being set to 256. We use ReLU as the non-linear activation layer.

Network 2: $f_\alpha(.)$ This network is used for feature fusion as shown in Equation 4. It is an MLP with one linear layer that maps the appearance term to a vector with the same dimension.

Network 3: $f_\gamma(.)$ This network is also used for feature fusion as shown in Equation 4. It has two layers with their hidden dimensions being set to 1024. We use ReLU as the non-linear activation layer. There is also a residual branch. We also use layer norm.

Network 4: $f_\beta(.)$ This network is for query classifier. It is a linear layer that maps a vector with a dimension of L to two scores followed by a sigmoid function with the output representing the positive and negative probability, respectively.

A2. Visualizations

Qualitative results on Shelf and Campus. We have visualized some of the estimated results on the images from the CMU Panoptic dataset in Figure 5. We also visualize the results on the images from the Shelf and Campus datasets, in Figure A1. Our method can accurately estimate the poses for people in different postures even when severe occlusion occurs in some views. While the results are nearly perfect on the Shelf dataset, we notice that in the Campus dataset, our method gets inaccurate estimates for the person in a pink shirt (lower arm, yellow skeleton). This usually occurs when the number of cameras is small and the body joints are occluded in most views. We discuss the limitation in Sec A7.

Results from Each Decoder Layer We visualize the 2D and 3D pose estimations of 3 persons on the CMU Panoptic dataset from each decoder layer in Figure A2. The geometry queries are refined from coarse initializations to accurate poses through four transformer decoder layers.

A3. Camera Arrangements

We use different camera arrangements following [1]. We show the details of the camera arrangements of CMU0-CMU4 used in the generalization experiments in Table A1, and visualize them in Figure A3. Among them, CMU2 and CMU3 are more challenging because they either have a small number of cameras, or the cameras miss the other side of views.

A4. More Ablation Studies

Number of Cameras To further investigate the influence of the number of cameras, we train the model on CMU0(7) with two extra cameras on CMU0 as can be seen in Figure A3 (b). Then we gradually decrease the number of cameras and evaluate their performance in Table A2. Our method is consistently better than VoxelPose, which validates its strong generalization performance. More importantly, we can see that AP100 barely changes when we decrease the number of cameras from 7 to 4. In extreme cases, when we only have two cameras available, AP100 of our method is still reasonable considering the severe occlusions in the dataset. It may be helpful to note that this experiment is different from Table 1 in the main paper where the model is trained on five cameras CMU0(5).

Loss weights of decoder layers As shown in Table A3, "All layers" means adding loss to all four layers. "w/o decay" means using the same loss weight for each layer without weight decay. "Exp decay" means the loss weights decaying exponentially as $1, 0.5, 0.25, 0.125$ for the last layer to the first layer, respectively. The idea is to tolerate the errors in the earlier layers. "Linear decay" means the loss weights decaying linearly as $1, 0.75, 0.5, 0.25$ for the last layer to the first layer, respectively. We get the best performance when adding the same weight loss to all the layers. The performance decreases dramatically when only adding loss to the final layer, denoted by "Final layer", which shows it is important to add a supervision signal for each layer for good convergence.

Denoise Training In the main experiments, we use uniform sampling to initialize geometry queries during both

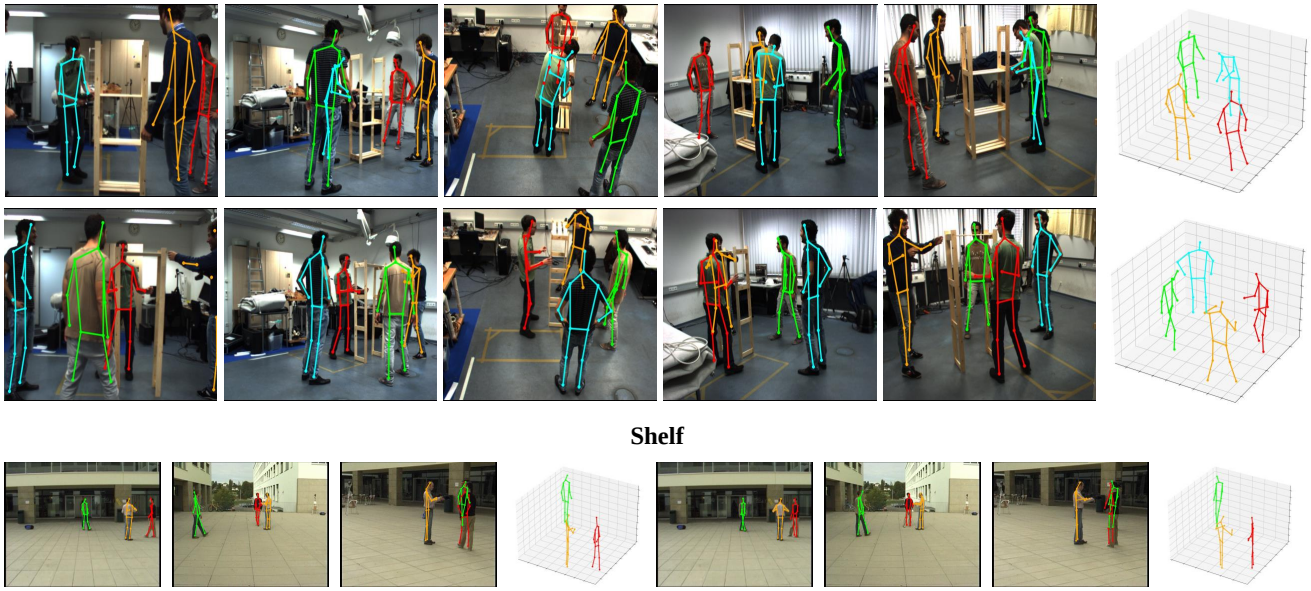


Figure A1. Sample results on Shelf and Campus datasets.

Camera Arrangements	Camera IDs	Camera Num
CMU1	1, 2, 3, 4, 6, 7, 10	7
CMU2	12, 16, 18, 19, 22, 23, 30	7
CMU3	10, 12, 16, 18	4
CMU4	6, 7, 10, 12, 16, 18, 19, 22, 23, 30	10
CMU0	3, 6, 12, 13, 23	5
CMU0 w/ 2 extra cameras	3, 6, 12, 13, 23, 10, 16	7
CMU0(K)	First K cameras in CMU0 w/ 2	K

Table A1. The details of the camera arrangements.

Cam Num	VoxelPose [10]			Ours		
	AP25	AP100	MPJPE	AP25	AP100	MPJPE
7	89.8	98.3	16.1	**95.3**	99.5	14.6
6	87.1	98.0	17.1	**94.5**	99.4	15.2
5	68.7	84.8	18.9	**91.1**	99.2	17.0
4	35.8	80.8	25.1	**75.6**	98.9	21.0
3	1.8	40.6	67.4	**35.1**	94.3	36.3
2	0.0	2.1	164.7	**1.9**	35.7	97.0

Table A2. Ablation study on the number of cameras. Both models are trained on CMU0(7).

training and inference. Here we explore another method for initialization during training. In particular, we add noises to the ground truth poses with a sigma of σ and use the noised poses as the initialized queries. This method is denoted by "GT Noise σ". During inference, we use the output of VoxelPose to initialize the queries. As shown in Table A4, our system can boost up the AP25 of VoxelPose from 85.3% to 92.7% under "GT Noise 20", which is also slightly higher than the sampling-based initialization. It proves that our system has a higher accuracy when given an accurate ini-

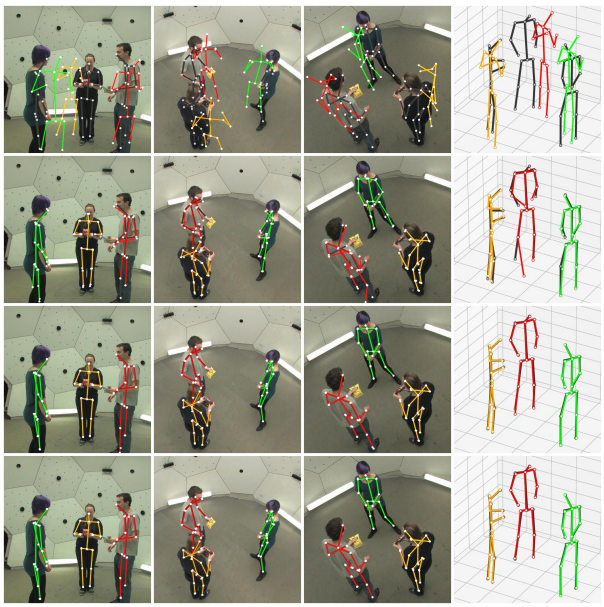


Figure A2. Progressive refinement of the 2D and 3D poses over four decoder layers. The outputs from Layer 1 to Layer 4 are shown from top to bottom. The images of Camera 1 to Camera 4 and the 3D pose are shown from left to right. Black and colorful skeletons denote the GT and estimated poses, respectively.

Configurations	AP25	AP100	MPJPE
All layers w/o decay	**92.3**	**99.3**	**16.0**
All layers w/ Exp decay	91.2	99.0	17.2
All layers w/ Linear decay	90.4	99.0	17.0
Only final layer	0.2	57.0	96.1

Table A3. Adding weight decay among decoder layers.

Model	Training Init	Infer Init	AP25
VoxelPose	/	/	85.3
Ours	GT Noise 200	VoxelPose	90.3
Ours	GT Noise 100	VoxelPose	91.4
Ours	GT Noise 50	VoxelPose	92.4
Ours	GT Noise 20	VoxelPose	**92.7**
Ours	Sampling	Sampling	92.3

Table A4. Denoise training experiments. During training, we initialize with noisy ground truth poses; during inference, we initialize with VoxelPose. We can further improve the results of VoxelPose.

tialization, and can act as a refiner to further improve the performance of other human pose estimation methods.

Parameters Sharing Since all decoder layers share the same goal of refining the 2D and 3D poses, we evaluate whether we can use the same parameters for them which can reduce the number of parameters. As shown in Table A5, if we share the parameters, the performance decreases significantly in terms of AP25. But for AP100, it barely changes. The results suggest that using independent parameters is helpful to improve estimation precision. The front and last layers tend to learn different focuses for coarse and fine updates.

Decoder Layer	AP25	AP100	MPJPE
Independent	**92.3**	**99.3**	**16.0**
Sharing	85.5	98.1	19.0

Table A5. Ablation on parameter sharing for decoder layers.

FFN	AP25	AP100	MPJPE
w/	**92.3**	**99.3**	**16.0**
w/o	88.6	99.0	17.6

Table A6. Ablation study of the MLP $f_\gamma(.)$ in Feature Fusion.

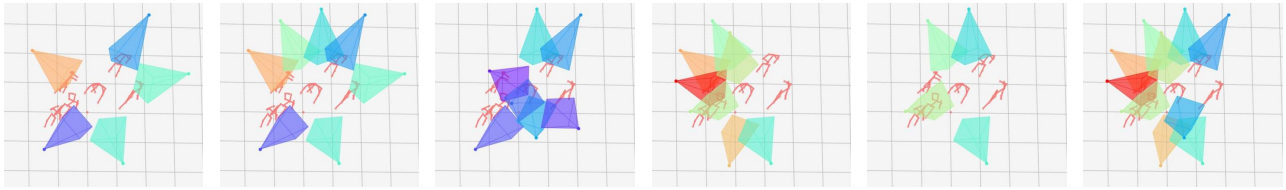
The MLP in Feature Fusion As shown in Table A6, when adding the MLP f_γ in the *feature fusion* process, the accuracy can be further improved.

A5. Camera Parameters

Camera parameters including the extrinsic poses and intrinsic matrix are needed for triangulation. Camera parameters are practical to get in real applications. For scenarios like surveillance where cameras are fixed, a one-time camera calibration [9] can provide the poses and intrinsic. There are also online calibration methods [6, 7] to estimate camera poses dynamically. Since the camera calibration is well studied in the literature and is out of the scope of this paper, we assume the camera pose is known and use the provided parameters in the datasets following the common practices in the literature [10, 11]. All the baselines assume the camera poses are provided, so we have a fair comparison to them and can highlight the performance improvement brought by our model generalization ability.

A6. Computation Analysis

We show parameter counts, flops (evaluated by MACs), and running times in Table A7. Ours has the fewest parameters among the 3 methods and comparable MACs with MvP. We use a single V100 GPU for inference. Ours needs 0.21s for each inference, which is faster than VoxelPose, which needs 0.29s. Ours lies inside a 19% range with MvP which needs 0.17s. A more efficient query sampling strategy remains in future work.



(a) CMU0 (b) CMU0 with 2 extra cameras (c) CMU1 (d) CMU2 (e) CMU3 (f) CMU4

Figure A3. Visualization of the camera arrangements of sequences CMU0 to CMU4.

	Param (M)	MACs (G)	Time (ms)
MvP	42.6	**567.0**	**169.5**
VoxelPose	40.6	972.2	291.3
Ours	**37.6**	639.7	205.1

Table A7. Computation analysis.

A7. Limitation and Future Work

As can be seen from the visualizations in Figure A1 (the lower arm of the yellow skeleton on Campus dataset is inaccurate), our method suffers when the number of camera views is extremely small and the body joints are severely occluded in most views. This is because our system relies on triangulation which requires accurate 2D positions in at least two views to recover the 3D position accurately. Note that this also poses challenges for other methods and our method actually performs better than them. Some methods such as VoxelPose are slightly more robust to occlusion because they use 3D convolutions to mix the features of all joints, which allows to make coarse predictions for the occluded joints based on the visible ones. Inspired by that, one possible way to enhance our method is to use robust structural triangulation [2] instead of the current keypoint-wise triangulation [5] to explore the dependency among all joints to help estimate the occluded joints. Besides, we can take advantage of the constraints from the scenes and human interactions [4] to further improve the accuracy. Finally, it will be interesting to extend the Transformer architecture into a video-based system [3, 8, 12, 13] which further fuses temporal information for robust tracking.

References

[1] Kristijan Bartol, David Bojanić, Tomislav Petković, and Tomislav Pribanić. Generalizable human pose triangulation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 11028–11037, 2022. 1

[2] Zhuo Chen, Xu Zhao, and Xiaoyue Wan. Structural triangulation: A closed-form solution to constrained 3d human pose estimation. In *Computer Vision–ECCV 2022: 17th European Conference, Tel Aviv, Israel, October 23–27, 2022, Proceedings, Part V*, pages 695–711. Springer, 2022. 4

[3] Junting Dong, Qi Fang, Wen Jiang, Yurou Yang, Qixing Huang, Hujun Bao, and Xiaowei Zhou. Fast and robust multi-person 3d pose estimation and tracking from multiple views. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 44(10):6981–6992, 2021. 4

[4] Mohamed Hassan, Partha Ghosh, Joachim Tesch, Dimitrios Tzionas, and Michael J Black. Populating 3d scenes by learning human-scene interaction. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 14708–14718, 2021. 4

[5] Karim Iskakov, Egor Burkov, Victor Lempitsky, and Yury Malkov. Learnable triangulation of human pose. In *Proceedings of the IEEE/CVF international conference on computer vision*, pages 7718–7727, 2019. 4

[6] Sang-Eun Lee, Keisuke Shibata, Soma Nonaka, Shohei Nobuhara, and Ko Nishino. Extrinsic camera calibration from a moving person. *IEEE Robotics and Automation Letters*, 7(4):10344–10351, 2022. 3

[7] Jens Puwein, Luca Ballan, Remo Ziegler, and Marc Pollefeys. Joint camera pose estimation and 3d human pose estimation in a multi-camera setup. In *Computer Vision–ACCV 2014: 12th Asian Conference on Computer Vision, Singapore, Singapore, November 1-5, 2014, Revised Selected Papers, Part II 12*, pages 473–487. Springer, 2015. 3

[8] N Dinesh Reddy, Laurent Guigues, Leonid Pishchulin, Jayan Eledath, and Srinivasa G Narasimhan. Tessetrack: End-to-end learnable multi-person articulated 3d pose tracking. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 15190–15200, 2021. 4

[9] Joaquim Salvi, Xavier Armangué, and Joan Batlle. A comparative review of camera calibrating methods with accuracy evaluation. *Pattern recognition*, 35(7):1617–1635, 2002. 3

[10] Hanyue Tu, Chunyu Wang, and Wenjun Zeng. Voxelpose: Towards multi-camera 3d human pose estimation in wild environment. In *Computer Vision–ECCV 2020: 16th European Conference, Glasgow, UK, August 23–28, 2020, Proceedings, Part I 16*, pages 197–212. Springer, 2020. 2, 3

[11] Jianfeng Zhang, Yujun Cai, Shuicheng Yan, Jiashi Feng, et al. Direct multi-view multi-person 3d pose estimation. *Advances in Neural Information Processing Systems*, 34: 13153–13164, 2021. 3

[12] Yuxiang Zhang, Liang An, Tao Yu, Xiu Li, Kun Li, and Yebin Liu. 4d association graph for realtime multi-person

motion capture using multiple video cameras. In *CVPR*, pages 1324–1333, 2020. 4

[13] Yifu Zhang, Chunyu Wang, Xinggang Wang, Wenyu Liu, and Wenjun Zeng. Voxeltrack: Multi-person 3d human pose estimation and tracking in the wild. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 45(2):2613–2626, 2022. 4