J, C	✓	✓	✓	✓	✗	✗	✗	✗	✓	✓	✓	✗	✓	✗	✗
H	✓	✓	✓	✗	✓	✗	✓	✓	✗	✗	✓	✗	✗	✓	✗
K	✓	✓	✗	✓	✓	✓	✗	✓	✗	✓	✗	✓	✗	✗	✗
$\mathcal{L}$	✓	✗	✓	✓	✓	✓	✓	✗	✓	✗	✗	✗	✗	✗	✓
4	**32**	33	**32**	45	**32**	48	33	**32**	45	45	**32**	46	45	34	nan
3	36	**35**	36	49	37	50	38	39	49	50	37	50	49	44	nan
2	**44**	**44**	46	53	43	59	40	56	53	58	45	55	51	45	nan

Table 3. **Ablation** – PMPJPE↓ of our method on Human3.6M with different number of cameras with different inputs passed to the neural optimizer. Heatmap **H** contributes most to the final performance, but all inputs are necessary to achieve the state-of-art performance.

7. Supplementary

7.1. Acknowledgement

We would like to thank Bastian Wandt, Nori Kanazawa and Diego Ruspini, Thomas Leung for help with CanonPose, stacked hourglass pose estimator, and AniPose. This work was financially supported by NSF, DARPA and Google.

7.2. Ablation Tables

- **Table 3** shows the performance of the neural optimizer trained with different subsets of inputs;
- **Table 4** shows the latency breakdown across model components and models;
- **Table 5** shows that the performance of equivariant (S1+S2) and non-equivariant (S1+S2/MLP) models differs by at most 4mm on both datasets;
- **Table 6** shows that MetaPose outperforms prior work with corresponding supervision signals;
- **Table 7** shows that the personalized bone length prior improves the performance of both the iterative and neural refiners in the majority of cases;
- **Table 8** shows that the student-teacher loss inspired by Ma et al. [45] to draw the predicted solution into the correct basin of the loss hurts the performance in all cases;
- **Table 9** summarizes reference performance of monocular pose estimation components across different splits of data (train, val, test) for reproducibility, and shows strong overfitting on SkiPose;
- **Table 10** shows that at least 20mm of error is due to imperfect heatmaps, up to 10mm is due to the weak camera model, and only up to 3mm is due to imperfect init;
- **Table 11** shows that on H36M with just 1/5th of the entire training dataset (i.e. 5k labeled training samples, each sample containing several cameras) we can get a model that achieves PMPJPE within 5-10mm of the performance we achieve on full data.
- **Table 12** shows the effect of varying the number of Gaussian mixture components on the performance of different methods.

7.3. Extended Results

A TensorFlow implementation, videos with predictions on Human36M and SkiPose, additional tables with camera estimation errors, as well as qualitative results on KTI Football [34] dataset can be found on our project website: https://metapose.github.io/.

7.4. Extended Related Work

Human body priors. Early methods [43] for human pose estimation from uncalibrated video streams relied on conservation of length between rotational joints (i.e. bones lengths) over time. Current state-of-the-art methods use parametric full-body priors based on skinning and blend shapes learned from thousands of 3D body scans of real subjects, such as SMPL [44] and SMPL-X [51]. SMPLy [42] is a common algorithm and a common benchmark for monocular 3D pose estimation using SMPL. Many pose estimation models utilize strong prior of SMPL [44] to improve the downstream performance in other setups. For example, Arnab et al. [5] performs bundle adjustment over estimated SMPL parameters and 2D poses aggregates over time to account for temporal consistency. Dong et al. [14] used SMPL parameters estimated from many short videos of subjects performing the same action from different viewpoints found on the internet (e.g. squat, tennis serve, etc.) to build a better model of these actions that accounts for self-occlusion present in individual videos. MonoClothCap [64] adds a deformable clothing model on top of SMPL [44] and an off-the-shelf CNN trained to estimate surface normals from individual frames.

Learnable optimizers. Starting from the seminal work of Andrychowicz et al. [4], several works proposed training LSTM-based general purpose neural optimizers. For example, LS-Net [12] is an LSTM-based neural solver that predicts a sequence of updates for estimated depth maps and camera poses from a continuous RGB video. LS-Net takes task-specific Hessian approximations as input and uses ground truth depths and poses for supervision. More recent examples include RAFT [58] that uses a recurrent transformer to estimate optical from from a video by computing a full 4D cross-frame correlation volume and iteratively refining the optimal flow to minimize the error with respect to the ground truth flow using a learned optimizer.

Neural solvers for ill-posed problems. Adler and Öktem [1] rigorously analyzed how learned parametric (neural) updated schemes affect the performance of inverse solvers in the ill-posed inverse tomography setup and proposed the learned gradient decent (LGD). Adler and Öktem [2] showed that replacing a proximal operator in the proximal primal-dual optimization schemes further improves the performance of inverse solvers on the inverse tomography task and leads to faster convergence. For example, DeepView

	PoseNet	GMM	S1	Solver	Total	Error [mm]
AniPose	$0.03 \cdot 4$	-	-	7	7.1	75
MetaPose (S1)	$0.03 \cdot 4$	-	$0.01 \cdot 4$	-	**0.15**	74
MetaPose (S1+S2)	$0.03 \cdot 4$	$0.01 \cdot 4$	$0.01 \cdot 4$	0.006	0.2	**40**
MetaPose (S1+IR)	$0.03 \cdot 4$	$0.01 \cdot 4$	$0.01 \cdot 4$	1.5	1.7	43
AniPose	$0.5 \cdot 4$	-	-	10	12	75
MetaPose (S1)	$0.5 \cdot 4$	-	$0.20 \cdot 4$	-	**2.8**	74
MetaPose (S1+S2)	$0.5 \cdot 4$	$0.25 \cdot 4$	$0.20 \cdot 4$	0.01	4	**40**
MetaPose (S1+IR)	$0.5 \cdot 4$	$0.25 \cdot 4$	$0.20 \cdot 4$	3.5	7.5	43

Table 4. Latency breakdown in seconds for estimating the full 3D pose on H36M with four cameras on a GPU (V100, top) and a CPU (bottom) across four components: per-view 2D heatmap estimation (PoseNet), heatmap GMM fitting, per-view monocular 3D and initialization, multi-view bundle adjustment (neural network forward pass in case of MetaPose S1+S2, Adam [35] in case of S1+IR, and a 2nd-order CPU-only TRR [7, 9] solver in case of AniPose). MetaPose (S1+S2) achieves lowest error with an at least six times (on GPU; two times on CPU) faster inference as the iterative refiner.

[16] uses these ideas to accelerate inference of multi-plane images (MPIs) with LGD. Song et al. [56] also used LGD to accelerate fitting of SMPL [44] human body model to monocular images of human subjects.

7.5. Weighted EM-algorithm for spherical GMM

We used grid points x_i weighted by corresponding probabilities p_i to fit a GMM to a 2D probability heatmap. Following Frisch and Hanebeck [17] on each step $t = 0 \dots T$ of the EM algorithm we performed usual **(non-weighted)** E-step to compute the new assignment matrix $\eta_{i,m}^{(t+1)}$ between points x_i and spherical clusters $m = 0 \dots M$ with means $\mu_m^{(t)}$, and standard variations $\sigma_m^{(t)}$, and weights $w_m^{(t)}$,

$$\eta_{i,m}^{(t+1)} = \frac{w_m^{(t)} \mathcal{N}(x_i | \mu_m^{(t)}, \sigma_m^{(t)} \cdot I)}{\sum_{m'} w_{m'}^{(t)} \mathcal{N}(x_i | \mu_{m'}^{(t)}, \sigma_{m'}^{(t)} \cdot I)}$$

where $\mathcal{N}(x|\mu, \Sigma)$ is a two-dimentional Gaussian pdf, followed by a **weighted** M-step:

$$w_m^{(t+1)} = \frac{\sum_i \eta_{i,m}^{(t+1)} p_i}{\sum_{m'} \sum_i \eta_{i,m'}^{(t+1)} p_i}$$

$$\mu_m^{(t+1)} = \frac{\sum_i \eta_{i,m}^{(t+1)} p_i x_i}{\sum_i \eta_{i,m}^{(t+1)} p_i}$$

$$\sigma_m^{(t+1)} = \sqrt{\frac{\sum_i \eta_{i,m}^{(t+1)} p_i ||x_i - \mu_m^{(t+1)}||_2^2}{\sum_i \eta_{i,m}^{(t+1)} p_i}}$$

7.6. Camera model

First, weak camera model enables closed-form estimation of camera parameters from monocular 3D pose estimates in Stage 1, as shown in Section 7.8. Moreover, the weak is easier to use as a part of a learning algorithm, since it avoids re-projection singularities during training (e.g. when predicted joint depth approaches zero). More specifically, with the weak camera model, we can avoid singularities by replacing the problematic f/Z_{avg} term with a single inferred non-negative scale, as discussed in the beginning of Section 3. This is not an issue for AniPose because there is no training involved, and it is initialized with GT camera parameters making singularities extremely unlikely. In prior work, Imry Kissos et al. [23] and Kocabas et al. [39] showed that estimating camera parameters from images and using full perspective projection with SMPL [44] yields results superior to the classical weak-camera SMPL.

7.7. Implementation Details

Progressive training of refinement steps. Expanding upon the "progressive" training in Eq. (16): the first refinement step network $\mathcal{F}_\theta^{(1)}$ is first trained to predict residuals of pose/camera parameters that minimize the reprojection loss (16) starting from the initial guess (S1). The second refinement step network $\mathcal{F}_\theta^{(2)}$ is trained analogously - to minimize the reprojection loss (16), but starting from the **fixed** guess of the first network - no backprop to the first network $\mathcal{F}_\theta^{(1)}$.

Architecture. For monocular 2D pose estimation we used the stacked hourglass network [49] pre-trained on COCO pose dataset [20]. We additionally trained a linear regression adapter to convert between COCO and H36M label formats (see supplementary Figure 8 for labeling format comparison). The resulting procedure yields good generalization on H36M, as shown in supplementary Table 9). The COCO-pretrained network generalized very poorly to SkiPosePTZ dataset because of the visual domain shift, so we fine-tuned the stacked hourglass network using ground truth 2D labels. For monocular 3D estimates used in Stage 1, we applied EpipolarPose [37] on Human3.6M and CanonPose [62] on SkiPosePTZ. We would like to note that, despite the significant shift in the labeling format between

Figure 8. Both H36M ground truth poses, COCO dataset (used to train the hourglass network), and EpipolarPose predictions (used to generate the 3D initialization) have different label formats from H36M. We trained a small "adapter" to convert COCO-to-H36M, and used EpipolarPose predictions as-is.

predictions of these monocular 3D methods and the format used in datasets we used for evaluation, this does not affect the quality of camera initialization we acquired via rigid alignment. Similar to prior work [45], each "neural optimizer step" is trained separately, and the fresh new neural net is used at each stage, and stop gradient is applied to all inputs. For MLP architecture, we used L fully-connected 512-dimensional layers followed by a fully-connected 128-dimensional, all with selu with L=4 for H36M and L=2 for SkiPose. For equivalent network, the optimal network for H36M had following layers: [512, 512, CC, 512, 512, CC, 512] and for SkiPose had following layers: [512, 512, CC, 512, 512, CC, 512, 512, CC, 512, 512, CC, 512] - where CC corresponds to concatenation of first two moments and numbers correspond to dense layers with SeLU [36]. We re-trained each stage multiple times until the *validation* PMPJPE improved or the total number of "stage training attempts" exceeded 100.

Hyperparameters. We used Adam [35] optimizer with learning rate `1e-2` for 100 steps for exact refinement, and `1e-4` for the neural optimizer.

Reference 2D performance. Tables 9 shows performance of 2D pose prediction networks and the resulting MetaPose network on different splits of different datasets. It shows that both the 2D network and MetaPose to certain degree overfit to SkiPose because of its smaller size.

7.8. Closed Form Expressions for Stage 1

Below we describe the solution to the rigid alignment problem (12) for monocular 3D pose estimates $\mathbf{q}_c$ and inferred weak camera parameters from them. Assume that we have monocular 3D predictions $\mathbf{q}_c$ in frame of the camera c. The parameters of the first camera are assumed to be known and fixed

$$R^{(0)}_{\text{init}}{=}I, \boldsymbol{t}^{(0)}{=}\bar{0}, s^{(0)}{=}1$$

whereas the rotation of other cameras are inferred using optimal rigid alingment $R^{(c)}_{\text{init}} = (U^{(c)})^T V^{(c)}$ where

$$U^{(c)}, \Lambda, V^{(c)} = \text{SVD}(\text{centered}(\mathbf{q}_c) \cdot \text{centered}(\mathbf{q}_0))^T)$$

The scale s and shift $\boldsymbol{t}$ can be acquired by comparing the original monocular $\mathbf{q}_{c,[:,0:2]}$ in pixels to $[R^{(c)}_{\text{init}} \text{centered}(\mathbf{q}_0)]_{[:,0:2]}$ rotated back into each camera frame, for example:

$$s^{(c)}_{\text{init}} = \frac{||[(R^{(c)}_{\text{init}})^T \text{centered}(\mathbf{q}_0)]_{[:,0:2]}||}{||\,\text{centered}(\mathbf{q}_c)_{[:,0:2]}||} \tag{20}$$

$$\boldsymbol{t}^{(c)}_{\text{init}} = \hat{\mu}([(R^{(c)}_{\text{init}})^T \text{centered}(\mathbf{q}_0)]_{[:,0:2]}) - \hat{\mu}([\mathbf{q}_c]_{[:,0:2]})) \tag{21}$$

where $\hat{\mu}(\boldsymbol{a}) = (\sum_k^K \boldsymbol{a}_k)/K$ is the center of the 3D pose and $\text{centered}(\boldsymbol{a})_k = (\boldsymbol{a}_k - \hat{\mu}(\boldsymbol{a}))$ and the initial pose estimate is the average of aligned, rotated and predictions from other cameras. The initial guess for the pose is the average of all monocular poses rotated into the first camera frame:

$$\boldsymbol{J}_{\text{init}} = \frac{1}{C}\sum_{c=0}^{C}(s^{(c)}_{\text{init}} \cdot R^{(c)}_{\text{init}} \text{centered}(\mathbf{q}_c)) + \hat{\mu}(\mathbf{q}_0) \tag{22}$$

7.9. 6D rotation re-parameterization

We used for following parameterization: $R(x, y) = \text{stack}[n(x), n(x \times y), n(x \times (x \times y))]$ where $n(x)$ is a normalization operation, and $a \times b$ is a vector product. This is essentially Gram-Schmidt orthogonalization. Rows of the resulting matrix is guaranteed to form an orthonormal basis. This rotation representation was shown to be better suited for optimization [70].

7.10. Stable Gaussian Mixture Likelihood

We used the following numerically stable spherical GMM log-likelihood to compute (6):

$$\log\left[\sum_r w_r \cdot \frac{\exp(-\frac{1}{2}(\boldsymbol{x}-\boldsymbol{\mu}_r)^T \cdot (\sigma_r^2 \cdot I)^{-1}(\boldsymbol{x}-\boldsymbol{\mu}_r)}{\sqrt{(2\pi)^2\sigma_r^4}}\right]$$
$$= \log\sum_r \exp\left[\log\left(\frac{w_r}{2\pi\sigma_r^2}\right) - \frac{||\boldsymbol{x}-\boldsymbol{\mu}_r||^2}{2\sigma_r^2}\right]$$
$$= \text{LSE}_r\left[\log\left(\frac{w_r}{2\pi\sigma_r^2} + \varepsilon\right) - \frac{||\boldsymbol{x}-\boldsymbol{\mu}_r||^2}{2\sigma_r^2}\right]$$

where (μ, σ^2, w) are mean, variance and weight of the corresponding mixture component, $\mathrm{LSE}(l_0, \ldots, l_r)$ is a numerically stable "log-sum-exp" often implemented as $\mathrm{LSE}(l_0, \ldots, l_r) = l^* + \log(\sum_k \exp(l_k - l^*))$, where $l^* = \max(l_0, \ldots, l_r)$, and ε is a small number.

7.11. Teacher loss

In addition to the reprojcetion loss, the student-teacher ablation (S2/TS) used the following additional loss inspired by Ma et al. [45] to draw the predicted solution $\boldsymbol{J}_{\text{neur}}$ into the basin of the correct solution by penalizing its deviation from the solution $\boldsymbol{J}_{\text{ref}}$ produced by the iterative refiner (IR).

$$\begin{aligned}\mathcal{L}_y\big(&\boldsymbol{J}_{\text{neur}}, R^{(c)}_{\text{neur}}, \boldsymbol{t}^{(c)}_{\text{neur}}, s^{(c)}_{\text{neur}}; \boldsymbol{J}_{\text{ref}}, R^{(c)}_{\text{ref}}, \boldsymbol{t}^{(c)}_{\text{ref}}, s^{(c)}_{\text{ref}}\big)\\ &= \lambda_p \cdot ||\boldsymbol{J}_{\text{neur}} - \boldsymbol{J}_{\text{ref}}||_2^2 + \lambda_t \cdot \sum ||\boldsymbol{t}^{(c)}_{\text{neur}} - \boldsymbol{t}^{(c)}_{\text{ref}}||_2^2\\ &+ \lambda_R \cdot \sum ||(R^{(c)}_{\text{neur}})^T R^{(c)}_{\text{ref}} - I||_2^2\\ &+ \lambda_s \cdot \sum ||\log(s^{(c)}_{\text{neur}}) - \log(s^{(c)}_{\text{ref}})||_2^2. \end{aligned} \tag{23}$$

Table 8 shows that it hurts the performance of the model.

7.12. Qualitative Results

We provide qualitative examples (failure cases, success cases) on the test set of H36M and SkiPose in Figures 9-19. Videos with more test prediction visualizations are available at https://metapose.github.io/. Circles around joints on 2D views represent the absolute *reprojection error* for that joint for that view. Our qualitative findings:

1. MetaPose considerably improves over the initial guess when a lot of self-occlusion is present

2. MetaPose fails on extreme poses for which monocular estimation fails (e.g. somersaults)

3. In two-camera SkiPose setup, AniPose often yields smaller reprojection error while producing very bad 3D pose results

7.13. Other 3D pose estimation datasets

To our knowledge SkiPose is the only publicly available annotated multi-view dataset with moving cameras actively used in recent prior work. For example, H36M [24], CMU [28], 3DHP [47], Shelf&Campus [6], TotalCapture[59] - all have fixed cameras. [57] and [8] used internal datasets. KTI Multiview Football [34] is an older and smaller dataset that has not been used for quantitative evaluation in the past eight years, precluding meaningful comparison to prior work. Analogously to [26] additional qualitative results on KTI Multiview Football dataset that can be found in the **extended supplementary** (https://metapose.github.io/), but no quantitative comparison to recent prior work can be made.

7.14. Generalization error

The ability of our method to generalize to a new dataset can be factorized into three components: generalization of the off-the-shelf networks on *new visual domains* (already verified in prior work), generalization of the neural optimizer on *new pose distributions*, and to *new camera configurations*. We show that our method successfully generalizes across pose distributions (greet/sit/call in H36M vs skiing in SkiPose), and to differences between actors present in train and test sequences of both H36M and SkiPose. We also show that it consistently generalizes across two camera setups (fixed short-focus in H36M, and moving long-focus in SkiPose) and all subsets of given cameras in each setup. Overall, prior work shows that the off-the-shelf components of the proposed method successfully adapt to new visual domains, and our experiments show that the neural optimizer is resilient against the remaining sources of generalization error. Moreover, Wang et al. [63] showed that models that jointly predict camera parameters and 3D poses better generalize to novel visual domains.

(a) H36M

Method	**4**	**3**	**2**
Metapose (S1+S2)	32	36	44
Metapose (S1+S2/MLP)	**30**	**35**	**41**

(b) SkiPose

Method	**6**	**4**	**2**
Metapose (S1+S2)	**42**	**45**	**50**
Metapose (S1+S2/MLP)	46	44	54

Table 5. Equivariant (S1+S2) and non-equivariant (S1+S2/MLP) performance networks have comparable performance across different numbers of cameras on H36M (top) and SkiPose (bottom).

(a) H36M

Method and supervision type		PMPJPE↓		NMPJPE↓		Δt [s]
		4	2	4	2	
Isakov et al. [26]	**3D**	20.8	-	-	-	-
AniPose [33] w/ GT	S	74.6	167.3	103.1	229.8	7.1
Rhodin et al. [53]	2/3D	65.1	-	80.1	-	-
CanonPose [62]	S	53.0	-	81.9	-	-
EpipolarPose (EP) [37]	S	70.7	-	77.7	-	-
Iqbal et al. [25]	2D	54.5	-	64.5	-	-
MetaPose (S1)	S	74.3	87.2	83.4	94.8	0.2
MetaPose (S1+S2)	2D	**32.1**	**44.2**	**48.8**	**54.6**	0.2
MetaPose (S1+IR)	S	43.2	65.9	52.8	75.4	1.7
MetaPose (S1+S2/SS)	S	39.2	50.4	55.9	63.3	0.2

(b) SkiPose

Method and supervision type		PMPJPE↓		NMPJPE↓		Δt [s]
		6	2	6	2	
AniPose [33] w/ GT	S	50.3	62.4	220.8	272.7	7.1
Rhodin et al. [53]	2/3D	-	-	85.1	-	-
CanonPose (CP) [62]	S	89.6	-	128.1	-	-
MetaPose (S1)	S	81.2	86.4	140.3	143.7	0.3
MetaPose (S1+S2)	2D	42.1	**49.9**	**53.2**	**59.3**	0.4
MetaPose (S1+IR)	S	**30.3**	77.1	53.7	94.2	2.5
MetaPose (S1+S2/SS)	S	42.4	94.9	59.3	101.8	0.4

Table 6. **MetaPose outperforms SotA on H36M (top) and SkiPose (bottom)** – Same notation as in Table 1. Also includes the self-supervised (S2/SS) and iterative solver (SS/IR) flavours of MetaPose. Supervision signal used *during training*: 2D - ground truth 2D keypoints, 3D - ground truth 3D poses, S - self-supervision (i.e. using a pose estimation network pre-trained on a different dataset), 2/3D - 2D keypoint data with 3D poses on few subjects.

(a) H36M

Method	**4**	**3**	**2**
Metapose S1+S2	32	36	44
Metapose S1+S2 **+ bone**	**31**	**34**	**37**
Metapose S1+IR	43	52	53
Metapose S1+IR **+ bone**	**38**	**44**	**47**
Metapose S1+S2/SS	39	47	**50**
Metapose S1+S2/SS **+ bone**	**38**	**45**	**50**

(b) SkiPose

Method	**6**	**4**	**2**
Metapose S1+S2	**41**	**43**	**47**
Metapose S1+S2 **+ bone**	45	46	49
Metapose S1+IR	30	32	77
Metapose S1+IR **+ bone**	**26**	**28**	**46**
Metapose S1+S2/SS	41	46	95
Metapose S1+S2/SS **+ bone**	**44**	**45**	**53**

Table 7. Personalized **bone lengths prior** helps in all cases for H36M (top), especially in the few-camera setup; and in the majority of cases on SkiPose (bottom).

(a) H36M

Method	**4**	**3**	**2**
MetaPose S1+S2	**32**	**36**	**44**
MetaPose S1+S2/TS	38	45	45

(b) SkiPose

Method	**6**	**4**	**2**
MetaPose S1+S2	**42**	45	**50**
MetaPose S1+S2/TS	**42**	**43**	72

Table 8. **Teacher-student loss** analogous to the one proposed by Ma et al. [45] to bring the neural optimizer into the basin of the correct solution either hurts or does significantly affect the performance in all cases.

Metric	Train			Validation			Test		
GT log-prob.	-5.17			-5.58			-5.06		
Stage:	S1	S1+IR	S1+S2	S1	S1+IR	S1+S2	S1	S1+IR	S1+S2
Pred log-prob.	-4.22	-5.86	-5.00	-4.6	-6.07	-5.41	-3.92	-5.77	-4.95
PMPJPE [mm]	69	38	15	65	34	17	74	43	32
NMPJPE [mm]	78	58	36	69	56	46	88	66	49
MSE 2D (10^{-4})	15	5	0.6	6	2	0.6	20	7	5

(a) H36M

Metric	Train			Validation			Test		
GT log-prob.	-5.49			-5.49			-4.81		
Stage:	S1	S1+IR	S1+S2	S1	S1+IR	S1+S2	S1	S1+IR	S1+S2
Pred log-prob.	-2.83	-5.79	-5.49	-2.90	-5.75	-4.51	-3.04	-5.59	-5.33
PMPJPE [mm]	71	17	1	72	17	10	80	30	42
NMPJPE [mm]	139	35	1	143	38	15	140	54	53
MSE 2D (10^{-4})	34	6	0.01	37	6	1	30	7	7

(b) SkiPose

Table 9. **Details about predictions across different stages: initialization using monocular 3d (S1), iterative refinement (S1+IR), and neural refinement (S1+S2) on H36M with four cams (top) and SkiPose with six cams (bottom)**. 2D error is scaled so that the entire pose lies in $[0, 1]^2$. The GT log probability is the log probability of ground truth points given predicted heatmaps and measures how well heatmaps generated by our 2D prediction network match the ground truth. Significantly larger discrepancy between GT log probabilities on train and test on SkiPose shows that 2D pose network overfits much more on SkiPose than on H36M due to its limited size.

(a) H36M

Method	**2D**	**S1**	**4**	**3**	**2**
S1+IR	HT	EP	43	52	53
S1+IR	HT	GT	40	49	48
S1+IR	full-GT	EP	17	20	24
S1+IR	full-GT	GT	14	16	20
S1+IR	weak-GT	EP	4	6	18
S1+IR	weak-GT	GT	1.4	1.7	2

(b) SkiPose

Method	**2D**	**S1**	**6**	**4**	**2**
S1+IR	HT	CP	30	33	77
S1+IR	HT	GT	28	30	41
S1+IR	full-GT	CP	8	8	29
S1+IR	weak-GT	CP	8	7	28

Table 10. MetaPose S1+IR trained with either ground truth pseudo-heatmaps centered around full and **weak-projected 3D joints** and with different S1 **initialization** (either predicted via EpipolarPose or "perfect"). This experiment shows that imperfect heatmaps contribute to at least 20mm of error in both cases, weak camera model contribute to 10mm of error on H36M and no error on SkiPose, and imperfect initialization contributes to at most 3mm of error.

# cam \ %	100	89	84	79	73	68	63	58	52	47	42	37	31	29	26	24	21	18	16	13	10	8	5	3
4	**32**	**32**	**32**	**32**	33	36	33	35	33	35	34	42	37	39	36	38	36	41	48	41	41	44	48	70
3	36	**35**	36	37	37	36	37	39	37	37	38	39	40	39	41	45	42	43	44	45	46	51	53	70
2	44	48	48	47	46	48	48	**40**	54	60	43	43	51	57	53	58	54	50	52	48	48	51	68	87

Table 11. **Test PMPJE of MetaPose on H36M as a function of the fraction of training examples with 2D ground truth used (i.e. first X%)**. Reminder: we **never** use any ground truth 3D annotations for either cameras or poses, these are percentages of 2D labels used for training. We can see that MetaPose produces high-accuracy predictions (within 10mm of the original performance) with up to 1/5-th (≈18%) of the H36M training 2D pose annotations (≈5k training examples each containing multiple cameras). The few-camera setup exhibits more variations in test error due to random network initialization.

(a) H36M

Method	**GMM**	**4**	**3**	**2**
MetaPose S1+IR	4	43	52	53
MetaPose S1+IR	3	42	51	52
MetaPose S1+IR	2	42	51	52
MetaPose S1+IR	1	42	52	53
MetaPose S1+S2	4	32	39	44
MetaPose S1+S2	3	31	36	47
MetaPose S1+S2	2	32	36	50
MetaPose S1+S2	1	32	36	48

(b) SkiPose

Method	**GMM**	**6**	**4**	**2**
MetaPose S1+IR	4	30	33	77
MetaPose S1+IR	3	30	32	77
MetaPose S1+IR	2	31	34	75
MetaPose S1+IR	1	43	43	58
MetaPose S1+S2	4	42	45	50
MetaPose S1+S2	3	44	41	50
MetaPose S1+S2	2	42	49	51
MetaPose S1+S2	1	41	43	47

Table 12. The number of Gaussian Mixture components does not significantly affect the performance of the network in all cases on both SkiPose (top) and H36M (bottom), except for MetaPose S1+IR on SkiPose with a single Gaussian.

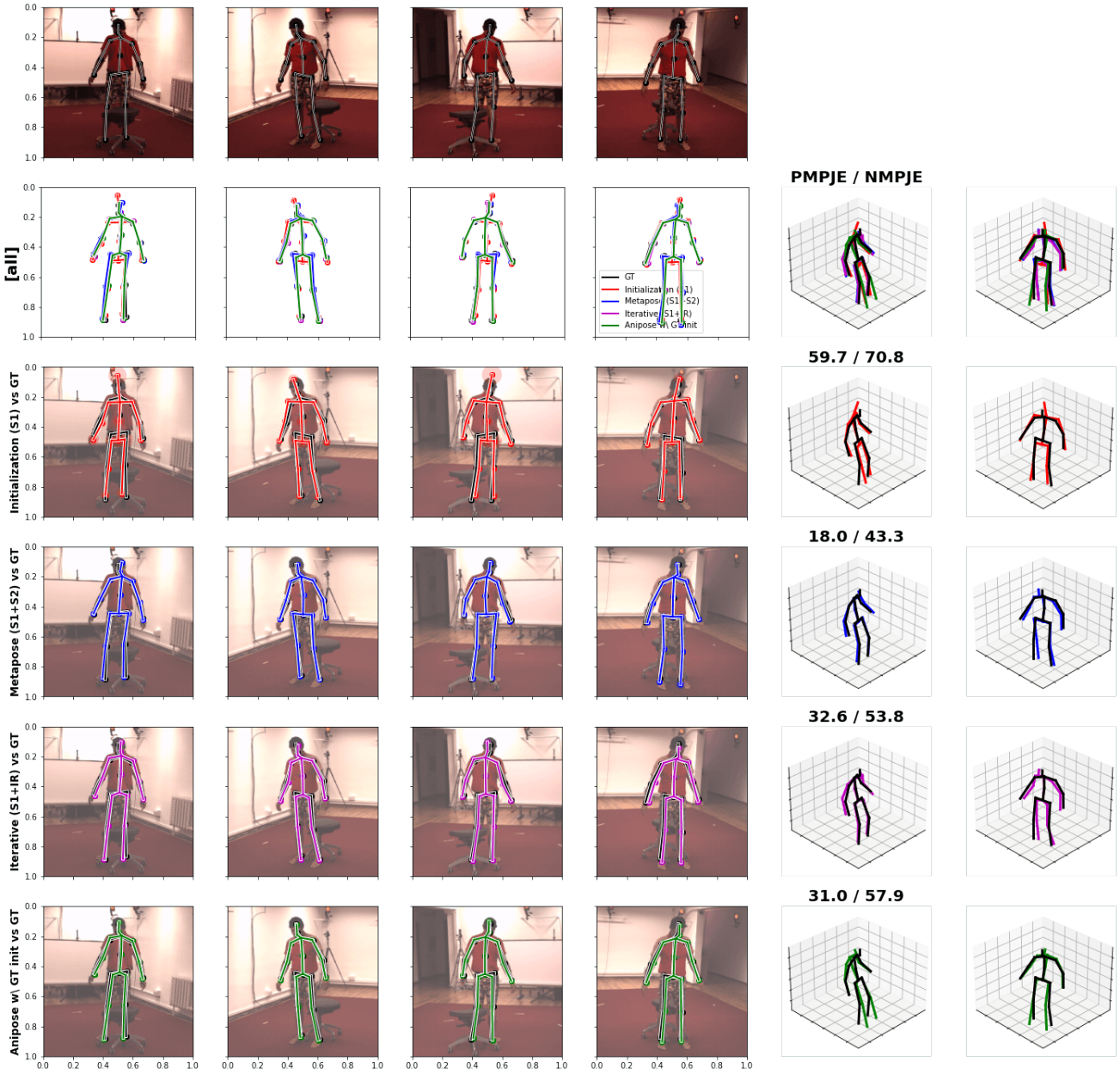

Figure 9. Full MetaPose (S1+S2) outperforms initialization (S1), Iterative Solver (S1+IR), and AniPose w/ GT camera init.

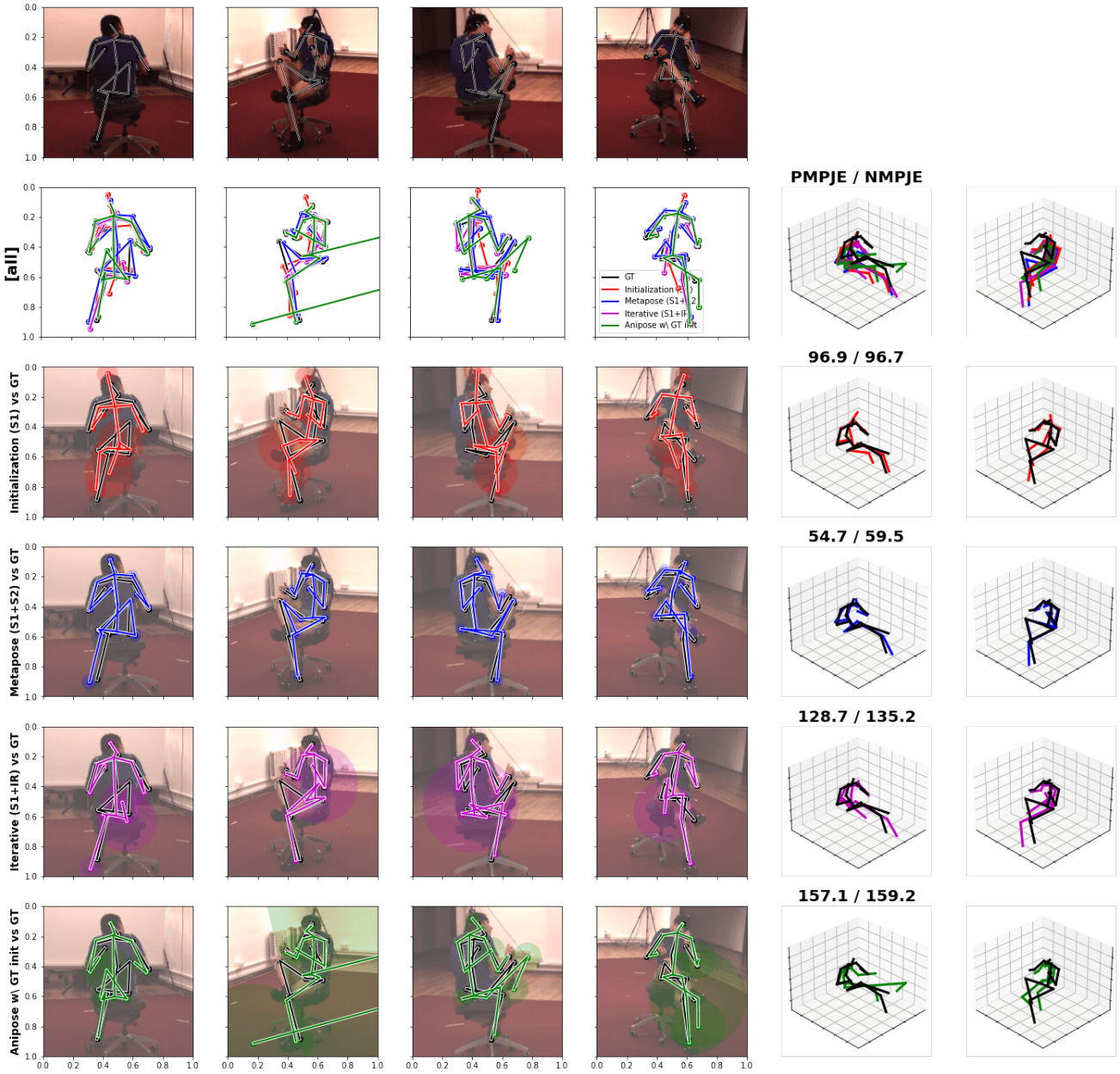

Figure 10. MetaPose improves over the initial guess under high self-occlusion.

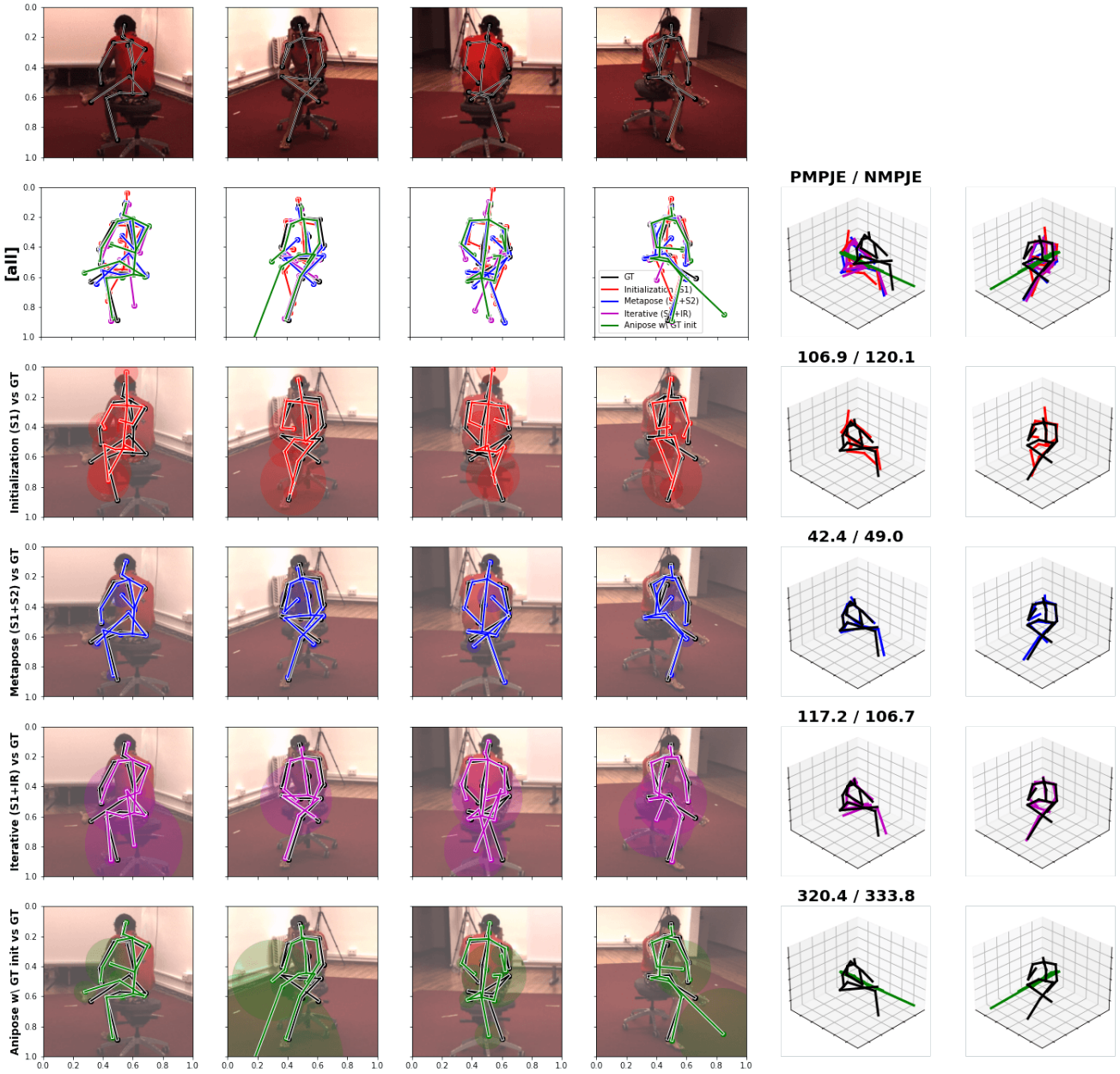

Figure 11. MetaPose improves over the initial guess under high self-occlusion.

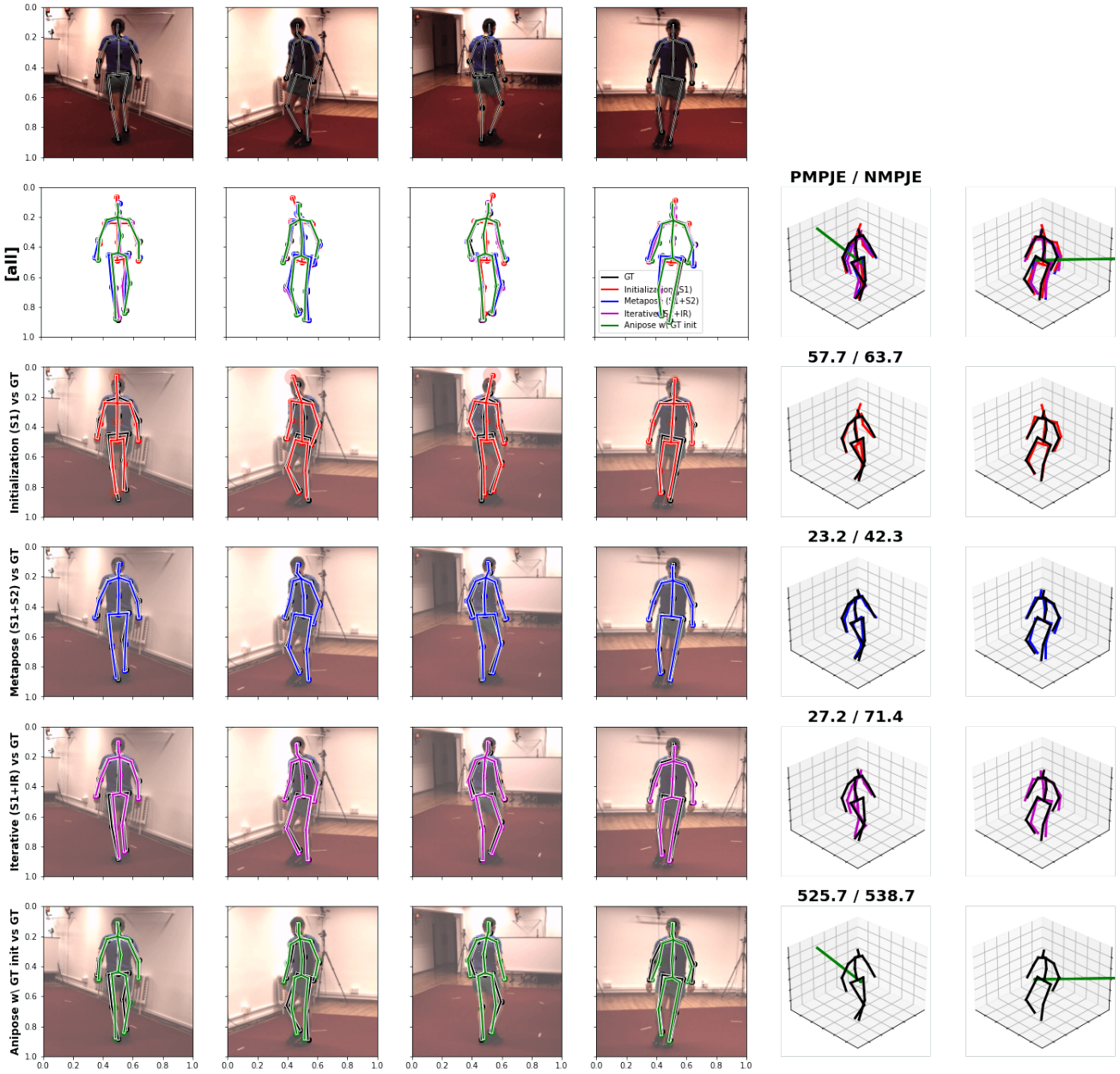

Figure 12. AniPose w/ GT camera initialization can yields low re-projection error but high 3D estimation error.

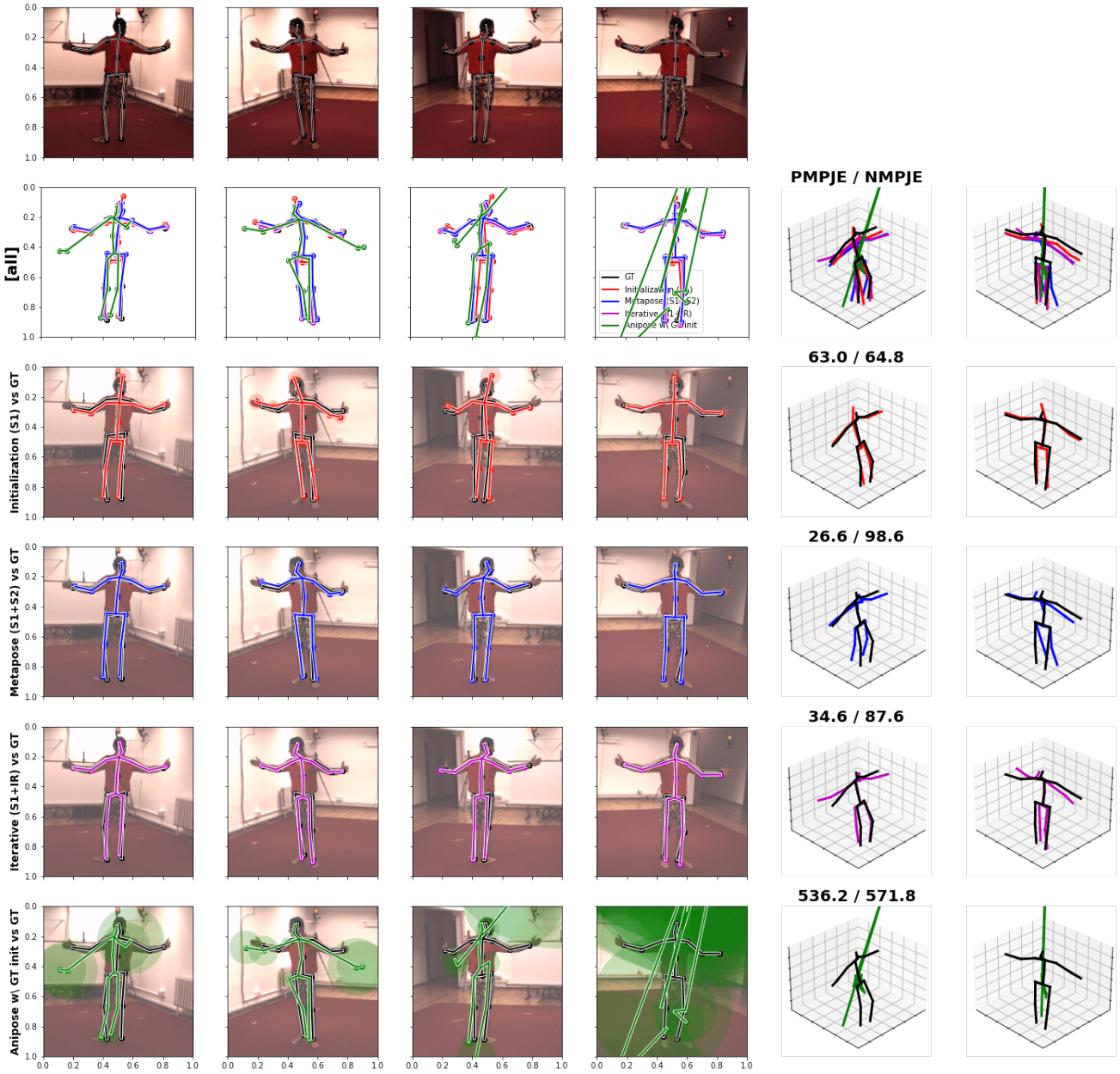

Figure 13. AniPose with GT init fails due to poor choice of 2D predictions to ignore during refinement.

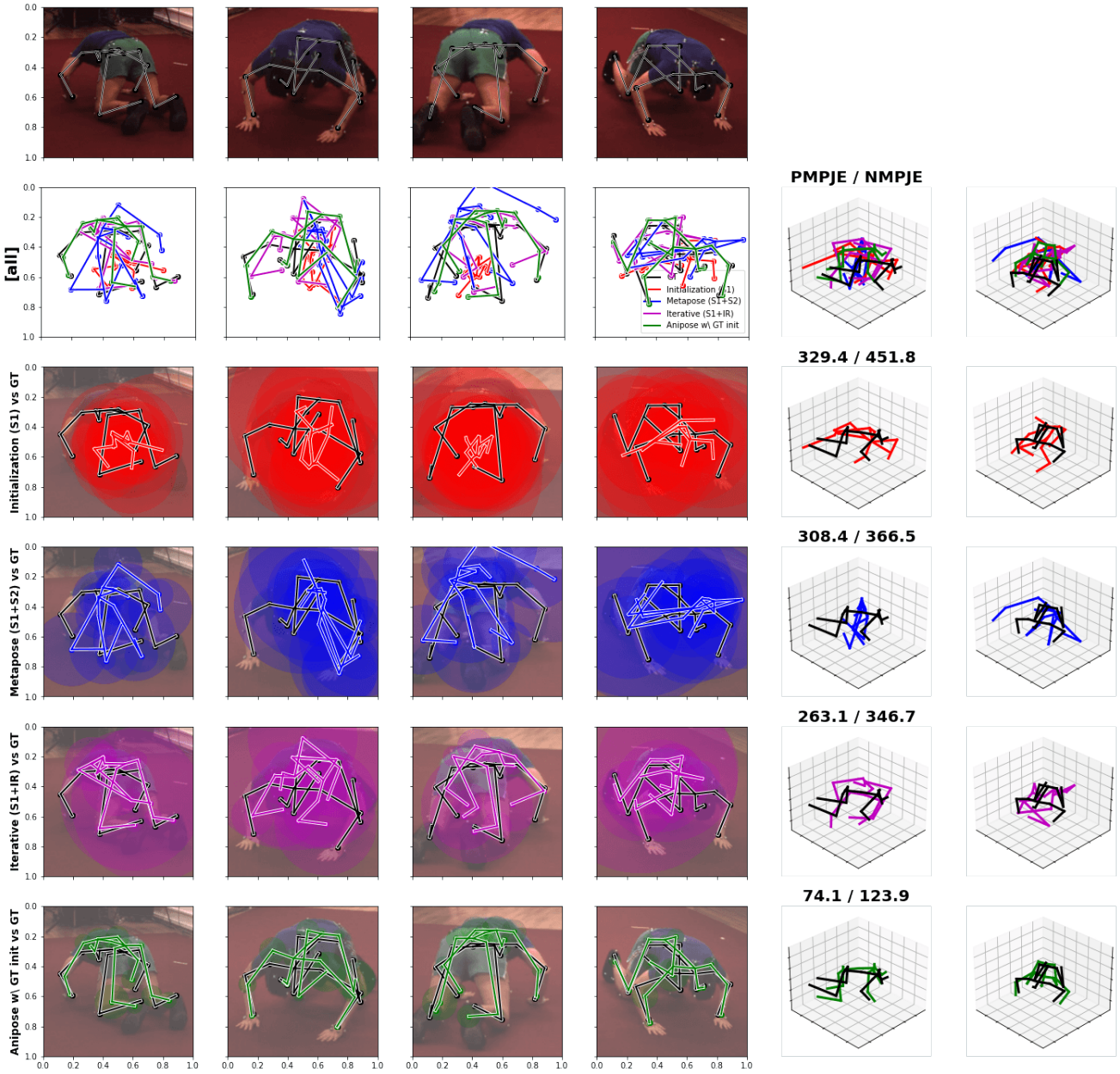

Figure 14. MetaPose fails on few extreme poses that have much poorer than average initialization quality.

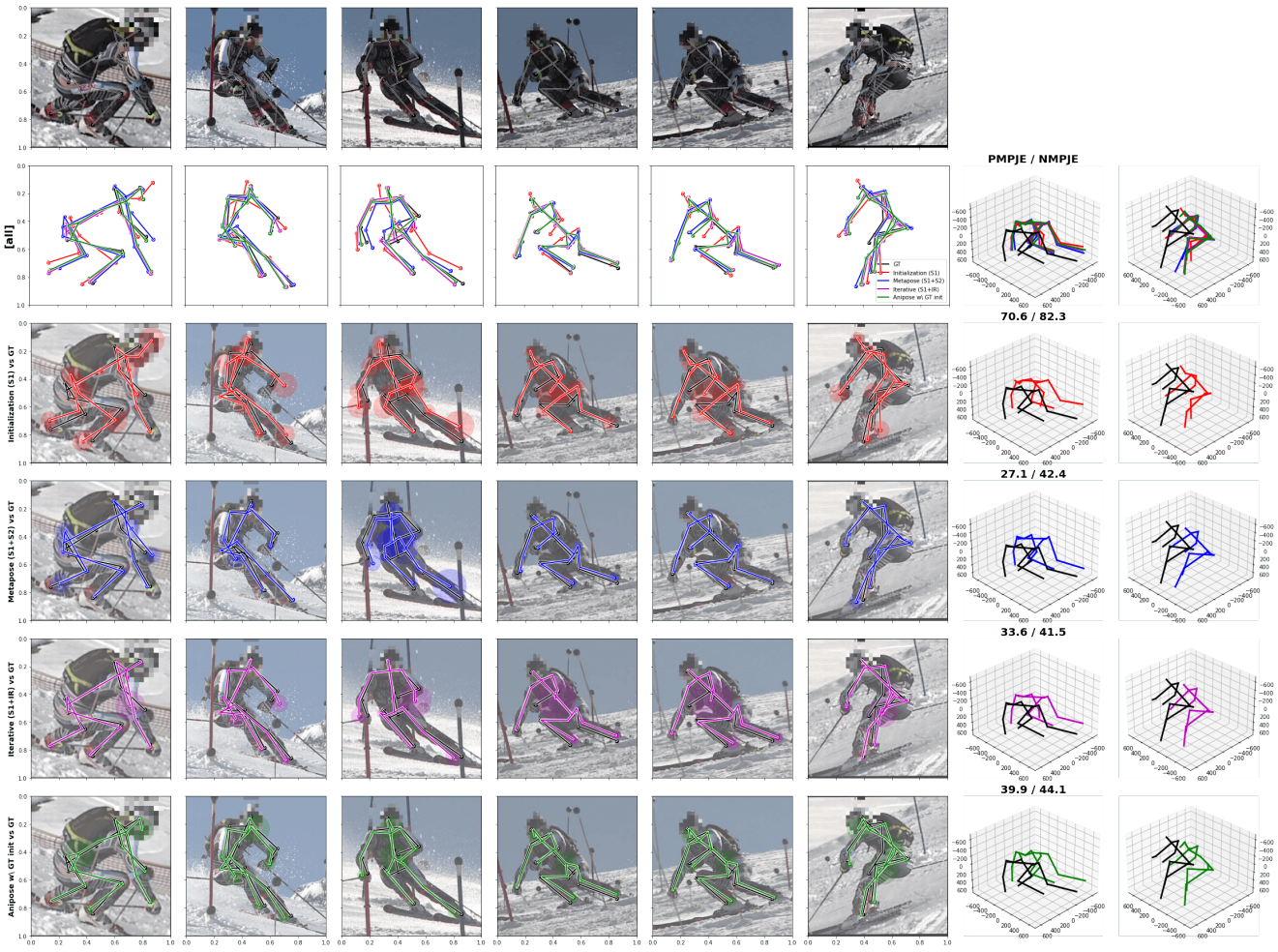

Figure 15. MetaPose improves over the initial guess under high self-occlusion.

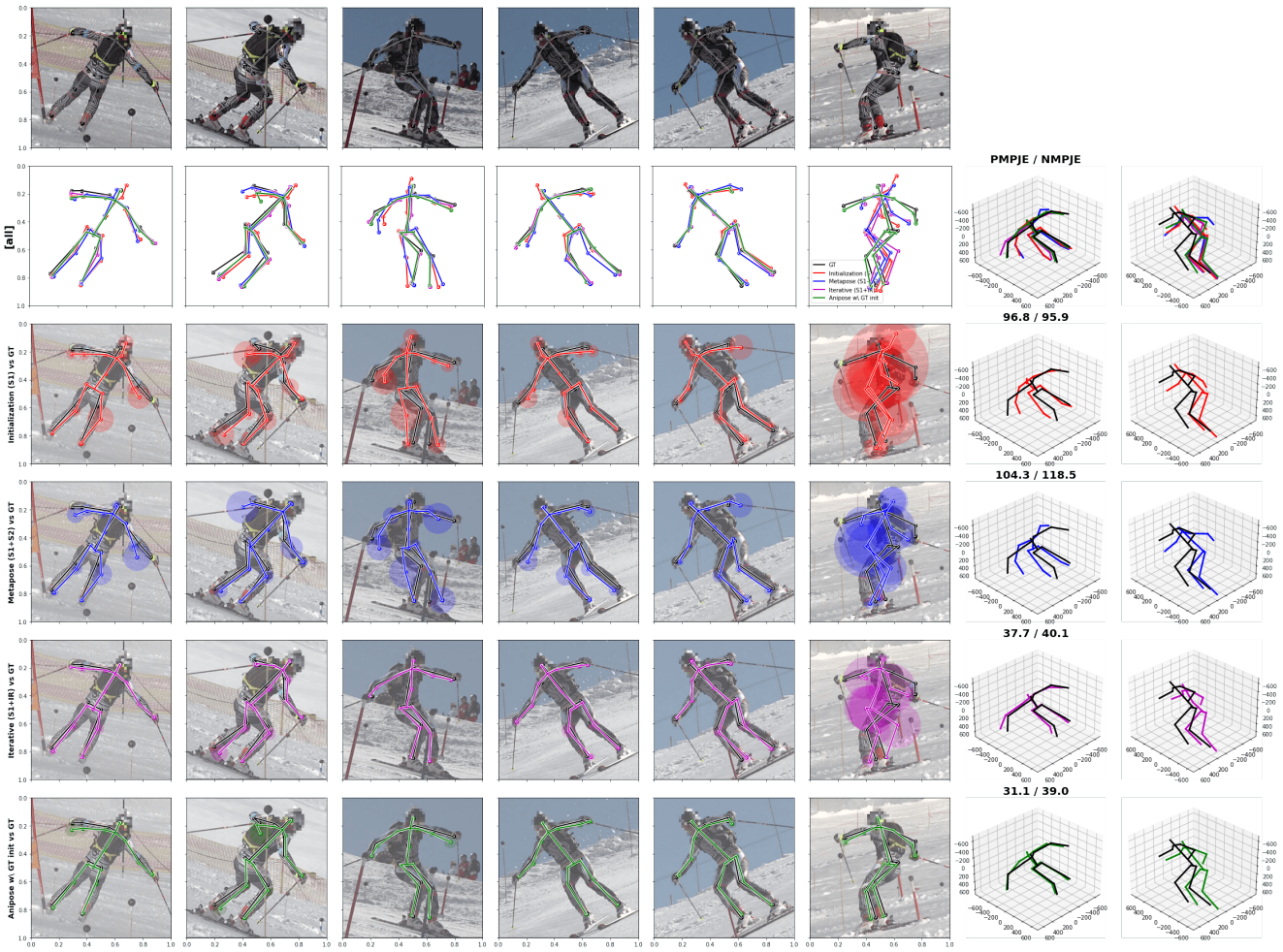

Figure 16. MetaPose fails on poses that have much poorer (than average) initialization quality.

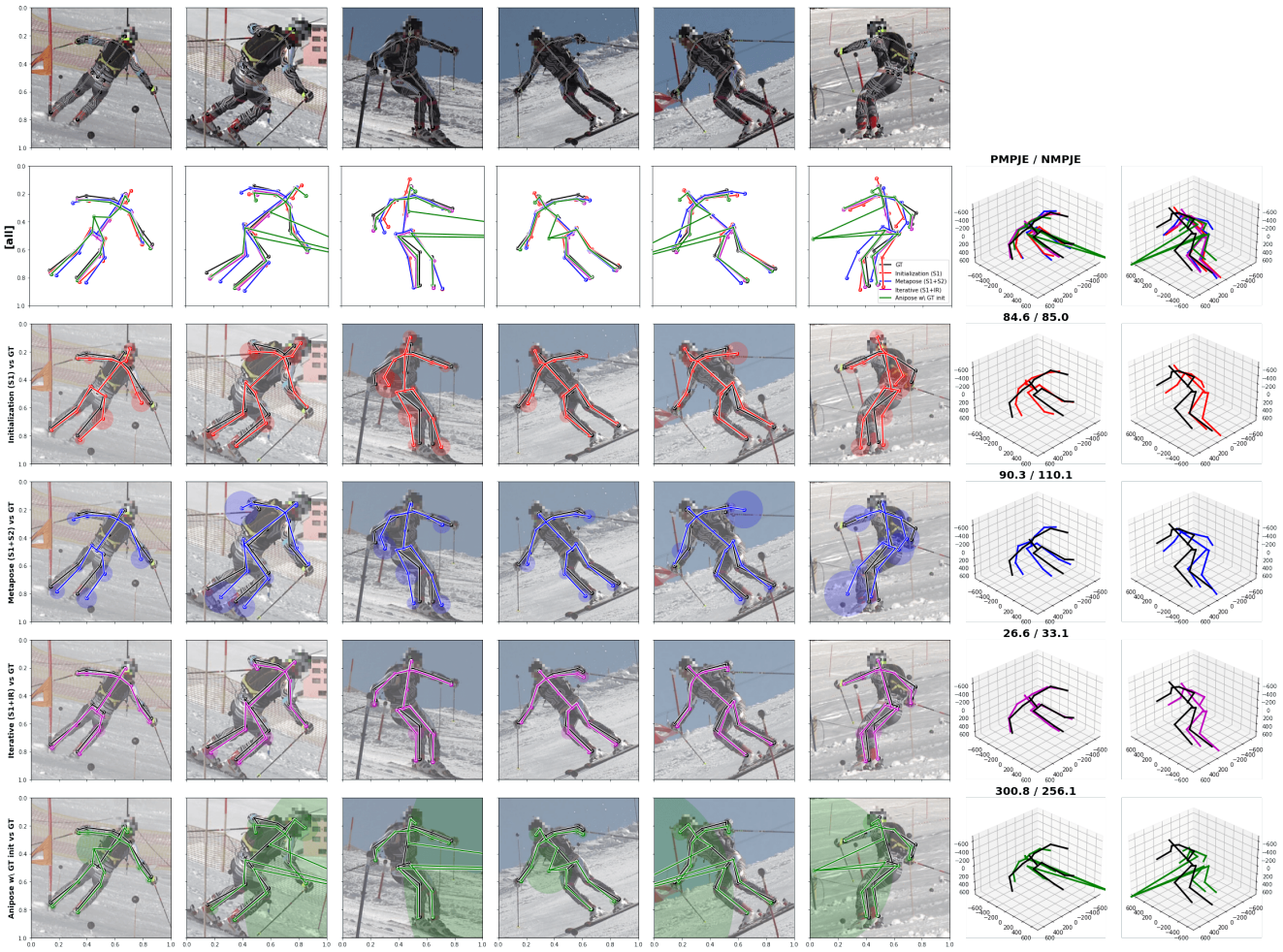

Figure 17. AniPose with GT init fails due to poor choice of 2D predictions to ignore during refinement.

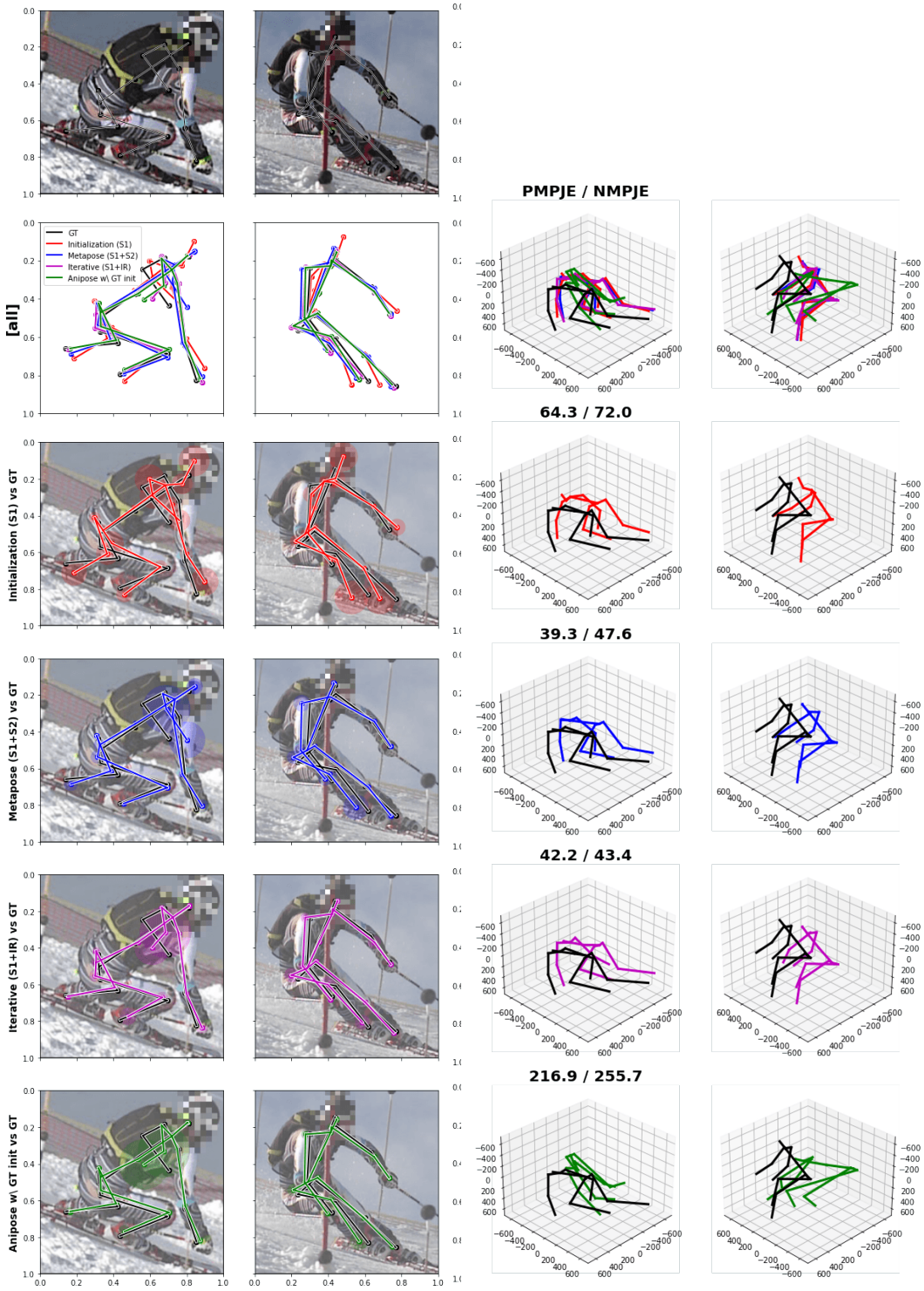

Figure 18. With **two cameras** AniPose with GT camera init often yields low reprojection error but bad 3D estimation error

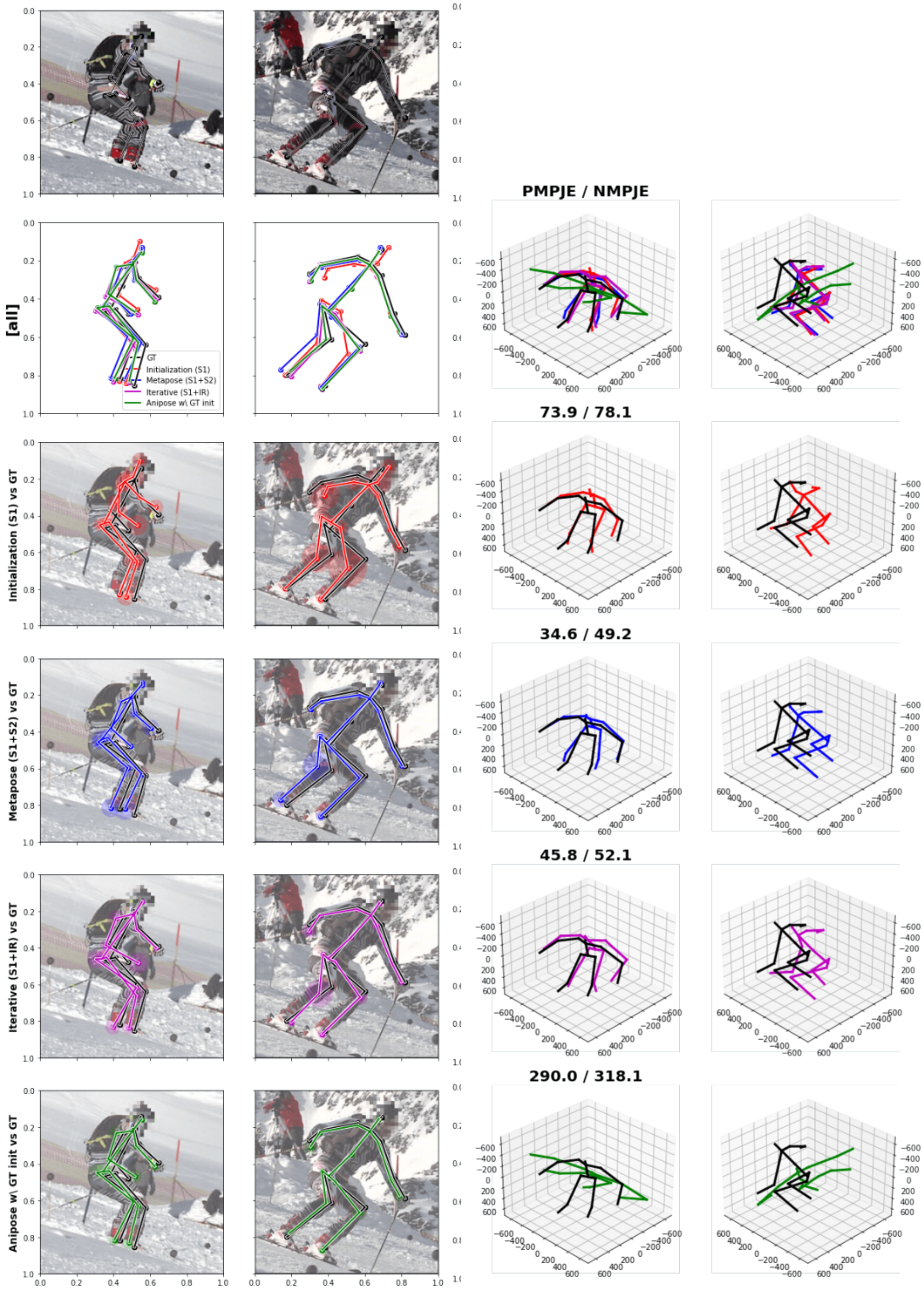

Figure 19. With **two cameras** AniPose with GT camera init often yields low reprojection error but bad 3D estimation error