Pose2Room: Understanding 3D Scenes from Human Activities

Yinyu Nie[1], Angela Dai[1], Xiaoguang Han[2], and Matthias Nießner[1]

[1] Technical University of Munich
[2] The Chinese University of Hong Kong, Shenzhen

In this supplementary material, we describe network parameters and specifications (Sec. A), details of our data generation and distribution (Sec. B), evaluation metrics in our experiments (Sec. C), additional quantitative results (Sec. D), additional qualitative results (Sec. E), tolerance to noise (Sec. F), and qualitative results on other real data (Sec G). Our code and data are also attached with this file.

A Network Specifications

We detail the full list of network parameters and specifications in this section. MLP layers used in our network are uniformly denoted by MLP$[l_1, l_2, ..., l_d]$, where l_i is the neuron number in the i-th layer. Each layer is followed by a batch normalization and a ReLU layer except the final one. We also report the efficiency and memory usage during inference at the end. Our code will be publicly available.

A.1 Skeleton Configuration

In P2R-Net, the input is a pose trajectory with N frames and J joints as the sequence of 3D locations $\boldsymbol{T} \in \mathbb{R}^{N\times J\times 3}$, where $N = 768$, $J = 53$. For each trajectory, the humanoid agent interacts with up to 10 different objects in a scene, with the frame number varying among sequences (depending on the object interactions). To enable mini-batch training, we uniformly sample N frames per sequence for training.

For the human skeleton structure, we use the predefined human body model in VirtualHome [9], which uses the Unity3D human body template. We refer readers to [1] for the detailed definition of the skeleton specifications. The root joint $\boldsymbol{r} \in \mathbb{R}^{N\times 3}$ that we use is the centroid of the hips, as illustrated in Figure 1.

A.2 Relative Position Encoder

We list the layer details of our relative position encoder in Figure 2. In Section 3.1, the output pose feature dimension $d_1 = 64$, the number of temporal neighbors $k = 20$. From the input pose trajectory $\boldsymbol{T}$, it outputs the relative pose features $\boldsymbol{P}^r \in \mathbb{R}^{N\times J\times 64}$ for spatio-temporal encoding.

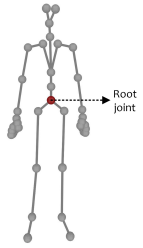


Fig. 1: Human skeleton configuration, following VirtualHome [9].

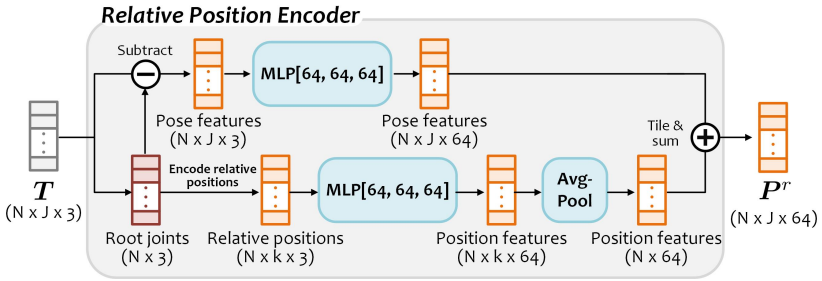


Fig. 2: Relative position encoder.

A.3 Spatio-Temporal Pose Encoder

Figure 3 illustrates the layer details of our spatio-temporal pose encoder in Section 3.2. From the input pose feature $\boldsymbol{P}^r$, we use a graph convolutional layer to learn intra-skeleton joint features. Edges in the graph convolution are constructed following the skeleton bones (as in Figure 1), which encodes skeleton-wise spatial information. We use the method of [13] to build the edge connections from skeleton bones. With the learned joint features on each skeleton, we then adopt a 1-D convolutional layer (feature dimension at 64, kernel size at 3, padding size at 1) to process joint features in temporal domain. The kernel size presents its receptive field on neighboring frames. We connect a graph layer and a 1-D convolutional layer into a block with a residual summation from the input (see Figure 3). We duplicate the block six times and stack them in a sequence to construct the spatio-temporal pose encoder. After the spatio-temporal layers, we then flatten all joint features in a skeleton, which results in a $64J$-dimensional feature per pose, followed by an MLP[256] to process each pose and produce the final spatio-temporal pose features $\boldsymbol{P}^{st} \in \mathbb{R}^{N \times d_2}$. $d_2 = 256$.

A.4 Locality-Sensitive Voting

In Section 3.3 of the main paper, we sample M seeds $\boldsymbol{r}_s$ from root joints $\boldsymbol{r}$, where $M = 512$. The M seeds are uniformly sampled along the trajectory of root joints to ensure a even spatial distribution. $\boldsymbol{P}_s^{st}$ are the corresponding pose features

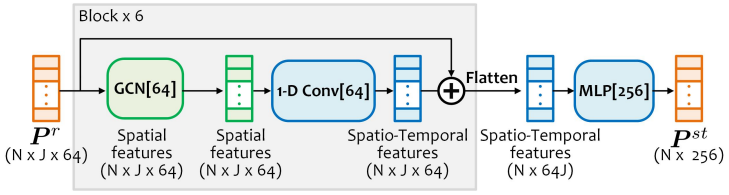


Fig. 3: Spatio-temporal pose encoder.

of $\boldsymbol{r}_s$, where $\boldsymbol{r}_s \in \mathbb{R}^{M\times 3}$, $\boldsymbol{P}_s^{st} \in \mathbb{R}^{M\times d_2}$, $d_2 = 256$. In Eq. 2 of the paper, we use two MLPs, f_3 and f_4, to learn the vote locations and features $(\boldsymbol{v}, \boldsymbol{P}^v)$ from $(\boldsymbol{r}_s, \boldsymbol{P}_s^{st})$, where f_3 and f_4 share the first two MLP layers and correspondingly predict their targets from the last layer. We illustrate them in Figure 4.

From the 512 votes $(\boldsymbol{v}, \boldsymbol{P}^v)$, we group them into V clusters following [10], which results in $(\boldsymbol{v}^c, \boldsymbol{P}^c)$ cluster centers and features, where $\boldsymbol{v}^c \in \mathbb{R}^{V\times 3}$, $\boldsymbol{P}^c \in \mathbb{R}^{V\times 256}$, V=128. For poses whose root joint is not close to any object during training (beyond a distance threshold t_d=1 m), we do not consider them to vote for any object.

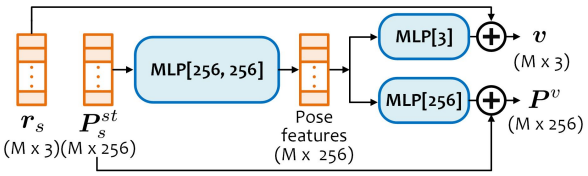


Fig. 4: Locality-sensitive voting.

A.5 Probabilistic Mixture Decoder

In Section 3.4 of the main paper, we learn multiple Gaussian distributions $\mathcal{N}(\mu_\tau^k, \Sigma_\tau^k)$ for each regression target $\tau \in \{c, s, \theta\}$, k=1,...,P, where c, s, θ respectively denote the box center, size and orientation; P is the number of distributions (P=100). For each distribution, we also learn a mode score $f_\tau^k(*) \in [0, 1]$ as its weight in predicting the target (see Eq.3 in the paper). $v^c \in \boldsymbol{v}^c$, $P^c \in \boldsymbol{P}^c$ represent a cluster center and feature, from which a proposal box is predicted.

The learnable parameters are $\{(\mu_\tau, \Sigma_\tau)\}$ and $\{f_\tau\}$, where $\mu_\tau \in \mathbb{R}^{P\times d_\tau}$; $\Sigma_\tau \in \mathbb{R}^{P\times d_\tau\times d_\tau}$. For each target $\tau \in \{c, s, \theta\}$, μ_τ represents the mean values of the P Gaussian distributions, and Σ_τ stores the corresponding covariance matrices. $\mu_\tau^k \in \mathbb{R}^{d_\tau}$ and $\Sigma_\tau^k \in \mathbb{R}^{d_\tau\times d_\tau}$ are the k-th item in μ_τ and Σ_τ respectively. d_τ is the dimension for each target, where d_c=3; d_s=3; d_θ=2. Here we consider variables in each Gaussian are independently distributed, resulting in a diagonal covariance matrix. We formulate $\{(\mu_\tau, \Sigma_\tau)\}$ as learnable embeddings shared among all

samples in training and testing. Since covariance matrix in Σ_τ is diagonal and non-negative, we use a exponential function to process it diagonal elements.

f_τ is realized with an MLP layer to predict mode scores for each target, where we use MLP[128,128,128,100] appended with a sigmoid layer.

For the proposal objectness and probability distribution of class category, we predict them directly from P^c with MLP[128,128,d_{obj}+d_{cls}], where d_{obj}=2 and d_{cls}=17 (the number of object categories), followed by a softmax layer to get their probability scores in [0, 1].

To generate a hypothesis for each object during inference, we sample box parameters with Eq. 4 in our paper. Each hypothesis is an average of N_s samples of y_τ. N_s is also a random number in $[1, 100]$. Additionally, we sample the object class based on the predicted classification probability distribution.

We discard proposed object boxes which have low objectness scores ($\leq t_o$) after a 3D non-maximum suppression ($t_o = 0.5$), which then outputs N_h hypotheses in a scene.

A.6 Efficiency and Memory in Inference

We train our network with batch size of 32 with 4 NVIDIA RTX 2080 GPUs using PyTorch 1.7.1, and test it with a single GPU. We report the model size, inference timing and allocated GPU memory in a single forward pass.

Model size	Avg. time	Peak. time	Avg. memory	Peak memory
2.04 M	0.092 s	0.582 s	11.64 MB	260.82 MB

Table 1: Model size, efficiency and memory usage of P2R-Net.

A.7 Specifications of Baselines

We explain the network details of each baseline as the following.

Pose-VoteNet Pose-VoteNet is a variant of VoteNet [10] to make it able to vote for object centers from human poses. In our experiments, we replace the PointNet++ in original VoteNet with our position encoder, which produces relative pose feature $\boldsymbol{P}^r \in \mathbb{R}^{N\times J\times 64}$. We then flatten all joint features for each pose ($\mathbb{R}^{N\times 64J}$) and learn the seed feature with MLP[256, 256, 256]. The coordinates of each seed is located at the root joint, similar with ours. For the remaining structures and loss functions, we keep them consistent with VoteNet.

Pose-VN To make Pose-VoteNet able to capture rotation information of poses, we augment it by replacing MLP layers in Pose-VoteNet encoder with vector neurons [4]. Vector neurons are a set of SO(3)-equivariant operators that capture arbitrary rotations of object poses to estimate objects. For each MLP layer in

Pose-VoteNet encoder, we replace it with a 'VNLinearLeakyReLU' layer, and the final MLP layer with a 'VNLinear' layer, with equal number of parameters. For the details of layer design in vector neurons, we refer readers to [4].

Motion Attention We replace the MLP layers (i.e., MLP[256, 256, 256]) in Pose-VoteNet encoder with the attention module in [7] to learn inter-frame pose features in the entire temporal domain. Specifically, for each pose feature in $\boldsymbol{P}^r$, we use it to query similar features among all frames, which assembles repetitive pose patterns to regress object boxes. For the layer details in motion attention, we refer readers to [7].

P2R-Net-D We replace our probabilistic decoder with the VoteNet decoder [10] along with their loss functions to produce deterministic results.

P2R-Net-G We ablate the P2R-Net decoder with a probabilistic generative model [11,8], where we first lean a latent code $\boldsymbol{z} \sim \mathcal{N}(\mathbf{0}, \mathbf{1})$ from cluster features $\boldsymbol{P}^c$. By decoding from the summation of $\boldsymbol{z}$ and $P^c \in \boldsymbol{P}^c$, we can predict box parameters in a probabilistic generative way.

P2R-Net-H We discretize each regression target into a binary heatmap, where box centers are discretized into 10^3 bins in [-0.3 m, 0.3 m]^3, centered at cluster centers $\boldsymbol{v}^c$; box sizes are discretized into 10^3 bins in [0.05 m, 3m]^3; box orientations are discretized into 12 bins in $[-\pi, \pi]$. Then the box regression is converted into a classification task. We train them by cross-entropy losses. In testing, we sample the heatmaps to produce different predictions.

B Data Generation and Data Statistics

We create our dataset using the VirtualHome simulation environment [9], which is built on the Unity3D game engine. It consists of 29 rooms, where each room has 88 objects on average. Each object is annotated with available interaction types. For the detailed specification of interaction manners for different object categories, we refer to [9,2].

VirtualHome allows users to customize action scripts to direct humanoid agents to execute a series of complex interactive tasks. In our work, we focus on the static, interactable objects under 17 common class categories (i.e., bed, bench, bookshelf, cabinet, chair, desk, dishwasher, faucet, fridge, lamp, microwave, monitor, nightstand, sofa, stove, toilet, computer).

In each room, we select up to 10 random objects in the scene, and script the agent to interact with each of the objects in a sequential fashion. For each object, we also select a random interaction type associated with the object class category. Sequences are trained with and evaluated against only the objects that are interacted with during the input observation, resulting in different variants of each room under different interaction sequences.

Then we randomly sample 13,913 different sequences with corresponding object boxes to construct the dataset. The human pose trajectories are recorded

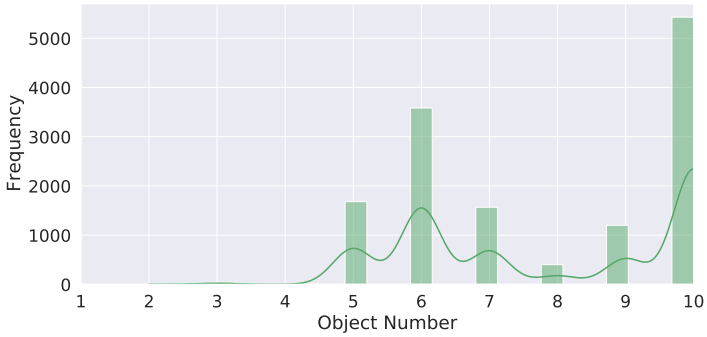


Fig. 5: Distribution over number of objects in a pose trajectory.

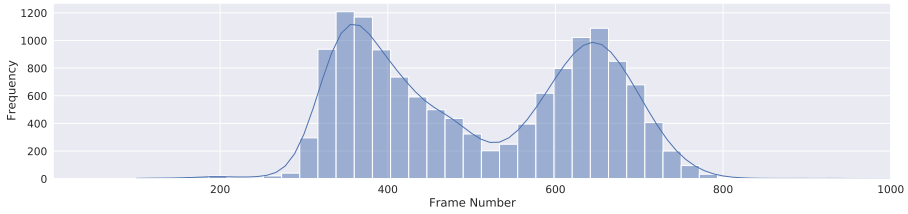


Fig. 6: Distribution of frame lengths in pose trajectories.

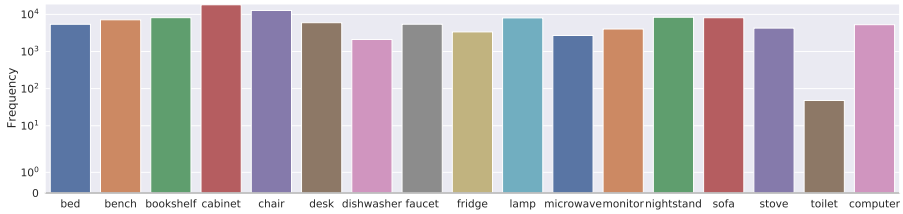


Fig. 7: Frequency of object class categories among generated interactions.

with a frame rate of 5 frames per second. Over the sequences, the average number of objects is 7.86, and the average frame length is 509.34. The distributions of the frame length and object number in a interaction trajectory are shown in Figure 6 and Figure 5. The interaction frequency for each object class category is illustrated in Figure 7, and we also list the frequency of each interaction type in Figure 8.

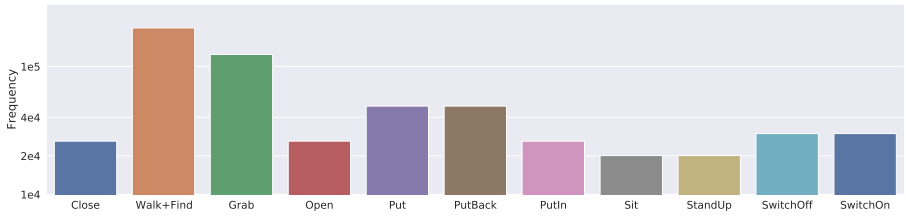


Fig. 8: Frequency of interaction types.

C Evaluation Metrics

In our method, we use mAP@0.5 to evaluate the detection accuracy by comparing our maximum likelihood prediction with the ground-truth. We also use Minimal Matching Distance (MMD) and Total Mutual Diversity (TMD) to respectively evaluate the quality and diversity of our multi-modal predictions. For each input sequence, we sample our probabilistic mixture decoder and produce 10 hypotheses of object arrangements. We adapt MMD from [3] to evaluate our task, and measure the best matching mAP@0.5 score with the ground-truth out of the 10 hypotheses. For TMD, we follow [12] to evaluate the multi-modality of our predictions, formulated as

$$
\begin{aligned}
\mathrm{TMD}(O^i) &= [1 + \mathrm{Entropy}(\mathcal{C}(O^i))] \cdot [1 + \mathrm{Div}(\mathcal{B}(O^i))], \\
\mathrm{Div}(\mathcal{B}(O^i)) &= \frac{1}{10} \sum_{p=1}^{10} \sum_{q=1}^{10} \mathrm{Dist}(B_p^i, B_q^i);, \\
\mathrm{Dist}(B_p^i, B_q^i) &= \frac{1}{8} \sum_{j=1}^{8} ||P - Q||_2, \qquad P \in B_p^i, Q \in B_q^i, \\
\mathcal{C}(O^i) &= \{c_1^i, c_2^i, ..., c_{10}^i\}, \\
\mathcal{B}(O^i) &= \{B_1^i, B_2^i, ..., B_{10}^i\}, \qquad B_k^i \in \mathbb{R}^{8\times 3}.
\end{aligned}
\tag{1}
$$

In Eq. 1, TMD is defined at the object-level; for each object O^i in a scene, we have 10 hypotheses which are represented by 10 bounding boxes $\mathcal{B}(O^i)$ with the corresponding 10 predicted class labels $\mathcal{C}(O^i)$; B_k^i is the k-th hypothesis in $\mathcal{B}(O^i)$, which can be represented by a point set with eight box corners, and c_k^i is the corresponding class label; Entropy$(*)$ denotes the Shannon Entropy to measure the variance of class labels; Div$(*)$ measures the diversity of predicted bounding boxes among hypotheses, which is defined by the average of distance sum from a hypothesis B_p^i to all other hypotheses; Dist$(*)$ is the average Euclidean distance between pair-wise points from B_p^i and B_q^i.

TMD(O^i)=1 if all hypotheses are the same, where $\mathcal{C}(O^i))$=0; Div$(\mathcal{B}(O^i)) = 0$, which indicates no diversity. In Section 5.3 of the main paper, we report the average TMD score over all objects.

	bed	bench	bkshlf	cabnt	chair	desk	dishws	faucet	fridge	lamp	microw	monitor	nstand	sofa	stove	toilet	cmpter	mAP@0.5
Pose-VoteNet	2.90	15.00	73.79	33.14	18.77	58.52	32.14	0.00	0.00	6.07	6.49	23.55	51.30	62.32	49.82	0.00	3.06	25.70
Pose-VoteNet + VN	20.81	18.13	78.25	49.76	18.68	70.92	33.56	0.00	0.00	5.60	5.23	22.71	64.46	67.24	46.76	0.00	6.11	29.90
Motion Attention	36.42	7.54	66.25	23.35	19.50	77.71	15.59	0.03	17.13	2.35	6.42	29.87	30.59	78.61	51.03	14.81	5.50	28.39
P2R-Net-D	93.77	12.63	37.21	11.98	5.77	95.93	61.80	0.10	73.95	0.58	7.39	9.68	9.62	88.44	70.42	0.00	14.17	34.91
P2R-Net-G	91.69	7.56	41.27	36.61	10.05	93.47	67.53	0.10	77.45	1.21	6.36	10.51	11.32	92.97	64.86	5.56	18.67	37.48
P2R-Net-H	85.84	8.04	43.78	22.04	10.91	76.08	55.20	0.00	55.15	0.00	1.57	4.24	20.31	83.92	57.60	5.00	4.33	31.41
Ours	94.21	10.12	41.75	54.72	8.02	93.32	56.33	0.06	59.89	3.25	6.49	12.84	46.53	90.92	57.86	61.11	19.94	42.20

Table 2: Quantitative evaluation on split $\mathcal{S}_1$.

	bed	bench	bookshelf	cabinet	chair	desk	dishwasher	faucet	fridge	microwave	nightstand	stove	mAP@0.5
Pose-VoteNet	0.00	4.48	22.84	2.30	0.56	4.05	14.64	0.00	0.00	0.61	0.00	73.33	10.23
Pose-VN	0.00	0.96	19.88	7.99	0.51	0.14	10.61	0.00	0.00	0.00	0.01	33.33	6.12
Motion Attention	28.16	1.81	24.19	1.43	0.00	0.00	0.00	0.00	0.00	0.00	22.77	0.00	6.53
P2R-Net-D	39.86	25.17	26.78	3.29	0.18	56.53	76.39	0.00	86.50	30.75	0.00	33.33	31.56
P2R-Net-G	50.65	17.89	13.86	0.54	0.04	56.97	87.61	0.06	70.91	13.76	0.00	66.67	31.59
P2R-Net-H	48.22	10.43	39.12	15.12	0.02	41.26	71.18	0.00	76.10	0.69	0.01	33.33	27.96
P2R-Net	81.48	12.05	17.18	6.43	0.10	72.07	100.00	0.11	63.39	4.61	0.04	66.66	35.34

Table 3: Quantitative evaluation on split $\mathcal{S}_2$.

D Additional Quantitative Results

We list the mAP@0.5 scores on all object categories by split $\mathcal{S}_1$ and $\mathcal{S}_2$ in Table 2 and Table 3 respectively. P2R-Net variants are evaluated by the maximum likelihood predictions to calculate mAP scores. Note that there are fewer categories in the test set of $\mathcal{S}_2$.

E Additional Qualitative Results

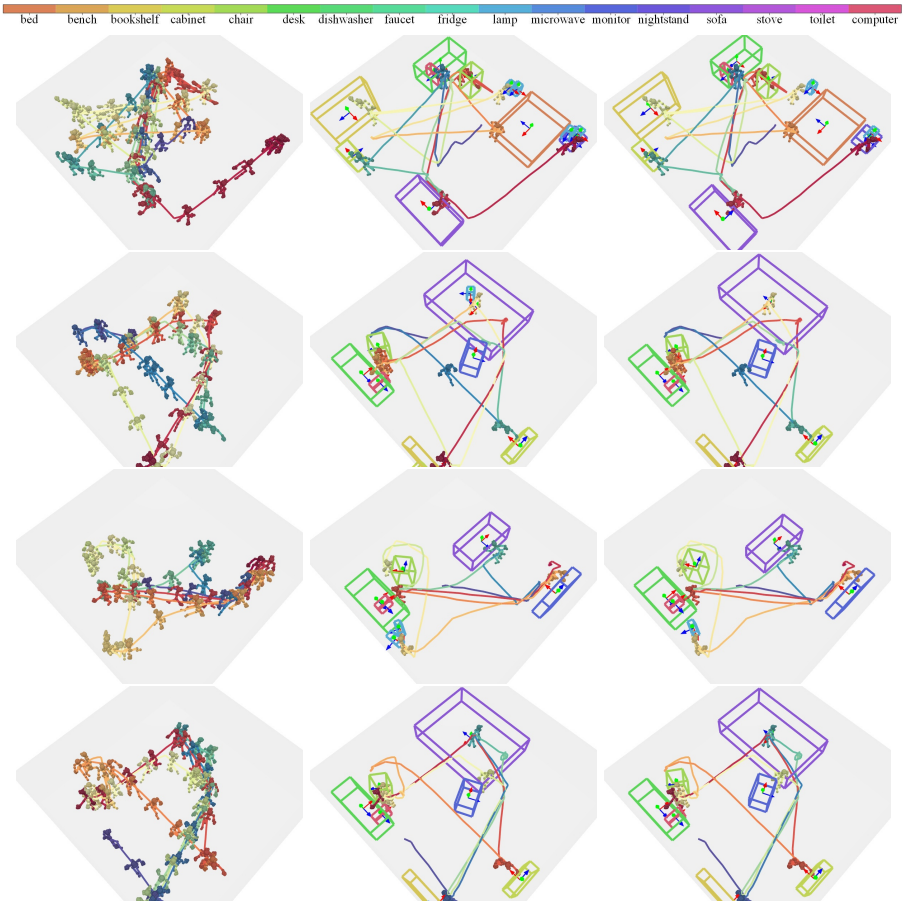
We show additional qualitative results on test splits $\mathcal{S}_1$ and $\mathcal{S}_2$ in Figures 9-11 and 12, respectively. We additionally visualize various multi-modal predictions from our model on $\mathcal{S}_1$ in Figure 13.

F Tolerance to noise

As real data often contain noise, we additionally study the effect of Gaussian noise (std=5 cm) onto the xyz coordinates of all joints in training and testing with our dataset. Table 4 shows the effect of different noise levels in evaluation. We also visualize some sampled predictions under the noise level at 10σ in Figure 14, where our method presents compelling tolerance for very noisy inputs.

Noise level	σ	2σ	3σ	5σ	10σ
$\mathcal{S}_1$	38.58	38.77	38.36	37.16	31.71
$\mathcal{S}_2$	27.16	26.13	27.72	29.18	26.18

Table 4: mAP@0.5 under varying levels of noise (σ=1 cm).

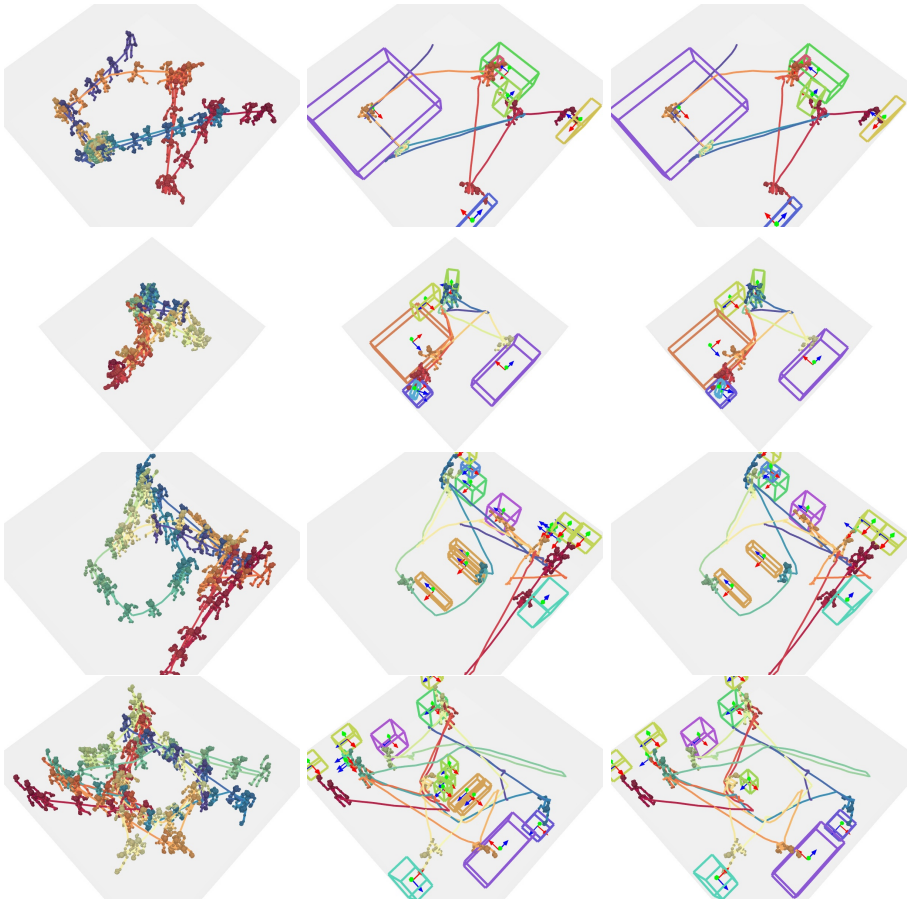


(a) Input (b) Prediction (c) GT

Fig. 9: Additional results on estimating object layouts from a pose trajectory on the sequence-level split $\mathcal{S}_1$ (unseen interaction sequences).

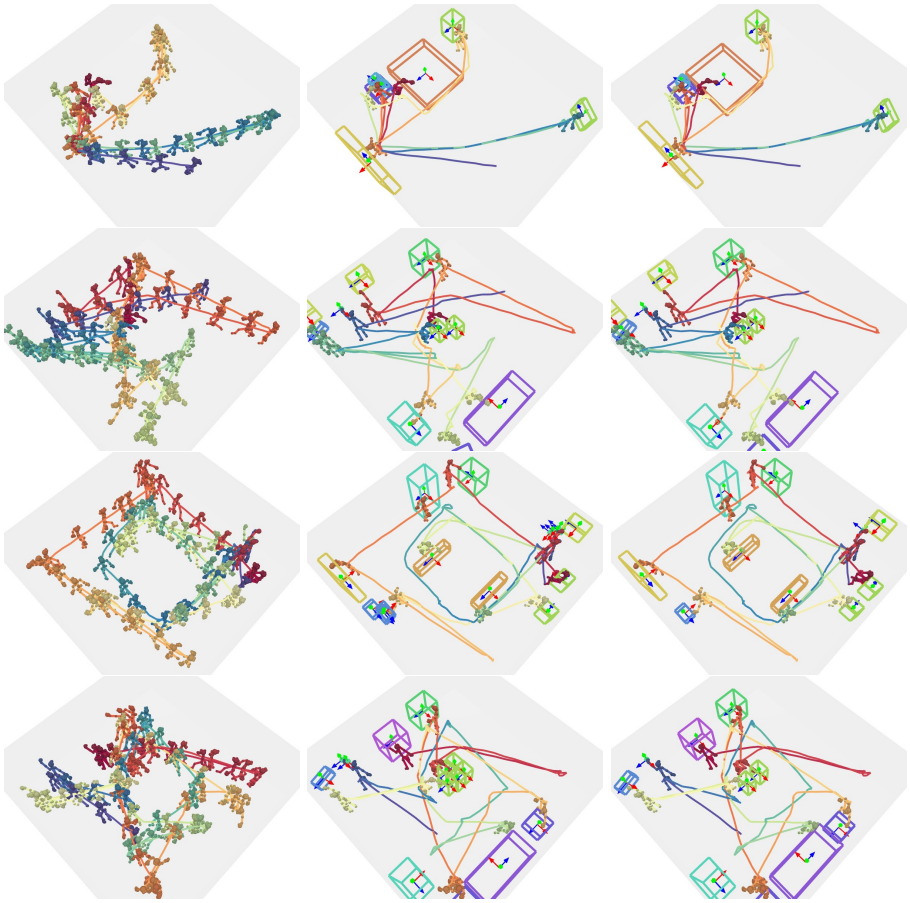
G Qualitative Results on Other Real Data

Besides the experiments on VirtualHome [9] and PROX [6,14], we additionally qualitatively evaluate P2R-Net by training it on our dataset, and apply it to the real-world human pose trajectory data provided by [5] describing human interactions with single objects. The qualitative results are illustrated in Figure 15, where we see that our method still can provide plausible object explanations from natural and diverse real human poses.



(a) Input (b) Prediction (c) GT

Fig. 10: Additional results on estimating object layouts from a pose trajectory on the sequence-level split $\mathcal{S}_1$ (unseen interaction sequences).

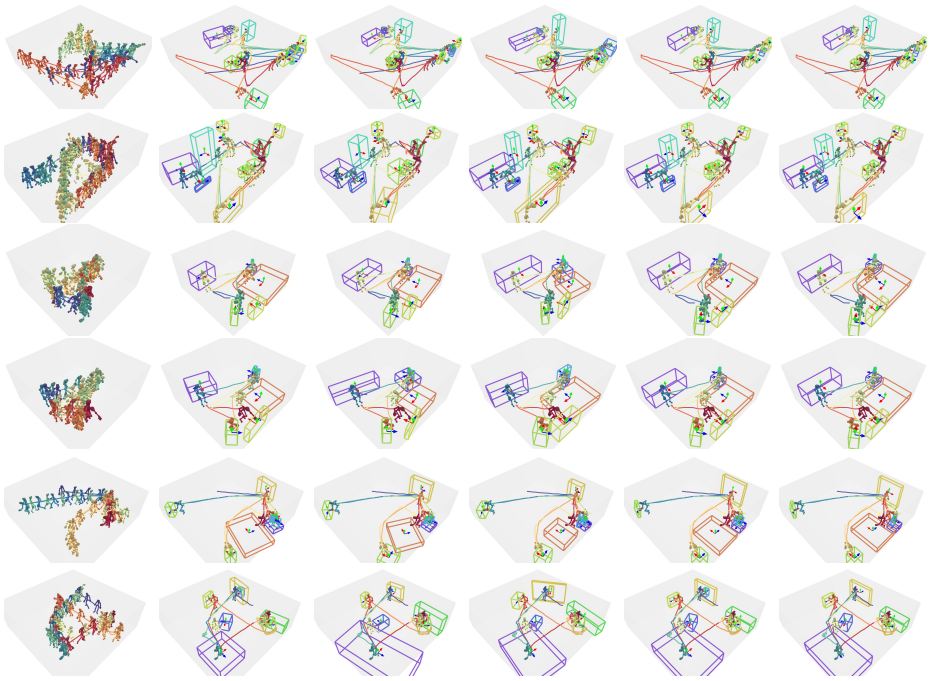


(a) Input (b) Prediction (c) GT

Fig. 11: Additional results on estimating object layouts from a pose trajectory on the sequence-level split $\mathcal{S}_1$ (unseen interaction sequences).

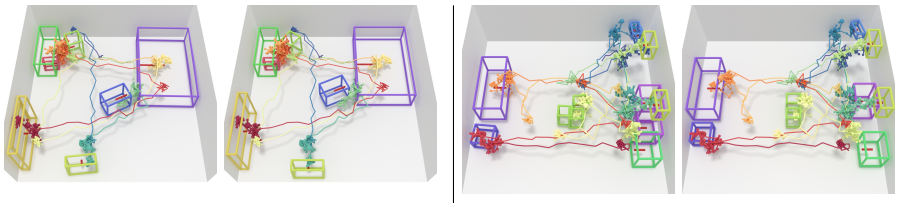
(a) Input (b) Prediction (c) GT

Fig. 12: Additional results on estimating object layouts from a pose trajectory on the room-level split $\mathcal{S}_2$ (unseen interaction sequences and rooms).



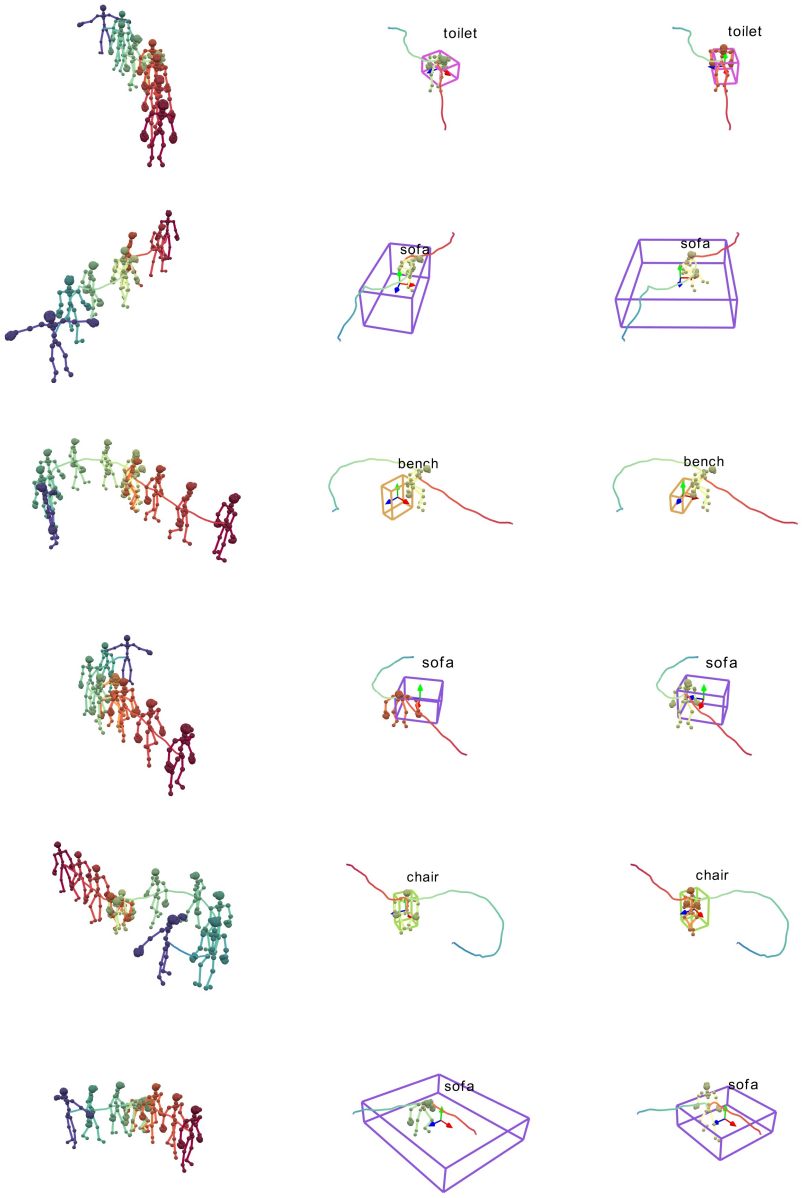
(a) Input (b) Sample 1 (c) Sample 2 (d) Sample 3 (e) Max. likelihood prediction (f) GT

Fig. 13: Additional multi-modal predictions of P2R-Net. By sampling our probabilistic decoder multiple times, we can obtain various different plausible box predictions.



(a) Prediction (b) GT (c) Prediction (d) GT

Fig. 14: Predictions on noisy inputs (std=10 cm).



(a) Input (b) Multi-modal sample (c) Max. likelihood prediction

Fig. 15: Multi-modal predictions on the real human pose trajectory input of [5].

References

1. Unity documentation: Humanbodybones. https://docs.unity3d.com/ScriptReference/HumanBodyBones.html (03 2020) 1
2. Virtualhome homepage. http://virtual-home.org/ (11 2021) 5
3. Achlioptas, P., Diamanti, O., Mitliagkas, I., Guibas, L.: Learning representations and generative models for 3d point clouds. In: International conference on machine learning. pp. 40–49. PMLR (2018) 7
4. Deng, C., Litany, O., Duan, Y., Poulenard, A., Tagliasacchi, A., Guibas, L.: Vector neurons: a general framework for so(3)-equivariant networks. arXiv preprint arXiv:2104.12229 (2021) 4, 5
5. Hassan, M., Ceylan, D., Villegas, R., Saito, J., Yang, J., Zhou, Y., Black, M.J.: Stochastic scene-aware motion prediction. In: Proceedings of the IEEE/CVF International Conference on Computer Vision. pp. 11374–11384 (2021) 9, 14
6. Hassan, M., Choutas, V., Tzionas, D., Black, M.J.: Resolving 3d human pose ambiguities with 3d scene constraints. In: Proceedings of the IEEE/CVF International Conference on Computer Vision. pp. 2282–2292 (2019) 9
7. Mao, W., Liu, M., Salzmann, M.: History repeats itself: Human motion prediction via motion attention. In: European Conference on Computer Vision. pp. 474–489. Springer (2020) 5
8. Nie, Y., Hou, J., Han, X., Nießner, M.: Rfd-net: Point scene understanding by semantic instance reconstruction. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 4608–4618 (2021) 5
9. Puig, X., Ra, K., Boben, M., Li, J., Wang, T., Fidler, S., Torralba, A.: Virtualhome: Simulating household activities via programs. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. pp. 8494–8502 (2018) 1, 2, 5, 9
10. Qi, C.R., Litany, O., He, K., Guibas, L.J.: Deep hough voting for 3d object detection in point clouds. In: Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV) (October 2019) 3, 4, 5
11. Wu, J., Zhang, C., Xue, T., Freeman, W.T., Tenenbaum, J.B.: Learning a probabilistic latent space of object shapes via 3d generative-adversarial modeling. In: Proceedings of the 30th International Conference on Neural Information Processing Systems. pp. 82–90 (2016) 5
12. Wu, R., Chen, X., Zhuang, Y., Chen, B.: Multimodal shape completion via conditional generative adversarial networks. In: Computer Vision–ECCV 2020: 16th European Conference, Glasgow, UK, August 23–28, 2020, Proceedings, Part IV 16. pp. 281–296. Springer (2020) 7
13. Yan, S., Xiong, Y., Lin, D.: Spatial temporal graph convolutional networks for skeleton-based action recognition. In: Thirty-second AAAI conference on artificial intelligence (2018) 2
14. Yi, H., Huang, C.H.P., Tzionas, D., Kocabas, M., Hassan, M., Tang, S., Thies, J., Black, M.J.: Human-aware object placement for visual environment reconstruction. In: Computer Vision and Pattern Recognition (CVPR) (2022) 9