Embodied Question Answering in Photorealistic Environments with Point Cloud Perception Supplemental Material

Erik Wijmans[1†], Samyak Datta[1†], Oleksandr Maksymets[2†], Abhishek Das[1], Georgia Gkioxari[2], Stefan Lee[1], Irfan Essa[1], Devi Parikh[1,2], Dhruv Batra[1,2]
[1]Georgia Institute of Technology [2]Facebook AI Research
[1]{etw, samyak, abhshkdz, steflee, irfan, parikh, dbatra}@gatech.edu
[2]{maksymets, gkioxari}@fb.com

1. Color Label Collection Interface

Fig. 1 shows the interface we used to collect dominate color annotations from workers on Amazon Mechanical Turk.

2. Point Cloud Rendering

In order to render 2.5D RGB-D frames, we first construct a global point cloud from all of the panoramas provided in an environment in the Matterport3D dataset [1]. Next, we determine what parts of the global point cloud are in the agent's current view. The agent's current position, A_{pos}, camera parameters (field of view, and aspect ratio), and the mesh reconstruction are used to determine which points are within its view pyramid (frustum).

For each point p_i in view, we first check if the point lies within the agents view frustum using the extrinsic and intrinsic camera matrices. After determining which points lie within the view frustum, we check for occlusions. We then draw a line between the agent's camera and p_i, let L_i be that line. We intersect this line with the provided mesh and keep the intersection that is *closest* to the agent, let $\text{argmin}(L_i \cap \mathcal{M})$ be that intersection. If the distance to the *closest* intersection with the mesh, $||A_{pos} - \text{argmin}(L_i \cap \mathcal{M})||$, is less than the distance to the point, $||A_{pos} - p_i||$, indicating that there is a closer (occluding) point, we remove the point. We also perform this check in the other direction and remove the point if $||A_{pos} - p_i|| < ||A_{pos} - \text{argmin}(L_i \cap \mathcal{M})||$, indicating that the point is in the free-space. Points in the free-space are a result of panorama alignment errors and scanning oddities from reflections. In practice, we find that equality is too strict of a criteria due to mesh reconstruction errors, we instead remove a point if the absolute difference of the distances is greater than $\epsilon = 0.25$ cm, $\text{abs}(||A_{pos} - \text{argmin}(L_i \cap \mathcal{M})|| - ||A_{pos} - p_i||) > \epsilon$.

The implementation implied by the description above would be very slow. What we have described above implies something akin to a ray-tracing rendering. As in normal graphics pipelines, we can significantly speed this up by approximation with rasterization. We rasterize the provided mesh at A_{pos} and capture the depth buffer. The depth buffer provides the distance from the agent to the *closest* intersection with the mesh for some finite number of rays. Let R_j be a ray in that set and $||A_{pos} - \text{argmin}(R_j \cap \mathcal{M})||$ be the distance to the closest intersection with the mesh as provided by the depth buffer. We can then approximate $||A_{pos} - \text{argmin}(L_i \cap \mathcal{M})||$ for every point p_i in the point cloud by finding the R_j that is the closest to parallel to L_i and using the value of $||A_{pos} - \text{argmin}(R_j \cap \mathcal{M})||$.

Further, a single global point cloud in Matterport3D environments has hundreds of millions of points and, in theory, we would need to check every single one to determine what points are visible. This would be quite slow and use a prohibitive amount of memory. We alleviate this by creating a significantly sparser point cloud and perform an initial visibility check on that instead. We then recheck the dense point cloud only in areas of the sparse point cloud that passed the initial visibility check.

3. Perception Models

Here we provide full details on the architectures of our perceptual encoders and the decoder heads used for their pre-training tasks.

3.1. PC – PointNet++

We use notation similar to the notation from Qi *et al.* [2] to specify our PointNet++ architecture. Let $P^{(0)}$ be the

† denotes equal contribution

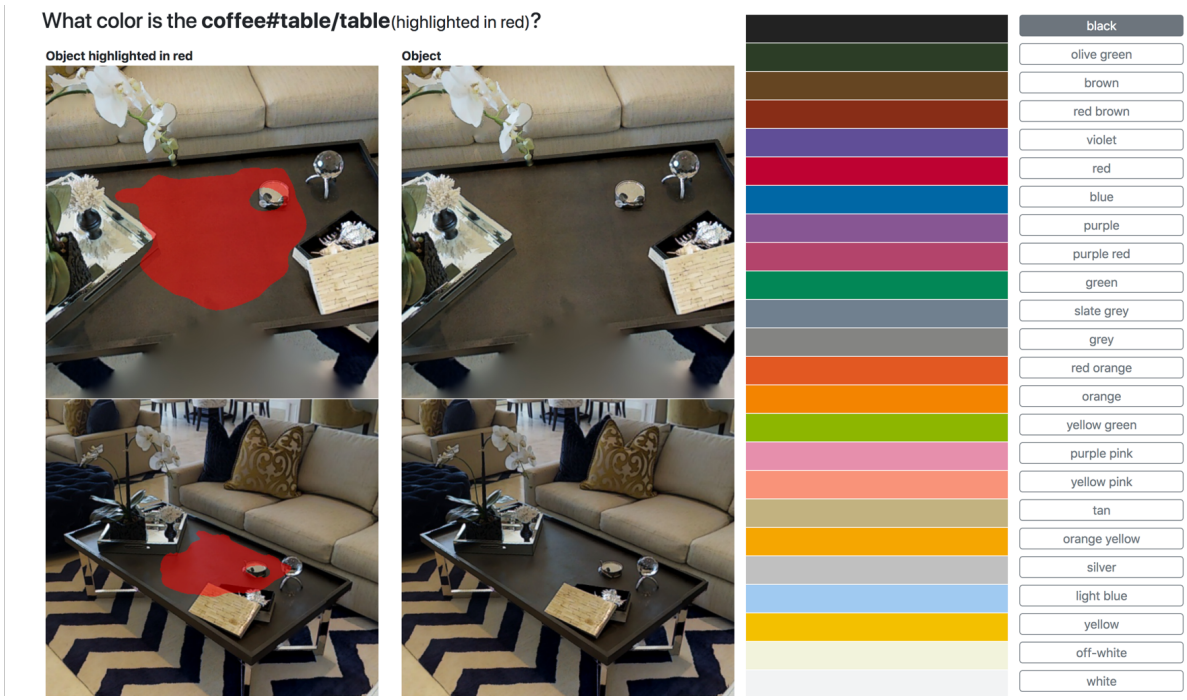

Figure 1: Interface shown to AMT workers for collecting the dominate color name for objects in the Matterport3D dataset [1]. Workers were shown up to two good views of the object (some objects only had one good view) and the corresponding instance segmentation mask and asked to selected the dominate color of the object from a predefined list of colors.

input point cloud. Then,

$$SA^{(k+1)}(P^{(k)}, N, [r^{(1)}, ..., r^{(m)}], \\ [[l_1^{(1)}, ..., l_d^{(1)}], ..., [l_1^{(m)}, ..., l_d^{(m)}]])$$

is a set abstraction module with multi-scale grouping that takes input the $k\ th$ level point cloud, $P^{(k)}$, and produces the $k+1\ th$ level point cloud, $P^{(k+1)}$, with N points. See section 4.1 in the main paper for a full description of a set abstraction module with single-scale grouping. A set abstraction module with multi-scale grouping is a logical extension. The feature descriptor for each point in $P^{(k+1)}$ is calculated across m different scales (m balls with different radii). $[l_1^{(1)}, ..., l_d^{(1)}]$ specifies the number of output channels for each layer in the shared-weighted multi-layer perceptron (MLP) used a each scale. The feature descriptor for any point is then the concatenation of the feature descriptor calculated at each scale.

As an analogy, a multi-scale convolution would be achieved by by first convolving the input with convolutions of different kernel sizes and then concatenating the outputs from each convolution.

$SA^{(k+1)}(P^{(k)}, [l_1, ..., l_d])$ is a global set abstraction module and produces a l_d dimensional feature vector.

Our encoder is specified by

$$SA^{(1)}(P^{(0)}, 1024, [0.05, 0.1], \\ [[32, 32, 64], [32, 64, 128]]) \\ SA^{(2)}(P^{(1)}, 256, [0.1, 0.2, 0.4], \\ [[64, 128, 128], [128, 128, 256], \\ [128, 128, 256, 256]]) \\ SA^{(3)}(P^{(2)}, 64, [0.4, 0.8], \\ [[128, 128, 128, 256, 256], \\ [128, 128, 256, 256, 256, 512]]) \\ SA^{(4)}(P^{(3)}, [256, 512, 1024])$$

$FP(P^{(k+1}, P^{(k)}, use\_skip, [l_1, ...l_d])$ is a feature propagation layer that transfers the features from $P^{(k+1)}$ to $P^{(k)}$ [2]. For each point in $P^{(k)}$, its three nearest neighbors in $P^{(k+1)}$ are found and their feature descriptors are combined via inverse-distance weighted interpolation. The interpolated feature descriptor is then processed by shared-weight MLP with output channels specified by $[l_1, ...l_d]$. use_skip is a $True$ or $False$ boolean that determines if any existing features of $P^{(k)}$ are used. If use_skip is true, the the interpolated feature and the existing feature are concatenated together before the shared-weight MLP. If use_skip is false, only the interpolated feature is processed by the shared-weight MLP.

The semantic segmentation head is specified as

$$\begin{aligned} Prop &= FP(P^{(4)}, P^{(3)}, True, [256, 512]) \\ Prop &= FP(Prop, P^{(2)}, True, [256, 256]) \\ Prop &= FP(Prop, P^{(1)}, True, [256, 256]) \\ Output &= FP(Prop, P^{(0)}, True, [256, 128, 128, K]) \end{aligned}$$

where K is the number of class, 40 in our case.
The color prediction head is specified as

$$\begin{aligned} Prop &= FP(P^{(4)}, P^{(3)}, True, [256, 512]) \\ Prop &= FP(Prop, P^{(2)}, True, [256, 256]) \\ Prop &= FP(Prop, P^{(1)}, False, [256, 256]) \\ Output &= FP(Prop, P^{(0)}, False, [256, 128, 128, 3]) \end{aligned}$$

The structure prediction head is specified as

$$\begin{aligned} &(a)\ FC(P^{(4)}, 256) \\ &(b)\ FC((a), 256) \\ &(c)\ FC((b), N * 3) \end{aligned}$$

$FC(in, d_{out})$ is a fully connected layer that takes in and produces a vector of size d_{out}. The output of the structure decoding head, (c), is then reshaped into a $(N, 3)$ point cloud.

Training details. We train with a batch size of 32 and utilize Adam [3]. We use an initial learning rate of 10^{-3} and decay the learning rate by 70% every 6.2×10^{-3} batches. We select the checkpoint by performance on a held-out validation set.

3.2. RGB – ResNet50

We use the initial convolution and 4 residual blocks from ResNet50 [4] as the encoder for RGB images. Each of the three decoders (semantic, depth, and color) are identical and consistent of 1x1 convolutions and bi-linear interpolation.

Let $RB2$, $RB3$, and $RB4$ be the outputs from the second, third, and fourth residual blocks in ResNet50 respectively. Each decoder is then parameterized as

$$\begin{aligned} Up4 &= 1x1Conv(RB4, C) \\ Up3 &= UpSample(Up4) + 1x1Conv(RB3, C) \\ Up2 &= UpSample(Up3) + 1x1Conv(RB2, C) \\ Output &= UpSample(Up2) \end{aligned}$$

$1x1Conv$ is simply a 1-by-1 convectional layer that transforms its input to have C channels. $UpSample$ is a bi-linear interpolation layer that up-samples its input to be the necessary size. Where C is the number of output channels for the given decoder type, K for semantic, 1 for depth, and 3 for color.

Training details. Due to the high prevalence of walls and floors among indoor scenes, we use class weighted cross-entropy loss for semantic segmentation. For depth and autoencoding, we use smooth-ℓ_1 The total loss is then the weighted sum of the individual task losses:

$$L = \lambda_{Seg.} L_{Seg.} + \lambda_{Depth} L_{Depth} + \lambda_{AE} L_{AE}$$

We find $\lambda_{Seg.} = 0.1$, $\lambda_{Depth} = 10$, and $\lambda_{AE} = 10$ balances the magnitudes of the various losses and works well.

Tab. 1 provides a comparison of pre-training tasks when trained for separately vs. jointly and comparison with the Shallow CNN used in Das *et al.* [5]. The experiments showed that training jointly doesn't sacrifice performance on depth and autoencoding.

We train with a batch size of 20 and utilize Adam [3]. We use an initial learning rate of 10^{-5}, maximum epochs considered 300, we sampled every 3-rd frame from shortest paths. The checkpoint was selected by performance on a held-out validation set.

4. Question answering model training

- `lstm-question-only`, we train with a batch size of 40 and utilize Adam with a learning rate of 10^{-3};
- `nn-question-only` and `bow-question-only` we use the code provided in [6];
- `attention+*`, we train with a batch size of 20 and utilize Adam with a learning rate of 10^{-3};
- `spatial+RGB+Q` , we train with a batch size of 32 and utilize Adam with a learning rate of 10^{-3}

For all models, the best checkpoint is selected via performance on a held out validation set.

5. Navigation model training

For models trained with and without inflection weighted loss, we train and select checkpoints with the same procedure.

- `R+*`, we train with a batch size of 20 and utilize Adam with a learning rate of 10^{-3};
- `M+*`, we train with a batch size of 5 full sequences and utilize Adam with a learning rate of 2×10^{-4}. Note that due to the length of sequences, we compute the forward and backward pass for each sequence individually and average the gradients

Due to the difference in end-to-end evaluation and teacher-forcing accuracy, we use the following method to select checkpoints for navigation models: First we run non-minimal suppression with a window size of 5 on teacher forcing validation loss (vanilla cross-entropy for `NoIW-*`

Model	Total loss	AE loss	Depth loss	Sem. loss	PA	MPA	MIOU
Shallow CNN [5] – All Tasks	0.369	0.0047	0.0077	2.45	0.384	0.14	0.070
ResNet50 – AE only	1.774	**0.0030**	0.0873	8.71	0.008	0.02	0.002
ResNet50 – Depth only	2.311	0.1051	0.0072	11.89	0.005	0.02	0.002
ResNet50 – AE and depth only	1.003	0.0034	**0.0067**	9.02	0.006	0.03	0.002
ResNet50 – All Tasks	**0.356**	0.0040	0.0069	**2.48**	**0.390**	**0.15**	**0.078**

Table 1: Performance on the MP3D-EQA v1 validation set for Shallow CNN [5] and for ResNet50 when trained for autoencdoing only, for depth only, for autoencoding and depth, and for all tasks jointly. For segmentation we report the overall pixel accuracy (PA), mean pixel accuracy (MPA) averaged over all semantic classes and the mean IOU intersection over union (MIOU). For depth and autoencoder, we report the smooth-ℓ_1 on the validation set.

models, and inflection weighted cross-entropy otherwise). After NMS, we select the top 5 checkpoints by validation loss and run them through end-to-end evaluation on the validation set. The best checkpoint is the model that has the highest value of $\mathbf{QA} + \mathbf{d_\Delta}$ at T_{-50}. We find that inflection weighted cross-entropy is a significantly better predictor of end-to-end performance than vanilla cross-entropy. Interestingly, we also find that teacher forcing validation accuracy is a good predictor of end-to-end performance when training with inflection weighting and don't see a significant difference in performance if the procedure above is run with teacher forcing validation error instead of loss.

6. Results

We provide the full tables we first analyzed in their entirety and then sliced for the analysis provided in the main paper. Tab. 2 shows our primary results with inflection weighting and Tab. 3 shows the same set of models trained without inflection weighting.

We also provide the full tables with confidence intervals. Confidence intervals are 90% confidence intervals calculated with empirical bootstrapping. See Tab. 2 and Tab. 5.

As a reminder, we use the following notation to specify our models: For the base architecture, `R` denotes reactive models and `M` denotes memory models. The base architectures are then augmented with their input types, `+PC`, `+RGB`, and `+Q`. So a memory model that utilizes point clouds (but no question) is denoted as `M+PC`. Unless otherwise specified (by the prefix `NoIW`), models are trained with inflection weighting. We denote the two baseline navigators, forward-only and random, as `Fwd` and `Random`, respectively.

6.1. Navigation Performance Correlation Analysis

Here we provide some additional insight on the effect the question has on our navigation models. We find that while the question does significantly impact performance on average, it does significantly change a model's behavior.

Fig. 2 shows the Pearson correlation correlation coefficient between $\mathbf{d_T}$ at T_{-30} for all models and cluster discovery using the Nearest Point Algorithm. All navigators that use the reactive base architecture build a strong cluster

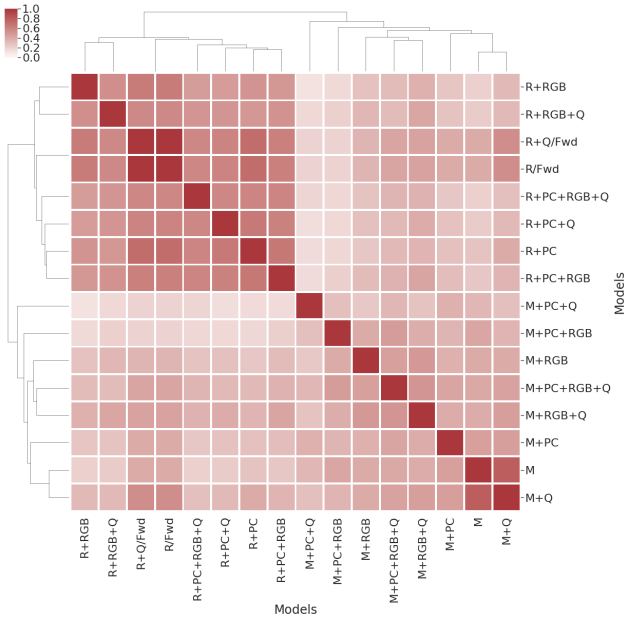

Figure 2: Correlation and clustering of correlation for d_T per episode at T_{-30}. Reactive models build a strong cluster, while memory models are less correlated. Usage of the question changes navigators with vision significantly.

and are highly correlated with each-other. The vision-less memory models are also well correlated with each-other, indicating that the question has little relation to the distribution of actions along a shortest path. Memory models that use vision don't tend to be well correlated with their counterpart that also uses the question in-spite of the question having little effect on overall navigation metrics.

Navigator	Navigation																		QA		
	$\mathbf{d_0}$ (For reference)			$\mathbf{d_T}$ (Lower is better)			$\mathbf{d_{min}}$ (Lower is better)			$\mathbf{d_\Delta}$ (Higher is better)			$\%_{\mathbf{collision}}$ (Lower is better)			$\mathbf{IoU_T}$ (Higher is better)			$\mathbf{Top-1}$ (Higher is better)		
	T_{-10}	T_{-30}	T_{-50}	T_{-10}	T_{-30}	T_{-50}	T_{-10}	T_{-30}	T_{-50}	T_{-10}	T_{-30}	T_{-50}	T_{-10}	T_{-30}	T_{-50}	T_{-10}	T_{-30}	T_{-50}	T_{-10}	T_{-30}	T_{-50}
R	0.354	1.898	3.547	0.933	1.330	2.154	**0.011**	0.346	1.397	−0.579	0.568	1.393	79.554	66.182	62.563	0.062	0.050	0.030	0.390	0.379	0.354
R+Q	0.354	1.898	3.547	0.933	1.330	2.154	**0.011**	0.346	1.397	−0.579	0.568	1.393	79.554	66.182	62.563	0.062	0.050	0.030	0.390	0.379	0.354
R+RGB	0.354	1.898	3.547	1.194	1.617	2.340	0.040	0.375	1.349	−0.840	0.281	1.207	59.959	51.460	48.425	0.077	0.058	0.031	0.395	0.396	0.372
R+RGB+Q	0.354	1.898	3.547	1.407	1.740	2.521	0.034	0.340	1.332	−1.053	0.157	1.026	51.128	44.160	42.692	0.111	0.070	0.054	0.383	0.388	0.375
R+PC	0.354	1.898	3.547	1.428	1.754	2.352	0.021	**0.320**	1.164	−1.074	0.144	1.195	50.148	41.612	42.203	0.070	0.067	0.047	0.356	0.394	0.375
R+PC+Q	0.354	1.898	3.547	1.514	1.812	2.394	0.033	0.325	**1.160**	−1.160	0.085	1.153	46.910	36.303	39.012	0.059	0.052	0.043	0.364	0.364	0.363
R+PC+RGB	0.354	1.898	3.547	1.547	1.791	2.336	0.020	0.322	1.211	−1.193	0.107	1.211	44.941	34.859	37.138	0.084	0.077	0.044	0.374	0.390	0.366
R+PC+RGB+Q	0.354	1.898	3.547	1.539	1.843	2.420	0.032	0.323	1.170	−1.185	0.055	1.127	42.018	34.318	37.069	0.067	0.072	0.055	0.370	0.395	0.369
M	0.354	1.898	3.547	**0.366**	**0.830**	1.833	0.090	0.505	1.460	**−0.012**	**1.068**	1.714	6.903	10.989	23.250	0.128	0.091	0.081	0.365	0.375	0.363
M+Q	0.354	1.898	3.547	0.508	0.933	1.920	0.052	0.426	1.421	−0.154	0.965	1.627	16.268	19.808	32.856	0.147	0.109	0.068	0.391	0.395	0.376
M+RGB	0.354	1.898	3.547	0.637	1.157	2.177	0.099	0.538	1.479	−0.283	0.741	1.370	12.582	15.130	26.179	0.188	0.136	0.075	0.397	0.403	0.384
M+RGB+Q	0.354	1.898	3.547	0.707	1.171	2.194	0.071	0.423	1.386	−0.353	0.727	1.353	14.212	15.908	25.578	0.189	0.141	0.083	**0.407**	0.394	0.384
M+PC	0.354	1.898	3.547	0.494	1.020	1.817	0.098	0.484	1.236	−0.140	0.878	1.730	6.647	9.169	18.319	0.163	0.114	0.083	0.396	**0.411**	**0.390**
M+PC+Q	0.354	1.898	3.547	0.502	1.030	1.910	0.081	0.497	1.272	−0.148	0.868	1.637	5.584	**8.833**	**15.783**	0.184	0.158	**0.118**	0.382	0.387	0.374
M+PC+RGB	0.354	1.898	3.547	0.461	0.940	**1.791**	0.103	0.513	1.269	−0.107	0.958	**1.756**	**4.957**	9.574	18.890	**0.209**	**0.179**	0.111	0.381	0.393	0.363
M+PC+RGB+Q	0.354	1.898	3.547	0.574	1.044	1.898	0.083	0.431	1.203	−0.220	0.854	1.649	8.328	10.674	19.797	**0.209**	0.148	0.112	0.389	0.390	0.373
Random	0.354	1.898	3.547	0.912	1.273	2.654	0.048	0.796	2.263	−0.558	0.625	0.893	13.775	10.708	10.677	0.098	0.072	0.041	0.365	0.368	0.364
ShortestPath	0.354	1.898	3.547	0.005	0.005	0.005	0.005	0.005	0.005	0.349	1.893	3.542	0.000	0.000	0.000	0.581	0.581	0.581	0.451	0.451	0.451

Table 2: Evaluation of EmbodiedQA agents trained with inflection weighting on navigation and answering metrics for the MP3D-EQA v1 test set. RGB models perceive the world via RGB images and use ResNet50. PC models perceive the world via point clouds and use PointNet++. PC+RGB models use both perception modalities and their respective networks.

Navigator	Navigation																		QA		
	$\mathbf{d_0}$ (For reference)			$\mathbf{d_T}$ (Lower is better)			$\mathbf{d_{min}}$ (Lower is better)			$\mathbf{d_\Delta}$ (Higher is better)			$\%_{\mathbf{collision}}$ (Lower is better)			$\mathbf{IoU_T}$ (Higher is better)			$\mathbf{Top-1}$ (Higher is better)		
	T_{-10}	T_{-30}	T_{-50}	T_{-10}	T_{-30}	T_{-50}	T_{-10}	T_{-30}	T_{-50}	T_{-10}	T_{-30}	T_{-50}	T_{-10}	T_{-30}	T_{-50}	T_{-10}	T_{-30}	T_{-50}	T_{-10}	T_{-30}	T_{-50}
NoIW-R	0.354	1.898	3.547	0.933	1.330	**2.154**	**0.011**	0.346	1.397	−0.579	0.568	**1.393**	79.554	66.182	62.563	0.062	0.050	0.030	0.390	0.379	0.354
NoIW-R+Q	0.354	1.898	3.547	0.933	1.330	**2.154**	**0.011**	0.346	1.397	−0.579	0.568	**1.393**	79.554	66.182	62.563	0.062	0.050	0.030	0.390	0.379	0.354
NoIW-R+RGB	0.354	1.898	3.547	1.419	1.713	2.528	0.041	0.404	1.417	−1.065	0.185	1.019	56.718	50.376	45.866	0.086	0.049	0.035	0.382	0.386	0.360
NoIW-R+RGB+Q	0.354	1.898	3.547	1.405	1.829	2.658	0.051	0.455	1.463	−1.051	0.069	0.889	43.226	38.538	36.172	0.075	0.060	0.054	0.384	0.390	0.373
NoIW-R+PC	0.354	1.898	3.547	1.385	1.662	2.483	0.026	**0.343**	1.294	−1.031	0.236	1.064	43.067	34.100	37.078	0.074	0.061	0.043	0.375	0.398	0.376
NoIW-R+PC+Q	0.354	1.898	3.547	1.515	1.858	2.646	0.038	0.394	1.333	−1.161	0.040	0.901	37.669	31.714	**33.563**	0.078	0.059	0.054	0.372	**0.399**	0.378
NoIW-R+PC+RGB	0.354	1.898	3.547	1.462	1.759	2.523	0.025	0.347	**1.285**	−1.108	0.139	1.024	53.785	41.955	40.293	0.067	0.052	0.042	0.384	0.389	0.369
NoIW-R+PC+RGB+Q	0.354	1.898	3.547	1.297	1.704	2.543	0.030	0.425	1.417	−0.943	0.194	1.004	47.506	39.554	36.242	0.069	0.062	0.046	0.368	0.375	0.353
NoIW-M	0.354	1.898	3.547	0.933	1.330	2.186	**0.011**	0.346	1.430	−0.579	0.568	1.361	79.554	66.182	62.818	0.062	0.050	0.029	0.390	0.379	0.356
NoIW-M+Q	0.354	1.898	3.547	0.933	1.330	2.202	**0.011**	0.346	1.445	−0.579	0.568	1.345	79.554	66.182	62.931	0.062	0.050	0.029	0.390	0.379	0.354
NoIW-M+RGB	0.354	1.898	3.547	0.902	1.396	2.512	0.024	0.400	1.608	−0.548	0.502	1.035	67.347	59.661	59.378	0.080	0.061	0.033	**0.406**	0.384	0.353
NoIW-M+RGB+Q	0.354	1.898	3.547	0.911	1.394	2.573	0.024	0.410	1.644	−0.557	0.504	0.974	66.198	59.317	58.941	0.094	0.064	0.030	0.390	0.380	0.356
NoIW-M+PC	0.354	1.898	3.547	0.811	1.245	2.244	0.034	0.370	1.414	−0.457	0.652	1.303	37.865	30.964	38.521	0.113	0.114	**0.091**	0.386	**0.399**	0.376
NoIW-M+PC+Q	0.354	1.898	3.547	**0.790**	**1.233**	2.213	0.038	0.379	1.416	**−0.436**	**0.665**	1.334	36.215	31.431	38.911	0.115	0.099	0.076	0.393	**0.399**	**0.379**
NoIW-M+PC+RGB	0.354	1.898	3.547	0.929	1.405	2.435	0.016	0.375	1.526	−0.575	0.492	1.112	61.658	51.595	52.093	0.100	0.075	0.041	0.388	0.385	0.353
NoIW-M+PC+RGB+Q	0.354	1.898	3.547	0.871	1.322	2.256	0.046	0.381	1.377	−0.517	0.576	1.291	**32.496**	**27.391**	34.923	**0.145**	**0.121**	0.088	0.383	0.398	0.374
NoIW-Random	0.354	1.898	3.547	0.912	1.273	2.654	0.048	0.796	2.263	−0.558	0.625	0.893	13.775	10.708	10.677	0.098	0.072	0.041	0.365	0.368	0.364
NoIW-ShortestPath	0.354	1.898	3.547	0.005	0.005	0.005	0.005	0.005	0.005	0.349	1.893	3.542	0.000	0.000	0.000	0.581	0.581	0.581	0.451	0.451	0.451

Table 3: Evaluation of EmbodiedQA agents trained **without** inflection weighting on navigation and answering metrics for the MP3D-EQA v1 test set.

Navigator	Navigation																		QA		
	$\mathbf{d_R}$ (For reference)			$\mathbf{d_T}$ (Lower is better)			$\mathbf{d_{min}}$ (Lower is better)			$\mathbf{d_A}$ (Higher is better)			$\overline{\mathbf{\%}}_{\mathbf{collision}}$ (Lower is better)			$\mathbf{IoU_T}$ (Higher is better)			$\mathbf{Top-1}$ (Higher is better)		
	T_{-10}	T_{-50}	T_{-50}	T_{-10}	T_{-50}	T_{-50}	T_{-10}	T_{-50}	T_{-50}	T_{-10}	T_{-50}	T_{-50}	T_{-10}	T_{-50}	T_{-50}	T_{-10}	T_{-50}	T_{-50}	T_{-10}	T_{-50}	T_{-50}
NoIM-R	0.354	1.898	3.547	0.933 (0.896, 0.969)	1.330 (1.290, 1.370)	2.154 (2.109, 2.196)	**0.011** (0.010, 0.013)	0.346 (0.333, 0.359)	1.397 (1.367, 1.426)	−0.579 (−0.615, −0.542)	0.568 (0.524, 0.611)	1.393 (1.347, 1.443)	79.554 (79.043, 80.065)	66.182 (65.604, 66.768)	62.563 (61.911, 63.244)	0.062 (0.056, 0.069)	0.050 (0.043, 0.056)	0.030 (0.026, 0.035)	0.390 (0.376, 0.404)	0.379 (0.364, 0.392)	0.354 (0.340, 0.367)
NoIM-R+Q	0.354	1.898	3.547	0.933 (0.896, 0.969)	1.330 (1.290, 1.370)	2.154 (2.109, 2.196)	**0.011** (0.010, 0.013)	0.346 (0.333, 0.359)	1.397 (1.367, 1.426)	−0.579 (−0.615, −0.542)	0.568 (0.524, 0.611)	1.393 (1.347, 1.443)	79.554 (79.043, 80.065)	66.182 (65.604, 66.768)	62.563 (61.911, 63.244)	0.062 (0.056, 0.069)	0.050 (0.043, 0.056)	0.030 (0.026, 0.035)	0.390 (0.376, 0.404)	0.379 (0.364, 0.392)	0.354 (0.340, 0.367)
NoIM-R+RGB	0.354	1.898	3.547	1.194 (1.152, 1.234)	1.617 (1.568, 1.665)	2.340 (2.294, 2.385)	0.040 (0.037, 0.043)	0.375 (0.360, 0.390)	1.349 (1.319, 1.379)	−0.840 (−0.881, −0.797)	0.281 (0.230, 0.333)	1.207 (1.156, 1.258)	59.959 (58.970, 60.958)	51.460 (50.585, 52.356)	48.425 (47.578, 49.301)	0.077 (0.069, 0.084)	0.058 (0.051, 0.064)	0.031 (0.026, 0.035)	0.395 (0.380, 0.409)	0.396 (0.381, 0.409)	0.372 (0.358, 0.386)
NoIM-R+RGB+Q	0.354	1.898	3.547	1.407 (1.361, 1.453)	1.740 (1.692, 1.789)	2.521 (2.467, 2.573)	0.034 (0.031, 0.037)	0.340 (0.327, 0.354)	1.332 (1.301, 1.363)	−1.053 (−1.101, −1.005)	0.157 (0.106, 0.209)	1.026 (0.969, 1.085)	51.126 (50.100, 52.154)	44.160 (43.263, 45.066)	42.692 (41.744, 43.618)	0.111 (0.101, 0.121)	0.070 (0.063, 0.078)	0.054 (0.047, 0.061)	0.383 (0.369, 0.397)	0.388 (0.374, 0.403)	0.375 (0.360, 0.389)
NoIM-R+PC	0.354	1.898	3.547	1.428 (1.379, 1.478)	1.754 (1.706, 1.800)	2.352 (2.299, 2.402)	0.021 (0.019, 0.022)	**0.320** (0.308, 0.333)	1.164 (1.135, 1.192)	−1.074 (−1.123, −1.023)	0.144 (0.092, 0.193)	1.195 (1.140, 1.253)	50.148 (49.084, 51.218)	41.612 (40.672, 42.570)	42.203 (41.226, 43.117)	0.070 (0.063, 0.077)	0.067 (0.060, 0.074)	0.047 (0.042, 0.053)	0.356 (0.342, 0.369)	0.394 (0.380, 0.408)	0.375 (0.361, 0.389)
NoIM-R+PC+Q	0.354	1.898	3.547	1.514 (1.466, 1.563)	1.812 (1.763, 1.862)	2.394 (2.344, 2.445)	0.033 (0.030, 0.035)	0.325 (0.312, 0.338)	**1.160** (1.131, 1.189)	−1.160 (−1.209, −1.110)	0.085 (0.034, 0.138)	1.153 (1.096, 1.210)	46.910 (45.833, 47.956)	36.303 (35.341, 37.244)	39.012 (38.074, 39.942)	0.059 (0.052, 0.065)	0.052 (0.045, 0.058)	0.043 (0.038, 0.048)	0.364 (0.350, 0.378)	0.364 (0.350, 0.378)	0.363 (0.349, 0.377)
NoIM-R+PC+RGB	0.354	1.898	3.547	1.547 (1.497, 1.594)	1.791 (1.743, 1.838)	2.336 (2.287, 2.385)	0.020 (0.018, 0.022)	0.322 (0.309, 0.335)	1.211 (1.182, 1.241)	−1.193 (−1.241, −1.143)	0.107 (0.055, 0.159)	1.211 (1.158, 1.265)	44.941 (43.856, 46.040)	34.859 (33.871, 35.853)	37.138 (36.190, 38.091)	0.084 (0.076, 0.093)	0.077 (0.068, 0.086)	0.044 (0.039, 0.049)	0.374 (0.360, 0.388)	0.390 (0.376, 0.404)	0.366 (0.352, 0.380)
NoIM-R+PC+RGB+Q	0.354	1.898	3.547	1.539 (1.490, 1.587)	1.843 (1.795, 1.890)	2.420 (2.369, 2.470)	0.032 (0.029, 0.035)	0.323 (0.311, 0.337)	1.170 (1.141, 1.200)	−1.185 (−1.233, −1.137)	0.055 (0.004, 0.107)	1.127 (1.072, 1.180)	42.018 (40.933, 43.100)	34.318 (33.370, 35.271)	37.069 (36.127, 38.004)	0.067 (0.060, 0.075)	0.072 (0.064, 0.080)	0.055 (0.049, 0.062)	0.370 (0.356, 0.384)	0.395 (0.381, 0.409)	0.369 (0.355, 0.382)
NoIM-H	0.354	1.898	3.547	**0.366** (0.354, 0.379)	**0.830** (0.811, 0.849)	1.833 (1.806, 1.867)	0.090 (0.085, 0.095)	0.505 (0.490, 0.521)	1.460 (1.432, 1.489)	**−0.012** (−0.026, 0.001)	**1.068** (1.042, 1.092)	1.714 (1.674, 1.754)	6.903 (6.508, 7.289)	10.989 (10.516, 11.464)	23.250 (22.632, 23.882)	0.128 (0.117, 0.138)	0.091 (0.083, 0.099)	0.081 (0.072, 0.089)	0.365 (0.351, 0.379)	0.375 (0.361, 0.389)	0.363 (0.349, 0.376)
NoIM-H+Q	0.354	1.898	3.547	0.508 (0.491, 0.524)	0.933 (0.912, 0.955)	1.920 (1.884, 1.955)	0.052 (0.048, 0.056)	0.426 (0.412, 0.441)	1.421 (1.392, 1.450)	−0.154 (−0.171, −0.136)	0.965 (0.937, 0.992)	1.627 (1.586, 1.669)	16.268 (15.645, 16.890)	19.808 (19.165, 20.458)	32.856 (32.135, 33.588)	0.147 (0.135, 0.157)	0.109 (0.099, 0.119)	0.068 (0.061, 0.075)	0.391 (0.376, 0.405)	0.395 (0.380, 0.409)	0.376 (0.363, 0.390)
NoIM-H+RGB	0.354	1.898	3.547	0.637 (0.611, 0.663)	1.157 (1.125, 1.187)	2.177 (2.129, 2.224)	0.099 (0.093, 0.105)	0.538 (0.521, 0.556)	1.479 (1.447, 1.510)	−0.283 (−0.310, −0.257)	0.741 (0.707, 0.775)	1.379 (1.319, 1.422)	12.582 (11.849, 13.341)	15.130 (14.404, 15.853)	26.179 (25.319, 27.038)	0.188 (0.174, 0.201)	0.136 (0.125, 0.148)	0.075 (0.067, 0.082)	0.397 (0.384, 0.411)	0.403 (0.388, 0.417)	0.384 (0.370, 0.397)
NoIM-H+RGB+Q	0.354	1.898	3.547	0.707 (0.681, 0.733)	1.171 (1.140, 1.202)	2.194 (2.146, 2.240)	0.071 (0.066, 0.076)	0.423 (0.408, 0.437)	1.386 (1.355, 1.416)	−0.353 (−0.380, −0.327)	0.727 (0.692, 0.762)	1.353 (1.301, 1.406)	14.212 (13.490, 14.925)	15.908 (15.219, 16.604)	25.578 (24.771, 26.392)	0.189 (0.175, 0.203)	0.141 (0.129, 0.153)	0.083 (0.075, 0.091)	**0.407** (0.392, 0.421)	0.394 (0.380, 0.408)	0.384 (0.370, 0.398)
NoIM-H+PC	0.354	1.898	3.547	0.494 (0.475, 0.512)	1.020 (0.993, 1.047)	1.817 (1.774, 1.859)	0.098 (0.092, 0.103)	0.464 (0.468, 0.500)	1.236 (1.207, 1.265)	−0.140 (−0.159, −0.120)	0.878 (0.846, 0.910)	1.230 (1.683, 1.778)	6.647 (6.073, 7.217)	9.169 (8.522, 9.804)	18.319 (17.510, 19.148)	0.163 (0.152, 0.174)	0.114 (0.104, 0.123)	0.083 (0.076, 0.090)	0.396 (0.381, 0.410)	**0.411** (0.396, 0.425)	**0.390** (0.376, 0.404)
NoIM-H+PC+Q	0.354	1.898	3.547	0.502 (0.483, 0.521)	1.030 (1.002, 1.059)	1.910 (1.864, 1.954)	0.081 (0.076, 0.086)	0.497 (0.481, 0.514)	1.272 (1.242, 1.302)	−0.148 (−0.168, −0.128)	0.868 (0.836, 0.900)	1.637 (1.588, 1.688)	5.584 (5.105, 6.066)	**8.833** (8.203, 9.454)	**15.783** (15.007, 16.548)	0.184 (0.171, 0.196)	0.158 (0.146, 0.170)	**0.118** (0.107, 0.128)	0.382 (0.368, 0.396)	0.387 (0.372, 0.401)	0.374 (0.360, 0.388)
NoIM-H+PC+RGB	0.354	1.898	3.547	0.461 (0.443, 0.478)	0.940 (0.915, 0.964)	**1.791** (1.751, 1.831)	0.103 (0.097, 0.109)	0.513 (0.497, 0.529)	1.269 (1.241, 1.297)	−0.107 (−0.125, −0.088)	0.959 (0.928, 0.987)	**1.756** (1.709, 1.802)	**4.957** (4.519, 5.391)	9.574 (8.973, 10.184)	18.890 (18.105, 19.660)	**0.209** (0.195, 0.222)	**0.179** (0.166, 0.191)	0.111 (0.102, 0.121)	0.381 (0.367, 0.395)	0.393 (0.378, 0.407)	0.363 (0.350, 0.377)
NoIM-H+PC+RGB+Q	0.354	1.898	3.547	0.574 (0.551, 0.598)	1.044 (1.014, 1.074)	1.898 (1.853, 1.941)	0.083 (0.078, 0.088)	0.431 (0.416, 0.446)	1.263 (1.173, 1.232)	−0.220 (−0.244, −0.196)	0.854 (0.819, 0.887)	1.649 (1.600, 1.700)	8.328 (7.687, 8.962)	10.674 (10.000, 11.342)	19.797 (18.973, 20.624)	**0.209** (0.195, 0.222)	0.148 (0.137, 0.160)	0.112 (0.102, 0.123)	0.389 (0.375, 0.404)	0.390 (0.376, 0.404)	0.373 (0.359, 0.387)
NoIM-Random	0.354	1.898	3.547	0.912 (0.890, 0.935)	1.273 (1.248, 1.298)	2.654 (2.619, 2.686)	0.048 (0.045, 0.051)	0.796 (0.778, 0.815)	2.263 (2.237, 2.289)	−0.558 (−0.583, −0.533)	0.625 (0.597, 0.651)	0.803 (0.861, 0.936)	13.775 (13.476, 14.069)	10.708 (10.431, 10.985)	10.677 (10.408, 10.946)	0.098 (0.088, 0.107)	0.072 (0.064, 0.080)	0.041 (0.035, 0.046)	0.365 (0.352, 0.380)	0.368 (0.354, 0.382)	0.364 (0.350, 0.378)
NoIM-ShortestPath	0.354	1.898	3.547	0.005 (0.004, 0.006)	0.005 (0.004, 0.006)	0.005 (0.004, 0.006)	0.005 (0.004, 0.006)	0.005 (0.004, 0.006)	0.005 (0.004, 0.006)	0.349 (0.340, 0.358)	1.893 (1.876, 1.909)	3.542 (3.522, 3.563)	0.000 (0.000, 0.000)	0.000 (0.000, 0.000)	0.000 (0.000, 0.000)	0.581 (0.567, 0.595)	0.581 (0.567, 0.595)	0.581 (0.567, 0.595)	0.451 (0.437, 0.466)	0.451 (0.437, 0.466)	0.451 (0.437, 0.465)

Table 4: Tab. 2 (navigation results with inflection weightings) with 90% bootstrap confidence intervals

Navigator	Navigation																				QA		
	$\mathbf{d_E}$ (Use reference)			$\mathbf{d_T}$ (Lower is better)			$\mathbf{d_{cube}}$ (Lower is better)			$\mathbf{d_A}$ (Higher is better)			$\overline{\%}_{\text{collision}}$ (Lower is better)			$\mathbf{IoU_T}$ (Higher is better)			Top − 1 (Higher is better)		
	T_{-10}	T_{-30}	T_{-50}	T_{-10}	T_{-30}	T_{-50}	T_{-10}	T_{-30}	T_{-50}	T_{-10}	T_{-30}	T_{-50}	T_{-10}	T_{-30}	T_{-50}	T_{-10}	T_{-30}	T_{-50}	T_{-10}	T_{-30}	T_{-50}
`NoIW-R`	0.354	1.898	3.547	0.933 (0.896, 0.969)	1.330 (1.290, 1.370)	**2.154** (2.109, 2.196)	**0.011** (0.010, 0.013)	0.346 (0.333, 0.359)	1.397 (1.367, 1.426)	−0.579 (−0.615, −0.542)	0.568 (0.524, 0.611)	**1.393** (1.347, 1.443)	79.554 (79.043, 80.065)	66.182 (65.604, 66.768)	62.563 (61.911, 63.244)	0.062 (0.056, 0.069)	0.050 (0.043, 0.056)	0.030 (0.026, 0.035)	0.390 (0.376, 0.404)	0.379 (0.364, 0.392)	0.354 (0.340, 0.367)
`NoIW-RnQ`	0.354	1.898	3.547	0.933 (0.896, 0.969)	1.330 (1.290, 1.370)	**2.154** (2.109, 2.196)	**0.011** (0.010, 0.013)	0.346 (0.333, 0.359)	1.397 (1.367, 1.426)	−0.579 (−0.615, −0.542)	0.568 (0.524, 0.611)	**1.393** (1.347, 1.443)	79.554 (79.043, 80.065)	66.182 (65.604, 66.768)	62.563 (61.911, 63.244)	0.062 (0.056, 0.069)	0.050 (0.043, 0.056)	0.030 (0.026, 0.035)	0.390 (0.376, 0.404)	0.379 (0.364, 0.392)	0.354 (0.340, 0.367)
`NoIW-RnRGB`	0.354	1.898	3.547	1.419 (1.372, 1.468)	1.713 (1.666, 1.759)	2.528 (2.475, 2.580)	0.041 (0.037, 0.044)	0.404 (0.389, 0.419)	1.417 (1.385, 1.449)	−1.065 (−1.114, −1.016)	0.185 (0.135, 0.235)	1.019 (0.963, 1.077)	56.718 (55.722, 57.732)	50.376 (49.506, 51.248)	45.866 (44.964, 46.762)	0.086 (0.077, 0.094)	0.049 (0.043, 0.055)	0.035 (0.030, 0.040)	0.382 (0.368, 0.396)	0.386 (0.372, 0.400)	0.360 (0.346, 0.373)
`NoIW-RnRGB+Q`	0.354	1.898	3.547	1.405 (1.353, 1.454)	1.829 (1.775, 1.882)	2.658 (2.599, 2.716)	0.051 (0.047, 0.055)	0.455 (0.438, 0.472)	1.463 (1.430, 1.496)	−1.051 (−1.101, −0.999)	0.069 (0.012, 0.125)	0.889 (0.827, 0.953)	43.236 (42.166, 44.334)	38.538 (37.559, 39.510)	36.172 (35.216, 37.109)	0.075 (0.066, 0.083)	0.060 (0.053, 0.067)	0.054 (0.047, 0.061)	0.384 (0.369, 0.398)	0.390 (0.376, 0.404)	0.373 (0.359, 0.387)
`NoIW-RnPC`	0.354	1.898	3.547	1.385 (1.338, 1.430)	1.662 (1.616, 1.708)	2.483 (2.428, 2.535)	0.026 (0.023, 0.028)	**0.343** (0.329, 0.358)	1.294 (1.263, 1.324)	−1.031 (−1.078, −0.983)	0.236 (0.185, 0.286)	1.064 (1.005, 1.123)	43.067 (41.951, 44.169)	34.100 (33.167, 35.046)	37.078 (36.097, 38.050)	0.074 (0.066, 0.082)	0.061 (0.054, 0.068)	0.043 (0.038, 0.048)	0.375 (0.361, 0.389)	0.398 (0.384, 0.412)	0.376 (0.362, 0.390)
`NoIW-RnPC+Q`	0.354	1.898	3.547	1.515 (1.464, 1.564)	1.858 (1.806, 1.910)	2.646 (2.589, 2.700)	0.038 (0.035, 0.042)	0.394 (0.378, 0.410)	1.333 (1.302, 1.365)	−1.161 (−1.212, −1.108)	0.040 (−0.016, 0.096)	0.901 (0.842, 0.962)	37.669 (36.572, 38.772)	31.714 (30.788, 32.640)	**33.563** (32.594, 34.548)	0.078 (0.070, 0.085)	0.059 (0.052, 0.066)	0.054 (0.047, 0.060)	0.372 (0.359, 0.387)	**0.399** (0.385, 0.413)	0.376 (0.364, 0.392)
`NoIW-RnPC+RGB`	0.354	1.898	3.547	1.462 (1.415, 1.508)	1.759 (1.712, 1.806)	2.525 (2.468, 2.576)	0.025 (0.022, 0.027)	0.347 (0.332, 0.362)	**1.285** (1.254, 1.316)	−1.108 (−1.155, −1.060)	0.139 (0.087, 0.189)	1.024 (0.966, 1.084)	53.785 (52.783, 54.794)	41.955 (41.038, 42.882)	40.293 (39.360, 41.224)	0.067 (0.059, 0.075)	0.052 (0.045, 0.058)	0.042 (0.036, 0.047)	0.384 (0.370, 0.398)	0.389 (0.375, 0.403)	0.369 (0.356, 0.383)
`NoIW-RnPC+RGB+Q`	0.354	1.898	3.547	1.297 (1.251, 1.342)	1.704 (1.654, 1.753)	2.543 (2.489, 2.595)	0.030 (0.028, 0.033)	0.425 (0.408, 0.441)	1.417 (1.385, 1.449)	−0.943 (−0.989, −0.897)	0.194 (0.142, 0.247)	1.004 (0.947, 1.063)	47.506 (46.410, 48.588)	39.554 (38.569, 40.565)	36.242 (35.263, 37.210)	0.069 (0.062, 0.076)	0.062 (0.055, 0.069)	0.046 (0.040, 0.052)	0.368 (0.354, 0.382)	0.375 (0.360, 0.389)	0.353 (0.340, 0.367)
`NoIW-H`	0.354	1.898	3.547	0.933 (0.896, 0.969)	1.330 (1.290, 1.370)	2.186 (2.141, 2.229)	**0.011** (0.010, 0.013)	0.346 (0.333, 0.359)	1.430 (1.400, 1.460)	−0.579 (−0.615, −0.542)	0.568 (0.524, 0.611)	1.301 (1.314, 1.410)	79.554 (79.043, 80.065)	66.182 (65.604, 66.768)	62.818 (62.155, 63.487)	0.062 (0.056, 0.069)	0.050 (0.043, 0.056)	0.029 (0.025, 0.033)	0.390 (0.376, 0.404)	0.379 (0.364, 0.392)	0.356 (0.341, 0.369)
`NoIW-HnQ`	0.354	1.898	3.547	0.933 (0.896, 0.969)	1.330 (1.290, 1.370)	2.202 (2.156, 2.244)	**0.011** (0.010, 0.013)	0.346 (0.333, 0.359)	1.445 (1.415, 1.475)	−0.579 (−0.615, −0.542)	0.568 (0.524, 0.611)	1.345 (1.299, 1.395)	79.554 (79.043, 80.065)	66.182 (65.604, 66.768)	62.931 (62.273, 63.603)	0.062 (0.056, 0.069)	0.050 (0.043, 0.056)	0.029 (0.025, 0.033)	0.390 (0.376, 0.404)	0.379 (0.364, 0.392)	0.354 (0.340, 0.368)
`NoIW-HnRGB`	0.354	1.898	3.547	0.902 (0.866, 0.937)	1.396 (1.355, 1.436)	2.512 (2.456, 2.566)	0.024 (0.021, 0.026)	0.400 (0.385, 0.415)	1.608 (1.574, 1.643)	−0.548 (−0.584, −0.511)	0.502 (0.458, 0.547)	1.035 (0.980, 1.093)	67.347 (66.413, 68.309)	59.661 (58.861, 60.466)	59.378 (58.660, 60.133)	0.080 (0.072, 0.088)	0.061 (0.054, 0.068)	0.033 (0.028, 0.038)	**0.406** (0.392, 0.420)	0.384 (0.370, 0.398)	0.353 (0.339, 0.366)
`NoIW-HnRGB+Q`	0.354	1.898	3.547	0.911 (0.874, 0.948)	1.394 (1.351, 1.435)	2.573 (2.515, 2.630)	0.024 (0.021, 0.026)	0.410 (0.394, 0.425)	1.644 (1.609, 1.678)	−0.557 (−0.594, −0.519)	0.504 (0.460, 0.549)	0.974 (0.917, 1.034)	66.198 (65.281, 67.131)	59.317 (58.523, 60.122)	58.941 (58.212, 59.682)	0.094 (0.086, 0.103)	0.064 (0.057, 0.071)	0.030 (0.026, 0.035)	0.390 (0.376, 0.404)	0.380 (0.365, 0.393)	0.356 (0.343, 0.370)
`NoIW-HnPC`	0.354	1.898	3.547	0.811 (0.780, 0.841)	1.245 (1.209, 1.282)	2.244 (2.194, 2.292)	0.034 (0.031, 0.037)	0.370 (0.357, 0.385)	1.414 (1.383, 1.447)	−0.457 (−0.488, −0.425)	0.652 (0.612, 0.693)	1.303 (1.251, 1.357)	37.865 (36.674, 39.002)	30.964 (29.981, 31.909)	38.521 (37.532, 39.517)	0.113 (0.103, 0.122)	0.114 (0.104, 0.123)	**0.091** (0.081, 0.100)	0.386 (0.372, 0.401)	**0.399** (0.385, 0.413)	0.376 (0.362, 0.390)
`NoIW-HnPC+Q`	0.354	1.898	3.547	**0.790** (0.760, 0.820)	**1.233** (1.197, 1.269)	2.213 (2.163, 2.262)	0.038 (0.034, 0.041)	0.379 (0.366, 0.393)	1.416 (1.385, 1.448)	**−0.436** (−0.466, −0.404)	**0.665** (0.626, 0.704)	1.334 (1.282, 1.387)	36.215 (35.059, 37.351)	31.431 (30.439, 32.446)	38.911 (37.929, 39.887)	0.115 (0.106, 0.124)	0.099 (0.090, 0.108)	0.076 (0.068, 0.083)	0.393 (0.379, 0.407)	**0.399** (0.385, 0.413)	**0.379** (0.364, 0.392)
`NoIW-HnPC+RGB`	0.354	1.898	3.547	0.929 (0.894, 0.965)	1.405 (1.362, 1.448)	2.435 (2.378, 2.490)	0.016 (0.014, 0.018)	0.375 (0.360, 0.389)	1.526 (1.494, 1.558)	−0.575 (−0.610, −0.539)	0.492 (0.446, 0.539)	1.112 (1.055, 1.171)	61.658 (60.620, 62.668)	51.595 (50.659, 52.541)	52.093 (51.216, 52.969)	0.100 (0.092, 0.109)	0.075 (0.068, 0.082)	0.041 (0.036, 0.046)	0.388 (0.374, 0.402)	0.385 (0.371, 0.400)	0.353 (0.339, 0.366)
`NoIW-HnPC+RGB+Q`	0.354	1.898	3.547	0.871 (0.837, 0.905)	1.322 (1.283, 1.362)	2.256 (2.203, 2.308)	0.046 (0.043, 0.050)	0.381 (0.367, 0.396)	1.377 (1.345, 1.410)	−0.517 (−0.551, −0.482)	0.576 (0.532, 0.620)	1.291 (1.236, 1.349)	**32.496** (31.347, 33.629)	**27.391** (26.415, 28.397)	34.923 (33.916, 35.928)	**0.145** (0.133, 0.157)	**0.121** (0.111, 0.131)	0.088 (0.079, 0.098)	0.383 (0.369, 0.398)	0.398 (0.384, 0.413)	0.374 (0.360, 0.388)
`NoIW-Random`	0.354	1.898	3.547	0.912 (0.890, 0.935)	1.273 (1.248, 1.298)	2.654 (2.619, 2.686)	0.048 (0.045, 0.051)	0.796 (0.778, 0.815)	2.263 (2.237, 2.289)	−0.558 (−0.583, −0.533)	0.625 (0.597, 0.651)	0.803 (0.861, 0.926)	13.775 (13.476, 14.069)	10.708 (10.431, 10.985)	10.677 (10.408, 10.946)	0.098 (0.088, 0.107)	0.072 (0.064, 0.080)	0.041 (0.035, 0.046)	0.365 (0.352, 0.380)	0.368 (0.354, 0.382)	0.364 (0.350, 0.378)
`NoIW-ShortestPath`	0.354	1.898	3.547	0.005 (0.004, 0.006)	0.005 (0.004, 0.006)	0.005 (0.004, 0.006)	0.005 (0.004, 0.006)	0.005 (0.004, 0.006)	0.005 (0.004, 0.006)	0.349 (0.340, 0.358)	1.893 (1.876, 1.909)	3.542 (3.522, 3.563)	0.000 (0.000, 0.000)	0.000 (0.000, 0.000)	0.000 (0.000, 0.000)	0.581 (0.567, 0.595)	0.581 (0.567, 0.595)	0.581 (0.567, 0.595)	0.451 (0.437, 0.466)	0.451 (0.437, 0.466)	0.451 (0.437, 0.465)

Table 5: Tab. 5 (navigation results **without** inflection weightings) with 90% bootstrap confidence intervals

References

[1] Angel Chang, Angela Dai, Thomas Funkhouser, Maciej Halber, Matthias Niessner, Manolis Savva, Shuran Song, Andy Zeng, and Yinda Zhang. Matterport3D: Learning from RGB-D data in indoor environments. *International Conference on 3D Vision (3DV)*, 2017. 1, 2

[2] Charles Ruizhongtai Qi, Li Yi, Hao Su, and Leonidas J Guibas. Pointnet++: Deep hierarchical feature learning on point sets in a metric space. In *NIPS*, 2017. 1, 2

[3] Diederik Kingma and Jimmy Ba. Adam: A Method for Stochastic Optimization. In *ICLR*, 2015. 3

[4] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 770–778, 2016. 3

[5] Abhishek Das, Samyak Datta, Georgia Gkioxari, Stefan Lee, Devi Parikh, and Dhruv Batra. Embodied Question Answering. In *CVPR*, 2018. 3, 4

[6] Ankesh Anand, Eugene Belilovsky, Kyle Kastner, Hugo Larochelle, and Aaron Courville. Blindfold Baselines for Embodied QA. *arXiv preprint arXiv:1811.05013*, 2018. 3