Supplementary Material for: GAIT: Generating Aesthetic Indoor Tours with Deep Reinforcement Learning

Desai Xie[1] Ping Hu[1] Xin Sun[2] Sören Pirk[2]
Jianming Zhang[2] Radomír Měch[2] Arie E. Kaufman[1]
[1]Stony Brook University [2]Adobe Research

In this supplementary material document, we cover additional implementation details (Section 1), more experimental results (Section 2), and user study details in Section 2.3.

1. Additional Implementation Details

1.1. Orientation Observation

We implement the 2D orientation of yaw θ and pitch ψ as doing a yaw rotation and then a pitch rotation from the default, forward orientation. In order to provide a linear relationship between $\hat{x}^P_{t+1} - \hat{x}^P_t$ and $\hat{a}_t$ for the networks to learn, rotation actions $\hat{a}_t$ are thus implemented by first resetting to the default orientation and then rotating to the new yaw and pitch angles

$$[\theta_{t+1}, \psi_{t+1}] = \hat{x}^P_{t+1} = \hat{x}^P_t \oplus \hat{a}_t. \quad (1)$$

When θ exceeds π or $-\pi$, it is reset to be within the range by subtracting or adding 2π respectively. ψ is hard bounded within the range $[-\frac{\pi}{2}, \frac{\pi}{2}]$.

We do not consider camera roll because it is common practice not to have roll rotation (i.e., camera's Up vector is always vertically up) when capturing aesthetic photos in indoor scenes. If a task requires to control the roll, it can easily be added to our GAIT framework.

1.2. Training Process

We train *GAIT-DrQ-v2* and *GAIT-CURL* in a single 3D indoor scene. We run the Data Worker for $3M$ and $1.5M$ data steps for *GAIT-DrQ-v2* and *GAIT-CURL* respectively.

Data Worker: For every $T = 15$ steps: an episode terminates in the Data Worker; this episode is pushed to the shared Replay Buffer; the environment is reset to prepare for the next episode. Each episode is reset to $t = 0$, a random initial pose within $[-1, 1]^5$, a exclusion pose list of four NULL exclusion poses, i.e. $[-1.5, -1.5, -1.5, -1.5, -1.5]$, every 5 episodes, or the ending pose of the previous episode is appended to the exclusion pose list, and a initialized recent actions list consisting only the zero-action $[0, 0, 0, 0, 0]$. Staring with the initial observations x_0, the agent takes $\{a_t\}_{t=0,\dots,T-1}$ actions according to the Actor Network $a_t = \pi(x_t)$, and observes $\{x_{t+1}\}_{t=0,\dots,T-1}$.

Update Worker: We run the Update Worker with a $2:1$ data-to-update step ratio with the Data Worker. At each update step, a batch of n-step transitions $[x_t, a_t, r_t, x_{t+n}]$ is sampled from the replay buffer, where $n = 3, 1$ for *GAIT-DrQ-v2* and *GAIT-CURL* respectively. With this batch the Actor and Critic Networks are updated separately. The Actor and Critic Networks share an encoder, which is updated with the Critic loss [18, 7]. For *GAIT-CURL*, the encoder is additionally updated with the contrastive loss [7]

1.3. Network Architecture

DrQ-v2 [18] and CURL [7] share the network architecture originally proposed in SAC-AE [19]. A notable feature of this architecture is that it has an information bottleneck on the image observations, where the flattened feature maps of size $35 \times 35 \times 32 = 39200$ is first reduced to size 50 after the linear layer, and then passed through a MLP with two hidden layers of size 1024 before it eventually reaches the output layer. In *GAIT-DrQ-v2* and *GAIT-CURL* we include additional observations (see Section 3.1 in the main text) in addition to the image observation input and concatenate them with the size 50 image features at the information bottleneck. x^D_t and x^S_t both pass through a separate linear layer with an outputs size of 128 before the concatenation. The Actor and Critic both share this architecture: the Actor outputs a 5D action a_t, while the Critic takes a_t as an additional input and outputs a scalar state-action value $Q(x_t, a_t)$. As previously mentioned, the Actor and Critic Networks share the same CNN encoder network.

Visual DRL literature commonly use this kind of shallow, simple network architecture as opposed to the usage of large pre-trained backbone networks in Computer Vision literature [9, 14, 5]. This is because Visual RL algorithms such as DrQ-v2 [18] and CURL [7] are able to achieve efficient representation learning, while naively using a large pre-trained backbone network for Visual RL actually reduces learning efficiency and training throughput [17].

1.4. Multi-GPU Design

We use Ray [8] to scale *GAIT-DrQ-v2* and *GAIT-CURL* to multi-GPU. As shown in Figure 2 in main text , on a compute node with 8 GPUs, we run 7 Data Workers and 1 Update Worker, so each worker occupies a GPU. Data Workers and the Update Worker run Data steps and Update steps asynchronously but synchronizes to maintain the $2:1$ Data-Update step ratio every 15 steps.

1.5. Fast Replay Buffer and Image Augmentation

DrQ-v2 [18] introduces implementations of a fast *replay buffer* and fast *image augmentation*. The fast *replay buffer* is composed of a *replay storage* and a *replay loader*. The replay storage receives transitions from the data loop (or the data Worker in multi-GPU *GAIT*). Once an episode is finished, it is written to disk as a Numpy [3] .npz file. In the update loop (or the update worker in multi-GPU *GAIT*), the replay loader loads new episode .npz files to memory, randomly samples a batch of transitions from the loaded episode. Here, the batch is loaded in page-locked memory by using PyTorch Dataloader's Pin Memory feature [12, 11].

This allows for significant speedup of data transfer from CPU to GPU memory [4]. This implementation also leads to 10x larger replay buffer capacity. We scale this replay buffer implementation to multi-GPU by using a replay storage for each Data Worker and have the Replay Loader to run in the update worker. We also integrate this fast, multi-GPU replay buffer with *GAIT-CURL*, which by default has a naive in-memory Replay Buffer.

The DrQ-v2 fast image augmentation [18] exploits the fast GPU computation provided by the PyTorch [11] grid_sample function. It is a flow-field image sampling method, which is used to do random shift augmentation. Similarly, we replace CURL's naive CPU random crop augmentation [7] with one implemented with PyTorch grid_sample [11]. This leads to a 7% more throughput than the naive CPU random crop augmentation.

1.6. Gaussian Smoothness Function Visualization

In Figure S1, we visualize the Gaussian Temporal Smoothness function with mean $||x|| - ||a_{t-i}||$, where $||a_{t-i}|| = 0.2, 0.3, 0.4$ and standard deviation $\frac{1}{2}||a_{t-i}|| = 0.1, 0.15, 0.2$ for green, red, purple respectively.

1.7. CMA-ES

Covariance Matrix Adaptation Evolution Strategy (CMA-ES) [2] is a stochastic, gradient-free optimization method for the non-linear, non-convex functions in the continuous domain. As opposed to a Deep Neural Network, it has no observations and no training process. Instead, it optimizes the given fitness function by repeatedly trying different input variable values. We use CMA-ES in the *GAIT* environment following a similar setup as *GAIT* DRL agents. An aesthetic indoor tour sequence is produced by running CMA-ES for 15 separate optimizations, where each optimizes the *GAIT* reward of the current step r_t by trying different 5D actions a_t. The greedy action, which produces the highest *GAIT* reward is chosen as the action for the current step

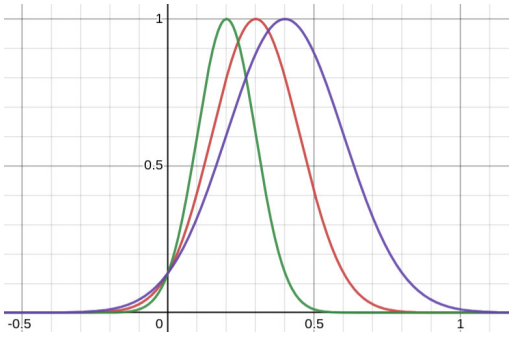


Figure S1: Gaussian Temporal Smoothness functions with mean $||x|| - ||a_{t-i}||$, where $||a_{t-i}|| = 0.2, 0.3, 0.4$ and standard deviation $\frac{1}{2}||a_{t-i}|| = 0.1, 0.15, 0.2$ for green, red, purple respectively.

$$a_t^* = \underset{a_t}{\operatorname{argmax}}(r_t). \tag{2}$$

Note that this optimization process does not consider *GAIT* observations x_t. In our experiments we use the pycma package [1] to implement CMA-ES for *GAIT*.

Unlike our *GAIT* DRL agent, which explores the 3D indoor scene during the training process, CMA-ES chooses actions that only optimizes the current reward. As shown in Figure S8, CMA-ES gets stuck next to an exclusion pose, because the small actions of staying in that view give the highest immediate reward. On the other hand, a *GAIT* DRL agent would be able to sacrifice the reward of one step to get out of that pose, knowing that it is possible to get much higher later rewards.

Compared to a *GAIT* DRL agent, CMA-ES does not scale as easily. CMA-ES requires more than an hour to generate a sequence, while a *GAIT* DRL agent can generate a sequence in less than a second. Therefore, a *GAIT* DRL agent can repeatedly generate aesthetic indoor tour sequences in a scene with different initial poses and different exclusion poses and distances, while CMA-ES needs to rerun the optimization process if a different initial pose or a different exclusion pose is given.

1.8. MPC

Model Predictive Control (MPC) [10] is a classic planning algorithm used to solve control problems, where a model of the environment is used to predict future timesteps

to choose the current action. Here we detail our implementation of MPC with the *GAIT* framework. To decide the next action, we roll out trajectories of length $k = 5$ from the current state, and choose the action whose trajectory has the maximum cumulative reward. To deal with the 5D continuous action space, we randomly sample $n = 8$ actions from the action space for each roll-out step [6]. With our implementation of MPC, it takes 1.25 hours to generate a sequence. Its average evaluation reward is significantly lower than *GAIT-DrQ-v2*, *GAIT-CURL*, and CMA-ES, because it is unable to explore the 5D continuous action space efficiently.

1.9. 3D Indoor Scenes in Replica Dataset

We perform experiments in the Replica [15] dataset using the Habitat-sim Simulator [13, 16]. There are a total of 18 scenes in Replica. We selected 6 out of the 18 that contain the least amount of visual artifacts, has a box-shaped room and a distinct appearance and color distribution in comparison to the other 5. In the paper, we test *GAIT* on three scenes, Room0, Apartment2, Office3, which are abbreviated as Room, Apartment, and Office. In this supplementary material, we use their full names of Room0, Apartment2, Office3. We report some results on three additional scenes, Room1, Apartment0, and Office0.

For some 3D indoor scenes the furniture placements and visual features are more diverse in some areas. This causes a skewed distribution of aesthetic scores. In our experiments, we notice that in Office0 and Office3, *GAIT* agents are mostly attracted to the blue display, which has a distinct color from the neighboring region and dense information from the world map on the display (See Figure S13 and S15). Views containing the blue display indeed have higher aesthetic scores than the others. Similarly, in Room1, *GAIT-CURL* agent is attracted to the pillows on the bed, where one pillow has an owl pattern on it (see Figure S17).

1.10. Interpolation for Generating Video Clips

Given the generated sequence of 16 camera poses including the initial pose, 9 intermediate camera poses are further interpolated between any two adjacent poses thus a video clip of 5 seconds is produced. For an intermediate camera pose, its rotation is evaluated with spherical linear interpolation between the adjacent two frames while its position is linearly interpolated.

1.11. Training Hardware

We run single-GPU and multi-GPU training on 8-GPU compute nodes. It is equipped with dual Intel Xeon Silver 4214R 2.40GHz CPUs, 8 GPUs of NVIDIA RTX A5000 (24GB VRAM), and 251GB memory.

Table S1: ***GAIT* Environment**

Hyperparameter	Value
Evaluation every episodes	300
Evaluation episodes number	10
Action a dimensions	5
Pose x^P dimensions	5
Azimuth orientation range	$[-\pi, \pi]$
Elevation orientation range	$[-\frac{\pi}{2}, \frac{\pi}{2}]$
Step sizes	$[0.25, 0.25, 0.25, 0.25, 0.25]$
Episode length T	15
Out of boundary penalty $r^{\mathcal{B}}$	-10
Diversity exclusion poses number	4
Smoothness recent actions window	3
Smoothness Gaussian translation denominator	2
Smoothness Gaussian rotation denominator	1
Camera FOV	60
Aesthetic model image resolution	240×240
Data Worker model sync per step	30
Data Worker step number sync per Data step	15
Update Worker model sync per step	15
Update Worker step number sync per Update step	10
Data-Update step ratio	$2:1$

1.12. Full Hyperparameter List

Table S1, S2, S3, and S4 lists our hyperparameters.

Table S2: **Shared network architecture of *GAIT-DrQ-v2* and *GAIT-CURL*, listed in the order of layers**

Hyperparameter	Value
Encoder input image observation resolution	84×84
Encoder number of filters	$[32, 32, 32, 32]$
Encoder kernel size	$[3, 3, 3, 3]$
Encoder stride	$[2, 1, 1, 1]$
Encoder output image features size	$32 \cdot 35 \cdot 35 = 39200$
Information bottleneck image feature size	50
Diversity observation hidden size	128
Smoothness observation hidden size	128
MLP hidden size	$[1024, 1024]$

Table S3: ***GAIT-DrQ-v2***

Hyperparameter	Value
Total Data steps	$3 \cdot 10^6$
Linearly decayed noise schedule	$1 \cdot 10^6$
Discount factor γ	0.99
No update before	4000
Replay Buffer size	10^6
n-step TD	3
Batch size	256
learning rate	10^{-4}
Image observation resolution	84×84
Critic τ	0.01

2. Additional Experimental Results

2.1. Aesthetic, Diversity, and Smoothness Scores

Table S7, S8, and S9 shows the aesthetics, diversity, and smoothness scores for Figure 4, 6, and 7 in main text, allowing for a quantitative evaluation of *GAIT*. Each cell in

Table S4: ***GAIT-CURL***

Hyperparameter	Value
Total Data steps	$1.5 \cdot 10^6$
Discount factor γ	0.99
No update before	4000
Replay Buffer size	10^6
n-step TD	1
Batch size	512
Critic learning rate	10^{-3}
Critic β	0.9
Critic τ	0.01
Critic target update frequency	2
Actor learning rate	10^{-3}
Actor β	0.9
Actor log std min	-10
Actor log std max	2
Actor update frequency	2
Encoder learning rate	10^{-3}
Encoder τ	0.05
CURL contrastive learning latent dimensions	128
SAC initial temperature	0.1
SAC α learning rate	10^{-4}
SAC α β	0.5
Image observation resolution before random crop	100×100
Image observation resolution after random crop	84×84

the score tables corresponds to a frame in the sequence visualization figures. Readers are encouraged to view the sequence visualization figures and the score tables side by side to gain a better understanding of the sequences under the *GAIT* reward.

2.2. Sequence Visualizations and Training Curves in Additional Scenes

In the main text, we only have space to report sequence visualization figures in Room0 and training curves on average evaluation return. In Figure S12, S13, S14, and S15, we replicate the comparison conditions for **(a)** Trajectory Generation, **(b)** Diversity, and **(c)** Smoothness (Figure 4, 6, 7 in main text) in Apartment2 and Office3 with DrQv2 and CURL respectively. Specifically, for **(a)** Trajectory Generation sequences, we show that 3 sequences, with two initial poses close to each other and one far from the other two, all converge to target poses that are close; for **(b)** Diversity, we show that 3 sequences, with the same initial pose and 0, 1, 2 prior ending poses as exclusion poses respectively, all manage to avoid the exclusion poses while keeping a high aesthetics and smoothness score; for **(c)** Smoothness, we compare 2 sequences with the same initial pose, generated with agents trained with and without the Temporal Smoothness Regularization.

In Figure S16 and S17, we show 3 sequences per 3 additional scenes (Room1, Apartment0, Office0) for *GAIT-DrQ-v2* and *GAIT-CURL* respectively. Each 3 sequences have the same initial pose and use the 0, 1, 2 prior ending poses as exclusion poses, resembling the Diversity comparison sequences.

A training curve refers to the agent's average evaluation return over the training process (See Section 4.4 in main text). In Figure S9, we report the training curves for *GAIT-DrQ-v2* and *GAIT-CURL* on three additional scenes (Room1, Apartment0, Office0).

In Figure S10, we report the training curve distributions over 3 seeds for *GAIT-DrQ-v2* and *GAIT-CURL* on three scenes (Room0, Apartment2, Office3). Both *GAIT-DrQ-v2* and *GAIT-CURL* are not sensitive to random seeds.

In Figure S11, we compare training curves of *GAIT-DrQv2* on Room0 with the default settings. The single-GPU version is able to converge with fewer training steps. This is because its Data loop and Update loop uses the same set of networks, so that the updated networks are immediately used in Data collection. On the contrary, the Data Workers in the multi-GPU version have to delay the synchronization of the updated models from the Update Worker in order to have an improved training throughput while sacrificing learning efficiency. We tested different model synchronization frequencies that achieve the shortest time-to-converge and a balance between throughput and efficiency.

2.3. User Study

We conducted the user study in three scenes with different object placement and varying colors, including Room 0, Office 3, and the living room of Apartment 2. In each scene, three initial camera poses were randomly selected in the room. We generated three view sequences (with 0, 1, 2 ending pose exclusion, respectively) starting from each initial pose. Furthermore, the corresponding video clip was produced based on each view sequence as described in Section 3.4 in the main manuscript. Since we trained our model in both *GAIT-DrQ-v2* and *GAIT-CURL*, we compare these two methods to see if one method ourperforms the other. Furthermore, we compare the generated tours in *CMA-ES* with those generated in *GAIT-DrQ-v2*. For simplicity, we also use Method C, M, and D to represent the method *GAIT-CURL*, *CMA-ES*, and *GAIT-DrQ-v2* correspondingly in the following description and reports in this section (Section 2.3). In other words, given any initial pose, participants were asked to compare three video clip pairs of C $vs.$ D and M $vs.$ D because of three exclusion settings of ending poses. After watching the two video clips in a pair one by one, each participant were asked to answer questions $Q1$ and $Q2$ (as reported in Table S5). After the three pairs, $Q3$ and $Q4$ (in Table S5) were asked to rate the two methods' overall performance. In addition, to overcome the effect from previous video clips, the order of playing the video clips was randomized among participants and among different trials.

The users' votes are summarized in Figure S2 and Figure S3. We find that users' voting pattern varies among dif-

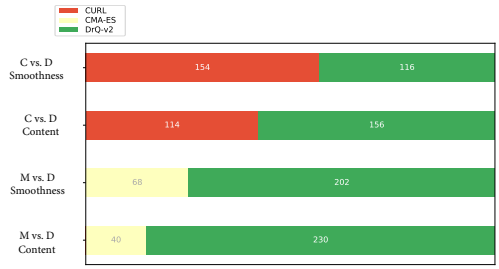


Figure S2: The summation of user votes in $Q1$ and $Q2$, reflecting the general subjective preference to the tour smoothness and the content aesthetics. The votes for Method D, C, and M are colored in green, red, and yellow, respectively.

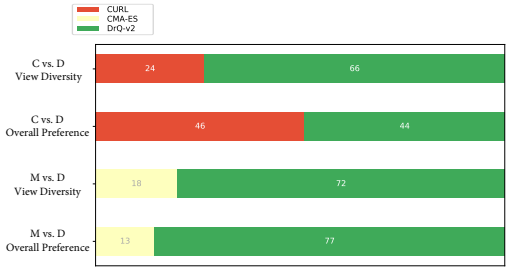


Figure S3: The summation of user votes in $Q3$ and $Q4$, reflecting the general subjective preference to the view diversity in different methods and the preference to the overall results in these methods. The votes for Method D, C, and M are colored in green, red, and yellow, respectively.

ferent scenes. Hence, the votes are further illustrated based on scenes in Figure S4, Figure S5, Figure S6, and Figure S7. According to the resulting user votes, when being asked for which method produces a higher diversity of the view content, 80.0% participants agreed that *GAIT-DrQ-v2* outperforms CMA-ES and 73.3% favor *GAIT-DrQ-v2* rather than *GAIT-CURL*. On the other hand, 51.1%, 48.9%, and 38.9% users in the three scenes chose *GAIT-DrQ-v2* over *GAIT-CURL* as the method for generating smooth videos.

Since our experiment is in Two-alternative forced choice (2AFC), we analyze the user voting pattern using binomial test (reported in Table S6). According to the analysis, in view diversity in all three evaluated scenes, *GAIT-DrQ-v2* outperforms *CMA-ES* significantly in both the binomial test and the voting percentage. However, although majority of participants prefer *GAIT-DrQ-v2* over *GAIT-CURL* and *CMA-ES* in view diversity, it does not always demonstrate significance in the voting distribution in different scenes. In content aesthetics evaluation, there is no significance between *CMA-ES* and *GAIT-DrQ-v2*. It is possible that different participants' subjective metrics regarding aesthetics vary greatly. In the overall rating, the model *GAIT-DrQ-v2* gains most of the participants' preference.

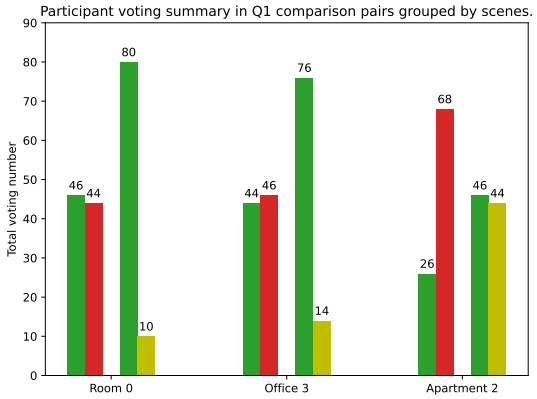


Figure S4: User votes in $Q1$ per scene. In each scene, the first and second comparisons represent $Cvs.D$ and $Mvs.D$. Green, red, and yellow bars refer to votes for Method D, C, and M, respectively.

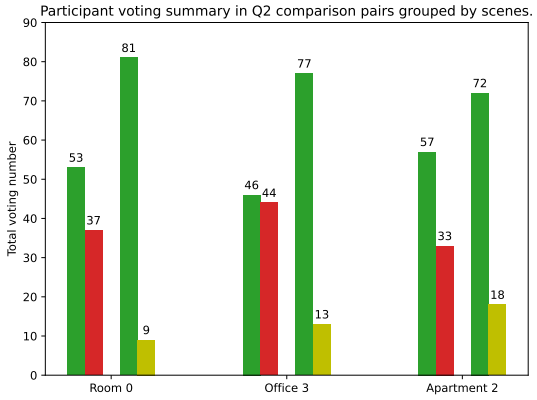


Figure S5: User votes in $Q2$ per scene. In each scene, the first and second comparisons represent $Cvs.D$ and $Mvs.D$. Green, red, and yellow bars refer to votes for Method D, C, and M, respectively.

Question	Specific Statement	When To Ask	Total Sample Number per User
Q1	"Which tour do you think is better regarding video smoothness?"	Each tour pair	54
Q2	"Which tour do you think provides better aesthetic content?"	Each tour pair	54
Q3	"After watching the three tours generated using the two methods, which method do you think provides higher view diversity in the tour?"	Each initial pose	18
Q4	"After watching the three tours generated using the two methods respectively, overall which method do you think can provide better aesthetic tours?"	Each initial pose	18

Table S5: Our user study questionnaire. Q1 and Q2 were asked after users watch every pair of tours. Q3 and Q4 were asked after watching the three generated sequences with the exclusion setting.

Scene name	C vs. D Smooth-ness	C vs. D Content	M vs. D Smooth-ness	M vs. D Content	C vs. D View Diversity	M vs. D View Diversity	C vs. D Overall	M vs. D Overall
Room 0	0.9161	0.1133	**0.0161**	0.3616	**0.0000**	**0.0000**	**5.948e-5**	**0.0000**
Office 3	0.9161	0.9161	0.5847	0.8555	**0.0000**	**0.0000**	0.3616	**0.0000**
Apartment 2	**7.657e-5**	**0.0149**	**0.0000**	0.0987	0.9161	**0.0000**	**0.0000**	0.0987

Table S6: Binomial test in the comparisons in each scene. The table shows each p-value in Binomial test for all the pair comparisons in each scene. The significant p-values (p value is smaller than 0.05) are highlighted.

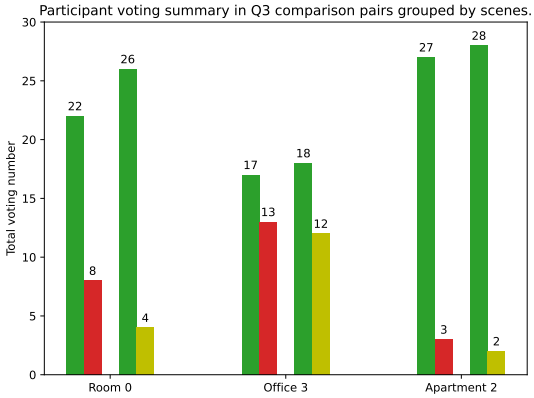


Figure S6: User votes in $Q3$ per scene. In each scene, the first and second comparisons represent $Cvs.D$ and $Mvs.D$. Green, red, and yellow bars refer to votes for Method D, C, and M, respectively.

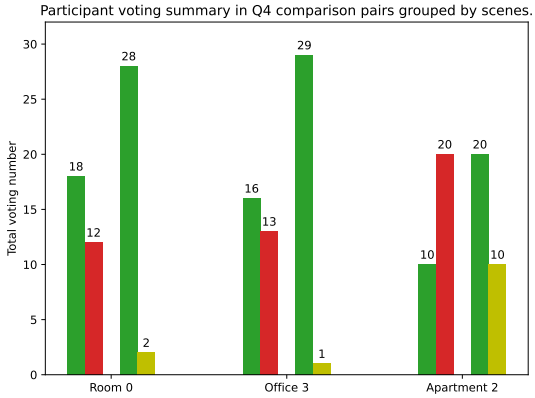


Figure S7: User votes in $Q4$ per scene. In each scene, the first and second comparisons represent $Cvs.D$ and $Mvs.D$. Green, red, and yellow bars refer to votes for Method D, C, and M, respectively.

Table S7: **Aesthetic and smoothness scores for Figure 4 in main text .** The cells in this table correspond to the frames in Figure 4 in main text. The values in each cell are the aesthetic score and smoothness score separated by a bar |. Seq. 0 and 1 shows similar aesthetic and smoothness scores over all steps. Although started with different initial poses, the 3 sequences all move to views with high aesthetic scores and keep a decent smoothness score after step 3.

$r^{\mathcal{A}}\|r^{\mathcal{S}}$	Initial pose	Time step 3	Time step 6	Time step 9	Time step 12	Time step 15
Seq. 0	N/A	1.04|0.22	4.61|0.78	5.41|0.60	4.86|0.90	4.64|0.82
Seq. 1	N/A	−0.30|0.21	4.32|0.80	5.25|0.58	4.52|0.89	4.40|0.83
Seq. 2	N/A	0.57|0.31	2.22|0.42	5.70|0.84	3.86|0.85	4.76|0.84

Table S8: **Aesthetic, Smoothness, and diversity scores for Figure 6 in main text .** The cells in this table correspond to the frames in Figure 6 in main text. The values in each cell are the aesthetic score, diversity score, and smoothness score separated by bars |. seq. 0 keeps a diversity score of 1 since there is no exclusion pose. seq. 1 also keeps a diversity score of 1 except for a 0.99, as it is able to avoid the exclusion pose easily. seq. 2's manages to avoid the first exclusion pose, but gets a bit too close to the second exclusion pose because it also has to keep aesthetics and smoothness scores high. Eventually it manages to get further from exclusion pose.

$r^{\mathcal{A}}\|r^{\mathcal{D}}\|r^{\mathcal{S}}$	Initial pose	Time step 3	Time step 6	Time step 9	Time step 12	Time step 15
Seq. 0	N/A	4.13|1|0.32	6.24|1|0.59	5.71|1|0.63	4.69|1|0.94	3.67|1|0.59
Seq. 1	N/A	4.35|1|0.30	4.94|0.99|0.70	4.96|1|0.48	4.65|1|0.92	4.89|1|0.51
Seq. 2	N/A	3.94|1|0.32	4.66|0.94|0.77	4.59|0.84|0.75	3.83|0.27|0.31	3.93|1|0.90

Table S9: **Aesthetic and smoothness scores for Figure 7 in main text .** The cells in this table correspond to the frames in Figure 7 in main text. The values in each cell are the aesthetic score and smoothness score separated by bars |. Without the smoothness constraint, seq. 1 converges to the near global optimal view and stays there to achieve maximum return. However, this results in a trivial aesthetic sequence. With the smoothness constraint, seq. 0 keeps a steady camera pose trajectory while maintaining a high aesthetics score and smoothness score.

$r^{\mathcal{A}}\|r^{\mathcal{S}}$	Initial pose	Time step 3	Time step 6	Time step 9	Time step 12	Time step 15
Seq. 0	N/A	2.07|0.21	5.66|0.70	4.91|0.68	4.10|0.85	4.89|0.94
Seq. 1	N/A	4.66|1	5.50|1	5.41|1	5.59|1	5.32|1

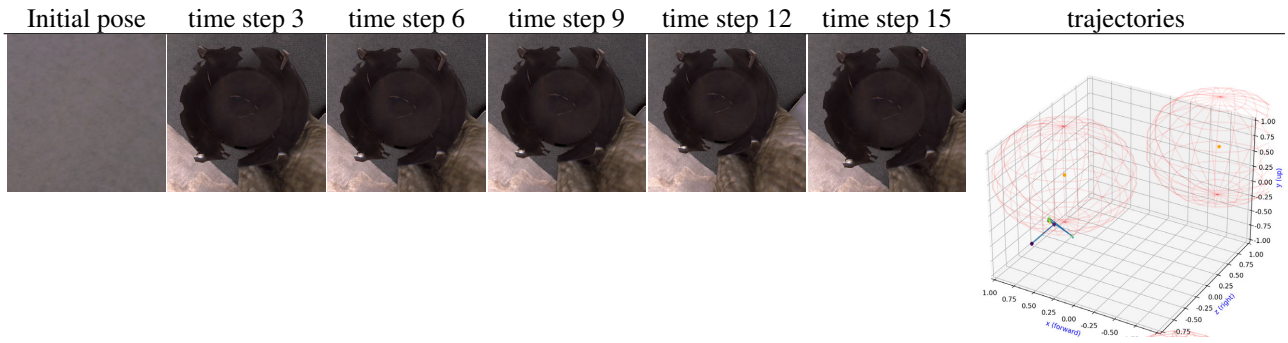
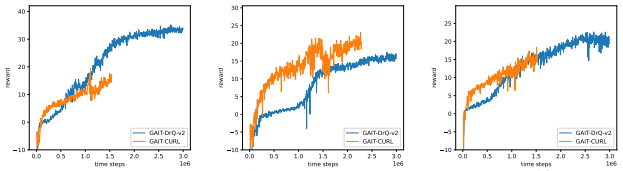
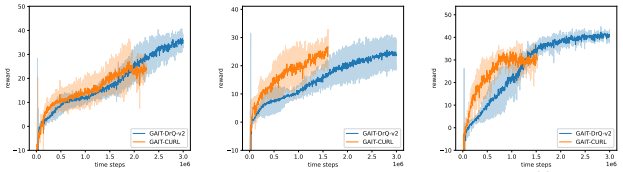


Figure S8: CMA-ES in Room0. It is stuck in this view next to the exclusion pose. This is because it takes greedy actions, but in this case, the exclusion pose gives a portion of possible actions low diversity score, while the rest of the actions all give a lower reward than staying at this view by taking small actions to satisfy the smoothness constraint.



(a) Room1 (b) Apartment0 (c) Office0

Figure S9: The training in the three additional scenes.



(a) Room0 (b) Apartment2 (c) Office3

Figure S10: The distribution of training. The average and the standard deviation are evaluated by 3 runs of training with different random seeds.

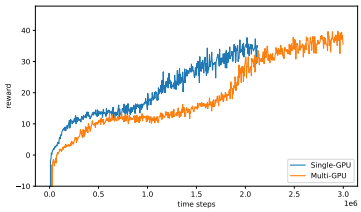


Figure S11: The training with different number of GPUs. There are 8 GPUs used for multi-GPU training. Multi-GPU converges slower than single-GPU training at the beginning, but the difference is very minor after 2 million time steps.

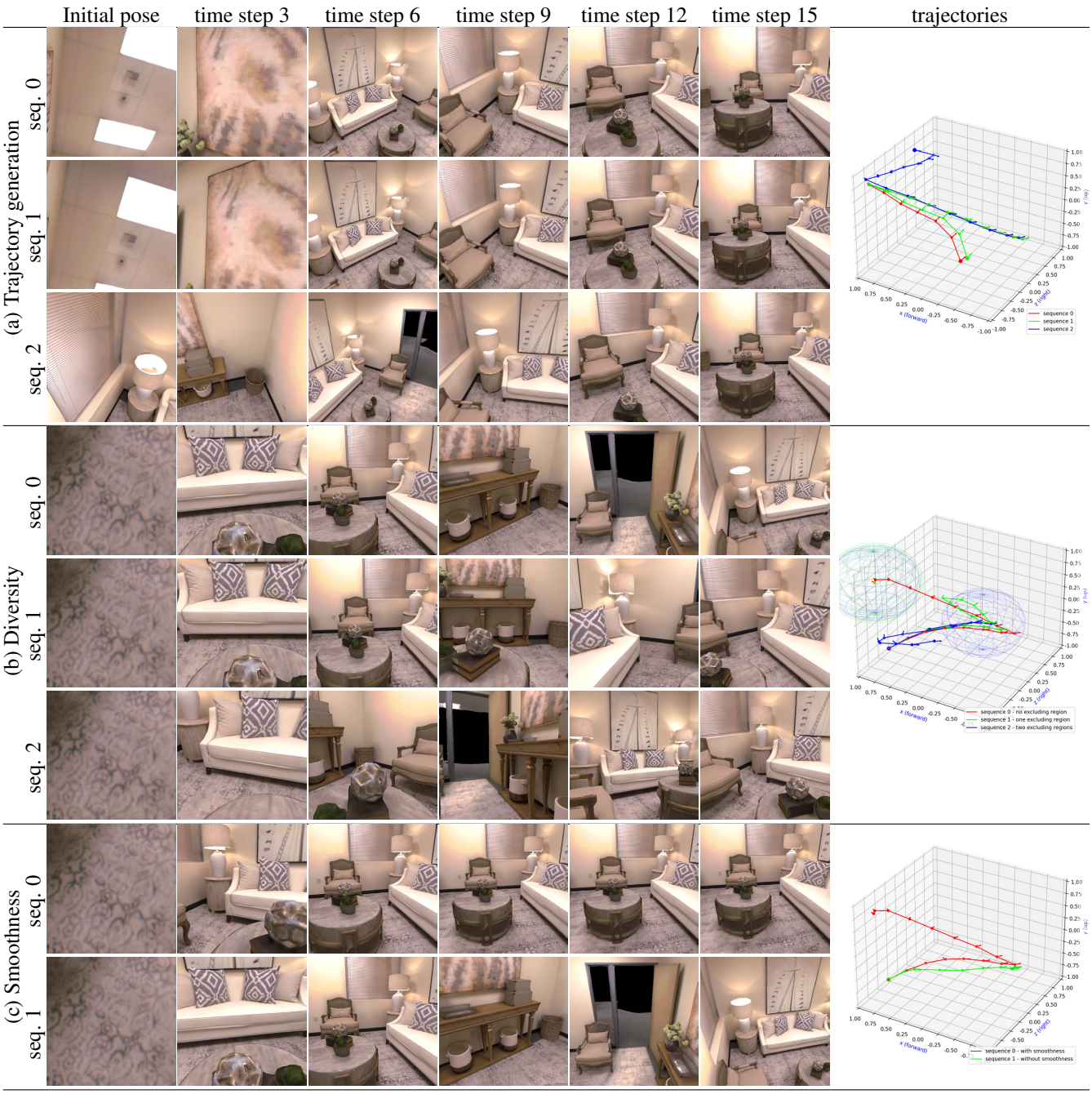


Figure S12: The camera pose trajectories generated by the agent trained with *GAIT-DrQ-v2* in Apartment2. (a) with different initial camera poses, the cameras are all transformed to a similar pose in the end. (b) comparison of diversity regularization, with 0, 1 and 2 exclusion regions for sequence 0, 1 and 2 respectively. (c) comparison of with and without temporal smoothness regularization for sequence 0 and 1 respectively.

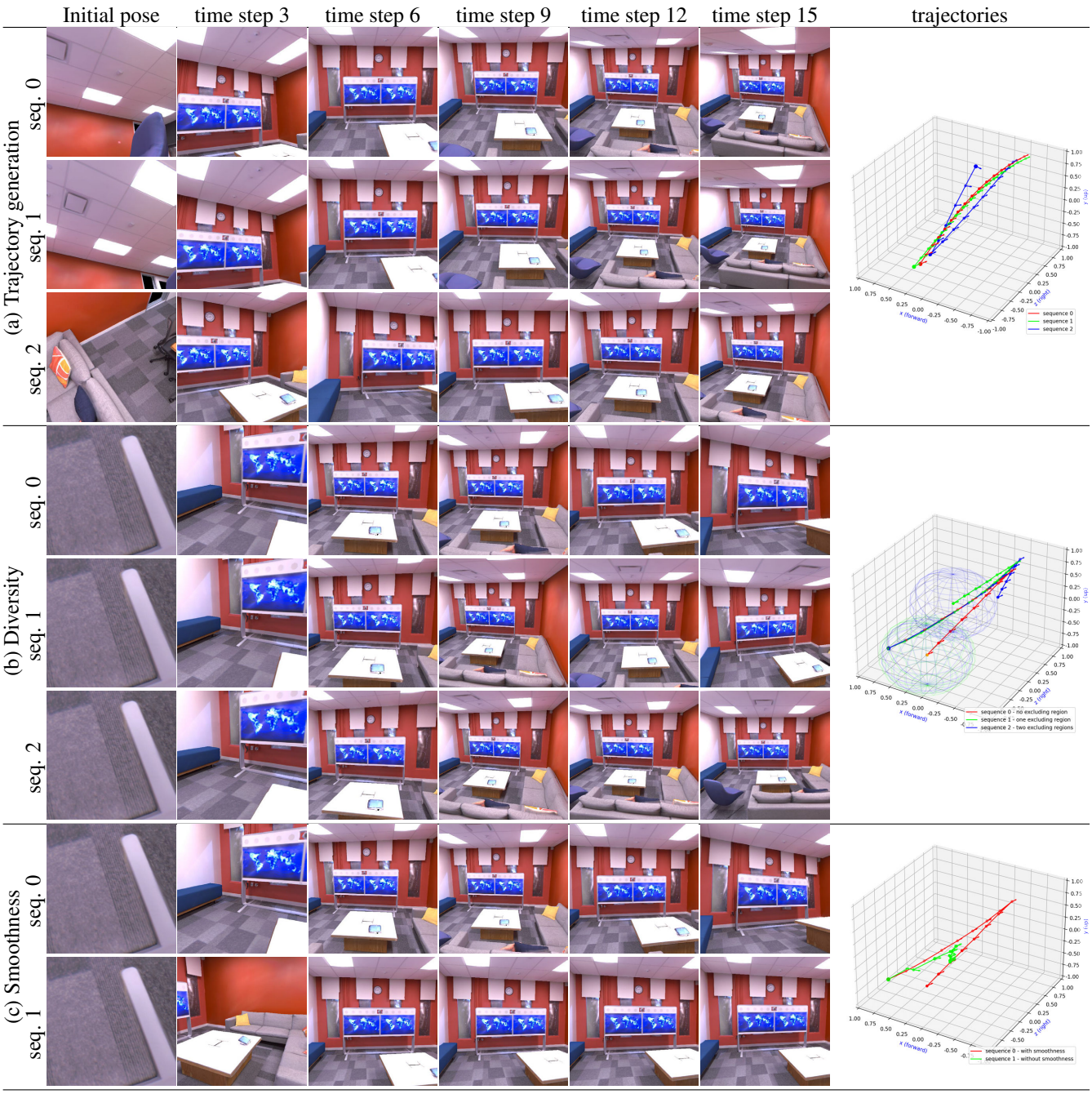


Figure S13: The camera pose trajectories generated by the agent trained with *GAIT-DrQ-v2* in Office3. (a) with different initial camera poses, the cameras are all transformed to a similar pose in the end. (b) comparison of diversity regularization, with 0, 1 and 2 exclusion regions for sequence 0, 1 and 2 respectively. (c) comparison of with and without temporal smoothness regularization for sequence 0 and 1 respectively.

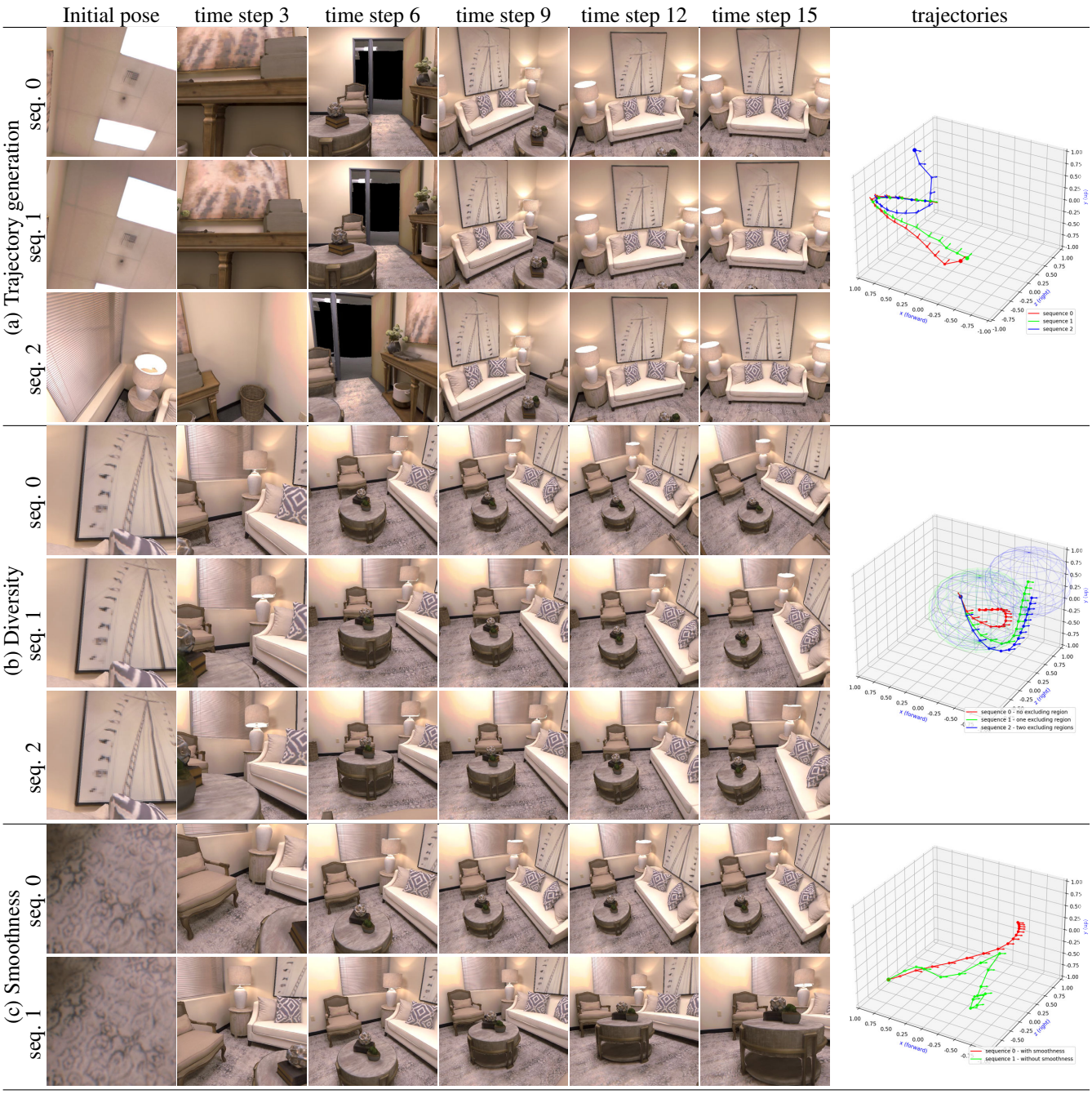


Figure S14: The camera pose trajectories generated by the agent trained with *GAIT-CURL* in Apartment2. (a) with different initial camera poses, the cameras are all transformed to a similar pose in the end. (b) comparison of diversity regularization, with 0, 1 and 2 exclusion regions for sequence 0, 1 and 2 respectively. (c) comparison of with and without temporal smoothness regularization for sequence 0 and 1 respectively.

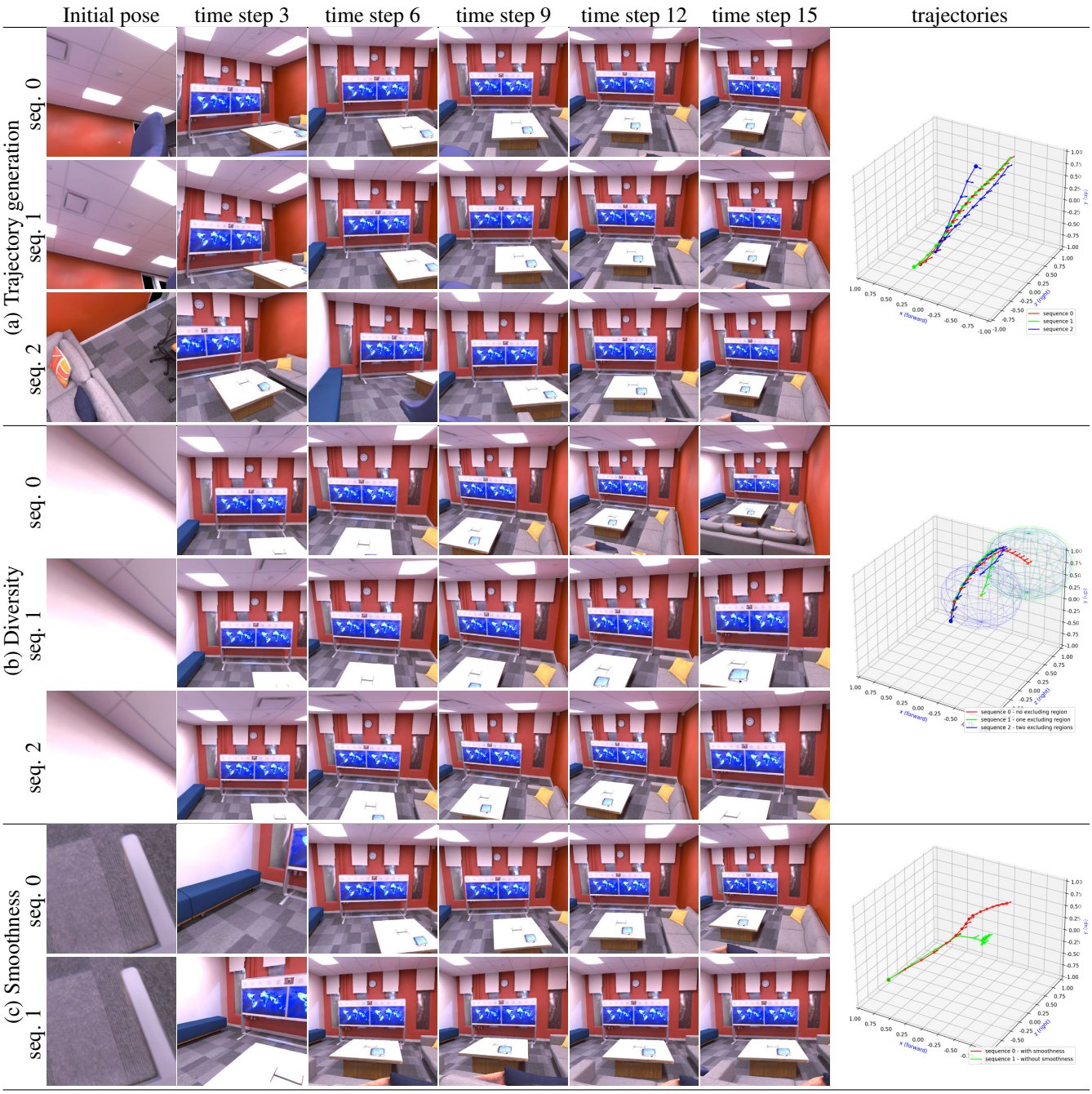


Figure S15: The camera pose trajectories generated by the agent trained with *GAIT-CURL* in Office3. (a) with different initial camera poses, the cameras are all transformed to a similar pose in the end. (b) comparison of diversity regularization, with 0, 1 and 2 exclusion regions for sequence 0, 1 and 2 respectively. (c) comparison of with and without temporal smoothness regularization for sequence 0 and 1 respectively.

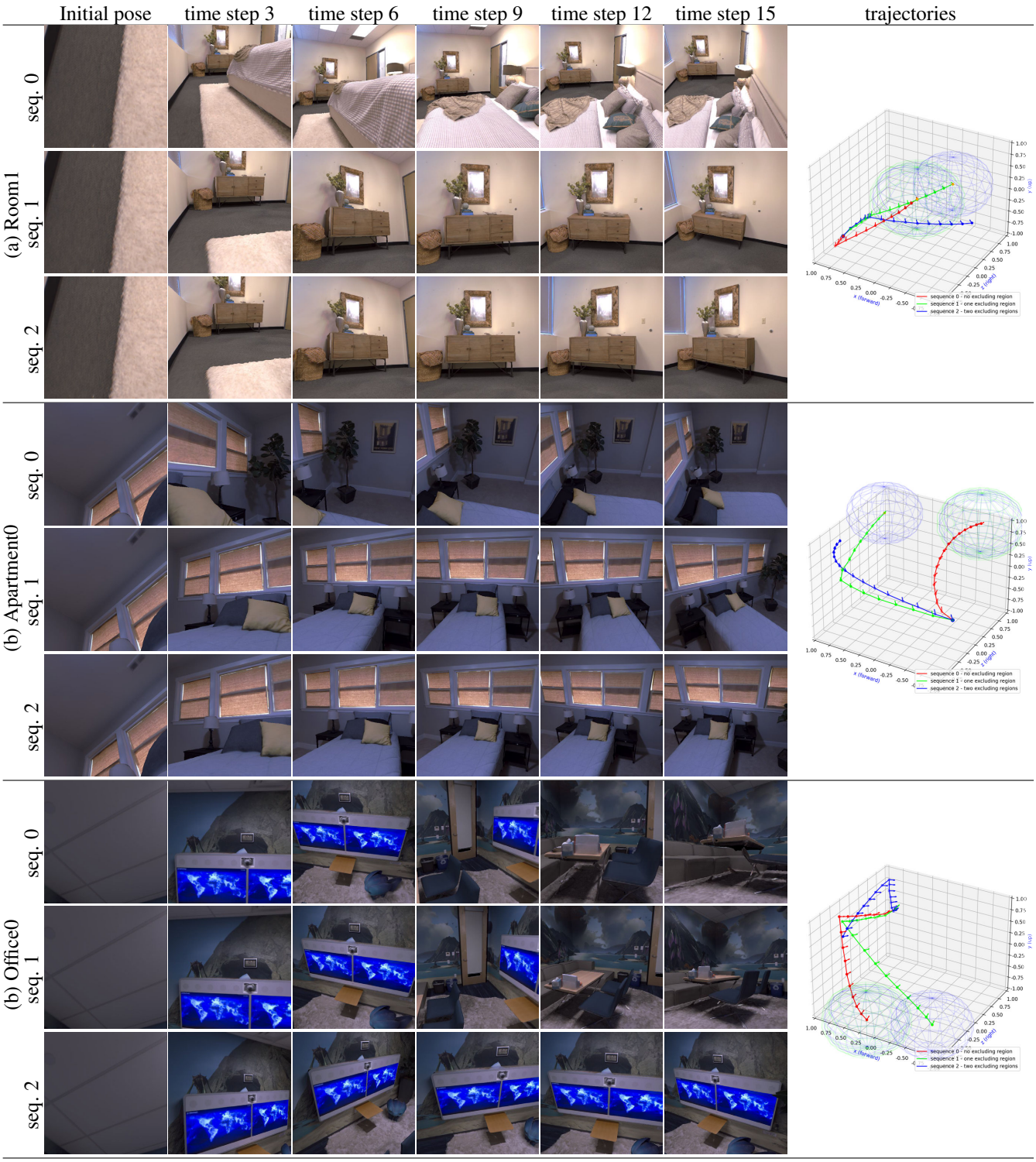


Figure S16: The camera pose trajectories generated by the agent trained with *GAIT-DrQ-v2* in the three additional scenes. In each sub-figure, the sequence 0, 1 and 2 are generated with 0, 1 and 2 exclusion regions respectively.

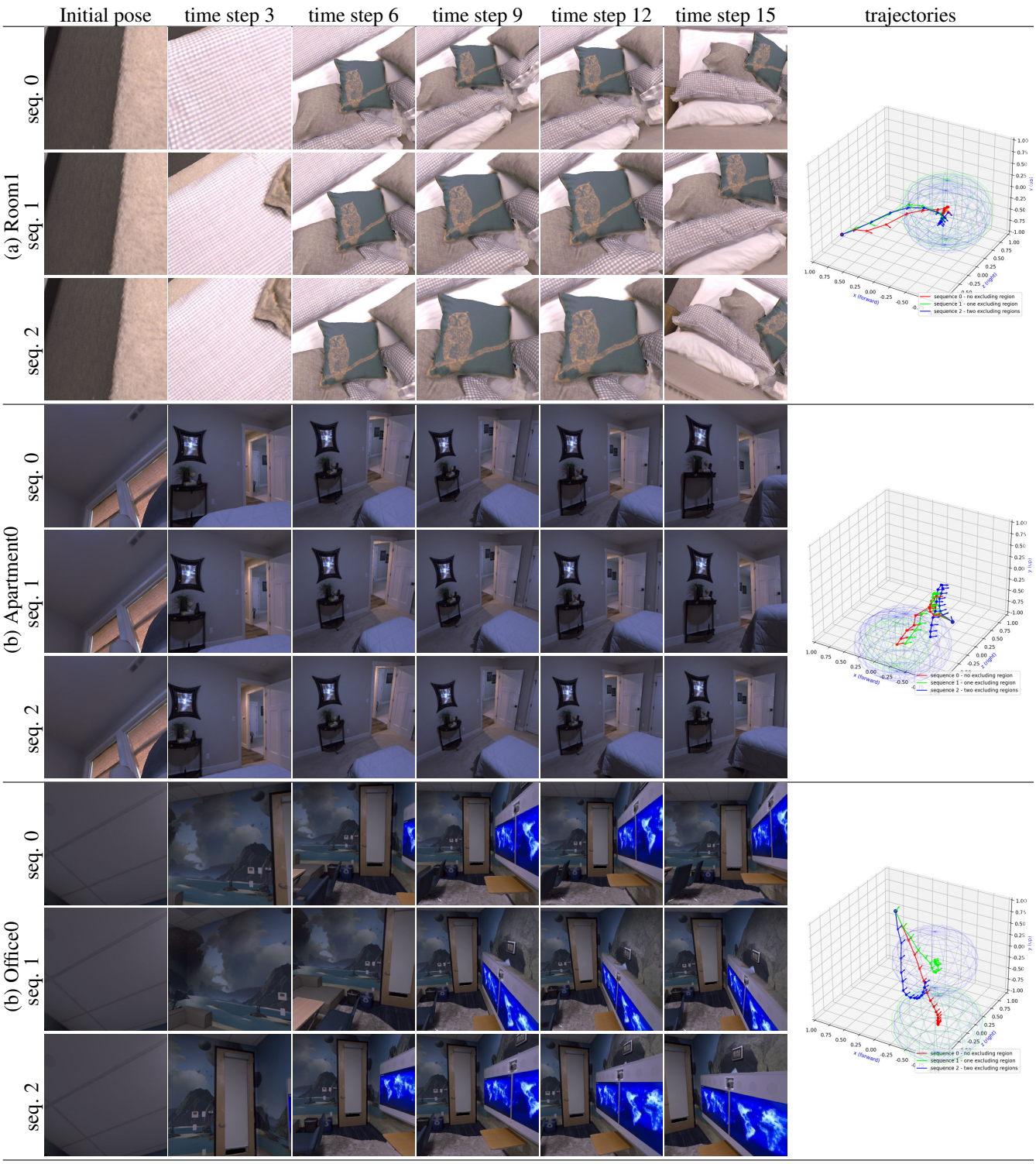


Figure S17: The camera pose trajectories generated by the agent trained with *GAIT-CURL* in the three additional scenes. In each sub-figure, the sequence 0, 1 and 2 are generated with 0, 1 and 2 exclusion regions respectively.

References

[1] Nikolaus Hansen, Youhei Akimoto, and Petr Baudis. CMA-ES/pycma on Github. Zenodo, DOI:10.5281/zenodo.2559634, Feb. 2019. 2

[2] Nikolaus Hansen and Andreas Ostermeier. Completely derandomized self-adaptation in evolution strategies. *Evolutionary Computation*, 9(2):159–195, 2001. 2

[3] Charles R. Harris, K. Jarrod Millman, Stéfan J. van der Walt, Ralf Gommers, Pauli Virtanen, David Cournapeau, Eric Wieser, Julian Taylor, Sebastian Berg, Nathaniel J. Smith, Robert Kern, Matti Picus, Stephan Hoyer, Marten H. van Kerkwijk, Matthew Brett, Allan Haldane, Jaime Fernández del Río, Mark Wiebe, Pearu Peterson, Pierre Gérard-Marchant, Kevin Sheppard, Tyler Reddy, Warren Weckesser, Hameer Abbasi, Christoph Gohlke, and Travis E. Oliphant. Array programming with NumPy. *Nature*, 585(7825):357–362, Sept. 2020. 2

[4] Mark Harris. How to optimize data transfers in cuda c/c++, 2012. 2

[5] 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. 1

[6] Thomas Hubert, Julian Schrittwieser, Ioannis Antonoglou, Mohammadamin Barekatain, Simon Schmitt, and David Silver. Learning and planning in complex action spaces, 2021. 3

[7] Michael Laskin, Aravind Srinivas, and Pieter Abbeel. Curl: Contrastive unsupervised representations for reinforcement learning. In *International Conference on Machine Learning*, pages 5639–5650. PMLR, 2020. 1, 2

[8] Philipp Moritz, Robert Nishihara, Stephanie Wang, Alexey Tumanov, Richard Liaw, Eric Liang, Melih Elibol, Zongheng Yang, William Paul, Michael I Jordan, et al. Ray: A distributed framework for emerging {AI} applications. In *13th {USENIX} Symposium on Operating Systems Design and Implementation ({OSDI} 18)*, pages 561–577, 2018. 2

[9] Ashvin V Nair, Vitchyr Pong, Murtaza Dalal, Shikhar Bahl, Steven Lin, and Sergey Levine. Visual reinforcement learning with imagined goals. *Advances in neural information processing systems*, 31, 2018. 1

[10] Rudy R Negenborn, Bart De Schutter, Marco A Wiering, and Hans Hellendoorn. Learning-based model predictive control for markov decision processes. *IFAC Proceedings Volumes*, 38:354–359, 2005. 2

[11] Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, et al. Pytorch: An imperative style, high-performance deep learning library. *Advances in neural information processing systems*, 32, 2019. 2

[12] PyTorch. Memory pinning. 2

[13] Manolis Savva, Abhishek Kadian, Oleksandr Maksymets, Yili Zhao, Erik Wijmans, Bhavana Jain, Julian Straub, Jia Liu, Vladlen Koltun, Jitendra Malik, Devi Parikh, and Dhruv Batra. Habitat: A Platform for Embodied AI Research. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, 2019. 3

[14] Karen Simonyan and Andrew Zisserman. Very deep convolutional networks for large-scale image recognition. *arXiv preprint arXiv:1409.1556*, 2014. 1

[15] Julian Straub, Thomas Whelan, Lingni Ma, Yufan Chen, Erik Wijmans, Simon Green, Jakob J. Engel, Raul Mur-Artal, Carl Ren, Shobhit Verma, Anton Clarkson, Mingfei Yan, Brian Budge, Yajie Yan, Xiaqing Pan, June Yon, Yuyang Zou, Kimberly Leon, Nigel Carter, Jesus Briales, Tyler Gillingham, Elias Mueggler, Luis Pesqueira, Manolis Savva, Dhruv Batra, Hauke M. Strasdat, Renzo De Nardi, Michael Goesele, Steven Lovegrove, and Richard Newcombe. The Replica dataset: A digital replica of indoor spaces. *arXiv preprint arXiv:1906.05797*, 2019. 3

[16] Andrew Szot, Alex Clegg, Eric Undersander, Erik Wijmans, Yili Zhao, John Turner, Noah Maestre, Mustafa Mukadam, Devendra Chaplot, Oleksandr Maksymets, Aaron Gokaslan, Vladimir Vondrus, Sameer Dharur, Franziska Meier, Wojciech Galuba, Angel Chang, Zsolt Kira, Vladlen Koltun, Jitendra Malik, Manolis Savva, and Dhruv Batra. Habitat 2.0: Training home assistants to rearrange their habitat. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2021. 3

[17] Che Wang, Xufang Luo, Keith Ross, and Dongsheng Li. Vrl3: A data-driven framework for visual deep reinforcement learning. *arXiv preprint arXiv:2202.10324*, 2022. 1

[18] Denis Yarats, Rob Fergus, Alessandro Lazaric, and Lerrel Pinto. Mastering visual continuous control: Improved data-augmented reinforcement learning. *arXiv preprint arXiv:2107.09645*, 2021. 1, 2

[19] Denis Yarats, Amy Zhang, Ilya Kostrikov, Brandon Amos, Joelle Pineau, and Rob Fergus. Improving sample efficiency in model-free reinforcement learning from images, 2019. 1