NavDP: Learning Sim-to-Real Navigation Diffusion Policy with Privileged Information Guidance

Anonymous Author(s)
Affiliation
Address
email

Abstract: Learning navigation in dynamic open-world environments is an important yet challenging skill for robots. Most previous methods rely on precise localization and mapping or learn from expensive real-world demonstrations. In this paper, we propose the Navigation Diffusion Policy (NavDP), an end-to-end framework trained solely in simulation and can zero-shot transfer to different embodiments in diverse real-world environments. The key ingredient of NavDP's network is the combination of diffusion-based trajectory generation and a critic function for trajectory selection, which are conditioned on only local observation tokens encoded from a shared policy transformer. Given the privileged information of the global environment in simulation, we scale up the demonstrations of good quality to train the diffusion policy and formulate the critic value function targets with contrastive negative samples. Our demonstration generation approach achieves about 2,500 trajectories/GPU per day, 20× more efficient than real-world data collection, and results in a large-scale navigation dataset with 363.2km trajectories across 1244 scenes. Trained with this simulation dataset, NavDP achieves state-of-the-art performance and consistently outstanding generalization capability on quadruped, wheeled, and humanoid robots in diverse indoor and outdoor environments. In addition, we present a preliminary attempt at using Gaussian Splatting to make in-domain real-to-sim fine-tuning to further bridge the sim-to-real gap. Experiments show that adding such real-to-sim data can improve the success rate by 30% without hurting its generalization capability.

Keywords: Robot Navigation, Diffusion Policy, Sim-to-Real, Cross-Embodiment

1 Introduction

Navigation in dynamic open-world is a fundamental yet challenging skill for robots. For pursuing embodied intelligent generalists, the navigation system is expected to be capable of zero-shot generalizing across different embodiment and unstructured scenes. However, the traditional modular-based methods suffer from system latency and compounding errors which limits their performance, while the scarcity of high-quality data limits the scale-up training and performance of learning-based methods. Although several studies try to address this problem by collecting robot trajectories in the real world [1, 2, 3], the scaling process is still time-consuming and expensive.

In contrast, simulation data is diverse and scalable. With large-scale 3D digital replica scenes available [4, 5, 6, 7, 8], we can efficiently generate customized infinite navigation trajectories with different types of observations and goals. Furthermore, with the increasing diversity of 3D assets and rapid progress of neural rendering algorithms, the long-standing sim-to-real gap problem can also be alleviated shortly. For learning a generalized navigation policy, imitation learning methods [9, 10] typically train the policy with demonstration trajectories but lack interaction and negative feedback from the environment. RL-based methods [11, 12] fully depend on interaction and reward function, but are often limited in learning efficiency.

Submitted to the 9th Conference on Robot Learning (CoRL 2025). Do not distribute.

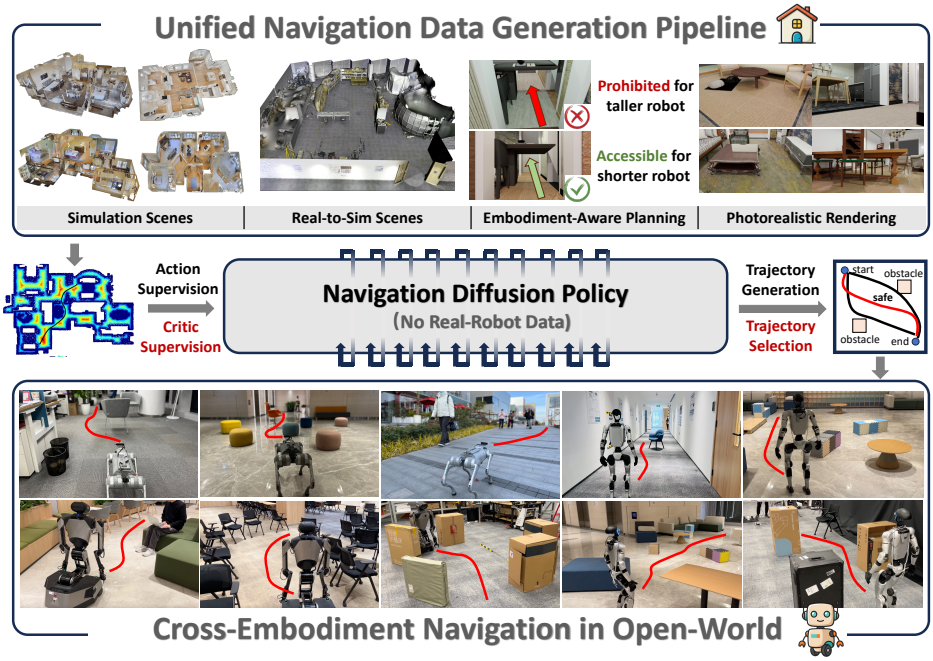

Figure 1: NavDP is solely trained with simulation trajectories but can achieve zero-shot sim-to-real transfer to different types of robots. By learning from the prioritized knowledge in the simulation data, NavDP adaptively selects a safe navigation routes towards the goal without any maps.

In this paper, we propose a new end-to-end transformer-based framework to combine the advantages of these two streams, **Nav**igation **D**iffusion **P**olicy **(NavDP)**, which achieves zero-shot sim-to-real policy transfer and cross-embodiment generalization with only simulation data. The NavDP network includes two stages at inference for trajectory generation and selection. It takes RGB-D images with navigation goal as input and fuses the encoded tokens with a policy transformer for diffusion-based trajectory generation. Further, the encoding of generated trajectories with RGB-D observation tokens are further fused with a shared policy transformer, where a critic head is then used to select a goal-agnostic safe navigation trajectory. This framework can fully take advantages of the privileged information in the simulation from two aspects: On the one hand, the trajectory generation head can be trained under the guidance from global-optimal planner within simulation environments. On the other hand, the critic function can learn spatial understanding from negative trajectories with the global Euclidean Signed Distance Field (ESDF) in simulation as a fine-grained guidance.

Our simulation navigation data generation approach achieves about 2,500 trajectories/GPU per day, 20$\times$ more efficient than real-world data collection, and results in a large-scale navigation dataset with 363.2km trajectories over 1244 scenes. Trained with this dataset, NavDP achieves zero-shot generalization capability on quadruped, wheeled, and humanoid robots in diverse indoor and outdoor environments and outperforms previous methods consistently in variant scenarios. In addition, given the observed sim-to-real gap in visual observations, we leverage the latest Gaussian Splatting approaches [13] to achieve the real-to-sim reconstruction and provide a more photorealistic environment for training and evaluation. With a preliminary attempt at building a real-to-sim lab, we make a study on training the policy with the combination of diverse simulation samples and real-to-sim samples. Our experiments show that adding 27% real-to-sim samples can improve the success rate in the target scene by 30% without hurting its generalization capability. The real-to-sim evaluation platform also shows its consistency with real-world evaluation, making it a promising pathway towards efficient and faithful benchmarking for navigation in the future.

2 Related Works

Robot Diffusion Policy. Advanced generative models have shown great potential in capturing multimodal distribution of robot policy learning. The diffusion policy [14] was the first to introduce

the diffusion process into manipulation tasks, sparking numerous efforts to enhance its capabilities. These enhancements span various aspects, including state representations [15, 16, 17, 18], inference speed [19, 20], and deployment across diverse robot applications [21, 22, 23, 24, 10]. However, as diffusion policies operate within an offline imitation learning framework, achieving strong real-world performance often depends on real-world teleoperation datasets, which are labor-intensive and challenging to scale. In contrast, our approach develops robot policies entirely from scalable simulation datasets. To enhance generalization and ensure safety during sim-to-real transfer, we introduce a critic function to estimate the safety of policy outputs. This mechanism leverages prioritized simulation data to enable the diffusion policy to better understand the consequences of actions, improving both safety and performance.

End-to-End Visual Navigation Models. Recent end-to-end visual navigation models have demonstrated significant potential in cross-embodiment adaptation and multi-task generalization [25, 26, 10, 27, 28, 29, 30, 11, 31, 32]. These approaches tackle navigation challenges at various levels of abstraction. Vision-Language-Action (VLA) models [30, 11, 31, 32, 33] offer flexibility by leveraging language instructions for task specification. In contrast, end-to-end navigation path planning models excel in cross-embodiment generalization and demonstrate superior adaptability with real-time inference in open-world environments [25, 26, 10]. In this paper, we focus on developing efficient end-to-end cross-embodiment navigation path planning models and our proposed method supports multiple types of input prompts, which can seamlessly attach to the VLA model and compensate for the VLA large models inference latency for the dynamic real-world scenarios.

Real-to-Sim for Sim-to-Real Transfer. Advances in 3D reconstruction [34, 35, 36, 37] have enabled the recovery of high-fidelity visual and physical properties of real-world environments within simulations, alleviating data scarcity for sim-to-real transfer in trained robot policies. The real-to-sim-to-real pipeline has proven effective in tasks like cable manipulation [38, 39]. For instance, RialTo [40] demonstrates that reinforcement learning (RL) in real-to-sim reconstructed scenes can significantly enhance real-world robot performance in manipulation tasks. Similarly, IKER [41] utilizes the real-to-sim-to-real pipeline to improve skill diversity with vision-language model (VLM)-guided reward functions, while ACDC [42] enhances policy generalization through randomized scene configurations in reconstructed environments. To the best of our knowledge, we are the first to adopt the real-to-sim-to-real pipeline for navigation tasks. And we demonstrate that pre-training on large-scale simulation datasets, augmented with a small amount of real-to-sim generated trajectories, can effectively bridge the embodiment gap and enhance the real-world performance.

3 Data Generation

We introduce the navigation data generation pipeline in this part, which composes of 1) robot model in simulation, 2) trajectory generation with global maps, 3) scene assets and simulation platform.

Robot Model. We build the robot as a cylindrical rigid body with two-wheel differential drive model for cross-embodiment generalizability. The navigation safe radius of the robots is set to $r_b = 0.25\text{m}$. To imitate the variation of the observation views of cross-embodiment robots, we assume one RGB-D camera is installed on the top of the robot and the height of the robot h_b is randomized in the range $(0.25\text{m}, 1.25\text{m})$. Therefore, the objects that are higher than the camera configuration height will not be consider as obstacles for navigation trajectory planning process. To ensure the local navigable area remains visible within the field of view, the camera's pitch angle is randomized in the range $(-30°, 0°)$, depending on the robot's height. The horizontal field of view (HFOV) and vertical field of view (VFOV) of camera are set to $(69°, 42°)$, same as the RealSense D435i camera.

Trajectory Generation. To generate collision-free robot navigation trajectories, we first convert the scene meshes into a voxel map with a voxel size of 0.05m to estimate the Euclidean Signed Distance Field (ESDF) of the navigable areas. Navigable areas are defined as voxel elements with z-axis coordinates below the threshold h_{nav}, while obstacle areas are defined as voxel elements with z-axis coordinates exceeding the threshold h_{obs}. The thresholds h_{nav} and h_{obs} vary across scenes and depend on the robot height h_b. Voxels with distance values lower than the robot radius r_b are truncated to prevent collisions. The ESDF map of the navigable area is downsampled to 0.2m resolution to facilitate

efficient A* path planning. Navigation start and target points are selected randomly on the navigable area, and the A* algorithm generates a planned path $\tau^* = [(x_0, y_0), (x_1, y_1), (x_2, y_2), \ldots, (x_k, y_k)]$. For each waypoint (x_n, y_n), a greedy search is performed in a local area of the original ESDF map to refine the position by maximizing the distance to nearby obstacles. This refinement process shifts waypoints further from obstacles. Finally, the refined waypoints are smoothed into a continuous navigation trajectory using cubic spline interpolation. Examples of the generated trajectories and global ESDF are shown in Appendix.

Scene Assets and Simulation. Following the pipeline described in the previous section, we can generate a large-scale dataset of robot navigation trajectories and corresponding camera movements across diverse scenes. To collect intermediate visual sensing data, we utilize BlenderProc [43] to render photorealistic RGB and depth images. The global poses of the cameras and robot base are stored as the navigation action labels. We collect navigation trajectories from over 1200 scenes selected from 3D-Front [6] and Matterport3D [4]. For each scene, we sample 100 pairs of starting points and destinations. In total, our dataset comprises over 56,000 trajectories and 10 million rendered RGB-D images, covering more than 360 kilometers of navigation distance. To increase the dataset diversity, we introduce texture randomization on the walls, floors and doors as well as light randomization during the rendering process. A detailed comparison of dataset scales is shown in Table 1. Our navigation dataset, generated through a highly efficient scripted pipeline and simulation framework which generates the data at the speed of 2.5k trajectories/GPU per day, thus significantly surpasses real-world teleoperated navigation datasets in scale.

Dataset	Scene	Distance(Km)	Trajectory(K)	Images.(M)	Collection Method
SCAND [2]	604	40	0.6	0.10	teleop
Go-Stanford [1]	27	16.7	3.7	0.17	teleop
HuRoN [3]	5	58.7	2.9	0.24	**scripted**
AMR [9]	54	-	**500**	7.5	**scripted**
NavDP(Ours)	**1244**	**363.2**	56	**10**	**scripted**

Table 1: Statistics of navigation datasets. Our efficient data generation pipeline enables a largest and most diverse navigation dataset.

4 Navigation Diffusion Policy

Our proposed NavDP consist of a diffusion head to capture the multi-modal distribution of navigation trajectories and a critic value function to select an optimal trajectory for safety. Details are illustrated in this section. Overview of the NavDP policy architecture is shown in Figure 2.

4.1 Model Architecture

Multi-modal Encoder. NavDP processes single-frame RGB-D images and navigation goals as input. To mitigate the sim-to-real gap in depth perception, depth values are clipped to (0.1m, 3.0m) for both training and inference. Depth data is encoded via a scratch Vision Transformer (ViT) [44], while RGB observations utilize a pretrained DepthAnything [45] ViT encoder, both producing 256 patch tokens. A transformer decoder compresses these 512 RGB-D tokens into 16 fused tokens. The system supports four zero-labeling-cost navigation goal types: (1) Point goal: relative coordinates on the 2D navigable plane; (2) Image goal: RGB observations from target locations; (3) Trajectory goal: preferred navigation trajectory projected onto the first-person view; (4) No goal: No specific goal is provided. The agent should try to roam in the environment without collision. Each goal type is encoded into a distinct token, which along with RGB-D fused tokens and a trajectory token forms the core input for downstream processing. Notably, all goal types can be automatically derived from raw navigation trajectories without manual annotation. The role of the trajectory token will be elaborated in subsequent sections.

Generative Diffusion Policy. Our diffusion policy head generates 24-step future waypoints by predicting a sequences of relative pose change $(\Delta x, \Delta y, \Delta\omega)$. NavDP employs the conditional U-

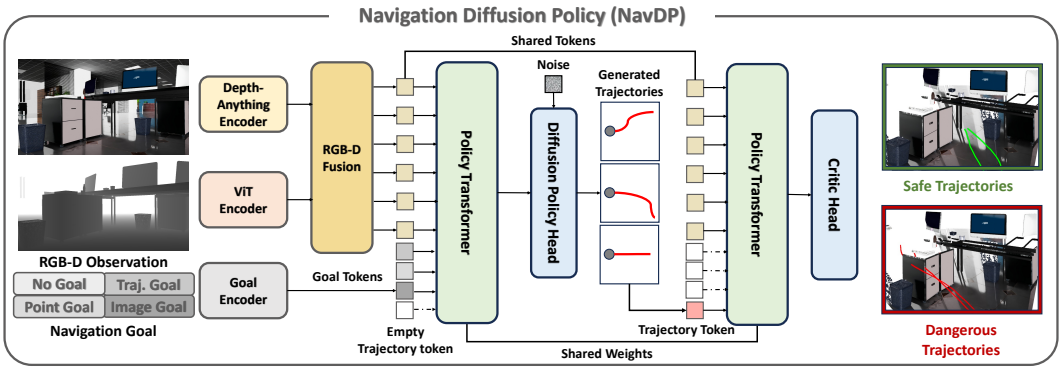

Figure 2: NavDP processes a single RGB-D observation frame along with a navigation goal. The inputs are tokenized and processed through a unified transformer architecture to generate navigation trajectories or evaluate corresponding trajectory values. A safe trajectory is then selected based on these values for execution by the robot.

Net [46] and DDPM scheduler [47] for the denoising process. To construct the conditional context for trajectory generation, we utilize a two-layer transformer encoder to process the input tokens. The input tokens are ordered as follows: The first 16 indices correspond to the RGB-D perception tokens and the next 3 tokens are used to represent the navigation goals and the final index is reserved as a placeholder for trajectory token. During the waypoints generation process, as no prior knowledge of trajectories is available, the policy transformer encoder does not attend to the trajectory token. To prevent training interference among multiple tasks, only one of the three tokens is attended to by the policy transformer layers, depending on the specific navigation task. No-goal navigation task attends neither of these 3 tokens. Finally, the fused tokens from the policy transformer at specific indices are provided as a global condition to the conditional U-Net for trajectory generation.

Critic Function. The diffusion policy is capable of generating multiple navigation trajectories. And constantly random selecting one trajectory may lead to collision because of the compounding prediction errors in the sequential decision-making problem. But in the real-world, ensuring the robot safety is of paramount importance. To address this, we propose a critic function that can universally evaluate the safety of any navigation trajectory without depending on any goals. The critic function head shares the same policy transformer weights and the RGB-D tokens as the diffusion policy head, but it does not attend to the tokens at the goal indexes. The trajectory token takes the last index. Trajectory token is encoded with 1D convolutional network. During inference, the generated batch of trajectories will be selected based on the predicted critic scores.

4.2 Model Training

Training Diffusion Policy with Multi-modal Goals. Both the navigation policy head and the critic head are trained using simulation data. To enhance data diversity during training, we augment the dataset by randomly sampling the trajectory's starting and ending points, using these sub-trajectories as the basic unit for policy training. The labeled actions are defined as the relative poses of intermediate frames with respect to the starting point. Consequently, the endpoint pose serves as the point goal, the corresponding RGB image captured at the endpoint defines the image goal, and the trajectory projected back onto the first-person view images forms the trajectory goal. Additionally, by masking out the goal information, we can treat the task as a no-goal scenario. These four task types collectively form a multi-task training objective for the diffusion policy head prediction. By adjusting the input mask of the policy transformer, the diffusion policy head receives varying conditions for action sequence generation. The mean squared error (MSE) loss between the predicted noise ϵ_θ^m under the four conditions and the label noise ϵ_k^m is used for backpropagation. Here, m represents the task index, and k denotes the denoising steps.

$$\mathcal{L}^{act} = \sum_{m=0,1,2,3} MSE(\epsilon_k^m, \epsilon_\theta^m(x_0 + \epsilon_k^m, k)) \tag{1}$$

Training Critic Function with the Global ESDF. We aim to build a critic function that distinguishes between safe and dangerous trajectories. However, the trajectories in the dataset consist entirely of perfect, collision-free actions, which are insufficient to form such knowledge. To address this, we apply simple yet effective data augmentation techniques. For each sub-trajectory used to train the diffusion policy head, we apply a random rotation to the original path. Denote the original path as τ_i and the rotated path as τ_i^r. We then randomly sample a weight β from the range $(0, 1)$, and the augmented trajectory is an interpolation between τ_i and τ_i^r, expressed as $\hat{\tau}_i = (1-\beta)\cdot\tau_i + \beta\cdot\tau_i^r$. During training, the augmented trajectory $\hat{\tau}_i$ is encoded and fed into the policy transformer for critic value prediction. Since the global ESDP map is available in the simulation dataset, it is straightforward to compute the distance to obstacles for each waypoint on the predicted trajectory. Denote the distance to the obstacle at the k-th waypoint on the augmented trajectory as $d_{\hat{\tau}}^k$. The critic value labels are then calculated as follows:

$$V(\hat{\tau}) = -\sum_{k=0}^{T} \mathbb{I}(d_{\hat{\tau}}^k < d_{safe}) + \alpha \sum_{k=0}^{T-1} (d_{\hat{\tau}}^{k+1} - d_{\hat{\tau}}^k) \tag{2}$$

We prefer the trajectory with more waypoints far from obstacles or own the trend of moving further from the obstacles. T is the trajectory prediction length, d_{safe} is a safe distance threshold and α is a re-weight hyperparameter. In default, the d_{safe} is set to 0.5m, and α is set to 0.1. Thus, the loss for the critic function is:

$$\mathcal{L}^{critic} = MSE(V(\hat{\tau}), V_\theta(I, D, \hat{\tau})) \tag{3}$$

5 Experiments

5.1 Evaluation and Metrics

We build the navigation evaluation benchmark based on IsaacSim which offers high-quality simulation of physics and reflects the potential sim-to-real gap in robot dynamics. Three functional scenarios (Hospital, Office, Warehouse) and three robot platforms (ClearPath Dingo, Unitree Go2, Galaxea R1) are considered for a comprehensive cross-embodiment generalization study. For evaluating the potential visual domain gap for sim-to-real transfer, we also build three real-to-sim scenes for evaluation. Details about evaluation scenarios is introduced in the Appendix. Two fundamental navigation tasks are considered in the evaluation: no-goal and point-goal navigation. In the no-goal task, we evaluate the safety and consistency of the navigation policy, thus two metrics **Episode Time** and **Explored Areas** are considered. Once collision occurs, the episode terminates and the maximum episode time is set to 120 seconds. In the point-goal task, we evaluate the path-planning accuracy and efficiency of the policy, thus two metrics **Success Rate** and **SPL** are considered. The episode is considered success if the robot arrives the area within a distance to goal lower than 1m and maintains a linear speed lower than 0.5m/s. For both tasks, we randomize the robot's spawn position across 100 different coordinates in each scene. For the point goal navigation task, we sample a point goal within the range of 3m to 15m from the spawn point.

NoGoal	**Sim Scene**						**Real-to-Sim Scene**					
	Dingo		**Unitree-Go2**		**Galaxea-R1**		**Dingo**		**Unitree-Go2**		**Galaxea-R1**	
Methods	Time(↑)	Area(↑)	Time(↑)	Area(↑)	Time(↑)	Area(↑)	Time(↑)	Area(↑)	Time(↑)	Area(↑)	Time(↑)	Area(↑)
GNM [25]	41.7	61.8	23.9	34.9	35.8	66.5	-	-	-	-	-	-
ViNT [26]	33.1	38.4	21.6	37.7	24.6	61.3	-	-	-	-	-	-
NoMad (Finetune) [10]	43.7	79.4	32.7	36.6	33.3	93.8	-	-	-	-	-	
NoMad (Pretrain) [10]	61.5	119.2	18.2	36.9	20.1	57.9	33.5	60.2	20.5	58.7	22.3	49.6
Ours	**104.5**	**280.2**	**95.8**	**359.1**	**98.9**	**300.4**	**88.1**	**90.4**	**76.5**	**102.6**	**70.2**	**95.7**

Table 2: We use the no-goal task to evaluate the exploration task performance across recent learning-based navigation methods. We find it difficult for prior works to generalize to the environments with large domain gaps in visual conditions. However, our approach, with the privileged map guidance and the training of the critic value function, can safely operate in diverse environments.

PointGoal			Sim Scene						Real-to-Sim Scene					
			Dingo		Unitree-Go2		Galaxea-R1		Dingo		Unitree-Go2		Galaxea-R1	
Methods	mSR(↑)	mSPL(↑)	SR(↑)	SPL(↑)	SR(↑)	SPL	SR(↑)	SPL(↑)	SR(↑)	SPL(↑)	SR(↑)	SPL(↑)	SR(↑)	SPL(↑)
PointNav [48]	22.1	16.6	44.6	36.5	14.6	6.3	7.0	6.9	-	-	-	-	-	-
EgoPlanner [49]	64.7	54.6	**85.6**	**66.4**	53.3	48.6	55.3	48.8	-	-	-	-	-	-
iPlanner [27]	48.2	40.7	72.6	59.3	72.0	62.8	0.0	0.0	-	-	-	-	-	-
ViPlanner [28]	65.6	55.4	78.0	58.8	80.0	**67.9**	62.0	47.2	**70.0**	**65.8**	**55.6**	**52.8**	48.3	40.1
Ours	**70.4**	**58.6**	81.3	62.3	**83.0**	61.8	**75.0**	**50.2**	66.0	63.5	52.6	51.7	**64.6**	**62.1**

Table 3: We compare our NavDP with recent learning-based navigation approaches as well as a planning-based method. We discover that the prior learning-based method generalize poorly across different robot platform. And planning-based method suffers from imperfect trajectory-following error as well as mapping error.

5.2 Experiment Analysis

In this part, we try to answer the following questions with both quantitative and qualitative experiment results: **Q1:** How well does our proposed NavDP **generalize across different robot platforms? Q2:** Which component **contributes most to the superior performance** of the NavDP? **Q3:** Does the NavDP be able to achieve **zero-shot sim-to-real transfer** across different scenes? **Q4:** Does the long-standing challenge of the **sim-to-real transfer can be alleviated by the real-to-sim reconstruction? Q5:** What are the advantages of our NavDP over planning-based methods?

For **Q1**, we compare our NavDP with a variety of baseline methods for both navigation tasks. The baseline method includes both learning-based approaches [25, 26, 10, 27, 28] and planning-based approaches [49]. Details of baseline methods are introduced in Appendix. As shown in Table 2, in the no-goal navigation task, prior methods generalize poorly on different embodiments compared with NavDP. We empirically analyze the evaluation episodes and conclude two main reasons: Firstly, the prior diffusion-based approach (NoMaD [10]) cannot perform test-time trajectory selection with only local information as ours. As the diffusion policy models a distribution of the expert demonstration, the variance of the generated results introduce compounding errors during the decision process, which limits the safety. Secondly, our model efficiently utilizes the foundation model for perception, which accelerate the learning of downstream navigation task. This is concluded from the performance of the fine-tuning version of NoMad. Even if learning with RGB-D dataset same as NavDP, the NoMad cannot achieve satisfied performance across different embodiments. In the point-goal navigation task, prior RL-based approach trained in Habitat cannot generalize well with realistic robot motion, and only achieves 20% success. iPlanner performs well in Dingo and Go2 platform, but always fail to stop at the target location with Galaxea robot. Although ViPlanner can deal with cross-embodiment navigation task, but our NavDP achieves the best performance.

For **Q2**, we conduct detailed ablation studies on the the critic function and training task numbers. We found that the critic function is important as both auxiliary loss function for training and test-time selection for inference. This is concluded from the left sub-figure in Figure 4. Without training the critic value, the point-nav performance is worse than the policy only without critic inference. And the no-goal task training objectives is of most important for the overall collision avoidance behavior, as shown in the middle sub-figure in Figure 4.

For **Q3**, we deploy our trained NavDP policy without any fine-tuning on three real robots, which are Unitree Go2, Galaxea R1 and Unitree G1. We test our policy on both indoor and outdoor scenarios with dynamic pedestrain interference. Our policy can consistently generates and selects a safe navigation trajectory on different scenarios as shown in Figure 3. Although observation views, camera field of views, varying light conditions, the existence of motion blur dramatically make the observation images different from the training dataset, NavDP can still generalize well. More illustration demos can be found in the accompanying video.

For **Q4**, we reconstruct a real-world laboratory scene with Gaussian-Splatting and generate a small proportion of in-domain navigation data ($\sim$4k trajectories) following the same pipeline in Section

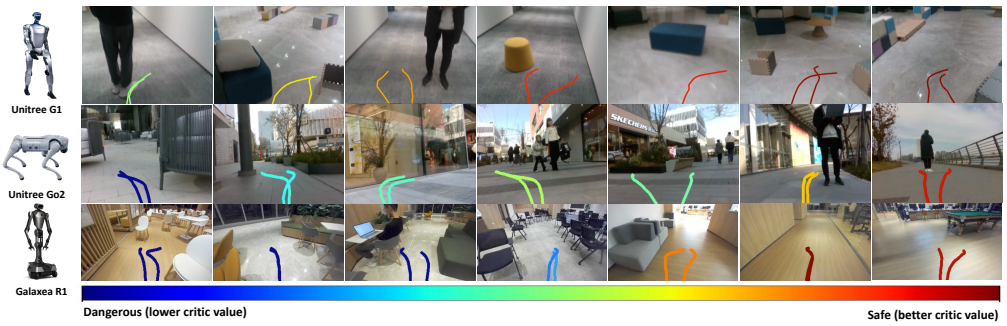

Figure 3: Trajectory visualization of on different robots. We project the predicted trajectories back to the image space and colorize them according to the corresponding critic values. The bluer trajectories indicate higher risk, whereas the redder trajectories represent safer paths.

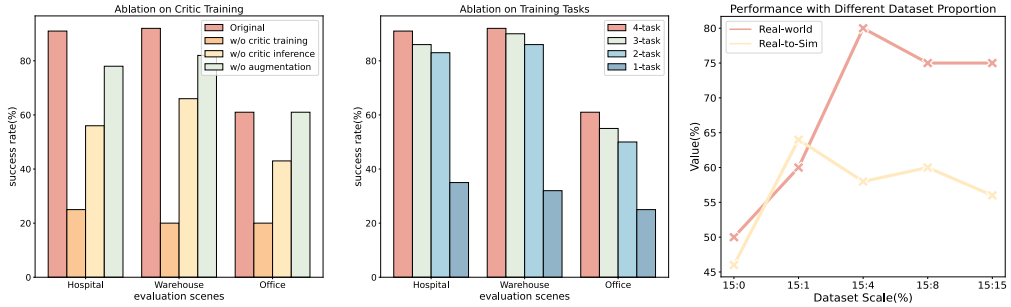

Figure 4: Ablation results for the NavDP. The left figure illustrate the entire NavDP network can benefit from critic function from test-time selection and training objectives. The middle illustrate the influence of using different tasks for training. The right illustrates the policy performance on both real-to-sim scenes and real-world scenes with respect to different data proportion.

3. We train the NavDP network with different proportion of the in-domain data. We control the proportion by fix the amount of simulation data and copy the real-to-sim data at different scale. We find that with a small proportion of in-domain data, the success rate in real-world evaluation increases from 50% to 80%, and from 45% to 65% in real-to-sim evaluation. But with a larger proportion of the real-to-sim data, the performance drops slightly. This hints that a real-to-sim reconstruction do improve the sim-to-real policy transfer, but a trade-off between diverse sim data and in-domain real-to-sim data should be carefully tuned.

For **Q5**, we evaluate the performance of a planning-based method - EgoPlanner [49] in the simulation point-goal navigation benchmark. We found it performs well on the wheeled robots, but can be easily stuck on the qrudruped robots. The reasons can be divided into two folds: The quadruped robot is driven by a locomotion policy trained with RL, the response of trajectory-following often delays and cause compounding erros. The camera of the Go2 robot often heads down and captures a restricted view for map updates, this influences the quality of planning trajectory. Further, our end-to-end policy can achieve real-time inference (>10Hz) with a GeForce RTX 5080 laptop, which enables fast collision avoidance in dynamic scenarios (shown in the accompanying video). The robot can operate at a maximum speed at 2.0m/s. And high-speed dynamic obstacle avoidance and navigation is quite challenging for the traditional map-based planning methods.

6 Conclusion

In this paper, we introduce a novel navigation diffusion policy (NavDP) that leverages a large-scale simulated navigation dataset and privileged simulation information to train a cross-embodiment sim-to-real navigation policy. The policy demonstrates efficient inference speed which enables path-planning and collision-avoidance ability under both static and dynamic scenarios. Three key ingredients contributes to the NavDP performance. The first is the critic function which serve as both training objective and test-time sample selection. The second is the use of multi-task training objectives. The third is the complementary usage of real-to-sim datasets.

Limitations

Our proposed NavDP has several limitations, which guide our future works: Firstly, the current version of NavDP doesn't support language instruction as a navigation goal, which limits the interaction ability between humans and robots. To that end, we will try to introduce additional vision-language navigation datasets and support NavDP training. Secondly, the current version of NavDP doesn't explicitly include embodiment encoding as network input. This makes accurate collision avoidance in a rather cluttered environment difficult. The policy may lead the camera to actively avoid the obstacles, but leave the body behind and cause a collision. Thirdly, our NavDP policy generates trajectory-level actions, which still depend on an extra locomotion policy for trajectory following. The decoupling of locomotion and navigation policy works well on the condition that a navigable path exists within the 2D plane. But in extremely complex scenarios where a navigable path only exists in the 3-D space, an end-to-end policy that can directly map the raw observations into joint control and distinguish the most affordable path. Building such a generalizable agile navigation policy will be one most important research topics for us in the future.

References

[1] N. Hirose, F. Xia, R. Martín-Martín, A. Sadeghian, and S. Savarese. Deep visual mpc-policy learning for navigation. *IEEE Robotics and Automation Letters*, 4(4):3184–3191, 2019.

[2] H. Karnan, A. Nair, X. Xiao, G. Warnell, S. Pirk, A. Toshev, J. Hart, J. Biswas, and P. Stone. Socially compliant navigation dataset (scand): A large-scale dataset of demonstrations for social navigation. *IEEE Robotics and Automation Letters*, 7(4):11807–11814, 2022.

[3] N. Hirose, D. Shah, A. Sridhar, and S. Levine. Sacson: Scalable autonomous control for social navigation. *IEEE Robotics and Automation Letters*, 2023.

[4] A. Chang, A. Dai, T. Funkhouser, M. Halber, M. Niessner, M. Savva, S. Song, A. Zeng, and Y. Zhang. Matterport3d: Learning from rgb-d data in indoor environments. *International Conference on 3D Vision (3DV)*, 2017.

[5] K. Yadav, R. Ramrakhya, S. K. Ramakrishnan, T. Gervet, J. Turner, A. Gokaslan, N. Maestre, A. X. Chang, D. Batra, M. Savva, et al. Habitat-matterport 3d semantics dataset. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 4927–4936, 2023.

[6] H. Fu, B. Cai, L. Gao, L.-X. Zhang, J. Wang, C. Li, Q. Zeng, C. Sun, R. Jia, B. Zhao, et al. 3d-front: 3d furnished rooms with layouts and semantics. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 10933–10942, 2021.

[7] M. Khanna, Y. Mao, H. Jiang, S. Haresh, B. Shacklett, D. Batra, A. Clegg, E. Undersander, A. X. Chang, and M. Savva. Habitat synthetic scenes dataset (hssd-200): An analysis of 3d scene scale and realism tradeoffs for objectgoal navigation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 16384–16393, 2024.

[8] H. Wang, J. Chen, W. Huang, Q. Ben, T. Wang, B. Mi, T. Huang, S. Zhao, Y. Chen, S. Yang, et al. Grutopia: Dream general robots in a city at scale. *arXiv preprint arXiv:2407.10943*, 2024.

[9] X. Meng, X. Yang, S. Jung, F. Ramos, S. S. Jujjavarapu, S. Paul, and D. Fox. Aim my robot: Precision local navigation to any object. *arXiv preprint arXiv:2411.14770*, 2024.

[10] A. Sridhar, D. Shah, C. Glossop, and S. Levine. Nomad: Goal masked diffusion policies for navigation and exploration. In *2024 IEEE International Conference on Robotics and Automation (ICRA)*, pages 63–70. IEEE, 2024.

[11] K.-H. Zeng, Z. Zhang, K. Ehsani, R. Hendrix, J. Salvador, A. Herrasti, R. Girshick, A. Kembhavi, and L. Weihs. Poliformer: Scaling on-policy rl with transformers results in masterful navigators. In *8th Annual Conference on Robot Learning*.

[12] A. Eftekhar, L. Weihs, R. Hendrix, E. Caglar, J. Salvador, A. Herrasti, W. Han, E. VanderBil, A. Kembhavi, A. Farhadi, et al. The one ring: a robotic indoor navigation generalist. *arXiv preprint arXiv:2412.14401*, 2024.

[13] T. Lu, M. Yu, L. Xu, Y. Xiangli, L. Wang, D. Lin, and B. Dai. Scaffold-gs: Structured 3d gaussians for view-adaptive rendering. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 20654–20664, 2024.

[14] C. Chi, S. Feng, Y. Du, Z. Xu, E. Cousineau, B. Burchfiel, and S. Song. Diffusion policy: Visuomotor policy learning via action diffusion. In *Proceedings of Robotics: Science and Systems (RSS)*, 2023.

[15] T.-W. Ke, N. Gkanatsios, and K. Fragkiadaki. 3d diffuser actor: Policy diffusion with 3d scene representations. *arXiv preprint arXiv:2402.10885*, 2024.

[16] X. Li, V. Belagali, J. Shang, and M. S. Ryoo. Crossway diffusion: Improving diffusion-based visuomotor policy via self-supervised learning. In *2024 IEEE International Conference on Robotics and Automation (ICRA)*, pages 16841–16849. IEEE, 2024.

[17] Y. Wang, Y. Zhang, M. Huo, R. Tian, X. Zhang, Y. Xie, C. Xu, P. Ji, W. Zhan, M. Ding, et al. Sparse diffusion policy: A sparse, reusable, and flexible policy for robot learning. *arXiv preprint arXiv:2407.01531*, 2024.

[18] Y. Ze, G. Zhang, K. Zhang, C. Hu, M. Wang, and H. Xu. 3d diffusion policy: Generalizable visuomotor policy learning via simple 3d representations. In *Proceedings of Robotics: Science and Systems (RSS)*, 2024.

[19] A. Prasad, K. Lin, J. Wu, L. Zhou, and J. Bohg. Consistency policy: Accelerated visuomotor policies via consistency distillation. *arXiv preprint arXiv:2405.07503*, 2024.

[20] Z. Wang, Z. Li, A. Mandlekar, Z. Xu, J. Fan, Y. Narang, L. Fan, Y. Zhu, Y. Balaji, M. Zhou, et al. One-step diffusion policy: Fast visuomotor policies via diffusion distillation. *arXiv preprint arXiv:2410.21257*, 2024.

[21] X. Huang, Y. Chi, R. Wang, Z. Li, X. B. Peng, S. Shao, B. Nikolic, and K. Sreenath. Diffuseloco: Real-time legged locomotion control with diffusion from offline datasets. In *8th Annual Conference on Robot Learning*, 2024. URL https://openreview.net/forum?id=nVJm2RdPDu.

[22] J. Zhang, M. Wu, and H. Dong. Generative category-level object pose estimation via diffusion models. In *Thirty-seventh Conference on Neural Information Processing Systems*, 2023. URL https://openreview.net/forum?id=l6ypbj6Nv5.

[23] M. Xu, Z. Xu, C. Chi, M. Veloso, and S. Song. Xskill: Cross embodiment skill discovery. In *Conference on Robot Learning*, pages 3536–3555. PMLR, 2023.

[24] C. Wang, H. Shi, W. Wang, R. Zhang, L. Fei-Fei, and C. K. Liu. Dexcap: Scalable and portable mocap data collection system for dexterous manipulation. *arXiv preprint arXiv:2403.07788*, 2024.

[25] D. Shah, A. Sridhar, A. Bhorkar, N. Hirose, and S. Levine. Gnm: A general navigation model to drive any robot. In *2023 IEEE International Conference on Robotics and Automation (ICRA)*, pages 7226–7233. IEEE, 2023.

[26] D. Shah, A. Sridhar, N. Dashora, K. Stachowicz, K. Black, N. Hirose, and S. Levine. Vint: A foundation model for visual navigation. *arXiv preprint arXiv:2306.14846*, 2023.

[27] F. Yang, C. Wang, C. Cadena, and M. Hutter. iplanner: Imperative path planning. *arXiv preprint arXiv:2302.11434*, 2023.

[28] P. Roth, J. Nubert, F. Yang, M. Mittal, and M. Hutter. Viplanner: Visual semantic imperative learning for local navigation. In *2024 IEEE International Conference on Robotics and Automation (ICRA)*, pages 5243–5249. IEEE, 2024.

[29] W. Cai, S. Huang, G. Cheng, Y. Long, P. Gao, C. Sun, and H. Dong. Bridging zero-shot object navigation and foundation models through pixel-guided navigation skill. In *2024 IEEE International Conference on Robotics and Automation (ICRA)*, pages 5228–5234. IEEE, 2024.

[30] K. Ehsani, T. Gupta, R. Hendrix, J. Salvador, L. Weihs, K.-H. Zeng, K. P. Singh, Y. Kim, W. Han, A. Herrasti, et al. Spoc: Imitating shortest paths in simulation enables effective navigation and manipulation in the real world. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 16238–16250, 2024.

[31] J. Zhang, K. Wang, R. Xu, G. Zhou, Y. Hong, X. Fang, Q. Wu, Z. Zhang, and H. Wang. Navid: Video-based vlm plans the next step for vision-and-language navigation. *Robotics: Science and Systems*, 2024.

[32] J. Zhang, K. Wang, S. Wang, M. Li, H. Liu, S. Wei, Z. Wang, Z. Zhang, and H. Wang. Uninavid: A video-based vision-language-action model for unifying embodied navigation tasks. *arXiv preprint arXiv:2412.06224*, 2024.

[33] A.-C. Cheng, Y. Ji, Z. Yang, X. Zou, J. Kautz, E. Bıyık, H. Yin, S. Liu, and X. Wang. Navila: Legged robot vision-language-action model for navigation. *arXiv preprint arXiv:2412.04453*, 2024.

[34] B. Kerbl, G. Kopanas, T. Leimkühler, and G. Drettakis. 3d gaussian splatting for real-time radiance field rendering. *ACM Trans. Graph.*, 42(4):139–1, 2023.

[35] A. Guédon and V. Lepetit. Sugar: Surface-aligned gaussian splatting for efficient 3d mesh reconstruction and high-quality mesh rendering. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 5354–5363, 2024.

[36] T. Lu, M. Yu, L. Xu, Y. Xiangli, L. Wang, D. Lin, and B. Dai. Scaffold-gs: Structured 3d gaussians for view-adaptive rendering. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 20654–20664, 2024.

[37] H. Matsuki, R. Murai, P. H. J. Kelly, and A. J. Davison. Gaussian Splatting SLAM. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2024.

[38] P. Chang and T. Padif. Sim2real2sim: Bridging the gap between simulation and real-world in flexible object manipulation. In *2020 Fourth IEEE International Conference on Robotic Computing (IRC)*, pages 56–62. IEEE, 2020.

[39] V. Lim, H. Huang, L. Y. Chen, J. Wang, J. Ichnowski, D. Seita, M. Laskey, and K. Goldberg. Real2sim2real: Self-supervised learning of physical single-step dynamic actions for planar robot casting. In *2022 International Conference on Robotics and Automation (ICRA)*, pages 8282–8289. IEEE, 2022.

[40] M. Torne, A. Simeonov, Z. Li, A. Chan, T. Chen, A. Gupta, and P. Agrawal. Reconciling reality through simulation: A real-to-sim-to-real approach for robust manipulation. *Arxiv*, 2024.

[41] S. Patel, X. Yin, W. Huang, S. Garg, H. Nayyeri, L. Fei-Fei, S. Lazebnik, and Y. Li. A real-to-sim-to-real approach to robotic manipulation with vlm-generated iterative keypoint rewards. In *2nd CoRL Workshop on Learning Effective Abstractions for Planning*.

[42] T. Dai, J. Wong, Y. Jiang, C. Wang, C. Gokmen, R. Zhang, J. Wu, and L. Fei-Fei. Automated creation of digital cousins for robust policy learning. *arXiv preprint arXiv:2410.07408*, 2024.

[43] M. Denninger, D. Winkelbauer, M. Sundermeyer, W. Boerdijk, M. Knauer, K. H. Strobl, M. Humt, and R. Triebel. Blenderproc2: A procedural pipeline for photorealistic rendering. *Journal of Open Source Software*, 8(82):4901, 2023. doi:10.21105/joss.04901. URL https://doi.org/10.21105/joss.04901.

[44] A. Dosovitskiy. An image is worth 16x16 words: Transformers for image recognition at scale. *arXiv preprint arXiv:2010.11929*, 2020.

[45] L. Yang, B. Kang, Z. Huang, X. Xu, J. Feng, and H. Zhao. Depth anything: Unleashing the power of large-scale unlabeled data. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 10371–10381, 2024.

[46] O. Ronneberger, P. Fischer, and T. Brox. U-net: Convolutional networks for biomedical image segmentation. In *Medical image computing and computer-assisted intervention–MICCAI 2015: 18th international conference, Munich, Germany, October 5-9, 2015, proceedings, part III 18*, pages 234–241. Springer, 2015.

[47] J. Ho, A. Jain, and P. Abbeel. Denoising diffusion probabilistic models. *Advances in neural information processing systems*, 33:6840–6851, 2020.

[48] N. Yokoyama, S. Ha, D. Batra, J. Wang, and B. Bucher. Vlfm: Vision-language frontier maps for zero-shot semantic navigation. In *International Conference on Robotics and Automation (ICRA)*, 2024.

[49] X. Zhou, Z. Wang, H. Ye, C. Xu, and F. Gao. Ego-planner: An esdf-free gradient-based local planner for quadrotors. *IEEE Robotics and Automation Letters*, 6(2):478–485, 2020.

[50] J. Xiang, Z. Lv, S. Xu, Y. Deng, R. Wang, B. Zhang, D. Chen, X. Tong, and J. Yang. Structured 3d latents for scalable and versatile 3d generation. *arXiv preprint arXiv:2412.01506*, 2024.

Appendix

Large-Scale Navigation Dataset in Simulation. To build a generalized navigation policy that can achieve zero-shot transfer in real-world scenarios, how to increase quality and diversity of the training dataset is one most important problem for mitigating the sim-to-real gap. To that end, we scale up the navigation trajectory dataset with both synthetic scene data and real-world scan scene data and introduce multiple domain randomization techniques. The main data source for synthetic scene data is 3D-Front, and we filter 1,176 scenes for data generation. The main data source for real-world scene data is Matterport3D, and we filter 68 scenes for data generation. Our pipeline supports domain randomization techniques includes light randomization, texture randomization and observation view randomization. Examples are shown in Figure 5. Our dataset supports cross-embodiment policy learning from two aspects: (1) The observation views varies when rendering the first-person-view images, this can mimic the captured images variance on different robot platforms. (2) The path-planning results depend on the observation view, for example, the taller robot is not allowed walk under the table, while a shorter robot can. This makes the policy formulates different navigation preference with respect to the observed images.

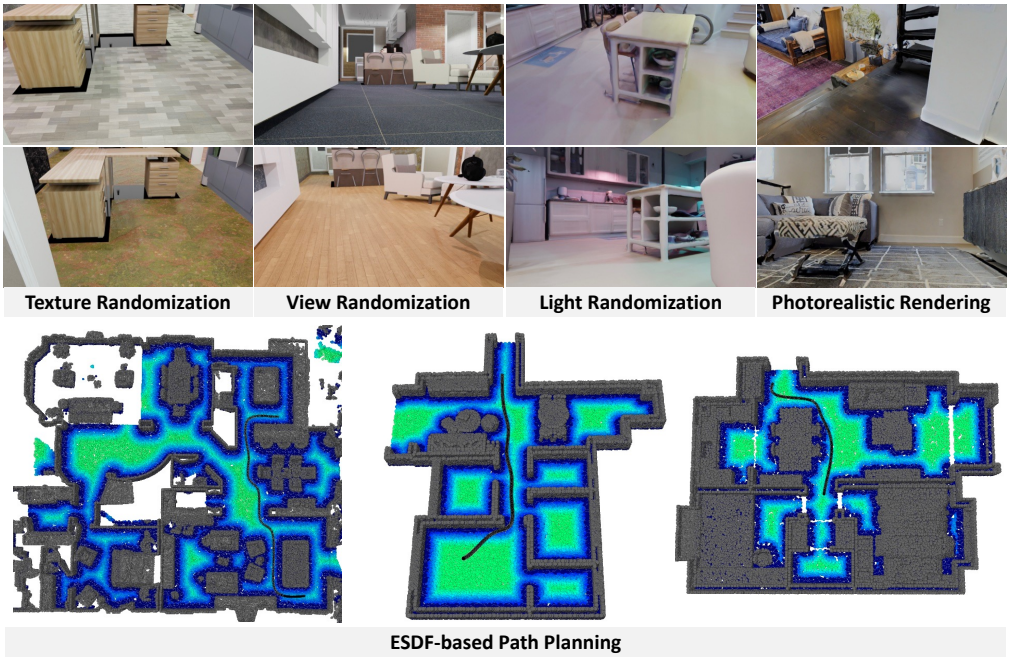

Figure 5: Examples of our simulation navigation dataset. Our dataset generation pipeline supports texture randomization, view randomization, light randomization and provide photorealistic rendering with BlenderProc.

Inference process of NavDP. During the inference, we convert the NavDP prediction trajectory into a feasible linear speed and angular speed as a unified action space for different robots. This is achieved by multiplying the middle point on the predicted trajectory with a proportional coefficient. Although this simple way cannot guarantee the robot strictly follows the predicted path, the robot can instantly adjust the path according to RGB-D frames as feedback, making it easy to deploy on different robot platforms. To get the best navigation trajectory, our methods follows a two-stage inference process. Firstly, based on the navigation goal and RGB-D frames, the NavDP generates a batch of trajectories with the diffusion policy head. Secondly, the NavDP critic function assign scores for each trajectory by receiving the trajectories and the same RGB-D frame, not conditioned on any navigation goal.

Evaluation Benchmark. For the evaluation with simulation scenes, we refer to three high-quality reconstructed scenes included in the IsaacSim (Hospital, Office and Warehouse). All scenes own realistic room layouts and rendering results. For the real-world evaluation, we place different type

of obstacles and compose three real-world scenarios as shown in Figure 6. We test each type robots with 20 episodes with different spawn points and report the metrics shown in Figure 4. For building a replica of the real-world scenarios, we first remove the obstacles and reconstruct the background with Scaffold-GS [36], and then capture first-person-view images for each real-world obstacles and reconstruct the 3-D structure with Trellis [50]. We manually adjust the reconstructed 3-D assets into the GS scene. And this real-to-sim scenes are used as additional evaluation platforms.

Figure 6: Visualization of the navigation evaluation benchmark. Simulation evaluation, Real-to-Sim evaluation as well as Real-world evaluation are conducted in this work.

Baseline Methods. In the no-goal navigation task, we evaluate three cross-embodiment navigation methods (GNM [25], ViNT [26], NoMad [10]) trained with real-world navigation trajectories. For the former two methods (GNM and ViNT), as they do not naturally support no-goal task, we use our dataset to fine-tune the network with introducing the goal masking technique same as NoMad. The fine-tuned weights are used to report the metrics. For the NoMad method, one version directly uses the pre-trained weights for no-goal task deployment and the metrics are reported as NoMad (pretrain). As all three are RGB-only methods, for a fair comparision, we introduce another baseline - NoMad (finetune), which adds a depth-branch and use our simulation data for fine-tuning. The depth branch encodes a single frame of depth image with efficient-net and all the tokens are fused with the subsequent transformer layers. The depth branch enables better performance on Go2 and Galaxea platform, but still achieves worse performance than ours. For the point-goal navigation task, we evaluated a discrete PointNav policy trained with Habitat-Sim, a mapping-based method EgoPlanner and two recent learning-based sim-to-real approaches, iPlanner [27], ViPlanner [28]. To make the discrete PointNav policy works in our continuous evaluation benchmark, we directly map the discrete action output into a pre-defined linear and angular speed. We find that the temporal shift in the continuous settings can dramatically interrupt the pre-trained RNN-based prediction results, and leading to large performance gap as in the Habitat platform. For the mapping-based EgoPlanner, we restrict the valid depth sensing range are $(0\text{m}, 10\text{m})$, and assume there are no localization errors. Therefore, on the Dingo wheeled robots, with an open-view and idea trajectory-following, the Ego-Planner achieves the best performance. But on the Galaxea-R1 and Unitree-Go2 platforms, with a delay response for trajectory-following and a heading down view, the performance decreases greatly. As for the iPlanner and ViPlanner, we directly use the pretrained weights to report the metrics.