# Prior Does Matter: Visual Navigation via Denoising Diffusion Bridge Models

Supplementary Material

## A. Details of DDBM in Imitation Learning

As shown in Section 4.1, the evolution of conditional probability of denoising diffusion bridfge models $q(\mathbf{a}_t|\mathbf{a}_T)$ has a time-reversed SDE of the form:

$$d\mathbf{a}_t = [f(\mathbf{a}_t,t) - g(t)^2(s(\mathbf{a}_t,t,\mathbf{a}_T,T) - h(\mathbf{a}_t,t,\mathbf{a}_T,T))]dt + g(t)d\bar{\mathbf{w}}_t, \tag{1}$$

with an associated probability flow ODE:

$$d\mathbf{a}_t = [f(\mathbf{a}_t,t) - g(t)^2(\frac{1}{2}s(\mathbf{a}_t,t,\mathbf{a}_T,T) - h(\mathbf{a}_t,t,\mathbf{a}_T,T))]dt + g(t), \tag{2}$$

on $t < T - \epsilon$ for any $\epsilon > 0$, where $\hat{\mathbf{w}}_t$ denotes a Wiener process.

The derivation of $q(\mathbf{a}_t|\mathbf{a}_0,\mathbf{a}_T)$ is shown at Section 4.1. Clearly defining each variable in the formula is necessary to complete the computation process. Following the definition of [2], the bridge processes generated by both VP and VE diffusion are in Table 1.

As mentioned in Section 4.1, with the reparametrization of Elucidating Diffusion Models, score function $\mathbf{s}(\mathbf{a},t,\mathbf{a}_T,T)$ can be expressed as:

$$\nabla_{\mathbf{a}_t}\log q(\mathbf{a}_t|\mathbf{a}_T) \approx \mathbf{s}(D_\theta,\mathbf{a}_t,t,\mathbf{a}_T,T) = \frac{\mathbf{a}_t - (\frac{SNR_T}{SNR_t}\frac{\alpha_t}{\alpha_T}\mathbf{a}_T + \alpha_t D_\theta(\mathbf{a}_t,t,\mathbf{a}_T)(1-\frac{SNR_T}{SNR_t}))}{\sigma_t^2(1-\frac{SNR_T}{SNR_t})}, \tag{3}$$

We following DDBM [2] to define the scaling functions and weighting function $w(t)$:

$$c_{\text{in}}(t) = \frac{1}{\sqrt{a_t^2\sigma_T^2 + b_t^2\sigma_0^2 + 2a_t b_t\sigma_0 T + c_t}} \tag{4}$$

$$c_{\text{skip}}(t) = \left(b_t\sigma_0^2 + a_t\sigma_0 T\right)\cdot c_{\text{in}}^2(t) \tag{5}$$

$$c_{\text{out}}(t) = \sqrt{a_t^2\left(\sigma_T^2\sigma_0^2 - \sigma_0 T^2\right) + \sigma_0^2 c_t}\cdot c_{\text{in}}(t) \tag{6}$$

$$w(t) = \frac{1}{c_{\text{out}}(t)^2} \tag{7}$$

$$c_{\text{noise}}(t) = \frac{1}{4}\log(t) \tag{8}$$

where $a_t = \frac{\alpha_t}{\alpha_T}\cdot\frac{\text{SNR}_T}{\text{SNR}_t}$, $b_t = \alpha_t\left(1-\frac{\text{SNR}_T}{\text{SNR}_t}\right)$, $c_t = \sigma_t^2\left(1-\frac{\text{SNR}_T}{\text{SNR}_t}\right)$.

Following EDM [1], the model output can be parameterized as:

$$D_\theta(\mathbf{a}_t,t) = c_{\text{skip}}(t)\mathbf{a}_t + c_{\text{out}}(t)F_\theta\big(c_{\text{in}}(t)\mathbf{a}_t, c_{\text{noise}}(t)\big), \tag{9}$$

where $F_\theta$ is a neural network with parameter $\theta$ that predicts $x_0$.

## B. Rule-based Prior Design and Derivation

### B.1. Rule-based Prior Design

The rule-based prior design employed in this work aims to generate an initial action that reflects plausible movement patterns based on the contextual features $\boldsymbol{c}_t$. To achieve this, the action space is divided into five distinct decision categories: moving straight, turning left, turning right, making a U-turn to the left, and making a U-turn to the right. This partitioning simplifies the prior generation process while ensuring comprehensive coverage of potential movement behaviors.

We utilize a fully connected layer to map the contextual vector $\boldsymbol{c}_t$ into a low-dimensional representation of action-relevant features to establish the prior. These features explicitly include the decision category and the predicted path length $d$. This mapping effectively encapsulates the essential elements required for generating an initial action, balancing the complexity of motion planning with computational efficiency.

Based on the decision category and path length derived from the contextual features, the initial direction and length of the path are determined. To introduce variability and prevent overly deterministic priors, noise is added to the angular direction and path length. This noise ensures diversity in the generated prior actions, enabling the model to explore a broader range of potential actions while maintaining adherence to the underlying behavioral patterns.

The initial action is then constructed following a parabolic path definition. The parabolic path is formulated such that it passes through the starting point and the determined endpoint. By leveraging the properties of parabolic curves, the path naturally accommodates smooth transitions and realistic motion patterns. We sample a fixed number of evenly spaced points along the parabola to represent this curve, providing a discrete sequence that constitutes the initial action.

This rule-based approach offers a robust and interpretable framework for generating prior actions, serving as a foundation for subsequent optimization and refinement. Integrating decision categories, noise, and parabolic curve modeling ensures that the priors are both expressive and adaptable, facilitating effective downstream action generation. Further details on the mathematical formulation of the parabolic path and noise modeling can be found in Section B.2.

|  | $\mathbf{f}(\mathbf{a}_t, t)$ | $g^2(t)$ | $p(\mathbf{a}_t \mid \mathbf{a}_0)$ | $\mathbf{SNR}_t$ | $\nabla_{\mathbf{a}_t} \log p(\mathbf{a}_T \mid \mathbf{a}_t)$ |
|---|---|---|---|---|---|
| VP | $\frac{d \log \alpha_t}{dt} \mathbf{a}_t$ | $\frac{d}{dt}\sigma_t^2 - 2\frac{d \log \alpha_t}{dt}\sigma_t^2$ | $\mathcal{N}(\alpha_t \mathbf{a}_0, \sigma_t^2 \mathbf{I})$ | $\frac{\alpha_t^2}{\sigma_t^2}$ | $\frac{\left(\frac{\alpha_t}{\alpha_T}\mathbf{a}_T - \mathbf{a}_t\right)}{\sigma_t^2(\text{SNR}_t/\text{SNR}_T - 1)}$ |
| VE | 0 | $\frac{d}{dt}\sigma_t^2$ | $\mathcal{N}(\mathbf{a}_0, \sigma_t^2 \mathbf{I})$ | $\frac{1}{\sigma_t^2}$ | $\frac{\mathbf{a}_T - \mathbf{a}_t}{\sigma_T^2 - \sigma_t^2}$ |

Table 1. VP and VE instantiations of diffusion bridges.

## B.2. Parabolic Constraint Condition Derivation

The desired path is modeled using a family of parabolas that meet the following constraints: the trajectory passes through the robot's origin and a specified endpoint. The initial direction aligns with the robot's forward orientation, transitioning smoothly to the endpoint in a parabolic arc. The parabola's vertex lies between the starting point and the endpoint, with its opening directed toward the robot's rear.

The parabolic constraint condition is derived based on the standard and vertex forms of a parabola, ensuring the curve passes through two fixed points $(x_1, y_1)$ and $(x_2, y_2)$, with control over its shape and orientation. The standard form of a parabolic equation is $y = ax^2 + bx + c$, where $a$, $b$, and $c$ are parameters. To uniquely define the parabola, three conditions are required. By incorporating the axis of symmetry $h$ as an additional control condition, a family of parabolas can be generated through the fixed points.

Alternatively, the vertex form, $y = a(x - h)^2 + k$, provides a parameterization where $h$ represents the axis of symmetry, $k$ denotes the vertical coordinate of the vertex, and $a$ determines the parabola's width and direction. Substituting the fixed points into the vertex form equations:

$$y_1 = a(x_1 - h)^2 + k \quad \text{and} \quad y_2 = a(x_2 - h)^2 + k, \tag{10}$$

the parameter $a$ can be derived by eliminating $k$:

$$a = \frac{y_2 - y_1}{(x_2 - x_1)(x_2 + x_1 - 2h)}. \tag{11}$$

Substituting $a$ back, the expression for $k$ becomes:

$$k = y_1 - a(x_1 - h)^2. \tag{12}$$

This formulation results in a family of parabolas passing through the fixed points, given as:

$$\begin{aligned} y = &\frac{y_2 - y_1}{(x_2 - x_1)(x_2 + x_1 - 2h)}(x - h)^2 + y_1 \\ &- \frac{y_2 - y_1}{(x_2 - x_1)(x_2 + x_1 - 2h)}(x_1 - h)^2 \end{aligned} \tag{13}$$

To ensure the parabola opens downward, the condition $a < 0$ must hold. Since $a = \frac{y_2 - y_1}{(x_2 - x_1)(x_2 + x_1 - 2h)}$, the numerator and denominator must have opposite signs. The numerator's sign depends on the relative magnitudes of $y_2$ and $y_1$, while the denominator's sign is determined by $(x_2 - x_1)(x_2 + x_1 - 2h)$.

For cases where $y_2 > y_1$, the numerator is positive, requiring the denominator to be negative. If $x_2 > x_1$, this implies $h > \frac{x_1 + x_2}{2}$. Conversely, if $x_2 < x_1$, $h < \frac{x_1 + x_2}{2}$. Similarly, when $y_2 < y_1$, the numerator is negative, necessitating a positive denominator. For $x_2 > x_1$, this requires $h < \frac{x_1 + x_2}{2}$, and for $x_2 < x_1$, $h > \frac{x_1 + x_2}{2}$. These relationships enable precise control over the parabola's orientation.

The position of the vertex relative to the fixed points is governed by the axis of symmetry $h$. To place the vertex between $x_1$ and $x_2$, $x_1 < h < x_2$ or $x_2 < h < x_1$ must hold. Conversely, for the vertex to lie outside this range, $h < \min(x_1, x_2)$ or $h > \max(x_1, x_2)$ is required. Adjusting $h$ accordingly provides flexibility in the parabola's configuration while satisfying the desired constraint conditions.

## B.3. Noise Modeling Description

In this framework, the standard deviation of Gaussian noise is adaptively adjusted based on prediction confidence. Higher confidence leads to lower noise, ensuring precision, while lower confidence increases noise to encourage exploration. The relationship is defined by:

$$\sigma = \text{min\_std} + (\text{max\_std} - \text{min\_std}) \cdot (1 - \text{confidence}), \tag{14}$$

where confidence ranges from 0 to 1. This linear interpolation maps confidence to a noise level between predefined minimum min_std and maximum max_std values.

The adaptive noise is applied to both angular direction and path length. For angular direction, the noise is centered on the midpoint of the decision interval, while for path length, it is centered on the predicted value, with variability scaled by the corresponding confidence.

## C. Detailed Analysis of Error Bound

The key goal is to bound the mean squared error (MSE), represented as $\mathbb{E}_{\mathbf{a}_t, \mathbf{a}_0}\left[\|\mathbf{a}_t - \mathbf{a}_0\|^2\right]$.

This error is driven by the difference between the source distribution $\pi_s(\mathbf{a}_T)$ and the target distribution $\pi(\mathbf{a}_0)$. The DDBM framework is designed to reduce this discrepancy

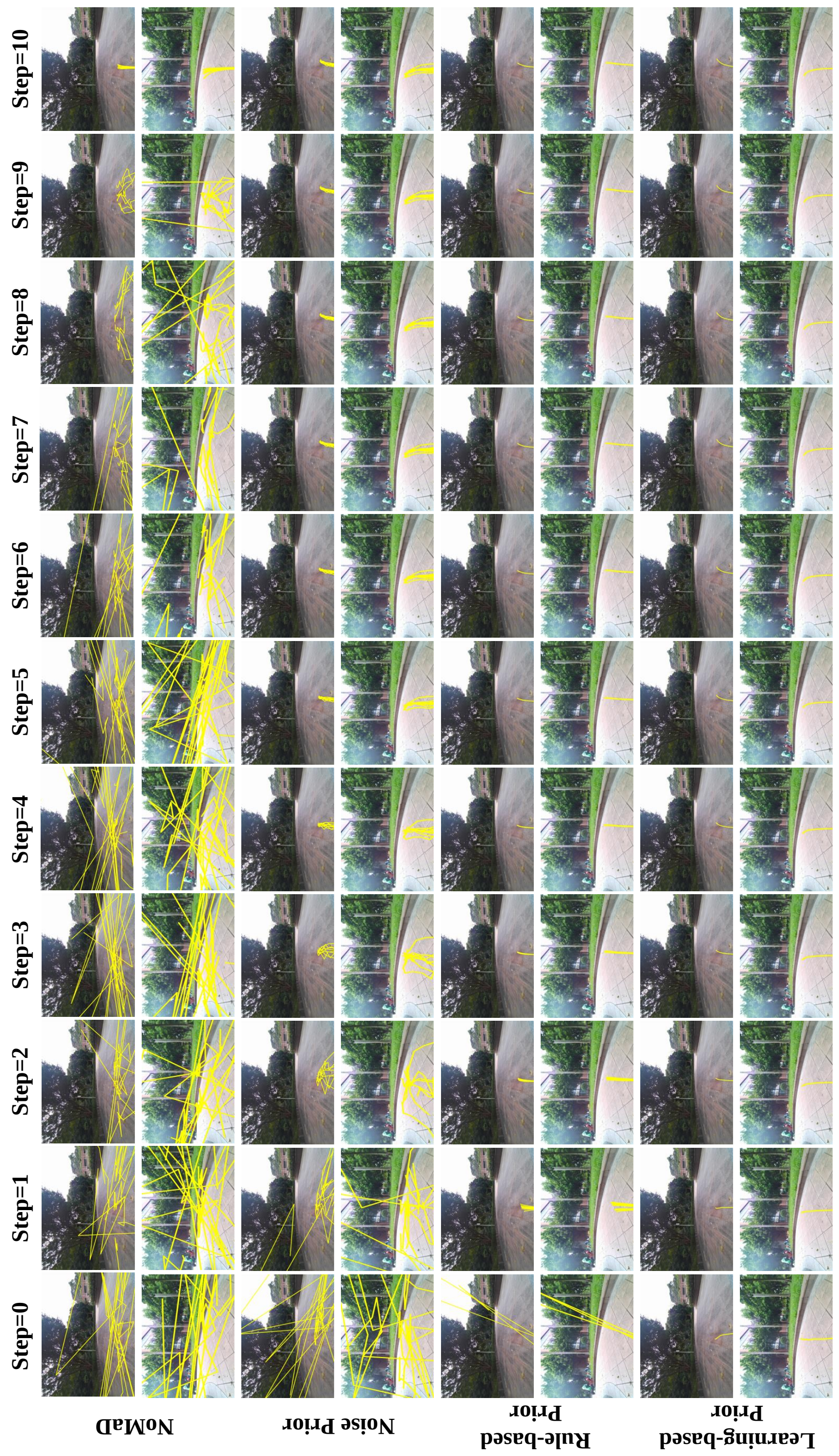

Figure 1. The figure visualizes feature correspondences over ten steps for four methods: NoMaD, Noise Prior, Rule-Based Prior, and Learning-Based Prior, in the context of on-device RGB observations. From top to bottom, the rows represent the progression of steps (Step 0 to Step 10). At the same time, from left to right, the columns correspond to the four method types—NoMaD, Noise Prior, Rule-Based Prior, and Learning-Based Prior—showcasing the impact of each prior on the feature-matching process across the sequence.

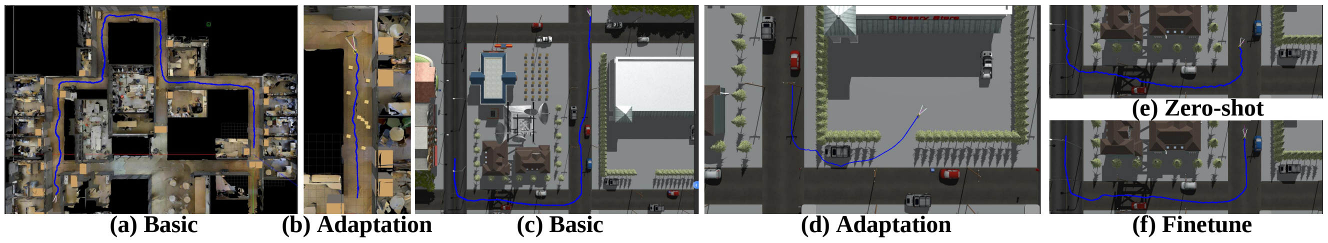

Figure 2. Simulation Tasks and Zero-Shot vs. Fine-Tuned Model

through a diffusion process and a drift term in its probability flow ODE.

The initial difference between the source and target distributions is fundamental to the bound. As shown in Section 4.2, we represent this discrepancy using the Kullback-Leibler (KL) divergence:

$$D_{\mathrm{KL}}\left(\pi_s(\mathbf{a}_T)\|\pi(\mathbf{a}_0)\right) = \int \pi_s(\mathbf{a}_T)\log\frac{\pi_s(\mathbf{a}_T)}{\pi(\mathbf{a}_0)}d\mathbf{a}_T. \quad (15)$$

This KL divergence measures how much information is ”lost” if we approximate $\pi(\mathbf{a}_0)$ by $\pi_s(\mathbf{a}_T)$. A smaller divergence indicates that the source and target distributions are close, implying that a smaller amount of drift and noise adjustment will be necessary for the DDBM to match the target.

The evolution of samples in a DDBM is controlled by a **probability flow ODE** that includes a **drift term** and a **noise term**. The ODE has the form which is equal to Equation 2:

$$\frac{d\mathbf{a}_t}{dt} = f(\mathbf{a}_t,t) - g^2(t)\left(\frac{1}{2}\nabla_{\mathbf{a}_t}\log\pi(\mathbf{a}_t|\mathbf{a}_0) - \nabla_{\mathbf{a}_t}\log\pi(\mathbf{a}_0|\mathbf{a}_t)\right), \quad (16)$$

where $f(\mathbf{a}_t,t)$ is a deterministic part of the drift that ensures gradual movement from the source to the target distribution. $g^2(t)$ controls the level of noise at time $t$, modulated by the signal-to-noise ratio (SNR).

The term $\frac{1}{2}\nabla_{\mathbf{a}_t}\log\pi(\mathbf{a}_t|\mathbf{a}_0) - \nabla_{\mathbf{a}_t}\log\pi(\mathbf{a}_0|\mathbf{a}_t)$ is a correction factor that adjusts the position of $\mathbf{a}_t$ to bring it closer to $\mathbf{a}_0$. This adjustment term is critical for reducing error in the final output and directly depends on the initial discrepancy $D_{\mathrm{KL}}(\pi_s(\mathbf{a}_T)\|\pi(\mathbf{a}_0))$.

The initial discrepancy contributes a fundamental limit to how close the source and target can be through the evolution process. The noise $g(t)$ and the signal-to-noise ratio $\mathrm{SNR}_t$ at each time step both affect how much control we have over the diffusion process. Higher $g(t)$ implies more noise, making it harder for the model to accurately adjust the path toward the target. Higher SNR (meaning a larger signal relative to noise) allows for more precise alignment with $\mathbf{a}_0$ as the process evolves.

To incorporate these elements into the error bound, we arrive at the following formula:

$$\mathbb{E}_{\mathbf{a}_t,\mathbf{a}_0}\left[\|\mathbf{a}_t - \mathbf{a}_0\|^2\right] \le \frac{1}{2}\cdot\frac{g^2(t)}{\mathrm{SNR}_t}\cdot D_{\mathrm{KL}}(\pi_s(\mathbf{a}_T)\|\pi(\mathbf{a}_0)), \quad (17)$$

where $\frac{1}{2}$ arises from the specific form of the adjustment term in the probability flow ODE. $\frac{g^2(t)}{\mathrm{SNR}_t}$ shows how the noise and signal-to-noise ratio affect the error bound. Because it is invariant under a fixed schedule, it can be expressed as a constant $C$.

## D. Detailed Experimental Settings and Results

The experimental setup in the simulation environment is illustrated in Fig. 2, where (a) and (c) represent indoor and outdoor base tasks, while (b) and (d) show their adaptation counterparts. For adaptation tasks, significant environmental differences exist between the target image collection phase and navigation execution phase: randomly placed boxes (b) and vehicles (d) along paths are absent in target images but present during algorithm operation, validating environmental adaptability. Subfigures (e)-(f) evaluate zero-shot generalization capabilities. Given the substantial domain gap between our publicly available real-world training dataset and simulation/test-bed environments (particularly in visual appearance), (e) demonstrates pure zero-shot performance without simulation data, while (f) shows improved results after fine-tuning with limited simulation trajectory data unrelated to target tasks. Experimental results confirm the algorithm's strong zero-shot transfer capability, with notable performance gains achieved through minimal fine-tuning.

The metrics in Table 3 and Fig. 5 are derived from adaptation tasks in indoor scenarios, while the real-world results in Fig. 4 are based on on-device experiments. All the simulation experiments were deployed on a Nvidia RTX2080Ti GPU, utilizing a Jackal move robot. The on-device experiments utilized the wheeled-legged robot Diablo, equipped with an Intel Realsense D435i and relying solely on RGB observations as shown in Fig. 4. As mentioned in the main text, the model was deployed on a Nvidia Jetson Orin AGX. The complete navigation process relies on a high-level plan-

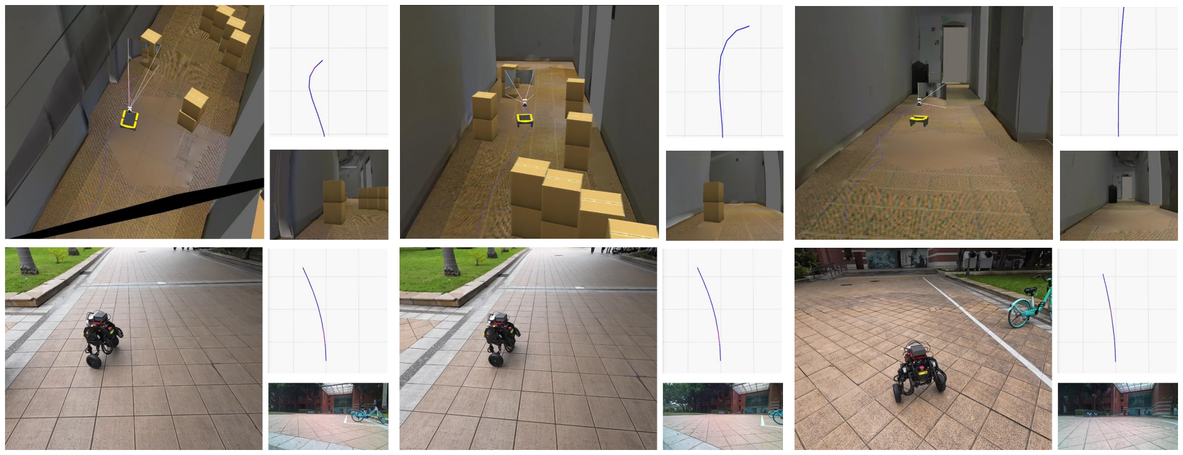

Figure 3. The figure presents navigation trajectories from experiments in both simulated and real-world environments. The top row shows obstacle navigation in a simulated corridor with visual observations and trajectory curves, while the bottom row displays outdoor navigation on tiled pathways with real camera observations. The columns highlight different test scenarios, illustrating the robot's ability to adapt and maintain consistent navigation strategies across varied environments.

ner based on a topological map. Sub-goals on the topological map are used as target image inputs for the network, guiding the policy toward the destination and enabling long-distance navigation.

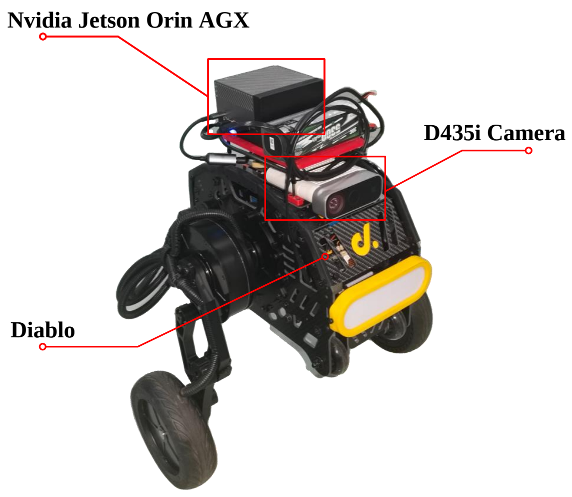

Figure 4. The wheeled-legged robot named Diablo, equipped with an Intel RealSense D435i camera and an Nvidia Jetson Orin AGX, was utilized for on-device experiments relying solely on RGB observations.

Fig. 1 illustrates the complete denoising process for NoMaD and NaviBridger with different prior actions based on DDPM. It can be observed that NaviBridger achieves higher denoising efficiency, requiring only a few steps to complete the process, whereas NoMaD based on DDPM needs more steps. Moreover, as the prior action becomes closer to the target action, the difference between the source and target distributions decreases, reducing the difficulty of the denoising network and leading to better action results.

Fig. 3 illustrates the results of navigation experiments using the learning-based prior approach in both simulated and real-world environments, highlighting its zero-shot generalization capabilities. The top row demonstrates the robot navigating a simulated corridor with static obstacles, where smooth and collision-free trajectories are achieved without prior data collection from similar test environments. The rendered visualizations and robot perspectives further emphasize the system's ability to adapt seamlessly to structured indoor scenarios. The bottom row presents the robot navigating outdoor real-world environments on tiled pathways, showcasing consistent and effective trajectories despite differences in terrain and environmental context. The learning-based prior method excels in adaptability, exhibiting robust performance across diverse scenarios and robot configurations. This zero-shot capability demonstrates its ability to generalize navigation strategies without needing domain-specific data or adjustments for different robots. The video is available in the supplementary materials.

## References

[1] Tero Karras, Miika Aittala, Timo Aila, and Samuli Laine. Elucidating the design space of diffusion-based generative models. *Advances in neural information processing systems*, 35: 26565–26577, 2022. 1

[2] Linqi Zhou, Aaron Lou, Samar Khanna, and Stefano Ermon. Denoising diffusion bridge models. In *The Twelfth International Conference on Learning Representations*, 2024. 1