HALO: Human Preference Aligned Offline Reward Learning for Robot Navigation

Gershom Seneviratne **Jianyu An** **Sahire Ellahy** **Kasun Weerakoon**
Mohamed Bashir Elnoor **Jonathan Deepak Kannan** **Amogha Thalihalla Sunil**
Dinesh Manocha

University of Maryland, College Park *

Abstract: In this paper, we introduce HALO, a novel Offline Reward Learning algorithm that quantifies human intuition in navigation into a vision-based reward function for robot navigation. HALO learns a reward model from offline data, leveraging expert trajectories collected from mobile robots. During training, actions are sampled around a reference action and ranked using preference scores derived from a Boltzmann distribution centered on the preferred action, and shaped by binary navigation feedback. The reward model is trained via the Plackett-Luce loss to align with these ranked preferences. We deploy HALO in two downstream applications: (i) an offline learned policy trained directly on the HALO-derived rewards, and (ii) a model-predictive-control (MPC) based planner that incorporates the HALO reward as an additional cost term. This showcases the versatility of HALO across both learning-based and classical navigation frameworks. Our real-world deployments on a Clearpath Husky across diverse scenarios show that policies trained with HALO generalize effectively to unseen environments and hardware setups. HALO outperforms state-of-the-art vision based navigation methods, achieving at least a 33.3% improvement in success rate, a 6.171% reduction in normalized trajectory length, and a 26.6% reduction in Fréchet distance compared to human expert trajectories.

Keywords: Reward Modelling, Preference Alignment, Vision-based Navigation

1 Introduction

Autonomous visual navigation is a fundamental capability for mobile robots operating in complex, real-world environments [1, 2, 3, 4]. Traditional systems often rely on depth sensors such as LiDAR or stereo cameras to estimate geometry and avoid obstacles. While effective, these sensors introduce significant cost, power, and hardware complexity, limiting scalability across robot platforms[3, 5]. In contrast, RGB cameras offer a low-cost and widely deployable sensing modality. However, building reliable navigation systems based solely on RGB input remains challenging, particularly due to the difficulty of inferring navigability and obstacle relevance directly from raw visual observations under dynamic lighting, ambiguous pathways, or unstructured terrain. In this context, a method that can leverage human intuition to interpret visual scenes and guide navigation decisions could enhance trajectory safety and improve navigation success [6, 7].

Reinforcement Learning (RL) has demonstrated significant potential for navigation, allowing robots to learn from direct interaction with the environment (i.e., online RL) and adapt based on experiential data [8, 9]. Many online RL methods utilize simulators during training, leveraging parallel processing and faster-than-real-time learning to improve efficiency. However, a key limitation is the *sim-to-real transfer gap* [10], where policies trained in simulation often fail to generalize effectively to real-world environments. To mitigate this, recent works have leveraged large-scale navi-

*Video, Code and more information: http://gamma.umd.edu/halo/.
Supported by ARO Grants W911NF2310046, W911NF2310352, and U.S. Army Cooperative Agreement W911NF2120076.

9th Conference on Robot Learning (CoRL 2025), Seoul, Korea.

gation datasets [11, 12] to train policies offline, an approach known as offline RL, which enables learning directly from real-world observations without requiring costly or unsafe online interaction [7, 13, 14, 15]. Advances in offline RL [16, 17] have further enabled robots to learn robust policies in a data-efficient manner using pre-collected experience, reducing reliance on online interactions. Despite these advances, both online and offline RL approaches typically depend on *hand-engineered reward functions*, requiring substantial domain expertise to identify key components and mathematically formalize human intuition [18, 19]. Moreover, reward functions must be meticulously tuned, often necessitating multiple rounds of real-world training and testing, a costly and resource-intensive process. Furthermore, many RL-based navigation algorithms rely on LiDAR or depth cameras for obstacle distance estimation during reward computation, which, while effective, introduces scalability challenges due to sensor cost and deployment constraints [20, 13].

Recent research has also explored the use of large language models (LLMs) and vision-language models (VLMs) to tackle the challenges of visual navigation. These models offer an alternative paradigm: instead of explicitly optimizing reward functions, they use natural language understanding to interpret goals, identify obstacles, or infer human intent from egocentric observations [21, 22, 23]. Building on this foundation, recent research has explored the application of these models to enhance autonomous robot navigation [24, 25, 26, 27]. Typically, these approaches involve providing the model with an egocentric view of the environment, along with a detailed text description of the scene, and then prompting the model to generate a high-level plan or to infer the likely behavior of nearby agents. Although these methods offer promising avenues for capturing nuanced human intentions and facilitating adaptive navigation strategies, they have some limitations. Specifically, LLM/VLM-based solutions often demand significant computational resources, require continuous internet connectivity due to their reliance on cloud-based services, and are not easily deployable on edge hardware [17, 16]. Consequently, these constraints impede their real-time performance and scalability, ultimately limiting their applicability in safety-critical and resource-constrained navigation scenarios.

Main contributions: To address the limitations of hand-crafted reward functions in vision based navigation, we propose **HALO**, **H**uman Preference **AL**igned **O**ffline Reward Learning, a novel framework that uses human feedback to train reward models from egocentric visual inputs and action-conditioned trajectories. Our key contributions are:

- **A preference-driven reward learning framework** that uses Plackett-Luce loss to align ranked action sets with human visual intuition, based on responses to binary feasibility queries (e.g., "Can the robot turn left?", "Can the robot accelerate?") from egocentric camera views. The resulting reward model is deployed in two downstream settings: (i) training a goal-conditioned offline policy, and (ii) augmenting a model predictive control (MPC) planner with the learned reward as a cost term, demonstrating strong generalization across diverse indoor and outdoor environments in both learning-based and classical navigation frameworks. Furthermore, the annotated user preference dataset for sub-optimal trajectories will be released along with the code for the paper.

- **An action-conditioned visual feature aggregation mechanism** that uses a binary mask of the predicted robot path generated via homography, to identify spatially relevant regions in the image. This mask is processed through a CNN to produce a weighting map that modulates visual features, allowing the model to aggregate highly relevant image information based on the intended action.

- **Extensive real-world evaluation** on the Clearpath Husky platform, showcasing HALO's ability to generalize across diverse indoor and outdoor environments not encountered during training. Our reward-model-based policies operate in real time at approximately 50 Hz on a laptop (Nvidia 3060 GPU, Intel Core i7 CPU), and consistently outperforms state of the art vision based navigation methods, achieving an 33.3% improvement in success rate, 6.171% reduction in normalized trajectory length, and 26.6% lower Fréchet distance to expert trajectories.

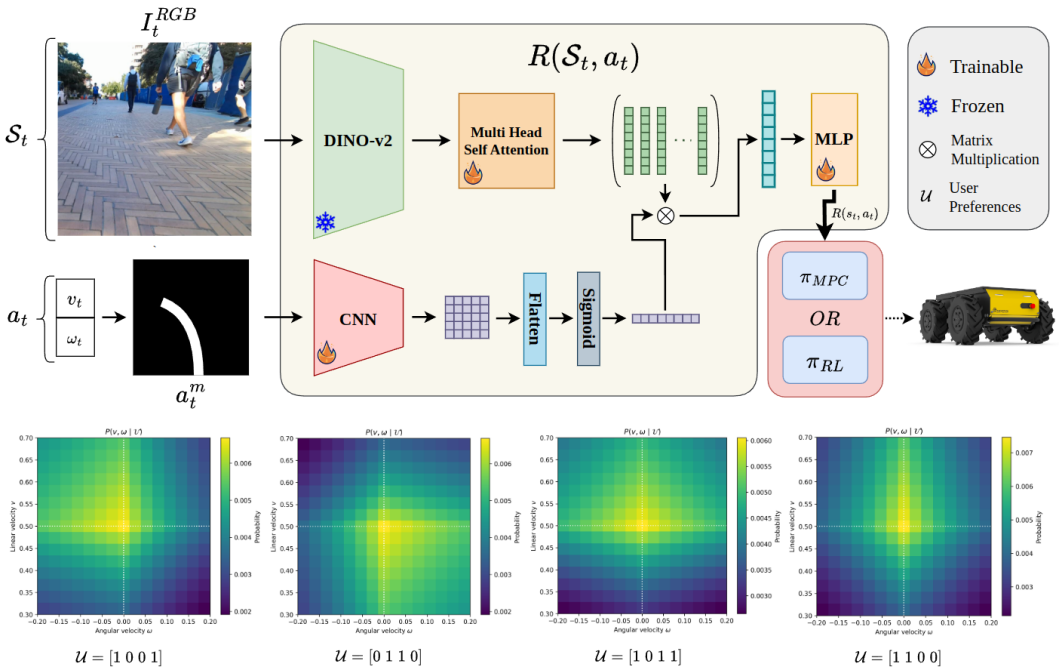

Figure 1: Architecture of the proposed reward model. Given the current observation I_t^{RGB}, a frozen DINO-v2 encoder extracts patch embeddings. Simultaneously, the candidate action $a_t = (v_t, \omega_t)$ is projected into image space via a homography transform to produce a trajectory mask a_t^m, which is passed through a trainable CNN to yield a spatial relevance weighting. This vector modulates the image patch embeddings [28], focusing on regions relevant to the trajectory. The resulting feature is passed through a trainable MLP to produce the scalar reward $R(s_t, a_t)$. This reward can be used in either a model predictive controller (π_{MPC}) or an offline reinforcement learning policy (π_{RL}). **Bottom:** Boltzmann distributions $P(v, \omega \mid \mathcal{U})$ generated from binary user preferences $\mathcal{U}$, used to assign preference scores to actions during reward model training.

2 Related Work

In this section, we review key literature on autonomous visual navigation and reward modeling for navigation based on preference data.

2.1 Navigation Algorithms for Mobile Robots

Classical geometric and rule-based methods laid the foundation for autonomous navigation [29], but their reliance on depth sensors such as LiDAR limits scalability due to high cost and hardware complexity [30, 3]. Reinforcement learning (RL) based methods enable robots to learn adaptive navigation behaviors through interaction with their environments, often in simulation [31]. Building on this, offline RL allows policies to be learned directly from large, real-world datasets without requiring online exploration [7, 13]. However, many offline RL approaches still depend on hand-crafted reward functions, which are difficult to design for nuanced tasks like obstacle avoidance or visual goal-following, and often require extensive tuning [32]. For example, Shah et al. use multiple manually defined objectives to train multi-objective policies [7], but none of these explicitly account for obstacle avoidance, limiting their scalability and generalization to complex, real-world scenarios.

2.2 Reward Modeling for Navigation Based on Preference Data

Preference-based reward modeling has emerged as an effective way to align reinforcement learning agents with human intentions. Christiano et al. introduced a method using human preferences over trajectory segments to train reward models for policy optimization [33]. Lee et al. extended this with PEBBLE, improving sample efficiency via unsupervised pre-training and experience relabeling [34]. More recently, Wang et al. leveraged VLMs to automate preference labeling, eliminating manual annotation for training reward models in simulation [35]. Most prior methods rely on the

Bradley-Terry model [36], which simplifies feedback into binary comparisons, limiting nuance and introducing noise into the reward signal.

3 Background

In this section, we introduce key preliminaries necessary for a deeper exploration of the topics discussed in this work.

3.1 Markov Decision Processes

We model robot navigation as a Markov Decision Process (MDP) with continuous states $s \in \mathcal{S}$ (egocentric RGB observations) and actions $a \in \mathcal{A}$ (linear and angular velocities). A policy $\pi(a|s)$ selects actions to maximize the expected cumulative reward. Our work focuses on learning the reward function $\mathcal{R}(s, a)$ directly from human preferences.

3.2 Offline Reinforcement Learning

Offline RL enables training policies from fixed datasets without further environment interaction, making it suitable when online exploration is unsafe or costly [37]. Algorithms such as CQL [17] and IQL [16] address distribution shift and overestimation issues, and we build on these methods to train policies with our learned reward.

3.3 Plackett-Luce Model for preference modeling

The Plackett-Luce model is a probabilistic framework for preference modeling [38] . Given a set of actions $\mathcal{A} = \{a_1, a_2, \ldots, a_n\}$ and associated scores θ_i, the probability of observing a specific ranking σ is defined as

$$P(\sigma) = \prod_{j=1}^{n} \frac{\exp(\theta_{\sigma(j)})}{\sum_{k=j}^{n} \exp(\theta_{\sigma(k)})}. \tag{1}$$

This formulation converts the scores into a probability distribution over rankings by exponentiating the scores and normalizing them sequentially.

4 Our Approach: HALO

We present HALO, an offline algorithm that uses the Plackett-Luce framework to learn a reward model aligned with human navigational preferences. In this section, we will focus on six key aspects of our approach: (1) the dataset used for training and evaluation, (2) formulating the reward learning problem, (3) generating probabilistic scores for actions based on human feedback, (4) offline reward learning based on the Plackett-Luce model, (5) the proposed reward model architecture, and (6) robot navigation using the learned reward model.

4.1 Dataset

We use the Socially CompliAnt Navigation Dataset (SCAND) [12] for offline training. It includes demonstrations in indoor and outdoor settings, where a human operator teleoperates a wheeled or legged robot around the UT Austin campus. From the 139 available scenes, we manually annotated 25 scenes (≈107,000 frames) with human preference scores (Section 4.3).

To augment this, we collected 116 additional scenes (≈33,000 frames) using a legged robot. Approximately 100 of these consist of short, truncated trajectories that would result in collisions or unsafe behavior if executed further. These were deliberately included as negative examples to counterbalance the overwhelmingly successful demonstrations in SCAND. Section 4.3 explains how

these negative trajectories are incorporated into the preference scoring framework to guide the reward model toward safer behavior and improve generalization in safety-critical situations.

4.2 Formulating the Reward Learning Problem

We model human-aligned navigation as an MDP and focus on learning a reward function $\mathcal{R}(s_t, a_t)$ that captures human navigational preferences from offline data.

In our framework, the action space $\mathcal{A}$ consists of continuous control commands represented by linear and angular velocity pairs (v, ω), and the state space $\mathcal{S}$ consists of the robot's egocentric RGB camera observation I^{RGB}. The reward model is goal-independent and quantifies the quality of actions based solely on the egocentric visual input, guided by human preference data.

The policy $\pi(a_t|s_t)$ is defined as a stochastic distribution over continuous actions conditioned on the current state. This formulation supports smooth and adaptive control strategies, suitable for navigation in dynamic and complex environments.

4.3 Preference Scores from Human Feedback

To capture human intuition, we collect binary responses to five navigation queries based on the robot's egocentric view and expert action (v^*, ω^*): *(1) Can the robot turn left? (2) Turn right? (3) Decelerate? (4) Accelerate? (5) Is the robot in immediate danger or behaving suboptimally?* These responses assess the feasibility of alternative actions. To reduce annotation effort, annotators were able to reuse previous responses by simply advancing through frames without re-entering inputs as long as the answers to the navigation queries remained unchanged. Labeling resumed when a meaningful change in the scene, such as a hallway turn or a pedestrian entering, required a different response, based on the annotator's visual judgment rather than a predefined threshold.

If the user flags danger or suboptimal behavior, a corrective reference action (v^*, ω^*) is assigned based on user preference data. This differs from the expert action recorded in the dataset and typically involves reduced linear velocity and a sharp turn. This encourages the reward model to favor safer and more responsive behaviors in similar scenarios.

We generate a discrete, dynamically feasible action set centered around (v^*, ω^*) that spans the robot's field of view:

$$\mathcal{A}_{\text{local}} = \{(v, \omega) \mid v \in \mathcal{V},\ \omega \in \Omega\}$$

Each action $(v, \omega) \in \mathcal{A}_{\text{local}}$ is assigned a probability score based on its proximity to (v^*, ω^*), using a separable Boltzmann distribution (Eq. 2, 3):

$$P(v, \omega, \mathcal{U}) = P(v, \mathcal{U}) \cdot P(\omega, \mathcal{U}), \tag{2}$$

$$P(v, \mathcal{U}) = \frac{\exp\left(-\frac{|v - v^*|}{\tau_v(\mathcal{U})}\right)}{\sum\limits_{v' \in \mathcal{V}} \exp\left(-\frac{|v' - v^*|}{\tau_v(\mathcal{U})}\right)}, \quad P(\omega, \mathcal{U}) = \frac{\exp\left(-\frac{|\omega - \omega^*|}{\tau_\omega(\mathcal{U})}\right)}{\sum\limits_{\omega' \in \Omega} \exp\left(-\frac{|\omega' - \omega^*|}{\tau_\omega(\mathcal{U})}\right)}. \tag{3}$$

The temperature values $\tau_v(\mathcal{U})$ and $\tau_\omega(\mathcal{U})$ are selected adaptively based on user preferences $\mathcal{U} \in \{0, 1\}^4$, which correspond to the four directional queries. When a directional preference is indicated, the corresponding temperature is set such that the preferred direction receives at least 95% of the total probability mass in its respective marginal distribution. In the absence of preference, the distribution is flattened by assigning a higher temperature. This ensures that undesirable actions are assigned a selection probability of less than 0.05, and that the distribution sharpens or flattens depending on the user's certainty (Figure 1:**Bottom**).

To reflect the user's perceived desirability of a scene, the final score is scaled by a scalar factor λ (Eq. 4), defined as:

$$\lambda(\mathcal{U}, \mathcal{U}_{\text{danger}}) = \begin{cases} -\dfrac{1}{1+\sum_{i=1}^{4}\mathcal{U}_i}, & \text{if } \mathcal{U}_{\text{danger}} = 1, \\ 1+\sum_{i=1}^{4}\mathcal{U}_i, & \text{otherwise.} \end{cases} \tag{4}$$

The final preference score assigned to an action (v, ω) is therefore:

$$\text{Pref}(v, \omega \mid \mathcal{U}) = \lambda(\mathcal{U}, \mathcal{U}_{\text{danger}}) \cdot P(v, \omega \mid \mathcal{U}). \tag{5}$$

These scores are used to rank the $n_v \times n_\omega$ candidate actions, with the reference action always ranked highest. The resulting preference-ordered list σ serves as supervision for reward model training.

4.4 Reward model learning based on preference data using the Plackett Luce loss

We use the Plackett-Luce model (Section 3.3) to learn a reward function that maximizes the probability of observing the preference rankings generated in Section 4.3. To achieve this, we minimize the negative log-likelihood of the Plackett-Luce ranking probability defined in Eq. (1), ensuring that the learned reward function aligns with human preferences.

$$\mathcal{L}_{\text{PL}} = -\sum_{i=1}^{n-1}\theta_{\sigma(i)} + \sum_{i=1}^{n-1}\log\left(\sum_{j=i}^{n}\exp(\theta_{\sigma(j)})\right) \tag{6}$$

By minimizing the Plackett-Luce loss defined in Eq. (6), the reward model is trained to assign higher values to actions that appear earlier in the ranked preference list, ensuring that the learned reward function captures human intuition in navigation. This loss offers a scalable and context-aware formulation for learning from ranked actions, capturing the relative importance of all candidates in a single ranking. Unlike pairwise comparison methods, it avoids the $\binom{n_v \times n_\omega}{2}$ comparisons required per timestep, resulting in a more stable loss landscape and efficient optimization.

To the best of our knowledge, this is the first approach to incorporate Plackett-Luce preference-based reward learning for modeling human intuition in autonomous navigation. This formulation offers a structured, scalable, and globally consistent framework for aligning navigation policies with human preferences.

4.5 Reward Model Architecture

The overall architecture of our system is illustrated in Figure 1. Given a state $s_t = I^{RGB}$ and a candidate action $a_t = (v_t, \omega_t) \in \mathbb{R}^2$, the model outputs a scalar reward $\mathcal{R}(s_t, a_t)$.

Image Feature Extraction: We use DINOv2 [39], a self-supervised feature extractor based on the Vision Transformer (ViT) architecture, to extract patch-level embeddings from the input image I^{RGB}. The frozen encoder outputs N_p spatial patch embeddings of dimension d_e, denoted as $F_t \in \mathbb{R}^{N_p \times d_e}$. These embeddings are further refined via N_{sa} Transformer layers with h self-attention heads, yielding $F_t^{\text{sa}} \in \mathbb{R}^{N_p \times d_e}$.

Action-Conditioned Visual Feature Aggregation: Given an action (v, ω), we generate a binary trajectory mask by projecting the path the robot would follow under that action over a fixed time horizon onto the image plane using a homography transform (see Figure 1). This results in a binary mask $a_t^m \in \mathbb{R}^{1 \times H_m \times W_m}$, which is processed by a lightweight CNN composed of stacked convolutional and pooling layers, followed by a final sigmoid activation. The output is a spatial relevance map $W_t \in \mathbb{R}^{\sqrt{N_p} \times \sqrt{N_p}}$, which is flattened into an N_p-dimensional vector. This vector

is then used to modulate the DINOv2 patch embeddings $F_t^{\text{sa}} \in \mathbb{R}^{N_p \times d_e}$ via element-wise multiplication, allowing the model to emphasize image regions relevant to the given action and support action-conditioned perception analogous to attention-based gating.

Final Reward Prediction: The aggregated feature representation is passed through a simple MLP head that maps it directly to a scalar reward value. This architecture provides an efficient mechanism for capturing the relationship between visual input and action-specific context.

4.6 Training and Offline Navigation Policy

Once trained, the reward model can be used in two ways: (i) to supervise a goal-conditioned offline policy, or (ii) to augment classical navigation algorithms such as Dynamic Window Approach (DWA) or Model Predictive Control (MPC) by serving as an additional cost term.

To improve reward learning stability and generalization, we incorporate two regularization strategies during training: (i) a focal-style penalty applied to the raw preference scores generated in Section 4.3, and (ii) a diversity regularizer that discourages similar rewards for dissimilar actions. These additions help the model produce more discriminative and informative reward outputs.

Additional training details, including offline policy learning methodology, MPC implementation and hyperparameters are provided in Appendix 7.2.

5 Results and Analysis

5.1 Implementation and Comparisons

We integrate our method into the Clearpath Husky wheeled robot for real-world experiments. The Clearpath Husky is outfitted with a Realsense d435i camera, and a laptop containing an Intel i7 processor and an NVIDIA RTX 3060 GPU. To compare against our reward model (section 4.2), we also designed a hand-engineered reward (HER) model that accounted for distance to the goal, distance to any obstacles utilizing lidar data, and a smoothness term to discourage sudden acceleration. We validated our method by comparing the IQL [40] policy trained on our reward model to the IQL policy trained on hand-engineered rewards (HER). Additionally, we trained a behavioral cloning policy (BC) [41] on the same data used to train the reward model. We also compared our method with a classical method [42] and a learning-based vision-action method, VANP [43].

5.2 Evaluation Metrics

We evaluate our reward model on three navigation metrics: (1) Success rate, (2) Normalized trajectory length, and Fréchet distance [44] w.r.t. the human teleoperation. Detailed descriptions of these metrics and their computation can be found in Appendix 7.1.1.

5.3 Discussion

We present HALO's performance results and comparisons quantitatively in Table 1 and qualitatively in Figure 2. These comparisons span three diverse real-world navigation scenarios. In the following discussion, we analyze how each method performs within these settings, highlighting differences in social compliance and obstacle avoidance. HALO consistently outperforms baseline methods in terms of Success Rate and Fréchet Distance, demonstrating its ability to produce safer and more human-aligned trajectories without relying on LiDAR or hand-crafted rewards. Trajectory length and Fréchet distance are reported as N/A when the success rate is 0, and improvement metrics are computed only against vision-based policies; DWA, which uses LiDAR for planning, is not included. Further qualitative results comparing HALO and HER methods are discussed in the Appendix 7.1.3.

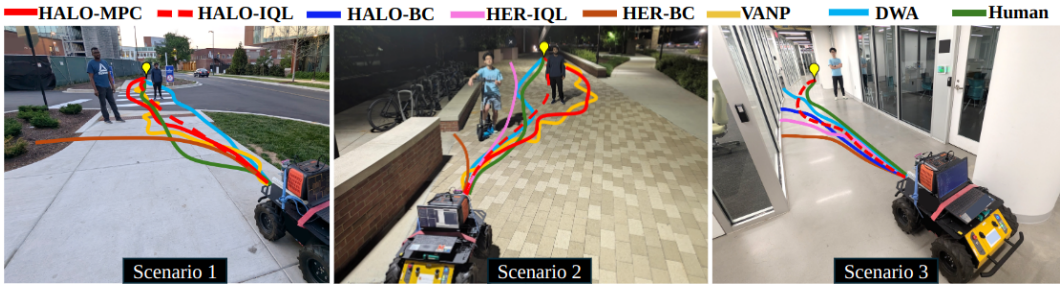

Figure 2: Comparison across three diverse real-world navigation scenarios (outdoors, low light, indoors). Each row shows the trajectory taken by different methods, DWA [42], VANP [43], HER + IQL [40]/BC[41], and HALO-based policies, highlighting differences in social compliance, obstacle avoidance, and goal-reaching behavior.

Scenario	Method	Success Rate (%) ↑	Normalized Traj. Length ↓	Fréchet Distance ↓ (m)
Scenario 1	DWA (†) [42]	80	1.043	1.677
	VANP [43]	60	1.205	1.216
	HER + TD3-BC (†) [45]	0	N/A	N/A
	HER + IQL (†) [40]	50	1.118	3.665
	HALO + TD3-BC	10	0.565	6.758
	HALO + IQL	80	1.049	1.140
	HALO + DWA	70	1.212	0.892
Scenario 2	DWA (†) [42]	100	1.157	0.678
	VANP [43]	60	1.294	1.682
	HER + TD3-BC (†) [45]	0	N/A	N/A
	HER + IQL (†) [40]	10	1.271	3.851
	HALO + TD3-BC	70	1.269	1.040
	HALO + IQL	90	1.030	0.759
	HALO + DWA	70	1.306	1.573
Scenario 3	DWA (†) [42]	0	N/A	N/A
	HER + TD3-BC (†) [45]	0	N/A	N/A
	HER + IQL (†) [40]	0	N/A	N/A
	HALO + TD3-BC	10	0.481	4.022
	HALO + IQL	80	1.136	1.263
	HALO + DWA	40	0.687	4.179

Table 1: Performance comparison of different navigation methods across three scenarios. **Methods with a † have the benefit of using LiDAR**

Across outdoor, low-light, and indoor scenarios, HALO consistently outperformed baselines in both success rate and Fréchet Distance and Normalized Trajectory Length (Table 1). In outdoor settings with pedestrians, HALO+DWA produced smoother, socially compliant trajectories that aligned with human preferences, while HALO+IQL achieved comparable success without relying on LiDAR. Under low-light conditions, HALO-trained policies maintained larger safety buffers around pedestrians than baseline planners. In challenging indoor hallways with glass walls, only HALO-trained policies generalized successfully, while LiDAR-based and hand-engineered reward methods failed. Overall, HALO provided safer, more socially aligned navigation across diverse environments and modalities. Detailed scenario-by-scenario analysis of each of these settings are provided in Appendix 7.1.2

6 Conclusions

We presented HALO, a novel offline reward learning algorithm that aligns robot navigation with human preferences by leveraging egocentric visual data and expert trajectories. HALO quantifies intuitive human navigation behavior through ranked action preferences and uses the Plackett-Luce model to learn a reward function without requiring hand-engineered heuristics. We demonstrated its adaptability by deploying it in both offline policy learning and model predictive control planners, where HALO achieves superior performance over state-of-the-art baselines across success rate, trajectory safety, and Frechet distance metrics. Our real-world evaluations highlight HALO's robustness across different robot platforms and scenarios.

Limitations

It is hard to estimate the performance of the HALO rewards early on in the training process. Since HALO trains the reward model first, the performance of the reward model is unknown until a policy is trained with the selected reward model. This sequential coupling of reward modeling and policy means the performance of the reward model is not apparent until much later into the training process. The coupling of the reward model with the policy can also make it difficult to pinpoint whether the reward model or the policy should be fine-tuned. If a policy does not perform well, one may need to fine-tune both the reward model and the corresponding policy.

The method has demonstrated inconsistent performance, where the robot will fail to avoid obstacles in scenarios where it has successfully navigated before. When the model fails to avoid an obstacle, it is often unclear why. During the training process, different versions of the model demonstrated certain tendencies, where one checkpoint model may be great at navigating outdoor scenes, and another checkpoint would be better at avoiding people while indoors or in conditions where there are many reflective surfaces such as glass walls. An ensemble of policies trained on the same reward model could be explored to mitigate this limitation.

Our method is limited by its field of view, without any explicit memory buffer. Since our method was trained on a variety of robots with varying size, width, and turning radii, the robot trained on our method showed a lack of awareness of its own width. Combined with a narrow field of view, the robot sometimes successfully avoid an obstacle only to turn back into the very same obstacle. By the time the robot sees an obstacle again it is too late to change course. The method struggles especially when the turning clearance is tight. Inclusion of a short-term memory and parameters about the vehicle width may mitigate these limitations.

References

[1] S. Paluch, J. Wirtz, and W. H. Kunz. *Service Robots and the Future of Services*, pages 423–435. Springer Fachmedien Wiesbaden, Wiesbaden, 2020. ISBN 978-3-658-31563-4. doi:10.1007/978-3-658-31563-4_21. URL https://doi.org/10.1007/978-3-658-31563-4_21.

[2] A. H. Raj, Z. Hu, H. Karnan, R. Chandra, A. Payandeh, L. Mao, P. Stone, J. Biswas, and X. Xiao. Rethinking social robot navigation: Leveraging the best of two worlds. In *2024 IEEE International Conference on Robotics and Automation (ICRA)*, pages 16330–16337. IEEE, 2024.

[3] K. Weerakoon, M. Elnoor, G. Seneviratne, V. Rajagopal, S. H. Arul, J. Liang, M. K. M. Jaffar, and D. Manocha. Behav: Behavioral rule guided autonomy using vlms for robot navigation in outdoor scenes. *arXiv preprint arXiv:2409.16484*, 2024.

[4] 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.

[5] G. Seneviratne, K. Weerakoon, M. Elnoor, V. Rajgopal, H. Varatharajan, M. K. M. Jaffar, J. Pusey, and D. Manocha. Cross-gait: Cross-attention-based multimodal representation fusion for parametric gait adaptation in complex terrains. *arXiv preprint arXiv:2409.17262*, 2024.

[6] A. Payandeh, D. Song, M. Nazeri, J. Liang, P. Mukherjee, A. H. Raj, Y. Kong, D. Manocha, and X. Xiao. Social-llava: Enhancing robot navigation through human-language reasoning in social spaces. *arXiv preprint arXiv:2501.09024*, 2024.

[7] D. Shah, A. Bhorkar, H. Leen, I. Kostrikov, N. Rhinehart, and S. Levine. Offline reinforcement learning for visual navigation. *arXiv preprint arXiv:2212.08244*, 2022.

[8] T. X. Tung and T. D. Ngo. Socially aware robot navigation using deep reinforcement learning. In *2018 IEEE Canadian Conference on Electrical & Computer Engineering (CCECE)*, pages 1–5. IEEE, 2018.

[9] A. Sigal, H.-C. Lin, and A. Moon. Improving generalization in reinforcement learning training regimes for social robot navigation. *arXiv preprint arXiv:2308.14947*, 2023.

[10] J. Yao, X. Zhang, Y. Xia, Z. Wang, A. K. Roy-Chowdhury, and J. Li. Sonic: Safe social navigation with adaptive conformal inference and constrained reinforcement learning. *arXiv preprint arXiv:2407.17460*, 2024.

[11] P. Jiang, P. Osteen, M. Wigness, and S. Saripalli. Rellis-3d dataset: Data, benchmarks and analysis. In *2021 IEEE international conference on robotics and automation (ICRA)*, pages 1110–1116. IEEE, 2021.

[12] 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.

[13] K. Weerakoon, A. J. Sathyamoorthy, M. Elnoor, and D. Manocha. Vapor: Legged robot navigation in outdoor vegetation using offline reinforcement learning. *arXiv preprint arXiv:2309.07832*, 2023.

[14] H. Caesar, V. Bankiti, A. H. Lang, S. Vora, V. E. Liong, Q. Xu, A. Krishnan, Y. Pan, G. Baldan, and O. Beijbom. nuscenes: A multimodal dataset for autonomous driving. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 11621–11631, 2020.

[15] J. Geyer, Y. Kassahun, M. Mahmudi, X. Ricou, R. Durgesh, A. S. Chung, L. Hauswald, V. H. Pham, M. Mühlegg, S. Dorn, et al. A2d2: Audi autonomous driving dataset. *arXiv preprint arXiv:2004.06320*, 2020.

[16] I. Kostrikov, A. Nair, and S. Levine. Offline reinforcement learning with implicit q-learning. *arXiv preprint arXiv:2110.06169*, 2021.

[17] A. Kumar, A. Zhou, G. Tucker, and S. Levine. Conservative q-learning for offline reinforcement learning. *Advances in neural information processing systems*, 33:1179–1191, 2020.

[18] A. Kapoor, S. Swamy, P. Bachiller, and L. J. Manso. Socnavgym: a reinforcement learning gym for social navigation. In *2023 32nd IEEE International Conference on Robot and Human Interactive Communication (RO-MAN)*, pages 2010–2017. IEEE, 2023.

[19] J. Liang, U. Patel, A. J. Sathyamoorthy, and D. Manocha. Crowd-steer: Realtime smooth and collision-free robot navigation in densely crowded scenarios trained using high-fidelity simulation. In *Proceedings of the Twenty-Ninth International Conference on International Joint Conferences on Artificial Intelligence*, pages 4221–4228, 2021.

[20] U. Patel, N. K. S. Kumar, A. J. Sathyamoorthy, and D. Manocha. Dwa-rl: Dynamically feasible deep reinforcement learning policy for robot navigation among mobile obstacles. In *2021 IEEE International Conference on Robotics and Automation (ICRA)*, pages 6057–6063. IEEE, 2021.

[21] P. Mondorf and B. Plank. Beyond accuracy: Evaluating the reasoning behavior of large language models–a survey. *arXiv preprint arXiv:2404.01869*, 2024.

[22] J. Huang and K. C.-C. Chang. Towards reasoning in large language models: A survey. *arXiv preprint arXiv:2212.10403*, 2022.

[23] M. Parmar, N. Patel, N. Varshney, M. Nakamura, M. Luo, S. Mashetty, A. Mitra, and C. Baral. Logicbench: Towards systematic evaluation of logical reasoning ability of large language models. *arXiv preprint arXiv:2404.15522*, 2024.

[24] D. Song, J. Liang, A. Payandeh, A. H. Raj, X. Xiao, and D. Manocha. Vlm-social-nav: Socially aware robot navigation through scoring using vision-language models. *IEEE Robotics and Automation Letters*, 2024.

[25] C. Huang, O. Mees, A. Zeng, and W. Burgard. Visual language maps for robot navigation. *2023 IEEE International Conference on Robotics and Automation (ICRA)*, pages 10608–10615, 2022. URL https://api.semanticscholar.org/CorpusID:252846548.

[26] A. J. Sathyamoorthy, K. Weerakoon, M. B. Elnoor, A. Zore, B. Ichter, F. Xia, J. Tan, W. Yu, and D. Manocha. Convoi: Context-aware navigation using vision language models in outdoor and indoor environments. *2024 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 13837–13844, 2024. URL https://api.semanticscholar.org/CorpusID:268681321.

[27] M. Elnoor, K. Weerakoon, G. Seneviratne, R. Xian, T. Guan, M. K. M. Jaffar, V. Rajagopal, and D. Manocha. Robot navigation using physically grounded vision-language models in outdoor environments. *arXiv preprint arXiv:2409.20445*, 2024.

[28] E. Perez, F. Strub, H. De Vries, V. Dumoulin, and A. Courville. Film: Visual reasoning with a general conditioning layer. In *Proceedings of the AAAI conference on artificial intelligence*, volume 32, 2018.

[29] D. F. W. B. S. Thrunyz. The dynamic window approach to collision avoidance. 1997.

[30] G. Kulathunga, A. Yilmaz, Z. Huang, I. Hroob, H. Arunachalam, L. Guevara, A. Klimchik, G. Cielniak, and M. Hanheide. Resilient timed elastic band planner for collision-free navigation in unknown environments. *arXiv preprint arXiv:2412.03174*, 2024.

[31] A. Dosovitskiy, G. Ros, F. Codevilla, A. Lopez, and V. Koltun. Carla: An open urban driving simulator. In *Conference on robot learning*, pages 1–16. PMLR, 2017.

[32] J. Kim, S. Kang, S. Yang, B. Kim, J. Yura, and D. Kim. Transformable gaussian reward function for socially aware navigation using deep reinforcement learning. *Sensors (Basel, Switzerland)*, 24(14):4540, 2024.

[33] P. F. Christiano, J. Leike, T. Brown, M. Martic, S. Legg, and D. Amodei. Deep reinforcement learning from human preferences. *Advances in neural information processing systems*, 30, 2017.

[34] K. Lee, L. Smith, and P. Abbeel. Pebble: Feedback-efficient interactive reinforcement learning via relabeling experience and unsupervised pre-training. *arXiv preprint arXiv:2106.05091*, 2021.

[35] Y. Wang, Z. Sun, J. Zhang, Z. Xian, E. Biyik, D. Held, and Z. M. Erickson. Rl-vlm-f: Reinforcement learning from vision language foundation model feedback. *ArXiv*, abs/2402.03681, 2024. URL https://api.semanticscholar.org/CorpusID:267499679.

[36] R. A. Bradley and M. E. Terry. Rank analysis of incomplete block designs: I. the method of paired comparisons. *Biometrika*, 39:324, 1952. URL https://api.semanticscholar.org/CorpusID:125209808.

[37] S. Levine, A. Kumar, G. Tucker, and J. Fu. Offline reinforcement learning: Tutorial, review, and perspectives on open problems. *arXiv preprint arXiv:2005.01643*, 2020.

[38] R. L. Plackett. The analysis of permutations. *Journal of The Royal Statistical Society Series C-applied Statistics*, 24:193–202, 1975. URL https://api.semanticscholar.org/CorpusID:116534299.

[39] M. Oquab, T. Darcet, T. Moutakanni, H. V. Vo, M. Szafraniec, V. Khalidov, P. Fernandez, D. Haziza, F. Massa, A. El-Nouby, R. Howes, P.-Y. Huang, H. Xu, V. Sharma, S.-W. Li, W. Galuba, M. Rabbat, M. Assran, N. Ballas, G. Synnaeve, I. Misra, H. Jegou, J. Mairal, P. Labatut, A. Joulin, and P. Bojanowski. Dinov2: Learning robust visual features without supervision, 2023.

[40] I. Kostrikov, A. Nair, and S. Levine. Offline reinforcement learning with implicit q-learning. *arXiv preprint arXiv:2110.06169*, 2021.

[41] S. Fujimoto, H. Hoof, and D. Meger. Addressing function approximation error in actor-critic methods. In *International conference on machine learning*, pages 1587–1596. PMLR, 2018.

[42] D. Fox, W. Burgard, and S. Thrun. The dynamic window approach to collision avoidance. *IEEE Robotics & Automation Magazine*, 4(1):23–33, 1997.

[43] M. Nazeri, J. Wang, A. Payandeh, and X. Xiao. Vanp: Learning where to see for navigation with self-supervised vision-action pre-training. In *2024 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 2741–2746, 2024. doi:10.1109/IROS58592.2024.10802451.

[44] H. Alt and M. Godau. Computing the fréchet distance between two polygonal curves. *International Journal of Computational Geometry & Applications*, 5(01n02):75–91, 1995.

[45] S. Fujimoto and S. S. Gu. A minimalist approach to offline reinforcement learning. *Advances in neural information processing systems*, 34:20132–20145, 2021.

[46] D. Tarasov, A. Nikulin, D. Akimov, V. Kurenkov, and S. Kolesnikov. Corl: Research-oriented deep offline reinforcement learning library. In *3rd Offline RL Workshop: Offline RL as a "Launchpad"*, 2022. URL https://openreview.net/forum?id=SyAS49bBcv.

7 Appendix

7.1 Results and Analysis

This section provides definitions for the evaluation metrics used in the main paper and presents additional qualitative results comparing HALO-based and hand-engineered reward policies in diverse real-world navigation scenarios.

7.1.1 Metrics

Metric	Definition	Description
Success Rate	$\frac{N_{\text{success}}}{N_{\text{total}}}$	Fraction of episodes where the robot reaches the goal without collisions. $N_{\text{total}} = 10$
Normalized Trajectory Length	$\frac{L_{\text{actual}}}{L_{\text{expert}}}$	Ratio of the executed trajectory length to the expert trajectory length. Lower is better.
Fréchet Distance to Expert Trajectory	$\inf_{\alpha,\beta} \max_{t\in[0,1]} \lVert T_r(\alpha(t)) - T_e(\beta(t)) \rVert$	Measures the maximum distance between points on the robot trajectory T_r and expert trajectory T_e under continuous, non-decreasing time reparameterizations. This metric captures both spatial proximity and temporal alignment. Lower is better.

Table 2: Evaluation metrics used for assessing navigation performance.

All metrics are averaged over both the successful (reaching the goal) and unsuccessful trials (collision/ not reaching the goal)

7.1.2 Detailed Analysis of Evaluation Scenarios

Scenario 1 depicts an outdoor environment featuring pedestrians and a crosswalk. In this setting, a human would naturally prefer to avoid pedestrians and navigate along the crosswalk to reach the goal. The MPC planner equipped with the HALO reward model (HALO + DWA) mirrors this human-like behavior, producing smoother, more socially compliant trajectories. As a result, it achieves a lower Fréchet distance and a higher success rate. In contrast, the DWA planner opts for a direct path that ignores the sidewalk and pedestrian context, largely due to its reliance on LiDAR-only perception. While VANP demonstrates the ability to avoid pedestrians, it exhibits jerky motion, particularly within the crosswalk region, attributed to inconsistent action predictions. Similar to the HALO + DWA method, the IQL policy trained with the HALO model (HALO + IQL) also demonstrates social compliance by achieving a low Fréchet distance, although not as low as HALO + DWA. It achieves a success rate comparable to the DWA method, but without relying on LiDAR. The IQL policy trained using the hand-engineered reward (HER + IQL) achieved moderate success rates but did not perform as well as its HALO counterpart. The behavior cloning policy trained with HALO rewards (HALO + BC) was able to successfully avoid obstacles but failed to reach the goal. The BC policy trained using hand-engineered rewards (HER + BC) became stuck in a slight left-hand turn, ultimately driving itself into the bushes.

Scenario 2 features pedestrians and pavement regions under low-light conditions. While the DWA planner avoids pedestrians and proceeds directly toward the goal, both VANP and the MPC planner with the HALO reward model (HALO + DWA) adopt a more socially aware approach by maintaining a large buffer radius around pedestrians, even if it requires adjusting their heading back

toward the goal after passing. This behavior reflects the socially compliant navigation induced by the HALO reward model in the MPC-based planner. The IQL and behavior cloning (BC) policies trained with the HALO reward model both achieved relatively high success rates but demonstrated less social compliance. They tended to take more direct paths toward the goal, often passing closer to pedestrians compared to HALO + DWA. The policies trained with the hand-engineered reward (HER) consistently failed to reach the goal. Specifically, the behavior cloning policy trained with the hand-engineered reward (HER + BC) became stuck heading toward the curb.

Scenario 3 takes place in an indoor hallway environment with glass walls and pedestrians. LiDAR-based approaches like DWA struggle in this setting, failing to detect glass walls and resulting in collisions. Similarly, IQL and behavior cloning (BC) methods trained with hand-engineered rewards fail to generalize to this unseen scenario. In contrast, the BC policy trained using the HALO reward demonstrates relatively better performance, indicating improved generalization. Notably, the IQL policy trained with the HALO model exhibits human-like behavior, successfully recognizing both glass walls and pedestrians and navigating the hallway effectively to reach the goal.

7.1.3 Further Qualitative Analysis

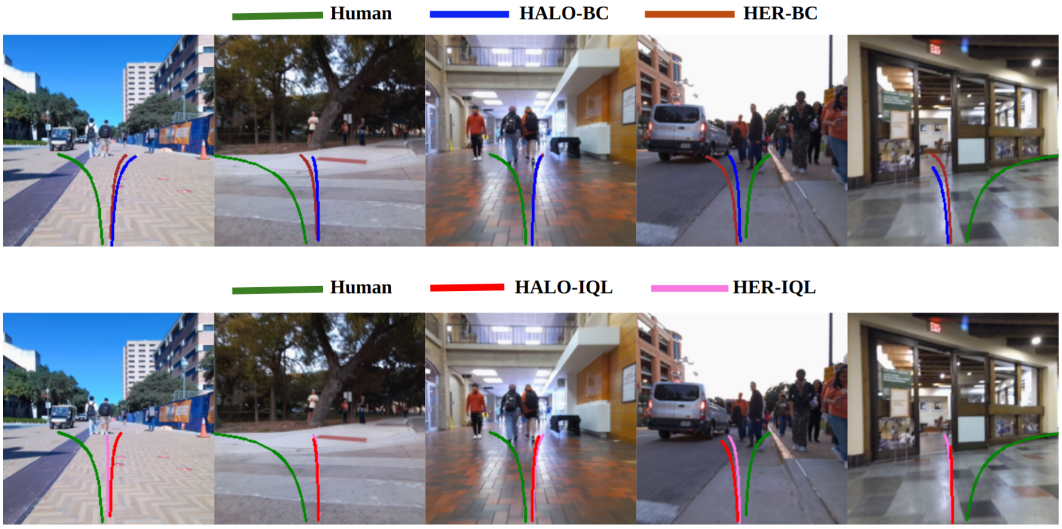

Figure 3: Qualitative Analysis for Behavioral Cloning (BC) and Implicit Q-Learning (IQL) policies for HALO and Hand Engineered Reward (HER)

(top) illustrates example outputs from HALO-BC and HER-BC policies across varied navigation scenarios. The bottom panel shows similar comparisons for IQL policies.

Figure 3 **(top)** illustrates example outputs from HALO-BC and HER-BC policies across varied navigation scenarios, while Figure 3 **(bottom)** presents comparable results for Implicit Q-Learning (IQL) policies. In both cases, we showcase five egocentric images and compare the actions selected by the policies to those taken by a human operator.

The HALO-BC/HALO-IQL policies are trained using reward models learned from human preference feedback, while the hand-engineered reward (HER) BC/IQL policies are trained using manually defined rewards (see Section 7.4). In all images, the human-selected actions correspond to higher velocities or sharper turns, while the BC/IQL policies tend to produce more conservative actions. The divergence between human and policy behavior may be attributable to the human operator's access to contextual information beyond the robot's egocentric view. For example, in the leftmost image, the human executes a sharp left turn, potentially into oncoming traffic. While this action may appear suboptimal based solely on the visible scene, it could reflect additional cues, such

as an approaching agent or a clearer path outside the frame. Given only the egocentric input, the actions chosen by the BC/IQL policies often appear more cautious and contextually appropriate. Each trajectory represents a three-second prediction horizon; the policies will generate new actions before the current trajectory completes.

The HALO-trained policies generate plausible and socially compliant behaviors across diverse scenarios. In the crosswalk image (Figure 3 second from the left), while the human prepares to turn left, the policies prioritize entering the sidewalk, reflecting goal-directed yet safe behavior. When pedestrians are present, the policies prefer to pass on the right, consistent with social norms. In the second-to-last image, a crowded scenario, the human elects to turn right over a yellow curb and through a dense group of pedestrians. This may be feasible due to the use of a legged robot in the SCAND dataset, but the policies instead select a leftward trajectory, trading crowd avoidance for slight proximity to vehicle lanes. In the rightmost image, the policies successfully infer a trajectory through an open door, demonstrating spatial reasoning using only egocentric input.

7.2 Implementation Details

This section details the policy network implementation, the CNN used to process trajectory masks, the key hyperparameters, and model dimensions used in HALO.

Policy Network. The policy network shares the same visual encoder and MLP structure as the reward model but replaces trajectory-based attention with uniform aggregation. Specifically, patch embeddings F_t are averaged to obtain a global context vector, which is passed through an MLP to predict the parameters of a Gaussian distribution over actions. During deployment, actions are sampled from this distribution; during training, gradients are propagated using the reparameterization trick. Q and V networks follow the same structure as the reward model and the policy respectively but output a single scalar value instead of action parameters.

Trajectory Mask CNN. To convert the binary trajectory mask a_t^m into a spatial weighting map W_t, we use a lightweight convolutional neural network with the following structure:

- `Conv2D(1, 4, kernel_size=3, stride=1, padding=1)`, ReLU
- `MaxPool2D(kernel_size=2, stride=2)`
- `Conv2D(4, 8, kernel_size=3, stride=1, padding=1)`, ReLU
- `Conv2D(8, 4, kernel_size=3, stride=1, padding=1)`
- `Conv2D(4, 1, kernel_size=3, stride=1, padding=1)`, Sigmoid

7.2.1 Model Parameters

Symbol	Definition	Value
H, W	Input image height and width	224, 224
H_m, W_m	Mask resolution after homography projection	32, 32
N_p	Number of image patches (from DINO-v2)	256
d_e	Embedding dimension of each patch	384
f_t	Aggregated feature vector from weighted patches	$\mathbb{R}^{384}$
R_t	Scalar reward output	$\mathbb{R}$
n_v, n_ω	Number of discrete velocity and angular bins	5, 5
	Dropout rate for MLPs	0.1

Table 3: Definitions and values of key model dimensions and constants used throughout the architecture.

7.3 Training Methodology

7.3.1 Reward Model Training

The reward model is trained using the Plackett-Luce (PL) loss, which aligns predicted scalar rewards with human-labeled preference rankings over discrete action sets. We apply a 20% validation split and employ early stopping based on validation loss. In addition to the PL loss, we incorporate three auxiliary loss terms to regularize the reward predictions:

- **Reward Diversity Regularization:** Encourages the model to assign more distinct rewards to dissimilar actions. This is achieved by penalizing pairs of actions whose reward differences are too small relative to their distances in action space, using a symmetric hinge-based loss:

$$\mathcal{L}_{\text{div}} = \mathbb{E}_{(a_i,a_j)}\left[\max\left(0, \|a_i - a_j\| - \frac{1}{c}|r_i - r_j|\right) + \max\left(0, c|r_i - r_j| - \|a_i - a_j\|\right)\right]$$

  where a_i, a_j are actions and r_i, r_j are their predicted rewards. The constant c defines the desired proportionality between action and reward differences.

- **Focal Regression Regularization:** A weighted mean squared error (MSE) that penalizes large reward errors more heavily, thereby encouraging the model to capture both order (from PL loss) and relative magnitudes:

$$\mathcal{L}_{\text{focal}} = \mathbb{E}\left[(\hat{r} - r)^2 \cdot |\hat{r} - r|^{\gamma}\right]$$

  where $\hat{r} \in \mathbb{R}^Q$ and $r \in \mathbb{R}^Q$ are the predicted and target reward vectors, and γ controls the emphasis on larger errors.

- **L2 Regularization:** A regularization term that penalizes large reward magnitudes to improve numerical stability and prevent overly confident predictions:

$$\mathcal{L}_{\text{L2}} = \mathbb{E}\left[\|\hat{r}\|_2^2\right]$$

  where $\hat{r}$ denotes the vector of predicted rewards.

The final training objective is given by:

$$\mathcal{L}_{\text{total}} = \mathcal{L}_{\text{PL}} + \lambda_1 \mathcal{L}_{\text{div}} + \lambda_2 \mathcal{L}_{\text{L2}} + \lambda_3 \mathcal{L}_{\text{focal}}$$

Table 4 summarizes the hyperparameters used for training the reward model.

Hyperparameter	**Value**
Optimizer	AdamW
Learning rate	2×10^{-4}
Weight decay	1×10^{-3}
Batch size	256
Number of epochs	200
Scheduler	Cosine Annealing with Warm Restarts
Warm-up epochs	3
Cosine T_0	25
Cosine T_{mult}	2
Validation split	20%
λ_1 (reward diversity)	2.0
λ_2 (L2 penalty)	0.01
λ_3 (focal regression)	0.05

Table 4: Training hyperparameters used for reward model optimization.

7.3.2 Policy Training

We train goal-conditioned policies using two offline methods: Behavior Cloning (TD3-BC) [41], and Implicit Q-Learning (IQL) [40]. Since the emphasis of our method is the training of the reward model, we leveraged the standard implementation of the clean offline reinforcement learning (CORL) [46] library. The offline RL implementations such as learning rate, loss functions, etc were mostly kept as-is with changes implemented when appropriate, such as including image transformations for the DINOV2 encoder, and providing the trajectory mask to the critic network. In all cases, the reward signal is provided by our trained preference model.

7.4 Baselines and Classical Control Details

7.4.1 Hand-Engineered Reward for Offline Comparison Methods

Each action (v, ω) is simulated forward in time into a trajectory $\tau_{v,\omega} = \{(x_t, y_t, \theta_t)\}_{t=1}^{T}$. *The comparison methods benefit from access to LiDAR, enabling the use of hand-crafted reward functions that exploit precise geometric cues for obstacle avoidance.*

The reward is computed as a weighted sum of three cost components:

$$R(v,\omega) = -\left(\lambda_{\text{goal}} \cdot \underbrace{d_{\text{goal}}(\tau_{v,\omega}, g)}_{\text{goal distance}} + \lambda_{\text{head}} \cdot \underbrace{\theta_{\text{err}}(\tau_{v,\omega}, g)}_{\text{heading error}} + \lambda_{\text{obs}} \cdot \underbrace{c_{\text{obs}}(\tau_{v,\omega})}_{\text{obstacle cost}} + \lambda_{\text{smooth}} \cdot \underbrace{\|a - a_{\text{prev}}\|_2}_{\text{smoothness}}\right)$$

where:

- $d_{\text{goal}}(\tau_{v,\omega}, g)$: Euclidean distance between the final state of the trajectory and the goal position,
- $\theta_{\text{err}}(\tau_{v,\omega}, g)$: absolute difference in heading between the final trajectory angle and the goal orientation,
- $c_{\text{obs}}(\tau_{v,\omega})$: an exponential penalty that emphasizes proximity, assigning higher cost to trajectories near LIDAR-detected obstacles by weighting inversely with distance.
- $\|a - a_{\text{prev}}\|_2$: L2 norm enforcing smoothness relative to the previous action.

We use the following weights, empirically selected based on our offline dataset:

$$\lambda_{\text{goal}} = \frac{1}{5}, \quad \lambda_{\text{head}} = 1.0, \quad \lambda_{\text{obs}} = \frac{1}{8}, \quad \lambda_{\text{smooth}} = 1.0$$

7.4.2 Goal Conditioning with HALO

To enable goal-directed navigation with our learned reward model, we modify the hand-engineered reward by replacing the obstacle penalty with the output of HALO, which captures human preferences, including implicit notions of obstacle avoidance and social compliance:

$$R(v,\omega) = -\left(\lambda_{\text{goal}} \cdot d_{\text{goal}}(\tau_{v,\omega}, g) + \lambda_{\text{head}} \cdot \theta_{\text{err}}(\tau_{v,\omega}, g)\right) + \lambda_{\text{halo}} \cdot \underbrace{R_{\text{HALO}}(s,a)}_{\text{learned reward}}$$

This formulation preserves geometric alignment to the goal while incorporating semantics from human-labeled preferences. All existing weights are retained from the hand-engineered formulation, and we set $\lambda_{\text{halo}} = \frac{1}{5}$.

7.4.3 DWA Planner Configuration

We adopt a standard goal-conditioned Dynamic Window Approach (DWA) planner with a 1.0-second planning horizon, 5 Hz control frequency, and 25 uniformly sampled linear and angular

velocities. To integrate learned preferences, we simply add the negative of our reward model's output as a cost term:

$$J(a) = J_{\text{DWA}}(s, a) - \lambda_{\text{halo}} \cdot R_{\text{HALO}}(s, a)$$

Here, $J_{\text{DWA}}(s, a)$ denotes the hand-crafted reward based on goal distance, heading error , and action transition smoothing (excluding obstacle cost), and $R_{\text{HALO}}(s, a)$ is the learned preference-based reward. We set $\lambda_{\text{halo}} = \frac{1}{5}$.