2026 IEEE International Conference on Robotics and Automation (ICRA 2026)
June 1-5, 2026. Vienna, Austria

Predictive Local Planning with Multi-Step Reward and Q-Value Forecasting

Yuhan Du[1], Yuxiang Cui[1], Yulin Peng[1], Yiyuan Pan[1], Tianhao Cai[1], Yue Wang[1*], and Rong Xiong[1]

***Abstract*—Planning in dynamic environments often relies on explicit future observation prediction or value-based estimation, both of which can be brittle or hard to generalize in uncertain settings. We propose a novel model-based reinforcement learning framework that performs trajectory rollout and optimization entirely in a learned latent space. Instead of predicting future observations explicitly, our method evaluates candidate trajectories through multi-step reward prediction and terminal Q-value estimation in the latent domain, enabling robust and generalizable planning in dynamic environments. A policy model generates an initial trajectory in latent space, which is then refined via a smoothness-regularized optimization using Model Predictive Path Integral (MPPI), guided by the predicted cumulative reward and Q-values. This avoids the complexity of future state reconstruction while ensuring dynamically feasible execution. To enhance the model's deployment performance in crowded or interactive scenarios, we further introduce a lightweight social reward that penalizes unsafe overtaking and encourages yielding behavior. Experiments in both simulation and real-world environments show improved success rate, efficiency, and social acceptability compared to strong baselines.**

I. Introduction

Autonomous navigation in dynamic and uncertain environments is a long-standing challenge in robotics, where the robot must plan safe, efficient, and socially acceptable paths based on limited sensory information. While imitation learning has shown effectiveness in learning reactive behaviors from demonstrations, it typically requires large-scale, high-quality expert data and often struggles with compounding errors in long-horizon planning tasks [1] [2]. In contrast, reinforcement learning (RL) enables autonomous agents to learn complex behaviors through trial-and-error interaction with the environment, offering the flexibility to optimize long-term objectives without expert supervision [3].

Within RL, *model-free* methods have demonstrated strong performance across a variety of tasks but often suffer from low sample efficiency. As a result, *model-based* reinforcement learning (MBRL) has gained increasing attention for its ability to simulate interactions in a learned model and generate synthetic rollouts to improve learning efficiency [4] [5].

However, most existing model-based reinforcement learning (MBRL) methods face two critical limitations related to environmental prediction and trajectory evaluation. In terms of environmental prediction, many methods rely on explicit environment prediction to forecast future obstacle states. In uncertain, multi-modal dynamic environments, such non-decoupled explicit prediction becomes prohibitively difficult due to the high dimensionality of raw observations, partial observability, and the stochastic behaviors of dynamic agents. To reduce this complexity, some approaches adopt decoupled prediction [6]. However, this simplification neglects the interaction between the robot and its surroundings, causing the loss of crucial information for safe and adaptive navigation. Even when explicit predictions are available, using them for online planning remains challenging, since anticipating every obstacle's trajectory does not ensure robust decision-making under diverse and unpredictable behaviors.

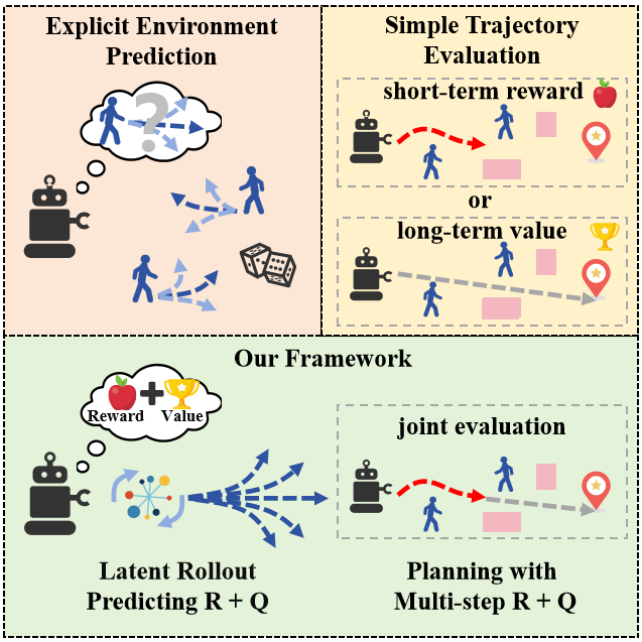

Fig. 1: Overview of the motivation and framework. (**Top-left**) Explicit environment prediction is difficult in uncertain, multi-modal dynamics. (**Top-right**) Existing planning often relies on single-objective evaluation, either short-term reward or long-term value. (**Bottom**) Our method avoids explicit environment prediction by forecasting multi-step rewards and terminal Q-values, which are then integrated to evaluate and optimize trajectories through latent-space planning.

*This work was supported by the National Nature Science Foundation of China under Grant 62373322, the National Nature Science Foundation of China under Grant 62522317 and Zhejiang Provincial Natural Science Foundation of China under Grant No. LD24F030001.

[1]Yuhan Du, Yuxiang Cui, Yulin Peng, Yiyuan Pan, Tianhao Cai, Yue Wang, and Rong Xiong are with [1]Zhejiang University, Hangzhou, China.
*Corresponding author.

At the trajectory evaluation level, existing approaches can be broadly categorized into two types. One class scores trajectories based on short-horizon cumulative rewards or intermediate feedback within a limited prediction window. This provides timely guidance but often overlooks the consequences of actions beyond the planning horizon. The other class relies on estimating the expected long-term return of the entire trajectory, offering goal-oriented optimization but lacking sensitivity to immediate risks or short-term disruptions [7]. Both strategies, when used alone, may lead to suboptimal planning—either being overly myopic or insufficiently responsive in complex and uncertain environments [8].

To address these limitations, we propose a trajectory evaluation and planning framework that operates entirely in a learned latent space, as demonstrated in Fig. 1. Specifically, we utilize a set of predictive models to estimate multi-step cumulative rewards and terminal Q-values based on the latent representation of recent observations. These predictions serve as the foundation for evaluating and optimizing candidate trajectories in the latent space. Crucially, the prediction process in the latent space is non-decoupled, meaning it captures the interaction dynamics between the robot and its environment rather than modeling them independently. This allows the model to account for how the robot's actions influence the future evolution of the scene. Because both reward estimation and Q-value approximation are conducted internally in the latent space, our method supports full latent-space rollout and trajectory refinement without requiring accurate, explicit prediction of future observations or obstacle states. This avoids the need for complex and potentially error-prone visible environment forecasting, enabling robust planning in dynamic and uncertain settings.

Our main contributions are:

- We propose a novel latent-space planning framework that avoids explicit environment prediction by performing interaction-aware rollouts and jointly forecasting multi-step rewards and terminal Q-values entirely in a compact latent representation.
- We introduce a multi-objective trajectory optimization strategy that leverages the predicted rewards and Q-values, augmented with a smoothness constraint, to effectively balance short-term rewards and long-term returns.
- We design a lightweight social reward mechanism that enhances socially compliant navigation in crowded and dynamic environments.

II. Related Work

A. Modular vs. End-to-End Navigation Frameworks

Traditional robot navigation pipelines typically adopt a modular design, dividing the problem into perception, mapping, planning, and control. While modularity improves interpretability and debugging, it often leads to suboptimal performance due to cascading errors and the need for manual parameter tuning [9] [10].

To overcome these limitations, end-to-end learning-based approaches have been explored [11] [12], where policies are trained to map sensor inputs directly to control commands. These include supervised imitation learning, behavior cloning, and reinforcement learning-based frameworks. Although promising in handling raw observations, end-to-end methods may suffer from poor generalization and lack of interpretability, especially in safety-critical scenarios.

B. Model-Based vs. Learning-Based Methods

Model-based approaches, such as Model Predictive Control (MPC), plan trajectories by explicitly modeling system dynamics and constraints. They offer strong safety and feasibility guarantees but are often computationally expensive and hard to scale in dynamic environments [13] [14].

On the other hand, model-free reinforcement learning (RL) methods learn navigation behaviors purely from interaction. While sample-inefficient, they are more flexible and easier to deploy once trained. Hybrid model-based RL methods attempt to combine the sample efficiency of model-based techniques with the adaptability of learning-based strategies [15].

However, many learning-based planners either rely on predicting future observations explicitly—leading to high model complexity—or evaluate candidate trajectories using global value functions that are hard to generalize [6]. This motivates the use of learned latent dynamics models for compact and robust planning.

C. Latent Rollout and Predictive Scoring

Planning in latent space has become a popular technique in model-based RL, offering an efficient alternative to full observation prediction. By rolling out trajectories in a learned low-dimensional latent space, models like PlaNet and Dreamer bypass the need to reconstruct high-dimensional sensor inputs [16].

TDMPC and its variants go further by evaluating latent trajectories using both multi-step reward prediction and Q-value estimation [17] [18]. While this provides more robust scoring, such methods are primarily validated on fully-observed continuous-control benchmarks and often do not explicitly account for motion feasibility constraints or dynamic social interactions [19] [20].

Our method follows this line of work but adapts it to dynamic navigation tasks. We initialize candidate trajectories using a policy model and refine them via lightweight optimization that incorporates smoothness constraints and social rewards. This improves trajectory feasibility and real-world deployability, especially in environments involving moving obstacles.

III. Method

We propose a latent-space planning framework that integrates perception encoding, model-based latent rollout, and trajectory optimization. The system, shown in Fig. 2, operates in a fully end-to-end manner and avoids explicit prediction of future environmental states. Given multi-frame LiDAR, robot velocity, and goal information as input, an encoder extracts latent representations. These are used to perform

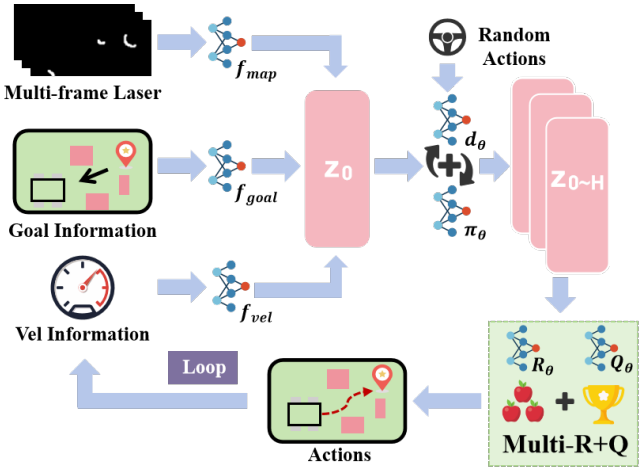

Fig. 2: System overview of our latent-space navigation framework. Multi-frame LiDAR, goal, and velocity inputs are encoded into a latent state z_0, from which multiple rollouts are generated via the dynamics d_θ and policy prior π_θ. Candidate trajectories are scored with predicted multi-step rewards and terminal Q-values (Multi-R+Q), and the optimized action is executed in a closed loop.

multiple latent rollouts via learned world models, enabling trajectory-level reasoning that implicitly considers dynamic agent interactions.

Each sampled trajectory is evaluated using both multi-step predicted rewards and terminal value estimates, allowing for joint optimization of short-term behavior and long-term goal reaching. The best action is selected accordingly, and the process repeats in a closed-loop manner.

A. Observation Encoding via Dynamic Obstacle Map

To robustly perceive dynamic environments for socially interactive navigation, we encode multi-frame LiDAR, robot motion, and goal information into a unified latent state [9]. Instead of feeding raw laser readings [21] [2], we transform the last eight laser scans and odometry into the current robot frame and aggregate them into a stacked grayscale obstacle map: static structures are reinforced by overlap, while moving agents appear as motion trails. Following prior practice, we compensate the robot's ego-motion by adjusting scans using the relative heading change between adjacent steps, and use different grayscale intensities to distinguish static vs. dynamic obstacles.

Let $o_t^l \in \mathbb{R}^{W\times H\times 8}$ be the stacked map, $o_t^v \in \mathbb{R}^2$ the linear/angular velocity, and $o_t^g \in \mathbb{R}^2$ the relative goal. Three encoders $f_{\text{map}}, f_{\text{vel}}, f_{\text{goal}}$ extract features and are fused as

$$z_t = \text{concat}\big(f_{\text{map}}(o_t^l),\ f_{\text{vel}}(o_t^v),\ f_{\text{goal}}(o_t^g)\big),$$

where $z_t \in \mathbb{R}^d$ is the compact latent representation of the current state.

B. Latent-Space Model-Based Reinforcement Learning

We adopt a reinforcement learning architecture that operates entirely in the latent space, with an action space defined by linear and angular velocities (v, ω). Transitions, rewards, and value estimations are predicted without explicit future observation modeling. This allows the agent to implicitly capture environment dynamics and adapt through latent representations, leading to more robust decision-making in complex settings.

Model Rollout and Supervised Training Objective. As detailed in Section III.A, observation $o_t = (o_t^1, o_t^2, o_t^3)$ is transformed to a latent state space via the encoder h_θ:

$$z_t = h_\theta(o_t)$$

This latent representation captures task-relevant dynamics and structure from laser scans and robot states.

Given the current latent state $\mathbf{z}_t$ and action $\mathbf{a}_t \sim \pi_\theta(\mathbf{z}_t)$, we predict:

$$\mathbf{z}_{t+1} = d_\theta(\mathbf{z}_t, \mathbf{a}_t), \quad \hat{r}_t = R_\theta(\mathbf{z}_t, \mathbf{a}_t), \quad \hat{q}_t = Q_\theta(\mathbf{z}_t, \mathbf{a}_t)$$

The model is unrolled over H steps using stored transitions $\Gamma = \{(\mathbf{s}_i, \mathbf{a}_i, r_i)\}_{i=t}^{t+H}$.

Loss Function. We jointly train the reward model, Q-function, and latent dynamics with a temporally discounted objective over trajectories $\mathcal{J} \sim \mathcal{B}$:

$$\mathcal{J}(\theta;\Gamma) = \sum_{i=t}^{t+H} \lambda^{i-t}\, \mathcal{L}(\theta;\Gamma_i),$$

where $\Gamma_i = (z_i, a_i, r_i)$ and $z_i = h_\theta(o_i)$, $z_{i+1} = d_\theta(z_i, a_i)$. The single-step loss is

$$\begin{aligned}\mathcal{L}(\theta;\Gamma_i) = c_1\|R_\theta(\mathbf{z}_i,\mathbf{a}_i) - r_i\|^2 \\ + c_2\Big\|Q_\theta(\mathbf{z}_i,\mathbf{a}_i) - \big(r_i + \gamma Q_{\theta^-}(\mathbf{z}_{i+1}, \\ \pi_{\theta^-}(\mathbf{z}_{i+1}))\big)\Big\|^2 \\ + c_3\|d_\theta(\mathbf{z}_i,\mathbf{a}_i) - h_\theta(\mathbf{s}_{i+1})\|^2\end{aligned}$$

with weights c_1, c_2, c_3. The target parameters θ^- are updated slowly to stabilize learning. All multi-step predictions and gradients are computed in latent space, avoiding explicit future observation prediction while enforcing reward/value accuracy and latent consistency.

Reward Function. We design the reward to encourage goal reaching, collision avoidance, and socially compliant behaviors:

$$R(o_t) = R_g(o_t) + R_c(o_t) + R_s(o_t).$$

Goal reward provides terminal success reward and dense shaping by progress:

$$R_g(o_t) = \begin{cases} r_{\text{arrival}}, & \text{if } \|p^t - p^*\| \le \varepsilon \\ w_1(\|p_{t-1} - p^*\| - \|p_t - p^*\|), & \text{otherwise.}\end{cases}$$

Collision penalty penalizes collision and near-obstacle states based on the minimum laser distance d:

$$R_c(o_t) = \begin{cases} r_{\text{collision}}, & \text{if collision occurs} \\ w_2 \left(1 - \frac{d}{r+1.0}\right), & \text{if } d \le r + 1.0 \\ 0, & \text{otherwise.} \end{cases}$$

Social reward encourages yielding at potential interaction points with dynamic agents [10] [11]. Using short-horizon interaction checking, we reward yielding and penalize aggressive crossing:

$$R_s(o_t) \mathrel{-}= \omega_3 \left(1 - \frac{d_{\text{min}}}{r_s}\right), \quad R_s(o_t) \mathrel{+}= \omega_3 \left(1 - \frac{d_{\text{min}}}{r_s}\right),$$

where d_{min} is the predicted minimum distance and r_s is the social radius threshold.

Overall, these terms guide efficient goal reaching with safety and social compliance in dynamic environments [22].

Policy Learning. The policy $\pi_\theta(z)$ is trained with a hybrid objective that integrates entropy-regularized Q-learning and behavior cloning, encouraging both value-driven actions and consistency with trajectories optimized by MPPI. Given a latent rollout $\{z_0, \ldots, z_H\}$ and reference actions $\{a_0^{\text{mppi}}, \ldots, a_H^{\text{mppi}}\}$, the policy minimizes:

$$\mathcal{L}_\pi = \sum_{t=0}^{H} \rho^t \cdot \left[\alpha \log \pi_\theta(a_t|z_t) - Q_\theta(z_t, a_t) + \lambda \|a_t - a_t^{\text{mppi}}\|^2\right],$$

where α balances entropy, λ weights imitation, and ρ discounts over the horizon. This formulation jointly promotes exploration, high-value actions, and adherence to optimized rollouts, yielding a stable and sample-efficient policy.

C. Trajectory Optimization with Smooth MPPI

We employ a sampling-based MPPI optimizer to refine latent-space action sequences with policy-guided initialization and smoothness-aware scoring. At each step, MPPI samples action sequences from a time-varying Gaussian, rolls them out through the learned latent dynamics, and updates the sampling distribution using high-scoring trajectories [7] [23].

MPPI Objective. Each trajectory is scored by predicted multi-step rewards, a terminal Q-value, and a smoothness penalty:

$$\phi_i = \sum_{t=0}^{H-1} \gamma^t R(z_t^{(i)}, a_t^{(i)}) + \gamma^H Q(z_H^{(i)}, a_H^{(i)}) - \lambda_{\text{smooth}} \hat{\mathcal{P}}_i,$$

where $\hat{\mathcal{P}}_i$ penalizes abrupt changes between consecutive actions.

Policy-Guided Sampling. To improve sample efficiency, we initialize a subset of candidates from the current policy π_θ and sample the rest randomly, balancing exploitation and exploration.

Sub-trajectory Execution. Instead of executing only the first action, we execute a random prefix $\{a_0^*, \ldots, a_{L-1}^*\}$ of the optimized sequence, with $L \sim \text{Uniform}(1, H)$, before replanning. This encourages consistency over longer horizons and prevents overly greedy short-term actions.

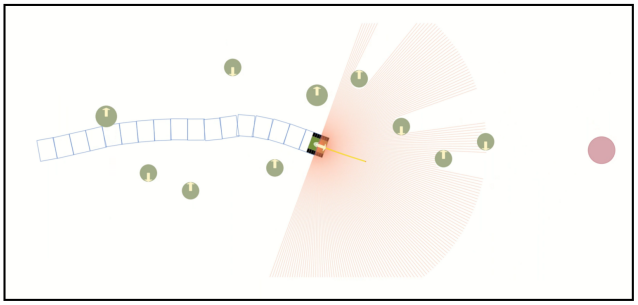

Fig. 3: Illustration of the IRSim simulation environment. The robot (black rectangle) navigates toward the goal (red circle) using LiDAR scans (red fan shape) while avoiding static and dynamic obstacles (green circles).

IV. Experiments

We evaluate our proposed method across both simulated and real-world navigation tasks, focusing on dynamic environments with moving obstacles. All the experiments are conducted without any prior map, relying solely on the onboard LiDAR for navigation and exploration.

A. Implementation and Training Details

All components in our framework are implemented in PyTorch using deterministic networks. Except for the observation encoder, which adopts a lightweight 4-layer 3D CNN with increasing channel widths and temporal-spatial pooling to encode multi-frame laser-based local obstacle maps, all other models—including the dynamics model, reward predictor, Q-value estimator, and policy network—are implemented as multi-layer perceptrons (MLPs) with 256-dimensional latent features. We use the Adam optimizer with a learning rate of 1×10^{-4} for the encoder and dynamics model, and 3×10^{-4} for the remaining networks.

The planning horizon is set to $H = 6$, and during training, the agent executes sub-trajectories of random length $L \sim \mathcal{U}(1, H)$ before re-planning, which improves the stability of long-horizon learning. In the planning phase, we use Model Predictive Path Integral (MPPI) optimization with 6 iterations per step. Each iteration samples 512 candidate trajectories from a time-dependent Gaussian distribution, and the top 64 trajectories are selected to update the mean and variance based on their combined reward and Q-value scores.

Training is conducted entirely in the IRSIM simulation platform using onboard LiDAR inputs, without any prior map. We linearly anneal the exploration parameter ϵ from 0.5 to 0.05 over the first 25k decision steps. All experiments are run on a single NVIDIA RTX 4090 GPU.

B. Comparative Study

All simulations are conducted in the IRSim platform like Figure 3, which supports realistic robot dynamics and interaction with moving obstacles. We provide a thorough evaluation of the performance of our social navigation framework with end-to-end model-based reinforcement learning by comparing

TABLE I: Performance under Interaction-Aware Obstacle Motion in Open Environment

	Uni-modal Slow				Uni-modal Fast				Multi-modal Slow				Multi-modal Fast			
	Succ.↑	SPL↑	Smooth↓	Inference.↓	Succ.↑	SPL↑	Smooth↓	Inference.↓	Succ.↑	SPL↑	Smooth↓	Inference.↓	Succ.↑	SPL↑	Smooth↓	Inference.↓
NEUPAN	1.00	0.84	0.10	0.39	0.96	0.81	0.11	0.39	0.95	0.83	0.11	0.39	0.75	0.62	0.13	0.39
SAC	0.97	0.82	0.18	0.013	0.94	0.80	0.18	0.013	0.90	0.82	0.17	0.013	0.71	0.59	0.19	0.013
TD-MPC-style	0.98	0.92	0.15	0.017	0.95	0.87	0.15	0.017	0.92	0.81	0.16	0.017	0.83	0.73	0.19	0.017
OURS H=6 E=6	1.00	0.98	**0.10**	**0.002**	0.97	0.90	0.10	**0.002**	0.97	0.87	0.11	**0.002**	0.78	0.72	**0.11**	**0.002**
OURS H=6 E=1	**1.00**	**0.98**	0.11	0.017	**0.98**	**0.94**	**0.11**	0.017	**0.98**	**0.95**	**0.10**	0.017	**0.90**	**0.88**	0.12	0.017

TABLE II: Performance under Non-Interaction Obstacle Motion in Open Environment

	Uni-modal Slow				Uni-modal Fast				Multi-modal Slow				Multi-modal Fast			
	Succ.↑	SPL↑	Smooth↓	Inference.↓	Succ.↑	SPL↑	Smooth↓	Inference.↓	Succ.↑	SPL↑	Smooth↓	Inference.↓	Succ.↑	SPL↑	Smooth↓	Inference.↓
NEUPAN	1.00	0.87	0.10	0.39	0.98	0.85	0.10	0.39	0.97	0.83	0.11	0.39	0.92	0.81	0.12	0.39
SAC	0.97	0.83	0.16	0.013	0.96	0.82	0.16	0.013	0.92	0.79	0.17	0.013	0.90	0.76	0.19	0.013
TD-MPC-style	0.98	0.95	0.16	0.019	0.96	0.91	0.16	0.019	0.94	0.86	0.18	0.019	0.90	0.80	0.19	0.019
OURS H=6 E=6	1.00	0.98	0.10	**0.002**	0.98	0.92	**0.10**	**0.002**	0.97	0.88	0.10	**0.002**	0.93	0.82	0.11	**0.002**
OURS H=6 E=1	**1.00**	**0.98**	**0.10**	0.017	**1.00**	**0.95**	0.11	0.017	**1.00**	**0.97**	**0.10**	0.017	**0.96**	**0.92**	**0.11**	0.017

with existing approaches. We vary the usage strategy under a planning horizon of $H = 6$, comparing two execution modes: re-planning after one step ($E = 1$) or executing the full trajectory ($E = 6$).

To compare performance, we report:

- **Success Rate (%):** Percentage of successful goal-reaching trials without collision over 100 runs.
- **SPL (Success weighted by Path Length):** Standard path efficiency metric.
- **Inference Time (s):** Average algorithm inference time.
- **Smoothness (↓):** Mean squared action variation, $\frac{1}{T-1}\sum_{t=1}^{T-1}\|a_t - a_{t-1}\|_2^2$ with $a_t = [v_t, \omega_t]$; lower indicates less jitter.
- **Social Safety (↓):** We report the average number of cutting-off events per episode, and additionally Avg. MinDist and TTC violation rate in open environments.

With this definition, Smooth= 0.10 vs. 0.16 corresponds to about a 60% increase in average squared action changes, which typically manifests as sharper turns or more stop-and-go behavior.

We compare against the following baselines:

- **Soft Actor-Critic (SAC):** A model-free deep RL algorithm known for sample efficiency, serving as our value-based baseline. It learns directly from LiDAR observations without explicit trajectory modeling [4] [9].
- **NEUPAN:** A model-based navigation framework using Plug-and-Play Proximal Alternating Minimization for end-to-end trajectory optimization, avoiding intermediate perception stages [6].
- **TD-MPC-style:** A latent MPC baseline that ranks candidate action sequences using predicted multi-step rewards and a terminal value, implemented with the same observation encoder and network capacity for fair comparison. [18].

We evaluate in three representative navigation scenarios:

- **Corridor with Wall:** A narrow hallway with constrained geometry and moving obstacles.
- **Open Environment (Slow Obstacles):** An open space with low-speed dynamic obstacles.
- **Open Environment (Fast Obstacles):** An open space with high-speed dynamic obstacles, requiring more anticipatory behavior.

To evaluate the effectiveness of our proposed latent-space rollout and reward/Q prediction strategy, we design comprehensive comparative experiments under varying dynamic environments. We consider two key dimensions of obstacle behavior:

- **Interaction-aware vs. Non-interaction Obstacle Motion:** In the *interaction-aware* setting, obstacles are equipped with simple avoidance strategies and react to the robot's motion (e.g., actively avoiding collision), while in the *non-interaction* setting, obstacles move directly toward their goals regardless of the robot's trajectory.
- **Uni-modal vs. Multi-modal Obstacle Dynamics:** In the *uni-modal* setting, obstacles move toward a single fixed goal without deviation. In the *multi-modal* setting, each obstacle has a probability of changing direction midway—either returning to its original position or heading toward a new randomly sampled goal—introducing randomness and behavioral ambiguity.

These variations allow us to test the robustness and adaptability of our method under increasingly challenging real-world-like conditions. We compare our approach against NEUPAN (a model-based method using decoupled environment prediction), SAC (a model-free baseline), and a TD-MPC-style latent MPC baseline.

The results in Tables I, II, and Figure 4 offer compelling evidence for the advantages of our latent-space, implicit reward prediction approach over traditional model-based and model-free baselines.

Superior Interaction Modeling. First, we observe that our method consistently outperforms NEUPAN across all settings, especially under interaction-aware and multi-modal obstacle dynamics. NEUPAN, being a decoupled prediction-based model, attempts to simulate the environment in isolation from the robot's motion. This can lead to loss of critical robot-environment interaction information, especially in scenes with feedback loops such as avoidance and dynamic yielding. In contrast, our approach leverages joint latent rollouts condi-

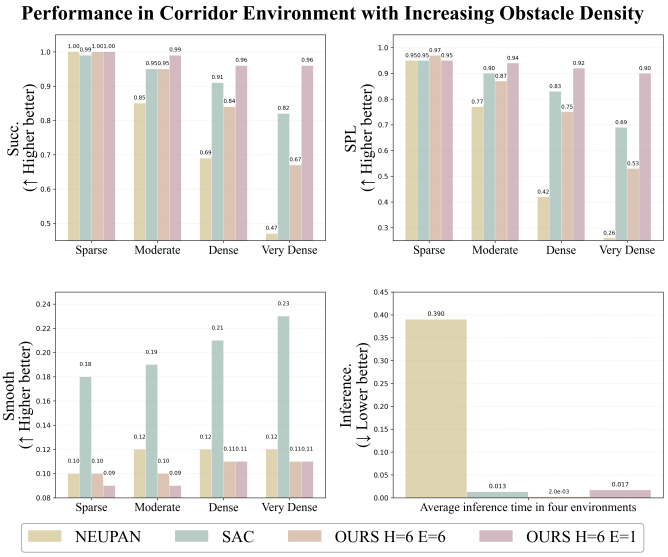

Fig. 4: Performance in corridor environments with increasing obstacle density. **Obstacle settings:** Sparse (2 static, 0 dynamic), Moderate (3 static, 2 dynamic), Dense (3 static, 4 dynamic), Very Dense (4 static, 6 dynamic).

tioned on robot actions, enabling accurate modeling of environment evolution as influenced by the robot itself. Compared with the TD-MPC-style baseline, our method remains stronger under interaction-aware dynamics, benefiting from the explicit social reward and the smoothness-aware trajectory evaluation.

Robustness to Stochastic Dynamics. Second, under stochastic and multi-modal obstacle motion, our framework exhibits remarkable robustness. NEUPAN and SAC suffer significant performance drops due to their reliance on either deterministic environment modeling or value-only returns, which fail to capture the uncertainty. TD-MPC-style is more competitive, but still degrades in highly stochastic multi-modal cases without explicit regularization for socially safe and smooth behaviors. Our multi-step reward prediction, together with the social reward and smoothness-aware evaluation, yields more stable trajectory scoring and more consistent performance.

Effective Hybrid Planning in Constrained Spaces. Third, in constrained and complex environments such as narrow corridors with high obstacle density, our hybrid planning strategy—combining cumulative multi-step reward and terminal Q-value—achieves superior balance between success rate and path efficiency. Unlike NEUPAN's limited-step reward accumulation, our combined reward+Q optimization allows the robot to reason over both short-term safety and long-term goal progress, yielding smoother and more socially acceptable trajectories even under tight spatial constraints.

Superior Sample Efficiency. To further compare the sample efficiency between model-free and model-based approaches, we plot the training curves of SAC and our proposed method in Figure 5. While SAC requires a large number of episodes to converge, our method achieves comparable or

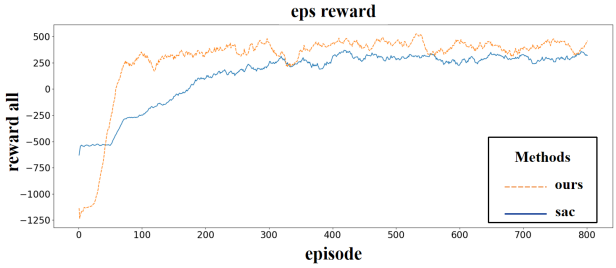

Fig. 5: Training reward curves under the same environment. Our method achieves faster convergence and higher final episode rewards compared to SAC, indicating improved learning efficiency and overall performance stability.

superior success rates with significantly fewer episodes.

Overall, our results demonstrate that latent-space rollouts with joint reward–Q evaluation enable superior interaction modeling, robustness to stochastic dynamics, effective planning in constrained spaces, and higher sample efficiency. These advantages validate the core insight of our approach: avoiding explicit prediction while leveraging implicit, interaction-aware latent modeling leads to more robust and efficient navigation.

C. Horizon Length and Execution Strategy Study

We investigate the impact of planning horizon length H and action execution strategy on navigation performance. Specifically, we compare three different horizons: $H = 3$, $H = 6$, and $H = 10$, and evaluate two execution strategies: executing the entire planned trajectory ($E = H$) and executing only the first action at each step ($E = 1$).

TABLE III: Effect of Horizon Length and Execution Strategy (Very Dense Corridor Scenario)

Method	Success Rate↑	SPL↑	Smooth↓
OURS H=3 E=3	0.73	0.71	0.11
OURS H=3 E=1	0.91	0.89	**0.10**
OURS H=6 E=6	0.60	0.57	0.15
OURS H=6 E=1	**0.96**	**0.90**	0.11
OURS H=10 E=10	0.52	0.48	0.17
OURS H=10 E=1	0.87	0.83	0.13

As shown in Table III, horizon length strongly influences performance. Short horizons (e.g., $H = 3$) limit foresight and cause reactive decisions, while long horizons (e.g., $H = 10$) increase prediction errors, reducing success rate and path quality. The best trade-off is achieved with $H = 6$ and `Use 1`, where only the first action is executed and replanned at each step. This balances planning depth with uncertainty, yielding stable and robust trajectories. Hence, we adopt $H = 6$ and first-step-only execution in our main experiments.

Discussion on horizon and approximation error. Our score uses an H-step truncated return with a bootstrapped tail, $\hat{J} = \sum_{t=0}^{H-1} \gamma^t \hat{R}_t + \gamma^H \hat{Q}_H$. With bounded errors $|\hat{R}_t - R_t| \le \epsilon_R$ and $|\hat{Q}_H - Q_H| \le \epsilon_Q$, we have

$$|\hat{J} - J| \le \frac{1-\gamma^H}{1-\gamma}\epsilon_R + \gamma^H \epsilon_Q, \tag{1}$$

illustrating the trade-off that larger H reduces reliance on $\hat{Q}$ but accumulates more rollout/reward errors, consistent with Table III.

D. Ablation Studies

We conduct ablation experiments to analyze the contributions of four key components in our planning framework: the MPPI refinement step, the smoothness penalty, the combination of multi-step reward with Q-value optimization, and the effectiveness of our proposed social reward mechanism.

TABLE IV: Ablation Study on MPPI Refinement (Very Dense Corridor Scenario)

Method	Success Rate↑	SPL↑	Smooth↓
Policy-only (no MPPI)	0.93	0.85	0.19
MPPI (no policy prior)	0.90	0.82	0.14
Ours (policy prior + MPPI)	**0.96**	**0.90**	**0.11**

Table IV isolates the impact of MPPI refinement by comparing policy-only execution, MPPI without a learned policy prior, and the full policy-prior-guided MPPI planner.

TABLE V: Ablation Study on Smoothness Penalty (Very Dense Corridor Scenario)

Method	Success Rate↑	SPL↑	Smooth↓
OURS H=6 E=6 w/o Smooth	0.59	0.57	0.18
OURS H=6 E=1 w/o Smooth	**0.97**	**0.92**	0.17
OURS H=6 E=6 w/ Smooth	0.60	0.57	0.15
OURS H=6 E=1 w/ Smooth	0.96	0.90	**0.11**

TABLE VI: Ablation Study on Reward and Q-value Components (Very Dense Corridor Scenario)

Method	Success Rate↑	SPL↑	Smooth↓
SAC (Q-only)	0.91	0.83	0.18
Reward-only	0.94	0.89	0.15
Ours (Reward + Q)	**0.96**	**0.90**	**0.11**

Table V highlights the role of the smoothness penalty in generating natural and jitter-free trajectories. Without the penalty term, although the robot can still reach the goal, the resulting paths tend to exhibit unstable or abrupt changes, reducing the smoothness and interpretability of motion. Enabling the smoothness constraint significantly reduces action variation (and thus abrupt turning/stop-and-go behaviors) while maintaining high success rates.

We perform an ablation study to evaluate the individual and joint contributions of multi-step reward prediction and terminal Q-value estimation within the MPPI planning framework. As shown in Table VI, we compare the following variants in the corridor scenario:

- **SAC (Q-only):** A baseline that selects trajectories purely based on terminal Q-values, similar to standard value-based reinforcement learning.
- **Reward-only:** A variant that removes Q-value estimation and ranks trajectories using cumulative predicted rewards over a finite horizon.
- **Ours (Reward + Q):** Our full method that combines short-term multi-step reward estimation with long-term Q-value guidance for trajectory scoring.

The results show that SAC can be effective but may suffer from unstable long-horizon value estimates, leading to suboptimal and less smooth behaviors. The reward-only variant improves short-term decisions but lacks long-term foresight. By combining multi-step rewards with a terminal Q-value, our hybrid objective achieves the best Succ./SPL and smoothness, validating the dual-objective optimization for complex navigation.

TABLE VII: Social-reward ablation in Open Environment.

Method	Succ↑	SPL↑	CutOff↓	MinDist↑	TTC$<$ 1s↓
OURS (H=6,E=6) w/o Soc.	0.74	0.67	6.11	0.68	10.1
OURS (H=6,E=6) w/ Soc.	**0.91**	**0.83**	**2.32**	**0.96**	**4.3**

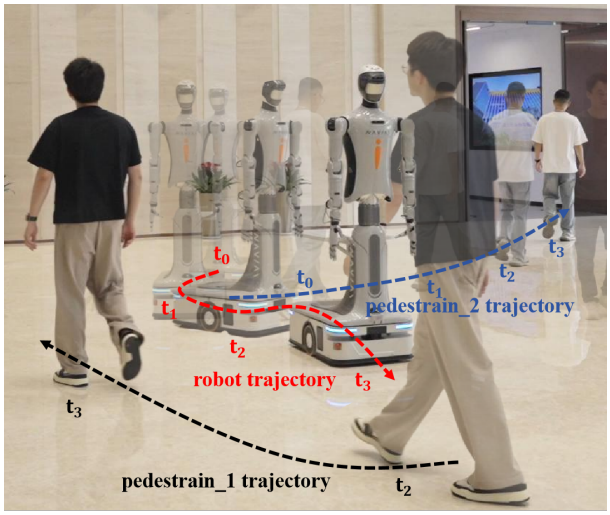

Fig. 6: Real-world navigation with dynamic pedestrians. Red: robot; blue/black: pedestrians. Markers t_0–t_3 show the interaction over time, where the robot adapts its path for safe, smooth, and socially compliant navigation.

To evaluate the proposed social reward, we perform an ablation in open-space navigation with dynamic pedestrians. Table VII shows that removing it decreases Succ./SPL (0.91/0.83 $\rightarrow$ 0.74/0.67) and worsens social safety: cutting-off rises (2.32 $\rightarrow$ 6.11 /ep), MinDist drops (0.96 $\rightarrow$ 0.68 m), and TTC$<$ 1.0s increases (4.3% $\rightarrow$ 10.1%). This indicates the social reward promotes earlier yielding and proactive deceleration.

E. Real-World Experiments

We deploy our method on a real differential-drive robot equipped with a 2D LiDAR, operating in a public indoor envi-

ronment that includes dynamic pedestrians and static obstacles such as walls, desks. The robot navigates between predefined goal locations without access to any prior map, relying purely on onboard sensing and local decision-making via our end-to-end framework.

Extensive real-world testing demonstrates the robustness of our approach, achieving a navigation success rate of over 0.9 across more than 50 missions, while maintaining collision-free operation for over 5 hours in dynamic settings.

As shown in Fig. 6, the robot successfully completes navigation tasks in challenging scenes with moving pedestrians and narrow passages. The visualized trajectories illustrate the robot's ability to anticipate potential conflicts, adjust its motion proactively, and maintain smooth and collision-free behavior across a variety of dynamic situations.

V. Conclusion

This paper presents a model-based reinforcement learning framework that performs trajectory planning entirely in a compact latent space, avoiding explicit prediction of future observations. By combining multi-step reward prediction with terminal Q-value estimation, our method enables robust trajectory evaluation. With smoothness-aware MPPI optimization, it generates dynamically feasible trajectories in complex and uncertain environments. Real-world experiments further show that a lightweight social reward promotes safe and polite navigation among dynamic obstacles. Overall, implicit interaction-aware latent-space prediction offers an effective and scalable solution for real-world navigation.

References

[1] Z. Chen and Y. Li, "Fdspc: Fast and direct smooth path planning via continuous curvature integration," *ArXiv*, vol. abs/2405.03281, 2024. [Online]. Available: https://api.semanticscholar.org/CorpusID:269605391

[2] C. Xiong, Y. Huang, F. Yu, C. Chen, Y. Wang, S. Xia, and L. Pei, "Sensing, social, and motion intelligence in embodied navigation: A comprehensive survey," 2025. [Online]. Available: https://api.semanticscholar.org/CorpusID:280700152

[3] A. H. Qureshi, Y. Miao, A. Simeonov, and M. C. Yip, "Motion planning networks: Bridging the gap between learning-based and classical motion planners," *IEEE Transactions on Robotics*, vol. 37, pp. 48–66, 2019. [Online]. Available: https://api.semanticscholar.org/CorpusID:196622676

[4] T. Haarnoja, A. Zhou, K. Hartikainen, G. Tucker, S. Ha, J. Tan, V. Kumar, H. Zhu, A. Gupta, P. Abbeel, and S. Levine, "Soft actor-critic algorithms and applications," *ArXiv*, vol. abs/1812.05905, 2018. [Online]. Available: https://api.semanticscholar.org/CorpusID:55703664

[5] C. Li, A. Krause, and M. Hutter, "Robotic world model: A neural network simulator for robust policy optimization in robotics," *ArXiv*, vol. abs/2501.10100, 2025. [Online]. Available: https://api.semanticscholar.org/CorpusID:275606663

[6] R. Han, S. Wang, S. Wang, Z. Zhang, J. Chen, S. Lin, C. Li, C. Xu, Y. C. Eldar, Q. Hao, and J. Pan, "Neupan: Direct point robot navigation with end-to-end model-based learning," *IEEE Transactions on Robotics*, vol. 41, pp. 2804–2824, 2024. [Online]. Available: https://api.semanticscholar.org/CorpusID:268364034

[7] H. Liu, Y. Feng, W. Dong, K. Fan, C. Wang, and Y. Gao, "Hierarchical learning-enhanced mpc for safe crowd navigation with heterogeneous constraints," *ArXiv*, vol. abs/2506.09859, 2025. [Online]. Available: https://api.semanticscholar.org/CorpusID:279305482

[8] S. Kousik, B. Zhang, P. Zhao, and R. Vasudevan, "Safe, optimal, real-time trajectory planning with a parallel constrained bernstein algorithm," *IEEE Transactions on Robotics*, vol. 37, pp. 815–830, 2020. [Online]. Available: https://api.semanticscholar.org/CorpusID:211989889

[9] Y. Cui, H. Zhang, Y. Wang, and R. Xiong, "Learning world transition model for socially aware robot navigation," *2021 IEEE International Conference on Robotics and Automation (ICRA)*, pp. 9262–9268, 2020. [Online]. Available: https://api.semanticscholar.org/CorpusID:226281698

[10] Y. Cui, X. Huang, Y. Wang, and R. Xiong, "Socially-aware multi-agent following with 2d laser scans via deep reinforcement learning and potential field," *2021 IEEE International Conference on Real-time Computing and Robotics (RCAR)*, pp. 515–520, 2021. [Online]. Available: https://api.semanticscholar.org/CorpusID:237378790

[11] Y. Cui, L. Lin, X. Huang, D. Zhang, Y. Wang, and R. Xiong, "Learning observation-based certifiable safe policy for decentralized multi-robot navigation," *2022 International Conference on Robotics and Automation (ICRA)*, pp. 5518–5524, 2021. [Online]. Available: https://api.semanticscholar.org/CorpusID:237532368

[12] A. Francis, A. Faust, H.-T. L. Chiang, J. Hsu, J. C. Kew, M. Fiser, and T.-W. E. Lee, "Long-range indoor navigation with prm-rl," *IEEE Transactions on Robotics*, vol. 36, pp. 1115–1134, 2019. [Online]. Available: https://api.semanticscholar.org/CorpusID:67855822

[13] L. Li, Y. Miao, A. H. Qureshi, and M. C. Yip, "Mpc-mpnet: Model-predictive motion planning networks for fast, near-optimal planning under kinodynamic constraints," *IEEE Robotics and Automation Letters*, vol. 6, pp. 4496–4503, 2021. [Online]. Available: https://api.semanticscholar.org/CorpusID:231632344

[14] B. Zhang, R. Rajan, L. Pineda, N. Lambert, A. Biedenkapp, K. Chua, F. Hutter, and R. Calandra, "On the importance of hyperparameter optimization for model-based reinforcement learning," in *International Conference on Artificial Intelligence and Statistics*, 2021. [Online]. Available: https://api.semanticscholar.org/CorpusID:232069022

[15] D. Yarats, R. Fergus, A. Lazaric, and L. Pinto, "Mastering visual continuous control: Improved data-augmented reinforcement learning," *ArXiv*, vol. abs/2107.09645, 2021. [Online]. Available: https://api.semanticscholar.org/CorpusID:236134152

[16] R. Sun, H. Zang, X. Li, and R. Islam, "Learning latent dynamic robust representations for world models," *ArXiv*, vol. abs/2405.06263, 2024. [Online]. Available: https://api.semanticscholar.org/CorpusID:269740932

[17] N. Hansen, X. Wang, and H. Su, "Temporal difference learning for model predictive control," in *International Conference on Machine Learning*, 2022. [Online]. Available: https://api.semanticscholar.org/CorpusID:247318709

[18] N. Hansen, H. Su, and X. Wang, "Td-mpc2: Scalable, robust world models for continuous control," *ArXiv*, vol. abs/2310.16828, 2023. [Online]. Available: https://api.semanticscholar.org/CorpusID:264451720

[19] C. Chen, Y. Liu, S. Kreiss, and A. Alahi, "Crowd-robot interaction: Crowd-aware robot navigation with attention-based deep reinforcement learning," *2019 International Conference on Robotics and Automation (ICRA)*, pp. 6015–6022, 2018. [Online]. Available: https://api.semanticscholar.org/CorpusID:52812470

[20] Y. Qu, H. Chu, S. Gao, J. Guan, H. Yan, L. Xiao, S. E. Li, and J. Duan, "Rl-driven mppi: Accelerating online control laws calculation with offline policy," *IEEE Transactions on Intelligent Vehicles*, vol. 9, pp. 3605–3616, 2024. [Online]. Available: https://api.semanticscholar.org/CorpusID:266669474

[21] Z. Han, Y. Wu, T. Li, L. Zhang, L. Pei, L. Xu, C.-A. Li, C. Ma, C. Xu, S. Shen, and F. Gao, "An efficient spatial-temporal trajectory planner for autonomous vehicles in unstructured environments," *IEEE Transactions on Intelligent Transportation Systems*, vol. 25, pp. 1797–1814, 2022. [Online]. Available: https://api.semanticscholar.org/CorpusID:258048901

[22] T. Fan, P. Long, W. Liu, and J. Pan, "Distributed multi-robot collision avoidance via deep reinforcement learning for navigation in complex scenarios," *The International Journal of Robotics Research*, vol. 39, pp. 856 – 892, 2020. [Online]. Available: https://api.semanticscholar.org/CorpusID:219636651

[23] P. Wang, C. Li, C. Weaver, K. Kawamoto, M. Tomizuka, C. Tang, and W. Zhan, "Residual-mppi: Online policy customization for continuous control," *ArXiv*, vol. abs/2407.00898, 2024. [Online]. Available: https://api.semanticscholar.org/CorpusID:270870032