2024 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)
October 14-18, 2024. Abu Dhabi, UAE

Towards Accurate And Robust Dynamics and Reward Modeling for Model-Based Offline Inverse Reinforcement Learning

Gengyu Zhang, Yan Yan

Abstract— **This paper enhances model-based offline inverse reinforcement learning (IRL) by refining conservative Markov decision process (MDP) frameworks, traditionally employing uncertainty penalties to deter exploitation in uncertain areas. Existing methods, dependent on neural network ensembles to model MDP dynamics and quantify uncertainty through ensemble prediction heuristics, face limitations: they presume Gaussian-distributed state transitions, leading to simplified environmental representations. Additionally, ensemble modeling often results in high variance, indicating potential overfitting and a lack of generalizability. Moreover, the heuristic reliance for uncertainty quantification struggles to fully grasp environmental complexities, offering an incomplete foundation for informed decisions. Maintaining multiple models also demands substantial computational resources. Addressing these shortcomings, we propose leveraging score-based diffusion generative models for dynamic modeling. This method significantly broadens the scope of representable target distributions, surpassing Gaussian constraints. It not only improves the accuracy of transition modeling but also roots uncertainty quantification in diffusion models' theoretical underpinnings, enabling more precise and dependable reward regularization. We further innovate by incorporating a transition stability regularizer (*TSR*) into the reward estimation. This novel element embeds stability into the reward learning process, diminishing the influence of transition variability and promoting more consistent policy optimization. Our empirical studies on diverse Mujoco robotic control tasks demonstrate that our diffusion-based methodology not only furnishes more accurate transition estimations but also surpasses conventional ensemble approaches in policy effectiveness. The addition of the *TSR* marks a distinctive advancement in offline IRL by enhancing the reward and policy learning efficacy. Code: https://github.com/GabrielZH/doc-irl.**

I. INTRODUCTION

Inverse Reinforcement Learning (IRL) [2], [18], [22] plays a crucial role in scenarios where direct reward signals are unavailable, and crafting reward functions manually is prohibitively challenging [17], [1]. By inferring the latent reward function that underpins observed decision-making processes, IRL aims to replicate expert behavior, offering insights into complex decision-making under uncertainty. Within the IRL domain, model-based offline approaches [32], [33], [31], [30], [3], [20] use an estimated dynamics model to predict action outcomes without interacting with the real environment. This feature enhances sample efficiency, extends applicability to unseen states, and reduces exploration risks inherent in online methods [19], [21]. These benefits are particularly appealing for applications with limited online experimental opportunities.

However, current model-based offline IRL methods often use ensembles of probabilistic neural networks to approximate the enviroment's dynamics model [16], [13]. This approach introduces two limitations: **(i)** It assumes a Gaussian distribution for state transitions, denoted as $\{\hat{T}_i(s'|s,a) = \mathcal{N}(\mu_i(s,a),\Sigma_i(s,a))\}_{i=1}^{N}$, where N represents the ensemble size. This assumption restricts the model's ability to accurately capture the complex dynamics of real-world environments. **(ii)** These methods use heuristics, such as variance or standard deviation across ensemble outputs, to ifyquant uncertainty. This practice lacks a solid theoretical foundation and may yield misleading results in the presence of outliers. To address these issues, we propose several key innovations in this paper:
(i) Diffusion-Based Dynamics Modeling (*DDM*): We incorporate diffusion generative models, particularly those based on stochastic differential equations (SDEs) and ordinary differential equations (ODEs) [26], [12], to go beyond Gaussian assumptions. This approach enables modeling a broader range of distributions, enhancing the fidelity and robustness of dynamics modeling. To our knowledge, we are the first to use diffusion models in this context.
(ii) Systematic Uncertainty Quantification: We propose an original framework for quantifying both aleatoric and epistemic uncertainties. Our approach provides a theoretical basis for quantifying aleatoric uncertainty underpinned by *DDM*. For epistemic uncertainty, we use Bayesian Neural Networks (BNNs). This method eliminates the reliance on ensemble networks, enhancing computational efficiency.
(iii) Stability-Aware Reward Learning: We refine reward and policy learning by incorporating a transition stability regularizer (*TSR*) based on predicted next states from *DDM*. This regularizer discourages the policy from interacting with unstable state transitions, i.e., uncertain state-action pairs, thereby promoting safer policy learning.

Our empirical evaluation in Mujoco [27] environments using D4RL [5] benchmarks demonstrates the effectiveness of our approach. We enhance model accuracy and enable better decision-making under uncertainty, outperforming existing model-based offline IRL techniques. By integrating diffusion models with a robust uncertainty quantification framework, our work addresses key limitations of previous methods. This advancement leads to more precise and efficient IRL models, paving the way for future research in the field.

II. RELATED WORK AND PRELIMINARIES

A. Model-Based Offline IRL with Maximum Entropy Principle

IRL aims to deduce the underlying reward function that guides expert behavior [17]. As one of its important branches, offline IRL leverages a static dataset to avoid the need for direct environmental interaction. This property facilitates

Department of Computer Science, University of Illinois Chicago, IL, USA.

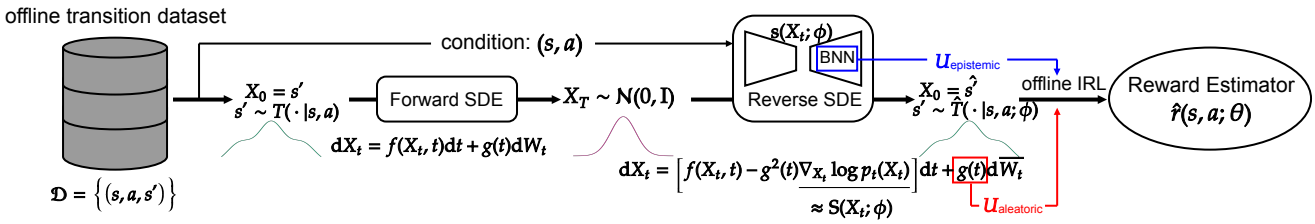

Fig. 1: **Overview of the framework architecture.** This figure depicts the two core components of our framework—the diffusion dynamics estimator and the reward estimator. The diffusion dynamics estimator, parameterized by ϕ and leveraging an offline transition dataset $\mathscr{D}$, seeks to accurately model the underlying transition distribution T with an approximated distribution $\hat{T}$, utilizing a ScoreSDE. Subsequently, the estimated dynamics and derived uncertainties inform the conservative reward and policy optimization performed by the reward estimator, parameterized by θ.

learning in scenarios where direct interaction is impractical or hazardous. Particularly, model-based offline IRL differs from its model-free counterpart by explicitly estimating underlying environmental dynamics. This enhances the interpretability and simulation of action outcomes, which is crucial for understanding complex decision processes.

The incorporation of the maximum entropy (MaxEnt) principle [34] in model-based IRL signifies a notable advancement. It introduces a probabilistic approach to decision-making by optimizing the reward function to maximize the entropy of the derived policy. This approach handles uncertainties in expert behavior and provides more accurate modeling of dynamics. Recent methodologies that adopt MaxEnt principles further improve model-based offline IRL [32], [33], enabling the inference of adaptable reward functions that capture the inherent variability and rationality of decision processes.

Model-based offline IRL that is underpinned by the MaxEnt principle relies on dynamics modeling and uncertainty quantification for accurate reward inference. This is crucial when dealing with limited or noisy data. The estimated dynamics model facilitates the evaluation of inferred rewards and policies without direct environmental interaction. Besides, uncertainty quantification helps regularizing learned rewards, guiding policy optimization towards actions with higher certainty. This approach strengthens the theoretical basis of model-based offline IRL and aligns with principled learning and decision-making under uncertainty.

B. Score-Based Diffusion Generative Models

Diffusion generative models split into two main approaches: score matching with Langevin dynamics (SMLD) [24], [28], [25] and denoising diffusion probabilistic models (DDPM) [23], [10]. Both approaches are based on the idea of incrementally perturbing data with noise at multiple scales and learning a reverse process to recover noiseless samples from the target distribution. The two approaches represent different facets of the same model family, each providing a unique perspective.

SMLD estimates the score function, a key component that guides the reverse diffusion process. This approach facilitates exact log-likelihood calculation and provides a direct solution to inverse problems, connecting it to energy-based models [4]. On the other hand, DDPM reverses a discrete diffusion process through a Markov chain [6]. This is achieved by optimizing the evidence lower bound (ELBO) [11] and using a learned decoder for sampling, which relates it to variational autoencoders (VAEs) [14], [15] and lossy compression.

It turns out that the ELBO used for training the DDPM is essentially equivalent to the weighted combination of score matching objectives used in SMLD. Based on this fact, Song et al. [26] unify the two approaches under a framework that extends the concept of noise scales to a continuum, effectively converging both methodologies to SDEs. This unification leads to a more robust predictor-corrector sampler, combining the sampling efficacy of DDPM with the annealed Langevin dynamics [29] inherent in score-based models.

SDEs control the addition or removal of noise through a continuous process, as detailed in Eq. 1. $\mathbf{f}(\cdot,\cdot)$ and $g(\cdot)$ represent the drift and diffusion coefficients, respectively. $\mathbf{W}_t$ denotes stochastic noise via the Wiener process. The reverse process involves simulating the removal of noise to recover the original data distribution, using the backward Wiener process and the score function for denoising.

$$\begin{aligned} \mathrm{d}\mathbf{X}_t &= \mathbf{f}(\mathbf{X}_t, t)\mathrm{d}t + g(t)\mathrm{d}\mathbf{W}_t \\ \mathrm{d}\mathbf{X}_t &= [\mathbf{f}(\mathbf{X}_t, t) - g(t)^2 \nabla_{\mathbf{X}_t} \log p_t(\mathbf{X}_t)]\mathrm{d}t + g(t)\mathrm{d}\bar{\mathbf{W}}_t \end{aligned} \tag{1}$$

The objective of Eq. 1 is to learn the score function across various noise levels to enable sample generation by inversely navigating the noise addition trajectory. We can applying various solutions, such as the Euler-Maruyama method [9], to incrementally remove noise from data, ensuring samples conform to the learned data distribution. For clarity, throughout this manuscript, we refer to score-based diffusion generative models governed by SDEs as ScoreSDEs.

Deterministic sampling via probability flow (PF) ODEs, which recover the same distribution as SDEs in theory, enjoys fast adaptive sampling and exact likelihood computation. By excluding the diffusion term, PF ODEs focus on deterministic dynamics as shown in Eq.2. However, deterministic sampling typically results in worse output quality. Karras et al. [12] discover the significance of stochasticity in practice, which is driving samples towards the desired marginal distributions at any time and correcting errors made in earlier sampling steps. In this light, they optimize the design space across different model families and propose a novel stochastic sampler for better sampling efficacy. This refined approach emphasizes the versatility of score-based methods in modeling data distributions, whether through stochastic or deterministic pathways.

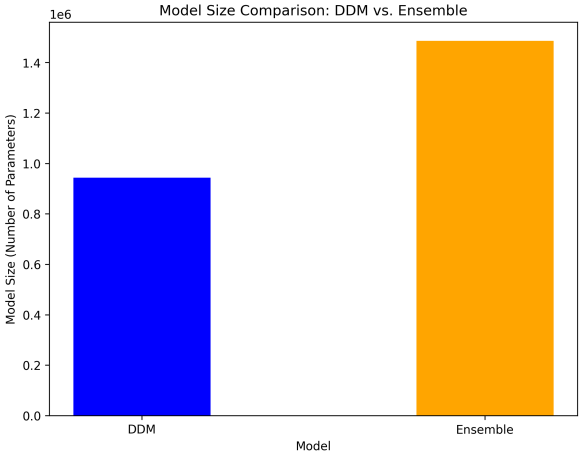

Fig. 2: **Model size regarding number of parameters for the two dynamics modeling methods.** The ensemble transition model's size is the product of a single model's size and the ensemble size. This makes it significantly larger than *DDM*, which requires only a single model.

Hence, it offers a robust framework for understanding and modeling complex data distributions. Similar to the naming of ScoreSDEs, we refer to the diffusion models driven by PF ODEs as ScoreODEs. We will also use ODEs to refer to PF ODEs in this paper's context.

$$\mathrm{d}\mathbf{X}_t = \left[\mathbf{f}(\mathbf{X}_t, t) - \frac{1}{2} g(t)^2 \nabla_{\mathbf{X}_t} \log_{p_t}(\mathbf{X}_t)\right] \mathrm{d}t. \tag{2}$$

III. Methodology

Our model-based offline IRL framework comprises two core components—the *DDM* and a reward estimator. The reward estimator depends on the dynamics model estimated by the *DDM* and the uncertainty quantified based on it. We adopt ScoreSDEs for a high-fidelity representation of state transitions, which offers theoretically guaranteed measure for aleatoric uncertainty. For epistemic uncertainty, we employ Bayesian inference instead of the commonly used ensemble networks. This substitution maintains the precision of uncertainty quantification while avoiding training multiple networks, which saves computational resources and provides more controllable variance estimation. The *DDM* and our proposed uncertainty quantification method help us better analyze the environment, thereby improving reward estimation. We further introduce transition stability, measured by the variance of the *DDM*'s predictions, as a regularizer for reward estimation. This ensures a more reliable and predictable reward and policy outcome.

A. Diffusion-Based Dynamics Modeling

The essence of *DDM* is to transfer the probabilistic sampling of Markovian state transition into a conditional denoising generation problem. Specifically, we formalize this as generating a desired next state s' by denoising Gaussian noise conditioned on the observed current state s and action a. During training, we use the ground-truth next states from offline datasets as labels to train the network in a supervised manner. Our objective function for approximating the underlying dynamics model with a ScoreSDE, defined by the denoising score matching objective, is formalized as:

$$\mathscr{J}_{\mathrm{DSM}}(\phi) = \int_0^T \mathbb{E}_{(s,a,s')\sim\mathscr{D}, \tilde{s}'\sim p(\cdot|s';\sigma(t))} \left[\mathscr{L}(s', \tilde{s}', \sigma(t); \phi)\right] \mathrm{d}t, \tag{3}$$

where

$$\mathscr{L}\left(s', \tilde{s}', \sigma(t); \phi\right) = \left\| \nabla_{\tilde{s}'} \log p(\tilde{s}'|s'; \sigma(t)) - \mathbf{S}\left(\tilde{s}', \sigma(t); \phi\right) \right\|_2^2.$$

Here, $p(\cdot|s';\sigma(t))$ denotes the transition density of perturbed states $\tilde{s}'$ under the noise level $\sigma(t)$. Whenever the drift coefficient $\mathbf{f}(s',t)$ is linear in s', $p(\cdot|s';\sigma(t))$ is a tractable distribution [26]. $\mathbf{S}\left(\tilde{s}', \sigma(t); \phi\right)$ represents the learned score by the model. The expectation is taken over state-action-next-state triplets from the dataset $\mathscr{D}$ and perturbed states following the specified noise schedule. This formulation captures the essence of diffusion dynamics and emphasizes the model's adaptability across various noise levels, ensuring a comprehensive understanding of the underlying dynamics.

Another way is to optimize the sample denoising objective, where the network acts as a denoiser. In this case, the network outputs a denoised sample rather than the estimated score function. We can follow the successful practice of the preconditioning method proposed in EDM [12] for better training. For inference, the second order sampler, known as Heun's method, provides better sampling quality than Eular's first-order method. We formally define the sampling denoising objective as:

$$\mathscr{J}_{\mathrm{SD}}(\phi) = \mathbb{E}_{(s,a,s')\sim\mathscr{D}} \left[\mathscr{L}(s', \sigma(t); \phi)\right], \tag{4}$$

where

$$\mathscr{L}(s', \sigma(t); \phi) = \mathbb{E}_{\varepsilon\sim\mathscr{N}(\mathbf{0}, \sigma^2\mathbf{I})} \left\| D\left(s' + \varepsilon, \sigma(t); \phi\right) - s' \right\|_2^2.$$

The learned denoiser implicitly captures the score function as Eq. 5 demonstrates. This is adapted from Eq. 3 of EDM [12].

$$\nabla_{s'} \log p\left(s'; \sigma(t)\right) = \frac{D(s', \sigma(t); \phi) - s'}{\sigma(t)^2}. \tag{5}$$

The left part of Fig. 1 shows the *DDM* operations. It illustrates noise addition in the forward SDE process for ground-truth next states s' during training and denoising in the reverse SDE process for predicted next states $\hat{s}'$ during inference. It also highlights the score function's role in guiding the reverse process. *DDM* presents better generalization to unseen states and more robust handling of uncertainties in complex dynamics. By accurately modeling state transitions, it improves reward estimation precision, leading to more effective policy optimization.

In addition to more accurate state transition modeling, *DDM* demonstrates superior efficiency compared to the ensemble approaches. Fig. 2 shows the model sizes of the two approaches in terms of the number of parameters. Since ensemble networks require maintaining multiple models, they are significantly larger than the single diffusion model used by *DDM*. The smaller model size makes *DDM* more adaptable in scenarios where memory and storage are crucial.

Combined with its better modeling performance, *DDM* clearly offers a significant advantage. Please refer to Appendix F for implementation details of *DDM*.

B. Uncertainty Quantification of Dynamics Estimation

To enhance risk-averse reward and policy learning with robust uncertainty quantification, we estimate both aleatoric and epistemic uncertainties within *DDM*. Aleatoric uncertainty, arising from the data noise, stems from the diffusion term in ScoreSDE (Eq. 1) and customized stochastic sampling in ScoreODE, which enables precise mathematical quantification. Epistemic uncertainty is assessed through Bayesian inference, providing more stable variance control than ensemble methods and eliminating the need for training multiple networks.

Theoretically guaranteed quantification of aleatoric uncertainty. ScoreSDEs inherently capture aleatoric uncertainty arising from the stochastic nature of the Wiener process within the diffusion term. The diffusion coefficient $g(t)$ controls the influence of this stochasticity on the model's evolution over time. This shapes the aleatoric uncertainty within the SDE framework. By analyzing $g(t)$, specifically its integrability and boundedness within $[0,T]$, we theoretically establish an upper bound for the uncertainty, as formalized in Theorem 1. This analysis offers a principled approach to assess the variability introduced by stochastic processes within the model.

Theorem 1 (Bounded Aleatoric Uncertainty): Let $g(t)$ be the diffusion coefficient in a ScoreSDE (Eq. 1). If $g(t)$ is integrable over $t \in [0,T]$ and bounded above a constant κ, then the aleatoric uncertainty defined in Eq. 6 is upper bounded.

$$U_{\text{aleatoric}} = \int_0^T g(t)^2 \mathrm{d}t. \tag{6}$$

Please refer to Appendix A for detailed proof.

Theorem 1 demonstrates that aleatoric uncertainty in diffusion models can be effectively controlled if $g(t)$ meets specific design criteria. This property is not only theoretically significant but also practically relevant, as it aligns with design principles in various diffusion generative models, such as score matching with Langevin dynamics (SMLD) [24] and denoising diffusion probabilistic models (DDPM) [23], [10]. Specific designs of these models are as follows:

- SMLD: the continuous stochastic process converges to the Variance Exploding (VE) SDE, $\mathrm{d}\mathbf{X}_t = \sqrt{\frac{\mathrm{d}[\sigma(t)^2]}{\mathrm{d}t}}\mathrm{d}\mathbf{W}_t$, where $\sigma(t)$ is a noise scheduling function of diffusion step indices $t \in [0,1]$.
- DDPM: the continuous stochastic process converges to the Variance Preserving (VP) SDE, $\mathrm{d}\mathbf{X}_t = -\frac{1}{2}\beta(t)\mathbf{X}_t\mathrm{d}t + \sqrt{\beta(t)}\mathrm{d}\mathbf{W}_t$, where $\beta(t)$ is a noise scaling function also indexed by $t \in [0,1]$, the continuous version of the variance sequence of forward diffusion process, $\{\beta_i\}_{i=1}^N$ [10]. There is also an improved version, namely sub-VP SDE, formulated as $\mathrm{d}\mathbf{X}_t = -\frac{1}{2}\beta\mathbf{X}_t\mathrm{d}t + \sqrt{\beta(t)(1-e^{-2\int_0^t\beta(s)\mathrm{d}s})}\mathrm{d}\mathbf{W}_t$.

These implementations show the applicability of the ScoreSDE-based *DDM*, where the aleatoric uncertainty is both theoretically quantifiable and empirically verifiable.

Shifting from ScoreSDEs to ScoreODEs leads to a deterministic sampling method, which initially lacks aleatoric uncertainty due to its deterministic nature. However, this approach often results in lower-quality samples due to the absence of stochastic corrections [26], [12]. To mitigate this, empirical methods reintroduce stochasticity, akin to SDE models, by injecting noise at strategic intervals during the ODE integration process. This adjustment effectively turns the ODE into a stochastic sampler, restoring the ability to quantify the aleatoric uncertainty.

Bayesian inference for quantification of epistemic uncertainty. Bayesian inference enables an analysis of the uncertainty associated with the network's parameters used to model the score function, $\mathbf{S}(\cdot,\cdot;\phi)$, or the denoiser, $D(\cdot,\cdot;\phi)$. We achieve this by implementing Bayesian Neural Networks (BNNs) as base unit of *DDM*, which treats network parameters as distributions instead of fixed values. As such, we can quantify the uncertainty in the network's predictions about the dynamics. This approach is particularly useful when inferring dynamics across different states and actions, allowing for more informed decision-making under uncertainty.

BNNs are optimized through variational inference, which minimizes a divergence measure along with the prediction loss. The networks represent distributions over weights within each layer to capture predictive uncertainty from parameter variability. Sampling from the trained distributions allows for multiple predictions per input, enabling evaluation of variability to quantify epistemic uncertainty. This feature improves model interpretability by providing probabilistic insights into confidence levels and highlights areas needing refinement. In *DDM*, $U_{\text{epistemic}}$ plays a key role in identifying uncertainties arising from data or model structural limitations, which, along with $U_{\text{aleatoric}}$, guides model improvement and informs further data collection efforts.

$U_{\text{aleatoric}}$ in predicting the next state s' given the current state-action pair (s,a) can be conceptually represented as:

$$U_{\text{epistemic}}(s,a) = \mathrm{Var}_{\phi\sim p(\phi|\mathscr{D})}\left[\mathbf{S}(s,a;\phi)\right], \tag{7}$$

where $\mathrm{Var}_{\phi\sim p(\phi|\mathscr{D})}$ denotes the variance over the posterior distribution of the model parameters ϕ given the observed data $\mathscr{D}$. This variance captures the spread of possible next states as predicted by the model, considering all plausible values of ϕ according to the posterior belief $p(\phi|\mathscr{D})$. A higher variance indicates greater uncertainty in the model's predictions due to ambiguity or lack of information in ϕ.

C. Stability-Aware Conservative Reward Learning

Once we reliably quantify the uncertainty in *DDM*, we proceed to solve a conservative MDP to determine the optimal policy while estimating the underlying reward function that guides the offline trajectories. Inspired by Offline ML-IRL [33], our reward learning module alternates between policy improvement and reward optimization at each iteration.

Policy improvement. In this step, we perform conservative policy iteration under *DDM*. We denote the diffusion dynamics model by $\hat{T}(s,a;\phi)$, which can be either the denoiser $D(s,a,\sigma(t);\phi)$ or the stochastic sampler using the approximated score $\mathbf{S}(\mathbf{s},\mathbf{a};\phi)$. The next state s' is sampled from this

TABLE I: **Mujoco dynamics modeling result.** The proposed *DDM* either outperforms or matches the ensemble method across various environments. The performance in the Ant environment is particularly notable, recognized for its significantly larger state and action spaces. In this context, our method substantially surpasses ensemble dynamics modeling in both metrics, emphasizing our approach's superior ability to navigate complex scenarios effectively.

Method	Halfcheetah		Walker2D		Hopper		Ant	
	RMSD ↓	Pearson ↑	RMSD ↓	Pearson ↑	RMSD ↓	Pearson ↑	RMSD ↓	Pearson ↑
Diffusion (ours)	0.710±0.045	0.989±0.001	**0.535±0.026**	**0.979±0.002**	0.136±0.007	0.998±0.001	**0.452±0.010**	**0.884±0.004**
Ensemble	**0.669±0.031**	**0.990±0.001**	0.680±0.030	0.966±0.003	**0.116±0.015**	0.998±0.001	0.852±0.014	0.695±0.010

model. We then solve a conservative MDP formalized as the following Bellman backup equation:

$$Q(s,a) \leftarrow r(s,a;\theta) + \alpha\mathscr{H}(\pi(\cdot|s)) - \delta U(s,a) - \eta Z(s,a) + \gamma\mathbb{E}_{s'\sim\hat{T}(s,a;\phi)}\left[V(s')\right]. \tag{8}$$

In Eq. 8, $r(s,a;\theta)$ represents the estimated reward function parameterized by θ. Our goal is to make $r(s,a;\theta)$ accurate and robust. $\mathscr{H}(\pi(\cdot|s))$ denotes the entropy we seek to maximize in the MaxEnt-IRL. $U = U_{\text{aleatoric}} + U_{\text{epistemic}}$ represents the overall uncertainty estimated for $\hat{T}$. $V(s')$ is the state value function used in traditional MDPs. In addition to uncertainty, we introduce a regularizer, *TSR*, denoted by $Z(s,a)$, that measures the stability of state transitions and enhances the conservative policy learning. The inclusion of *TSR* aligns with the diffusion model's stochastic nature during sampling. Since the reverse diffusion process may generate different samples each time during inference, capturing the variability of the model's outcomes is crucial to assess the stability of state transitions in dynamics modeling. We formalize the quantification of the $Z(s,a)$ as follows:

$$Z(s,a) = \exp\left(-\text{Var}\left[s'_{s'\sim\hat{T}(s,a;\phi)}\right]\right). \tag{9}$$

To ensure consistent state transitions, we penalize the variance in predicted states under $\hat{T}$ using $Z(s,a)$, thereby smoothing modeled dynamics. With all the key components obtained for estimating $Q_k(s,a)$, we then update the policy π_k using soft policy iteration [7], [8]:

$$\pi_k(a|s) \propto \exp\left[Q(s,a)\right]. \tag{10}$$

Reward optimization. In this step, we update the reward parameters to θ_{k+1} based on the current parameters θ_k and policy π_k by taking gradient steps on the objective:

$$\nabla_\theta \mathscr{J}(\theta) = \mathbb{E}_{(s,a)\sim\mathscr{D}^E}\left[\nabla_\theta r(s,a;\theta)\right] - \mathbb{E}_{(s,a)\sim\pi}\left[\nabla_\theta r(s,a;\theta)\right]. \tag{11}$$

Alternating the policy improvement and reward optimization enables efficient optimization for the objective function[33].

Our reward estimator captures underlying reward functions that reflect expert behavior, prioritizing safety and predictability. By considering transition stability, our approach accounts for both the stability and uncertainty of dynamics. It not only adheres to foundational reward optimization principles but also adapts to complex environments.

IV. Experimentation and Discussion

Our experiments, including those for dynamics modeling, reward estimation and policy optimization of the baselines and our own approach, are all conducted on the same Mujoco locomotion environments using the same version of D4RL benchmarks. The environments comprises Ant, Halfcheetah, Walker2d, and hopper, each containing offline trajectory data collected by three types of policies, including expert, expert-medium, and medium. The versions of the datasets are consistently *v2*. For more details, please refer to Appendix E.

A. Dynamics Model Estimation

Our experimentation begins by evaluating the accuracy and robustness of approximated dynamics models across various MuJoCo environments, as summarized in Table I. We compare the *DDM* with a traditional ensemble approach using root mean squared deviation (RMSD) and Pearson correlation coefficients on a held-out test set unseen during training. The *DDM* consistently outperforms the ensemble method across different environments, suggesting its superior ability to capture the complex, non-linear transitions. This is particularly evident in the Walker2d and Ant environments, where *DDM* shows a significant improvement in RMSD and Pearson correlation, indicating high prediction accuracy and stronger correlation with true state transitions. The ensemble model performs better in environments with potentially less complex dynamics, such as Halfcheetah and Hopper, where it shows a slight edge in RMSD scores. However, the diffusion model exhibits lower deviations in both metrics across three out of four environments, highlighting its consistency and reliability in modeling diverse environments.

As shown in Fig. 3, the diffusion model achieves faster convergence in simpler environments like Hopper and Halfcheetah, reaching stable accuracy with fewer training steps than

TABLE II: **MuJoCo reward gain results.**

Dataset type	Env	Off. ML-IRL [33]	CLARE [32]	Ours
medium	Ant	2058.46±358.88	1118.39±218.00	**2330.81±178.01**
medium	Halfcheetah	**7568.51±225.79**	2028.21±344.06	7243.90±497.83
medium	Hopper	1592.11±616.00	**3015.37±474.48**	2089.00±221.73
medium	Walker2d	3870.18±531.32	3263.28±160.62	**4008.65±362.00**
expert-medium	Ant	3027.22±509.98	2494.97±417.53	**3368.24±223.89**
expert-medium	Halfcheetah	11129.60±555.35	4509.00±433.35	**11211.83±459.43**
expert-medium	Hopper	1874.49±625.99	**2336.08±245.78**	1949.64±322.17
expert-medium	Walker2d	3876.19±745.65	2491.64±470.64	**4004.73±261.80**
expert	Ant	3718.28±486.13	2627.43±257.77	**4320.35±296.57**
expert	Halfcheetah	**11890.86±428.94**	4594.58±257.96	11280.79±427.37
expert	Hopper	**3115.11±253.36**	2773.59±282.11	2918.14±440.65
expert	Walker2d	3953.80±548.66	2360.87±410.09	**4149.06±218.33**

ensemble models. This efficiency underscores the model's superior ability to effectively capture environment's dynamics. In more complex environments, such as Ant and Walker2d, the diffusion model demonstrates significantly higher accuracy. Notably, in the Ant environment, the traditional ensemble model plateaus at a Pearson correlation of around 0.7 since the start, while *DDM* rapidly improves and converges to a significantly higher correlation of around 0.9. The diffusion model's efficiency, combined with its comparable or superior performance across various settings, highlights the benefits of employing *DDM* for modeling complex systems.

B. Reward Estimation and Policy Optimization

We evaluate the reward gains across the four MuJoCo locomotion environments by comparing our framework with two established baselines. As shown in Table II, our method consistently achieves superior or comparable performance, particularly in environments using both medium and expert datasets. Our approach demonstrates significant improvements in reward gains, especially in the expert-medium Ant and Walker2d environments, where it surpasses the benchmarks set by Offline ML-IRL with margins of 341.02 and 128.54 points. This highlights our method's capability to integrate detailed dynamics and precise uncertainty quantification, enabling effective exploration of complex scenarios.

Moreover, the variability in reward gains, indicated by standard deviations, suggests that our method not only enhances reward acquisition but also ensures consistency and reliability. This consistent performance across diverse environments and datasets underscores the benefit of leveraging *DDM* and *TSR*. It is clear that a deeper understanding of environment's dynamics and uncertainties, as facilitated by our method, leads to substantial improvements in complex scenarios. The reduced deviations across results further demonstrate the robustness of our approach, indicating advancements in both reward acquisition but and the stability of learning outcomes.

C. Ablation Study

The ablation study, presented in Table III, elucidates the contributions of each key component of our methodology. By introducing *DDM* and incrementally integrating *TSR* into our baselines, Offline ML-IRL and CLARE, we observe remarkable improvements in performance across all environments compared to their original versions. This shows that *DDM* enhances the model's understanding of environments' dynamics, while *TSR*, combined with comprehensively quantified uncertainty, improves policy regularization and boosts policy effectiveness. The most notable performance increase occurs when transitioning from original CLARE to CLARE+*DDM*+*TSR*, especially in the Halfcheetah environment. This highlights the synergistic effect of our method on reward estimation and policy derivation.

The ablation study highlights the individual and collective impact of *DDM* and *TSR* on the reward estimation process. The improvements across configurations and environments validate the effectiveness of our integrated approach. Additionally, the reduced standard deviations in the full version of our configuration across all environments indicate that *TSR* not only enhances performance but also contributes to the reliability and robustness of the learned policies.

TABLE III: **Ablation study.** The numbers present the enhancement in performance across three MuJoCo environments due to the integration of *DDM* and stability regularization (*TSR*), demonstrating improvements over offline ML-IRL and CLARE. The full version of our approach, shown in the first row, is essentially offline ML-IRL+*DDM*+*TSR*.

Ablation	Ant	Halfcheetah	Walker2d
Ours	**3368.24±223.89**	**11211.83±459.43**	**4004.73±261.80**
Off. ML-IRL original	3027.22±509.98	11129.60±555.35	3876.19±745.75
Off. ML-IRL+DDM	3256.45±355.92	11198.45±475.22	3922.34±396.19
CLARE original	2494.97±417.53	4509.00±433.35	2491.64±470.64
CLARE+*DDM*	2669.87±267.14	6079.41±447.33	2517.13±390.26
CLARE+*DDM*+*TSR*	2862.96±348.46	7643.55±457.20	2914.71±385.19

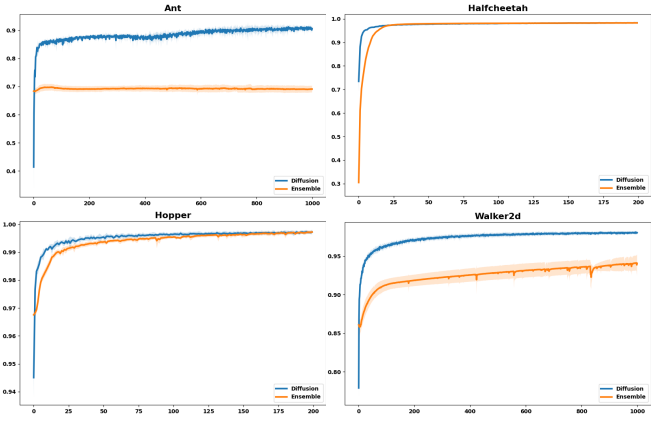

Fig. 3: **Convergence rate regarding Pearson correlation of two dynamics modeling methods across four MuJoCo environments.** The diffusion model converges at states with significantly higher scores in the Ant and Walker2d environments. In the remaining environments, while achieving similar scores, the diffusion model exhibits a faster convergence.

V. CONCLUSIONS

This paper addresses the long neglected limitation in model-based offline IRL by introducing *DDM* underpinned by ScoreSDEs and ScoreODEs. Transcending the Gaussian assumption for stochastic state transitions, this approach enhances modeling accuracy and provides a robust framework for uncertainty quantification. The introduction of *TSR* further ensures a certain and stable exploration of state-action space. Empirical results support the superiority of the proposed method over existing works in robotic control tasks, demonstrating more precise transition estimations and reduced variance in dynamics modeling and advancements in policy effectiveness, marking a substantial contribution to the field's ongoing development and application of IRL.

Acknowledgments: This research is supported by NSF IIS-2309073 and ECCS-2123521. This article solely reflects the opinions and conclusions of authors and not funding agencies.

REFERENCES

[1] P. Abbeel and A. Ng, "Apprenticeship learning via inverse reinforcement learning," in *ICML*, 2004.

[2] S. Arora and P. Doshi, "A survey of inverse reinforcement learning: Challenges, methods and progress," *Artificial Intelligence*, 2021.

[3] N. Das, S. Bechtle, T. Davchev, D. Jayaraman, A. Rai, and F. Meier, "Model-based inverse reinforcement learning from visual demonstrations," in *CoRL*, 2021.

[4] Y. Du and I. Mordatch, "Implicit generation and modeling with energy based models," in *NeurIPS*, 2019.

[5] J. Fu, A. Kumar, O. Nachum, G. Tucker, and S. Levine, "D4rl: Datasets for deep data-driven reinforcement learning," 2020.

[6] P. A. Gagniuc, *Markov chains: from theory to implementation and experimentation*. John Wiley & Sons, 2017.

[7] T. Haarnoja, H. Tang, P. Abbeel, and S. Levine, "Reinforcement learning with deep energy-based policies," in *ICML*, 2017.

[8] T. Haarnoja, A. Zhou, P. Abbeel, and S. Levine, "Soft actor-critic: Off-policy maximum entropy deep reinforcement learning with a stochastic actor," in *ICML*, 2018.

[9] N. Halidias and P. E. Kloeden, "A note on the euler–maruyama scheme for stochastic differential equations with a discontinuous monotone drift coefficient," *BIT Numerical Mathematics*, 2008.

[10] J. Ho, A. Jain, and P. Abbeel, "Denoising diffusion probabilistic models," in *NeurIPS*, 2020.

[11] M. Hoffman and M. Johnson, "Elbo surgery: yet another way to carve up the variational evidence lower bound," in *NeurIPS*, 2016.

[12] T. Karras, M. Aittala, T. Aila, and S. Laine, "Elucidating the design space of diffusion-based generative models," in *NeurIPS*, 2022.

[13] R. Kidambi, A. Rajeswaran, P. Netrapalli, and T. Joachims, "Morel: Model-based offline reinforcement learning," in *NeurIPS*, 2020.

[14] D. P. Kingma and M. Welling, "Auto-encoding variational bayes," in *ICLR*, 2014.

[15] D. P. Kingma, S. Mohamed, D. Jimenez Rezende, and M. Welling, "Semi-supervised learning with deep generative models," in *NeurIPS*, 2014.

[16] B. Lakshminarayanan, A. Pritzel, and C. Blundell, "Simple and scalable predictive uncertainty estimation using deep ensembles," in *NeurIPS*, 2017.

[17] A. Ng and S. Russell, "Algorithms for inverse reinforcement learning," in *ICML*, 2000.

[18] S. Russell, "Learning agents for uncertain environments," in *COLT*, 1998.

[19] R. Self, M. Abudia, and R. Kamalapurkar, "Online inverse reinforcement learning for systems with disturbances," in *ACC*, 2020.

[20] R. Self, M. Abudia, S. N. Mahmud, and R. Kamalapurkar, "Model-based inverse reinforcement learning for deterministic systems," *Automatica*, 2022.

[21] R. Self, S. N. Mahmud, K. Hareland, and R. Kamalapurkar, "Online inverse reinforcement learning with limited data," in *CDC*, 2020.

[22] Z. Shao and E. M. Joo, "A survey of inverse reinforcement learning techniques," *IJICC*, 2012.

[23] J. Sohl-Dickstein, E. Weiss, N. Maheswaranathan, and S. Ganguli, "Deep unsupervised learning using nonequilibrium thermodynamics," in *ICML*, 2015.

[24] Y. Song and S. Ermon, "Generative modeling by estimating gradients of the data distribution," in *NeurIPS*, 2019.

[25] Y. Song, S. Garg, J. Shi, and S. Ermon, "Score matching: a scalable approach to density and score estimation," in *UAI*, 2019.

[26] Y. Song, J. Sohl-Dickstein, D. Kingma, A. Kumar, S. Ermon, and B. Poole, "Score-based generative modeling through stochastic differential equations," in *ICLR*, 2021.

[27] E. Todorov, T. Erez, and Y. Tassa, "Mujoco: a physics engine for model-based control," in *IROS*, 2012.

[28] P. Vincent, "A connection between score matching and denoising autoencoders," *Neural computation*, 2011.

[29] M. Welling and Y. W. Teh, "Bayesian learning via stochastic gradient langevin dynamics," in *ICML*, 2011.

[30] T. Yu, A. Kumar, R. Rafailov, A. Rajeswaran, S. Levine, and C. Finn, "Combo: Conservative offline model-based policy optimization," in *NeurIPS*, 2021.

[31] T. Yu, G. Thomas, L. Yu, S. Ermon, J. Y. Zou, S. Levine, C. Finn, and T. Ma, "Mopo: Model-based offline policy optimization," in *NeurIPS*, 2020.

[32] S. Yue, G. Wang, W. Shao, Z. Zhang, S. Lin, J. Ren, and J. Zhang, "CLARE: Conservative model-based reward learning for offline inverse reinforcement learning," in *ICLR*, 2023.

[33] S. Zeng, C. Li, A. Garcia, and M. Hong, "When demonstrations meet generative world models: a maximum likelihood framework for offline inverse reinforcement learning," in *NeurIPS*, 2023.

[34] B. D. Ziebart, A. L. Maas, J. A. Bagnell, A. K. Dey *et al.*, "Maximum entropy inverse reinforcement learning." in *AAAI*, 2008.

APPENDIX

A. Proof of Theorem 1

Proof: We first prove the boundedness of $g(t)^2$. Given that $g(t)$ is bounded, there exists a constant $\kappa > 0$ such that for all $t \in [0,T]$, $|g(t)| \leq \kappa$. Consequently, $g(t)^2 \leq \kappa^2$. Next, we prove the integrability of $g(t)^2$. The boundedness of $g(t)^2$ implies its integrability over the compact interval $[0,T]$ because a bounded function on a compact interval is always Lebesgue integrable. Specifically, we can write:

$$\int_0^T g(t)^2 \mathrm{d}t \leq \int_0^T \kappa^2 \mathrm{d}t = \kappa^2 T. \tag{12}$$

The theorem is proved. ■

B. Bounded Aleatoric Uncertainty for VE SDE And Proof

Theorem 2 (Bounded Aleatoric Uncertainty for VE SDE): In VE SDE, $g(t)$ is defined as:

$$g(t) = \sigma_{\min} \left(\frac{\sigma_{\max}}{\sigma_{\min}}\right)^t \sqrt{2 \log \frac{\sigma_{\max}}{\sigma_{\min}}}, \tag{13}$$

and $\sigma(t)$ is defined as:

$$\sigma(t) = \sigma_{\min} \left(\frac{\sigma_{\max}}{\sigma_{\min}}\right)^t \quad \text{for} \quad t \in (0,1]. \tag{14}$$

The aleatoric uncertainty is quantified as:

$$\int_0^1 g(t) \mathrm{d}t. \tag{15}$$

Given all the conditions above, the aleatoric uncertainty is upper bounded within $[0,1]$.

Proof: First, we explicitly express $g(t)$ to facilitate integration:

$$g(t)^2 = \left[\sigma_{\min} \left(\frac{\sigma_{\max}}{\sigma_{\min}}\right)^t \sqrt{2 \log \frac{\sigma_{\max}}{\sigma_{\min}}}\right]^2 \tag{16}$$

$$= \sigma_{\min}^2 \left(\frac{\sigma_{\max}}{\sigma_{\min}}\right)^{2t} \cdot 2 \log \frac{\sigma_{max}}{\sigma_{\min}}. \tag{17}$$

Integrating $g(t)$ over $[0,1]$, we get:

$$\int_0^1 g(t)^2 \mathrm{d}t = \int_0^1 \sigma_{\min}^2 \left(\frac{\sigma_{\max}}{\sigma_{\min}}\right)^{2t} \cdot 2 \log \left(\frac{\sigma_{\max}}{\sigma_{\min}}\right) \mathrm{d}t \tag{18}$$

$$= 2\sigma_{\min}^2 \log \left(\frac{\sigma_{\max}}{\sigma_{\min}}\right) \int_0^1 \left(\frac{\sigma_{\max}}{\sigma_{\min}}\right)^{2t} \mathrm{d}t \tag{19}$$

$$= 2\sigma_{\min}^2 \log \left(\frac{\sigma_{\max}}{\sigma_{\min}}\right) \cdot \frac{\left(\frac{\sigma_{\max}}{\sigma_{\min}}\right)^2 - 1}{2 \log \left(\frac{\sigma_{\max}}{\sigma_{\min}}\right)} \tag{20}$$

$$= \sigma_{\min}^2 \left[\left(\frac{\sigma_{\max}}{\sigma_{\min}}\right)^2 - 1\right]. \tag{21}$$

This final expression provides a clear, finite value for the integral of $g(t)^2$ over $[0,1]$, demonstrating that the aleatoric uncertainty is upper bounded for the VE SDE. The upper bound is determined by the chosen values of $\sigma_{\max}$ and $\sigma_{\min}$, and is finite as long as these values are finite and $\sigma_{\max} > \sigma_{\min}$. ■

C. Bounded Aleatoric Uncertainty for VP SDE And Proof

Theorem 3: Given that $g(t) = \sqrt{\beta(t)}$, and assuming $\beta(t)$ is bounded and integrable over $[0,T]$, ensuring $g(t)$ is also bounded and integrable.

Proof: Since $\beta(t)$ is bounded, there exists a constant $M > 0$ such that $\beta(t) \leq M$ for all $t \in [0,T]$. Thus, $g^(t) = \beta(t) \leq M$, and $g(t)$ is square-integrable over $[0,T]$. We have:

$$\int_0^T g(t)^2 \mathrm{d}t \leq MT, \tag{22}$$

which confirms the aleatoric uncertainty is upper bounded. ■

D. Bounded Aleatoric Uncertainty for Sub-VP SDE And Proof

Theorem 4: In sub-VP, we have:

$$g(t) = \sqrt{\beta(t)(1 - e^{-2\int_0^t \beta(s)ds})}. \tag{23}$$

Assume $\beta(t)$ and its integral are bounded over $[0,T]$, making $g(t)$ bounded and integrable.

Proof: Since $\beta(t)$ is bounded, there exists a constant $M > 0$ such that $\beta(t) \leq M$ for all $t \in [0,T]$. The term $1 - e^{-2\int_0^t \beta(s)ds}$ is also bounded considering the properties of exponential functions. Specifically, for $x > 0$, e^{-x} decreases monotonically and ranges from 1 (as $x = 0$) to 0 (as $x \to \infty$). Thus, we always have:

$$0 \leq e^{-2\int_0^t \beta(s)ds} \leq 1. \tag{24}$$

Given the integrability and bounded nature of $\beta(t)$ over $[0,T]$, we can assert that $2\int_0^t \beta(s)\mathrm{d}s$ is also bounded since it is a linear scaling of an integrable, bounded function. Consequently, we get:

$$0 \leq 1 - e^{-2\int_0^t \beta(s)ds} \leq 1. \tag{25}$$

Now, we have:

$$g(t)^2 = \beta(t)(1 - e^{-2\int_0^t \beta(s)ds}) \leq 1 \quad \text{when} \quad e^{-2\int_0^t \beta(s)ds} = 0. \tag{26}$$

Therefore, the upper bound of $g(t)^2$ over $[0,T]$ Therefore, the upper bound of $\beta(t)$, which is M.

Finally, the integral representing aleatoric uncertainty can be mathematically bounded as:

$$\int_0^T g(t)^2 \mathrm{d}t \leq \int_0^T M \mathrm{d}t = M \cdot T. \tag{27}$$

■

E. Dataset Details

The D4RL offline datasets used for each MuJoCo locomotion environment are listed in the table below. All our experiments, including dynamics modeling and reward estimation for CLARE, ML-IRL, and our method, are conducted on these datasets. The models are trained on a portion of the training set from each listed dataset and tested on the corresponding held-out test set.

F. Implementation Details

In our implementation of *DDM*, the core component is a denoiser that converts noise into a valid state representation. As discussed in I, within the context of MDP dynamics, the state-action pair at the current timestep (s,a) serves as the condition for the conditional diffusion process. The state and action tensors are concatenated to form (s,a) as the input, which is then combined with the diffusion timestep embedding, encoded using a sinusoidal function. This combined input, along with noise sampled from a standard Gaussian distribution, is fed into the denoiser. The network architecture of the denoiser is detailed as follows.

Denoiser (Residual MLP Block)

- **Input:** noisy sample `x (state_dim)`
- **Input:** (s,a) `(state_dim + action_dim)`
- **Input:** diffusion timestep t `(scalar)`

TABLE IV: Datasets used for this paper's experiments.

Env Name	Policy Type	Dataset Name
Ant	expert	ant-expert-v2
	expert-medium	ant-medium-expert-v2
	medium	ant-medium-v2
Halfcheetah	expert	halfcheetah-expert-v2
	expert-medium	halfcheetah-medium-expert-v2
	medium	halfcheetah-medium-v2
Hopper	expert	hopper-expert-v2
	expert-medium	hopper-medium-expert-v2
	medium	hopper-medium-v2
Walker2d	expert	walker2d-expert-v2
	expert-medium	walker2d-medium-expert-v2
	medium	walker2d-medium-v2

- **Condition Embedding:**
  - Input 1: (s,a) `(state_dim + action_dim)`
  - Input 2: t `(scalar)`
  - Time-Embed Layer: `Sinusoidal(Input 2) (time_dim)`
  - Op: `Concat(Input 1, Input 2) (cond_dim + time_dim))`
  - Activation: `Swish`
  - Cond-Embed Layers: `FC(Activation(Op)) (cond_dim + time_dim, hidden_dim)`
  - Output: `cond_emb (hidden_dim)`
- **MLP Block 1:**
  - Input: `x (state_dim)`
  - Hidden Layers: `FC(Input) (state_dim, [hidden_dim]*3)`
  - Norm: `LayerNorm`
  - Activation: `Swish`
  - Output: `MLP_1_res (hidden_dim)`
- **Fusion**:
  - Input 1: `MLP_1_res (hidden_dim)`
  - Input 2: `cond_embed (hidden_dim)`
  - Op: `Add(Input 1, Input 2)`
  - Output: `fusion_res_1 (hidden_dim)`
- **MLP Block 2:**
  - Input: `fusion_res_1 (hidden_dim)`
  - Hidden Layers: `FC(Input) ([hidden_dim]*3, state_dim)`
  - Activation: `Swish`
  - Norm: `LayerNorm`
  - Output: `MLP_2_res (state_dim)`
- **Residual:**
  - Input: `x (state_dim)`
  - Hidden Layers `FC(Input) (state_dim, state_dim)`
  - Output: `residual_emb (state_dim)`
- **Fusion**:
  - Input 1: `MLP_2_res (hidden_dim)`
  - Input 2: `residual_emb (hidden_dim)`
  - Op: `Add(Input 1, Input 2)`
  - Output: `fusion_res_2 (state_dim)`
- **Output:** predicted s' *(fusion_res_2)* `(state_dim)`

During inference, the *DDM* module generates the next state step by step. First, a noise sequence is created based on the predefined schedule of EDM [12]. This noise is then refined through a sampling loop. At each timestep, the module adds small perturbations, uses the denoiser to compute updates, and refines the predicted state using the Heun method. This iterative process repeats until the final state is predicted, ensuring that the result stays within a valid range.