NOWCASTDIT: DIFFUSION TRANSFORMERS ARE EFFECTIVE PRECIPITATION NOWCASTERS

Anonymous authors
Paper under double-blind review

ABSTRACT

Precipitation nowcasting demands accurate short-term forecasts under strong spatiotemporal variability. Diffusion models are well suited to modeling complex precipitation distributions, yet existing approaches often introduce increasingly specialized designs, leaving the capability of a standard diffusion architecture underexplored. We show that a *standard* Diffusion Transformer already provides a simple and scalable foundation for precipitation nowcasting, with domain-specific requirements accommodated naturally within its design space. Based on this principle, we develop NowcastDiT and instantiate this flexibility through two complementary adaptations: a dynamics-aware noise prior for temporally coherent forecasts, and end-to-end reinforcement learning with timestep-aware rewards for meteorological skill. Experiments on SEVIR and MRMS benchmarks show that NowcastDiT achieves state-of-the-art performance in both perceptual quality and meteorological skill. These results suggest that standard DiT can serve as a effective foundation for precipitation nowcasting.

1 INTRODUCTION

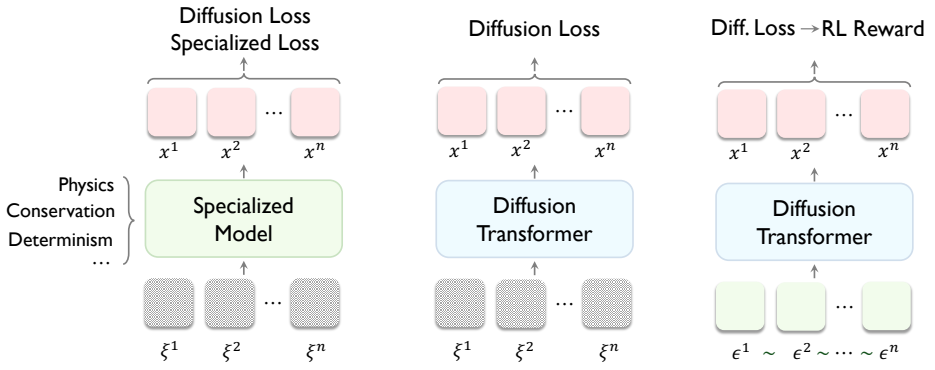
Precipitation nowcasting aims to predict the short-term evolution of rainfall fields at high spatial and temporal resolutions. An effective forecasting system is expected to accurately capture complex spatiotemporal dynamics, account for intrinsic uncertainty, and maintain reliability for rare but high-impact precipitation events (Wang et al., 2017). From a generative modeling perspective, precipitation nowcasting is fundamentally a conditional generation task, with diffusion models (Ho et al., 2020; Song et al., 2020; Lipman et al., 2023) providing a natural framework given their strong capability for probabilistic image and video generation(Peebles & Xie, 2023; Wan et al., 2025; Seedance et al., 2026; Agarwal et al., 2026).

Beyond these general modeling capabilities, existing works have focused on several precipitation-specific requirements, particularly temporal coherence across forecast frames and meteorological skill in capturing high-impact events. To address them, existing generative methods introduce various task-specific designs including physics-based motion constraints (Zhang et al., 2023), intensity continuity regularization (Gao et al., 2023), and decomposition-based modeling frameworks (Yu et al., 2024; Gong et al., 2024; Wen et al., 2026). Despite their different forms, these methods follow a common design philosophy: specializing the generative framework to explicitly encoding these requirements. While effective, this line of development narrows the exploration of foundational model designs for precipitaion nowcasting, leaving a more fundamental question insufficiently explored: *how necessary is task-specific specialization beyond a standard diffusion model?*

In this paper, we argue that a standard Diffusion Transformer provides a sufficient foundation for precipitation nowcasting. We posit that precipitation nowcasting is not fundamentally different from conditional video generation in terms of its core generative modeling requirements, as both require capturing complex spatiotemporal dynamics under uncertainty, capabilities that have already been demonstrated by modern video diffusion models (Seedance et al., 2026; Agarwal et al., 2026). Meanwhile, precipitation-specific requirements can be accommodated within the broader design space of diffusion models, as suggested by the versatility of modern diffusion models across diverse video generation tasks (Guo et al., 2025; Xue et al., 2025)

To this end, we propose the Nowcasting Diffusion Transformer (NowcastDiT), a simple and scalable framework built on a standard DiT with minimal precipitation-specific adaptations. From the diffu-

Reviewers: please read the Reviewer Guidelines (iclr.cc/Conferences/2027/ReviewerGuidelines) and the AI Policy for Reviewers (iclr.cc/Conferences/2027/AIPolicyForReviewers). If you used AI to expand, edit, or polish your review, please provide the input text to the LLM. Better still, consider skipping the LLM and submitting your original text: we, and the authors, are much more interested in your unedited thoughts than in what an LLM has to say. AI-assisted or not, you are putting your name and reputation behind your review: LLM-generated falsehoods, hallucinations or misrepresentations are subject to disciplinary action, which may include desk-rejecting all papers you have authored.



(a) Specialized Diffusion Nowcasters (b) Standard DiT (c) NowcastDiT (Ours)

Figure 1: **Comparison of diffusion modeling paradigms for precipitation nowcasting. (a):** Specialized diffusion nowcasters encode domain knowledge through task-specific models or losses. **(b):** A standard DiT is applicable for precipitation nowacsting. **(c):** NowcastDiT adds minimal precipitation-specific adaptation towards a standard DiT without changing its backbone.

sion perspective, NowcastDiT introduces a dynamics-aware noise prior that captures shared evolution patterns and variations across forecast frames to enforce temporal coherence. From the training perspective, NowcastNet incorporates end-to-end reinforcement learning with timestep-aware rewards to enhance meteorological skill. On two widely recognized radar-based benchmarks, SEVIR and MRMS, NowcastDiT achieves state-of-the-art performance across both perceptual quality and meteorological skill metrics. These results suggest that standard DiT can serve as an effective foundation for precipitation nowcasting.

Our contributions are summarized as follows:

- We revisit precipitation nowcasting from the perspective of standard diffusion models, and show that a standard DiT provides a sufficient foundation, with precipitation-specific requirements handled through targeted adaptation rather than model specialization.
- Based on this principle, we propose NowcastDiT, a simple and scalable framework that retains the standard DiT with a dynamics-aware noise prior to promote temporal coherence and reinforcement learning with timestep-aware rewards to enhance meteorological skill.
- Extensive experiments on SEVIR and MRMS demonstrate the state-of-the-art performance of NowcastDiT, validating standard DiT as an effective foundation for precipitation nowcasting.

2 Preliminaries

Precipitation nowcasting aims to predict S future frames $\mathbf{x}^{1:S}$ from past frames $\mathbf{x}^{-S_0:0}$. Following Rombach et al. (2022), we first train a VAE with encoder E and decoder D, then generate $\mathbf{z}_1 = E(\mathbf{x}^S)$ conditioned on $\mathbf{c} = E(\mathbf{x}^{-S_0:0})$ within the latent space. Diffusion models learn to denoise noise-corrupted data through simple linear interpolation, learning a conditional velocity field with the flow-matching objective:

$$\mathbf{z}_t = (1-t)\boldsymbol{\epsilon} + t\mathbf{z}_1, \tag{1}$$

$$\mathcal{L}_{\mathrm{FM}}(\theta) = \mathbb{E}_{\mathbf{z}_1,\mathbf{c},\boldsymbol{\epsilon},t}\left[\left\|\mathbf{v}_\theta(\mathbf{z}_t, t \mid \mathbf{c}) - (\mathbf{z}_1 - \boldsymbol{\epsilon})\right\|_2^2\right], \tag{2}$$

where $t \in [0,1]$ denotes the diffusion timestep and $\boldsymbol{\epsilon} \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$ is Gaussian noise. At inference, we solve $\frac{\mathrm{d}\mathbf{z}_t}{\mathrm{d}t} = \mathbf{v}_\theta(\mathbf{z}_t, t \mid \mathbf{c})$ from $\mathbf{z}_0 = \boldsymbol{\epsilon}$ to $\hat{\mathbf{z}}_1$ and decode $\hat{\mathbf{x}}^{1:S} = D(\hat{\mathbf{z}}_1)$.

3 Revisiting Precipitation Nowcasting Requirements

In this section, we revisit the modeling requirements of precipitation nowcasting and ask whether they necessitate specialized forecasting architectures. We organize the discussion around three aspects: spatiotemporal generative modeling, physical plausibility, and precipitation alignment.

Spatiotemporal generative modeling. Precipitation fields, like natural videos, are continuous spatiotemporal tensors whose structures span multiple scales, from localized convective cells to large-scale frontal systems. A suitable model must capture these hierarchical dependencies while representing the uncertainty of future evolution. The success of DiTs in video generation (OpenAI, 2024; Wan et al., 2025; Seedance et al., 2026) suggest that their scalable spatiotemporal representations provides a suitable foundation for this purpose.

Physical plausibility. Precipitation evolves through atmospheric processes, and plausible forecasts should exhibit consistent motion and physically reasonable temporal evolution. Explicit physical constraints offer one way to encourage such behavior, but they are not the only route. Modern generative models have demonstrated the ability to learn intuitive physics from video (Ali et al., 2025; Ye et al., 2026; Agarwal et al., 2026), suggesting that data-driven DiTs can also capture the coherent motion and local evolution from radar sequences.

Precipitation alignment. Precipitation differs from natural video mainly in its data distribution and task objectives. Its evolution combines large-scale advection with localized growth and decay, with precipitation intensities commonly sparse and long-tailed. Moreover, generic generative objectives do not directly optimize meteorological skills such as Critical Success Index (CSI) and Heidke Skill Score (HSS). These discrepancies therefore call for precipitation-aware adaptation of the generative process and learning objective, but do not by themselves prescribe a specialized model or alter the underlying spatiotemporal modeling problem.

Existing nowcasting methods often couple such domain-specific mechanisms with specialized forecasting models or frameworks. This coupling obscures how much task-specific specialization is genuinely needed beyond the capabilities of a general-purpose generative model. The analysis above suggests a cleaner separation: retain a standard DiT for general spatiotemporal modeling, and introduce specialization only where precipitation departs from generic generation.

Takeaway. A standard DiT provides the core modeling capacity required for precipitation nowcasting, while precipitation-specific requirements can be addressed through targeted adaptations rather than model re-design.

4 NowcastDiT

In this section, we present NowcastDiT, a precipitation nowcasting framework built on a standard Diffusion Transformer with minimal modifications. As illustrated in Figure 2, it promotes temporal coherence through scheduled noise correlation and meteorological skill through timestep-aware rewards, framing precipitation nowcasting as a natural extension of video diffusion modeling.

4.1 Transformer Backbone

Precipitation nowcasting shares the core spatiotemporal modeling requirements of video generation, making a standard diffusion transformer a natural backbone. Following standard visual-conditioning designs in video diffusion models (Blattmann et al., 2023a;b; Voleti et al., 2022; Gupta et al., 2024), NowcastDiT conditions future latent generation on historical observations by adding encoded observations to the noisy patch embeddings. One advantage of preserving a task-agnostic backbone is that NowcastDiT can inherit advances from broader generative modeling. Specifically, we incorporate two such advances that directly address the modeling requirements of precipitation nowcasting.

QK-Norm. Radar-based precipitation observations yield a sparse, long-tailed intensity distribution, which can produce large variations in token feature magnitudes and attention logits. To keep attention computation well scaled during optimization, we apply QK-Norm (Henry et al., 2020) to normalize query and key vectors before their dot product.

3D RoPE. Precipitation sequences form a regular spatiotemporal grid, and their evolution is characterized by relative displacement across frames and spatial locations. Following CogVideoX (Yang et al., 2024) and Wan (Wan et al., 2025), we apply RoPE (Su et al., 2024) along the temporal, height, and width axes, enabling attention to encode relative offsets along all three dimensions in spacetime.

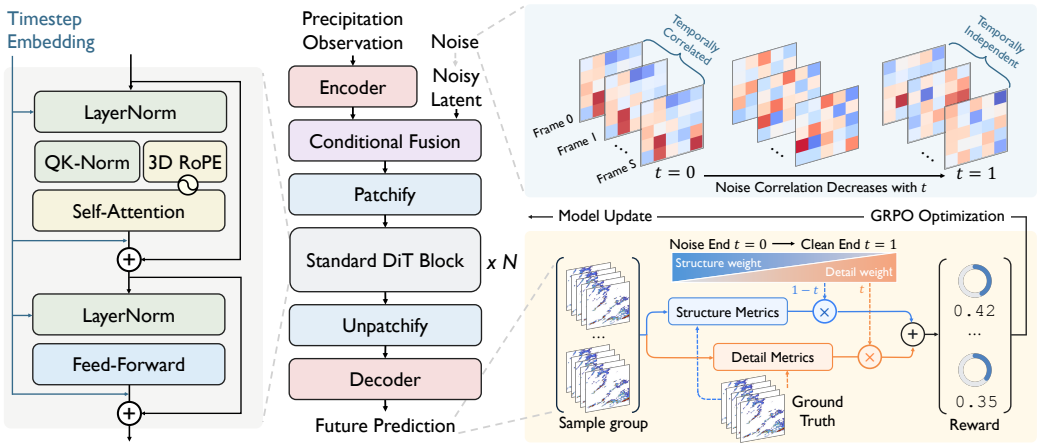


Figure 2: Overview of **NowcastDiT**. *Left:* A standard Diffusion Transformer predicts future precipitation conditioned on encoded observations, with QK-Norm and 3D RoPE. *Top right:* The dynamics-aware noise prior (DyPro) schedules cross-frame noise correlation from temporally correlated at the noise endpoint to independent at the clean endpoint. *Bottom right:* Timestep-aware post-training shifts the reward emphasis from structure to detail along the denoising trajectory.

Overall, NowcastDiT retains a deliberately simple framework: it relies on a standard DiT without task-specialized forecasting modules, explicit physical constraints, or cascaded architectures.

4.2 Temporal Coherence with Scheduled Noise Correlation

Standard video diffusion formulations typically adopt i.i.d. Gaussian noise over the spatiotemporal volume, yielding independent noise realizations across forecast frames. While simple and broadly applicable, this frame-wise independence is less aligned with the strong temporal coherence inherent in precipitation evolution.

Dynamics-aware noise prior. Existing approaches that adapt pretrained image diffusion models for video generation have shown that correlating noise across adjacent forecast frames can improve temporal consistency (Ge et al., 2023; Chang et al., 2024). Motivated by this insight, we introduce a dynamics-aware noise prior *(DyPro)* that correlates noise across forecast frames. To construct this prior, we draw on the progressive noise schedule following Dynamical Diffusion (Guo et al., 2025), with the strength of cross-frame correlation controlled by diffusion timestep. Let $s \in \{1, \dots, S\}$ index the forecast frames, and let $\boldsymbol{\epsilon}_t^{1:S}$ denote the corresponding noise sequence at timestep t. For a fixed diffusion timestep t, DyPro recursively constructs

$$\boldsymbol{\epsilon}_t^1 = \boldsymbol{\xi}^1, \quad \boldsymbol{\epsilon}_t^s = \sqrt{\gamma_t}\boldsymbol{\epsilon}_t^{s-1} + \sqrt{1-\gamma_t}\,\boldsymbol{\xi}^s. \tag{3}$$

Here, $\boldsymbol{\xi}^s \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$ are independent Gaussian samples, and the diffusion-timestep-dependent inheritance weight is

$$\gamma_t = \frac{\alpha^2(1-t)^2}{1+\alpha^2(1-t)^2}, \tag{4}$$

where hyperparameter $\alpha \geq 0$ controls the temporal dependence together with t. The recursion preserves the standard Gaussian marginal of each frame, $\boldsymbol{\epsilon}_t^s \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$, while modifying only the cross-frame covariance. Specifically, for two frames k steps apart, $\mathrm{Cov}(\boldsymbol{\epsilon}_t^s, \boldsymbol{\epsilon}_t^{s-k}) = \gamma_t^{k/2}\mathbf{I}$. Thus, adjacent frames share the most similar noise, and this similarity decays geometrically with temporal distance k. As denoising proceeds, γ_t decreases from $\alpha^2/(1+\alpha^2)$ at the beginning to zero at the clean endpoint. DyPro therefore promotes spatiotemporal coherence over the fixed radar grid during early denoising, while progressively relaxing the coupling during later stages to preserve variations among individual forecast frames associated with localized precipitation growth, decay, and intensity evolution.

Training and Inference. DyPro is independent of the backbone, preserving the model's input and output shapes and the form of the training objective. During training, we use DyPro noise to construct $\mathbf{z}_t$ and the flow-matching target (Algorithm 1). At inference, we use a sampler following

Algorithm 1 DyPro Training

```
# x: training latents; c: history
# net(z, t, c): predicts x - e

t = sample_t()

# Construct time-dependent noise
e_init = randn_like(x)
e = correlate(e_init, t)

z = t * x + (1 - t) * e
v = x - e

v_pred = net(z, t, c)
loss = l2_loss(v - v_pred)
update(net, loss)
```

Algorithm 2 DyPro Sampling (w/o CFG)

```
# c: history; shape: latent shape
# N: number of sampling steps

ts = linspace(0, 1, N + 1)
z = correlate(randn(shape), 0)

for t, t_next in zip(ts[:-1], ts[1:]):
    v = net(z, t, c)
    x = z + (1 - t) * v
    e = z - t * v
    e_init = decorrelate(e, t)
    e = correlate(e_init, t_next)
    z = t_next * x + (1 - t_next) * e

return z
```

Dynamical Diffusion that updates noise correlation at each step (Algorithm 2). The sampler requires one network evaluation per step, matching standard Euler sampling in styles.

4.3 Meteorological Skill with Timestep-Aware Rewards

Standard generative models are typically optimized with surrogate objectives such as flow matching and evaluated by metrics aligned with generative fidelity. Yet precipitation nowcasting also emphasizes task-specific meteorological skill that does not necessarily align with these objectives, making flow matching alone insufficient for optimizing forecasting performance.

Existing work has shown that task-specific evaluation metrics can serve directly as verifiable rewards for optimizing generative prediction models (Wu et al., 2025; Xue et al., 2025). Inspired by the pretraining–posttraining paradigm of foundation models, we adopt a two-stage training strategy that separates general generative modeling from meteorological skill optimization:

- **Stage 1:** Pretrain a diffusion model from scratch with surrogate flow matching objective.
- **Stage 2:** Posttrain the diffusion model by directly optimizing meteorological skills.

Skill optimization via GRPO. We formulate the posttraining stage as a GRPO objective for diffusion models (Shao et al., 2024; Liu et al., 2025). For each conditioning context paired with a ground-truth future sequence $\mathbf{x}^*$, we sample a group of forecasts $\{\hat{\mathbf{x}}^{(1)}, \hat{\mathbf{x}}^{(2)}, \ldots, \hat{\mathbf{x}}^{(G)}\}$ through a sampling trajectory where selected ODE transitions are converted into SDE transitions that serve as stochastic policy actions with tractable likelihoods, following the design of MixGRPO (Li et al., 2025). We score each $\hat{\mathbf{x}}^{(i)}$ against $\mathbf{x}^*$ and update the stochastic transitions using the clipped group-relative objective

$$\hat{A}_n^i = \frac{R(\hat{\mathbf{x}}^{(i)}, \mathbf{x}^*; t_n) - \text{mean}\left(\left\{R(\hat{\mathbf{x}}^{(j)}, \mathbf{x}^*; t_n)\right\}_{j=1}^{G}\right)}{\text{std}\left(\left\{R(\hat{\mathbf{x}}^{(j)}, \mathbf{x}^*; t_n)\right\}_{j=1}^{G}\right)}, \tag{5}$$

$$\mathcal{J}_{\text{GRPO}}(\theta) = \frac{1}{G|\mathcal{W}_\ell|} \sum_{n \in \mathcal{W}_\ell} \sum_{i=1}^{G} \min\left(\rho_n^i(\theta)\hat{A}_n^i, \text{clip}\left(\rho_n^i(\theta), 1-\varepsilon, 1+\varepsilon\right)\hat{A}_n^i\right), \tag{6}$$

where R denotes the reward evaluated on the final forecast, $\mathcal{W}_\ell$ indexes the stochastic transitions included in the GRPO update, and ρ_n^i is the policy likelihood ratio for forecast i at transition n. Further details on GRPO optimization and mixed ODE–SDE sampling are provided in Appendix C.3 and Appendix C.1, respectively.

Timestep-aware rewards. Reinforcement learning for diffusion models typically uses the same terminal reward across all denoising transitions. However, different denoising transitions are expected to contribute differently to the final forecast: transitions near the noise endpoint establish precipitation coverage and large-scale spatial patterns, whereas transitions near the clean endpoint refine local intensity variations and heavy-precipitation cores. This uniform supervision may therefore be

Table 1: Performance comparison on SEVIR and MRMS. ↑ indicates higher is better and ↓ indicates lower is better. Bold and underlined values denote the best and second-best results, respectively.

SEVIR							MRMS						
Method	CSI↑	$CSI^{\uparrow}_{181}$	$CSI^{\uparrow}_{219}$	HSS↑	LPIPS↓	SSIM↑	Method	CSI↑	$CSI^{\uparrow}_{16}$	$CSI^{\uparrow}_{32}$	HSS↑	LPIPS↓	SSIM↑
ConvLSTM	0.2912	0.0684	0.0336	0.3571	0.2989	<u>0.7257</u>	ConvLSTM	0.2159	0.0514	0.0172	0.2877	0.3013	0.8978
PhyDNet	0.2874	0.0664	0.0211	0.3523	0.3081	0.7252	PhyDNet	0.2332	0.0635	0.0268	0.3111	0.3005	<u>0.9001</u>
Earthformer	0.2752	0.0496	0.0213	0.3391	0.3376	0.7142	Earthformer	0.2353	0.0691	0.0297	0.3152	0.3001	0.8994
SimVP	0.2951	0.0753	0.0416	0.3630	0.3113	0.7247	SimVP	<u>0.2366</u>	0.0657	0.0277	0.3155	0.2992	**0.9006**
AlphaPre	<u>0.2980</u>	0.0860	0.0436	0.3677	0.2897	**0.7301**	AlphaPre	0.2309	0.0587	0.0211	0.2933	0.3014	0.8944
DiffCast	0.2977	0.0915	<u>0.0571</u>	<u>0.4033</u>	0.1812	0.6774	NowcastNet	0.2244	<u>0.1145</u>	<u>0.0729</u>	0.3083	0.2007	0.8502
PreDiff	0.2859	0.0893	0.0487	0.3647	<u>0.1543</u>	0.7006	PreDiff	0.2283	0.1083	0.0695	0.3148	<u>0.1906</u>	0.8633
CasCast	0.2905	0.0930	0.0503	0.3703	0.1583	0.7012	CasCast	0.2224	0.0925	0.0527	0.3064	0.2045	0.8791
DiT	0.2926	<u>0.0948</u>	0.0555	0.3733	0.1545	0.7048	DiT	0.2343	0.1015	0.0611	<u>0.3223</u>	0.1953	0.8824
NowcastDiT	**0.3240**	**0.1241**	**0.0750**	**0.4148**	**0.1469**	0.7197	NowcastDiT	**0.2696**	**0.1373**	**0.0958**	**0.3668**	**0.1827**	0.8642

suboptimal. We introduce a timestep-aware terminal reward that gradually shifts its emphasis from precipitation structure to local detail as denoising proceeds:

$$R_{\text{noise}}(\hat{\mathbf{x}}, \mathbf{x}^*) = \sum_{\tau \in \mathcal{T}_{\text{low}}} \lambda_\tau \text{CSI}^{\text{pool}}_\tau(\hat{\mathbf{x}}, \mathbf{x}^*) + \lambda_s \text{SSIM}(\hat{\mathbf{x}}, \mathbf{x}^*), \tag{7}$$

$$R_{\text{clean}}(\hat{\mathbf{x}}, \mathbf{x}^*) = \sum_{\tau \in \mathcal{T}_{\text{high}}} \lambda_\tau \text{CSI}_\tau(\hat{\mathbf{x}}, \mathbf{x}^*) - \lambda_p \text{LPIPS}(\hat{\mathbf{x}}, \mathbf{x}^*), \tag{8}$$

$$R(\hat{\mathbf{x}}, \mathbf{x}^*; t) = (1-t) R_{\text{noise}}(\hat{\mathbf{x}}, \mathbf{x}^*) + t R_{\text{clean}}(\hat{\mathbf{x}}, \mathbf{x}^*), \tag{9}$$

where $\mathcal{T}_{\text{low}}$ and $\mathcal{T}_{\text{high}}$ denote the light-precipitation and heavy-precipitation threshold sets, respectively. The coefficients λ_s and λ_p balance the constituent metrics, while the remaining weight is distributed uniformly across the corresponding CSI thresholds: $\lambda_\tau = (1-\lambda_s)/|\mathcal{T}_{\text{low}}|$ for R_{noise} and $\lambda_\tau = (1-\lambda_p)/|\mathcal{T}_{\text{high}}|$ for R_{clean}. We empirically set $\lambda_s = 0.5$ and $\lambda_p = 0.9$.

Specifically, R_{noise} combines spatially tolerant low-threshold pooled CSI for event coverage with SSIM (Wang et al., 2004) for local structural agreement, while R_{clean} combines high-threshold CSI for localized intense precipitation with negative LPIPS (Zhang et al., 2018) for perceptual similarity in deep feature space. Since $t = 0$ and $t = 1$ correspond to the noise and clean endpoints, respectively, the reward gradually shifts from R_{noise} to R_{clean}, emphasizing precipitation structure early and local detail later. All rewards are computed on the final decoded forecast, with the active transition time determining the metric mixture used for credit assignment in Equation 5.

5 Experiments

We evaluate NowcastDiT on two radar-based benchmarks, SEVIR and MRMS. Comparisons with deterministic and generative nowcasting baselines assess both meteorological skill and perceptual quality. Additional experiments examine how our design choices affect forecasting performance.

5.1 Experimental Setup

Datasets. SEVIR (Veillette et al., 2020) is a spatiotemporal radar observation dataset covering weather events across the United States. Following prior work (Yu et al., 2024; Lin et al., 2025), we use Vertically Integrated Liquid (VIL) observations at a 5-minute temporal resolution to predict 20 future frames given 5 observed frames at 128×128 resolution. MRMS (Zhang et al., 2016) is a composite radar dataset collected over the contiguous United States. Following NowcastNet (Zhang et al., 2023), we use precipitation-rate observations at a 10-minute temporal resolution to predict 20 future frames given 4 observed frames at 256×256 resolution. Additional dataset details, splits, and preprocessing are provided in Appendix D.1.

Evaluation. We compare NowcastDiT with deterministic and generative methods. The deterministic baselines include ConvLSTM (Shi et al., 2015), PhyDNet (Guen & Thome, 2020), Earthformer (Gao et al., 2022b), SimVP (Gao et al., 2022a), and AlphaPre (Lin et al., 2025), while the generative baselines include NowcastNet (Zhang et al., 2023), PreDiff (Gao et al., 2023), DiffCast (Yu et al., 2024), and CasCast (Gong et al., 2024). We evaluate meteorological skill using the

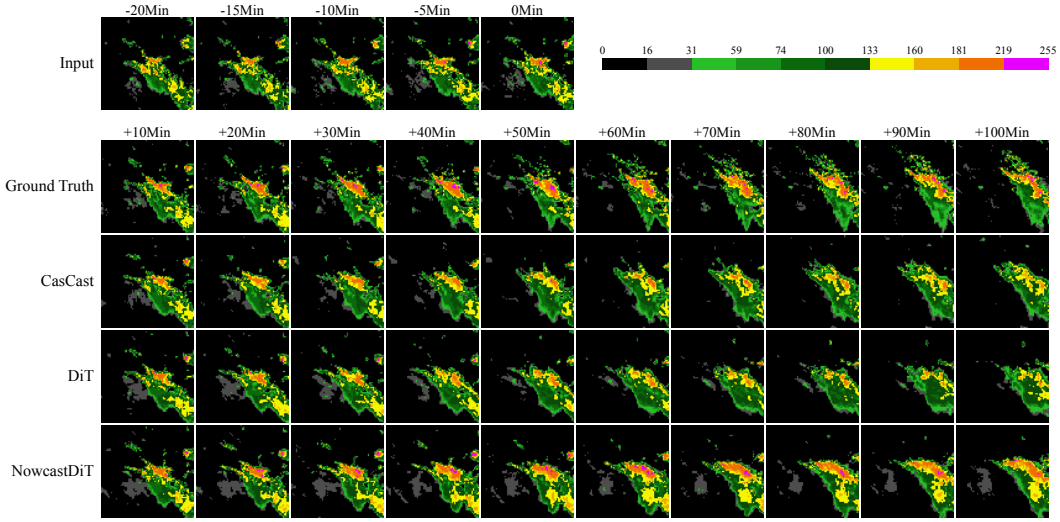


Figure 3: **Qualitative comparison on SEVIR.** Forecasts are shown every 10 minutes up to 100 minutes. Colors indicate VIL values.

Critical Success Index (CSI) (Schaefer, 1990) and Heidke Skill Score (HSS) (Jolliffe & Stephenson, 2012), and perceptual quality using the Structural Similarity Index (SSIM) (Wang et al., 2004) and Learned Perceptual Image Patch Similarity (LPIPS) (Zhang et al., 2018). CSI measures event detection accuracy, while HSS accounts for agreement expected by chance. Both are computed from pixelwise contingency counts aggregated over the evaluation set at each forecast lead time. We report their means over lead times and thresholds, using $\{16, 74, 133, 160, 181, 219\}$ in the VIL for SEVIR and $\{1, 2, 4, 8, 16, 32\}$ mm h^{-1} for MRMS. We additionally report high-threshold CSI to assess intense precipitation events. SSIM measures local structural agreement, whereas LPIPS measures perceptual distance in deep feature space. Higher CSI, HSS, and SSIM and lower LPIPS indicate better performance. Metric definitions are provided in Appendix D.2.

Implementation Details. We train NowcastDiT using AdamW with $(\beta_1, \beta_2) = (0.9, 0.95)$, a batch size of 32, and learning rates of 10^{-4} for pretraining and 10^{-5} for post-training. The default backbone has 12 Transformer blocks, a hidden dimension of 768, and 12 attention heads. The DyPro parameter α is set to 0.5. Post-training uses a group size of 16, 10 denoising steps, an SDE window size of 4, and a policy ratio clipping range of 10^{-4}. For reward computation, we apply 4×4 max pooling to low-threshold CSI and set the SSIM and LPIPS mixing weights to 0.5 and 0.9, respectively. All training runs use four NVIDIA A100 GPUs with 80 GB of memory each. Additional post-training settings are provided in Appendix C.6.

5.2 Main Results

We compare NowcastDiT with deterministic and generative nowcasting methods on SEVIR and MRMS, reported in Table 1. The standard DiT baseline already achieves competitive performance against specialized nowcasting models, demonstrating the strength of its general-purpose spatiotemporal modeling. Relative to the strongest baselines, NowcastDiT improves aggregate CSI by 8.7% and 15.1%, and HSS by 2.9% and 13.8%, on SEVIR and MRMS, respectively. It also reduces LPIPS by 4.8% and 4.1% while achieving the best SSIM among generative methods on SEVIR. On high-threshold CSI that directly reflects the ability to detect rare and operationally important intense rainfall events, NowcastDiT achieves relative gains of 19.9%–31.4% across the reported high thresholds, showing a clear advantage in heavy-rainfall prediction. We also provide per-lead-time CSI and HSS curves, shown in Appendix E.1. Taken together, these results show that a standard DiT, with precipitation-specific alignment can achieve strong nowcasting performance while retaining a simple architecture.

Qualitative comparisons. Figures 3 and 4 compare forecast sequences on SEVIR and MRMS, respectively. In the SEVIR example, NowcastDiT retains localized high-intensity regions at longer lead times, whereas the DiT baseline produces smoother precipitation fields. In the MRMS example, NowcastDiT better preserves the shape of the main rainband. These selected examples complement the test-set comparisons in Table 1.

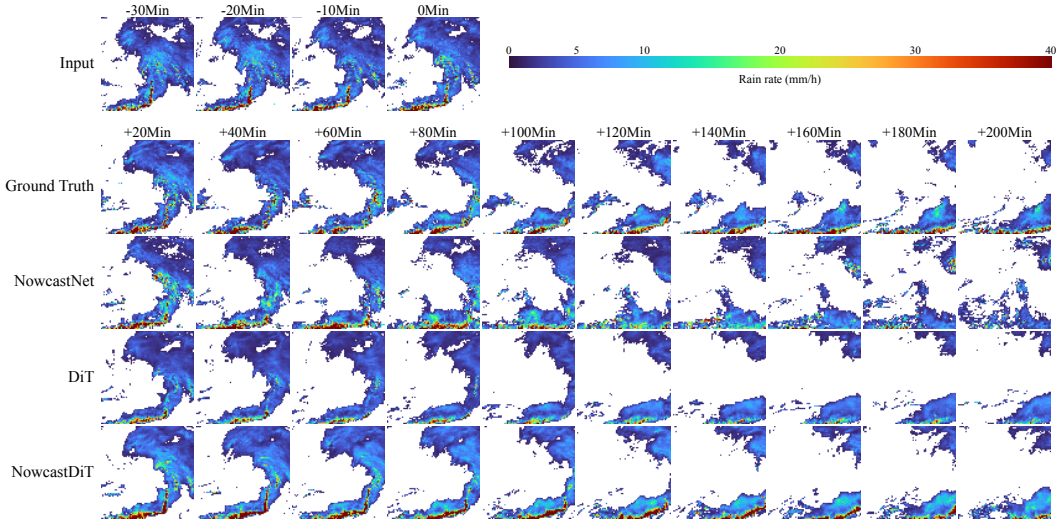


Figure 4: **Qualitative comparison on MRMS.** Forecasts are shown every 20 minutes up to 200 minutes. Colors indicate precipitation rate in mm h^{-1}.

5.3 Analyses

We analyze how the following components and design choices affect model performance on SEVIR.

Model parameter scaling. Diffusion Transformer is validated as a scalable backbone, and we investigate whether this scaling benifits extend to precipitation nowcasting. Following Peebles & Xie (2023), we evaluate NowcastDiT across a range of model sizes. As shown in Figure 5(a), increasing model size consistently improves aggregate CSI. This finding supports model scaling as an effective strategy for strengthening the forecasting capabilities of a standard DiT backbone, which forms the initial motivation of NowcastDiT.

CFG and skill-aware post-training. Classifier-free guidance (CFG) (Ho & Salimans, 2022) is a common capability of standard diffusion models and should be naturally inherited when adopting them for precipitation nowcasting. We therefore examine whether skill-aware post-training can provide complementary gains on top of CFG, whose integration with diffusion post-training is not always trivial (Zheng et al., 2026). As shown in Figure 5(b), both CFG and RL improve aggregate CSI over the base model, and their combination achieves the best performance. This suggests that meteorological alignment provides an additional optimization axis beyond standard diffusion guidance. Detailed methods and results are provided in Appendix E.3.

DyPro correlation strength. For simplicity, NowcastDiT adopts the factor $\alpha = 0.5$ controlling temporal correlations in DyPro. To assess the senstivity, we vary the DyPro correlation strength $\alpha \in \{0, 0.25, 0.33, 0.66, 0.75\}$, where $\alpha = 0$ infers the i.i.d. noise baseline. Figure 5(c) shows that all tested nonzero strengths improve aggregate CSI, with performance peaking at a moderate strength and declining as the correlation increases further, indicating the robustness of α in DyPro. Detailed results are provided in Appendix E.4.

Timestep-aware reward. The reinforcement learning stage of NowcastDiT uses a stage-aware reward to account for the different roles of reward metrics near the noise and clean endpoints. We compare it with a static baseline that mixes the same rewards without timestep-dependent weighting. As shown in Figure 5(d), timestep-aware weighting consistently performs better. To isolate this effect, we further repeat the comparison under CSI-only and perceptual-only objectives (i.e., $\lambda_s = \lambda_p = 0$ and $\lambda_\tau = 0$, respectively), where the same trend holds. These results confirm the benefit of timestep-aware weighting across reward objectives, while mixed metrics provide a better balance between meteorological skill and perceptual quality.

6 Related Work

Diffusion models for precipitation nowcasting. Diffusion models have become an important paradigm for probabilistic nowcasting due to their stable training dynamics and strong generative

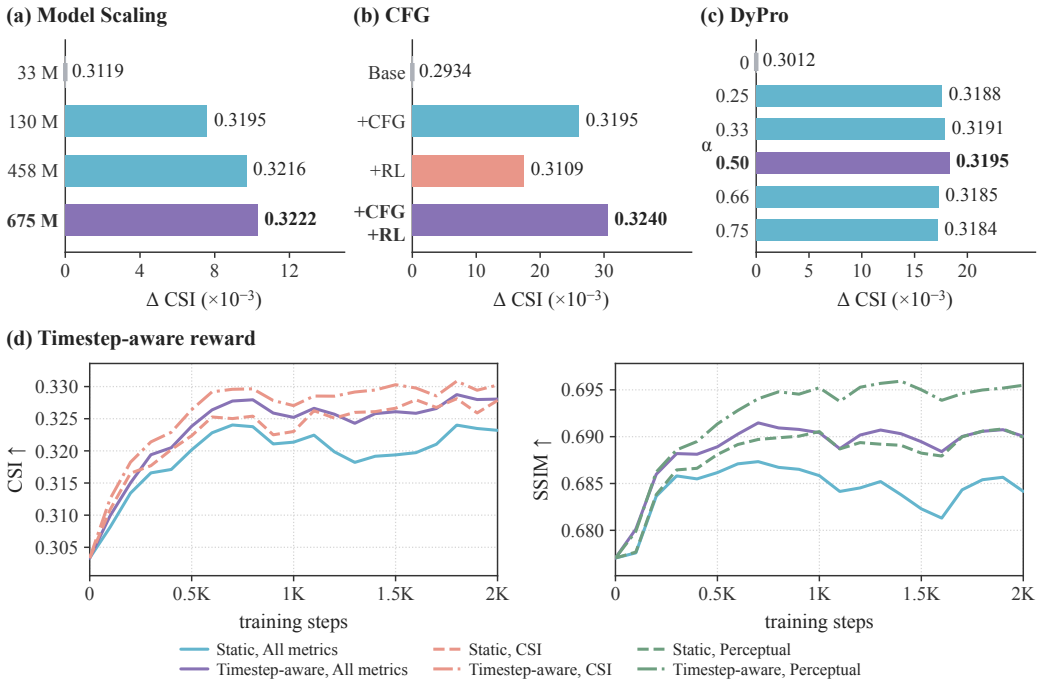


Figure 5: **Analysis on SEVIR.** Bar lengths indicate CSI gains relative to the first setting in each panel; labels report absolute CSI values.

fidelity (Gao et al., 2023; Yu et al., 2024). Existing diffusion-based approaches often incorporate task-specific mechanisms to improve meteorological applicability (Gao et al., 2023; Yu et al., 2024; Gong et al., 2024; Wen et al., 2026). PreDiff (Gao et al., 2023) introduces intensity continuity guidance over a spatiotemporal architecture to strength physical alignment. DiffCast (Yu et al., 2024) introduces decomposited training phases for the modeling of global deterministic motion and local stocastic variations. CasCast (Gong et al., 2024) employs a cascaded DiT framework in latent space to decouple deterministic and stochastic precipitation modeling for improved efficiency and extreme event fidelity. These approaches couple domain-specific mechanisms with architectural and framework design, leaving unresolved whether specialized designs are necessary for effective precipitation nowcasting. In contrast, NowcastDiT examines how a standard DiT can support effective nowcasting and address domain-specific requirements within its design space.

Advances in generative modeling techniques. Recent advances in image and video generation have substantially broadened the modeling capabilities of diffusion transformers (Peebles & Xie, 2023; Esser et al., 2024; Ma et al., 2025; Wan et al., 2025). Architectural refinements improve attention stability and positional modeling (Henry et al., 2020; Su et al., 2024; Gupta et al., 2024; Yang et al., 2024). Noise-prior designs introduce temporal dependencies to promote coherent video generation (Ge et al., 2023; Chang et al., 2024; Guo et al., 2025). Post-training strategies align generated outputs with task-specific objectives through reinforcement learning (Black et al., 2024; Liu et al., 2025; Xue et al., 2025; Li et al., 2025; Wu et al., 2025). Together, these developments establish the feasibility of meeting domain-specific modeling requirements within the design space of a standard DiT. Building on this foundation, NowcastDiT further develops noise-prior and post-training designs to model precipitation dynamics and enhance meteorological skill.

7 Conclusion

This work demonstrates that a standard Diffusion Transformer, with targeted adaptations to its diffusion and training strategies, is sufficient for effective precipitation nowcasting. NowcastDiT combines DyPro for temporally coherent forecasting with skill-aware post-training for meteorological alignment. Experiments on SEVIR and MRMS show strong performance in both meteorological skill and perceptual quality, supporting standard DiTs as a simple and scalable framework for precipitation nowcasting. Future work will evaluate its generalization across observation systems, resolutions, and longer forecast horizons.

AI USE STATEMENT

AI tools assisted with drafting and revising portions of the manuscript, language polishing, literature search, figure preparation, verification of mathematical proof and use of the ICLR 2027 template. The authors personally reviewed and revised the AI-assisted content, verified its accuracy, and take full responsibility for the final manuscript, including all claims and results.

REFERENCES

Niket Agarwal, Arslan Ali, Jon Allen, Martin Antolini, Adeline Aubame, Alisson Azzolini, Junjie Bai, Maciej Bala, Yogesh Balaji, Josh Bapst, et al. Cosmos 3: Omnimodal world models for physical ai. *arXiv*, 2026.

Arslan Ali, Junjie Bai, Maciej Bala, Yogesh Balaji, Aaron Blakeman, Tiffany Cai, Jiaxin Cao, Tianshi Cao, Elizabeth Cha, Yu-Wei Chao, et al. World simulation with video foundation models for physical ai. *arXiv*, 2025.

Kevin Black, Michael Janner, Yilun Du, Ilya Kostrikov, and Sergey Levine. Training diffusion models with reinforcement learning. In *ICLR*, 2024.

Andreas Blattmann, Tim Dockhorn, Sumith Kulal, Daniel Mendelevitch, Maciej Kilian, Dominik Lorenz, Yam Levi, Zion English, Vikram Voleti, Adam Letts, et al. Stable video diffusion: Scaling latent video diffusion models to large datasets. *arXiv*, 2023a.

Andreas Blattmann, Robin Rombach, Huan Ling, Tim Dockhorn, Seung Wook Kim, Sanja Fidler, and Karsten Kreis. Align your latents: High-resolution video synthesis with latent diffusion models. In *CVPR*, 2023b.

Pascal Chang, Jingwei Tang, Markus Gross, and Vinicius C Azevedo. How i warped your noise: a temporally-correlated noise prior for diffusion models. In *ICLR*, 2024.

Patrick Esser, Sumith Kulal, Andreas Blattmann, Rahim Entezari, Jonas Müller, Harry Saini, Yam Levi, Dominik Lorenz, Axel Sauer, Frederic Boesel, et al. Scaling rectified flow transformers for high-resolution image synthesis. *arXiv preprint arXiv:2403.03206*, 2024.

Zhangyang Gao, Cheng Tan, Lirong Wu, and Stan Z Li. Simvp: Simpler yet better video prediction. In *CVPR*, 2022a.

Zhihan Gao, Xingjian Shi, Hao Wang, Yi Zhu, Yuyang Bernie Wang, Mu Li, and Dit-Yan Yeung. Earthformer: Exploring space-time transformers for earth system forecasting. In *NeurIPS*, 2022b.

Zhihan Gao, Xingjian Shi, Boran Han, Hao Wang, Xiaoyong Jin, Danielle Maddix, Yi Zhu, Mu Li, and Yuyang Bernie Wang. Prediff: Precipitation nowcasting with latent diffusion models. In *NeurIPS*, 2023.

Songwei Ge, Seungjun Nah, Guilin Liu, Tyler Poon, Andrew Tao, Bryan Catanzaro, David Jacobs, Jia-Bin Huang, Ming-Yu Liu, and Yogesh Balaji. Preserve your own correlation: A noise prior for video diffusion models. In *ICCV*, 2023.

Junchao Gong, Lei Bai, Peng Ye, Wanghan Xu, Na Liu, Jianhua Dai, Xiaokang Yang, and Wanli Ouyang. Cascast: Skillful high-resolution precipitation nowcasting via cascaded modelling. In *ICML*, 2024.

Vincent Le Guen and Nicolas Thome. Disentangling physical dynamics from unknown factors for unsupervised video prediction. In *CVPR*, 2020.

Xingzhuo Guo, Yu Zhang, Baixu Chen, Haoran Xu, Jianmin Wang, and Mingsheng Long. Dynamical diffusion: Learning temporal dynamics with diffusion models. In *ICLR*, 2025.

Agrim Gupta, Lijun Yu, Kihyuk Sohn, Xiuye Gu, Meera Hahn, Li Fei-Fei, Irfan Essa, Lu Jiang, and José Lezama. Photorealistic video generation with diffusion models. In *ECCV*, 2024.

Alex Henry, Prudhvi Raj Dachapally, Shubham Shantaram Pawar, and Yuxuan Chen. Query-key normalization for transformers. In *EMNLP*, 2020.

Jonathan Ho and Tim Salimans. Classifier-free diffusion guidance. *arXiv*, 2022.

Jonathan Ho, Ajay Jain, and Pieter Abbeel. Denoising diffusion probabilistic models. In *NeurIPS*, 2020.

Ian T Jolliffe and David B Stephenson. *Forecast verification: a practitioner's guide in atmospheric science*. John Wiley & Sons, 2012.

Junzhe Li, Yutao Cui, Tao Huang, Yinping Ma, Chun Fan, Miles Yang, and Zhao Zhong. Mixgrpo: Unlocking flow-based grpo efficiency with mixed ode-sde, 2025.

Kenghong Lin, Baoquan Zhang, Demin Yu, Wenzhi Feng, Shidong Chen, Feifan Gao, Xutao Li, and Yunming Ye. AlphaPre: Amplitude-phase disentanglement model for precipitation nowcasting. In *CVPR*, pp. 17841–17850, 2025.

Yaron Lipman, Ricky TQ Chen, Heli Ben-Hamu, Maximilian Nickel, and Matt Le. Flow matching for generative modeling. In *ICML*, 2023.

Jie Liu, Gongye Liu, Jiajun Liang, Yangguang Li, Jiaheng Liu, Xintao Wang, Pengfei Wan, Di Zhang, and Wanli Ouyang. Flow-grpo: Training flow matching models via online rl. In *ICLR*, 2025.

Xin Ma, Yaohui Wang, Xinyuan Chen, Gengyun Jia, Ziwei Liu, Yuan-Fang Li, Cunjian Chen, and Yu Qiao. Latte: Latent diffusion transformer for video generation. In *TMLR*, 2025.

OpenAI. Sora: Video generation models as world simulators. `https://openai.com/research/video-generation-models-as-world-simulators`, 2024. Accessed: 2024-04-01.

William Peebles and Saining Xie. Scalable diffusion models with transformers. In *ICCV*, 2023.

Robin Rombach, Andreas Blattmann, Dominik Lorenz, Patrick Esser, and Björn Ommer. High-resolution image synthesis with latent diffusion models. In *CVPR*, 2022.

Joseph T Schaefer. The critical success index as an indicator of warning skill. *Weather and forecasting*, 1990.

Team Seedance, De Chen, Liyang Chen, Xin Chen, Ying Chen, Zhuo Chen, Zhuowei Chen, Feng Cheng, Tianheng Cheng, Yufeng Cheng, et al. Seedance 2.0: Advancing video generation for world complexity. *arXiv*, 2026.

Zhihong Shao, Peiyi Wang, Qihao Zhu, Runxin Xu, Junxiao Song, Xiao Bi, Haowei Zhang, Mingchuan Zhang, Y. K. Li, Y. Wu, and Daya Guo. Deepseekmath: Pushing the limits of mathematical reasoning in open language models. *arXiv*, 2024.

Xingjian Shi, Zhourong Chen, Hao Wang, Dit-Yan Yeung, Wai-Kin Wong, and Wang-chun Woo. Convolutional lstm network: A machine learning approach for precipitation nowcasting. In *NeurIPS*, 2015.

Yang Song, Jascha Sohl-Dickstein, Diederik P Kingma, Abhishek Kumar, Stefano Ermon, and Ben Poole. Score-based generative modeling through stochastic differential equations. In *ICLR*, 2020.

Jianlin Su, Murtadha Ahmed, Yu Lu, Shengfeng Pan, Wen Bo, and Yunfeng Liu. Roformer: Enhanced transformer with rotary position embedding. *Neurocomputing*, 2024.

Mark Veillette, Siddharth Samsi, and Chris Mattioli. Sevir: A storm event imagery dataset for deep learning applications in radar and satellite meteorology. In *NeurIPS*, 2020.

Vikram Voleti, Alexia Jolicoeur-Martineau, and Chris Pal. Mcvd-masked conditional video diffusion for prediction, generation, and interpolation. In *NeurIPS*, 2022.

Team Wan, Ang Wang, Baole Ai, Bin Wen, Chaojie Mao, Chen-Wei Xie, Di Chen, Feiwu Yu, Haiming Zhao, Jianxiao Yang, et al. Wan: Open and advanced large-scale video generative models. *arXiv preprint arXiv:2503.20314*, 2025.

Yong Wang, Estelle De Coning, Abdoulaye Harou, and Wilfried Jacobs. Guidelines for nowcasting techniques. Wmo-no. 1198, World Meteorological Organization, Geneva, Switzerland, 2017.

Zhou Wang, Alan C Bovik, Hamid R Sheikh, Eero P Simoncelli, et al. Image quality assessment: from error visibility to structural similarity. *IEEE transactions on image processing*, 13(4):600–612, 2004.

Penghui Wen, Mengwei He, Patrick Filippi, Na Zhao, Feng Zhang, Thomas Francis Bishop, Zhiyong Wang, and Kun Hu. DuoCast: Duo-probabilistic diffusion for precipitation nowcasting. In *AAAI*, 2026.

Jialong Wu, Shaofeng Yin, Ningya Feng, and Mingsheng Long. RLVR-World: Training world models with reinforcement learning. In *Advances in Neural Information Processing Systems*, volume 38, 2025.

Zeyue Xue, Jie Wu, Yu Gao, Fangyuan Kong, Lingting Zhu, Mengzhao Chen, Zhiheng Liu, Wei Liu, Qiushan Guo, Weilin Huang, et al. Dancegrpo: Unleashing grpo on visual generation. *arXiv*, 2025.

Zhuoyi Yang, Jiayan Teng, Wendi Zheng, Ming Ding, Shiyu Huang, Jiazheng Xu, Yuanming Yang, Wenyi Hong, Xiaohan Zhang, Guanyu Feng, et al. Cogvideox: Text-to-video diffusion models with an expert transformer. *arXiv*, 2024.

Seonghyeon Ye, Yunhao Ge, Kaiyuan Zheng, Shenyuan Gao, Sihyun Yu, George Kurian, Suneel Indupuru, You Liang Tan, Chuning Zhu, Jiannan Xiang, et al. World action models are zero-shot policies. *arXiv*, 2026.

Demin Yu, Xutao Li, Yunming Ye, Baoquan Zhang, Chuyao Luo, Kuai Dai, Rui Wang, and Xunlai Chen. Diffcast: A unified framework via residual diffusion for precipitation nowcasting. In *CVPR*, 2024.

Jian Zhang, Kenneth Howard, Carrie Langston, Brian Kaney, Youcun Qi, Lily Tang, Heather Grams, Yixin Wang, Stephen Cocks, Steven Martinaitis, et al. Multi-radar multi-sensor (MRMS) quantitative precipitation estimation: Initial operating capabilities. *Bulletin of the American Meteorological Society*, 97(4):621–638, 2016.

Richard Zhang, Phillip Isola, Alexei A Efros, Eli Shechtman, and Oliver Wang. The unreasonable effectiveness of deep features as a perceptual metric. In *CVPR*, 2018.

Yuchen Zhang, Mingsheng Long, Kaiyuan Chen, Lanxiang Xing, Ronghua Jin, Michael I Jordan, and Jianmin Wang. Skilful nowcasting of extreme precipitation with nowcastnet. *Nature*, 2023.

Kaiwen Zheng, Huayu Chen, Haotian Ye, Haoxiang Wang, Qinsheng Zhang, Kai Jiang, Hang Su, Stefano Ermon, Jun Zhu, and Ming-Yu Liu. Diffusionnft: Online diffusion reinforcement with forward process. In *ICLR*, 2026.

Jincheng Zhong, Xiangcheng Zhang, Jianmin Wang, and Mingsheng Long. Domain guidance: A simple transfer approach for a pre-trained diffusion model. In *ICLR*, 2025.

A Model Architecture and Configuration

A.1 VAE Architecture and Training

We adopt a 2D VAE which compress only the spatial resolution which acts independently across frames. The pretrained VAE provides the encoder E and decoder D and remains frozen during flow-matching pretraining and RL post-training. Table 2 summarizes its architecture and training settings.

Table 2: VAE architecture and training configuration. Parameters include the encoder, decoder, and latent projections.

Setting	SEVIR	MRMS
Architecture		
Parameters	82M	
Input shape	$25 \times 1 \times 128 \times 128$	$24 \times 1 \times 256 \times 256$
Encoder stages	4	
Residual blocks / encoder stage	2	
Encoder channels	[128, 256, 512, 512]	
Decoder stages	4	
Residual blocks / decoder stage	3	
Decoder channels	[512, 512, 256, 128]	
Latent channels	16	
Latent resolution	16×16	32×32
Spatial compression	$8\times$	
Temporal compression	$1\times$	
Normalization	GroupNorm, 32 groups	
Activation	SiLU	
Training		
# steps	1,488,000	2,000,000
Optimizer	AdamW, $(\beta_1, \beta_2) = (0.5, 0.9)$	
Batch size	8	
Learning rate	10^{-4}	
Weight decay	0.01	
EMA decay	0.9999	
Gradient clipping norm	1.0	
Adversarial warmup steps	100,000	

A.2 Transformer Architecture, Training, and Inference

Table 3 collects the default architecture, flow-matching pretraining settings, and pretrained-model inference configurations for MRMS and SEVIR. RL post-training settings are reported separately in Appendix C.6.

History Conditioning Let $\mathbf{z}_{\text{cond}} \in \mathbb{R}^{B\times T\times C\times H\times W}$ denote the conditioning sequence, which places the observed latent frames in the first L positions and masks the remaining future positions with zeros. The conditioner first applies a frame-wise convolutional patch encoder with the same patch size as the noisy-input embedder. It then adds a learnable spatial-topology embedding and a learnable frame-index embedding, followed by a SiLU activation and a zero-initialized 1×1 convolution. Denoting the resulting condition features by $\mathbf{c}$ and the noisy patch embeddings by $\mathbf{h}_t$, conditioning is performed through

$$\widetilde{\mathbf{h}}_t = \mathbf{h}_t + \mathbf{c}. \tag{10}$$

The conditioned embeddings are then flattened into spatiotemporal tokens and processed by the standard DiT blocks. Thus, the conditioner changes neither the Transformer blocks nor their input–output interface and requires no cross-attention or task-specific fusion pathway.

Table 3: Transformer architecture, flow-matching pretraining, and inference configuration. Parameters include the conditioner, time embedding, and output layer. Inference settings refer to the pretrained-model evaluation configurations.

Setting	SEVIR	MRMS
Architecture		
Parameters	130M	
Frames (history / future)	5 / 20	4 / 20
Transformer blocks	12	
Hidden dimension	768	
Feedforward dimension	3072	
Attention heads	12	
Head dimension	64	
Patch size	2×2	
Tokens per sample	$25 \times 8 \times 8 = 1600$	$24 \times 16 \times 16 = 6144$
Latent channels	16	
Block normalization	LayerNorm	
QK normalization	RMSNorm	
Positional encoding	3D RoPE	
Time modulation	adaLN-Zero	
FFN activation	GELU	
History fusion	Additive	
Training		
# steps	372,000	500,000
Optimizer	AdamW, $(\beta_1, \beta_2) = (0.9, 0.95)$	
Batch size	32	
Learning rate	10^{-4}	
Learning rate schedule	Constant	
EMA decay	0.9999	
Gradient clipping norm	1.0	
Time sampler	Logit-normal, $\mu = -1, \sigma = 1$	
DyPro correlation strength α	0.5	
Condition dropout	0.1 (zero conditioning)	
Inference		
Sampler	Euler with DyPro (Algorithm 2)	
Sampling steps	10	
Timestep grid	Uniform in $[0, 1]$	
CFG weight (without RL)	1.6	

Classifier-free guidance and Domain Guidance. During inference, we adopt classifier-free guidance that adjusts the influence of historical conditioning. Classifier-free guidance (Ho & Salimans, 2022) adjusts the influence of historical conditioning by combining conditional and unconditional velocity predictions from the same checkpoint. Let θ_{pre} denote the parameters before RL post-training and let $\emptyset$ denote zero conditioning, consistent with the condition-dropout setting in Table 3. The guided prediction is

$$\mathbf{v}^{\text{CFG}}(\mathbf{z}_t, t \mid \mathbf{c}) = w\, \mathbf{v}_{\theta_{\text{pre}}}(\mathbf{z}_t, t \mid \mathbf{c}) + (1 - w)\, \mathbf{v}_{\theta_{\text{pre}}}(\mathbf{z}_t, t \mid \emptyset), \tag{11}$$

where w is the guidance weight and $w = 1$ recovers the conditional prediction without guidance. We adopt $w = 1.6$ at pretraining stage.

B Foundation, Training and Sampling with DyPro

We describe the noise transform, training target, and deterministic sampling update associated with the DyPro parameterization in Section 4.2, followed by their distributional properties and continuous-time interpretation. The stochastic extension used for post-training is presented in Appendix B.4. Diffusion time increases from noise at $t = 0$ to data at $t = 1$. We condition throughout

Algorithm 3 DyPro Noise Correlation and Decorrelation

```
# xi, e: frame sequences; s indexes frames
# t: diffusion timestep; alpha: shared DyPro strength

def correlate(xi, t):
    a = alpha * (1 - t)
    q = 1 / sqrt(1 + a**2)
    r = a * q

    e = xi.copy()
    for s in range(1, len(xi)):
        e[s] = r * e[s - 1] + q * xi[s]
    return e

def decorrelate(e, t):
    a = alpha * (1 - t)
    q = 1 / sqrt(1 + a**2)
    r = a * q

    xi = e.copy()
    for s in range(1, len(e)):
        xi[s] = (e[s] - r * e[s - 1]) / q
    return xi
```

on the history $\mathbf{c}$ and write $\mathbf{y} = \mathbf{z}_1$ for the clean future latent sequence. Each of its S frames has d latent coordinates, so the flattened sequence has dimension $m = Sd$. The Gaussian innovations are independent of $\mathbf{y}$ and $\mathbf{c}$.

B.1 Correlated Noise

For finite $\alpha \geq 0$, define

$$r_t = \sqrt{\gamma_t} = \frac{\alpha(1-t)}{\sqrt{1+\alpha^2(1-t)^2}}, \qquad q_t = \sqrt{1-\gamma_t} = \frac{1}{\sqrt{1+\alpha^2(1-t)^2}}. \tag{12}$$

The frame-wise recursion defined in Equation 3, practically implemented through `correlate` in Algorithm 3, denotes a lower-triangular matrix representation. With frame indices $s, j \in \{0, \ldots, S-1\}$, let

$$(C_t)_{sj} = \begin{cases} r_t^s, & j = 0, \\ q_t r_t^{s-j}, & 1 \leq j \leq s, \\ 0, & j > s. \end{cases} \qquad \mathcal{C}_t = C_t \otimes \mathbf{I}_d, \qquad \boldsymbol{\epsilon}_t = \mathcal{C}_t \boldsymbol{\xi}, \tag{13}$$

where $\boldsymbol{\xi} \sim \mathcal{N}(\mathbf{0}, \mathbf{I}_m)$ and $r_t^0 = 1$, including when $r_t = 0$.

Because $q_t > 0$, this transform is invertible. Its inverse $\mathcal{C}_t^{-1} = C^{-1} \otimes \mathbf{I}_d$, implemented by `decorrelate` in Algorithm 3, follows directly from the recursion:

$$\boldsymbol{\xi}^1 = \boldsymbol{\epsilon}_t^1, \qquad \boldsymbol{\xi}^s = \frac{\boldsymbol{\epsilon}_t^s - r_t \boldsymbol{\epsilon}_t^{s-1}}{q_t}, \quad s \geq 2. \tag{14}$$

The joint noise $\boldsymbol{\epsilon}_t$ is Gaussian because it is a linear transform of independent Gaussian innovations. Since $r_t^2 + q_t^2 = 1$, the recursion preserves the standard Gaussian marginal of each frame. Independence of subsequent innovations gives $\mathrm{Cov}(\boldsymbol{\epsilon}_t^s, \boldsymbol{\epsilon}_t^j) = r_t^{|s-j|} \mathbf{I}_d = \gamma_t^{|s-j|/2} \mathbf{I}_d$, yielding the full covariance and distribution

$$\mathbf{Q}_t = \mathcal{C}_t \mathcal{C}_t^\top = \left[r_t^{|s-j|} \right]_{s,j=0}^{S-1} \otimes \mathbf{I}_d, \qquad \boldsymbol{\epsilon}_t \sim \mathcal{N}(\mathbf{0}, \mathbf{Q}_t). \tag{15}$$

For $\alpha = 0$, or at $t = 1$, $r_t = 0$ and $q_t = 1$, so $\mathcal{C}_t = \mathbf{I}_m$ and the noise is independent across frames.

B.2 Linear Interpolation under Correlated Noise

The linear interpolation between $\mathbf{y}$ and $\boldsymbol{\epsilon}_t$ incorporates the time dependence of the noise transform. Holding $\mathbf{y}$ and $\boldsymbol{\xi}$ fixed, the interpolation path and its derivative are

$$\mathbf{z}_t = t\mathbf{y} + (1-t)\boldsymbol{\epsilon}_t = t\mathbf{y} + (1-t)\mathcal{C}_t\boldsymbol{\xi}, \tag{16}$$

$$\dot{\mathbf{z}}_t = \mathbf{y} - \boldsymbol{\epsilon}_t + (1-t)\dot{\mathcal{C}}_t\boldsymbol{\xi}. \tag{17}$$

Consequently, for $t < 1$, the conditional distribution along this path is

$$p_t(\mathbf{z} \mid \mathbf{y}, \mathbf{c}) = \mathcal{N}\big(\mathbf{z}; t\mathbf{y}, (1-t)^2\mathbf{Q}_t\big)\,. \tag{18}$$

Marginalizing over $\mathbf{y} \sim p_{\text{data}}(\cdot \mid \mathbf{c})$ defines a probability path from $p_0 = \mathcal{N}(\mathbf{0}, \mathbf{Q}_0)$ to $p_1 = p_{\text{data}}(\cdot \mid \mathbf{c})$.

To express the path derivative in terms of the correlated noise, define

$$\mathbf{B}_t = \dot{\mathcal{C}}_t\mathcal{C}_t^{-1}, \qquad \dot{\boldsymbol{\epsilon}}_t = \mathbf{B}_t\boldsymbol{\epsilon}_t. \tag{19}$$

The operator $\mathbf{B}_t$ describes how the noise transform changes with diffusion time for fixed innovations. Averaging Equation 17 over the conditional distribution of the clean sequence and innovations gives the marginal velocity

$$\begin{aligned}\mathbf{v}_t(\mathbf{z} \mid \mathbf{c}) &= \mathbb{E}[\dot{\mathbf{z}}_t \mid \mathbf{z}_t = \mathbf{z}, \mathbf{c}] \\ &= \mathbb{E}[\mathbf{y} - \boldsymbol{\epsilon}_t \mid \mathbf{z}_t = \mathbf{z}, \mathbf{c}] + (1-t)\mathbf{B}_t\mathbb{E}[\boldsymbol{\epsilon}_t \mid \mathbf{z}_t = \mathbf{z}, \mathbf{c}]\,.\end{aligned} \tag{20}$$

This velocity satisfies the continuity equation $\partial_t p_t + \nabla_{\mathbf{z}} \cdot (p_t\mathbf{v}_t) = 0$. The associated deterministic generative dynamics are therefore

$$\frac{\mathrm{d}\mathbf{z}_t}{\mathrm{d}t} = \mathbf{v}_t(\mathbf{z}_t \mid \mathbf{c}), \qquad \mathbf{z}_0 = \mathcal{C}_0\boldsymbol{\xi}, \quad \boldsymbol{\xi} \sim \mathcal{N}(\mathbf{0}, \mathbf{I}_m). \tag{21}$$

B.3 Training Target and Deterministic Sampling

Algorithm 1 regresses $\mathbf{y} - \boldsymbol{\epsilon}_t$, providing a simple parameterization of the marginal velocity in Equation 20. Given one network prediction $\mathbf{f}_\theta = \mathbf{f}_\theta(\mathbf{z}_t, t \mid \mathbf{c})$, the clean latent, correlated noise, and innovations are estimated as

$$\begin{aligned}\widehat{\mathbf{y}}_\theta &= \mathbf{z}_t + (1-t)\mathbf{f}_\theta, \\ \widehat{\boldsymbol{\epsilon}}_\theta &= \mathbf{z}_t - t\mathbf{f}_\theta, \\ \widehat{\boldsymbol{\xi}}_\theta &= \mathcal{C}_t^{-1}\widehat{\boldsymbol{\epsilon}}_\theta.\end{aligned} \tag{22}$$

These identities follow directly from the interpolation path: $\mathbf{y} = \mathbf{z}_t + (1-t)(\mathbf{y} - \boldsymbol{\epsilon}_t)$ and $\boldsymbol{\epsilon}_t = \mathbf{z}_t - t(\mathbf{y} - \boldsymbol{\epsilon}_t)$. Substituting these estimates into Equation 20 gives the velocity estimate

$$\widehat{\mathbf{v}}_\theta(\mathbf{z}_t, t \mid \mathbf{c}) = \mathbf{f}_\theta + (1-t)\mathbf{B}_t\widehat{\boldsymbol{\epsilon}}_\theta = \mathbf{f}_\theta + (1-t)\dot{\mathcal{C}}_t\widehat{\boldsymbol{\xi}}_\theta. \tag{23}$$

For deterministic sampling, we integrate this velocity by holding the clean estimate $\widehat{\mathbf{y}}_\theta$ fixed within each step from t to $t' > t$. Using Equation 22, the velocity can be written as

$$\widehat{\mathbf{v}}_\theta(\mathbf{z}, t \mid \mathbf{c}) = \frac{\widehat{\mathbf{y}}_\theta - \mathbf{z}}{1-t} + \mathbf{B}_t\left(\mathbf{z} - t\widehat{\mathbf{y}}_\theta\right). \tag{24}$$

For $\tau \in [t, t')$, define the residual $\mathbf{w}_\tau = \mathbf{z}_\tau - \tau\widehat{\mathbf{y}}_\theta$. With the clean estimate fixed, its dynamics satisfy

$$\frac{\mathrm{d}\mathbf{w}_\tau}{\mathrm{d}\tau} = \left(\mathbf{B}_\tau - \frac{\mathbf{I}_m}{1-\tau}\right)\mathbf{w}_\tau. \tag{25}$$

Since

$$\frac{\mathrm{d}}{\mathrm{d}\tau}\left[(1-\tau)\mathcal{C}_\tau\right] = \left(\mathbf{B}_\tau - \frac{\mathbf{I}_m}{1-\tau}\right)(1-\tau)\mathcal{C}_\tau, \tag{26}$$

the residual has the closed-form update

$$\mathbf{w}_{t'} = \frac{1-t'}{1-t}\mathcal{C}_{t'}\mathcal{C}_t^{-1}\mathbf{w}_t. \tag{27}$$

Algorithm 4 DyPro SDE Sampling (w/o CFG)

```
# One active SDE transition: t -> t_next < 1
# kappa: stochasticity; 0 < kappa**2 * (t_next - t) <= 1

v = net(z, t, c)
x = z + (1 - t) * v
e = z - t * v
e_init = decorrelate(e, t)

a = kappa * sqrt(t_next - t)
e_mean = sqrt(1 - a**2) * e_init
e = correlate(e_mean + a * randn_like(e_init), t_next)
z_next = t_next * x + (1 - t_next) * e

mu = t_next * x + (1 - t_next) * correlate(e_mean, t_next)
residual = decorrelate(stopgrad(z_next) - mu, t_next)
scale = (1 - t_next) * a
log_score = avg_latent_dims(normal_log_prob(residual, 0, scale))

return z_next, log_score
```

Substituting $\mathbf{w}_t = (1-t)\widehat{\boldsymbol{\epsilon}}_\theta$ yields the deterministic sampling rule

$$\begin{aligned}\mathbf{z}_{t'} &= t'\widehat{\mathbf{y}}_\theta + (1-t')\mathcal{C}_{t'}\mathcal{C}_t^{-1}\widehat{\boldsymbol{\epsilon}}_\theta \\ &= t'\widehat{\mathbf{y}}_\theta + (1-t')\mathcal{C}_{t'}\widehat{\boldsymbol{\xi}}_\theta. \end{aligned} \tag{28}$$

The update is exact for the within-step dynamics with a fixed clean estimate and is first-order consistent with the learned ODE. It extends continuously to $t' = 1$, where it returns $\widehat{\mathbf{y}}_\theta$.

Algorithm 2 implements this update by applying `decorrelate` at t and `correlate` at t'. Each step uses one evaluation of $\mathbf{f}_\theta$ and transports the estimated innovations to the next timestep. For $\alpha = 0$, $\mathcal{C}_t = \mathbf{I}_m$ throughout, and the update reduces to $\mathbf{z}_{t'} = \mathbf{z}_t + (t'-t)\mathbf{f}_\theta$.

B.4 Stochastic Sampling

A stochastic extension of the deterministic dynamics refreshes the noise in independent-innovation coordinates. Let $\kappa_t \geq 0$ be a prescribed stochasticity schedule independent of the policy parameters. We replace the fixed innovations with the Ornstein–Uhlenbeck process

$$\mathrm{d}\boldsymbol{\xi}_t = -\frac{\kappa_t^2}{2}\boldsymbol{\xi}_t\,\mathrm{d}t + \kappa_t\,\mathrm{d}\mathbf{W}_t, \qquad \boldsymbol{\xi}_0 \sim \mathcal{N}(\mathbf{0}, \mathbf{I}_m), \tag{29}$$

where $\mathbf{W}_t$ is an m-dimensional standard Brownian motion, independent of the clean sequence and history. This process preserves the standard Gaussian distribution of the innovations at every timestep. Consequently, $\mathbf{z}_t = t\mathbf{y} + (1-t)\mathcal{C}_t\boldsymbol{\xi}_t$ retains the conditional distributions in Equation 18.

Applying the product rule, taking conditional expectations given $(\mathbf{z}_t, \mathbf{c})$, and parameterizing them with the network predictions gives

$$\mathrm{d}\mathbf{z}_t = \left[\mathbf{f}_\theta + (1-t)\left(\mathbf{B}_t - \frac{\kappa_t^2}{2}\mathbf{I}_m\right)\widehat{\boldsymbol{\epsilon}}_\theta\right]\mathrm{d}t + \kappa_t(1-t)\mathcal{C}_t\,\mathrm{d}\mathbf{W}_t. \tag{30}$$

Here, the network prediction and noise estimate are evaluated at $(\mathbf{z}_t, t, \mathbf{c})$. Setting $\kappa_t = 0$ recovers the deterministic dynamics.

For a sampling step from t to $t' > t$ with $t' < 1$, we hold the clean estimate $\widehat{\mathbf{y}}_\theta$ fixed and discretize the innovation process in Equation 29 from $\widehat{\boldsymbol{\xi}}_\theta$ using $h = t' - t$ and

$$a = \kappa_t\sqrt{h}, \qquad b = \sqrt{1-a^2}, \qquad 0 \leq a \leq 1. \tag{31}$$

Here, $a^2 + b^2 = 1$ preserves the standard Gaussian reference distribution in innovation coordinates. For an independent $\boldsymbol{\eta} \sim \mathcal{N}(\mathbf{0}, \mathbf{I}_m)$, the resulting update is

$$\mathbf{z}_{t'} = t'\widehat{\mathbf{y}}_\theta + (1-t')\mathcal{C}_{t'}\left(b\mathcal{C}_t^{-1}\widehat{\boldsymbol{\epsilon}}_\theta + a\boldsymbol{\eta}\right). \tag{32}$$

All network estimates are evaluated at the current state $(\mathbf{z}_t, t, \mathbf{c})$. Algorithm 4 implements this update by decorrelating the predicted noise, mixing the resulting innovations with independent Gaussian noise, and applying the noise transform at t'. For $a = 0$, the update reduces to Equation 28.

C TIMESTEP-AWARE POST-TRAINING

Post-training combines a terminal forecast reward with policy updates on selected stochastic transitions. We describe mixed rollouts, timestep-aware reward evaluation, and the local policy objective, together with the training algorithm and configuration.

C.1 MIXED SAMPLING AND WINDOW SCHEDULING

We discretize diffusion time as $0 = t_0 < \cdots < t_N = 1$ and activate a contiguous window $\mathcal{W}_\ell = \{\ell, \ldots, \ell + K - 1\}$, with default size $K = 4$. The rollout policy θ_{old} is held fixed while collecting the group. Transitions inside the window use the stochastic kernel from Appendix B.4, while the remaining transitions use the deterministic update from Appendix B.3:

$$\mathbf{z}_{t_{n+1}} = \begin{cases} \text{Sample}\left[p^{\text{DyPro}}_{\theta_{\text{old}}}(\cdot \mid \mathbf{z}_{t_n}, \mathbf{c})\right], & n \in \mathcal{W}_\ell, \\ \Phi^{\text{DyPro}}_{\theta_{\text{old}}}(\mathbf{z}_{t_n}, t_n, t_{n+1}, \mathbf{c}), & n \notin \mathcal{W}_\ell. \end{cases} \tag{33}$$

Here, p^{DyPro} is the transition distribution induced by Equation 32, and Φ^{DyPro} denotes the deterministic update in Equation 28. The trajectories share an initial DyPro latent and conditioning context, with independent stochastic innovations inside the window.

The stochastic window contains only nonterminal transitions, with $0 < \kappa^2_{t_n}(t_{n+1} - t_n) \leq 1$ for each active step. Its bounds are

$$1 \leq K \leq N - 1, \qquad 0 \leq \ell \leq \ell_{\max} = N - K - 1. \tag{34}$$

The final transition to $t_N = 1$ is deterministic. We start at $\ell = 0$ and cycle through window starts spaced by s_w, switching after every M_{shift} training iterations:

$$\ell \leftarrow \begin{cases} \ell + s_w, & \ell + s_w \leq \ell_{\max}, \\ 0, & \text{otherwise.} \end{cases} \tag{35}$$

With $N = 10$, $K = 4$, $s_w = 2$, and $M_{\text{shift}} = 20$ throughout post-training.

C.2 TIMESTEP-AWARE REWARD EVALUATION

Each rollout produces a final decoded forecast $\hat{\mathbf{x}}^i = D(\mathbf{z}^i_1)$, which is compared with the reference future sequence $\mathbf{x}^*$. The two reward components are defined in Equations 7 and 8: R_{noise} combines pooled low-threshold CSI with SSIM, and R_{clean} combines high-threshold CSI with negative LPIPS. Local max pooling applies only to the low-threshold CSI terms; the other metrics are evaluated on the decoded forecast without this pooling operation. Metric definitions are given in Appendix D.2.

For an active transition n, the terminal reward assigned to trajectory i is $R^i_n = R(\hat{\mathbf{x}}^i, \mathbf{x}^*; t_n)$, as defined in Equation 9. All active transitions are evaluated on the same final forecast. Their metric weights depend on their own diffusion times t_n, so rewards and group-relative advantages are computed separately for each transition. No reward is evaluated on an intermediate noisy latent.

C.3 MIXGRPO OBJECTIVE AND POLICY SCORES

For each active transition n, the group provides a relative measure of forecast quality. We normalize its terminal rewards across the G trajectories:

$$\hat{A}^i_n = \frac{R^i_n - \text{mean}(\{R^j_n\}_{j=1}^G)}{\text{std}(\{R^j_n\}_{j=1}^G) + \varepsilon_A}, \tag{36}$$

where $\varepsilon_A > 0$ stabilizes the denominator. The comparison is made across forecasts at the same transition, rather than across diffusion times. If all group rewards agree, their advantages are zero.

For a stochastic step from t to $t' < 1$ with $a > 0$, the update in Equation 32 defines the conditional density

$$p^{\text{DyPro}}_\theta(\mathbf{z}' \mid \mathbf{z}, \mathbf{c}) = \mathcal{N}\big(\mathbf{z}'; \boldsymbol{\mu}_\theta, \sigma^2 \mathbf{Q}_{t'}\big), \tag{37}$$

$$\boldsymbol{\mu}_\theta = t'\widehat{\mathbf{y}}_\theta + (1 - t')b\mathcal{C}_{t'}\mathcal{C}_t^{-1}\widehat{\boldsymbol{\epsilon}}_\theta, \qquad \sigma = (1 - t')a. \tag{38}$$

All network estimates are evaluated at $(\mathbf{z}, t, \mathbf{c})$; the transition times are implicit in $p_\theta^{\mathrm{DyPro}}$. Whitening the residual as $\mathbf{w}_\theta = \mathcal{C}_{t'}^{-1}(\mathbf{z}' - \boldsymbol{\mu}_\theta)$ gives the joint log density

$$\log p_\theta^{\mathrm{DyPro}}(\mathbf{z}' \mid \mathbf{z}, \mathbf{c}) = -\frac{m}{2}\log(2\pi) - m\log\sigma - \log|\det \mathcal{C}_{t'}| - \frac{\|\mathbf{w}_\theta\|_2^2}{2\sigma^2}. \tag{39}$$

For trajectory i at transition n, we write $\log p_{\theta,n}^i = \log p_\theta^{\mathrm{DyPro}}(\mathbf{z}_{t_{n+1}}^i \mid \mathbf{z}_{t_n}^i, \mathbf{c})$, with $\log p_{\mathrm{old},n}^i$ defined analogously for θ_{old}.

For a stored state and successor from the rollout, the policy ratio is

$$\rho_n^i(\theta) = \frac{p_\theta^{\mathrm{DyPro}}(\mathbf{z}_{t_{n+1}}^i \mid \mathbf{z}_{t_n}^i, \mathbf{c})}{p_{\theta_{\mathrm{old}}}^{\mathrm{DyPro}}(\mathbf{z}_{t_{n+1}}^i \mid \mathbf{z}_{t_n}^i, \mathbf{c})} = \exp\big(\log p_{\theta,n}^i - \log p_{\mathrm{old},n}^i\big), \tag{40}$$

Both policies are evaluated on the same stored transition. The states, terminal rewards, advantages, and old-policy scores are held fixed during the update; gradients flow through the current-policy log density.

When both policies share the time grid, α, and stochasticity schedule, their covariance and whitening Jacobian are identical and independent of θ. Let $\mathbf{w}_{\theta,n}^i$ and σ_n denote the residual and scale in Equation 39 for trajectory i at transition n. Cancelling the shared terms in the two log densities yields

$$\log \rho_n^i(\theta) = \frac{\|\mathbf{w}_{\mathrm{old},n}^i\|_2^2 - \|\mathbf{w}_{\theta,n}^i\|_2^2}{2\sigma_n^2}. \tag{41}$$

The Jacobian can thus be omitted from a score used only in this difference, although it remains part of the absolute joint log density.

Following MixGRPO (Li et al., 2025) and Flow-GRPO (Liu et al., 2025), we maximize the clipped local surrogate

$$\mathcal{J}_{\mathrm{MixGRPO}}(\theta) = \mathbb{E}\left[\frac{1}{G|\mathcal{W}_\ell|}\sum_{i=1}^{G}\sum_{n\in\mathcal{W}_\ell}\min\Big(\rho_n^i(\theta)\hat{A}_n^i, \mathrm{clip}(\rho_n^i(\theta), 1-\varepsilon_c, 1+\varepsilon_c)\hat{A}_n^i\Big)\right], \tag{42}$$

where the expectation is over conditioning examples and old-policy rollouts, $|\mathcal{W}_\ell| = K$, and ε_c is the clipping range. The implemented minimization loss is the negative of this objective. Only the stochastic transitions participate in the surrogate; deterministic transitions still influence the sampled final forecast.

C.4 Training Algorithm

Algorithm 5 presents the full training loop. The helper `clipped_mixgrpo` optimizes the negative of Equation 42, holding the rollout data fixed. Old-policy scores are cached internally, following the convention in Appendix C.3.

The stochastic helper `dypro_sde_step` implements Equation 32 and obtains the next time from the grid. The deterministic helper `ode_step` denotes Φ^{DyPro}, and `max_left` is the bound in Equation 34.

The helper `shared_noise(G)` repeats one DyPro initial latent across the group. Both reward helpers include the specified metric coefficients, and `group_normalize` includes ε_A. Contexts and targets are broadcast as needed. Latent denormalization is included in `decode`.

Algorithm 5 Stage-Aware Post-Training with Window Scheduling

```
# theta: trainable policy initialized from the flow-matching checkpoint
# K: active-window size (default: 4); max_left: last valid window start

left = 0
for iteration, (c, x_gt) in enumerate(dataloader):
    window = range(left, left + K)
    traj[:, 0] = shared_noise(G)

    for n, t in enumerate(timesteps[:-1]):
        if n in window:
            traj[:, n + 1] = dypro_sde_step(theta, traj[:, n], t, c)
        else:
            traj[:, n + 1] = ode_step(theta, traj[:, n], t, c)

    x_sample = decode(traj[:, -1])
    r_noise = R_noise(x_sample, x_gt)
    r_clean = R_clean(x_sample, x_gt)

    for n in window:
        t = timesteps[n]
        rewards[:, n] = (1 - t) * r_noise + t * r_clean

    advantages = group_normalize(rewards[:, window], dim=0)
    loss = clipped_mixgrpo(theta, traj, window, advantages)
    update(theta, loss)

    if (iteration + 1) % shift_interval == 0:
        left += window_stride
        if left > max_left:
            left = 0
```

C.5 Domain Guidance

After skill-aware RL post-training, we adopt Domain Guidance (DoG) (Zhong et al., 2025), which combines an adapted conditional model with the original pretrained model as the unconditional reference. In our setting, the conditional branch uses the RL-adapted parameters θ_{RL} with history $\mathbf{c}$, while the unconditional branch uses the frozen pre-RL parameters θ_{pre} with zero conditioning:

$$\mathbf{v}^{\mathrm{DoG}}(\mathbf{z}_t, t \mid \mathbf{c}) = w\, \mathbf{v}_{\theta_{\mathrm{RL}}}(\mathbf{z}_t, t \mid \mathbf{c}) + (1 - w)\, \mathbf{v}_{\theta_{\mathrm{pre}}}(\mathbf{z}_t, t \mid \emptyset). \tag{43}$$

This retains the pretrained unconditional reference while incorporating skill alignment through the conditional branch. Setting $w = 1$ recovers the RL model without guidance; using θ_{pre} for both branches recovers Equation 11. Post-RL evaluation settings use $w = 1.4$, as listed in Tables 4. Guidance is disabled during RL rollouts and applied only at inference. RL+CFG configuration in Appendix E.3 uses this DoG formulation.

C.6 Post-Training and Reward Configuration

Table 4 summarizes the post-training, reward, and evaluation configurations. The pretrained VAE remains frozen throughout post-training. Each rollout batch is used for one gradient update, followed by synchronization of the rollout policy.

The CSI coefficients incorporate uniform averaging within each threshold subset and the mixing weights for SSIM or negative LPIPS. No additional normalization is applied to the weighted sums in Equations 7 and 8. The listed checkpoints identify the models used for evaluation, rather than the total training budgets.

D Datasets and Evaluation Protocol

D.1 Datasets and Preprocessing

Table 5 summarizes the dataset configurations and split sizes for the forecasting tasks in Section 5.1.

Table 4: Post-training, reward, and evaluation configurations. Reward thresholds are expressed in VIL values for SEVIR and $\mathrm{mm\,h^{-1}}$ for MRMS.

Setting	SEVIR	MRMS
Optimization		
Optimizer	AdamW, $(\beta_1, \beta_2) = (0.9, 0.95)$	
Learning rate	10^{-5}, constant	
Batch size	8	
Policy ratio clipping range ε_c	10^{-4}	
Advantage clipping	$[-5, 5]$	
Rollout		
Group size G	16	
Sampling steps N	10	
Stochastic window size K	4	
Window stride s_w	2	
Window shift interval M_{shift}	20 training iterations (cyclic)	
DyPro correlation strength α	0.5	
CFG during rollout	Disabled	
Reward		
Low-threshold set $\mathcal{T}_{\text{low}}$	$\{16, 74, 133\}$	$\{1, 2, 4\}$
High-threshold set $\mathcal{T}_{\text{high}}$	$\{160, 181, 219\}$	$\{8, 16, 32\}$
Low-threshold CSI max pooling	4×4	
SSIM coefficient λ_s	0.5	
LPIPS coefficient λ_p	0.9	
CSI coefficient λ_τ, $\tau \in \mathcal{T}_{\text{low}}$	$(1-\lambda_s)/\lvert\mathcal{T}_{\text{low}}\rvert$	
CSI coefficient λ_τ, $\tau \in \mathcal{T}_{\text{high}}$	$(1-\lambda_p)/\lvert\mathcal{T}_{\text{high}}\rvert$	
Evaluation		
# steps	1,000	4,000
Sampling	10 Euler steps	
CFG weight	1.4	

Table 5: Dataset configurations. Spatial resolution refers to the source radar grids, and frame size denotes the model input and output size. Forecast frames are shown as observed → predicted; split sizes count sequences.

Setting	SEVIR	MRMS
Data variable	VIL	Precipitation rate
Spatial / temporal res.	1 km / 5 min	0.01° / 10 min
Forecast frames	$5 \rightarrow 20$	$4 \rightarrow 20$
Frame size	128×128	256×256
Train / validation / test	59,530 / 28,145 / 7,220	6,807,528 / 12,000 / 12,000

SEVIR (Veillette et al., 2020) contains spatiotemporal observations of weather events across the United States, each covering 384 km × 384 km. Following prior studies (Gao et al., 2022b; 2023), we use the VIL modality, which has a native spatial resolution of 1 km and a temporal resolution of 5 minutes. Following DiffCast and AlphaPre (Yu et al., 2024; Lin et al., 2025), we downsample the frames to 128×128 and predict 20 future frames from 5 observed frames. Training includes events up to January 1, 2019. Validation uses later events up to October 1, 2019, and testing uses events after that date. Both cutoffs are inclusive for the earlier split and refer to 00:00 UTC. We exclude events with missing VIL data or duplicate VIL records. From each 49-frame event, we extract 25-frame sequences every five frames. Training augmentation uses rotations by multiples of 90° and horizontal or vertical flips, applied consistently across all frames in a sequence. Since SEVIR stores VIL as dimensionless encoded values, we report its color bars and evaluation thresholds without physical units (Veillette et al., 2020).

MRMS (Zhang et al., 2016) is a composite radar dataset collected over the contiguous United States, spanning $20°$N–$55°$N in latitude and $130°$W–$60°$W in longitude at a spatial resolution of $0.01°$ per grid. Following NowcastNet (Zhang et al., 2023), we use data from 2016–2020 for model development and 2021 for testing. Within 2016–2020, we reserve the first day of each month for validation and use the remaining days for training. After excluding known anomalous timestamps, we take the last 24 frames of each continuous 40-frame sequence sampled at 10-minute intervals. For training, we sample sequences with replacement and select 256×256 crops from 512×512 patches, using precomputed sequence and spatial weights. Validation and testing use fixed sequence selections and 256×256 center crops.

Radar and latent normalization. For SEVIR, we normalize VIL values by dividing by 255, following DiffCast (Yu et al., 2024). For MRMS, we cap rain rates at $128\ \mathrm{mm\,h^{-1}}$ following NowcastNet (Zhang et al., 2023), then apply logarithmic normalization to $[-1, 1]$. For both datasets, we standardize VAE latents using dataset-specific means and standard deviations, building on latent rescaling in latent diffusion models (Rombach et al., 2022).

D.2 Evaluation Metrics and Aggregation

For meteorological skill, we use the Critical Success Index (CSI) and Heidke Skill Score (HSS) (Veillette et al., 2020; Yu et al., 2024; Zhang et al., 2023):

$$\mathrm{CSI}_\tau = \frac{\mathrm{TP}_\tau}{\mathrm{TP}_\tau + \mathrm{FP}_\tau + \mathrm{FN}_\tau}, \tag{44}$$

and

$$\mathrm{HSS}_\tau = \frac{2(\mathrm{TP}_\tau \cdot \mathrm{TN}_\tau - \mathrm{FP}_\tau \cdot \mathrm{FN}_\tau)}{(\mathrm{TP}_\tau + \mathrm{FN}_\tau)(\mathrm{FN}_\tau + \mathrm{TN}_\tau) + (\mathrm{TP}_\tau + \mathrm{FP}_\tau)(\mathrm{FP}_\tau + \mathrm{TN}_\tau)}. \tag{45}$$

Here, TP_τ, FP_τ, FN_τ, and TN_τ denote pixelwise true-positive, false-positive, false-negative, and true-negative counts at threshold τ, aggregated over the evaluation set at each lead time. We average CSI and HSS uniformly over lead times and evaluation thresholds.

For perceptual quality, we use LPIPS (Zhang et al., 2018) and SSIM (Wang et al., 2004):

$$\mathrm{LPIPS}(\hat{\mathbf{x}}, \mathbf{x}^*) = \sum_\ell \frac{1}{H_\ell W_\ell} \sum_{h,w,c} a_{\ell c} \big(\phi_\ell(\hat{\mathbf{x}})_{hwc} - \phi_\ell(\mathbf{x}^*)_{hwc}\big)^2, \tag{46}$$

where ϕ_ℓ denotes channel-normalized features at layer ℓ, $a_{\ell c}$ are learned channel weights, and H_ℓ, W_ℓ are the feature-map dimensions. SSIM is defined as

$$\mathrm{SSIM}(\hat{\mathbf{x}}, \mathbf{x}^*) = \frac{(2\mu_{\hat{x}}\mu_{x^*} + c_1)(2\sigma_{\hat{x}x^*} + c_2)}{(\mu_{\hat{x}}^2 + \mu_{x^*}^2 + c_1)(\sigma_{\hat{x}}^2 + \sigma_{x^*}^2 + c_2)}, \tag{47}$$

where μ, σ^2, and $\sigma_{\hat{x}x^*}$ denote local means, variances, and covariance, and c_1, c_2 are stabilization constants. Perceptual scores are spatially averaged within each frame and then uniformly averaged over forecast frames and test sequences. Higher CSI, HSS, and SSIM and lower LPIPS indicate better performance.

E Additional Experimental Results

E.1 Forecast Skill Across Lead Times

Figure 6 compares CSI and HSS across forecast lead times on MRMS and SEVIR, complementing the aggregate results in Table 1. The curves cover 10–200 minutes on MRMS and 5–100 minutes on SEVIR and include NowcastDiT both with and without RL post-training.

E.2 Cumulative Component Ablation

Table 6 reports a cumulative ablation on SEVIR, starting from the standard DiT baseline and successively adding 3D RoPE and QK-Norm, DyPro, and timestep-aware RL. Each row retains all components introduced in the preceding rows.

Table 6: Cumulative component ablation on SEVIR. ↑ indicates higher is better and ↓ indicates lower is better. Bold and underlined values denote the best and second-best results, respectively.

Method	CSI↑	CSI-181↑	CSI-219↑	HSS↑	LPIPS↓	SSIM↑
DiT	0.2926	0.0948	0.0555	0.3733	0.1545	0.7048
+ 3D RoPE, QK-Norm	0.3012	0.1082	0.0676	0.3647	**0.1317**	0.6812
+ DyPro	<u>0.3195</u>	<u>0.1204</u>	**0.0752**	<u>0.4092</u>	0.1486	<u>0.7099</u>
+ RL	**0.3240**	**0.1241**	<u>0.0750</u>	**0.4148**	<u>0.1469</u>	**0.7197**

E.3 CFG AND SKILL-AWARE POST-TRAINING

We adopt Domain Guidance (DoG) (Zhong et al., 2025) to integrate CFG with RL post-training, as detailed in Appendix C.5. Table 7 reports the individual and combined effects of CFG and RL. The combined configuration has the highest aggregate CSI, HSS, and SSIM among these four settings, while CFG alone has the lowest LPIPS, illustrating that the metrics need not improve together.

Table 7: Effects of RL and classifier-free guidance on SEVIR. ↑ indicates higher is better and ↓ indicates lower is better. Bold and underlined values denote the best and second-best results, respectively.

RL	Guidance	CSI↑	CSI-181↑	CSI-219↑	HSS↑	LPIPS↓	SSIM↑
✘	✘	0.2934	0.0956	0.0560	0.3749	0.1541	0.7041
✘	✔	<u>0.3195</u>	<u>0.1204</u>	**0.0752**	<u>0.4092</u>	<u>0.1486</u>	0.7099
✔	✘	0.3109	0.1050	0.0588	0.3961	0.1580	<u>0.7156</u>
✔	✔	**0.3240**	**0.1241**	<u>0.0750</u>	**0.4148**	**0.1469**	**0.7197**

E.4 DYPRO CORRELATION STRENGTH

Table 8 reports the SEVIR sensitivity to α across meteorological and perceptual metrics. Among the listed settings, $\alpha = 0.5$ gives the highest aggregate CSI. This experiment varies the noise correlation strength. All experiments in this ablation are conducted without RL post-training to isolate the effect of DyPro.

Table 8: DyPro correlation strength α on SEVIR. $\alpha = 0$ corresponds to the standard i.i.d. Gaussian noise prior. ↑ indicates higher is better and ↓ indicates lower is better. Bold and underlined values denote the best and second-best results, respectively.

α	CSI↑	CSI-181↑	CSI-219↑	HSS↑	LPIPS↓	SSIM↑
0.00	0.3012	0.1082	0.0676	0.3647	**0.1317**	0.6812
0.25	0.3188	0.1184	**0.0759**	0.4075	0.1485	0.7090
0.33	<u>0.3191</u>	0.1192	0.0742	0.4080	<u>0.1483</u>	0.7097
0.50	**0.3195**	**0.1204**	<u>0.0752</u>	**0.4092**	0.1486	**0.7099**
0.66	0.3185	<u>0.1194</u>	0.0740	0.4077	0.1495	<u>0.7098</u>
0.75	0.3184	0.1184	0.0750	<u>0.4084</u>	0.1494	0.7092

E.5 TIMESTEP-AWARE REWARD VARIANTS

Table 9 defines the six variants in Figures 5(d) and 7. Let C_L and C_H denote mean low-threshold pooled CSI and mean high-threshold CSI, using the threshold sets and pooling in Table 4, and let $S = \text{SSIM}$ and $P = \text{LPIPS}$. The mixed components are $M_s = 0.5C_L + 0.5S$ and $M_d = 0.1C_H - 0.9P$.

For the timestep-aware perceptual reward, $R_{\text{noise}} = 0.1S$ and $R_{\text{clean}} = -0.9P$, so the reward shifts from SSIM at the noise endpoint to negative LPIPS at the clean endpoint. The static counterpart

Table 9: Terminal rewards for the six ablation settings.

Reward	**Static**	**Timestep-aware**
Mixed	$(M_s + M_d)/2$	$(1-t)M_s + tM_d$
CSI	$(C_L + C_H)/2$	$(1-t)C_L + tC_H$
Perceptual	$(0.1S - 0.9P)/2$	$0.1(1-t)S - 0.9tP$

averages these two endpoint rewards, following the same convention as the CSI and mixed variants. This reward changes the relative weighting of structural agreement and perceptual similarity across denoising transitions. All six variants share the cyclic window schedule in Appendix C.1; the contribution of window scheduling is not separately ablated.

E.6 Additional Visualizations

We provide four additional forecast visualizations to complement the qualitative comparisons in the main text. Figures 8 and 9 show SEVIR cases 802 and 6155, comparing NowcastDiT with CasCast and the standard DiT baseline. Figures 10 and 11 show MRMS cases 239 and 1149, comparing NowcastDiT with NowcastNet and the standard DiT baseline. Each figure includes the input observations, ground truth, and forecasts at matched lead times, allowing comparison of precipitation structure and intensity evolution. These selected cases supplement the aggregate evaluation in Table 1.

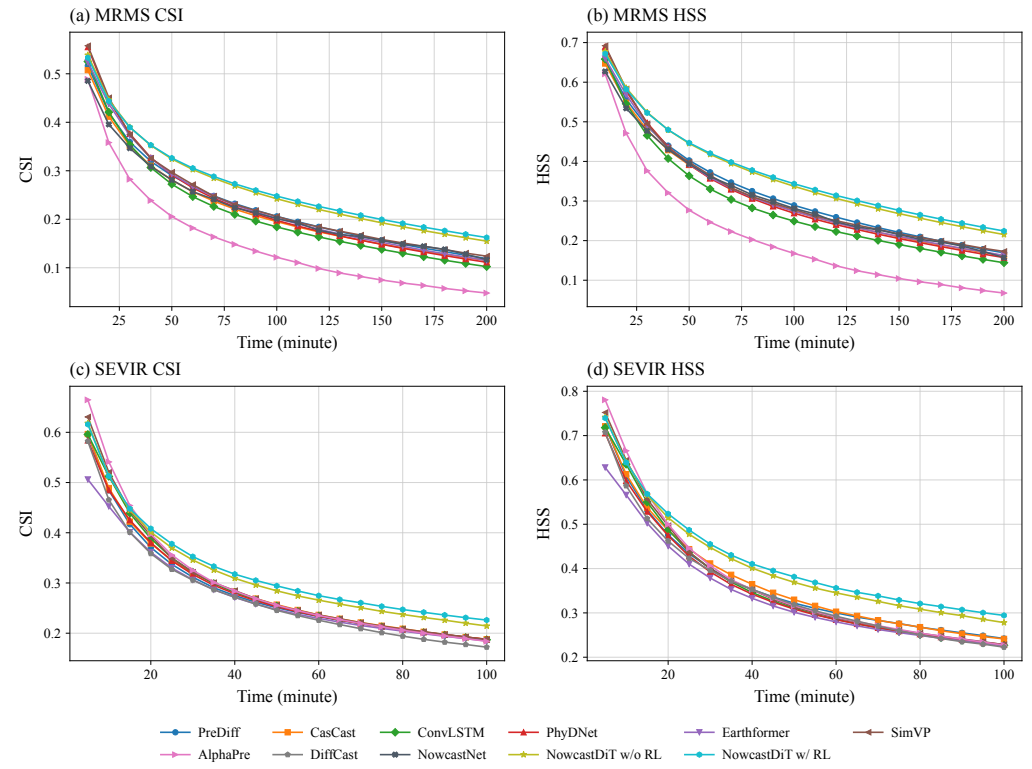


Figure 6: **Forecast skill across lead times on MRMS and SEVIR.** The horizontal axes show forecast lead time in minutes; higher CSI and HSS indicate better performance.

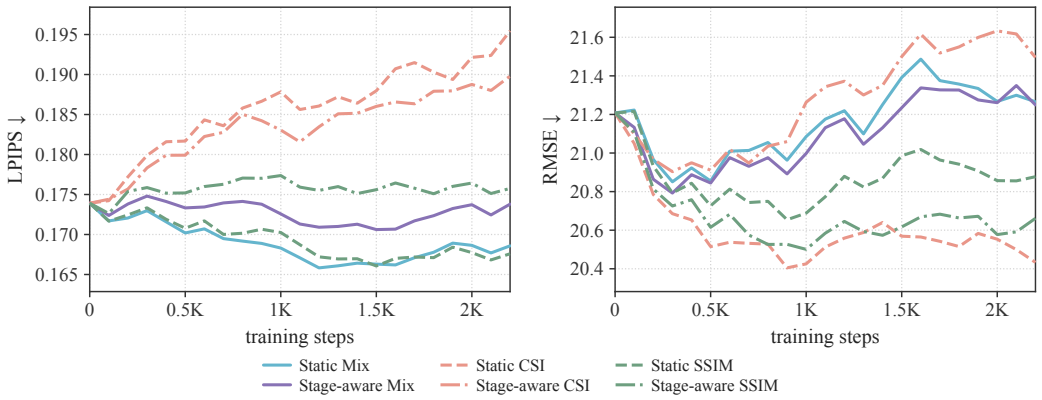


Figure 7: **Additional reward ablations on SEVIR.** LPIPS (left) and RMSE (right) during RL training for the rewards in Table 9. The legend follows Figure 5(d); the timestep-aware perceptual reward uses $0.1(1-t)\text{SSIM} - 0.9t\text{LPIPS}$. Lower values indicate better performance for both metrics.

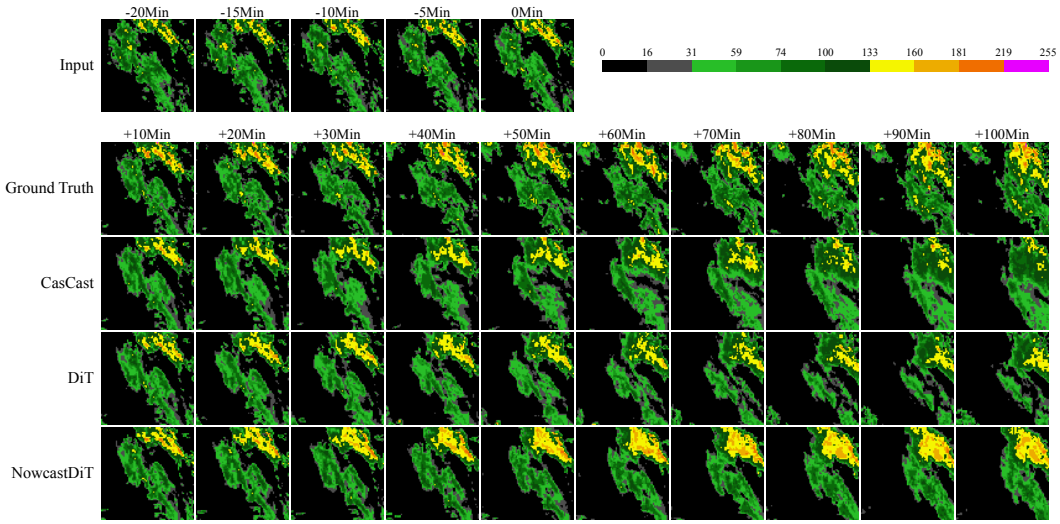


Figure 8: **Qualitative comparison on SEVIR (case 802).** Five input observations span -20 to 0 minutes at 5-minute intervals. Ground truth and forecasts are shown every 10 minutes up to 100 minutes. Colors indicate VIL values.

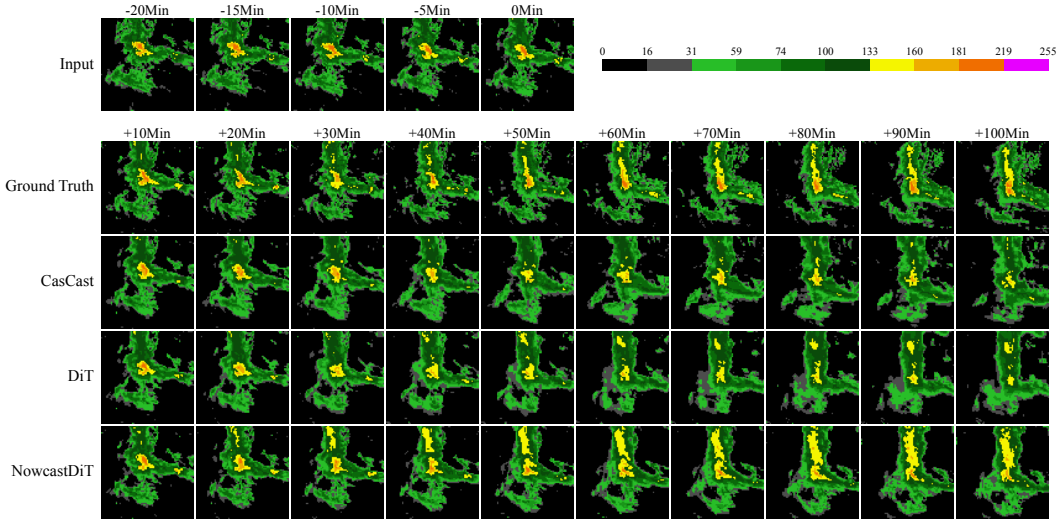


Figure 9: **Qualitative comparison on SEVIR (case 6155).** Five input observations span -20 to 0 minutes at 5-minute intervals. Ground truth and forecasts are shown every 10 minutes up to 100 minutes. Colors indicate VIL values.

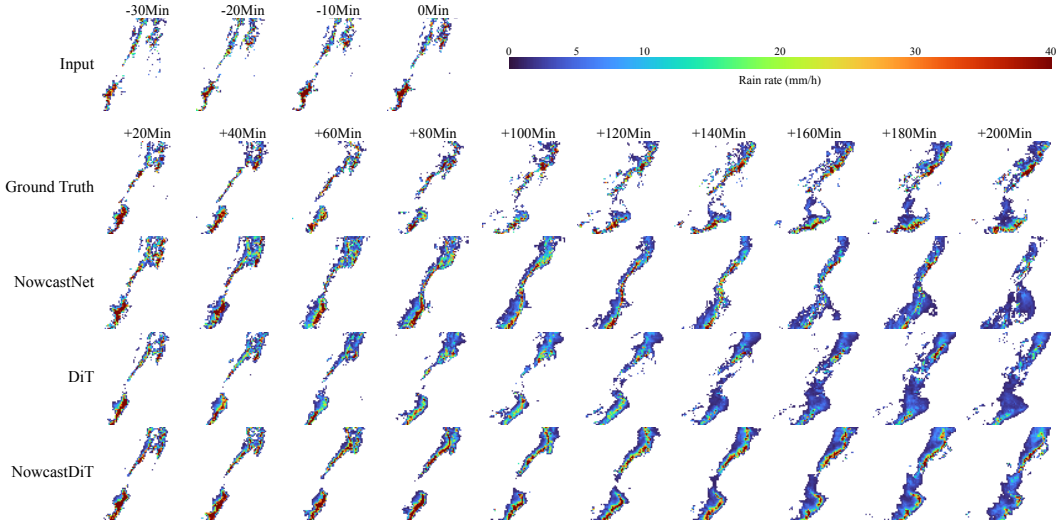


Figure 10: **Qualitative comparison on MRMS (case 239).** Four input observations span −30 to 0 minutes at 10-minute intervals. Ground truth and forecasts are shown every 20 minutes up to 200 minutes. Colors indicate precipitation rate in mm h^{-1}.

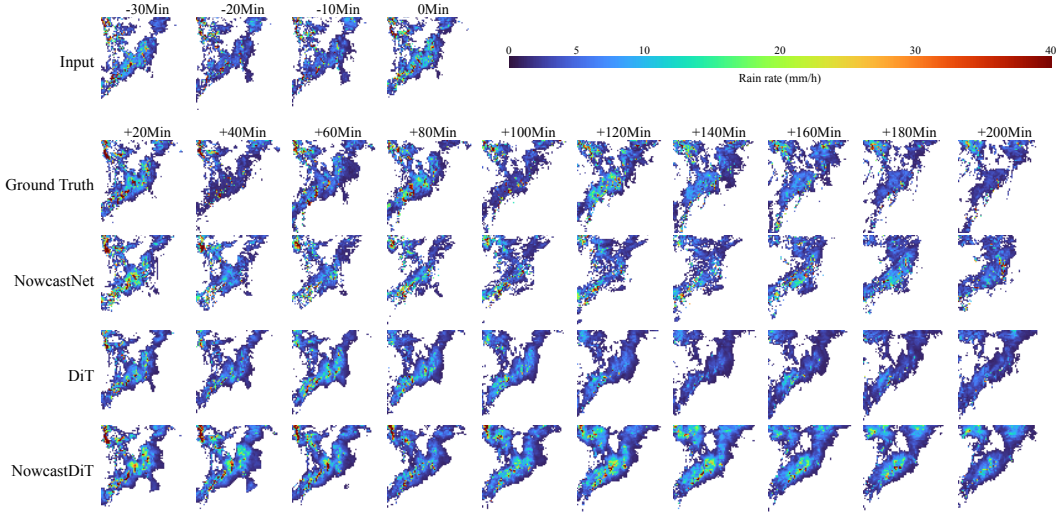


Figure 11: **Qualitative comparison on MRMS (case 1149).** Four input observations span −30 to 0 minutes at 10-minute intervals. Ground truth and forecasts are shown every 20 minutes up to 200 minutes. Colors indicate precipitation rate in mm h^{-1}.