UENR-600K: A LARGE-SCALE PHYSICALLY GROUNDED DATASET FOR NIGHTTIME VIDEO DERAINING

Anonymous authors
Paper under double-blind review

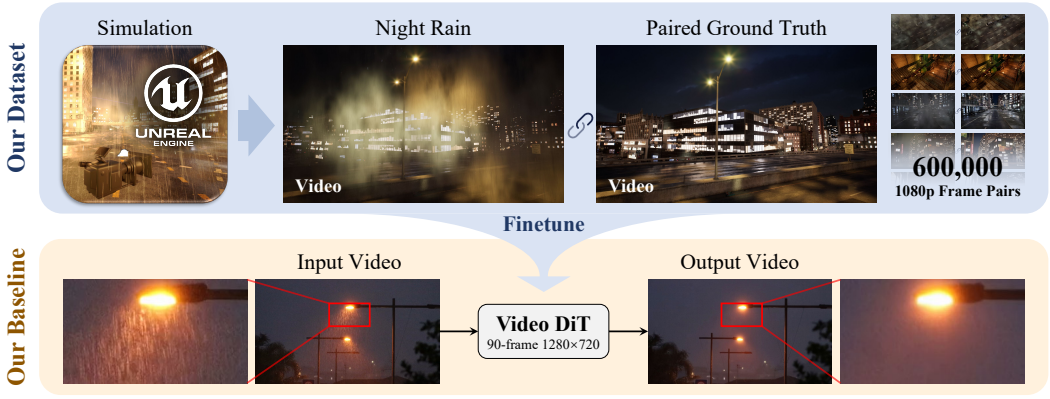


Figure 1: Paper Overview. **Top:** We use Unreal Engine 5 to simulate rain as 3D particles within virtual environments, producing 600,000 paired 1080p frames with physically grounded nighttime rain. Left: a rainy video frame; right: the paired ground truth. **Bottom:** We finetune a video Diffusion Transformer on our dataset for nighttime video deraining. Given a real nighttime rain video (left), our baseline removes rain and rain-induced fog near the streetlamp while preserving scene detail (right; see red crop).

ABSTRACT

Nighttime video deraining is uniquely challenging because raindrops interact with artificial lighting. Unlike daytime white rain, nighttime rain takes on various colors and appears locally illuminated. Existing small-scale synthetic datasets rely on 2D rain overlays and fail to capture these physical properties, causing models to generalize poorly to real-world night rain. Meanwhile, capturing real paired nighttime videos remains impractical because rain effects cannot be isolated from other degradations like sensor noise. To bridge this gap, we introduce **UENR-600K**, a large-scale, physically grounded dataset containing 600,000 1080p frame pairs. We utilize Unreal Engine to simulate rain as 3D particles within virtual environments. This approach guarantees photorealism and physically real raindrops, capturing correct details like color refractions, scene occlusions, and rain curtains. Leveraging this high-quality data, we establish a new state-of-the-art baseline by adapting the Wan 2.2 video generation model. Our baseline treats deraining as a video-to-video generation task, exploiting strong generative priors to almost entirely bridge the sim-to-real gap. Extensive benchmarking demonstrates that models trained on our dataset generalize significantly better to real-world videos. Please also inspect the video comparisons in the Supplementary Material.

1 INTRODUCTION

Nighttime video deraining remains a major challenge compared to its daytime counterpart (Lin et al., 2024a; Zhang et al., 2023). This difficulty stems from the complex optical properties of nighttime rain that differ significantly from daytime conditions (Garg & Nayar, 2007). First, nighttime rain is

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.

chromatic: raindrops refract colored artificial (*e.g*. neon) light rather than appearing as the white streaks typical of daytime rain (Garg & Nayar, 2007; Zhang et al., 2022a). Second, rain visibility is **localized** near light sources, unlike the uniform patterns in daytime videos (Zhang et al., 2023). Third, nighttime rain exhibits a **glimmer effect**, producing sudden high-intensity flashes as raindrops pass through focused light (Garg & Nayar, 2006). Fourth, wind-driven rain forms volumetric **rain curtains** that shift across the scene, visible especially under illumination (Li et al., 2019). These properties are absent in daytime rain, where uniform sunlight instead produces globally visible, colorless streaks.

Current research struggles to address these challenges due to a severe lack of high-quality data (Lin et al., 2024a). Nighttime models could not be trained using daytime datasets (Yang et al., 2017; Chen et al., 2018; Guo et al., 2023) because they lack these specific lighting interactions, and single-image datasets (Wang et al., 2019; Li et al., 2022; Zhang et al., 2022a) fail to provide either temporal rain dynamics or scene consistency. Furthermore, capturing large-scale real-world paired videos at night is impractical due to prohibitive collection costs and entangled degradations (*e.g*., severe sensor noise) that make isolating the rain effect impossible (Ba et al., 2022). Consequently, for nighttime video deraining, existing studies rely on limited synthetic datasets (Patil et al., 2022; Xue et al., 2025). However, these datasets essentially overlay rain as a global layer onto 2D video frames, without modeling scene content or lighting. Therefore, the resulting rain appears as floating, repetitive foggy patterns rather than realistic, localized, chromatic streaks.

To bridge this gap, we present UENR-600K, a large-scale dataset for nighttime video deraining. Unlike previous overlay methods, we utilize Unreal Engine 5 to simulate rain within a virtual raining environment. In this environment, we use cinematic cameras to capture paired videos, in which raindrops and rain curtains are correctly occluded by scene objects and exhibit accurate chromaticity. We generate 600,000 pairs of 1080p frames with extensive diversity. We dynamically control parameters such as wind direction, rain intensity, motion blur, camera parameters, and color grading.

To demonstrate the utility of this dataset, we retrain six existing deraining methods on both our dataset and SynNightRain (Patil et al., 2022), and conduct extensive benchmarking on real and synthetic videos. The models span CNN, RNN, Transformer, and diffusion architectures. We also establish a strong baseline by adapting the Wan 2.2 video generation architecture for deraining, and evaluate all methods on real nighttime rain videos using a vision-language model (VLM) as judge. Training on our dataset leads to better real-world deraining than SynNightRain for most methods, with our baseline deraining method rated best in 95.5% of test videos, averaged over VLM and human evaluations. On real videos, our baseline, by leveraging Wan's strong generative prior, almost entirely bridges the sim-to-real gap, allowing it to effectively handle night rain-specific phenomena such as chromatic rain, localized streaks, and rain-induced fog.

Our contributions are summarized as follows:

- **A dataset.** We propose UENR-600K, a large-scale, physically grounded dataset for nighttime video deraining, containing 600K 1080p frame pairs rendered in Unreal Engine.
- **A baseline.** We adapt the Wan 2.2 video generation model into a video-to-video architecture for nighttime video deraining. Finetuned on our dataset, this model achieves a dominating 95.5% average preference rate on real videos.
- **Benchmarking.** We conduct a comprehensive analysis of seven diverse deraining methods, including our own baseline, on both synthetic and real rains, demonstrating the advantage of our dataset compared to existing synthetic datasets. For most methods, training on our data produces better deraining results, demonstrating that high-quality synthetic data improves real-world generalization.

2 Related Works

Video deraining aims to remove rain streaks and raindrops from video sequences while preserving scene content and temporal consistency. Existing methods address this task through supervised temporal modeling (Chen et al., 2018; Liu et al., 2018; Zhang et al., 2022b; Wang et al., 2022), self-supervised or semi-supervised learning (Yang et al., 2020; Yue et al., 2021; Yan et al., 2021), and general video restoration architectures (Ghasemabadi et al., 2024). Notably, nearly every method

Table 1: Overview of deraining restoration datasets. Among the datasets surveyed, our dataset is the only one providing video data of in-scene nighttime rain.

Dataset	Year	Venue	Rain Type[a]	GT Type[b]	#Rainy Frames[c]	#GT Frames[c]	Resolution	Night[d]	Video[e]	Raining In-Scene[f]
Rain200L/H (Yang et al., 2017)	2017	CVPR	Synth.	Real	4,000	4,000	435×366			
DDN-Data (Fu et al., 2017)	2017	CVPR	Synth.	Real	13,000	13,000	489×428			
DID-Data (Zhang & Patel, 2018)	2018	CVPR	Synth.	Real	13,200	13,200	512×512			
NTURain[g] (Chen et al., 2018)	2018	CVPR	Synth.	Real	-	-	640×480		✓	
RainSynAll25 (Liu et al., 2018)	2018	CVPR	Synth.	Real	-	-	-		✓	
Rain800 (Zhang et al., 2020)	2020	TCSVT	Synth.	Real	800	800	518×419			
SPA-Data (Wang et al., 2019)	2019	CVPR	Real	Derived	29,500	29,500	256×256	✓		
Outdoor-Rain (Li et al., 2019)	2019	CVPR	Synth.	Real	10,500	10,500	720×480			
RainCityscapes (Hu et al., 2019)	2019	CVPR	Synth.	Real	10,620	10,620	2048×1024			
Rain13k (Jiang et al., 2020)	2020	CVPR	Synth.	Real	13,700	13,700	482×419			
RainDirection (Liu et al., 2021)	2021	ICCV	Synth.	Real	3,300	3,300	1945×1444			
RainDS (Quan et al., 2021)	2021	CVPR	Both	Real	5,800	5,800	818×460			
GT-Rain (Ba et al., 2022)	2022	ECCV	Real	Real	31,500	31,500	666×339			
SynNightRain (Patil et al., 2022)	2022	ECCV	Synth.	Real	6,000	6,000	1920×1080	✓	✓	
RealRain-1K (Li et al., 2022)	2022	-	Real	Derived	1,120	1,120	1512×973	✓		
GTAV-NightRain (Zhang et al., 2022a)	2022	-	Synth.	Synth.	14,146	14,146	1920×1080	✓		✓
LHP-Rain[h] (Guo et al., 2023)	2023	ICCV	Real	Derived	1,000,000 3,000	1,000,000 3,000	256×256 1920×1080	✓ ✓		
HQ-Rain (Chen et al., 2025)	2023	-	Synth.	Real	5,000	5,000	1367×931			
4K-Rain13k (Chen et al., 2026)	2026	TMM	Synth.	Real	13,000	13,000	3840×2160			
RAVD[i] (Xue et al., 2025)	2025	PR	Synth.	Real	1,147	1,147	640×480	✓	✓	
AllWeatherNight (Liu et al., 2026)	2026	AAAI	Both	Real	10,000	10,000	640×360	✓		
UENR-600K (Ours)	2026	-	Synth.	Synth.	600,000	600,000	1920×1080	✓	✓	✓

[a] **Rain Type**: Synth. = synthetically generated, Real = captured in real rain, Both = mixture. [b] **GT Type**: the source of clean counterparts; Real = separately captured clean images, Synth. = rendered clean frames, Derived = estimated from rainy inputs (*e.g.*, via temporal aggregation).
[c] **#Rainy/GT Frames**: total number of rainy and clean frames. [d] **Night**: includes nighttime rain data. [e] **Video**: provides video sequences (not just individual images).
[f] **Raining In-Scene**: rain is simulated within a virtual scene where it interacts with scene objects and lighting, as opposed to being overlaid onto existing footage.
[g] Contains 4,667 rain videos and 2,677 ground-truth videos. [h] Only counted open-sourced partitions. [i] Open-sourced partition (no nighttime deraining data).

introduces a new dataset tailored to its target scenario (Chen et al., 2018; Liu et al., 2018; Wang et al., 2022; Patil et al., 2022; Xue et al., 2025), reflecting the tight coupling between data quality and model performance. This pattern highlights the critical role of training data, particularly for underexplored conditions such as nighttime.

Nighttime deraining datasets. While numerous datasets (summarized in Tab. 1) target daytime deraining, nighttime data remain scarce (Lin et al., 2024a). Many nighttime data contain images only (Wang et al., 2019; Li et al., 2022; Zhang et al., 2022a), which cannot be used to train video models. Existing nighttime video datasets include SynNightRain (Patil et al., 2022) and RAVD (Xue et al., 2025), of which only SynNightRain releases its nighttime video data. Although RAVD (Xue et al., 2025) uses a Direct3D particle system for rain rendering, both datasets are constructed by overlaying synthesized rain globally onto the original footage. Such overlayed rain could not respond to scene objects or artificial lighting. As a result, they fail to produce nighttime rain properties such as chromaticity (raindrops refracting colored artificial light) or the glimmer effect (sudden high-intensity flashes near focused light sources) (Zhang et al., 2022a; 2023). These properties are also why daytime datasets cannot be used for nighttime settings. To align with nighttime rain properties, GTAV-NightRain (Zhang et al., 2022a) explore using a game engine (GTA-V) to capture paired data by toggling rain, but this dataset contains only 14,146 image pairs with no video data.

Furthermore, capturing real-world paired nighttime video is fundamentally impractical for two reasons: first, collecting large-scale paired videos is prohibitively costly; second, nighttime environments introduce entangled degradations (such as severe sensor noise), making it nearly impossible to isolate the rain effect. To date, no large-scale, physically grounded nighttime video deraining dataset exists, constraining progress in this field.

Video generation and editing. Beyond paired training data, generative priors from large-scale models offer another avenue for restoration. Diffusion-based video generation models (Brooks et al., 2024; Team Wan, 2025; Blattmann et al., 2023; Yang et al., 2025) synthesize videos from text or image inputs and learn strong priors about scene appearance and motion dynamics. These models, however, are designed for generation rather than editing; adapting them for specific restoration tasks such as deraining (Özdenizci & Legenstein, 2023; Lin et al., 2024b; Zhou et al., 2024) requires high-quality, physically grounded paired data to steer their capabilities correctly. This further motivates our dataset construction.

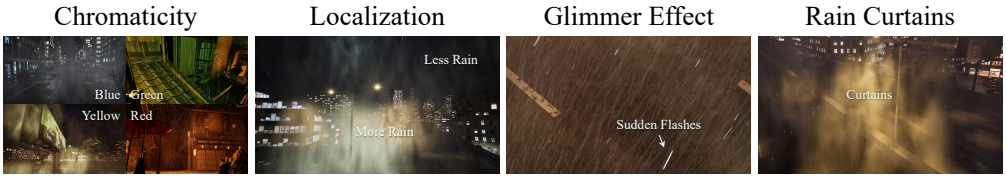


Figure 2: Properties of nighttime rain illustrated with frames from our dataset. **Chromaticity:** raindrops refract colored artificial light (blue, yellow, green, red) rather than appearing white. **Localization:** rain is visible near light sources but fades in unlit regions. **Glimmer effect:** raindrops produce sudden high-intensity flashes as they pass through focused light beams. **Rain curtains:** wind-driven sheets of rain form volumetric, shifting patterns.

3 The UENR-600K Dataset

3.1 Defining Nighttime Video Deraining

Nighttime rain is fundamentally more complex than daytime rain due to interactions with scene geometry and artificial lighting. We define the unique features of nighttime rain across four key aspects. **Chromaticity:** Raindrops act as refractive lenses. For example, rain passing near colored light sources like neon signs correctly refracts the local color rather than appearing white. **Localization:** Rain visibility is highly dependent on artificial light sources, such as the light cones of street lamps. **Glimmer Effect:** As raindrops rapidly pass through focused beams of light, they exhibit sudden and high-intensity flashes. **Rain Curtains:** Wind and heavy precipitation create shifting and volumetric sheets of rain. Based on these features, we define *nighttime video deraining* as the removal of rain streaks and rain-induced effects, including rain curtains, rain-induced fogs, and the glimmer effect, while ensuring all other video content remains unchanged.

3.2 Rain Simulation and Rendering

To capture these nighttime rain phenomena, we abandon the standard practice of 2D rain overlays (Patil et al., 2022). Instead, we utilize Unreal Engine 5 to construct a true 3D virtual raining environment. In this environment, rain exists as physical particles rather than a global layer. This particle-based approach guarantees physical realism. Because the rain exists in the 3D space, raindrops are naturally occluded by scene objects. Furthermore, the rain dynamically responds to the environment by refracting nearby artificial light colors and becoming illuminated by specific light sources to produce localized rain effects. We capture the paired clear and raining videos within this environment using virtual cinematic cameras.

3.3 Dataset Diversity and Parameterization

To prevent models from overfitting to specific camera views or specific types of rain, we introduce extensive variance into our rendering pipeline. We generate continuous large-scale camera movement through the 3D environments to capture numerous perspectives, including tracking, aerial, and backward-facing angles, so as to diversify the scenes and motion dynamics appearing in the video. This camera motion also introduces temporal dynamics to the captured video. Alongside camera movement, we randomize variables across three main categories:

- **Weather dynamics:** Rain intensity, rain curtain density, gust amount, wind speed, and wind direction.
- **Rain streak optical effects:** Motion blur intensity and glimmer intensity.
- **Cinematography:** Aperture (f/2.8 to f/5.6), focal length (8mm to 120mm), and white balance (3500K to 6000K).

Using this pipeline, we render a total of 600,000 paired frames, containing scenes that range from North American cities and skylines to Eastern alleyways with dense, colorful illumination.

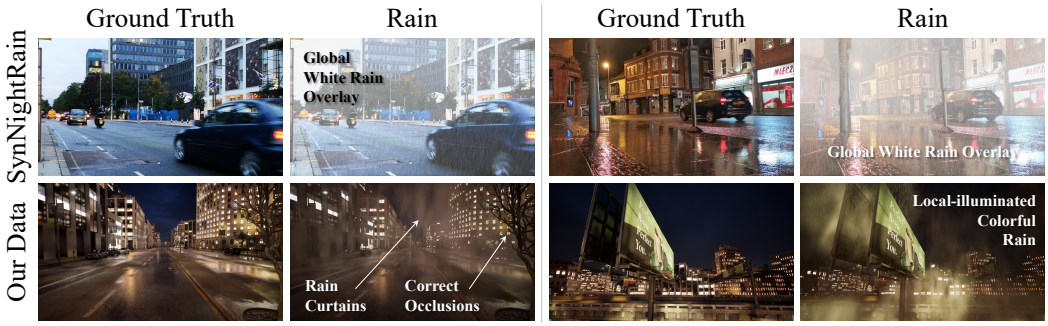


Figure 3: Comparison of rain synthesis between SynNightRain (Patil et al., 2022) (top row) and our dataset (bottom row). Each pair shows a ground-truth frame alongside its rainy counterpart. SynNightRain overlays rain as a global white layer that uniformly covers the entire frame, without responding to scene geometry or lighting. Our dataset simulates rain within a virtual scene: raindrops are correctly occluded by scene objects, form volumetric rain curtains, and refract local artificial light to produce colorful, spatially varying streaks.

4 METHODOLOGY

4.1 MODEL ARCHITECTURE

We establish a baseline for nighttime video deraining by adapting the Wan 2.2 video generation model (Team Wan, 2025) into a video-to-video architecture. As illustrated in Fig. 4, we begin by encoding the input rainy video into a sequence of *condition tokens*. We then concatenate these condition tokens with a parallel sequence of *generation tokens*. To ensure strict spatial and temporal correspondence between the input and output videos, we apply identical positional embeddings to both sets of tokens. During the subsequent denoising process, the model must not corrupt the input condition. To ensure this, we introduce a unidirectional mask into the self attention maps of the diffusion transformer blocks. This mask actively prevents the condition tokens from attending to the generation tokens. Finally, we isolate the denoised generation tokens and decode them to produce the clean video output.

4.2 FINETUNING AND INFERENCE OBJECTIVES

We finetune this model using a flow matching objective (Team Wan, 2025). This objective trains a network v_θ to predict a ground truth velocity v_t. We define this velocity along a linear path x_t interpolating between pure noise $x_0 \sim \mathcal{N}(0, I)$ and the clean latent tokens x_1 (Esser et al., 2024). Because the path $x_t = tx_1 + (1-t)x_0$ is linear, the ground truth v_t is the exact derivative of the path:

$$v_t = \frac{dx_t}{dt} = x_1 - x_0. \tag{1}$$

We optimize the model using a mean squared error loss to predict this exact velocity. Let x^{con} denote condition tokens (input video) and x^{gen} denote generation tokens (output video), and c_{text} be a fixed text prompt's embedding ("No rain video"). The loss function is then defined as:

$$L = \mathbb{E}_{x_1^{\text{con}}, x_t^{\text{gen}}, t} \| v_\theta(x_1^{\text{con}}, x_t^{\text{gen}}, c_{\text{text}}, t) - v_t \|^2. \tag{2}$$

During inference, we produce the final derained video by integrating the predicted velocity from $t = 0$ to 1:

$$x_1^{\text{gen}} = x_0 + \int_0^1 v_\theta(x_1^{\text{con}}, x_t^{\text{gen}}, c_{\text{text}}, t) dt. \tag{3}$$

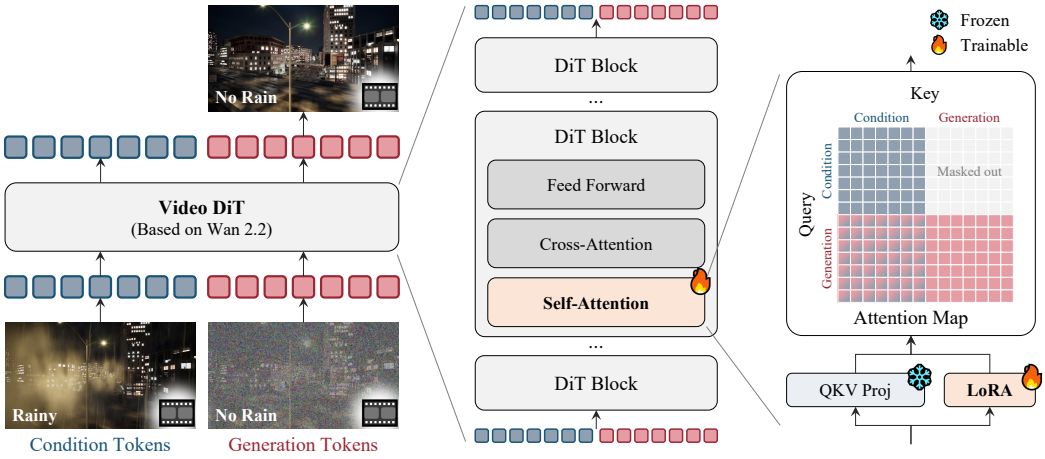


Figure 4: Our baseline architecture, adapted from the Wan 2.2 Video DiT. The rainy input is encoded into condition tokens (blue) and concatenated with generation tokens (red); the DiT denoises only the generation tokens while using the condition tokens as context. A unidirectional attention mask prevents condition tokens from attending to generation tokens, keeping the input uncorrupted.

5 Experiment

5.1 Experiment Setup

5.1.1 Datasets and Deraining Models

Datasets. We train all methods on two nighttime rain datasets: our UENR-600K (sampled 498,200 training / 1,800 test frames) and SynNightRain (Patil et al., 2022) (11,915 / 1,600 frames), the only publicly available nighttime video deraining dataset (Tab. 1). For real-world evaluation without ground truth, following prior work, we collect 124 nighttime rain videos (11,160 frames) from Pexels (Pexels, 2026), a large-scale video sharing platform whose diverse contributor base provides footage videos, including diverse real nighttime rain scenes. All frames are at 1280×720 resolution. Dataset splits are summarized in Tab. 5 (Appendix A).

Baseline deraining models. We train seven methods on both datasets, spanning four architecture families. **CNN-based:** UConNet (Zhuang et al., 2022) and RDD-Net (Wang et al., 2022). **RNN-based:** ESTINet (Zhang et al., 2022b). **Transformer-based:** RLP (Zhang et al., 2023) and Turtle (Ghasemabadi et al., 2024). **Diffusion-based:** WeatherDiff (Özdenizci & Legenstein, 2023) and our baseline. The six existing methods learn deraining from scratch, while our baseline finetunes a pretrained video generation model. All methods take video inputs except WeatherDiff, which processes single images.

Training and inference. All methods follow their original training configurations, with total gradient updates matched for fair comparison. Existing methods process fixed-size patches via non-overlapping sliding windows; our baseline operates at full 1280×720 resolution on 90-frame clips without patch decomposition. Full details are in Appendix B.

5.1.2 Evaluation and Metrics

Evaluating real nighttime rain deraining. We collect 124 real nighttime rain videos from Pexels (Pexels, 2026) (90 frames each, 1280×720) and run all seven methods on them. Since no ground truth exists, we perform VLM and human evaluation. Given output frames from competing methods, the evaluator selects "which frame does the deraining best". VLMs also write a justification for each evaluation, based on rain removal, detail preservation, artifact absence, and overall quality. Method labels are randomized per evaluation to avoid position bias (VLM models, prompt used and full protocol are in Appendix C).

Table 2: Comparison of deraining results (dataset-dimension) on real nighttime rain videos. A VLM or human evaluator rates "SynNightRain is better", "Our Data is better", or "Tie" based on rain removal and detail preservation. Input sizes are defined by each deraining method. Training on our dataset yields preferred results for all methods with $\geq 128\times128$ inputs. WeatherDiff is an exception, because 64×64 input limits its ability distinguishing localized rain effects from clean regions.

		VLM (Claude)			VLM (Qwen)			Human		
	Input Size	**SynNightRain**	**Tie**	**Our Data**	**SynNightRain**	**Tie**	**Our Data**	**SynNightRain**	**Tie**	**Our Data**
ESTINet	224×224	16.4%	6.5%	**77.2%**	26.6%	0.0%	**73.4%**	19.0%	23.8%	**57.1%**
RDD-Net	128×128	32.3%	1.6%	**66.1%**	48.4%	0.0%	**51.6%**	23.8%	23.8%	**52.4%**
RLP	256×256	27.7%	8.6%	**63.7%**	37.4%	0.0%	**62.6%**	23.8%	19.0%	**57.1%**
Turtle	128×128	16.1%	6.2%	**77.7%**	34.9%	0.0%	**65.1%**	14.3%	19.0%	**66.7%**
UConNet	128×128	3.2%	1.3%	**95.4%**	8.6%	0.0%	**91.4%**	9.5%	19.0%	**71.4%**
WeatherDiff	64×64	**91.7%**	3.0%	5.4%	**85.5%**	0.0%	14.5%	**71.4%**	19.0%	9.5%
Our Baseline	Full	19.6%	15.1%	**65.3%**	18.8%	0.0%	**81.2%**	14.3%	28.6%	**57.1%**
Average	–	29.6%	6.0%	**64.4%**	37.6%	0.0%	**62.4%**	25.2%	21.7%	**53.0%**

Evaluating synthetic nighttime rain deraining. Both datasets provide a test set with paired ground-truth frames, so we also report PSNR and SSIM on them. These scores measure how well each method fits the rain it was trained on, and how far the two datasets sit from each other. Appendix B.1 gives the full analysis.

Evaluating video temporal consistency. We measure temporal consistency with two complementary metrics. **Average frame difference (AFD)** computes the mean LPIPS (Zhang et al., 2018) between all pairs of consecutive output frames; lower values indicate smoother transitions. **VLM temporal rating** uses a VLM to rate each method's output on a 1 to 5 scale (5 = best). For each video clip, the VLM views four consecutive output frames together with pre-computed inter-frame difference maps that highlight pixel changes between adjacent frames. Bright regions in the difference maps reveal flickering or inconsistent rain removal, while a temporally consistent output produces mostly dark difference maps (full protocol in Appendix D). The Claude evaluations cost $197.95 in total (Tab. 10); the Qwen judge runs locally and the human study uses no API.

5.2 Dataset Comparison

Training on our dataset improves real-world deraining. For each method, we train two versions (one on our dataset, one on SynNightRain) and ask VLM judges and human annotators to select the better result on real nighttime rain videos (Tab. 2). Training on our dataset yields preferred results for all CNN, RNN, and Transformer methods, as well as our diffusion-based baseline. The exception is WeatherDiff, a diffusion model operating on 64×64 inputs, covering less than 1% of a 1280×720 frame. Without sufficient spatial context, the model cannot distinguish localized rain phenomena in our data (such as rain curtains near light sources) from clean dark regions, producing spurious foggy artifacts at inference. SynNightRain's globally uniform rain does not create this difficulty. For all methods with sufficient receptive fields, training on our dataset consistently improves real-world deraining across all architecture families. Additionally, models trained on SynNightRain tend to suppress scene content in dark regions, because SynNightRain overlays rain as a global bright layer and models learn to remove brightness across the entire frame.

Our dataset enables the baseline to handle diverse nighttime rain conditions. Fig. 6 compares our baseline finetuned on each dataset across five real scenes. The baseline finetuned on SynNightRain struggles with chromatic rain near colored lighting (row 1), localized rain around streetlamps (row 3), and rain-induced fog under direct illumination (rows 4 and 5; see red crops). The version finetuned on our dataset handles all these conditions effectively. Our dataset simulates these nighttime-specific phenomena through 3D particle rendering (Sec. 3.2), while SynNightRain does not model nighttime rain properties like chromaticity or localization.

5.3 Deraining Method Comparison

Quantitatively, our baseline achieves the best deraining preference among all methods. Experiments in Tab. 3 essentially answer "which method gives the best deraining result on real nighttime rain videos" when other variables are controlled. Our baseline dominates both settings, achiev-

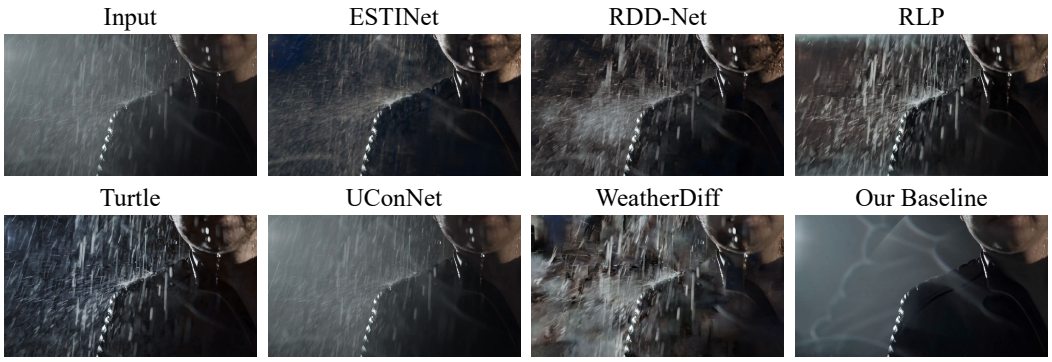


Figure 5: Qualitative comparison of the deraining performance of all methods trained on UENR-600K (our data). Our baseline removes the rain cleanly, while the others do not.

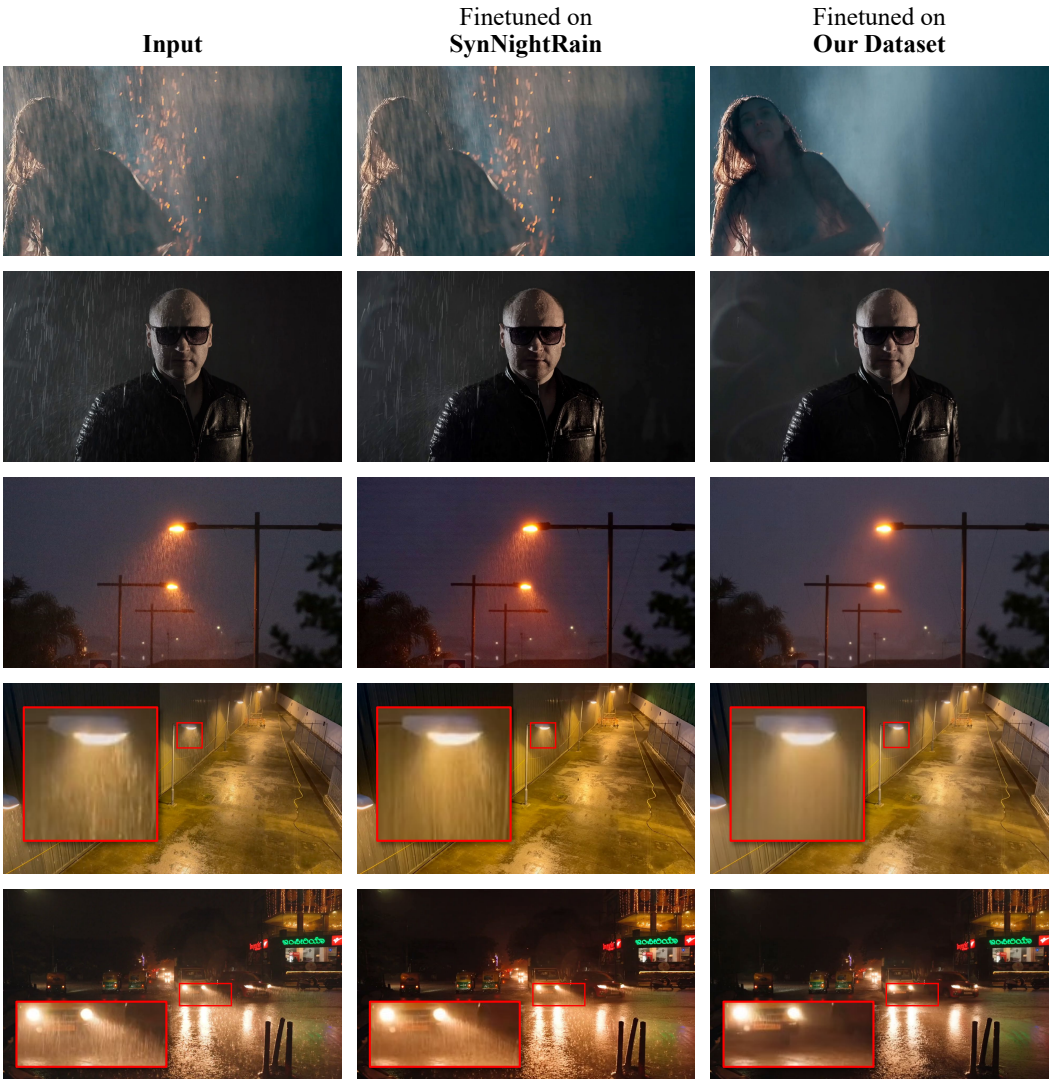


Figure 6: Our baseline finetuned on SynNightRain versus UENR-600K (our data) across five **real** nighttime rain scenes from Pexels. The SynNightRain-finetuned baseline struggles with chromatic rain near colored lighting (row 1), localized rain around streetlamps (row 3), and rain-induced fog (rows 4-5; see red crops). Finetuning on UENR-600K handles these conditions effectively.

Table 3: Comparison of deraining results (method-dimension) on real nighttime rain videos. A VLM or human evaluator selects the best result among all seven methods trained on the same dataset. Our baseline, the only pretrained method, dominates in both settings, taking 95.5% of the selections when trained on our dataset and 67.1% when trained on SynNightRain, averaged over the three evaluators. Each column is normalized over the seven methods.

			VLM (Claude)			VLM (Qwen)			Human		
	Architecture	**Pretrained**	**SynNightRain**	**Our Data**	**Avg**	**SynNightRain**	**Our Data**	**Avg**	**SynNightRain**	**Our Data**	**Avg**
ESTINet	CNN+RNN		13.0%	5.6%	9.3%	15.2%	1.6%	8.4%	9.5%	4.8%	7.2%
RDD-Net	CNN		10.6%	0.0%	5.3%	21.2%	0.0%	10.6%	4.8%	0.0%	2.4%
RLP	Transformer		4.1%	0.0%	2.1%	5.1%	1.6%	3.4%	4.8%	0.0%	2.4%
Turtle	Transformer		0.8%	0.0%	0.4%	0.0%	0.0%	0.0%	0.0%	0.0%	0.0%
UConNet	CNN		0.0%	0.0%	0.0%	1.7%	0.0%	0.9%	4.8%	0.0%	2.4%
WeatherDiff	Diffusion		1.6%	0.0%	0.8%	1.7%	0.0%	0.9%	0.0%	0.0%	0.0%
Our Baseline	Diffusion	✓	**69.9%**	**94.4%**	**82.2%**	**55.1%**	**96.8%**	**76.0%**	**76.2%**	**95.2%**	**85.7%**

Table 4: Temporal consistency of deraining results on real nighttime rain videos, evaluated by two complementary metrics. **AFD**↓: average inter-frame LPIPS ($\times 10^2$; lower is better), measuring perceptual difference between consecutive output frames. **VLM**↑: VLM-rated temporal smoothness on a 1 to 5 scale (higher is better).

	SynNightRain			**Our Data**			**Average**		
	AFD↓	VLM-Claude↑	VLM-Qwen↑	AFD↓	VLM-Claude↑	VLM-Qwen↑	AFD↓	VLM-Claude↑	VLM-Qwen↑
ESTINet	**6.40**	3.22	**3.04**	5.70	3.32	2.91	6.05	3.27	2.98
RDD-Net	7.04	3.37	2.90	9.93	2.37	2.28	8.49	2.87	2.59
RLP	7.75	3.11	2.94	10.11	2.60	2.29	8.93	2.86	2.62
Turtle	8.55	2.71	2.59	8.08	3.07	2.57	8.32	2.89	2.58
UConNet	7.94	2.47	2.24	9.34	3.03	2.69	8.64	2.75	2.47
WeatherDiff	10.56	2.13	2.18	22.79	1.04	1.19	16.68	1.59	1.69
Our Baseline	6.88	**3.56**	2.98	**2.47**	**4.46**	**3.81**	**4.68**	**4.01**	**3.40**

ing an average of 81.3% over VLM and human evaluation. The performance is followed by ESTINet and RDD-Net (8.3% and 6.1% average performance), and then the remaining methods (below 3%). Our baseline, as the only pretrained method, benefits from both its strong generative prior and our physically grounded data.

Qualitatively, on real videos, most methods still leave visible rain or introduce artifacts. Trained on our dataset, traditional methods (ESTINet, RDD-Net, RLP, Turtle, UConNet) reduce rain but leave visible streaks in heavy-rain regions such as under streetlamps (Fig. 10) and in close-up rain (Fig. 5), because they learn pixel-level mappings without a generative prior to fill in missing content. WeatherDiff, the 64×64 diffusion model, behaves differently in that it introduces severe patchy artifacts from independently denoising small tiles. Our baseline removes rain almost completely in this scene, including chromatic streaks and rain-induced fog, without suppressing scene detail or introducing artifacts. More visualizations are in Fig. 10 in the appendix.

Our baseline also achieves the best temporal consistency. In Tab. 4, our baseline achieves the lowest AFD and highest VLM temporal ratings on average. A key aspect is whether a method processes multiple frames jointly. Single-image methods and small-patch diffusion models score worse because each frame or patch is processed independently, producing visible flickering. Our baseline, through bi-directional self-attention, processes information jointly, achieving both the best deraining quality and the smoothest temporal output, while also being capable of deraining on videos with large camera motion and moving subjects (see video comparisons in the Supplementary Material).

6 Conclusion

We presented UENR-600K, a large-scale physically real nighttime deraining dataset rendered in Unreal Engine. Training on our data generalizes much better to real-world nighttime deraining, with our introduced new baseline achieving the best performance. **Limitations:** (1) We did not tune hyperparameters for our baseline; systematic tuning may further improve performance. (2) Our baseline is not optimized for speed, as it keeps no KV cache, decodes latents frame by frame, and runs in full precision.

AI USE STATEMENT

In this work, we used generative AI tools for implementing methods, assist with translation, cleaning and reformatting dataset, supporting qualitative and thematic data analysis, interpreting results. We have not used generative AI tools for generating synthetic datasets, helping develop theoretical models or conceptual frameworks, designing or provide feedback on research methodology or experiments, and formulating mathematical claims, providing critical ingredients for proving mathematical claims, assisting in the writing of proofs, proposing or refining hypotheses are not applicable to this work. Additionally, we used generative AI tools for VLM evaluation. We have reviewed all AI-assisted work. For example, LLM-generated code was verified and tested for correctness. We take responsibility for the final content of this work, including text, claims or artifacts produced with the aid of generative AI.

ETHICS STATEMENT

This work introduces a synthetic dataset rendered entirely in Unreal Engine, so it contains no human subjects and no personally identifiable information. The real-world evaluation set is drawn from Pexels (Pexels, 2026) under a license that permits research use, and we use it for evaluation only.

REPRODUCIBILITY STATEMENT

Appendix A documents every data processing step, including the train and test splits for all datasets. Appendix B lists the full training configuration for each method, together with the sliding window setup and the iteration matching procedure used to equalize gradient updates across datasets. Appendices C and D give the complete VLM evaluation prompts and batching procedures. We will release the dataset, the trained checkpoints, and all training and evaluation code upon acceptance.

REFERENCES

Anthropic. The Claude model card. `https://docs.anthropic.com/en/docs/about-claude/models`, 2025.

Yunhao Ba, Howard Zhang, Ethan Yang, Akira Suzuki, Arnold Pfahnl, Chethan Chinder Chandrappa, Celso de Melo, Suya You, Stefano Soatto, Alex Wong, and Achuta Kadambi. Not just streaks: Towards ground truth for single image deraining. In *Proceedings of the European Conference on Computer Vision (ECCV)*, 2022.

Andreas Blattmann, Tim Dockhorn, Sumith Kulal, Daniel Mendelevitch, Maciej Kilian, Dominik Lorenz, Yam Levi, Zion English, Vikram Voleti, Adam Letts, Varun Jampani, and Robin Rombach. Stable video diffusion: Scaling latent video diffusion models to large datasets. *arXiv preprint arXiv:2311.15127*, 2023.

Tim Brooks, Bill Peebles, Connor Holmes, Will DePue, Yufei Guo, Li Jing, David Schnurr, Joe Taylor, Troy Luhman, Eric Luhman, Clarence Ng, Ricky Wang, and Aditya Ramesh. Video generation models as world simulators. Technical Report, 2024.

Hongming Chen, Xiang Chen, Chen Wu, Zhuoran Zheng, Jinshan Pan, and Xianping Fu. Towards ultra-high-definition image deraining: A benchmark and an efficient method. *IEEE Transactions on Multimedia*, 2026.

Jie Chen, Cheen-Hau Tan, Junhui Hou, Lap-Pui Chau, and He Li. Robust video content alignment and compensation for rain removal in a cnn framework. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2018.

Xiang Chen, Jinshan Pan, Jiangxin Dong, and Jinhui Tang. Towards unified deep image deraining: A survey and a new benchmark. *IEEE Transactions on Pattern Analysis and Machine Intelligence (TPAMI)*, 2025.

Patrick Esser, Sumith Kulal, Andreas Blattmann, Rahim Entezari, Jonas Müller, Harry Saini, Yam Levi, Dominik Lorenz, Axel Sauer, Frederic Boesel, Dustin Podell, Tim Dockhorn, Zion English, Kyle Lacey, Alex Goodwin, Yannik Marek, and Robin Rombach. Scaling rectified flow transformers for high-resolution image synthesis, 2024. URL https://arxiv.org/abs/2403.03206.

Xueyang Fu, Jiabin Huang, Delu Zeng, Yue Huang, Xinghao Ding, and John Paisley. Removing rain from single images via a deep detail network. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2017.

Kshitiz Garg and Shree K. Nayar. Photorealistic rendering of rain streaks. *ACM Transactions on Graphics (Proceedings of SIGGRAPH)*, 25(3):996–1002, 2006.

Kshitiz Garg and Shree K. Nayar. Vision and rain. *International Journal of Computer Vision*, 75(1): 3–27, 2007.

Amirhosein Ghasemabadi, Muhammad Kamran Janjua, Mohammad Salameh, and Di Niu. Learning truncated causal history model for video restoration. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2024.

Yun Guo, Xueyao Xiao, Yi Chang, Shumin Deng, and Luxin Yan. From sky to the ground: A large-scale benchmark and simple baseline towards real rain removal. In *Proceedings of the IEEE International Conference on Computer Vision (ICCV)*, 2023.

Xiaowei Hu, Chi-Wing Fu, Lei Zhu, and Pheng-Ann Heng. Depth-attentional features for single-image rain removal. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2019.

Kui Jiang, Zhongyuan Wang, Peng Yi, Chen Chen, Baojin Huang, Yimin Luo, Jiayi Ma, and Junjun Jiang. Multi-scale progressive fusion network for single image deraining. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2020.

Ruoteng Li, Loong-Fah Cheong, and Robby T. Tan. Heavy rain image restoration: Integrating physics model and conditional adversarial learning. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2019.

Wei Li, Qiming Zhang, Jing Zhang, Zhen Huang, Xinmei Tian, and Dacheng Tao. Toward real-world single image deraining: A new benchmark and beyond. *arXiv preprint arXiv:2206.05514*, 2022.

Beibei Lin, Yeying Jin, Wending Yan, Wei Ye, Yuan Yuan, Shunli Zhang, and Robby T. Tan. Nighttrain: Nighttime video deraining via adaptive-rain-removal and adaptive-correction. In *Proceedings of the AAAI Conference on Artificial Intelligence (AAAI)*, 2024a.

Chih-Hao Lin, Zian Wang, Ruofan Liang, Yuxuan Zhang, Sanja Fidler, Shenlong Wang, and Zan Gojcic. Controllable weather synthesis and removal with video diffusion models. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, 2025.

Xinqi Lin, Jingwen He, Ziyan Chen, Zhaoyang Lyu, Bo Dai, Fanghua Yu, Yu Qiao, Wanli Ouyang, and Chao Dong. Diffbir: Towards blind image restoration with generative diffusion prior. In *Proceedings of the European Conference on Computer Vision (ECCV)*, 2024b.

Jiaying Liu, Wenhan Yang, Shuai Yang, and Zongming Guo. Erase or fill? deep joint recurrent rain removal and reconstruction in videos. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2018.

Yang Liu, Ziyu Yue, Jinshan Pan, and Zhixun Su. Unpaired learning for deep image deraining with rain direction regularizer. In *Proceedings of the IEEE International Conference on Computer Vision (ICCV)*, 2021.

Yuetong Liu, Yunqiu Xu, Yang Wei, Xiuli Bi, and Bin Xiao. Clear nights ahead: Towards multi-weather nighttime image restoration. In *Proceedings of the AAAI Conference on Artificial Intelligence (AAAI)*, 2026.

Ozan Özdenizci and Robert Legenstein. Restoring vision in adverse weather conditions with patch-based denoising diffusion models. *IEEE Transactions on Pattern Analysis and Machine Intelligence (TPAMI)*, 2023.

Prashant W. Patil, Sunil Gupta, Santu Rana, and Svetha Venkatesh. Video restoration framework and its meta-adaptations to data-poor conditions. In *Proceedings of the European Conference on Computer Vision (ECCV)*, 2022.

Pexels. Pexels: Free stock videos (url hidden in accordance with iclr policy), 2026.

Ruijie Quan, Xin Yu, Yuanzhi Liang, and Yi Yang. Removing raindrops and rain streaks in one go. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2021.

Team Wan. Wan: Open and advanced large-scale video generative models. *arXiv preprint arXiv:2503.20314*, 2025.

Shuai Wang, Lei Zhu, Huazhu Fu, Jing Qin, Carola-Bibiane Schönlieb, Wei Feng, and Song Wang. Rethinking video rain streak removal: A new synthesis model and a deraining network with video rain prior. In *Proceedings of the European Conference on Computer Vision (ECCV)*, 2022.

Tianyu Wang, Xin Yang, Ke Xu, Shaozhe Chen, Qiang Zhang, and Rynson W. H. Lau. Spatial attentive single-image deraining with a high quality real rain dataset. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2019.

Xinwei Xue, Jia He, Long Ma, Xiangyu Meng, Wenlin Li, and Risheng Liu. Asf-net: Robust video deraining via temporal alignment and online adaptive learning. *Pattern Recognition*, 2025.

Wending Yan, Robby T. Tan, Wenhan Yang, and Dengxin Dai. Self-aligned video deraining with transmission-depth consistency. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2021.

Wenhan Yang, Robby T. Tan, Jiashi Feng, Jiaying Liu, Zongming Guo, and Shuicheng Yan. Deep joint rain detection and removal from a single image. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2017.

Wenhan Yang, Robby T. Tan, Shiqi Wang, and Jiaying Liu. Self-learning video rain streak removal: When cyclic consistency meets temporal correspondence. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2020.

Zhuoyi Yang, Jiayan Teng, Wendi Zheng, Ming Ding, Shiyu Huang, Jiazheng Xu, Yuanming Yang, Wenyi Hong, Xiaohan Zhang, Guanyu Feng, Da Yin, Yuxuan Zhang, Weihan Wang, Yean Cheng, Bin Xu, Xiaotao Gu, Yuxiao Dong, and Jie Tang. Cogvideox: Text-to-video diffusion models with an expert transformer, 2025. URL https://arxiv.org/abs/2408.06072.

Zongsheng Yue, Jianwen Xie, Qian Zhao, and Deyu Meng. Semi-supervised video deraining with dynamical rain generator. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2021.

Fan Zhang, Shaodi You, Yu Li, and Ying Fu. Gtav-nightrain: Photometric realistic large-scale dataset for night-time rain streak removal. *arXiv preprint arXiv:2210.04708*, 2022a.

Fan Zhang, Shaodi You, Yu Li, and Ying Fu. Learning rain location prior for nighttime deraining. In *Proceedings of the IEEE International Conference on Computer Vision (ICCV)*, 2023.

He Zhang and Vishal M. Patel. Density-aware single image de-raining using a multi-stream dense network. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2018.

He Zhang, Vishwanath Sindagi, and Vishal M. Patel. Image de-raining using a conditional generative adversarial network. *IEEE Transactions on Circuits and Systems for Video Technology*, 30(11): 3943–3956, 2020.

Kaihao Zhang, Dongxu Li, Wenhan Luo, Wenqi Ren, and Wei Liu. Enhanced spatio-temporal interaction learning for video deraining: Faster and better. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 45(1):1287–1293, 2022b.

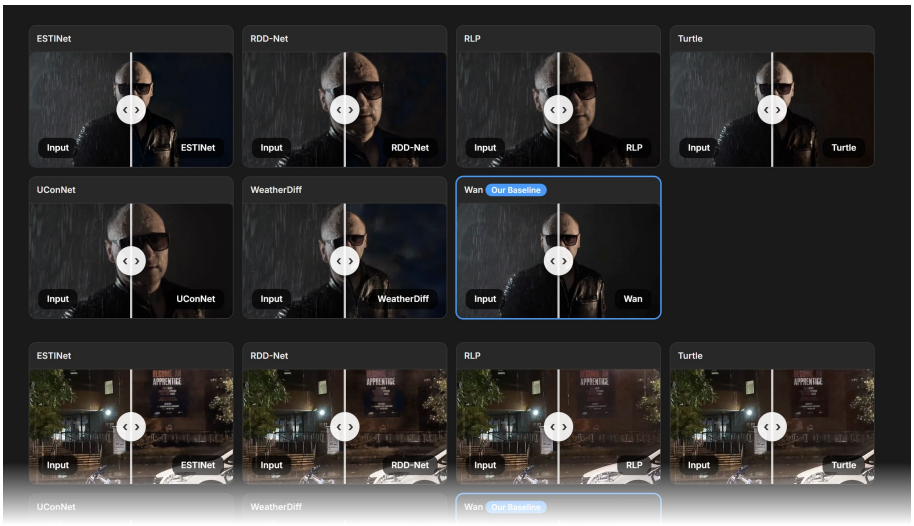
Richard Zhang, Phillip Isola, Alexei A. Efros, Eli Shechtman, and Oliver Wang. The unreasonable effectiveness of deep features as a perceptual metric. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2018.

Shangchen Zhou, Peiqing Yang, Jianyi Wang, Yihang Luo, and Chen Change Loy. Upscale-a-video: Temporal-consistent diffusion model for real-world video super-resolution. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2024.

Junhao Zhuang, Yisi Luo, Xile Zhao, Taixiang Jiang, and Bichuan Guo. Uconnet: Unsupervised controllable network for image and video deraining. In *Proceedings of the 30th ACM international conference on multimedia*, pp. 5436–5445, 2022.

Videos available in Supplementary Material

Please visit `project_page/index.html`



A Data Processing Details

Our dataset. We preprocess the frames in UENR-600K to 1280×720 resolution. These frames form a continuous camera trajectory in Unreal Engine 5, and adjacent frames maintain temporal coherence and thus can be grouped for video models. We designate frames 0 through 1,799 (1,800 frames) as the test set and frames 1,800 through 500,000 (498,200 frames) as the training set. For testing, we segment the 1,800 test frames into 20 non-overlapping clips of 90 frames each.

SynNightRain. SynNightRain (Patil et al., 2022) contains 68 videos across two subsets: NightRain (34 videos) and NightRainVeiling (34 videos). Each video contains 200 frames with paired rainy inputs and ground-truth clean frames. The videos have mixed native resolutions (3840×2160, 1920×1080, 1280×720, and 1080×1920 portrait). We follow the original train/test split: videos 002, 003, 005, and 007 from both subsets (8 videos, 1,600 frames) form the test set, and the remaining 60 videos (11,915 frames) form the training set. All frames are resized to 1280×720 using bicubic interpolation.

Real-world evaluation set. We collect 124 real-world nighttime rain videos from Pexels (Pexels, 2026) for out-of-domain evaluation. Each video is resized to 1280×720 and trimmed to 90 frames, yielding 11,160 test frames. These videos have no ground-truth clean counterparts and are used exclusively for VLM preference evaluation.

B Training Details

All existing methods are trained from scratch on each dataset following their original training configurations. To ensure a fair comparison, we match the total number of gradient updates to each method's original setting by adjusting the number of epochs according to our dataset size. Tab. 6 summarizes the key hyperparameters for each method.

Our baseline. We finetune `Wan 2.2-TI2V-5B` (Team Wan, 2025) using rank-96 LoRA for 1,400 steps on four NVIDIA H200 GPUs. Each training sample is a 1280×720 video of 90 frames, with a batch size of 1 per GPU. We use the AdamW optimizer with a learning rate of 1e-4, a 50-step linear warmup, weight decay of 1e-2, and betas of 0.9 and 0.95. This finetuning completes in

Table 5: Dataset splits used in our experiments. All frames are at 1280×720. UENR-600K and SynNightRain provide paired rainy/clean frames for training and synthetic evaluation. The Pexels set has no ground truth and is used exclusively for VLM-based evaluation.

	UENR-600K (Ours)	**SynNightRain (Patil et al., 2022)**	**Pexels (Pexels, 2026)**
Rain type	Synth.	Synth.	Real
Total frames	600,000†	13,515	11,160
Train frames	498,200	11,915	–
Test frames	1,800 (20 clips)	1,600 (8 clips)	11,160 (124 clips)
Ground truth	✓	✓	–

† 600,000 frames across two subsets (City Sample: 500,000; Kyoto: 100,000). Only the City Sample subset is used in this benchmark.

Table 6: Training hyperparameters for each method. All existing methods are trained from scratch; our baseline finetunes a pretrained video generation model using LoRA. Input type indicates whether the method processes single images (Img) or video clips of N consecutive frames (Vid-N). Patch size is the spatial crop used during training. Total iterations (Iters) are matched to each method's original setting (see Tab. 8).

	Architecture	**Input**	**Patch**	**Batch**	**Iters**	**Optimizer**	**LR**	**Loss**
ESTINet (Zhang et al., 2022b)	ResNet18 + R-CLSTM + BiLSTM	Vid-5	224	5	79.7K	Adam	1e-4	MSE
RDD-Net (Wang et al., 2022)	RDD-Net	Vid-7	128	8	320K	Adam	1e-4	L1
RLP (Zhang et al., 2023)	Uformer-T + RLP/RPIM	Img	256	20	74.7K	AdamW	2e-4	Charbonnier
Turtle (Ghasemabadi et al., 2024)	Transformer	Vid-5	128	1	200K	Adam	4e-4	L1
UConNet (Zhuang et al., 2022)	UConNet + AngleNet	Img	128	16	375K	Adam	1e-3	MSE
WeatherDiff (Özdenizci & Legenstein, 2023)	DDPM UNet (83M)	Img	64	80†	697K	Adam	2e-5	ϵ-pred
Our Baseline	`Wan2.2-TI2V-5B` (LoRA r=96)	Vid-90	Full	4	1.4K	AdamW	1e-4	Flow matching

† Effective samples per step (batch size × patches per sample). WeatherDiff: $5 \times 16 = 80$.

Table 7: Sliding window configuration for inference at 1280×720 resolution. Since each method operates on fixed-size patches, the original frame must first be enlarged to fit an exact grid of patches. **Padded Size** is the resulting resolution after zero-padding the bottom and right edges of the 1280×720 frame to the nearest multiple of the **Patch Size**; for example, with 224×224 patches, 1280×720 is padded to 1344×896 (= 6×224 by 4×224). The padded frame is then partitioned into a non-overlapping grid of tiles. WeatherDiff uses overlapping patches with stride r=16 instead of non-overlapping tiling, following its original inference procedure. Our baseline processes full-resolution frames without patch decomposition.

	Patch Size	**Padded Size**	**Grid**	**Patches/Frame**
ESTINet	224×224	1344×896	6×4	24
RDD-Net	128×128	1280×768	10×6	60
RLP	256×256	1536×768	6×3	18
Turtle	128×128	1280×768	10×6	60
UConNet	128×128	1280×768	10×6	60
WeatherDiff	64×64 (r=16)	Overlapping	77×42	3,234
Our Baseline	Full frame	N/A	1×1	1

approximately 35 hours and consumes roughly one epoch of training data. Unlike existing methods, our baseline operates at full 1280×720 resolution without patch decomposition.

Sliding window configuration. Since existing methods operate on fixed-size patches rather than full 1280×720 frames, we apply a non-overlapping sliding window strategy at both training and inference time. Each frame is padded with zeros (at the bottom and right edges) to the nearest multiple of the patch size, then partitioned into non-overlapping tiles. After processing, the tiles are stitched back and cropped to the original 1280×720 resolution. For video models, consecutive frames are grouped into non-overlapping temporal windows matching each method's input length. WeatherDiff uses an overlapping grid with stride r=16 instead of non-overlapping patches, following its original inference procedure. Tab. 7 lists the patch sizes, padded resolutions, and number of patches per frame for each method.

Table 8: Iteration matching between original training settings and our setup. For each method, we compute the total gradient updates from the original paper (dataset size × epochs / batch size) and reproduce a comparable number on our dataset by adjusting the number of epochs. Our baseline is finetuned for only 1,400 steps, consuming approximately one epoch of training data. The original datasets are NTU-derain (Chen et al., 2018), RainMotion (Wang et al., 2022), GTAV-NightRain (Zhang et al., 2022a), Rain100 (Yang et al., 2017) and Rain800 (Zhang et al., 2020), and AllWeather (Özdenizci & Legenstein, 2023).

	Original Setting				Our Setting (UENR-600K)			
	Dataset	**Size**	**Epochs**	**Iters**	**Size/Epoch**	**Batch**	**Epochs**	**Iters**
ESTINet	NTU-derain	3,023	120	∼181K	99,640	5	4	∼79.7K
RDD-Net	RainMotion	∼5,000	500	∼2.5M	71,171	8	36	∼320K
RLP	GTAV-NightRain	5,000	250	∼312K	498,201	20	3	∼74.7K
Turtle	(iter-based)	–	200K iter	200K	99,640	1	(iter)	200K
UConNet	Rain100/800	40,000*	150	∼375K	40,000*	16	150	∼375K
WeatherDiff	AllWeather	1,771	1,775	∼197K	498,201	80†	7	∼697K
Our Baseline	–	–	–	–	498,201	4	∼1	1.4K

* UConNet subsamples 40,000 images per epoch from the full dataset.
† Effective samples per step (see Tab. 6). WeatherDiff iterations are matched by total patch count rather than gradient steps.

Table 9: Same-dataset benchmarking: each method is trained and tested on the same dataset. ΔPSNR = Our Data PSNR minus SynNightRain PSNR; positive values indicate that SynNightRain is easier to fit, while the negative value (bold) indicates the opposite. All six existing methods achieve higher PSNR on SynNightRain, confirming its simpler degradation patterns. Our baseline is the only method that scores higher on our dataset, because its pretrained generative prior aligns better with rendering-based rain.

	Our Data		SynNightRain		ΔPSNR
	PSNR↑	**SSIM↑**	**PSNR↑**	**SSIM↑**	
ESTINet	24.88	0.845	26.89	0.867	−2.01
RDD-Net	24.34	0.835	25.03	0.851	−0.69
RLP	24.34	0.833	25.23	0.835	−0.89
Turtle	19.35	0.712	24.40	0.838	−5.05
UConNet	18.82	0.713	21.85	0.778	−3.03
WeatherDiff	23.88	0.848	27.52	0.896	−3.64
Our Baseline	25.49	0.834	23.31	0.755	+2.18

Iteration matching. Tab. 8 shows how we match training iterations between the original settings and our setup. For each method, we compute the total gradient updates from the original paper (original dataset size × original epochs / original batch size) and reproduce a comparable number using our dataset by adjusting the number of epochs.

B.1 Analysis on Synthetic Data

Metrics and evaluation. On synthetic test sets (1,800 frames for our dataset, 1,600 for SynNightRain), ground-truth clean frames are available. We report PSNR (pixel-level fidelity, in dB) and SSIM (structural similarity) in RGB space between derained outputs and ground-truth frames.

Our dataset is more challenging for existing methods, but easier for finetuning our baseline. As shown in Tab. 9, existing methods achieve an average of 2.55 dB higher PSNR on SynNightRain than on our dataset, indicating SynNightRain's algorithmic rain presents a simpler degradation pattern to fit. In contrast, our baseline achieves a 2.18 dB higher PSNR on our dataset. This divergence occurs because our baseline builds upon a video generation model pretrained on large-scale natural video (Team Wan, 2025). With this strong generative prior, our baseline inherently understands real physical interactions and lighting, and adapts easily to our physically simulated rain. Conversely, SynNightRain's algorithm-generated rain overlay deviates from natural physics, making it harder

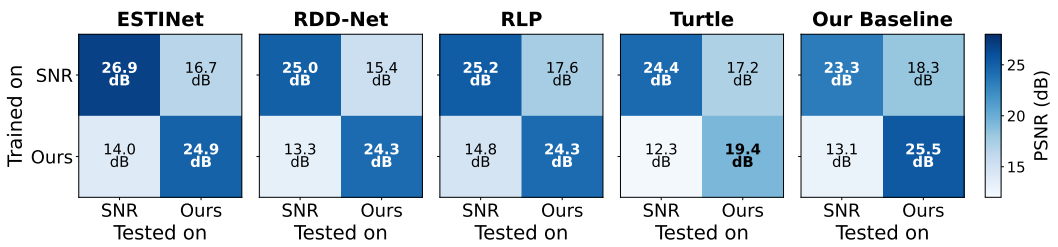


Figure 7: Domain gap between UENR-600K and SynNightRain, shown for five of the seven benchmarked methods. Each subplot reports the PSNR of one method trained on one dataset and tested on both datasets. Diagonal cells represent same-dataset performance, while off-diagonal cells show cross-dataset performance in PSNR. Every method shown experiences a substantial performance drop when tested on a different dataset than their training source, highlighting the significant domain gap between our physically grounded data and the algorithmically synthesized SynNightRain.

for the generative model to adapt. This performance difference validates the superior realism of our proposed data.

Cross-dataset evaluation shows a large domain gap, highlighting our data's novelty. To measure the domain gap between datasets, we train the models on one dataset and evaluate them on both. In Fig. 7, the off-diagonal cells display significantly lower PSNR values (5 to 12 dB drop) when testing across datasets. This mutual failure confirms a large domain gap: models optimized for the 2D overlays of SynNightRain cannot handle our physically grounded 3D rain, and vice versa. This contrast establishes the novelty of UENR-600K as a fundamentally different degradation domain that existing synthetic nighttime rain data does not capture.

Model rankings are consistent across evaluations. The synthetic data results (PSNR/SSIM) from Tab. 9 show the same tiers as the VLM preference results: our baseline ranks first in both, followed by ESTINet and RDD-Net as the strongest existing methods, and then the rest. This agreement between pixel-level metrics and VLM-based visual preference judgments indicates that our benchmark produces stable and reliable rankings.

C Deraining Quality Evaluation Using VLM as a Judge

We use `claude-sonnet-4-6` (Anthropic, 2025) as the VLM judge for deraining quality evaluation on the 124 real nighttime rain videos from Pexels (Pexels, 2026) (90 frames each). The VLM receives sampled frames from competing methods and selects the best result. To ensure reproducibility, we provide the complete evaluation prompt below.

Evaluation prompt. For each comparison, the VLM receives the rainy input image and N candidate derained images labeled with randomized anonymous letters (A, B, ...). The prompt instructs:

```
You are conducting a blind evaluation of image deraining quality. For
    ↪ each comparison, you will view a rainy input image and N
    ↪ candidate derained images labeled with letters. Your job is to
    ↪ pick the candidate with the best deraining quality.

Compare candidates on the following criteria:
(1) Rain removal: how effectively are rain streaks, rain drops, rain
    ↪ curtains, and rain-induced fog removed?
(2) Detail preservation: are scene structures, textures, edges, and fine
    ↪  details maintained?
(3) Artifact avoidance: does the result avoid introducing blocking
    ↪ artifacts, color distortions, excessive blurring, or visual
    ↪ hallucinations?
(4) Overall quality: considering all factors, which result looks most
    ↪ natural and visually clean?
```



Figure 8: Example of a multi-way VLM evaluation item. The VLM receives the rainy input (top-left) and seven candidate derained images labeled A through G. Method-to-label assignments are randomized per evaluation item, so the VLM cannot learn positional patterns. For this example, the VLM selected Candidate F with the justification: *"Candidate F achieves thorough rain streak removal while maintaining natural nighttime scene appearance without haze or color shift artifacts."*

```
For each comparison, view the input image first for context, then view
    ↪ all candidates, choose the best one based on the criteria above,
    ↪ and provide a one-sentence justification.
```

The VLM must provide a written justification before making its selection to encourage careful analysis. Method labels are randomized per evaluation item using a fixed seed to mitigate position bias.

Multi-way comparison. The VLM views results from all seven methods for the same frame and selects the single best output. We evaluate one frame per video across all 124 videos, yielding 124 comparisons per experiment (covering 1.1% of the 11,160 total test frames). Each comparison presents 8 images to the VLM: 1 rainy input and 7 candidate outputs.

Pairwise comparison. For comparing models trained on different datasets (e.g., our dataset versus SynNightRain), the VLM views two results side by side for three frames per video across all 124 videos, yielding 372 comparisons per experiment (covering 3.3% of test frames). A tie option is available when the two results are visually indistinguishable. For pairwise evaluations, the preference rate is computed as $\text{wins}/(\text{wins} + \text{losses})$, excluding ties from the denominator.

Parallelization and batching. Evaluations are distributed across multiple independent VLM instances (up to 4 concurrent), each processing a disjoint batch of evaluation items. Each instance is self-contained and receives the full prompt with its assigned images. This parallelization does not affect evaluation quality since each item is evaluated independently.

D Temporal Consistency Evaluation Using VLM as a Judge

We use `claude-opus-4-6` (Anthropic, 2025) as the VLM judge for temporal consistency evaluation, rating each method's output independently on a 1 to 5 scale. Unlike the pairwise deraining evaluation above, which compares methods against each other, the temporal evaluation rates each (model, video clip) pair independently to avoid conflating spatial quality with temporal smoothness.

Frame selection and difference maps. For each video clip, we select 4 consecutive frames from the temporal midpoint (e.g., frames 44 to 47 for 90-frame clips). In addition to raw frames, we pre-compute 3 inter-frame difference maps by taking the per-pixel absolute difference between consecutive output frames, converting to grayscale, and amplifying by a factor of 10. In a temporally consistent result, only regions with real scene motion appear bright in the difference maps; flickering artifacts appear as bright patches in regions that should be static (e.g., building facades, sky, parked cars).

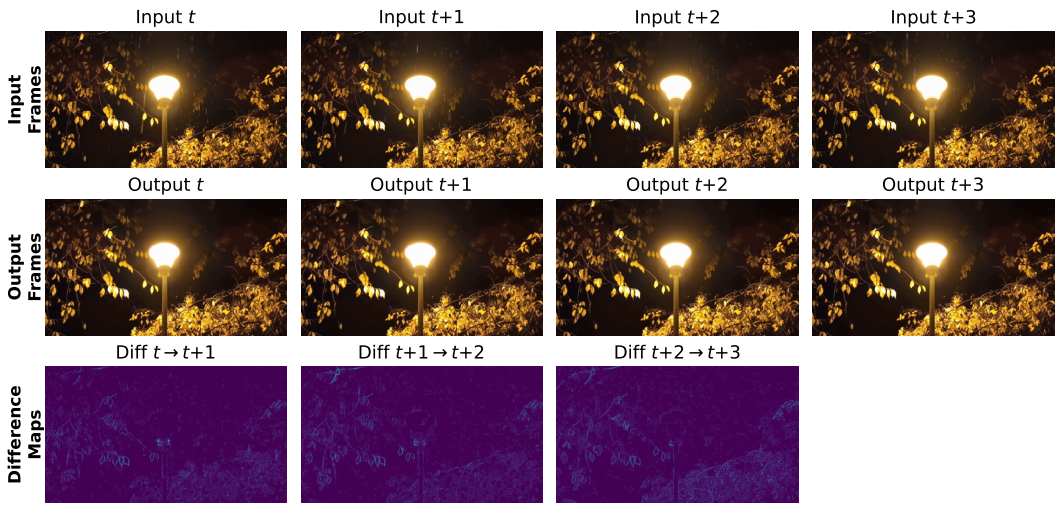


Figure 9: Example of a temporal consistency VLM evaluation item. **Top:** four consecutive rainy input frames providing scene context. **Middle:** four consecutive derained output frames to evaluate. **Bottom:** three inter-frame difference maps (amplified 10×), where bright regions indicate large pixel changes between adjacent output frames. In a temporally consistent result, only regions with real scene motion (e.g., swaying leaves) appear bright, while static regions (e.g., lamp post, sky) remain dark. Flickering artifacts would manifest as bright patches in static areas.

Evaluation prompt. Each evaluation item presents the VLM with 11 images: 4 rainy input frames (for scene context), 4 derained output frames (to evaluate), and 3 inter-frame difference maps. The prompt instructs:

```
You are evaluating the temporal consistency of video deraining results.
    ↪ Rate the following video sequence on a 1 to 5 scale:

5 (Excellent): No visible temporal artifacts. Smooth, natural frame-to-
    ↪ frame transitions.
4 (Good): Very minor temporal inconsistencies, barely noticeable.
3 (Fair): Noticeable temporal artifacts. Some flickering, inconsistent
    ↪ deraining across frames, or minor ghosting.
2 (Poor): Significant temporal artifacts. Obvious flickering, prominent
    ↪ frame-to-frame inconsistencies.
1 (Very Poor): Severe temporal artifacts. Extreme flickering, frames
    ↪ appear unrelated.

What to look for: flickering (unexpected brightness or color changes in
    ↪ static regions), inconsistent rain removal across consecutive
    ↪ frames, ghosting or bleeding artifacts, and whether real scene
    ↪ motion appears smooth and continuous. Use the difference maps to
    ↪ detect subtle flickering: bright patches in regions that should
    ↪ be static indicate temporal artifacts.

Read all images (input frames, output frames, difference maps), then
    ↪ rate 1 to 5 and give a one-sentence reason.
```

Batching and scale. Items are grouped into batches of 4 (44 images per batch) and shuffled with a fixed seed so that consecutive batches contain a mix of different models. Up to 4 VLM instances process batches in parallel. We evaluate all 124 clips per model across both training datasets (7 methods × 2 datasets = 14 configurations), yielding 1,736 total evaluations. Per-model scores are aggregated as the mean rating across all evaluated clips.

Table 10: API cost for reproducing all VLM evaluations using `claude-opus-4-6` (Anthropic, 2025). Each 1280×720 image consumes 1,229 input tokens; the 4,588 evaluations use 36.5M input tokens and 0.6M output tokens in total. The temporal evaluation is the most expensive because each item requires 11 images (4 input frames, 4 output frames, and 3 difference maps).

	Input (MTok)	Output (MTok)	Cost ($)
Pairwise quality (Tab. 2)	10.12	0.39	60.35
Multi-way quality (Tab. 3)	2.52	0.04	13.60
Temporal consistency (Tab. 4)	23.90	0.18	124.00
Total	**36.54**	**0.61**	**197.95**

Table 11: Inference time on the Ours test set (1,800 frames at 1280×720) using a single GPU. Non-diffusion methods complete within minutes to hours, while the diffusion-based WeatherDiff requires significantly longer due to iterative denoising over many patches. Our baseline processes full-resolution frames without patch decomposition.

	s/frame	Total Time
ESTINet (Zhang et al., 2022b)	4.34	130.2 min
RDD-Net (Wang et al., 2022)	2.79	83.7 min
RLP (Zhang et al., 2023)	1.09	32.7 min
Turtle (Ghasemabadi et al., 2024)	6.18	185.4 min
UConNet (Zhuang et al., 2022)	0.81	24.3 min
WeatherDiff (Özdenizci & Legenstein, 2023)	11.00	5.50 h
Our Baseline	24.61	12.31 h[†]

[†] Measured on NVIDIA H200; all others on A5000.
Scaled by FP32 throughput (2.4×), est. ∼29.5 h on A5000.

Evaluation cost. Tab. 10 reports the API cost for reproducing all VLM evaluations in this paper. Each 1280×720 image consumes 1,229 input tokens, and the 4,588 evaluations use 36.5M input tokens and 0.6M output tokens in total. The temporal consistency evaluation accounts for 63% of the total cost because each evaluation item presents 11 images to the VLM.

E Inference Time

Tab. 11 reports the per-frame and total inference time for each method on the Ours test set (1,800 frames at 1280×720). All existing methods are measured on a single NVIDIA A5000 GPU. Non-diffusion methods complete within 24 minutes (UConNet) to 185 minutes (Turtle). The diffusion-based WeatherDiff requires significantly longer due to iterative denoising over many patches, taking 5.50 hours. Our baseline processes full 1280×720 frames without patch decomposition, completing 1,800 frames in 12.31 hours on a single NVIDIA H200 GPU; based on FP32 throughput scaling, the estimated A5000-equivalent time is approximately 29.5 hours.

F Comparing with Weather Restylization Datasets

An adjacent problem to **weather restoration** is **weather restylization**. **Weather restoration** recovers the clean scene that lies behind a weather degradation, and leaves the rest of the frame unchanged. **Weather restylization** instead converts a scene from one weather condition into another, for example replacing a rainy scene with a sunny one, which also changes the illumination, the sky, and the standing water on the ground. It might be interesting to investigate whether weather restylization datasets could also be used to train for weather restoration tasks like video deraining.

F.1 Experimental Setup

Dataset Weather restylization datasets include WeatherWeaver (Lin et al., 2025), which releases 20,800 synthetic video pairs of 100 frames each. Every pair carries six weather effects with continuous strengths: cloud, fog, rain, snowfall, puddle, and persistent snow cover. The clean reference of

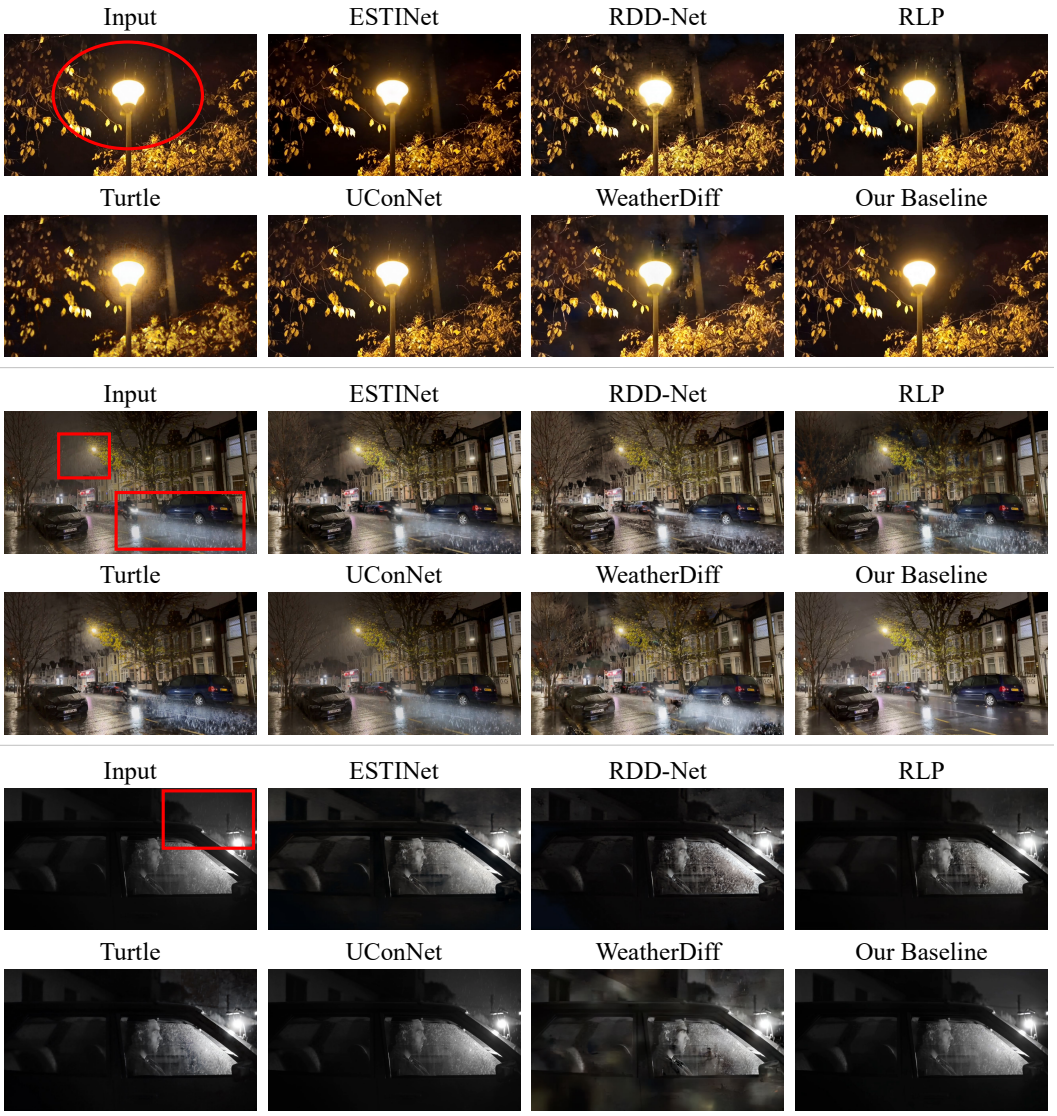


Figure 10: Qualitative comparison of all methods trained on UENR-600K (our data), on three further **real** nighttime rain scenes. Red regions are highlighted for comparison. Our baseline removes the rain cleanly, while the others do not.

a clip sets all six strengths to zero, so reaching it also clears the sky and dries the ground whenever the degraded frame carries cloud or puddle. The pairs come from 5,599 scene clips, of which 5,182 carry rain without snow and are therefore usable for deraining. All released scenes are captured in daylight, and the authors report that "Our model also struggles with nighttime videos, in part due to the scarcity of such footage in our current data-curation pipeline." We select WeatherWeaver for this comparison because it releases paired clean references at a scale that matches our own dataset.

Data Preparation We pair the strongest rain variant of each clip with its clean reference, center-crop each frame to 1920×1080, and resize it to 1280×720 with bicubic interpolation. WeatherWeaver provides no official split, so we hold out 20 clips with a fixed seed and take 90 frames from each, forming a 1,800 frame test set. The remaining 4,982 clips give 498,200 training frames. Both counts match the UENR-600K split exactly, so the two datasets drive identical training schedules.

Table 12: Pairwise comparison of deraining results on real nighttime rain videos, where a judge selects the better of two results. For each method, the version trained on our dataset competes against the version trained on WeatherWeaver (Lin et al., 2025), and bold marks the preferred training dataset. **Input Size** is the spatial resolution each method processes during training and inference. Every judge prefers our dataset on average.

		VLM (Claude)			VLM (Qwen)			Human		
	Input Size	**WeatherWeaver**	**Tie**	**Our Data**	**WeatherWeaver**	**Tie**	**Our Data**	**WeatherWeaver**	**Tie**	**Our Data**
ESTINet	224×224	**64.5%**	0.0%	35.5%	23.9%	0.8%	**75.3%**	33.3%	23.8%	**42.9%**
RDD-Net	128×128	**54.8%**	0.0%	45.2%	49.2%	0.0%	**50.8%**	**42.9%**	19.0%	38.1%
RLP	256×256	46.0%	0.3%	**53.8%**	26.1%	1.3%	**72.6%**	28.6%	19.0%	**52.4%**
Turtle	128×128	43.8%	0.3%	**55.9%**	35.5%	0.0%	**64.5%**	33.3%	19.0%	**47.6%**
UConNet	128×128	**56.2%**	0.0%	43.8%	29.8%	0.0%	**70.2%**	33.3%	19.0%	**47.6%**
WeatherDiff	64×64	8.1%	0.0%	**91.9%**	12.1%	0.0%	**87.9%**	9.5%	19.0%	**71.4%**
Our Baseline	Full	13.7%	1.3%	**84.9%**	31.2%	0.0%	**68.8%**	19.0%	23.8%	**57.1%**
Average	–	41.0%	0.3%	**58.7%**	29.7%	0.3%	**70.0%**	28.6%	20.4%	**51.0%**

Table 13: Multi-way comparison of deraining results on real nighttime rain videos, where a judge selects the best result among all seven methods trained on the same dataset. **Pretrained**: model was pretrained on large-scale video data before finetuning. Our baseline takes almost every selection under either training dataset, and it takes more of them when trained on our dataset.

			VLM (Claude)			VLM (Qwen)			Human		
	Architecture	**Pretrained**	**WeatherWeaver**	**Our Data**	**Avg**	**WeatherWeaver**	**Our Data**	**Avg**	**WeatherWeaver**	**Our Data**	**Avg**
ESTINet	CNN+RNN		5.6%	5.6%	5.6%	1.6%	1.6%	1.6%	4.8%	4.8%	4.8%
RDD-Net	CNN		0.0%	0.0%	0.0%	0.0%	0.0%	0.0%	0.0%	0.0%	0.0%
RLP	Transformer		0.8%	0.0%	0.4%	2.4%	1.6%	2.0%	0.0%	0.0%	0.0%
Turtle	Transformer		0.0%	0.0%	0.0%	0.0%	0.0%	0.0%	0.0%	0.0%	0.0%
UConNet	CNN		0.0%	0.0%	0.0%	0.8%	0.0%	0.4%	0.0%	0.0%	0.0%
WeatherDiff	Diffusion		0.0%	0.0%	0.0%	0.0%	0.0%	0.0%	0.0%	0.0%	0.0%
Our Baseline	Diffusion	✓	**93.5%**	**94.4%**	**94.0%**	**95.2%**	**96.8%**	**96.0%**	**95.2%**	**95.2%**	**95.2%**

Training We retrain all seven methods on WeatherWeaver with the configurations in Tab. 6 and the sliding window setup in Tab. 7. RLP is the single exception, as its training was preempted after 2 of 3 cosine annealing epochs, and we report its second-epoch checkpoint.

Evaluation We evaluate every model with PSNR and SSIM on the synthetic test sets, and with a human or a VLM judge on the 124 real Pexels videos (Pexels, 2026). The judging protocol follows the main paper. A pairwise comparison asks the judge for the better of two results, and a multi-way comparison asks it for the best result among all eight methods trained on the same dataset.

F.2 Results and Discussion

On real videos, weather restoration data trains better deraining models than weather restylization data. Every evaluator in Tab. 12 prefers the version trained on UENR-600K, which takes 58.7%, 70.0%, and 51.0% of the comparisons on average. The same ordering holds when all seven methods compete at once, where our baseline takes more selections under our dataset than under WeatherWeaver (Tab. 13). Fig. 11 shows where the baseline trained on WeatherWeaver loses ground: it removes the rain only partially. Slanted streaks and horizontal splashes survive in its output, and so do the raindrops that fall through a light cone. Our dataset covers these cases, because it renders rain under many wind directions and intensities, and because it simulates the localized, chromatic rain that only appears around an artificial light source. Nighttime rain of exactly this kind is what the WeatherWeaver authors report as scarce in their own footage.

On restylization data, our baseline still performs the best. Tab. 13 also ranks the seven methods against each other when all of them are trained on WeatherWeaver. Our baseline takes 94.6% of the selections on average across the three judges, and the six existing methods share the remaining 5.3%. This gap again shows that the pretraining and finetuning paradigm, which finetunes a strong foundation model on high quality synthetic data, reaches better weather restoration performance than training a deraining network from scratch.

Models trained on our dataset generalize better to the other dataset. Training on UENR-600K and testing on WeatherWeaver gives 18.21 dB on average, only 1.87 dB below what training on

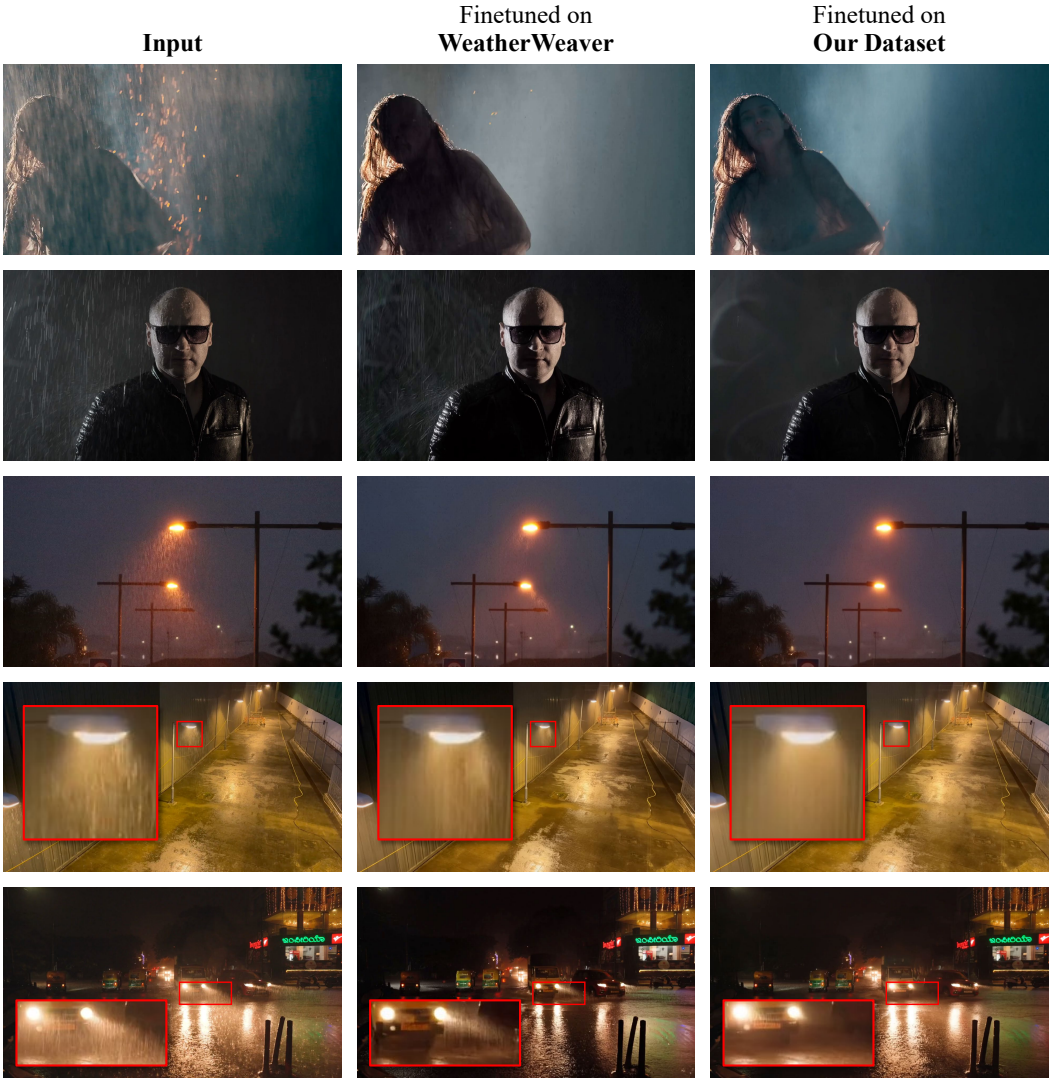


Figure 11: Our baseline finetuned on WeatherWeaver (Lin et al., 2025) and on our dataset, across five real nighttime rain scenes from Pexels (Pexels, 2026). The version finetuned on WeatherWeaver still leaves rain behind. Row 2 keeps the slanted streaks to the left of the subject and the horizontal splash off the subject's shoulder, and rows 3 to 5 keep the raindrops that fall through the streetlamp, the wall lamp, and the headlight beam. The version finetuned on our dataset clears the rain in all five scenes, from the colorful rain inside each light cone to heavy rain splashing in different directions.

WeatherWeaver itself reaches, while the reverse direction loses 5.29 dB (Tab. 14). Two of the seven methods, including our baseline, even score higher on the WeatherWeaver test set when they are trained on UENR-600K. This could be due to that weather restoration is more objective, in that it sets an objective target, because the clean frame differs from the degraded frame only where the weather was, and such a target is easier to learn. Weather restylization, instead, changes the weather together with the illumination and the shadows of the whole frame, which is a far more subjective target. Finetuning on it can leave a model uncertain about what it is supposed to change, for example, how many clouds should remain in the sky, or from which direction the sun should shine on the scene.



Figure 12: Comparison between weather restylization data (WeatherWeaver (Lin et al., 2025)) and weather restoration data (our UENR-600K). Weather restoration removes the rain and the rain-induced effects in an objective way, and it keeps the standing water on the ground, the clouds in the sky, and the illumination of the scene. A weather restylization dataset instead performs a more subjective edit, which clears the overcast sky, dries the road, and restores direct sunlight.

Table 14: Cross-dataset PSNR (dB). Each cell trains a method on the dataset named in the first header row and tests it on the dataset named in the second. Training on our dataset transfers to the WeatherWeaver test set almost as well as training on WeatherWeaver itself, while the reverse direction loses more. Bold marks the better training set within each test set.

	Train: WeatherWeaver		**Train: Our Data**	
	Test: Ours	Test: WW	Test: Ours	Test: WW
ESTINet	18.59	**21.72**	**24.88**	17.01
RDD-Net	18.43	**21.77**	**24.34**	17.11
RLP	18.71	**21.03**	**24.34**	17.92
Turtle	16.80	**19.50**	**19.35**	18.61
UConNet	15.86	18.97	**18.82**	**20.21**
WeatherDiff	17.46	**18.45**	**23.88**	17.07
Our Baseline	18.20	19.11	**25.49**	**19.52**
Average	**17.72**	**20.08**	**23.01**	**18.21**