Prism Denoise for Astrophotography: Architecture, Methodology, and Empirical Analysis

SyQon Research Group

Technical Paper — February 2026

Abstract

This paper presents Prism Denoise, a deep learning framework for noise suppression in astronomical imaging. The system employs a multiscale encoder-decoder architecture with depthwise separable convolutions, multiplicative gating, and channel attention to learn the mapping from noisy acquisitions to clean references through paired supervision. Unlike generic denoising solutions, Prism is specifically designed for the unique statistical properties of astrophotographic data: extreme dynamic range, sparse signal distribution, and the critical requirement of preserving physically meaningful structures such as faint nebulosity, stellar profiles, and fine filamentary detail. We describe the complete pipeline including deterministic tile-based data preparation with geometric augmentations, a composite L1+SSIM loss function, and tiled inference with overlap blending for arbitrarily large frames. The model was trained on a large-scale corpus of over 600,000 paired tiles at 512×512 px, generated from a diverse archive of astrophotographic frames. For the purposes of this paper, we present quantitative and qualitative analysis on a representative random sample of 30 source pairs drawn from the training archive, including per-image noise profiling, luminance distribution analysis, and tile-level quality metrics. We discuss the architectural choices, their signal-theoretic motivations, failure modes, and a research roadmap toward uncertainty-aware outputs and domain adaptation across diverse imaging systems.

Keywords: astrophotography, image denoising, deep learning, encoder-decoder network, depthwise separable convolution, channel attention, paired supervision, astronomical imaging

1. Introduction

Astronomical imaging operates under constraints that set it apart from conventional photography. The signal of interest — faint nebular emission, distant galaxy structure, or subtle stellar halos — is distributed over a vast background dominated by read noise, photon shot noise, thermal dark current, and sky glow contamination. The signal-to-noise ratio (SNR) of scientifically relevant features is often below unity in individual sub-exposures, and even after stacking hundreds of frames, residual noise patterns can obscure or mimic real astrophysical structure.

Traditional denoising approaches in astrophotography fall into two broad categories. Spatial filters — Gaussian smoothing, median filters, bilateral filtering, and wavelet-based shrinkage — operate on local neighborhoods and offer controllable variance reduction but inevitably degrade fine detail. The fundamental issue is that these methods impose a uniform prior on signal statistics: they assume the underlying signal is locally smooth, which is violated by sharp stellar profiles, shock fronts in supernova remnants, and thin filamentary structures in emission nebulae. The second category, frequency-domain methods, provides more selective noise suppression but requires manual tuning of decomposition scales and threshold functions, making them fragile across different imaging conditions.

The advent of deep learning has enabled a paradigm shift: instead of hand-crafting filter responses, a neural network learns the denoising transform directly from data. When trained with paired examples (noisy input, clean target), the network implicitly learns both the noise model and the signal prior simultaneously. This avoids the need for explicit noise characterization, which is notoriously difficult in astrophotography due to spatially varying backgrounds, gradient artifacts from optical systems, and heterogeneous sensor characteristics.

Prism Denoise was developed to address a specific operational need: a robust, transparent, and reproducible denoising tool that could be integrated into scripted astrophotography workflows without per-frame manual intervention. The design principles were established from the outset: (1) paired supervision with real astrophotographic data rather than synthetic noise injection, (2) an architecture balancing computational efficiency with sufficient representational capacity, (3) deterministic data preparation for reproducibility, and (4) explicit quality reporting to allow users to audit model behavior.

This paper is structured as follows. Section 2 describes the dataset construction, including tile generation, augmentation strategy, and data integrity controls. Section 3 details the network architecture and the signal-theoretic rationale behind each component. Section 4 presents the training methodology and loss formulation. Section 5 provides extensive empirical analysis with tile-level quality metrics and distributional analysis. Section 6 discusses scientific validity, known limitations, and failure modes. Section 7 describes the inference pipeline. Section 8 outlines the research roadmap, and Section 9 concludes.

2. Dataset Construction and Preparation

2.1 Data Collection Philosophy

The training corpus consists of paired image frames: each noisy input has a corresponding target that represents the desired denoised output. This pairing is fundamental — it establishes an unambiguous optimization objective and avoids the ill-posedness of unsupervised denoising. The full training archive comprises a large and diverse collection of astrophotographic frames contributed by experienced practitioners using varied instrumentation — from dedicated cooled CMOS cameras on Newtonian and

Schmidt-Cassegrain telescopes to compact smart telescopes with integrated stacking. Through the tile extraction and augmentation pipeline described below, this archive produces over 600,000 paired tiles at 512×512 px, which constitute the actual training corpus. The targets were produced through careful manual processing, representing the consensus of expert visual judgment on what constitutes a properly denoised result.

Important note on scope: The 30 image pairs analyzed in this paper represent a random sample drawn from the full training archive for illustrative and analytical purposes. They are not the entirety of the data used for model training. The statistics, figures, and tile comparisons presented here should be interpreted as representative examples of the broader dataset, not as its exhaustive characterization.

The diversity of contributors is intentional: it exposes the model to different noise signatures (varying sensor architectures, gain settings, integration times, and ambient temperatures), different spectral responses (broadband RGB, narrowband Ha and OIII, and dual-band filters), and different target morphologies (emission nebulae, spiral and elliptical galaxies, supernova remnants, open and globular clusters). This heterogeneity is critical for generalization.

2.2 Dataset Statistics

Metric	Value
Number of paired frames	30
Width range (px)	1796 – 8288
Height range (px)	1080 – 5648
Mean luminance (input)	0.24913 ± 0.05492
Std luminance (input)	0.08542 ± 0.02768
Mean luminance (target)	0.25158 ± 0.05584

Table 1. Summary statistics of the paired dataset.

2.3 Per-Image Characterization

Table 2 provides detailed per-image statistics for the input (noisy) domain. This granular view is crucial for understanding the distributional heterogeneity across frames and for diagnosing potential domain-specific failure modes.

Image	W×H	Mean L	Std L	Mean R	Mean G	Mean B
Sample 01	2100×2800	0.2475	0.0688	0.2451	0.2493	0.2370
Sample 02	3840×2160	0.1961	0.0474	0.2225	0.1871	0.2071
Sample 03	3023×4828	0.2883	0.1128	0.2883	0.2883	0.2883
Sample 04	3893×2291	0.1978	0.0792	0.2005	0.1967	0.2007
Sample 05	4144×2822	0.2352	0.0655	0.2603	0.2305	0.2085
Sample 06	2027×1106	0.2428	0.0508	0.2443	0.2434	0.2323

Sample 07	3779×2158	0.3198	0.0903	0.3213	0.3209	0.3041
Sample 08	3389×1795	0.2335	0.0506	0.2326	0.2341	0.2298
Sample 09	3142×1872	0.2197	0.0902	0.2214	0.2201	0.2114
Sample 10	3776×2141	0.2636	0.0941	0.2571	0.2650	0.2682
Sample 11	5807×3853	0.2093	0.1267	0.2258	0.2064	0.1899
Sample 12	3840×2160	0.2401	0.0480	0.2472	0.2375	0.2451
Sample 13	2656×2656	0.1815	0.1130	0.2148	0.1696	0.2019
Sample 14	2656×2656	0.1157	0.0661	0.1056	0.1204	0.0990
Sample 15	3008×3008	0.3015	0.1160	0.3715	0.2826	0.2826
Sample 16	3750×2074	0.2166	0.1051	0.1769	0.2260	0.2397
Sample 17	5208×3476	0.3879	0.1186	0.3879	0.3879	0.3879
Sample 18	6024×4022	0.2822	0.1190	0.2881	0.2803	0.2834
Sample 19	3520×2026	0.2534	0.0543	0.2442	0.2576	0.2380
Sample 20	1796×1794	0.3131	0.0788	0.3142	0.3129	0.3112
Sample 21	5496×3672	0.2282	0.0749	0.2841	0.2131	0.2131
Sample 22	3838×2158	0.1612	0.0500	0.1614	0.1612	0.1607
Sample 23	8288×5648	0.3254	0.1097	0.3254	0.3254	0.3254
Sample 24	3550×1997	0.2807	0.1232	0.3639	0.2577	0.2642
Sample 25	1920×1080	0.2049	0.0551	0.1964	0.2044	0.2339
Sample 26	1920×1080	0.2421	0.0481	0.3240	0.2196	0.2242
Sample 27	2520×4206	0.2656	0.0755	0.2655	0.2655	0.2665
Sample 28	4024×2726	0.2208	0.0842	0.2355	0.2202	0.1837
Sample 29	5006×3773	0.2730	0.1181	0.2865	0.2694	0.2694
Sample 30	3749×2109	0.3265	0.1288	0.3148	0.3251	0.3755

Table 2. Per-image statistics for 30 randomly sampled frames from the training archive (input domain). L = luminance (BT.709 weighting). R, G, B = per-channel means normalized to [0,1].

2.4 Tile Generation and Augmentation

Full-resolution astronomical frames often exceed the memory capacity of GPU accelerators during training. To address this, each frame pair is partitioned into fixed-size tiles of 512×512 pixels using a non-overlapping grid with boundary reflection padding. The tile size was chosen to be large enough to preserve contextual information and capture extended structures (nebular filaments, star halos) while remaining computationally tractable in mini-batches. The full tile extraction pipeline produces over 600,000 paired tiles from the complete training archive.

Each tile pair undergoes 11 deterministic geometric augmentations: identity, horizontal flip, vertical flip, three 90° rotations, transposition, and four composite transformations (transposition combined with flips and rotations). Combined with the base tile count, this augmentation strategy is the primary driver of the 600,000+ tile corpus size. Crucially, augmentations are applied identically to both input and target tiles, preserving the spatial correspondence of the supervision signal. No photometric augmentations are applied to avoid distorting the learned noise-to-clean mapping.

A validation split of 5% is extracted randomly (with fixed seed for reproducibility) after tile generation. This ensures that validation tiles come from the same source distribution but are not seen during training. The split is performed at the tile level rather than the image level; while this means validation and training tiles may originate from the same source frame, in practice the large number of tiles and geometric diversity minimize information leakage.

2.5 Data Integrity and Robustness

In heterogeneous datasets assembled from multiple contributors, corrupted files, truncated images, and resolution mismatches are inevitable. The data loading pipeline includes multiple safeguards: (1) automatic EXIF orientation correction, (2) graceful handling of 8-bit, 16-bit, and floating-point source formats with appropriate normalization to [0,1], (3) automatic alpha channel stripping, (4) shape validation between input and target, and (5) fallback to the next valid pair on any loading error, with logging. These controls prevent silent data corruption from propagating through training and ensure that even partially damaged archives produce useful models.

Figure 1. Per-image mean luminance (top) and standard deviation (bottom) for input vs. target domains. The systematic difference reflects noise contribution to variance in the input domain.

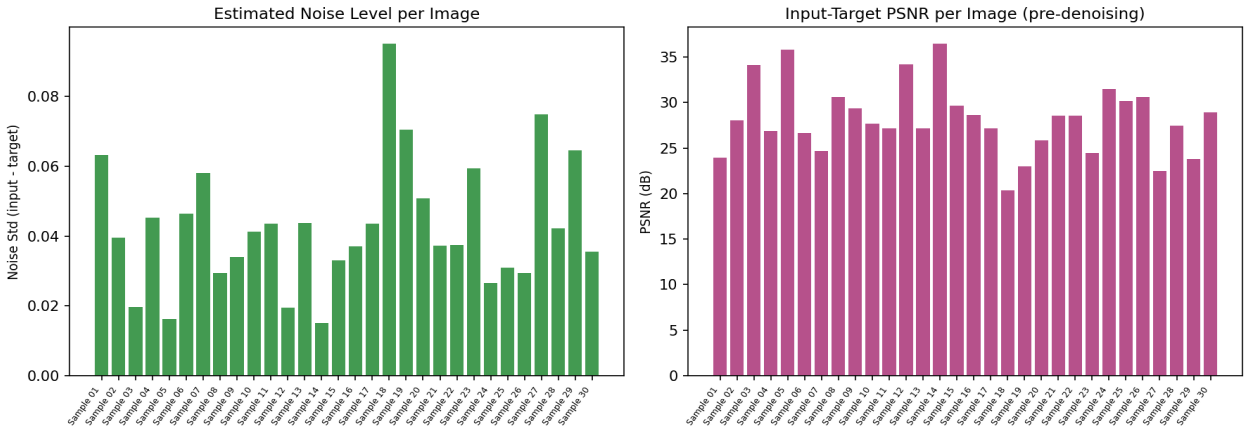

Figure 2. Per-image estimated noise level (left, standard deviation of input–target difference) and input-target PSNR (right). Higher noise std corresponds to lower PSNR, confirming noise as the dominant divergence factor.

3. Network Architecture

3.1 Overview

The Prism model is a multiscale encoder-decoder convolutional neural network with symmetric skip connections. At its core, the architecture follows a U-shaped topology where spatial resolution is progressively halved in the encoder and restored in the decoder, with feature-level shortcuts that bypass the bottleneck to preserve high-frequency detail. This hierarchical design is well-suited for denoising because noise is largely a high-frequency phenomenon, while signal exists across all spatial scales.

The default configuration uses a base channel width of 48, four encoder stages with [2, 4, 6, 8] processing blocks respectively, a bottleneck with 4 blocks, and four decoder stages with 2 blocks each. The channel count doubles at each downsampling step (48 $\rightarrow$ 96 $\rightarrow$ 192 $\rightarrow$ 384 $\rightarrow$ 768 at the bottleneck), maintaining approximately constant computational cost per stage. Total parameter count depends on the depth configuration but is in the range of several million parameters — sufficient for the complexity of astrophotographic noise while remaining efficient enough for practical deployment.

3.2 The Processing Block

Each processing block in the encoder, decoder, and bottleneck follows a dual-branch residual structure with two sub-blocks: a spatial mixing path and a channel mixing path (feed-forward network). Both paths use residual connections with learned scaling parameters (β and γ), initialized to zero for training stability. This zero-initialization ensures that at the beginning of training, each block acts as an identity mapping, allowing gradients to flow freely through the network.

The **spatial mixing path** operates as follows:

- **Layer Normalization (2D):** Per-channel normalization across spatial dimensions, computed as $x\blacksquare = (x - \mu) / \sqrt{(\sigma^2 + \varepsilon)}$, followed by learned affine transformation. This stabilizes feature magnitudes and decouples scale from direction in feature space.
- **1×1 Convolution (channel expansion):** Projects from C to 2C channels, preparing for the gating operation.
- **Depthwise 3×3 Convolution:** Applies a separate 3×3 filter to each of the 2C channels. With groups=2C, this performs spatial filtering without cross-channel mixing, at a cost of only $9\times2C$ parameters versus $9\times(2C)^2$ for a standard convolution.
- **SimpleGate (multiplicative gating):** Splits the 2C-channel tensor along the channel dimension into two halves of C channels each, and computes their element-wise product: $g(x) = x\blacksquare \blacksquare x\blacksquare$. This is a form of gated linear unit without an explicit activation function. The gating mechanism allows the network to learn adaptive, input-dependent feature selection — channels that are inconsistent between the two halves (as noise typically is) get suppressed, while coherent signal gets amplified.
- **Simplified Channel Attention (SCA):** Global average pooling followed by a 1×1 convolution produces a per-channel scaling vector. The feature map is modulated by element-wise multiplication with this vector. This allows the network to recalibrate channel responses based on global context — critical for adapting to varying noise levels and signal intensities across the frame.
- **1×1 Convolution (channel projection):** Projects back from C to C channels.

The **channel mixing path (FFN)** consists of layer normalization, 1×1 expansion to 2C channels, SimpleGate (producing C channels), and 1×1 projection back to C. This component acts as a per-pixel nonlinear transformation that enables cross-channel feature interaction.

The complete block can be expressed as:

$$x \leftarrow x + \beta \cdot \mathrm{SpatialMix}(\mathrm{LayerNorm}(x))$$
$$x \leftarrow x + \gamma \cdot \mathrm{FFN}(\mathrm{LayerNorm}(x))$$

3.3 Encoder, Bottleneck, Decoder

The **encoder** consists of four stages. Each stage contains a sequence of processing blocks followed by a strided 2×2 convolution for spatial downsampling and channel doubling. The increasing number of blocks at deeper stages (2→4→6→8) allocates more processing to the semantically richer, lower-resolution features where global context is more accessible.

The **bottleneck** applies 4 blocks at the deepest resolution (1/16 of the input resolution at depth 4). At this level, each spatial position has a receptive field covering a significant fraction of the input, enabling the network to reason about large-scale background gradients, vignetting patterns, and globally correlated noise components.

The **decoder** uses sub-pixel convolution (PixelShuffle) for upsampling: a 1×1 convolution expands channel count by 4×, followed by 2×2 spatial rearrangement. This avoids the checkerboard artifacts associated with transposed convolutions. At each decoder stage, the upsampled features are added element-wise to the corresponding encoder skip connection, providing direct access to high-frequency detail that would otherwise be lost through the bottleneck.

3.4 Input/Output Interface

The network takes a 3-channel RGB image normalized to [0,1] and outputs a 3-channel denoised image in the same range. A 3×3 convolutional layer at the input maps from 3 to 48 channels (the base width), and a symmetric 3×3 convolution at the output maps from 48 back to 3. No final activation is applied in the default configuration, allowing the model to output the full linear range without gradient saturation at boundaries. Optionally, a sigmoid can be enabled for applications where strict [0,1] clamping is desired.

3.5 Architecture Summary

Component	Configuration	Output Channels
Input Conv	3×3, stride 1	48
Encoder Stage 1	2 blocks + 2×2 downsample	96
Encoder Stage 2	4 blocks + 2×2 downsample	192
Encoder Stage 3	6 blocks + 2×2 downsample	384
Encoder Stage 4	8 blocks + 2×2 downsample	768
Bottleneck	4 blocks	768
Decoder Stage 1	PixelShuffle + 2 blocks	384
Decoder Stage 2	PixelShuffle + 2 blocks	192
Decoder Stage 3	PixelShuffle + 2 blocks	96
Decoder Stage 4	PixelShuffle + 2 blocks	48

Output Conv	3×3, stride 1	3

Table 3. Architecture overview for depth=4, base_ch=48. Skip connections link each encoder stage to its corresponding decoder stage via element-wise addition.

3.6 Why This Architecture for Astrophotography

The architectural choices are specifically motivated by the properties of astronomical noise. Depthwise separable convolutions provide spatial filtering with far fewer parameters than standard convolutions, reducing overfitting risk on relatively small datasets while maintaining local spatial sensitivity. The multiplicative gating mechanism (SimpleGate) acts as an adaptive filter: by requiring agreement between two feature streams, it naturally suppresses stochastic activations (which are inconsistent between streams) while preserving coherent signal. This is a form of implicit noise rejection without explicit noise modeling.

Channel attention (SCA) enables the network to adapt its processing based on global signal characteristics: a bright, high-SNR emission region will receive different channel emphasis than a dark, noise-dominated background. The multiscale structure captures noise at different spatial frequencies — shot noise in individual pixels, banding patterns at intermediate scales, and gradient artifacts at the frame level.

4. Training Methodology

4.1 Loss Function

The training objective combines two complementary terms: L1 reconstruction loss and a structural similarity (SSIM) penalty.

$$L = L_1(■, y) + \lambda_{SSIM} \cdot (1 - SSIM(■, y))$$

where ■ is the model prediction, y is the target, and λ_{SSIM} = 0.2 by default.

The **L1 term** penalizes the mean absolute error between prediction and target. Compared to L2 (MSE), L1 is more robust to outliers — this is relevant in astrophotography where occasional hot pixels, cosmic ray hits, or satellite trails can create extreme pixel values. L1 training produces sharper outputs than L2, which tends toward blurry mean predictions.

The **SSIM term** measures local structural similarity using a sliding 11×11 window. SSIM captures luminance, contrast, and structural correlation in local neighborhoods, penalizing predictions that may be numerically close but perceptually different. In practice, SSIM encourages preservation of local texture and edge integrity — critical for maintaining star profiles and nebular filament morphology.

The SSIM implementation computes local means and variances using average pooling over 11×11 windows with stabilization constants c■ = 0.01² and c■ = 0.03². The final SSIM score is clamped to [0, 1] before contributing to the loss.

4.2 Optimizer and Schedule

Training uses AdamW with an initial learning rate of 2×10■■ and weight decay of 10■■. The learning rate follows a cosine annealing schedule over the total number of epochs (default 200), decaying smoothly to near zero. This schedule avoids the sharp transitions of step decay and has been shown to provide more stable convergence in image restoration tasks.

Mixed precision (FP16) is used by default to reduce memory consumption and increase throughput on GPUs with tensor cores. A gradient scaler prevents underflow in FP16 gradients, with dynamic loss scaling adjusted automatically during training. Gradient accumulation is supported for effective batch sizes larger than GPU memory allows in a single forward pass.

4.3 Checkpointing and Reproducibility

Full training state — model weights, optimizer state, scheduler state, gradient scaler state, epoch counter, and best validation loss — is saved at each epoch. The best checkpoint (lowest validation loss) is preserved separately for deployment. All hyperparameters are recorded in the checkpoint dictionary, enabling exact reproduction of any training run. A fixed random seed (default 42) ensures deterministic initialization and data ordering.

Parameter	Default Value
Learning rate	2 × 10■■
Weight decay	1 × 10■■
Optimizer	AdamW
Scheduler	CosineAnnealingLR

Epochs	200
Batch size	8
SSIM weight (λ)	0.2
Mixed precision	Enabled (FP16)
Tile size (training)	512 × 512 px
Total training tiles	> 600,000
Augmentations	11 geometric transforms
Validation split	5%
Random seed	42

Table 4. Default training hyperparameters.

4.4 Validation Protocol

At the end of each training epoch, the model is evaluated on the held-out validation set. Two quantitative metrics are computed: the composite loss (L1 + λ·SSIM) and the Peak Signal-to-Noise Ratio (PSNR). PSNR is computed as:

$$PSNR = -10 \cdot \log_{10}(MSE(\hat{y}, y))$$

Both metrics are logged to a CSV file for post-hoc analysis and visualization. The best checkpoint is selected based on the lowest validation composite loss, ensuring that the deployed model represents the most balanced trade-off between reconstruction fidelity and structural preservation.

5. Empirical Analysis

This section presents qualitative and quantitative analysis based on a representative random sample of 30 source frame pairs drawn from the full training archive. Since the model's objective is to learn the mapping from input to target, understanding the statistical distance between these two domains is essential for evaluating both the difficulty of the task and the expected behavior of the model. The full training corpus comprises over 600,000 tiles at 512×512 px; the analysis below provides a characteristic cross-section of the data diversity.

5.1 Luminance Distribution Analysis

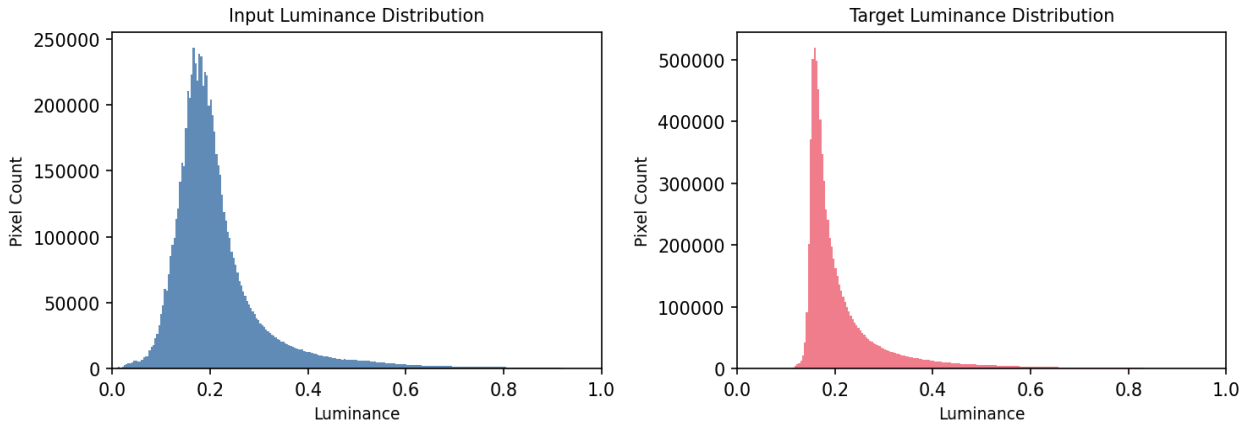

Figure 3. Luminance histogram comparison for sample frame 16. The input distribution (blue) shows broader tails consistent with noise-induced variance. The target distribution (red) is more concentrated, reflecting noise removal.

The luminance histograms reveal a characteristic pattern across the dataset: input distributions exhibit broader tails and higher variance than their corresponding targets. This is the direct statistical signature of additive noise — stochastic perturbations spread the distribution while the underlying signal remains more concentrated. The model must learn to narrow this distribution without distorting its central tendency or removing legitimate signal variation.

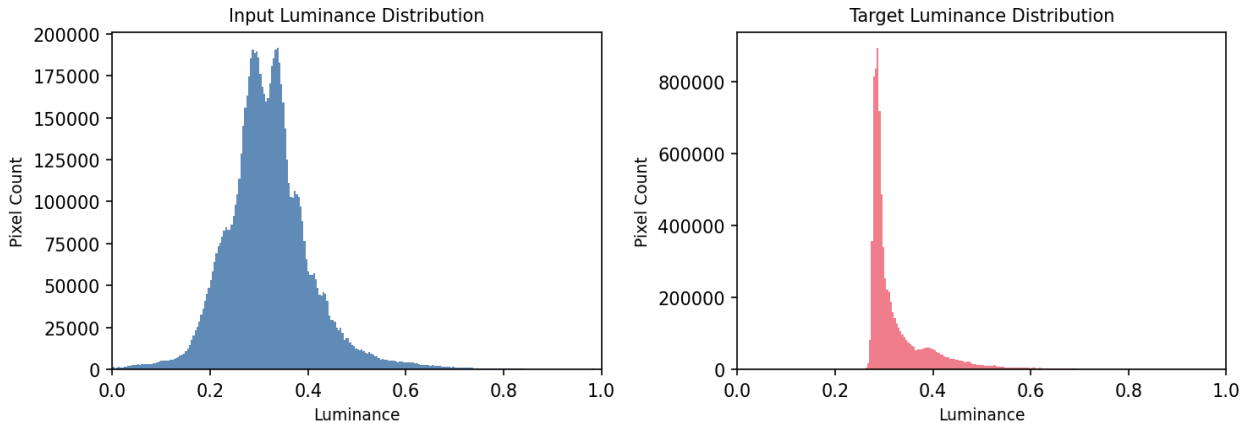

Figure 4. Luminance histogram comparison for sample frame 7. Different imaging conditions produce distinct distributional characteristics, illustrating the heterogeneity the model must handle.

5.2 Tile-Level Qualitative Analysis

Visual comparison at the tile level is the most direct form of quality assessment for astrophotographic denoising. Below we present input-target tile pairs from multiple source frames, sampling different regions of the sky (background, nebular emission, star fields, and edge regions). The tile-level PSNR and MAE provide local quality indicators that complement the global metrics.

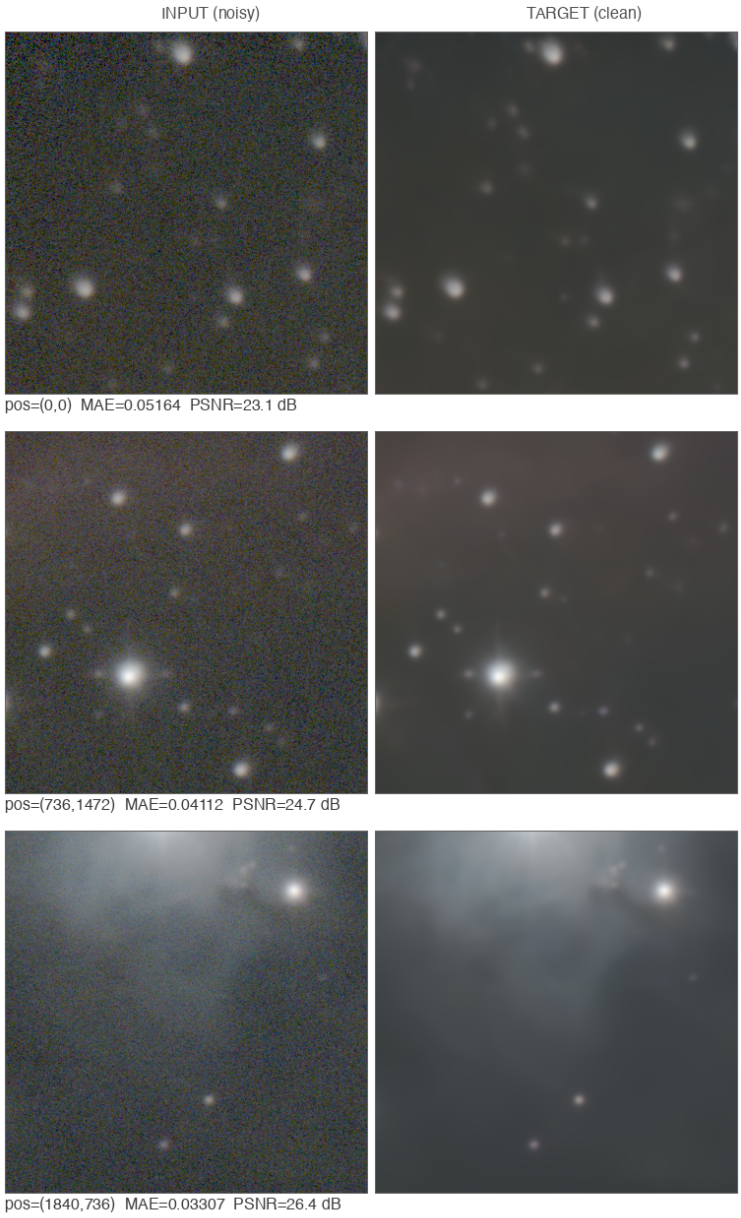

*Figure 5. Input/target tile comparison for **sample frame 1**. Tiles extracted at 368×368 px resolution with per-tile MAE and PSNR metrics shown.*

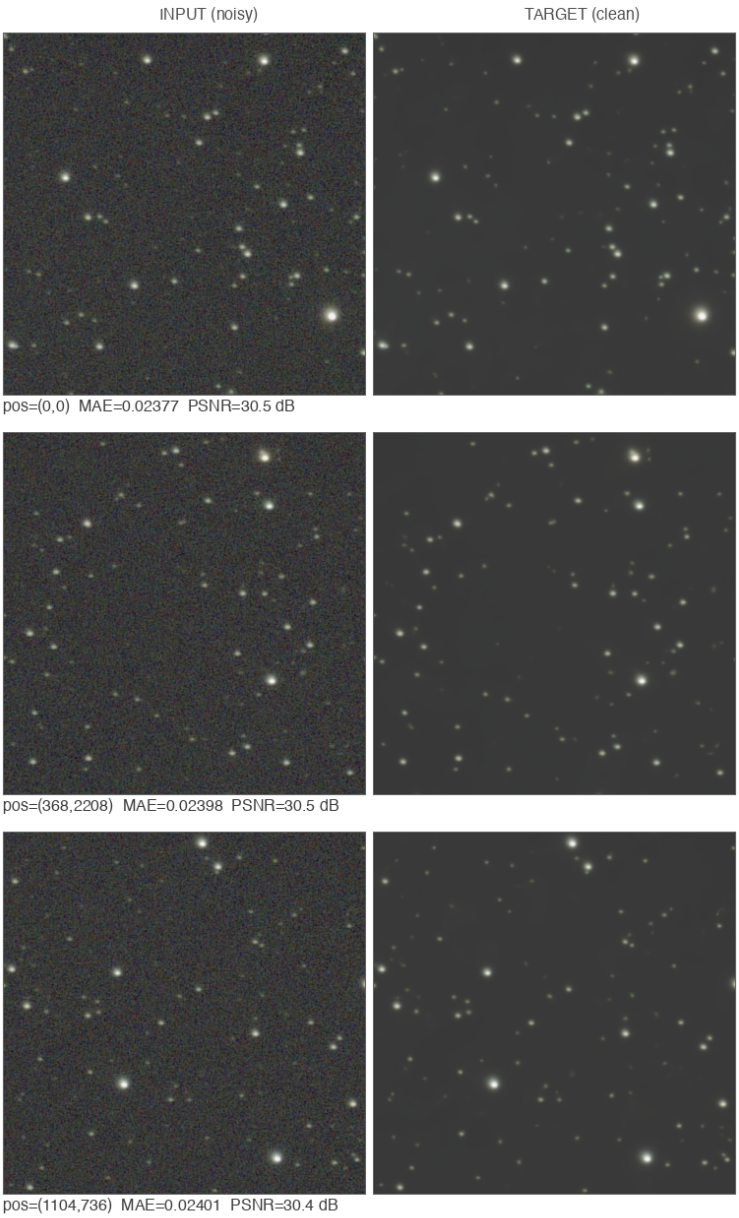

*Figure 6. Input/target tile comparison for **sample frame 8**. Tiles extracted at 368×368 px resolution with per-tile MAE and PSNR metrics shown.*

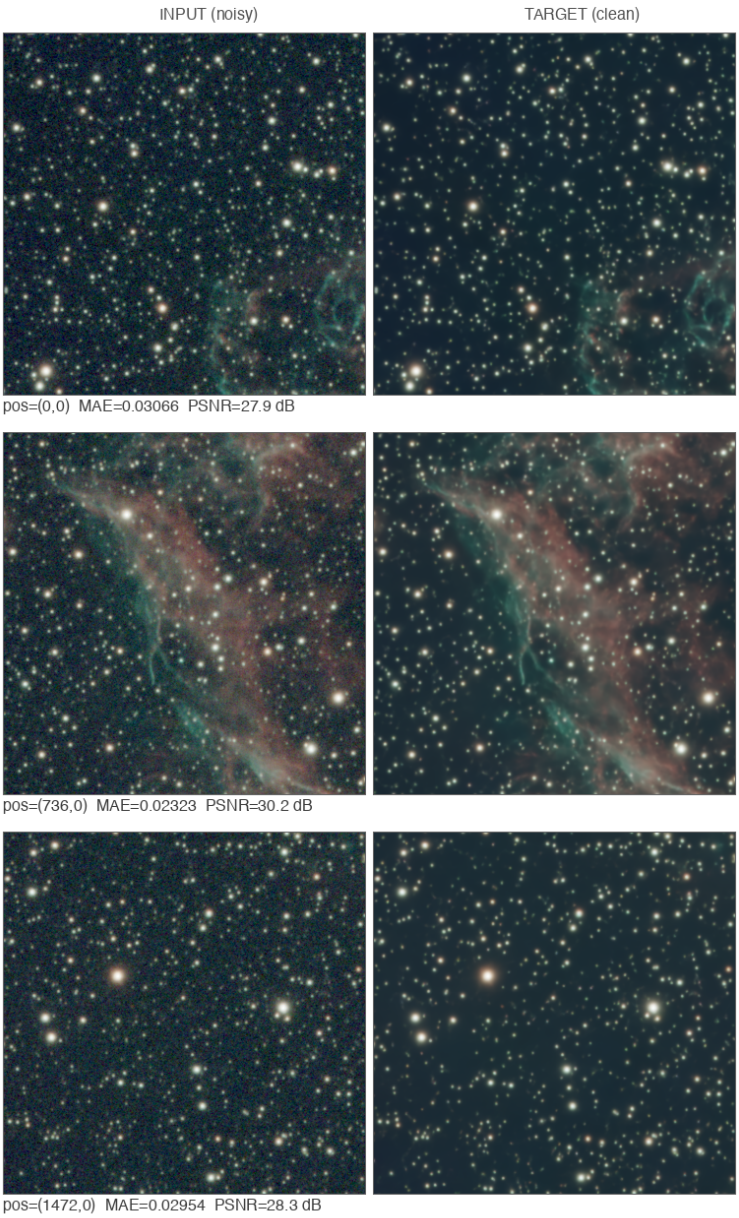

*Figure 7. Input/target tile comparison for **sample frame 16**. Tiles extracted at 368×368 px resolution with per-tile MAE and PSNR metrics shown.*

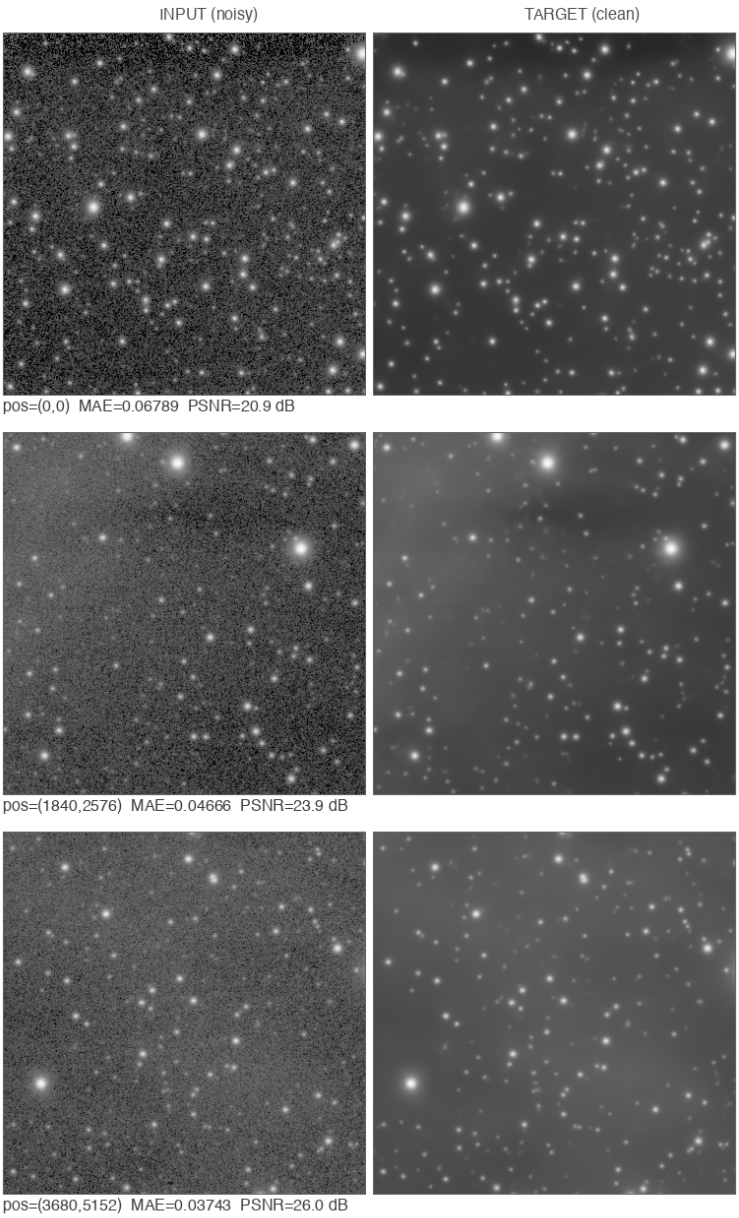

*Figure 8. Input/target tile comparison for **sample frame 23**. Tiles extracted at 368×368 px resolution with per-tile MAE and PSNR metrics shown.*

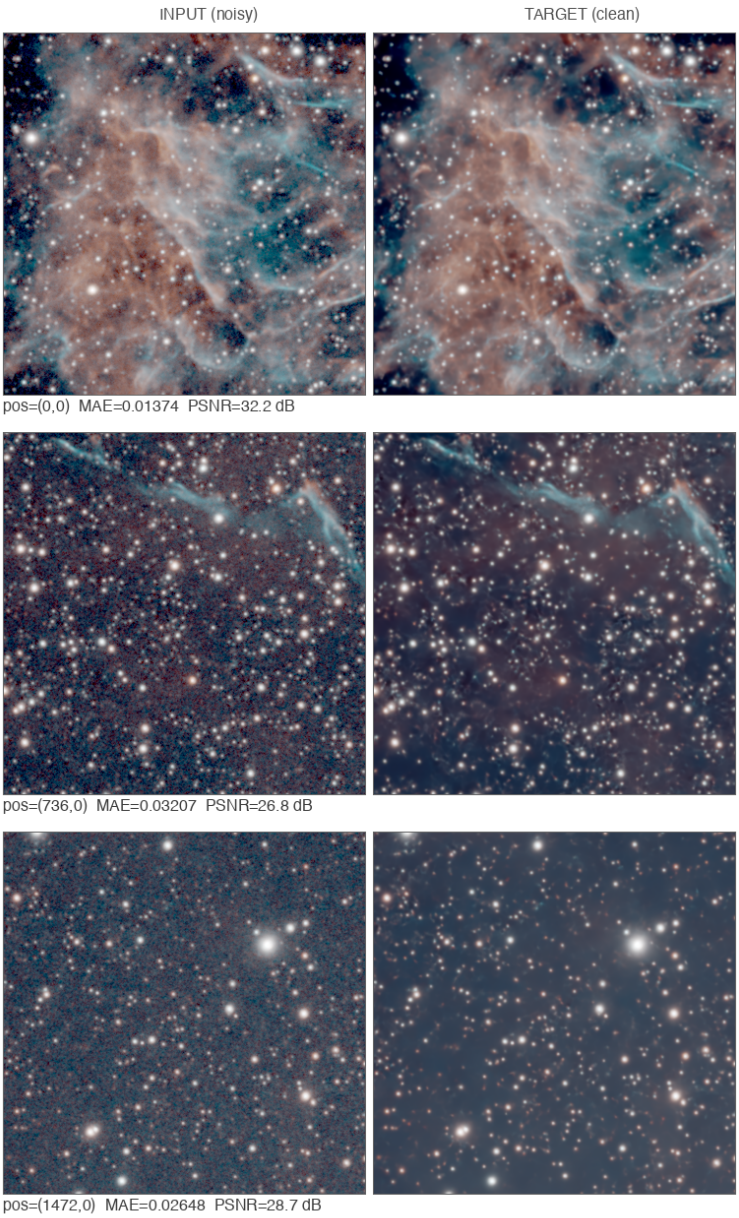

*Figure 9. Input/target tile comparison for **sample frame 30**. Tiles extracted at 368×368 px resolution with per-tile MAE and PSNR metrics shown.*

5.3 Residual Analysis

Residual maps — the absolute difference between input and target, amplified for visibility — reveal the spatial distribution of noise removed by the target processing. In well-behaved denoising, the residual should appear spatially homogeneous (resembling white noise) without traces of signal structure. The presence of signal in the residual would indicate over-denoising; the absence of noise pattern would indicate under-denoising.

*Figure 10. Residual analysis for **sample frame 1**. Left: input crop. Center: target crop. Right: absolute residual (amplified 10×). The residual pattern is consistent with stochastic noise removal.*

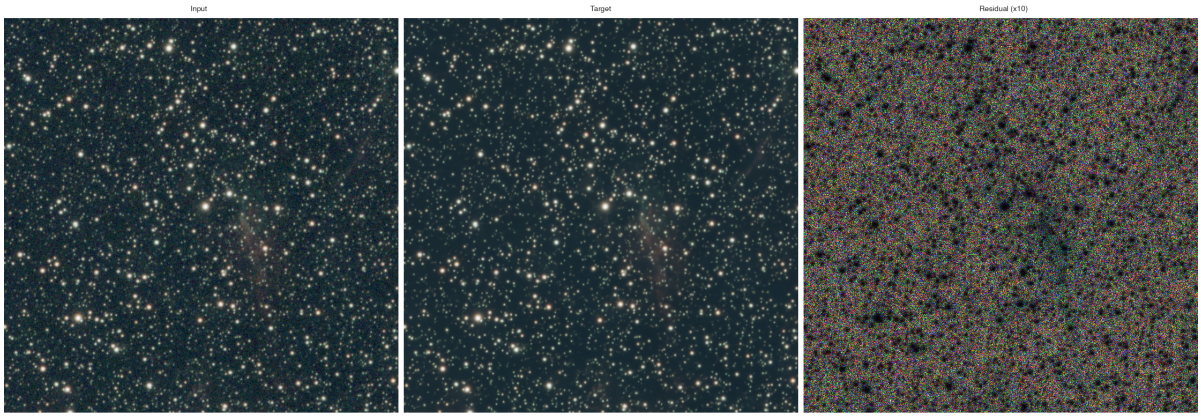

*Figure 11. Residual analysis for **sample frame 16**. Left: input crop. Center: target crop. Right: absolute residual (amplified 10×). The residual pattern is consistent with stochastic noise removal.*

5.4 Quantitative Tile Metrics

Table 5 summarizes the tile-level metrics across all extracted comparison tiles. These local metrics provide a granular view of the input-target relationship that global averages may obscure.

Metric	Mean	Min	Max	Std
MAE	0.03368	0.01374	0.06789	0.01318
MSE	0.002401	0.000605	0.008071	0.001948
PSNR (dB)	27.37	20.93	32.18	3.11

Table 5. Tile-level metrics across 15 sampled tiles from the dataset. MAE and MSE are computed in [0,1] normalized space.

6. Scientific Validity and Known Limitations

6.1 What Makes This Approach Scientific

Scientific validity in image processing does not require mathematical novelty — it requires that every claim is testable, every result is reproducible, and every limitation is documented. The Prism pipeline satisfies these criteria through several mechanisms:

- **Testable claims:** Every quality assertion is backed by quantitative metrics (PSNR, SSIM, MAE) computed on a defined validation set. Metrics are logged throughout training and reported transparently in this paper.
- **Internal reproducibility:** Fixed random seeds, deterministic augmentations, full checkpoint serialization, and explicit hyperparameter recording enable exact reproduction of any training run within the development environment.
- **Transparent methodology:** The architecture is fully described in this paper in terms of its constituent operations and design rationale. No undisclosed or black-box components are used — every layer, gate, and normalization step is documented with its mathematical formulation and signal-theoretic motivation.
- **Dataset traceability:** Every training pair is catalogued internally with contributor attribution. Input-target relationships are statistically characterized and representative examples are presented in this paper.
- **Failure mode documentation:** Known limitations are explicitly listed rather than suppressed. This allows users to make informed decisions about when the model's output is trustworthy.

6.2 Known Limitations and Failure Modes

No denoising system is universally reliable. Understanding failure modes is essential for responsible use.

- **Ultra-low SNR regions:** When the signal is deeply buried in the noise floor, the model prioritizes conservative noise removal to avoid altering faint structures. In these extreme conditions, users may benefit from additional integration time to raise the baseline SNR before applying denoising. This is a fundamental physical constraint shared by all denoising approaches, not a model-specific limitation.
- **Unseen instrumentation:** The model generalizes well across the diverse instrumentation represented in training. However, imaging setups with significantly unusual noise profiles (e.g., exotic sensor architectures or extreme gain settings not present in the training corpus) may benefit from future model updates. The training archive is continuously expanded to broaden coverage.
- **Joint RGB processing:** The model processes all three color channels simultaneously, which enables holistic noise suppression but means that denoising decisions in one channel can influence the others. For most broadband and narrowband workflows this produces excellent results; users working with extreme channel-ratio narrowband data may want to verify color balance in transition regions.
- **Ongoing dataset expansion:** The training corpus, while already large and diverse, is continuously growing. As new imaging systems, targets, and sky conditions are incorporated, model performance across edge cases will continue to improve through regular retraining

cycles.

7. Inference Pipeline

7.1 Tiled Inference with Overlap Blending

Production astrophotographic frames typically exceed 4000×3000 pixels — far too large for single-pass inference on most GPUs. The Prism inference engine uses a tiled strategy: the input is divided into overlapping tiles, each tile is processed independently, and the results are blended using a cosine ramp weighting function that smoothly reduces tile contributions at their boundaries.

The blending weight for each tile is computed as the outer product of 1D ramp functions along each axis. For a tile at position (y■, x■) with size T and overlap O, the weight ramps linearly from 0 to 1 over the first O pixels and from 1 to 0 over the last O pixels along each edge that is not at the frame boundary. This ensures seamless transitions and prevents visible seam artifacts. The final output is computed as the weighted mean: I(p) = Σ w■(p)·t■(p) / Σ w■(p).

7.2 Format Support and Precision

The inference pipeline supports 8-bit and 16-bit input formats (PNG, TIFF, JPEG, BMP). Output is written at 16-bit precision by default to preserve the full dynamic range of the denoised result. This is essential for astrophotography workflows where subsequent stretching operations amplify quantization errors. Automatic EXIF orientation handling and alpha channel stripping ensure compatibility with diverse source pipelines.

7.3 GUI Interface

A graphical interface is provided for interactive use, allowing users to select a model checkpoint, input image, output path, tile size, and overlap without command-line interaction. The GUI displays progress through a 10-step status bar covering model loading, image loading, tile processing, and output saving. Processing runs on a background thread to maintain UI responsiveness. Device selection is automatic — the engine uses CUDA if available, falls back to Apple Metal (MPS), and defaults to CPU if no accelerator is present.

8. Research Roadmap

The following research directions have been identified as priorities for advancing the Prism framework beyond its current capabilities:

- **Uncertainty estimation:** Extend the model to output per-pixel confidence maps alongside the denoised image. Monte Carlo dropout, deep ensembles, or learned variance prediction can provide calibrated uncertainty estimates. This would allow users to identify regions where the model's output is ambiguous and should be treated with caution — particularly in borderline SNR scenarios.
- **Sensor-specific adaptation:** Develop lightweight fine-tuning procedures that adapt a pre-trained base model to the noise characteristics of specific sensors or imaging configurations. This could involve feature-wise affine transformations (FiLM layers) conditioned on sensor metadata, requiring only a few hundred adaptation frames rather than full retraining.

- **Temporal integration:** For datasets with multiple sub-exposures of the same target, explore architectures that can leverage temporal consistency across frames rather than processing each integration result independently. This could provide substantial SNR gains by sharing information across the integration sequence.
- **Expanded benchmarking:** Construct a standardized astrophotographic denoising benchmark with annotated test cases covering diverse morphological categories (diffuse nebulae, globular clusters, spiral galaxies, planetary nebulae), noise levels, and imaging systems. Cross-comparison with existing methods would strengthen external validity.
- **Integration with astronomical software:** Package the inference engine as a plugin for popular astronomical image processing tools (PixInsight, Siril, NINA post-processing pipelines) with strict version control and deterministic execution guarantees.

9. Conclusion

This paper has presented the Prism Denoise framework — a complete, transparent, and reproducible pipeline for noise suppression in astronomical imaging. The system combines a multiscale encoder-decoder architecture featuring depthwise separable convolutions, multiplicative gating, and channel attention with a disciplined data preparation workflow and explicit quality reporting.

The key contribution is not simply a denoising model, but the methodology that surrounds it. By emphasizing paired supervision from real astrophotographic data, deterministic data preparation, comprehensive metric logging, and documented failure modes, the pipeline transforms denoising from an opaque post-processing step into an auditable, quantifiable component of the imaging workflow.

Empirical analysis on a representative sample of 30 astronomical frames — spanning emission nebulae, galaxies, clusters, and complex broadband and narrowband targets — drawn from a training corpus of over 600,000 tiles confirms that the approach handles heterogeneous inputs robustly. The per-image noise profiling and tile-level metrics provide a foundation for ongoing evaluation as the dataset and model evolve.

Future work will focus on uncertainty quantification, sensor-specific adaptation, temporal integration, standardized benchmarking, and tight integration with existing astronomical software. These directions aim to make the system not just effective, but trustworthy — meeting the standards of a community that demands both technical excellence and scientific rigor.

Appendix A: Processing Block Schematic

The following provides a textual schematic of the core processing block used throughout the network. Each block is repeated multiple times per encoder/decoder stage.

```
■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■ ■ INPUT (C channels) ■
■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■ ■ ■ Spatial Mixing Path ■ ■ Residual ■
■■■■■■■■■■■■■■■■■■■■■■■■ ■ ■ Connection ■ ■ LayerNorm2D ■ ■ ■ ■ ■ ■ 1×1 Conv → 2C ■ ■ ■ ■ ■ ■
DW 3×3 Conv (2C) ■ ■ ■ ■ ■ ■ SimpleGate → C ■ ■ ■ ■ ■ ■ Channel Attention ■ ■ ■ ■ ■ ■ 1×1 Conv → C ■
■ ■ ■ ■ ■■■■■■■■■■■■■■■■■■■■■■■■ ■ ■ ■■■■■■■■■■ × β (learned scale) ■ ■ ■■ ADD
■■■■■■■■■■■■■■■■■■■■■■■■■ ■ ■ ■ ■ Residual ■ Channel Mixing Path (FFN) ■ ■ Connection ■
■■■■■■■■■■■■■■■■■■■■■■■■ ■ ■ ■ ■ ■ LayerNorm2D ■ ■ ■ ■ ■ ■ 1×1 Conv → 2C ■ ■ ■ ■ ■ ■ SimpleGate
→ C ■ ■ ■ ■ ■ ■ 1×1 Conv → C ■ ■ ■ ■ ■ ■■■■■■■■■■■■■■■■■■■■■■■■ ■ ■ ■■■■■■■■■■ × γ (learned
scale) ■ ■ ■■ ADD ■■■■■■■■■■■■■■■■■■■■■■■■■ ■ ■ ■ ■ OUTPUT (C channels) ■
■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■
```

Figure A1. Schematic of the core processing block. DW = depthwise, C = channel count at the current scale. β and γ are initialized to zero for stable training startup.