

Fast Kernel-Space Diffusion for Remote Sensing Pansharpening

Hancong Jin

jhc261973@gmail.com

Zihan Cao

iamzihan666@gmail.com

Liang-jian Deng

liangjian.deng@uestc.edu.cn

Jingjing Li

jjl@uestc.edu.cn

University of Electronic Science and Technology of China

Abstract

Pansharpening seeks to fuse high-resolution panchromatic (PAN) and low-resolution multispectral (LRMS) images into a single image with both fine spatial and rich spectral detail. Despite progress in deep learning-based approaches, existing methods often fail to capture global priors inherent in remote sensing data distributions. Diffusion-based models have recently emerged as promising solutions due to their powerful distribution mapping capabilities, however, they suffer from heavy inference latency. We introduce KSDiff, a fast kernel-space diffusion framework that generates convolutional kernels enriched with global context to enhance pansharpening quality and accelerate inference. Specifically, KSDiff constructs these kernels through the integration of a low-rank core tensor generator and a unified factor generator, orchestrated by a structure-aware multi-head attention mechanism. We further introduce a two-stage training strategy tailored for pansharpening, facilitating integration into existing pansharpening architectures. Experiments show that KSDiff achieves superior performance compared to recent promising methods, and with over $500\times$ faster inference than diffusion-based pansharpening baselines. Ablation studies, visualizations and further evaluations substantiate the effectiveness of our approach. Code will be released upon possible acceptance.

1. Introduction

Due to the inherent physical limitations of satellite sensors, the direct acquisition of high-resolution multispectral images (HRMS) is not feasible. Instead, sensors typically provide high-resolution panchromatic (PAN) images and low-resolution multispectral images (LRMS). Pansharpening techniques aim to fuse the spatial detail of PAN images with the spectral richness of LRMS, thereby generating HRMS with enhanced spatial and spectral fidelity. As a foundational

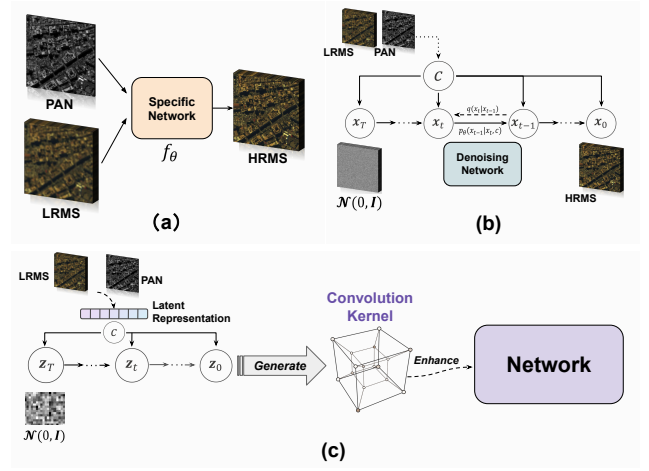


Figure 1. (a) Traditional DL-based methods directly learn a non-linear mapping f_θ to fuse PAN and LRMS images in a one-step manner. (b) Recent diffusion-based methods employ a multi-step refinement process in the pixel space conditioned on PAN and LRMS from pure Gaussian noise $\mathcal{N}(0, I)$. The $q(x_t|x_{t-1})$, $p_\theta(x_{t-1}|x_t, c)$ and c denote the noise adding process, the reverse denoising process, and the condition, respectively. (c) The proposed KSDiff, which generates convolution kernels to enhance regression-based pansharpening networks via a diffusion model performing in the latent space. This design enables high-quality reconstruction with fast inference.

preprocessing approach, pansharpening plays a critical role in numerous remote sensing applications, including image segmentation [67] and change detection [59].

Over the past decades, a variety of pansharpening techniques have been proposed, which can be broadly categorized into four families: component substitution (CS) [25, 35], multi-resolution analysis (MRA) [37, 52], variational optimization (VO) [61, 62], and deep learning (DL)-based methods. While classical CS and MRA methods offer interpretable mathematical formulations, they often intro-

duce noticeable spectral or spatial distortions due to oversimplified assumptions. VO-based methods improve flexibility by formulating pansharpening as an optimization problem with prior constraints, but their performance is sensitive to model design and hyperparameters.

In recent years, deep learning, particularly convolutional neural networks (CNNs), has emerged as a powerful paradigm for pansharpening, achieving impressive performance across benchmark datasets. Representative approaches such as PanNet [65], DiCNN [15], and FusionNet [8] model the fusion process as a non-linear mapping from PAN and LRMS inputs to an HRMS output. These methods typically rely on improved architectural designs [27, 60] or more effective modules [10, 20] to enhance learning capacity. Most existing deep learning-based models are deterministic, which enables fast inference but limits their capacity to capture global statistical dependencies and the complex high-dimensional data distributions inherent in remote sensing imagery.

Diffusion models [18, 21, 42], rooted in generative modeling, learn complex data distributions through iterative denoising steps, reconstructing samples from Gaussian noise. Unlike conventional deep learning methods mentioned above, they model the full conditional distribution of outputs given inputs, enabling flexible, context-aware generation. This makes them ideal for pansharpening, as they integrate heterogeneous inputs like PAN and LRMS images to produce spectrally consistent, high-resolution multispectral images with high fidelity [5, 34]. However, a common limitation of diffusion-based methods lies in their slow inference speed due to the iterative nature of the sampling process [21, 45]. This latency issue becomes particularly critical in the pansharpening task, where the target images are of high resolution and the input data typically contains more channels than standard RGB images. These factors significantly increase the difficulty of distribution mapping within diffusion models. Moreover, in the context of pansharpening, where most spatial and spectral information is already available, the role of the network is primarily to refine rather than reconstruct from scratch. Consequently, reconstructing the entire image from pure Gaussian noise is both counterintuitive and burdensome.

Considering the respective strengths and limitations of traditional deep learning methods and diffusion-based approaches for pansharpening, we propose a novel framework named KSDiff. KSDiff leverages the powerful distribution modeling capability of diffusion processes to enrich convolutional kernels with high-level remote sensing context, while preserving the inference efficiency of conventional DL-based methods. Specifically, we train a diffusion process in latent space [39, 63], significantly reducing the computational burden compared to modeling the diffusion process in the full pixel space and ensuring efficient inference. The

generated latent representations are then used to construct low-rank convolutional cores, which are subsequently expanded into full convolutional kernels via a structure-aware multi-head attention mechanism that integrates the generated convolution cores with features extracted from the PAN and LRMS images, as shown in Fig. 2. To effectively train KSDiff with regression backbones, we further introduce a two-stage training scheme and a Pyramid Latent Fusion Encoder (PLFE) tailored to the pansharpening task. KSDiff can be integrated into existing networks, boosting performance over baselines. Remarkably, it achieves inference speeds comparable to traditional DL methods, orders of magnitude faster than recent diffusion-based pansharpening approaches. Our main contributions are as follows:

- We propose KSDiff, a novel framework that integrates diffusion-generated latent representation into the design of convolutional kernels. This approach leverages the strong distribution modeling capabilities of diffusion models while achieving fast inference speed.
- We introduce a two-stage training scheme and a Pyramid Latent Fusion Encoder (PLFE) specifically tailored for pansharpening. These components enable effective integration of multi-scale spatial-spectral information and allow KSDiff to be incorporated into existing regression-based networks.
- We conduct extensive experiments on multiple benchmark datasets, demonstrating that KSDiff achieves competitive performance in both quantitative metrics and visual quality. In addition, it offers inference speeds comparable to traditional DL-based methods and significantly faster than recent diffusion-based pansharpening methods.

2. Related Works

2.1. Deep Learning Based Methods

PNN [33] was the first to apply CNNs to pansharpening, inspired by single-image super-resolution techniques. To better preserve high-frequency details, PanNet [65] introduced a strategy that injects high-pass PAN information into the upsampled LRMS image. DiCNN [15] further enhanced detail preservation using a detail-injection framework, while FusionNet [8] employed a residual network to explicitly learn high-frequency components. Subsequent works, such as DCFNet [60] and PMACNet [27], improved CNN architectures and achieved superior performance. CTINN [72] adopted a transformer-based design to capture long-range dependencies in the fusion process. PanMamba[16] explored the potential of the state-space model. To address the limitations of static convolutional kernels, LAGConv [20] introduced dynamic kernels conditioned on the input. Building on this idea, AKD [38] proposed dual dynamically generated branches for extracting spatial and spectral details. Another branch of methods delved into frequency domain analysis

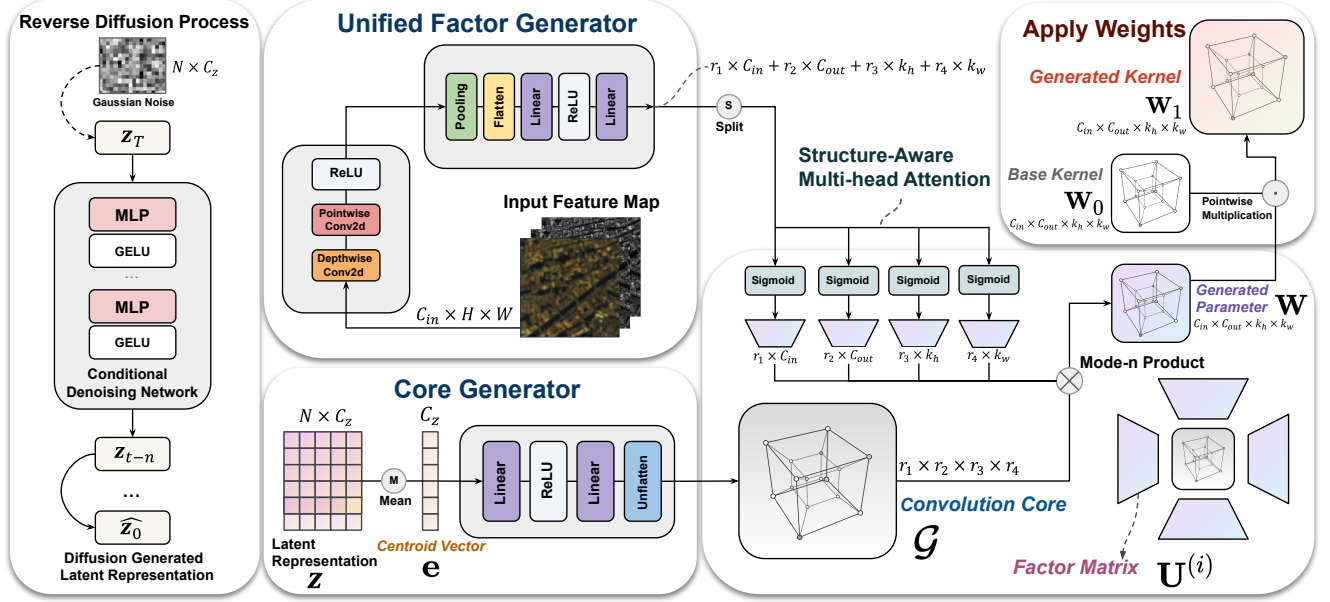


Figure 2. Kernel Generator of our proposed KSDiff. The kernel generator comprises two sub-modules: (1) a diffusion model-driven convolutional core generator, (2) a unified factor generator that takes feature maps as input. The outputs of these two modules are integrated using a structure-aware multi-head attention mechanism to reparameterize the base kernel. Note that in the pre-training stage, the latent representation $\mathbf{z}$ is the output of an encoder; in the diffusion model training stage and at inference time, this representation is generated by the diffusion model, namely $\hat{\mathbf{z}}_0$.

of PAN and LRMS images, such as HFIN [48]. These advancements have significantly expanded the research scope of DL-based pansharpening methods.

2.2. Diffusion Models

Diffusion models (DMs) [18, 46] have recently advanced generative modeling by formulating data synthesis as a sequence of denoising steps. While initially applied to image generation [18], subsequent developments enabled high-resolution synthesis [39], multimodal conditioning [11], and efficient sampling [45, 69]. DMs have also shown strong promise in low-level vision tasks [26]. StableSR [56] achieved real-world image super-resolution by leveraging the prior knowledge from large pre-trained text-to-image latent diffusion model. DiffIR [63] combined the training of a transformer with a diffusion model for realistic image restoration. DPS [6] introduced zero-shot posterior sampling for general inverse problems. In pansharpening, most methods [5, 34, 41, 64] use conditional diffusion models [42] in pixel space, conditioning on PAN and LRMS to iteratively generate HRMS outputs. Recent theory [2] unifies diffusion, flow matching [28], and diffusion-bridge models [7, 71] under stochastic differential equations (SDEs) and ordinary differential equations (ODEs), enabling flexible distribution mapping. However, solving these SDEs/ODEs demands many network function evaluations (NFEs), incurring high computational cost. Acceleration strategies, such

as advanced samplers [21, 43] and distillation techniques [30, 47], reduce NFEs but either require full model retraining or lead to performance trade-offs. Some recent work further explores the versatile use of diffusion models. For example, Neural Network Diffusion [57] employs an autoencoder-diffusion pipeline to synthesize latent representations of network parameters and decode them into novel, high-performing weights.

3. Methodology

In this section, we present the design of the kernel generator in our proposed KSDiff, as well as a two-stage training scheme tailored for the pansharpening task. Fig. 2 illustrates the architecture of the KSDiff kernel generator. Fig. 3 shows the structure of the latent encoder. Fig. 4 depicts the two-stage training pipeline and the inference. We introduce our method following the order of training procedures.

3.1. Pre-training Stage

In the pre-training stage, our objective is to encode the ground truth high-resolution multispectral (HRMS) image into a compact latent representation, which is then utilized to guide the convolutional kernels within pansharpening networks during image fusion. As illustrated in Fig. 3 and Fig. 4, we employ a Pyramid Latent Fusion Encoder (PLFE) to compress the ground truth HRMS image and extract a compact

prior feature. This prior feature is subsequently used to generate low-rank convolutional core tensors (i.e., $\mathcal{G}$ in Fig. 2), from which full convolutional kernels are reconstructed. By incorporating this learned prior, our method enables kernel-wise guidance across the pansharpening network, thereby enhancing the fidelity and spatial-spectral consistency of the reconstructed multispectral image. The components of this framework are described in detail below.

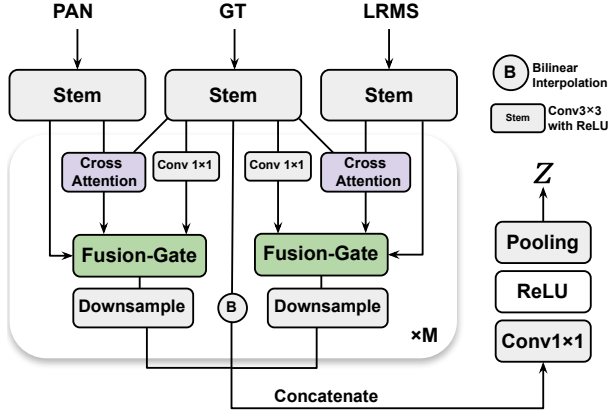


Figure 3. Pyramid Latent Fusion Encoder (PLFE). The figure only shows the structure of PLFE_1 ; since PLFE_2 only takes PAN and LRMS as input, its structure is slightly different (a halved version compared to PLFE_1). Further details can be found in supplementary materials Sec. 6.

Pyramid Latent Fusion Encoder (PLFE). To fully leverage the complementary priors of the PAN, LRMS and HRMS modalities while avoiding the spatial-spectral entanglement inherent in naive concatenation, we introduce the Pyramid Latent Fusion Encoder (PLFE). Its design is driven by two principles: first, a multi-scale pyramid architecture in which PAN and LRMS features are progressively refined under HRMS guidance at each level, integrating high-resolution spatial cues with contextual spectral semantics; and second, a dynamic fusion gate that adaptively balances the contributions of the original branch features and the HRMS-guided features to maintain spatial-spectral consistency throughout the encoding process. Specifically, given the PAN image $\mathbf{P} \in \mathbb{R}^{H \times W \times 1}$, the LRMS image $\mathbf{M} \in \mathbb{R}^{H \times W \times C}$ and the ground-truth HRMS image $\mathbf{G} \in \mathbb{R}^{H \times W \times C}$, PLFE first embeds each via a stem block, followed by M pyramid stages to capture features at multi-scale. At a certain stage, the PAN or LRMS branch features $\mathbf{X} \in \mathbb{R}^{H' \times W' \times d}$ are refined under guidance from the HRMS feature $\mathbf{Y} \in \mathbb{R}^{H' \times W' \times d}$ using cross attention with linear complexity [22, 44], alleviating the computational burden of this dual-branch attention when facing large images. Its mechanism (see supplementary ma-

terials Fig. 8 (b) for graphic illustration) is as follows:

$$\begin{aligned} \mathbf{Q} &= \text{Rsp}(\text{Softmax}(\mathbf{Q}, 1)), \mathbf{K} = \text{Rsp}(\text{Softmax}(\mathbf{K}, 2)) \\ \mathbf{V} &= \text{Rsp}(\mathbf{V}), \mathbf{A} = \mathbf{K} \odot \mathbf{V}^T, \mathbf{O} = \text{Conv}(\mathbf{A}^T \odot \mathbf{Q}) \end{aligned} \quad (1)$$

where $\mathbf{Q}$, $\mathbf{K}$, $\mathbf{V}$, $\mathbf{A}$, $\mathbf{O}$ represent the query, key, value, attention map, and output, respectively, $\odot$ is the elementwise multiplication. $\mathbf{Q}$, $\mathbf{K}$ and $\mathbf{V}$ are obtained from $\mathbf{X}$ and $\mathbf{Y}$ with several convolution layers. Rsp (Reshape) operation flattens the spatial dimensions of $\mathbf{Q}$, $\mathbf{K}$, and $\mathbf{V}$, into a single dimension, i.e., $\mathbb{R}^{H \times W \times d} \rightarrow \mathbb{R}^{HW \times d}$, and $\text{Softmax}(\mathbf{K}, j)$ represents the Softmax operation along the j th dimension. Consequently, the memory complexity is reduced from $\mathcal{O}((HW)^2)$ to $\mathcal{O}(d^2)$, with $d \ll HW$, compared to vanilla attention [50]. We then adaptively fuse the guidance from the cross attention via a dynamic fusion gate:

$$\begin{aligned} \mathbf{G}_{\text{gate}} &= \sigma(\text{Conv}_g[\mathbf{X}, \text{Proj}(\mathbf{Y})]), \\ \mathbf{F} &= \mathbf{G}_{\text{gate}} \odot \mathbf{X} + (1 - \mathbf{G}_{\text{gate}}) \odot \text{Proj}(\mathbf{Y}) + \mathbf{O}. \end{aligned} \quad (2)$$

where Proj is for channel alignment, $[\cdot]$ means concatenation, and σ is a Sigmoid function along channel dimension. By learning $\mathbf{G}_{\text{gate}}$, the network balances trusting the HRMS prior-refining features where guidance may misalign, thus enhancing spatial-spectral consistency and reducing artifacts, yielding better encoding quality. A strided convolution down-samples $\mathbf{F}$, while $\mathbf{Y}$ is bilinearly interpolated for the next stage. After M stages, the final features are concatenated and then projected into $\mathbf{z} \in \mathbb{R}^{N \times C_z}$, with N and C_z representing the token number and the embedded dimension respectively, obtaining a compact prior representation ($N \ll HW$) for downstream kernel modulation.

Kernel Generator. We explore an efficient strategy to enrich convolutional layers with encoded latent representations that capture high-level global context. Given a convolutional kernel represented as a 4D tensor of shape $C_{\text{in}} \times C_{\text{out}} \times k_h \times k_w$ (typically with $k_h = k_w = k$), a naive solution is to flatten the latent code $\mathbf{z}$, pass it through an MLP, and reshape the output to obtain the convolution kernel. However, this approach introduces a prohibitive number of parameters and lacks flexibility in controlling the influence of the latent representation across different kernels within a network. To address these issues, we propose to modulate a standard base kernel $\mathbf{W}_0 \in \mathbb{R}^{C_{\text{in}} \times C_{\text{out}} \times k_h \times k_w}$ with a learned weight tensor $\mathbf{W}$, yielding the final kernel as $\mathbf{W}_1 = \mathbf{W}_0 \odot \mathbf{W}$, where $\odot$ denotes element-wise multiplication. To reduce complexity, we factorize $\mathbf{W}$ via tensor algebra [24, 49]:

$$\mathbf{W} = \mathcal{G} \times_1 \mathbf{U}^{(1)} \times_2 \mathbf{U}^{(2)} \times_3 \mathbf{U}^{(3)} \times_4 \mathbf{U}^{(4)}, \quad (3)$$

where $\times_n$ denotes mode- n product, with a compact core tensor $\mathcal{G} \in \mathbb{R}^{r_1 \times r_2 \times r_3 \times r_4}$ and factor matrices $\{\mathbf{U}^{(n)}\}$ of size

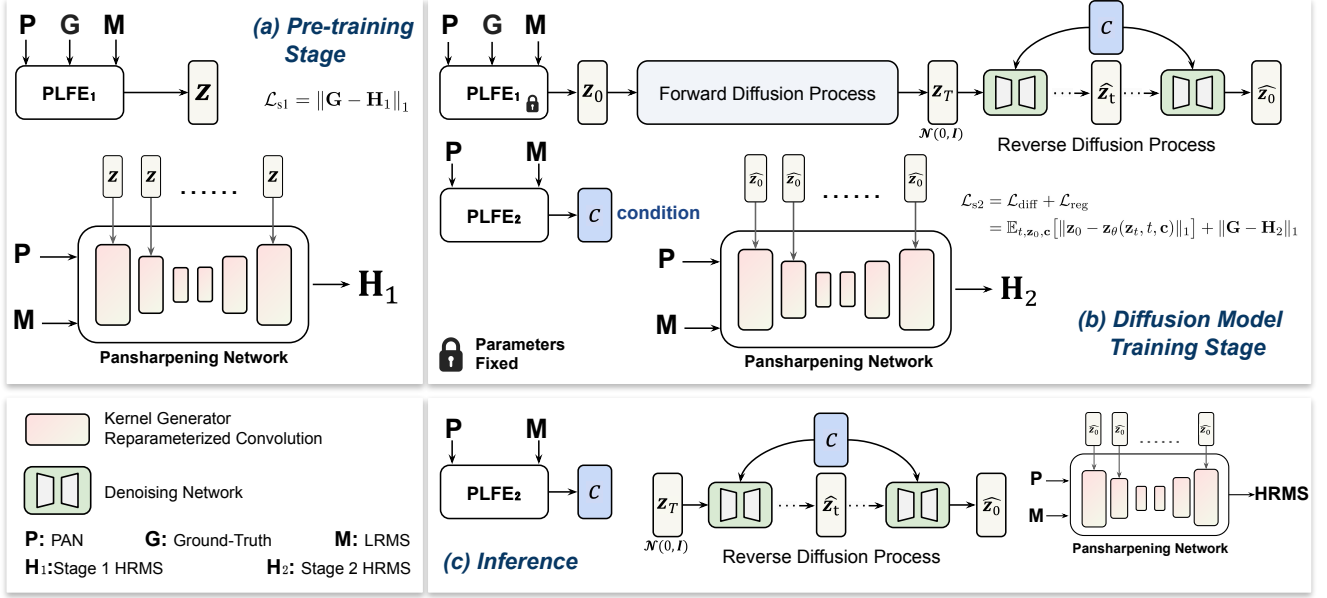


Figure 4. An overview of our two-stage training procedure and inference process. (a) **Pre-training Stage**: The goal is to extract a latent representation $\mathbf{z}$ by optimizing PLFE_1 jointly with the kernel generator and the pansharping network. (b) **Diffusion Model Training Stage**: The latent representation $\mathbf{z}_0$, extracted by the pre-trained PLFE_1 , is predicted by leveraging the strong distribution estimation ability of the diffusion model (no ground truth is used as input for its denoising network). (c) **Inference**: Only PLFE_2 , the reverse diffusion process, the kernel generator, and the pansharping network are involved.

$\mathbf{U}^{(1)} \in \mathbb{R}^{r_1 \times C_{\text{in}}}$, $\mathbf{U}^{(2)} \in \mathbb{R}^{r_2 \times C_{\text{out}}}$, $\mathbf{U}^{(3)} \in \mathbb{R}^{r_3 \times k_h}$, and $\mathbf{U}^{(4)} \in \mathbb{R}^{r_4 \times k_w}$. The core tensor is generated by applying mean pooling on $\mathbf{z}$ to obtain a centroid vector $\mathbf{e} \in \mathbb{R}^{C_z}$, which is processed by an MLP. The factor matrices are obtained from input features via a lightweight shared backbone followed by four attention heads, which we refer to as structure-aware multi-head attention, as shown in Fig. 2. This design achieves efficient, expressive kernel modulation with controllable prior influence. An experiment on the influence of different low-rank core tensor sizes is presented in Sec. 4.4. In comparison to the naive MLP-based approach with $\mathcal{O}(C_{\text{in}}C_{\text{out}}k^2C_z)$ complexity, this formulation has $\mathcal{O}(C_z r_1 r_2 r_3 r_4 + \sum_{n=1}^4 r_n d_n)$ theoretical complexity, where d_n denotes the size of the n -th mode. With $r_n \ll d_n$, this leads to notable savings in computation and memory while supporting flexible kernel-wise adaptation.

Pansharping Network. To better demonstrate the performance gains achieved by our proposed KSDiff, we construct a typical U-Net [40] as the pansharping backbone and integrate our kernel generation method into several of its convolutional layers. A detailed description of the network architecture is provided in supplementary materials Sec. 6.

Training Technique. To enhance the capability of the latent encoder in constructing an informative prior feature representation, we jointly optimize it alongside the kernel

generator within the integrated pansharping framework. The training objective employs an L_1 loss function, which is commonly chosen by reconstruction tasks [5, 66], formulated as:

$$\mathcal{L}_{s1} = \|\mathbf{G} - \mathbf{H}_1\|_1, \quad (4)$$

where $\mathbf{G}$ denotes the ground-truth high-resolution multispectral (HRMS) image, and $\mathbf{H}_1$ represents the reconstructed HRMS output.

3.2. Diffusion Model Training Stage

In this stage, as illustrated in Fig. 4 (b), we leverage the powerful distribution estimation capability of diffusion models to approximate the latent representation prior. Our framework adopts both DDPM [18] and DDIM [45] paradigms to formulate the forward and backward processes. While more recent advancements in diffusion scheduling [21, 28] or more efficient sampling strategies [32, 68, 69] could potentially be incorporated into our framework, we intentionally employ the classical approach to maintain both simplicity and excellent pansharping performance.

Diffusion Model. In the forward diffusion process, given the ground-truth high-resolution multispectral (HRMS) image, we first employ the pre-trained PLFE_1 from the previous stage to obtain the corresponding prior feature $\mathbf{z}_0 \in \mathbb{R}^{N \times C_z}$, which serves as the initial state of the forward

Markov chain. We then gradually add Gaussian noise over T iterations as follows:

$$q(\mathbf{z}_{1:T} | \mathbf{z}_0) = \prod_{t=1}^T q(\mathbf{z}_t | \mathbf{z}_{t-1}), \quad (5)$$

$$q(\mathbf{z}_t | \mathbf{z}_{t-1}) = \mathcal{N}(\mathbf{z}_t; \sqrt{1 - \beta_t} \mathbf{z}_{t-1}, \beta_t \mathbf{I}),$$

where $t = 1, \dots, T$, $\mathbf{z}_t$ denotes the noisy feature at step t , and $\beta_{1:T} \in (0, 1)$ controls the noise schedule. By utilizing the reparameterization trick [23], one can derive the closed-form marginal:

$$q(\mathbf{z}_t | \mathbf{z}_0) = \mathcal{N}(\mathbf{z}_t; \sqrt{\bar{\alpha}_t} \mathbf{z}_0, (1 - \bar{\alpha}_t) \mathbf{I}),$$

$$\alpha_t = 1 - \beta_t, \quad \bar{\alpha}_t = \prod_{i=1}^t \alpha_i. \quad (6)$$

The reverse diffusion process aims to recover the prior feature from an isotropic Gaussian initialization. It is formulated as a T -step Markov chain running backward from $\mathbf{z}_T$ to $\mathbf{z}_0$. At each reverse step, the posterior distribution is given by:

$$p(\mathbf{z}_{t-1} | \mathbf{z}_t, \mathbf{z}_0) = \mathcal{N}(\mathbf{z}_{t-1}; \boldsymbol{\mu}_t, \boldsymbol{\Sigma}_t),$$

$$\boldsymbol{\mu}_t = \frac{1}{\sqrt{\alpha_t}} \left(\mathbf{z}_t - \frac{1 - \alpha_t}{\sqrt{1 - \bar{\alpha}_t}} \boldsymbol{\epsilon} \right), \quad \boldsymbol{\Sigma}_t = \frac{1 - \bar{\alpha}_{t-1}}{1 - \bar{\alpha}_t} \beta_t \mathbf{I}. \quad (7)$$

where $\boldsymbol{\epsilon}$ denotes the noise component in $\mathbf{z}_t$, which can be estimated by a denoising network ϵ_θ (see Fig. 2 and Fig. 4 (b)). Since we perform diffusion in the latent space, we adopt a second PLFE, denoted PLFE₂, to condense the panchromatic image $\mathbf{P}$ and the LRMS image $\mathbf{M}$ into the conditional latent $\mathbf{c} \in \mathbb{R}^{N \times C_z}$. Note that PLFE₂ retains a halved structure compared to the pre-trained PLFE₁, which is symmetric (see Fig. 3). To accelerate sampling, we employ the DDIM sampler [45], which operates in a non-Markovian manner:

$$\mathbf{z}_{t-n} = \sqrt{\bar{\alpha}_{t-n}} \frac{\mathbf{z}_t - \sqrt{1 - \bar{\alpha}_t} \epsilon_\theta(\mathbf{z}_t, t, \mathbf{c})}{\sqrt{\bar{\alpha}_t}} + \sqrt{1 - \bar{\alpha}_{t-n} - \sigma_t^2} \epsilon_\theta(\mathbf{z}_t, t, \mathbf{c}) + \sigma_t^2 \boldsymbol{\epsilon}. \quad (8)$$

where σ_t is a predetermined function of t and n denotes the number of skipped steps. After sampling, we obtain the predicted prior $\hat{\mathbf{z}}_0 \in \mathbb{R}^{N \times C_z}$, which is then fed into the kernel generation process to guide the entire pansharpening network.

Training Technique. The training objective of standard DDPM (denoted as ϵ -prediction) often underperforms in some specific tasks [5, 29]. We therefore reparameterize the diffusion loss to predict the original sample $\mathbf{z}_0$, which is mathematically equivalent to ϵ -prediction [18]. Concretely, we optimize:

$$\mathcal{L}_{\text{diff}} = \mathbb{E}_{t, \mathbf{z}_0, \mathbf{c}} [\|\mathbf{z}_0 - \mathbf{z}_\theta(\mathbf{z}_t, t, \mathbf{c})\|_1]. \quad (9)$$

We observe that jointly training the diffusion model and the pansharpening regressor yields superior high-resolution multispectral (HRMS) reconstructions compared to a separate training scheme, corroborating findings in other domains [13, 63]. Accordingly, our overall objective becomes:

$$\mathcal{L}_{s2} = \mathcal{L}_{\text{diff}} + \lambda \mathcal{L}_{\text{reg}} = \mathbb{E}_{t, \mathbf{z}_0, \mathbf{c}} [\|\mathbf{z}_0 - \mathbf{z}_\theta(\mathbf{z}_t, t, \mathbf{c})\|_1] + \lambda \|\mathbf{G} - \mathbf{H}_2\|_1 \quad (10)$$

where $\mathbf{G}$ is the ground-truth HRMS image, and $\mathbf{H}_2$ denotes the pansharpened output produced during diffusion-model training. The weighting factor λ is empirically set to 1. This unified loss encourages both accurate diffusion estimation and faithful spectral-spatial reconstruction in a single end-to-end framework.

3.3. Inference.

At inference time (see Fig. 4 (c)), only the reverse diffusion process is employed. The PLFE₂ module extracts a conditional code $\mathbf{c}$ from PAN and LRMS images, which guides the sampling process starting from pure Gaussian noise $\mathbf{z}_T \in \mathbb{R}^{N \times C_z}$ through the trained conditional denoising network. The resulting latent representation $\hat{\mathbf{z}}_0$ is then fed into the kernel generator, enabling the pansharpening network to fuse PAN and LRMS images with the global context provided by this representation.

4. Experiments

4.1. Datasets, Metrics and Implementation Details

We validate our approach using datasets constructed following Wald’s protocol [8, 55], based on satellite data from WorldView-3 (WV3), GaoFen-2 (GF2), and QuickBird (QB). The data and preprocessing methods are obtained from the PanCollection repository [9]. We evaluate our method using SAM [4], ERGAS [54], Q2ⁿ [12], and SCC [70] for reduced-resolution datasets, and HQNR [3], D_λ , and D_s for full-resolution datasets. The model is trained with the AdamW optimizer [31] on a single NVIDIA GeForce RTX 4090 GPU. Additional details on datasets, compared methods, and training configurations are provided in *supplementary materials Sec. 7*, including those related to the main experiments, ablation study, and discussion section.

4.2. Experimental Results

Our proposed KSDiff demonstrates superior performance through extensive evaluations on three benchmark datasets: WV3, GF2, and QB. Tab. 1 to Tab. 3 compare KSDiff with state-of-the-art methods, including traditional, conventional DL-based, and diffusion-based approaches, confirming its robustness and ability to produce high-quality pan-sharpened images with fast inference. Visual comparisons in Fig. 5 show KSDiff’s results closely align with the ground truth.

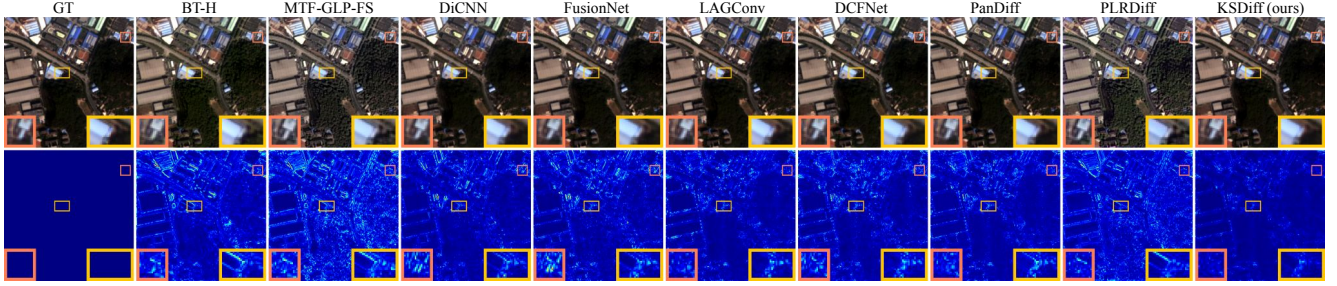


Figure 5. Comparison of qualitative results for representative methods on the GF2 reduced-resolution dataset. The first row displays RGB outputs, and the second row presents the error maps.

Table 1. Result on the WV3 reduced-resolution and full-resolution datasets. The best results are highlighted in bold and the second best results are underlined.

Method	Reduced resolution				Full resolution			Runtime (s)
	SAM ($\pm$ std)	ERGAS ($\pm$ std)	Q2 ⁿ ($\pm$ std)	SCC ($\pm$ std)	D_λ ($\pm$ std)	D_s ($\pm$ std)	HQNR ($\pm$ std)	
BDS-PC [51]	5.4675 $\pm$ 0.3842	4.6549 $\pm$ 0.3278	0.8117 $\pm$ 0.0238	0.9049 $\pm$ 0.0094	0.0625 $\pm$ 0.0053	0.0730 $\pm$ 0.0080	0.8698 $\pm$ 0.0119	0.059
MTF-GLP-FS [53]	5.3233 $\pm$ 0.3700	4.6452 $\pm$ 0.3229	0.8177 $\pm$ 0.0227	0.8984 $\pm$ 0.0104	0.0206 $\pm$ 0.0018	0.0630 $\pm$ 0.0064	0.9180 $\pm$ 0.0077	0.023
BT-H [1]	4.8985 $\pm$ 0.2913	4.5150 $\pm$ 0.2977	0.8182 $\pm$ 0.0228	0.9240 $\pm$ 0.0054	0.0574 $\pm$ 0.0052	0.0810 $\pm$ 0.0084	0.8670 $\pm$ 0.0121	0.321
PNN [33]	3.6798 $\pm$ 0.1705	2.6819 $\pm$ 0.1448	0.8929 $\pm$ 0.0206	0.9761 $\pm$ 0.0017	0.0213 $\pm$ 0.0018	0.0428 $\pm$ 0.0033	0.9369 $\pm$ 0.0047	0.042
DiCNN [15]	3.5929 $\pm$ 0.1705	2.6733 $\pm$ 0.1482	0.9004 $\pm$ 0.0195	0.9763 $\pm$ 0.0016	0.0362 $\pm$ 0.0025	0.0426 $\pm$ 0.0039	0.9195 $\pm$ 0.0058	0.083
MSDCNN [58]	3.7773 $\pm$ 0.1796	2.7608 $\pm$ 0.1539	0.8900 $\pm$ 0.0201	0.9741 $\pm$ 0.0017	0.0230 $\pm$ 0.0020	0.0467 $\pm$ 0.0045	0.9316 $\pm$ 0.0061	0.112
FusionNet [8]	3.3252 $\pm$ 0.1560	2.4666 $\pm$ 0.1442	0.9044 $\pm$ 0.0202	0.9807 $\pm$ 0.0015	0.0239 $\pm$ 0.0020	0.0364 $\pm$ 0.0031	0.9406 $\pm$ 0.0044	0.065
CTINN [72]	3.2523 $\pm$ 0.1439	2.3936 $\pm$ 0.1161	0.9056 $\pm$ 0.0188	0.9826 $\pm$ 0.0014	0.0550 $\pm$ 0.0064	0.0679 $\pm$ 0.0070	0.8815 $\pm$ 0.0109	1.329
LAGConv [20]	3.1042 $\pm$ 0.1249	2.2999 $\pm$ 0.1368	0.9098 $\pm$ 0.0188	0.9830 $\pm$ 0.0015	0.0368 $\pm$ 0.0033	0.0418 $\pm$ 0.0036	0.9230 $\pm$ 0.0055	1.381
MMNet [73]	3.0844 $\pm$ 0.1430	2.3428 $\pm$ 0.1400	0.9155 $\pm$ 0.0191	0.9829 $\pm$ 0.0013	0.0540 $\pm$ 0.0052	<u>0.0336$\pm$0.0026</u>	0.9143 $\pm$ 0.0063	0.348
DCFNet [60]	3.0264 $\pm$ 0.1653	2.1588 $\pm$ 0.1020	0.9051 $\pm$ 0.0197	0.9861 $\pm$ 0.0009	0.0781 $\pm$ 0.0182	0.0508 $\pm$ 0.0076	0.8771 $\pm$ 0.0225	0.548
PanMamba [16]	<u>2.9132$\pm$0.1230</u>	2.1843 $\pm$ 0.1092	<u>0.9204$\pm$0.0187</u>	0.9855 $\pm$ 0.0011	0.0183$\pm$0.0018	0.0531 $\pm$ 0.0078	0.9304 $\pm$ 0.0061	0.405
PanDiff [34]	3.2968 $\pm$ 0.2240	2.4647 $\pm$ 0.1305	0.8935 $\pm$ 0.0193	0.9860 $\pm$ 0.0014	0.0273 $\pm$ 0.0027	0.0542 $\pm$ 0.0059	0.9203 $\pm$ 0.0081	261.410
PLRDiff [41]	4.3704 $\pm$ 0.3288	3.4408 $\pm$ 0.2627	0.8539 $\pm$ 0.0297	0.9215 $\pm$ 0.0081	0.1796 $\pm$ 0.0083	0.1037 $\pm$ 0.0060	0.7361 $\pm$ 0.0113	40.142
KSDiff (ours)	2.8102$\pm$0.1147	2.0756$\pm$0.0973	0.9221$\pm$0.0183	0.9870$\pm$0.0010	0.0210 $\pm$ 0.0019	0.0322$\pm$0.0026	0.9468$\pm$0.0043	0.077
Ideal value	0	0	1	1	0	0	1	0

For additional results, please refer to supplementary materials Sec. 8.

4.3. Ablation Study

To validate the effectiveness of each component in our proposed method, we conducted ablation studies, with results summarized in Tab. 4. Firstly, to assess the impact of the latent diffusion prior, we evaluated our pansharpening network without incorporating the latent representation generated by the diffusion model. The results demonstrate that omitting the latent representation compromises performance, despite a reduction in latency. Secondly, we ablated the proposed Pyramid Latent Fusion Encoder (PLFE) by replacing it with an encoder that directly concatenates the PAN, LRMS, and GT images as input. This substitution led to degraded performance, highlighting the importance of PLFE in our framework. We further investigated the tensor structure-aware kernel generator by replacing it with an MLP containing an equivalent number of intermediate features. This modification increased the number of learnable parameters by over tenfold, resulting in convergence failure, underscoring the efficiency and necessity of our tensor-aware design. Finally,

to evaluate the joint training strategy in the diffusion model training stage, we conducted an ablation by first training the diffusion model to learn the latent representation independently and then integrating it with a pre-trained regression network for testing. This approach yielded inferior performance compared to our joint training strategy, confirming its critical role in achieving optimal results.

4.4. Discussions

Latent Representation Visualization. As shown in Fig. 6, we visualize latent diffusion representations from PAN and LRMS image pairs across diverse scenes. Representations from ocean-dominated images (left two subfigures) exhibit similar distributions, as do those from building and land-dominated images (right two subfigures), with variations due to building density. This demonstrates the diffusion model’s ability to effectively capture intrinsic global information in PAN and LRMS images.

Other Backbones. To further assess the effectiveness of KSDiff in enhancing pansharpening networks, we substi-



Figure 6. Visualization of latent representations generated by KSDiff from PAN and LRMS images across different scenes.

tuted convolutional layers in various pansharpening backbones with KSDiff. The results, shown in Tab. 5, indicate that KSDiff consistently improves the performance of DiCNN [15], FusionNet [8], and LAGNet [20].

Size of Low-Rank Core Tensor. We investigated how the size of the low-rank core tensor $\mathcal{G} \in \mathbb{R}^{r_1 \times r_2 \times r_3 \times r_4}$ affects performance on the WV3 reduced-resolution dataset by scaling its dimensions. Using FusionNet [8] with our KSDiff-enhanced backbone, we found that smaller core tensors can achieve comparable or even better results than larger ones under the same conditions, similar to LoRA [19]. However, as shown in Tab. 6, increasing the last two modes (kernel size) from 1 to 2 improves performance, likely because a $(r_1, r_2, 1, 1)$ tensor collapses to a matrix, losing useful 4D structure. This indicates that maintaining higher-dimensional tensor structure can benefit model performance.

5. Conclusion

In conclusion, we proposed KSDiff, a novel kernel space diffusion model that improves pansharpening by generating globally aware convolutional kernels through a diffusion process in latent space. By integrating a convolution core generator, a unified factor generator, and a structure-aware multi-head attention mechanism, KSDiff effectively balances high-quality fusion with fast inference. Extensive experiments on various datasets demonstrate its superior performance, offering a practical and scalable solution for remote sensing image fusion.

References

- [1] Bruno Aiazzi, Luciano Alparone, Stefano Baronti, Andrea Garzelli, and Massimo Selva. Mtf-tailored multiscale fusion of high-resolution ms and pan imagery. *Photogrammetric Engineering & Remote Sensing*, 72(5):591–596, 2006. 7, 8, 2, 3, 4
- [2] Michael S Albergo, Nicholas M Boffi, and Eric Vanden-Eijnden. Stochastic interpolants: A unifying framework for flows and diffusions. *arXiv preprint arXiv:2303.08797*, 2023. 3
- [3] Alberto Arienzo, Gemine Vivone, Andrea Garzelli, Luciano Alparone, and Jocelyn Chanussot. Full-resolution quality

Table 2. Result on the GF2 reduced-resolution dataset. The best results are highlighted in bold and the second best results are underlined.

Method	GaoFen-2			
	SAM ($\pm$ std)	ERGAS ($\pm$ std)	Q2 ⁿ ($\pm$ std)	SCC ($\pm$ std)
BDSD-PC [51]	1.7110 $\pm$ 0.0718	1.7025 $\pm$ 0.0907	0.8932 $\pm$ 0.0069	0.9448 $\pm$ 0.0037
MTF-GLP-FS [53]	1.6757 $\pm$ 0.0773	1.6023 $\pm$ 0.0793	0.8914 $\pm$ 0.0057	0.9390 $\pm$ 0.0044
BT-H [1]	1.6810 $\pm$ 0.0709	1.5524 $\pm$ 0.0814	0.9089 $\pm$ 0.0065	0.9508 $\pm$ 0.0034
PNN [33]	1.0477 $\pm$ 0.0506	1.0572 $\pm$ 0.0527	0.9604 $\pm$ 0.0022	0.9772 $\pm$ 0.0012
DiCNN [15]	1.0525 $\pm$ 0.0376	1.0812 $\pm$ 0.0561	0.9594 $\pm$ 0.0024	0.9781 $\pm$ 0.0012
MSDCNN [58]	1.0472 $\pm$ 0.0494	1.0413 $\pm$ 0.0516	0.9612 $\pm$ 0.0024	0.9782 $\pm$ 0.0011
FusionNet [8]	0.9735 $\pm$ 0.0473	0.9878 $\pm$ 0.0497	0.9641 $\pm$ 0.0021	0.9806 $\pm$ 0.0011
CTINN [72]	0.8251 $\pm$ 0.0310	0.9595 $\pm$ 0.0239	0.9772 $\pm$ 0.0026	0.9803 $\pm$ 0.0003
LAGConv [20]	0.7859 $\pm$ 0.0330	0.6869 $\pm$ 0.0252	0.9804 $\pm$ 0.0009	0.9843 $\pm$ 0.0005
MMNet [73]	0.9292 $\pm$ 0.0322	0.8117 $\pm$ 0.0265	0.9690 $\pm$ 0.0046	0.9859 $\pm$ 0.0005
DCFNet [60]	0.8896 $\pm$ 0.0353	0.8061 $\pm$ 0.0306	0.9727 $\pm$ 0.0022	0.9853 $\pm$ 0.0005
PanMamba [16]	0.7433 $\pm$ 0.0239	0.6840 $\pm$ 0.0245	0.9824 $\pm$ 0.0021	0.9896 $\pm$ 0.0004
PanDiff [34]	0.8881 $\pm$ 0.0268	0.7461 $\pm$ 0.0242	0.9727 $\pm$ 0.0022	0.9887 $\pm$ 0.0004
PLRDiff [41]	2.6755 $\pm$ 0.4609	2.7434 $\pm$ 0.5485	0.7938 $\pm$ 0.0367	0.9055 $\pm$ 0.0170
KSDiff (ours)	0.6675$\pm$0.0299	0.5973$\pm$0.0229	0.9855$\pm$0.0020	0.9900$\pm$0.0005
Ideal value	0	0	1	1

Table 3. Result on the QB reduced-resolution dataset. The best results are highlighted in bold and the second best results are underlined.

Method	QuickBird			
	SAM ($\pm$ std)	ERGAS ($\pm$ std)	Q2 ⁿ ($\pm$ std)	SCC ($\pm$ std)
BDSD-PC [51]	8.2620 $\pm$ 0.4583	7.5420 $\pm$ 0.1819	0.8323 $\pm$ 0.0226	0.9030 $\pm$ 0.0040
MTF-GLP-FS [53]	8.1131 $\pm$ 0.4371	7.5102 $\pm$ 0.1772	0.8296 $\pm$ 0.0202	0.8998 $\pm$ 0.0044
BT-H [1]	7.1943 $\pm$ 0.3470	7.4008 $\pm$ 0.1873	0.8326 $\pm$ 0.0197	0.9156 $\pm$ 0.0034
PNN [33]	5.2054 $\pm$ 0.2152	4.4722 $\pm$ 0.0835	0.9180 $\pm$ 0.0210	0.9711 $\pm$ 0.0027
DiCNN [15]	5.3795 $\pm$ 0.2295	5.1354 $\pm$ 0.1090	0.9042 $\pm$ 0.0211	0.9621 $\pm$ 0.0030
MSDCNN [58]	5.1471 $\pm$ 0.2088	4.3828 $\pm$ 0.0760	0.9180 $\pm$ 0.0216	0.9689 $\pm$ 0.0027
FusionNet [8]	4.9226 $\pm$ 0.2029	4.1594 $\pm$ 0.0718	0.9252 $\pm$ 0.0202	0.9755 $\pm$ 0.0023
CTINN [72]	4.6583 $\pm$ 0.1733	3.6969 $\pm$ 0.0646	0.9320 $\pm$ 0.0016	0.9829 $\pm$ 0.0016
LAGConv [20]	4.5473 $\pm$ 0.1855	3.8259 $\pm$ 0.0938	0.9335 $\pm$ 0.0196	0.9807 $\pm$ 0.0020
MMNet [73]	4.5568 $\pm$ 0.1629	3.6669 $\pm$ 0.0679	0.9337 $\pm$ 0.0210	0.9829 $\pm$ 0.0016
DCFNet [60]	4.5383 $\pm$ 0.1654	3.8315 $\pm$ 0.0652	0.9325 $\pm$ 0.0202	0.9741 $\pm$ 0.0022
PanMamba [16]	4.6253 $\pm$ 0.1773	4.2771 $\pm$ 0.0708	0.9292 $\pm$ 0.0224	0.9801 $\pm$ 0.0022
PanDiff [34]	4.5754 $\pm$ 0.1645	3.7422 $\pm$ 0.0693	0.9345 $\pm$ 0.0202	0.9818 $\pm$ 0.0202
PLRDiff [41]	26.8524 $\pm$ 1.1153	47.4010 $\pm$ 6.8066	0.2971 $\pm$ 0.0550	0.6780 $\pm$ 0.0336
KSDiff (ours)	4.4747$\pm$0.1558	3.6289$\pm$0.0640	0.9365$\pm$0.0180	0.9839$\pm$0.0016
Ideal value	0	0	1	1

Table 4. Ablation study on the WV3 reduced-resolution dataset. The best results are highlighted.

Method	SAM ($\pm$ std)	ERGAS ($\pm$ std)	Q2 ⁿ ($\pm$ std)	SCC ($\pm$ std)	Runtime (s)
Baseline Network	3.1428 $\pm$ 0.1333	2.2961 $\pm$ 0.1349	0.9070 $\pm$ 0.0191	0.9827 $\pm$ 0.0014	0.035
w/o PLFE	3.0071 $\pm$ 0.1306	2.2367 $\pm$ 0.1323	0.9119 $\pm$ 0.0188	0.9838 $\pm$ 0.0014	0.079
w/o Structure-Aware	-	-	-	-	-
Separate-Training	2.9799 $\pm$ 0.1295	2.1775 $\pm$ 0.1266	0.9118 $\pm$ 0.0189	0.9854 $\pm$ 0.0012	0.077
KSDiff (ours)	2.8102$\pm$0.1147	2.0756$\pm$0.0973	0.9221$\pm$0.0183	0.9870$\pm$0.0010	0.077
Ideal value	0	0	1	1	0

assessment of pansharpening: Theoretical and hands-on approaches. *IEEE Geoscience and Remote Sensing Magazine*, 10(3):168–201, 2022. 6

Table 5. Result of replacing convolution modules in other backbones with KSDiff on WV3 reduced-resolution dataset.

Method	Metrics			
	SAM ($\pm$ std)	ERGAS ($\pm$ std)	Q2 ^a ($\pm$ std)	SCC ($\pm$ std)
DiCNN [15]	3.5929 $\pm$ 0.1705	2.6733 $\pm$ 0.1482	0.9004 $\pm$ 0.0195	0.9763 $\pm$ 0.0016
DiCNN+KSDiff	3.4154 $\pm$ 0.1598	2.5278 $\pm$ 0.1410	0.9052 $\pm$ 0.0192	0.9791 $\pm$ 0.0016
FusionNet [8]	3.3252 $\pm$ 0.1560	2.4666 $\pm$ 0.1441	0.9044 $\pm$ 0.0202	0.9807 $\pm$ 0.0015
FusionNet+KSDiff	3.0622 $\pm$ 0.1199	2.2725 $\pm$ 0.1117	0.9111 $\pm$ 0.0188	0.9837 $\pm$ 0.0011
LAGNet [20]	3.1042 $\pm$ 0.1249	2.2999 $\pm$ 0.1368	0.9098 $\pm$ 0.0188	0.9830 $\pm$ 0.0015
LAGNet+KSDiff	2.9903 $\pm$ 0.1277	2.1538 $\pm$ 0.1352	0.9124 $\pm$ 0.0193	0.9851 $\pm$ 0.0014
Ideal value	0	0	1	1

Table 6. Impact of low-rank core tensor size (r_1, r_2, r_3, r_4) on the WV3 reduced-resolution dataset on FusionNet [8].

Size	Metrics			
	SAM ($\pm$ std)	ERGAS ($\pm$ std)	Q2 ^a ($\pm$ std)	SCC ($\pm$ std)
(4, 4, 1, 1)	3.0921 $\pm$ 0.1202	2.2896 $\pm$ 0.1119	0.9104 $\pm$ 0.0188	0.9833 $\pm$ 0.0012
(4, 4, 2, 2)	3.0622 $\pm$ 0.1199	2.2725 $\pm$ 0.1117	0.9111 $\pm$ 0.0188	0.9837 $\pm$ 0.0011
(8, 8, 1, 1)	3.1275 $\pm$ 0.1256	2.3009 $\pm$ 0.1174	0.9057 $\pm$ 0.0200	0.9828 $\pm$ 0.0011
(8, 8, 2, 2)	3.1040 $\pm$ 0.1215	2.2988 $\pm$ 0.1153	0.9096 $\pm$ 0.0191	0.9833 $\pm$ 0.0013
(16, 16, 1, 1)	3.1520 $\pm$ 0.1262	2.3146 $\pm$ 0.1180	0.9052 $\pm$ 0.0195	0.9827 $\pm$ 0.0015
(16, 16, 2, 2)	3.1475 $\pm$ 0.1258	2.3106 $\pm$ 0.1203	0.9055 $\pm$ 0.0194	0.9829 $\pm$ 0.0012
Ideal value	0	0	1	1

[4] Joseph W Boardman. Automating spectral unmixing of aviris data using convex geometry concepts. In *JPL, Summaries of the 4th Annual JPL Airborne Geoscience Workshop. Volume 1: AVIRIS Workshop*, 1993. 6

[5] Zihan Cao, Shiqi Cao, Liang-Jian Deng, Xiao Wu, Junming Hou, and Gemine Vivone. Diffusion model with disentangled modulations for sharpening multispectral and hyperspectral images. *Information Fusion*, 104:102158, 2024. 2, 3, 5, 6

[6] Hyungjin Chung, Jeongsol Kim, Michael T Mccann, Marc L Klasky, and Jong Chul Ye. Diffusion posterior sampling for general noisy inverse problems. *arXiv preprint arXiv:2209.14687*, 2022. 3

[7] Valentin De Bortoli, James Thornton, Jeremy Heng, and Arnaud Doucet. Diffusion schrödinger bridge with applications to score-based generative modeling. *Advances in Neural Information Processing Systems*, 34:17695–17709, 2021. 3

[8] Liang-Jian Deng, Gemine Vivone, Cheng Jin, and Jocelyn Chanussot. Detail injection-based deep convolutional neural networks for pansharpening. *IEEE Transactions on Geoscience and Remote Sensing*, 59(8):6995–7010, 2020. 2, 6, 7, 8, 9, 3, 4

[9] Liang-Jian Deng, Gemine Vivone, Mercedes E Paoletti, Giuseppe Scarpa, Jiang He, Yongjun Zhang, Jocelyn Chanussot, and Antonio Plaza. Machine learning in pansharpening: A benchmark, from shallow to deep networks. *IEEE Geoscience and Remote Sensing Magazine*, 10(3):279–315, 2022. 6, 2

[10] Yule Duan, Xiao Wu, Haoyu Deng, and Liang-Jian Deng. Content-adaptive non-local convolution for remote sensing pansharpening. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 27738–27747, 2024. 2

[11] Patrick Esser, Sumith Kulal, Andreas Blattmann, Rahim Entezari, Jonas Müller, Harry Saini, Yam Levi, Dominik Lorenz,

Axel Sauer, Frederic Boesel, et al. Scaling rectified flow transformers for high-resolution image synthesis. In *Forty-first international conference on machine learning*, 2024. 3

[12] Andrea Garzelli and Filippo Nencini. Hypercomplex quality assessment of multi/hyperspectral images. *IEEE Geoscience and Remote Sensing Letters*, 6(4):662–665, 2009. 6

[13] Chunming He, Kai Li, Guoxia Xu, Jiangpeng Yan, Longxiang Tang, Yulun Zhang, Yaowei Wang, and Xiu Li. Hqg-net: Unpaired medical image enhancement with high-quality guidance. *IEEE Transactions on Neural Networks and Learning Systems*, 2023. 6

[14] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 770–778, 2016. 1, 3

[15] Lin He, Yizhou Rao, Jun Li, Jocelyn Chanussot, Antonio Plaza, Jiawei Zhu, and Bo Li. Pansharpening via detail injection based convolutional neural networks. *IEEE Journal of Selected Topics in Applied Earth Observations and Remote Sensing*, 12(4):1188–1204, 2019. 2, 7, 8, 9, 3, 4

[16] Xuanhua He, Ke Cao, Jie Zhang, Keyu Yan, Yingying Wang, Rui Li, Chengjun Xie, Danfeng Hong, and Man Zhou. Panmamba: Effective pan-sharpening with state space model. *Information Fusion*, 115:102779, 2025. 2, 7, 8, 3, 4

[17] Dan Hendrycks and Kevin Gimpel. Gaussian error linear units (gelu). *arXiv preprint arXiv:1606.08415*, 2016. 3

[18] Jonathan Ho, Ajay Jain, and Pieter Abbeel. Denoising diffusion probabilistic models. *Advances in neural information processing systems*, 33:6840–6851, 2020. 2, 3, 5, 6

[19] Edward J Hu, Yelong Shen, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, Lu Wang, and Weizhu Chen. LoRA: Low-rank adaptation of large language models. In *International Conference on Learning Representations*, 2022. 8

[20] Zi-Rong Jin, Tian-Jing Zhang, Tai-Xiang Jiang, Gemine Vivone, and Liang-Jian Deng. Lagconv: Local-context adaptive convolution kernels with global harmonic bias for pansharpening. In *Proceedings of the AAAI conference on artificial intelligence*, pages 1113–1121, 2022. 2, 7, 8, 9, 3, 4

[21] Tero Karras, Miika Aittala, Timo Aila, and Samuli Laine. Elucidating the design space of diffusion-based generative models. *Advances in neural information processing systems*, 35:26565–26577, 2022. 2, 3, 5

[22] Angelos Katharopoulos, Apoorv Vyas, Nikolaos Pappas, and François Fleuret. Transformers are rnns: Fast autoregressive transformers with linear attention. In *International conference on machine learning*, pages 5156–5165. PMLR, 2020. 4

[23] Diederik P Kingma, Max Welling, et al. Auto-encoding variational bayes, 2013. 6

[24] Tamara G Kolda and Brett W Bader. Tensor decompositions and applications. *SIAM review*, 51(3):455–500, 2009. 4

[25] P Kwarteng and A Chavez. Extracting spectral contrast in landsat thematic mapper image data using selective principal component analysis. *Photogramm. Eng. Remote Sens*, 55(1): 339–348, 1989. 1

- [26] Xin Li, Yulin Ren, Xin Jin, Cuiling Lan, Xingrui Wang, Wenjun Zeng, Xinchao Wang, and Zhibo Chen. Diffusion models for image restoration and enhancement: a comprehensive survey. *International Journal of Computer Vision*, pages 1–31, 2025. 3
- [27] Yixun Liang, Ping Zhang, Yang Mei, and Tingqi Wang. Pmacnet: Parallel multiscale attention constraint network for pansharpening. *IEEE Geoscience and Remote Sensing Letters*, 19:1–5, 2022. 2
- [28] Yaron Lipman, Ricky TQ Chen, Heli Ben-Hamu, Maximilian Nickel, and Matt Le. Flow matching for generative modeling. *arXiv preprint arXiv:2210.02747*, 2022. 3, 5
- [29] Jiawei Liu, Qiang Wang, Huijie Fan, Yinong Wang, Yandong Tang, and Liangqiong Qu. Residual denoising diffusion models. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 2773–2783, 2024. 6
- [30] Xingchao Liu, Chengyue Gong, and Qiang Liu. Flow straight and fast: Learning to generate and transfer data with rectified flow. *arXiv preprint arXiv:2209.03003*, 2022. 3
- [31] Ilya Loshchilov and Frank Hutter. Decoupled weight decay regularization. *arXiv preprint arXiv:1711.05101*, 2017. 6, 3
- [32] Cheng Lu, Yuhao Zhou, Fan Bao, Jianfei Chen, Chongxuan Li, and Jun Zhu. Dpm-solver: A fast ode solver for diffusion probabilistic model sampling in around 10 steps. *Advances in Neural Information Processing Systems*, 35:5775–5787, 2022. 5
- [33] Giuseppe Masi, Davide Cozzolino, Luisa Verdoliva, and Giuseppe Scarpa. Pansharpening by convolutional neural networks. *Remote Sensing*, 8(7):594, 2016. 2, 7, 8, 3, 4
- [34] Qingyan Meng, Wenxu Shi, Sijia Li, and Linlin Zhang. Pandiff: A novel pansharpening method based on denoising diffusion probabilistic model. *IEEE Transactions on Geoscience and Remote Sensing*, 61:1–17, 2023. 2, 3, 7, 8, 4
- [35] Xiangchao Meng, Jie Li, Huanfeng Shen, Liangpei Zhang, and Hongyan Zhang. Pansharpening with a guided filter based on three-layer decomposition. *Sensors*, 16(7):1068, 2016. 1
- [36] Alexander Quinn Nichol and Prafulla Dhariwal. Improved denoising diffusion probabilistic models. In *International conference on machine learning*, pages 8162–8171. PMLR, 2021. 3
- [37] Xavier Otazu, María González-Audícana, Octavi Fors, and Jorge Núñez. Introduction of sensor spectral response into image fusion methods. application to wavelet-based methods. *IEEE Transactions on Geoscience and Remote Sensing*, 43(10):2376–2385, 2005. 1
- [38] Siran Peng, Liang-Jian Deng, Jin-Fan Hu, and Yu-Wei Zhuo. Source-adaptive discriminative kernels based network for remote sensing pansharpening. In *IJCAI*, pages 1283–1289, 2022. 2
- [39] Robin Rombach, Andreas Blattmann, Dominik Lorenz, Patrick Esser, and Björn Ommer. High-resolution image synthesis with latent diffusion models. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 10684–10695, 2022. 2, 3
- [40] Olaf Ronneberger, Philipp Fischer, and Thomas Brox. U-net: Convolutional networks for biomedical image segmentation. In *Medical image computing and computer-assisted intervention—MICCAI 2015: 18th international conference, Munich, Germany, October 5-9, 2015, proceedings, part III 18*, pages 234–241. Springer, 2015. 5, 1, 3
- [41] Xiangyu Rui, Xiangyong Cao, Li Pang, Zeyu Zhu, Zongsheng Yue, and Deyu Meng. Unsupervised hyperspectral pansharpening via low-rank diffusion model. *Information Fusion*, 107:102325, 2024. 3, 7, 8, 2, 4
- [42] Chitwan Saharia, Jonathan Ho, William Chan, Tim Salimans, David J Fleet, and Mohammad Norouzi. Image super-resolution via iterative refinement. *IEEE transactions on pattern analysis and machine intelligence*, 45(4):4713–4726, 2022. 2, 3
- [43] Neta Shaul, Juan Perez, Ricky TQ Chen, Ali Thabet, Albert Pumarola, and Yaron Lipman. Bespoke solvers for generative flow models. *arXiv preprint arXiv:2310.19075*, 2023. 3
- [44] Zhuoran Shen, Mingyuan Zhang, Haiyu Zhao, Shuai Yi, and Hongsheng Li. Efficient attention: Attention with linear complexities. In *Proceedings of the IEEE/CVF winter conference on applications of computer vision*, pages 3531–3539, 2021. 4
- [45] Jiaming Song, Chenlin Meng, and Stefano Ermon. Denoising diffusion implicit models. *arXiv preprint arXiv:2010.02502*, 2020. 2, 3, 5, 6
- [46] Yang Song, Jascha Sohl-Dickstein, Diederik P Kingma, Abhishek Kumar, Stefano Ermon, and Ben Poole. Score-based generative modeling through stochastic differential equations. *arXiv preprint arXiv:2011.13456*, 2020. 3
- [47] Yang Song, Prafulla Dhariwal, Mark Chen, and Ilya Sutskever. Consistency models. 2023. 3
- [48] Jiangtong Tan, Jie Huang, Naishan Zheng, Man Zhou, Keyu Yan, Danfeng Hong, and Feng Zhao. Revisiting spatial-frequency information integration from a hierarchical perspective for panchromatic and multi-spectral image fusion. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 25922–25931, 2024. 3
- [49] Ledyard R Tucker. Some mathematical notes on three-mode factor analysis. *Psychometrika*, 31(3):279–311, 1966. 4
- [50] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. *Advances in neural information processing systems*, 30, 2017. 4
- [51] Gemine Vivone. Robust band-dependent spatial-detail approaches for panchromatic sharpening. *IEEE transactions on Geoscience and Remote Sensing*, 57(9):6421–6433, 2019. 7, 8, 2, 3, 4
- [52] Gemine Vivone, Rocco Restaino, and Jocelyn Chanussot. A regression-based high-pass modulation pansharpening approach. *IEEE Transactions on geoscience and remote sensing*, 56(2):984–996, 2017. 1
- [53] Gemine Vivone, Rocco Restaino, and Jocelyn Chanussot. Full scale regression-based injection coefficients for panchromatic sharpening. *IEEE Transactions on Image Processing*, 27(7):3418–3431, 2018. 7, 8, 2, 3, 4
- [54] Lucien Wald. *Data fusion: definitions and architectures: fusion of images of different spatial resolutions*. Presses des MINES, 2002. 6

- [55] Lucien Wald, Thierry Ranchin, and Marc Mangolini. Fusion of satellite images of different spatial resolutions: Assessing the quality of resulting images. *Photogrammetric engineering and remote sensing*, 63(6):691–699, 1997. 6
- [56] Jianyi Wang, Zongsheng Yue, Shangchen Zhou, Kelvin CK Chan, and Chen Change Loy. Exploiting diffusion prior for real-world image super-resolution. *International Journal of Computer Vision*, 132(12):5929–5949, 2024. 3
- [57] Kai Wang, Dongwen Tang, Boya Zeng, Yida Yin, Zhaopan Xu, Yukun Zhou, Zelin Zang, Trevor Darrell, Zhuang Liu, and Yang You. Neural network diffusion. *arXiv preprint arXiv:2402.13144*, 2024. 3
- [58] Yancong Wei, Qiangqiang Yuan, Xiangchao Meng, Huanfeng Shen, Liangpei Zhang, and Michael Ng. Multi-scale-and-depth convolutional neural network for remote sensed imagery pan-sharpening. In *2017 IEEE International Geoscience and Remote Sensing Symposium (IGARSS)*, pages 3413–3416. IEEE, 2017. 7, 8, 2, 3, 4
- [59] Chen Wu, Bo Du, Xiaohui Cui, and Liangpei Zhang. A post-classification change detection method based on iterative slow feature analysis and bayesian soft fusion. *Remote Sensing of Environment*, 199:241–255, 2017. 1
- [60] Xiao Wu, Ting-Zhu Huang, Liang-Jian Deng, and Tian-Jing Zhang. Dynamic cross feature fusion for remote sensing pan-sharpening. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 14687–14696, 2021. 2, 7, 8, 3, 4
- [61] Zhong-Cheng Wu, Ting-Zhu Huang, Liang-Jian Deng, Jie Huang, Jocelyn Chanussot, and Gemine Vivone. Lrtcfpan: Low-rank tensor completion based framework for pansharpening. *IEEE Transactions on Image Processing*, 32:1640–1655, 2023. 1
- [62] Zhong-Cheng Wu, Ting-Zhu Huang, Liang-Jian Deng, and Gemine Vivone. A framelet sparse reconstruction method for pansharpening with guaranteed convergence. *Inverse Problems and Imaging*, 17(6):1277–1300, 2023. 1
- [63] Bin Xia, Yulun Zhang, Shiyin Wang, Yitong Wang, Xinglong Wu, Yapeng Tian, Wenming Yang, and Luc Van Gool. Diffir: Efficient diffusion model for image restoration. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 13095–13105, 2023. 2, 3, 6
- [64] Jin-Liang Xiao, Ting-Zhu Huang, Liang-Jian Deng, Guang Lin, Zihan Cao, Chao Li, and Qibin Zhao. Hyperspectral pansharpening via diffusion models with iteratively zero-shot guidance. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 12669–12678, 2025. 3
- [65] Junfeng Yang, Xueyang Fu, Yuwen Hu, Yue Huang, Xinghao Ding, and John Paisley. Pannet: A deep network architecture for pan-sharpening. In *Proceedings of the IEEE international conference on computer vision*, pages 5449–5457, 2017. 2
- [66] Wenming Yang, Xuechen Zhang, Yapeng Tian, Wei Wang, Jing-Hao Xue, and Qingmin Liao. Deep learning for single image super-resolution: A brief review. *IEEE Transactions on Multimedia*, 21(12):3106–3121, 2019. 5
- [67] Xiaohui Yuan, Jianfang Shi, and Lichuan Gu. A review of deep learning methods for semantic segmentation of remote sensing imagery. *Expert Systems with Applications*, 169:114417, 2021. 1
- [68] Zongsheng Yue, Jianyi Wang, and Chen Change Loy. Resshift: Efficient diffusion model for image super-resolution by residual shifting. *Advances in Neural Information Processing Systems*, 36:13294–13307, 2023. 5
- [69] Kaiwen Zheng, Cheng Lu, Jianfei Chen, and Jun Zhu. Dpm-solver-v3: Improved diffusion ode solver with empirical model statistics. *Advances in Neural Information Processing Systems*, 36:55502–55542, 2023. 3, 5
- [70] Jie Zhou, Daniel L Civco, and John A Silander. A wavelet transform method to merge landsat tm and spot panchromatic data. *International journal of remote sensing*, 19(4):743–757, 1998. 6
- [71] Linqi Zhou, Aaron Lou, Samar Khanna, and Stefano Ermon. Denoising diffusion bridge models. *arXiv preprint arXiv:2309.16948*, 2023. 3
- [72] Man Zhou, Jie Huang, Yanchi Fang, Xueyang Fu, and Aiping Liu. Pan-sharpening with customized transformer and invertible neural network. In *Proceedings of the AAAI conference on artificial intelligence*, pages 3553–3561, 2022. 2, 7, 8, 3, 4
- [73] Man Zhou, Keyu Yan, Jinshan Pan, Wenqi Ren, Qi Xie, and Xiangyong Cao. Memory-augmented deep unfolding network for guided image super-resolution. *International Journal of Computer Vision*, 131(1):215–242, 2023. 7, 8, 2, 3, 4

Fast Kernel-Space Diffusion for Remote Sensing Pansharpening

Supplementary Material

Abstract

In this supplementary material, we first provide a more detailed explanation of specific components of our proposed method that could not be thoroughly described in the main paper, including the network we use in our main experiments and further information on latent encoders. We then elaborate on the experimental settings and implementation details. Finally, we present additional experimental results, including generalization assessment on WorldView-2 dataset, further visualizations on the WorldView-3, GaoFen-2, and QuickBird datasets, as well as ablation studies on more factors.

6. Methods Explanation

6.1. Pansharpening Network

The baseline network we build is demonstrated in Fig. 7. The panchromatic (PAN) image $\mathbf{P} \in \mathbb{R}^{H \times W \times 1}$ is first duplicated along the channel dimension to match the number of channels in the low-resolution multispectral (LRMS) image $\mathbf{M} \in \mathbb{R}^{H \times W \times C}$. The duplicated PAN image is then subtracted by the LRMS image, and the resulting difference is fed into the network. After processing, the network output is added to the input LRMS image to produce the reconstructed high-resolution multispectral image $\mathbf{H} \in \mathbb{R}^{H \times W \times C}$, denoted as $\mathbf{H}_1$ or $\mathbf{H}_2$ as defined in the main text. To reduce spatial resolution and increase channel depth, downsampling layers are inserted between successive encoder blocks in the network. Conversely, in the decoder, upsampling operations with a scale factor of 2 are applied between adjacent layers to increase spatial resolution and reduce the number of channels. The skip connections adopt a concatenation-based strategy, similar to that of the standard U-Net [40]. Each block in the backbone corresponds to a ResBlock [14]. Within each block, only the 3×3 convolution layer is reparameterized using the proposed KSDiff method. Given the multi-scale nature of the network, the latent representation is appropriately rescaled before being injected into each convolution layer. Specifically, shallower layers, which operate on higher spatial resolutions, use latent representations with larger scales, whereas deeper layers employ downsampled versions of the generated latent representation. Detailed parameter settings are provided in Sec. 7.

6.2. PLFE₂

As shown in Fig. 8 (a), the structure of PLFE₂ is a simplified, halved version of PLFE₁, which is presented in Fig. 3 in

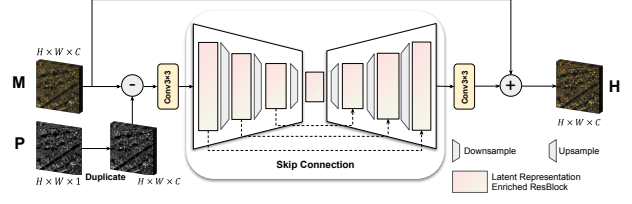


Figure 7. Baseline network.

the main text. All core components, including the cross-attention mechanism and the Fusion-Gate module, remain identical to those in PLFE₁. The role of PLFE₂ differs from PLFE₁. In PLFE₁, the module encodes the ground-truth, PAN, LRMS images to obtain a latent representation $\mathbf{z}_0 \in \mathbb{R}^{N \times C_z}$. In contrast, PLFE₂ produces a conditioning vector $\mathbf{c} \in \mathbb{R}^{N \times C_z}$ only with PAN and LRMS images as inputs, which is injected into the denoising network to guide the reverse diffusion process for latent reconstruction. The resulting latent representation is denoted as $\hat{\mathbf{z}}_0 \in \mathbb{R}^{N \times C_z}$, which is then fed into the kernel generator.

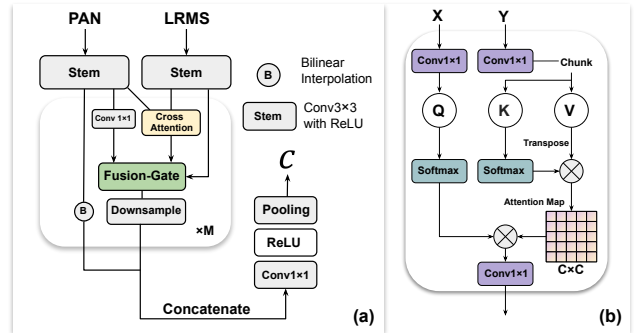


Figure 8. (a) PLFE₂. (b) The cross attention in latent encoders.

7. Details on Experiments

7.1. Datasets

We conducted experiments using datasets derived from WorldView-3 (WV3), QuickBird (QB), and GaoFen-2 (GF2) satellite imagery. These datasets consist of image patches cropped from full remote sensing scenes and are partitioned into training and testing subsets. The WV3 dataset contains four images from two geographic locations, Rio and Tripoli, captured in different seasons: Rio in January, Rio in May, Tripoli in March, and Tripoli in August. The right quarter of the Rio in May and Tripoli in August images was cropped to form the test set. The remaining areas of all images were

Table 7. Summary of compared pansharpening methods.

Method	Category	Year	Introduction
BDS-PC [51]	CS	2019	Discusses the constraints of the band-dependent spatial-detail (BDS-PC) approach in images with over four spectral bands.
MTF-GLP-FS [53]	MRA	2018	An iterative algorithm for estimating injection coefficients at full resolution in regression-based pansharpening.
BT-H [1]	MRA	2006	A multiresolution pansharpening framework using a generalized Laplacian pyramid (GLP) with an MTF-adjusted filter to enhance spatial details while preserving spectral information.
PNN [33]	ML	2016	The first deep CNN architectures designed specifically for pansharpening.
DiCNN [15]	ML	2019	Introduces a detail-injection mechanism to enhance spatial detail learning in CNNs.
MSDCNN [58]	ML	2017	Employs multiscale and densely connected convolutional layers to extract both spatial and spectral features.
FusionNet [8]	ML	2020	A residual learning-based network that emphasizes detail preservation for fusion tasks.
CTINN [72]	ML	2022	Use a customized transformer architecture and an information-lossless invertible neural module to model long-range dependencies.
LAGConv [20]	ML	2022	Employ local-context adaptive convolution kernels and global harmonic bias to enhance image fusion.
MMNet [73]	ML	2023	A model-driven deep unfolding network with memory-augmentation.
DCFNet [60]	ML	2021	A dynamic cross feature fusion network for pansharpening that integrates multi-scale branches and contextualized feature transfers.
PanMamba [16]	ML	2025	A state-space model based method for pansharpening.
PanDiff [34]	Diffusion-based ML	2023	Employs a diffusion model to learn HRMS-IMS difference map distributions, guided by PAN/LRMS images with a modal intercalibration module.
PLRDiff [41]	Diffusion-based ML	2024	Proposes a low-rank diffusion model for pansharpening, combining a pre-trained diffusion model for image structure capture with a Bayesian approach for spectral preservation.

used for training. This configuration avoids any spatial overlap and incorporates seasonal variation, thereby preventing both spatial and temporal leakage. For QB and GF2 datasets, each consists of a single image (QB Indianapolis and GF2 Guangzhou, respectively). The rightmost 1,000 pixels were cropped to form the test sets, while the rest of the image was used for training. Since the cropped test areas do not overlap with the training regions, spatial independence is maintained. Temporal leakage is not applicable in these cases, as only single-date imagery is available.

The training data comprise simulated downsampling-based PAN/LRMS/GT triplets, with dimensions of $64 \times 64 \times 1$, $64 \times 64 \times C$, and $64 \times 64 \times C$, respectively. Specifically, the WV3 training set includes approximately 10,000

eight-channel samples ($C = 8$), while the GF2 and QB training sets contain around 20,000 and 17,000 four-channel samples ($C = 4$), respectively. For evaluation, the reduced-resolution test set for each satellite consists of 20 simulated PAN/LRMS/GT triplets representing diverse land cover types, with dimensions of $256 \times 256 \times 1$, $256 \times 256 \times C$, and $256 \times 256 \times C$, respectively. Additionally, a full-resolution test set is provided, comprising 20 original PAN/LRMS image pairs with spatial sizes of 512×512 . All datasets and preprocessing procedures were obtained from the PanCollection repository [9].

7.2. Training Details

The proposed KSDiff framework is implemented using PyTorch 2.1.0 and Python 3.10. All experiments are conducted on a Linux system (Ubuntu 22.04) equipped with a single NVIDIA GeForce RTX 4090 GPU and CUDA version 12.1. We adopt the AdamW optimizer [31] with a learning rate of 0.8×10^{-4} and a weight decay of 1×10^{-4} for both the pre-training and diffusion training stages. The denoising diffusion process follows a cosine noise schedule [36], without learning the variance as proposed in [36]. The total number of diffusion timesteps is set to 500 across all datasets. An exponential moving average (EMA) with a decay rate of 0.995 is employed. For the pre-training stage, the number of training iterations is set to 60k for all datasets (WV3, GF2, and QB). In the diffusion model training stage, the iteration counts are set to 50k, 70k, and 10k for WV3, GF2, and QB, respectively. The batch size during pre-training is fixed at 64 for all datasets. For diffusion model training, batch sizes are set to 32 for WV3, and 16 for both GF2 and QB. For DDIM sampling [45], we use 25 sampling steps uniformly across the three datasets.

The denoising network consists of five MLP blocks utilizing GELU activation [17]. The conditioning of the diffusion model was implemented via feature concatenation [42]. The latent encoders PLFE_1 and PLFE_2 are designed with three pyramid stages. The latent representation $\mathbf{z}$ has dimensions $N = 16$ and $C_z = 128$. The pansharpening network employs a U-Net [40] architecture with four encoder and four decoder blocks. The encoder begins with 32 channels, which are scaled by factors of 1, 2, 2, and 4 across successive blocks. The dimension of the latent representation N is downsampled by the same factors before integration into kernel generation. For the kernel generator, the low-rank core tensor has dimensions (r_1, r_2, r_3, r_4) set to $(4, 4, 2, 2)$ for the first three blocks and $(8, 8, 2, 2)$ for the fourth block.

7.3. Compared Methods

Table 7 provides a brief overview of the pansharpening methods compared in the main text. We compare the proposed KSDiff with both traditional and recent machine learning (ML) approaches. Three traditional methods are considered: BDSD-PC [51], MTF-GLP-FS [53], and BT-H [1]. Nine machine learning methods without diffusion mechanisms are also included: PNN [33], DiCNN [15], MSDCNN [58], FusionNet [8], CTINN [72], LAGConv [20], MMNet [73], DCFNet [60], and PanMamba [16]. Additionally, we include two recent diffusion-based ML methods: PanDiff [34] and PLRDiff [41]. The runtimes of these methods (as reported in Tab. 1 in the main text) are evaluated and compared using the WV3 reduced-resolution dataset.

7.4. Ablation Study

Continuing from Tab. 4 in the main text, we provide the detailed implementation of our ablation studies. The experiment was implemented on the WV3 reduced-resolution dataset. To assess the impact of the latent diffusion prior, we trained the baseline U-Net independently for 100,000 iterations using an ℓ_1 loss function and a batch size of 64. For the ablation of our proposed PLFE module, we replaced the latent encoder with the one adopted in DiffIR [63]. In this setup, the PAN, LRMS, and GT images (for the pre-training stage), or the PAN and LRMS images (for the diffusion model training stage), were directly concatenated at the input of the encoder. The alternative encoder comprises six residual blocks. All ablation experiments were trained until full convergence. Other training configurations, including the optimizer, learning rate, and weight decay, were kept consistent with those used in the main experiments, as illustrated in Sec. 7.2.

7.5. Other Backbones

This section provides the implementation details of the experiments discussed in Sec. 4.4 in the main text, where we investigate the effect of replacing standard convolutional operations with our proposed KSDiff kernel generation pipeline in other backbone networks. The experiment was implemented on the WV3 reduced-resolution dataset. For the DiCNN model [15], which comprises a three-layer convolutional neural network with an inner channel dimension of 64, we modulated the middle convolution layer using the KSDiff mechanism. In the case of FusionNet [8], which contains four standard ResBlocks [14] with 32 inner channels each, we replaced the standard convolutional layers within the ResBlocks with their KSDiff-enhanced counterparts. For LAGNet [20], which already incorporates adaptive convolution kernels, we substituted the kernels in its adaptive convolution modules with diffusion-prior-enhanced versions generated by KSDiff. For all three models, the batch size was consistently set to 64 for both the pre-training and diffusion model training stages. The size of the latent representation $\mathbf{z}$ is $N = 16$, $C_z = 32$. The low-rank core tensor size (r_1, r_2, r_3, r_4) was fixed at $(4, 4, 2, 2)$, and the learning rate was set to 1×10^{-4} . All experiments were trained until full convergence.

7.6. Core Tensor Size

This section provides the detailed configuration of the experiments investigating the impact of low-rank core tensor sizes, as discussed in Sec. 4.4 in the main text. The experiment was implemented on the WV3 reduced-resolution dataset. The inner channel dimension of FusionNet [8] is fixed at 32, and all convolutional kernels have a size of $k = 3$, making it a suitable and efficient setting for controlled comparative experiments. All configurations presented in the main text

were trained for 100,000 iterations in both the pre-training stage and the diffusion model training stage, with a batch size of 64 and a learning rate of 1×10^{-4} . The experimental results highlight the advantageous properties of leveraging low-rank representations and the high-dimensional structure of 4D tensors. We leave the exploration of these properties across a broader range of tasks, architectures, and parameter settings as a direction for future work.

Table 8. Result on the GF2 full-resolution dataset. The best results are highlighted in bold and the second best results are underlined.

Method	GaoFen-2		
	D_λ ($\pm$ std)	D_s ($\pm$ std)	HQNR ($\pm$ std)
BDS-PC [51]	0.0759 $\pm$ 0.0067	0.1548 $\pm$ 0.0063	0.7812 $\pm$ 0.0091
MTF-GLP-FS [53]	0.0336 $\pm$ 0.0029	0.1404 $\pm$ 0.0062	0.8309 $\pm$ 0.0075
BT-H [1]	0.0602 $\pm$ 0.0043	0.1313 $\pm$ 0.0043	0.8165 $\pm$ 0.0068
PNN [33]	0.0367 $\pm$ 0.0065	0.0943 $\pm$ 0.0050	0.8726 $\pm$ 0.0083
DiCNN [15]	0.0413 $\pm$ 0.0029	0.0992 $\pm$ 0.0029	0.8636 $\pm$ 0.0037
MSDCNN [58]	0.0269 $\pm$ 0.0029	0.0730 $\pm$ 0.0021	0.9020 $\pm$ 0.0029
FusionNet [8]	0.0400 $\pm$ 0.0028	0.1013 $\pm$ 0.0030	0.8628 $\pm$ 0.0041
CTINN [72]	0.0586 $\pm$ 0.0058	0.1996 $\pm$ 0.0033	0.8381 $\pm$ 0.0053
LAGConv [20]	0.0324 $\pm$ 0.0029	0.0792 $\pm$ 0.0030	0.8910 $\pm$ 0.0046
MMNet [73]	0.0428 $\pm$ 0.0067	0.1033 $\pm$ 0.0029	0.8583 $\pm$ 0.0060
DCFNet [60]	0.0234 $\pm$ 0.0026	0.0659 $\pm$ 0.0021	0.9122 $\pm$ 0.0027
PanMamba [16]	0.0231$\pm$0.0025	0.0572$\pm$0.0023	<u>0.9210$\pm$0.0028</u>
PanDiff [34]	0.0265 $\pm$ 0.0044	0.0729 $\pm$ 0.0023	0.9025 $\pm$ 0.0047
PLRDiff [41]	0.2804 $\pm$ 0.0292	0.1413 $\pm$ 0.0057	0.6164 $\pm$ 0.0234
KSDiff (ours)	<u>0.0233$\pm$0.0028</u>	<u>0.0588$\pm$0.0023</u>	0.9257$\pm$0.0024
Ideal value	0	0	1

Table 9. Result on the QB full-resolution dataset. The best results are highlighted in bold and the second best results are underlined.

Method	QuickBird		
	D_λ ($\pm$ std)	D_s ($\pm$ std)	HQNR ($\pm$ std)
BDS-PC [51]	0.1975 $\pm$ 0.0075	0.1636 $\pm$ 0.0108	0.6722 $\pm$ 0.0129
MTF-GLP-FS [53]	0.0489 $\pm$ 0.0033	0.1383 $\pm$ 0.0053	0.8199 $\pm$ 0.0076
BT-H [1]	0.2300 $\pm$ 0.0161	0.1648 $\pm$ 0.0037	0.6434 $\pm$ 0.0144
PNN [33]	0.0569 $\pm$ 0.0025	0.0624 $\pm$ 0.0053	0.8844 $\pm$ 0.0068
DiCNN [15]	0.0920 $\pm$ 0.0032	0.1067 $\pm$ 0.0047	0.8114 $\pm$ 0.0069
MSDCNN [58]	0.0602 $\pm$ 0.0034	0.0667 $\pm$ 0.0065	0.8774 $\pm$ 0.0087
FusionNet [8]	0.0586 $\pm$ 0.0042	0.0522 $\pm$ 0.0020	0.8922 $\pm$ 0.0049
CTINN [72]	0.1738 $\pm$ 0.0074	0.0731 $\pm$ 0.0053	0.7663 $\pm$ 0.0097
LAGConv [20]	0.0844 $\pm$ 0.0053	0.0676 $\pm$ 0.0030	0.8536 $\pm$ 0.0040
MMNet [73]	0.0890 $\pm$ 0.0114	0.0972 $\pm$ 0.0085	0.8225 $\pm$ 0.0071
DCFNet [60]	<u>0.0454$\pm$0.0033</u>	0.1239 $\pm$ 0.0060	0.8360 $\pm$ 0.0035
PanMamba [16]	0.0491 $\pm$ 0.0028	<u>0.0443$\pm$0.0033</u>	<u>0.9102$\pm$0.0029</u>
PanDiff [34]	0.0587 $\pm$ 0.0039	0.0642 $\pm$ 0.0056	0.8813 $\pm$ 0.0093
PLRDiff [41]	0.7775 $\pm$ 0.0233	0.3038 $\pm$ 0.0204	0.1615 $\pm$ 0.0200
KSDiff (ours)	0.0380$\pm$0.0027	0.0426$\pm$0.0029	0.9130$\pm$0.0026
Ideal value	0	0	1

8. Additional Results

8.1. Main Results

Tab. 8 and Tab. 9 present the quantitative performance benchmarks on the full-resolution GF2 and QB datasets. The results indicate that the proposed KSDiff method exhibits strong generalization capabilities across different data domains. Fig. 9 to Fig. 14 provide qualitative comparisons

of visual outputs generated by various methods on representative samples from the WV3, GF2, and QB datasets. For the reduced-resolution data, corresponding error maps between the predicted outputs and ground-truth references are also included. The visual and quantitative comparisons consistently demonstrate that KSDiff produces results that are highly consistent with the ground-truth. Furthermore, the method shows robust performance in scenarios lacking reference images, attributed to the diffusion model’s powerful distribution estimation capability.

Table 10. Generalization of DL-based methods on WV2 dataset.

Method	WorldView-2			
	SAM ($\pm$ std)	ERGAS ($\pm$ std)	Q2 ⁿ ($\pm$ std)	SCC ($\pm$ std)
PNN [33]	7.1158 $\pm$ 0.3758	5.6152 $\pm$ 0.2108	0.7619 $\pm$ 0.0207	0.8782 $\pm$ 0.0039
DiCNN [15]	6.9216 $\pm$ 0.1766	6.2507 $\pm$ 0.1285	0.7205 $\pm$ 0.0167	0.8552 $\pm$ 0.0065
MSDCNN [58]	6.0064 $\pm$ 0.1425	4.7438 $\pm$ 0.1104	0.8241 $\pm$ 0.0179	0.8972 $\pm$ 0.0024
FusionNet [8]	6.4257 $\pm$ 0.1923	5.1363 $\pm$ 0.1152	0.7961 $\pm$ 0.0165	0.8746 $\pm$ 0.0030
CTINN [72]	6.4103 $\pm$ 0.1331	4.6435 $\pm$ 0.0847	0.8172 $\pm$ 0.0195	0.9147 $\pm$ 0.0023
LAGConv [20]	6.9545 $\pm$ 0.1059	5.3262 $\pm$ 0.0712	0.8054 $\pm$ 0.0187	0.9125 $\pm$ 0.0023
MMNet [73]	6.6109 $\pm$ 0.0717	5.2213 $\pm$ 0.0477	0.8143 $\pm$ 0.0177	0.9136 $\pm$ 0.0045
DCFNet [60]	5.6194 $\pm$ 0.1350	4.4887 $\pm$ 0.0841	0.8292 $\pm$ 0.0182	0.9154 $\pm$ 0.0019
KSDiff (ours)	5.1944$\pm$0.1197	4.1052$\pm$0.0796	0.8485$\pm$0.0185	0.9288$\pm$0.0017
Ideal value	0	0	1	1

Table 11. Ablation study on the trade-off hyperparameter λ in the loss function of the diffusion model training stage.

λ	WorldView-3			
	SAM $\downarrow$ ($\pm$ std)	ERGAS $\downarrow$ ($\pm$ std)	Qn $\uparrow$ ($\pm$ std)	SCC $\uparrow$ ($\pm$ std)
0.1	2.8232 $\pm$ 0.1167	2.0858 $\pm$ 0.1021	0.9167 $\pm$ 0.0186	0.9870 $\pm$ 0.0010
0.2	2.8159 $\pm$ 0.1155	2.0801 $\pm$ 0.1006	0.9168 $\pm$ 0.0188	0.9871 $\pm$ 0.0009
1	2.8102 $\pm$ 0.1147	2.0756 $\pm$ 0.0973	0.9221 $\pm$ 0.0183	0.9870 $\pm$ 0.0010
5	2.8078 $\pm$ 0.1147	2.0790 $\pm$ 0.1015	0.9203 $\pm$ 0.0185	0.9870 $\pm$ 0.0010
10	2.8106 $\pm$ 0.1149	2.0877 $\pm$ 0.1016	0.9189 $\pm$ 0.0190	0.9869 $\pm$ 0.0010
Ideal value	0	0	1	1

8.2. Generalization

To assess the generalization capability of deep learning-based methods, we evaluated models trained on the WV3 dataset using 20 reduced resolutions from the WorldView-2 dataset. As shown in Tab. 10, the quantitative results reveal that the KSDiff method consistently outperforms others across all four evaluation metrics, demonstrating the strong generalization ability of our approach.

8.3. Effect of Hyperparameter λ

For the total loss design in the diffusion model training stage, $\mathcal{L} = \mathcal{L}_{\text{diff}} + \lambda \mathcal{L}_{\text{reg}}$, we initially set $\lambda = 1$, which was found to be empirically effective. Throughout our experiments, no issues such as gradient collapse were observed. To further examine the influence of λ , we conducted an ablation study in which all models were trained until full convergence. The results, summarized in Tab. 11, demonstrate that the overall performance remains relatively consistent across a range of λ values, indicating that our method is not particularly sensitive to this hyperparameter.

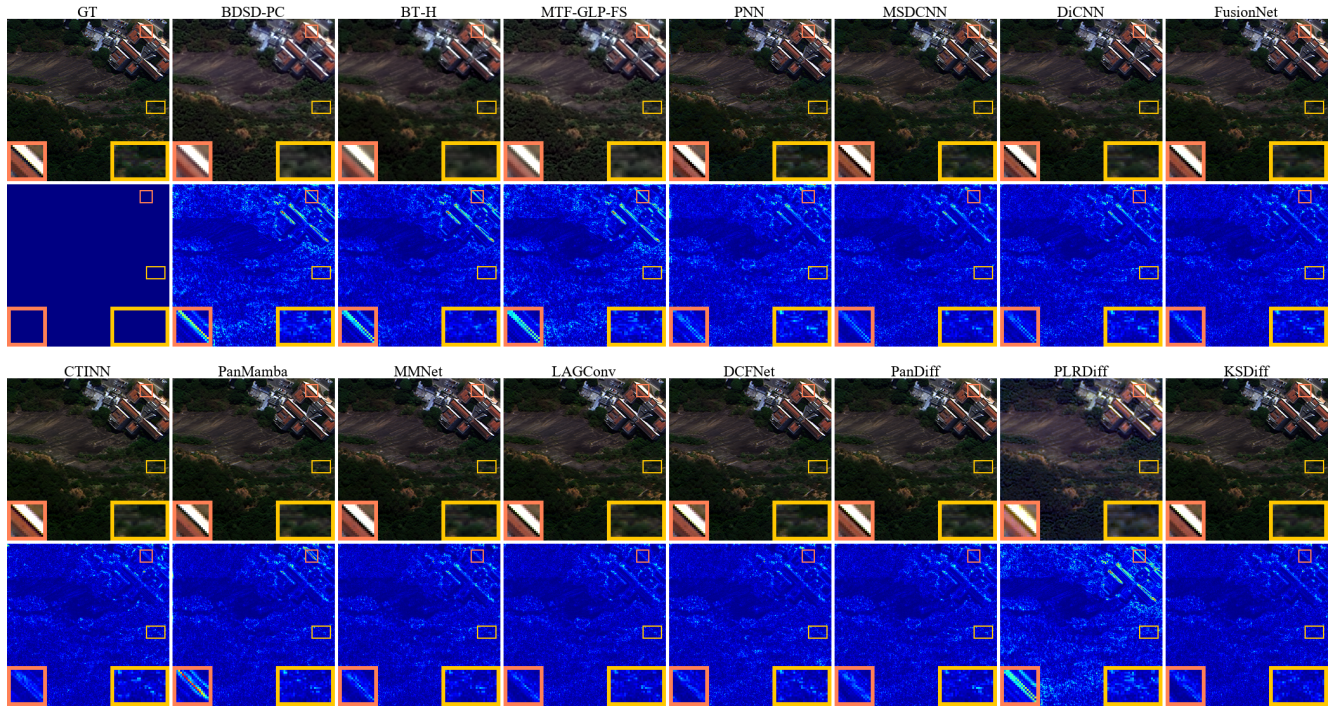


Figure 9. Comparison of qualitative results for representative methods on the WV3 reduced-resolution dataset.

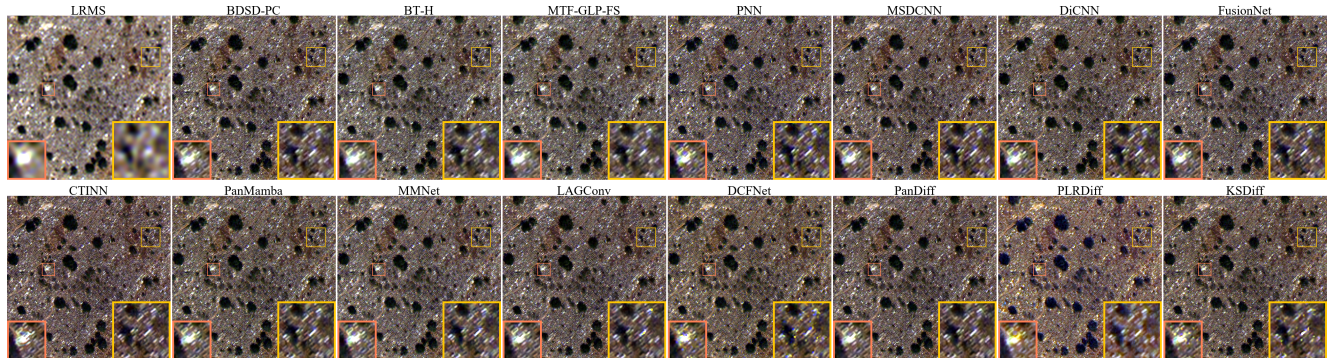


Figure 10. Comparison of qualitative results for representative methods on the WV3 full-resolution dataset.

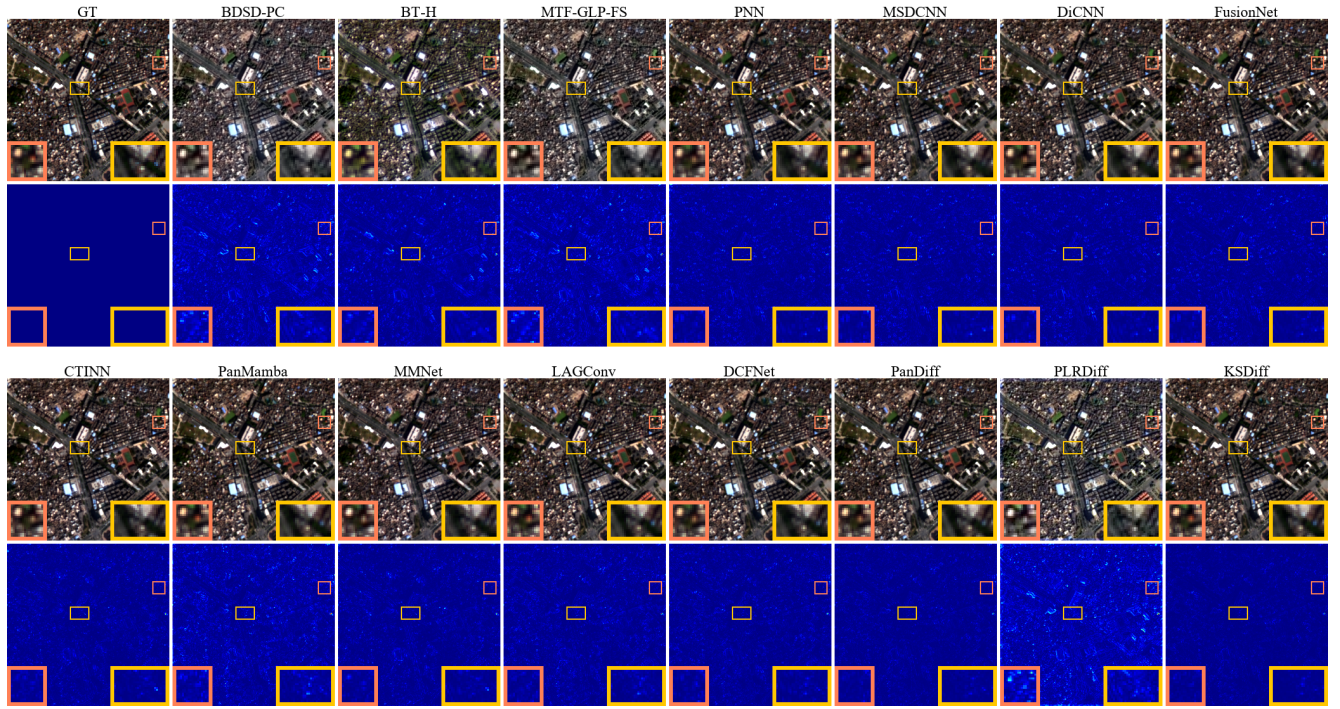


Figure 11. Comparison of qualitative results for representative methods on the GF2 reduced-resolution dataset.



Figure 12. Comparison of qualitative results for representative methods on the GF2 full-resolution dataset.

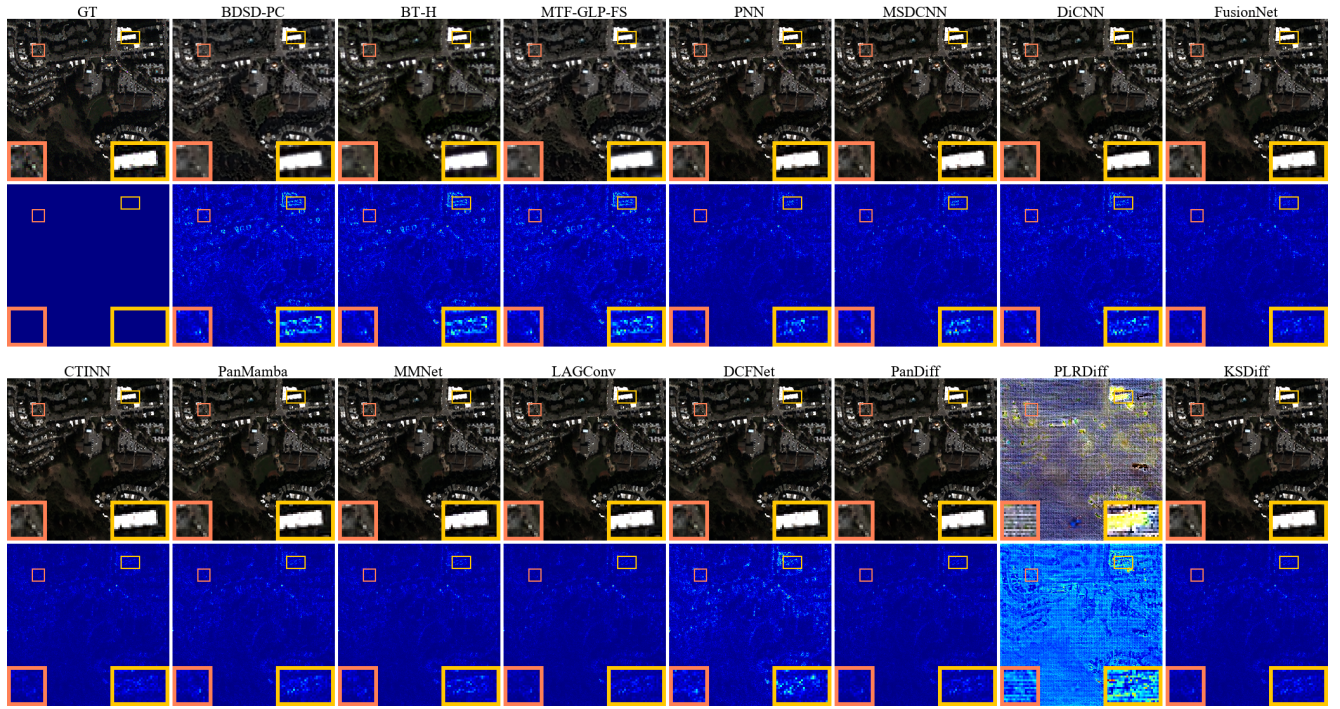


Figure 13. Comparison of qualitative results for representative methods on the QB reduced-resolution dataset.

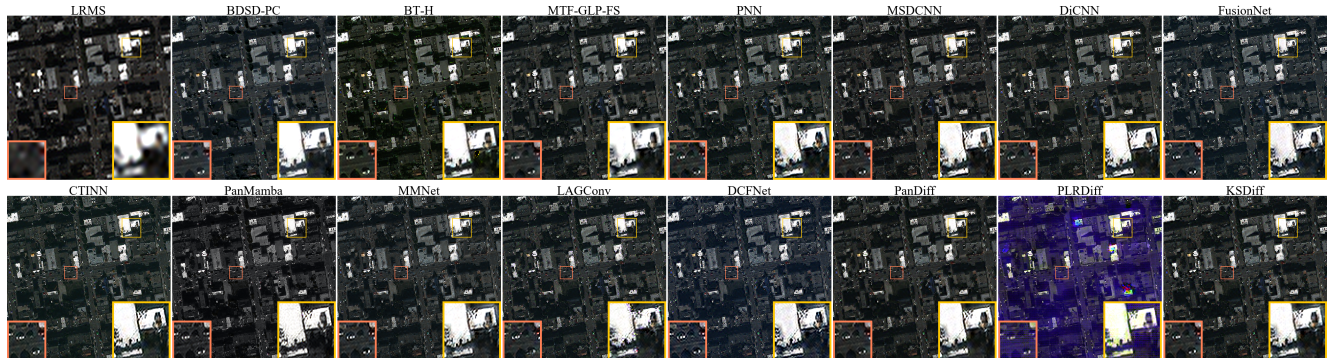


Figure 14. Comparison of qualitative results for representative methods on the QB full-resolution dataset.