Neural Super-Resolution for Real-time Rendering with Radiance Demodulation

Jia Li[1], Ziling Chen[1], Xiaolong Wu[1], Lu Wang[1,*], Beibei Wang[2,3,*], Lei Zhang[4]

[1]Shandong University, [2]State Key Laboratory for Novel Software Technology, Nanjing University,

[3]School of Intelligence Science and Technology, Nanjing University,

[4]The Hong Kong Polytechnic University

luwang_hcivr@sdu.edu.cn, beibei.wang@nju.edu.cn

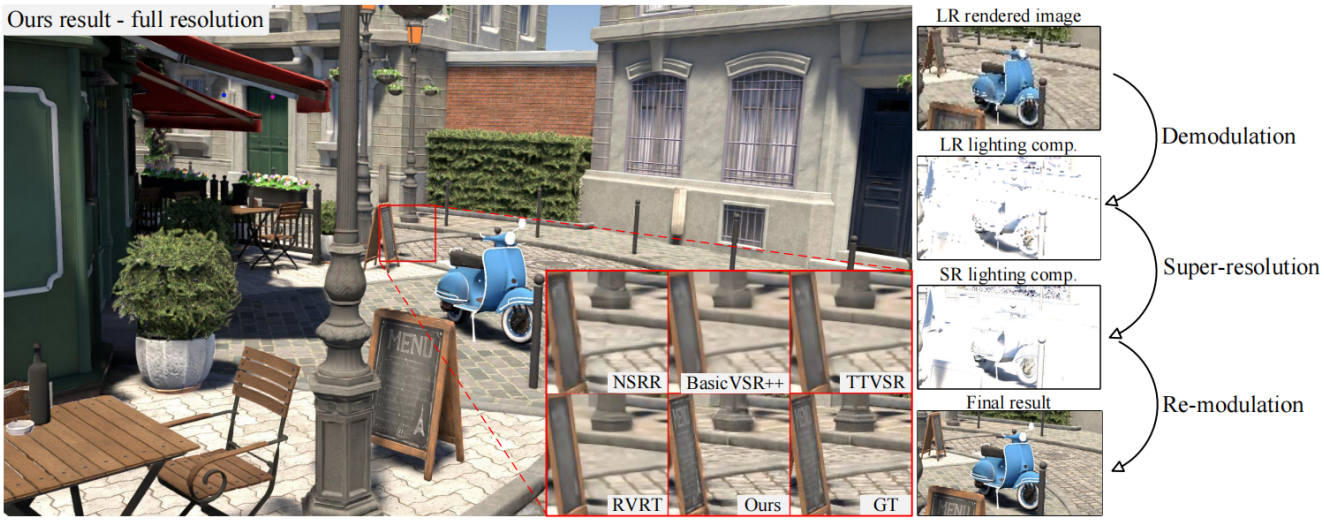


Figure 1. Quality comparison between our method and the state-of-the-art methods NSRR [53], BasicVSR++ [5], TTVSR [32] and RVRT [29]. The upsampling ratio is set as 4×4. By radiance demodulation, we perform super-resolution on the smooth lighting component only, allowing our method to preserve richer scene details after re-modulation with the high-resolution material component.

Abstract

It is time-consuming to render high-resolution images in applications such as video games and virtual reality, and thus super-resolution technologies become increasingly popular for real-time rendering. However, it is challenging to preserve sharp texture details, keep the temporal stability and avoid the ghosting artifacts in real-time super-resolution rendering. To address this issue, we introduce radiance demodulation to separate the rendered image or radiance into a lighting component and a material component, considering the fact that the light component is smoother than the rendered image so that the high-resolution material component with detailed textures can be easily obtained. We perform the super-resolution on the lighting component only and re-modulate it with the high-resolution material component to obtain the final super-resolution image with more texture details. A reliable warping module is proposed by explicitly marking the occluded regions to avoid the ghosting artifacts. To further enhance the temporal stability, we design a frame-recurrent neural network and a temporal loss to aggregate the previous and current frames, which can better capture the spatial-temporal consistency among reconstructed frames. As a result, our method is able to produce temporally stable results in real-time rendering with high-quality details, even in the challenging 4×4 super-resolution scenarios. Code is available at: https://github.com/Riga2/NSRD.

1. Introduction

Real-time rendering is widely used in various applications, like video games and virtual reality, where both low time cost and high resolution images are desired. To achieve such a goal, super-resolution (SR) rendering technologies have been popularly adopted by first rendering a low-resolution (LR) image and then performing SR on it. However, SR for real-time rendering is challenging since it needs to meet several requirements: detail-preserving, temporally stable, artifacts-free, and highly efficient.

*Corresponding author.

Many methods have been developed for a relevant task – video super-resolution (VSR) [4–6, 18, 20, 28, 29, 32, 49]. These methods have shown impressive results but can not be used in real-time rendering super-resolution (RRSR). First, most of them rely on a heavy network for optical flow estimation [38], resulting in an expensive time cost. Furthermore, these methods (e.g., [4, 5, 20, 28, 40, 49]) require both the precedent and the subsequent frames for bidirectional propagation. Unfortunately, only the precedent frames are available, since SR is performed simultaneously with rendering.

Another line of works [8, 12, 35, 44, 53] focus on RRSR. These methods exploit auxiliary buffers (e.g., depth buffer) to aid SR and can achieve real-time speed. However, due to the limited texture information in the LR image, even with these auxiliary buffers as inputs, the network can not recover the missing details, leading to blurry results. Furthermore, these methods have ghosting artifacts, when the motion vector [45] becomes unreliable for occluded regions in dynamic scenes. These artifacts can be alleviated by predicting regions with networks [12, 53], but still exist.

In this paper, we resolve these two issues with simple solutions. First, we introduce radiance demodulation into SR, together with a formulation to enable non-diffuse materials, inspired by real-time rendering denoising [62]. Two key observations that make radiance demodulation feasible in SR are: 1) the rendered image (radiance) can be demodulated into a material component with rich texture details and a relatively smooth lighting component compared to the radiance; 2) the material component can be captured quickly, even capturing a high-resolution (HR) one directly. More specifically, we design a demodulation module that separates the lighting component from the rendered image and performs SR on the smooth lighting component only, while the HR material component is used directly for remodulation with the SR results. In this way, we can obtain results with rich texture details. To our knowledge, we are the first to use radiance demodulation in RRSR. Second, we propose a simple way to get an occlusion-aware motion mask by subtracting two types of motion vectors, which can explicitly and accurately characterize the unreliable region for the network. Hence, the ghosting artifacts can be avoided. Finally, we design a lightweight frame-recurrent neural network using a convolutional long short-term memory (ConvLSTM) module, together with a temporal loss, which fully utilizes the intermediate features between adjacent frames to enhance the reconstruction quality and temporal stability further. Our method is specialized for real-time rendering and significantly outperforms prior work, including state-of-the-art RRSR and VSR methods, both visually and quantitatively.

To summarize, our main contributions include:

- we introduce radiance demodulation into super-resolution for non-diffuse materials to better rich texture details,
- we propose a simple way to get an occlusion-aware motion mask, which avoids the ghosting artifacts in dynamic scenes, and
- we design a lightweight frame-recurrent neural network for real-time SR, together with a temporal loss, to improve the reconstruction quality and the temporal stability.

2. Related Works

Video Super-Resolution. Existing VSR methods can be roughly categorized into sliding window-based [3, 17, 26, 47, 49] and recurrent-based [4, 5, 9, 13, 14, 16, 19, 31, 40]. The sliding window-based methods often take a short sequence of low-resolution frames as input without considering the correlation between reconstructed frames. By introducing the previously reconstructed images, recurrent-based methods have better temporal coherence. Huang et al. [15] are the first to introduce a recurrent neural network in VSR by connecting hidden layers between adjacent frames with recurrent convolutions. Later, Sajjadi et al. [40] propose to warp the previously reconstructed frame into the network, which is further improved by Chan et al. [4, 5] with enhanced propagation and alignment. Recently, transformer-based structures have been introduced into VSR problems (Liang et al. [28, 29]). And Liu et al. [32] propose a trajectory-aware transformer to learn spatio-temporal information more effectively. All of them can achieve good results, at the cost of large parameters and expensive time cost.

Frame alignment is an important operation in VSR. Most previous methods [3, 22, 30] utilize optical flow to warp the previous frames. Another group of methods uses 3D convolution [20] or non-local methods [56] to extract spatial-temporal information, without performing explicit frame alignment. Thus, they all need to design a more complex network for effective feature extraction.

VSR has shown impressive results but can not be exploited for real-time rendering. However, some networks indeed inspire the design of our method, like the recurrent framework.

Super-Resolution in Realtime Rendering. Existing methods for RRSR mainly include the traditional ones [8, 44] and deep-learning-based approaches [7, 11, 52, 53]. Temporal antialiasing upscaling (TAAU) [44] simply uses temporal accumulation to perform SR. Edelsten et al. [7] propose deep learning supersampling (DLSS), considering both temporal and spatial information. However, their method relies on NVIDIA's hardware platform, and there is no publicly available technical information. Unlike DLSS, FSR [8] does not rely on specific hardware platforms, and it generally achieves SR through upsampling and edge sharpening, with limited quality improvement, which is fur-

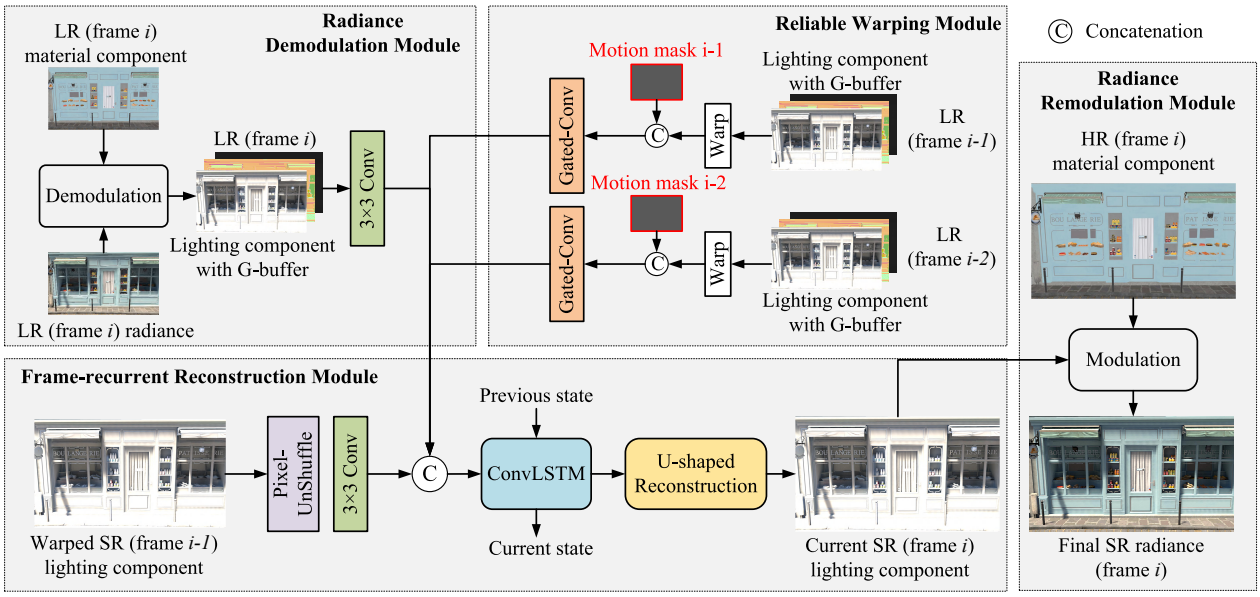


Figure 2. Our network includes four modules. *Radiance demodulation:* together with the material component, the LR rendered image (radiance) is demodulated to a lighting component for spatial feature extraction. *Reliable warping:* the warped lighting components of two previous frames and motion masks are fed into a gated convolution for temporal feature extraction. *Frame-recurrent reconstruction:* features from the previously reconstructed SR lighting component and other features are fed into a ConvLSTM followed by a U-shaped module to reconstruct the SR lighting component, which is later re-modulated with the HR material component to obtain the SR image.

ther improved by integrating TAAU, i.e., FSR 2.0. Intel's XeSS [52] is also a deep-learning-based SR method for rendered images, which targets production, and only demo samples are available.

Xiao et al. [53] propose NSRR, using the UNet [39] for SR reconstruction after zero-upsampling the features of multiple frames. However, NSRR suffers from ghosting artifacts in scenes with fast-moving objects or cameras. Gu et al. [11] propose to extract edge features for SR interpolation but without considering temporal stability. Guo et al. [12] train two networks separately to classify pixels into different categories, and then predict weights for frame blending. Yang et al. [55] use an alternate sub-pixel sample pattern during rasterization to create a small SR model that can be run on mobile devices. Mercier et al. [35] propose a lightweight recurrent network and a gaming dataset for RRSR. All of these methods can achieve real-time performance, but they are unable to restore complex texture details accurately. The concurrent work FuseSR [61] utilizes many HR auxiliary buffers and pre-integrated demodulation to improve the texture details, but its lack of design for temporal stability tends to cause flickering results.

Similar to the above methods, our method also targets real-time performance. However, we couple the rendering and SR to fully utilize the auxiliary buffer to compensate for the missing texture details, and employ a frame-recurrent design for temporal consistency between adjacent frames. Consequently, we achieve better reconstruction quality and temporal stability.

3. Method

Our method consists of four modules: a radiance demodulation module which extracts the spatial features coupled with a remodulation module (Section 3.1), a reliable warping module (Section 3.2) which extracts temporal features from the previous frame and a frame-recurrent reconstruction module (Section 3.3) which extracts features from the previously reconstructed images and performs the reconstruction as well as upsampling on all the concatenated features. The overview structure is shown in Figure 2.

3.1. Radiance Demodulation

Most existing methods perform SR on the rendered images (radiance). Besides this rendered image, existing works [12, 35, 53] also use G-buffer (e.g., depth) to aid the SR process, allowing geometry details reconstruction. However, the G-buffer can not assist the reconstruction of the detailed textures.

A well-known knowledge in rendering is that the radiance is a convolution of the lighting and materials. The texture details mainly come from the material component, and the lighting component tends to be smoother than the radiance, as shown in the supplementary. The decoupling of the lighting and materials is called radiance demodulation. The other key observation is that the material component, even in HR, can be generated efficiently, since it does not

need global light transport. Motivated by these insights, we introduce radiance demodulation into our SR pipeline.

The radiance for a diffuse material can be directly demodulated into a lighting component (called irradiance) and a material component (called albedo), which is widely used in denoising [1, 24, 41]. However, this idea only works for diffuse materials, and it's not applicable for a view-dependent material (e.g., glossy material). Recently, Zhuang et al. [62] partially separate a material component from the radiance image, significantly aiding the denoising task.

Following Zhuang et al. [62], the rendering equation [21] is reformulated as:

$$L(\omega_o) = \int L(\omega_i)\rho(\omega_i,\omega_o)\cos\theta_i \mathrm{d}\omega_i, \quad (1)$$

$$= F_\beta(\omega_o)\cdot I(\omega_o), \quad (2)$$

where

$$F_\beta(\omega_o) = \int \rho(\omega_i,\omega_o)\cos\theta_i \mathrm{d}\omega_i \quad (3)$$

$$I(\omega_o) = \frac{\int L(\omega_i)\rho(\omega_i,\omega_o)\cos\theta_i \mathrm{d}\omega_i}{F_\beta(\omega_o)} \quad (4)$$

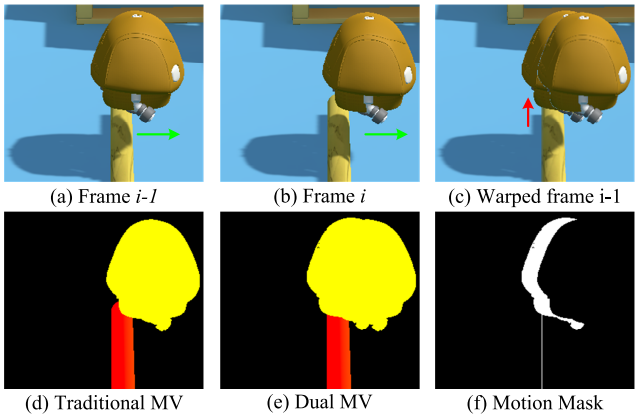
$L(\omega_o)$ and $L(\omega_i)$ represent the radiance at the outgoing direction ω_o and the incident direction ω_i, respectively. ρ represents the bidirectional reflectance distribution function (BRDF) and θ_i is the angle between the incoming direction and the shading normal. F_β and I represent the material component and the lighting component, respectively.

Now, we fit demodulated components into our SR pipeline. The renderer provides an LR lighting component I and an HR material component F_β. The SR is performed on I with the neural network, and then the reconstructed I is multiplied with the HR material component to get the final SR result of the radiance image. The details of I and F_β computation are shown in the supplementary material.

3.2. Reliable Warping

Most existing VSR and RRSR methods warp the previous frames to the current frame, using optical flow or motion vector [45]. For RRSR, the motion vector can be easily obtained during rendering. However, the motion vector becomes unreliable due to the occlusion of objects in dynamic scenes. As shown in Figure 3 (c), the regions (pointed by the red arrow) which are occluded in the previous frame but without occlusion in the current frame have been incorrectly warped into the current frame. These regions are called *motion-unreliable regions*, leading to ghosting artifacts since the SR network is not aware of these regions. Hence, we need to explicitly point out the unreliable regions for the network to aid the SR.

In this paper, we recognize these unreliable regions accurately and design a so-called *motion mask* in our network



(a) Frame *i-1* (b) Frame *i* (c) Warped frame i-1

(d) Traditional MV (e) Dual MV (f) Motion Mask

Figure 3. An example of the motion mask generation. Figures (a)-(b) represent the radiance of frame $i-1$ and frame i, respectively. The green arrows show the character's moving direction. Figure (c) is the warped result of frame $i-1$ using the traditional MV, where the occluded region has been warped incorrectly (red arrow). Figures (d)-(e) show the traditional MV and dual MV respectively. Figure (f) shows our motion mask, where the previously occluded region is marked.

with a gated convolution so that the motion-unreliable regions in the warped previous frames have small weights.

Motion Mask. We propose a novel yet simple idea to recognize the motion-unreliable regions. First, the traditional motion vector (TMV) can show the area of occluders (moving objects) in the current frame, as shown in Figure 3 (d). Second, a dual motion vector (DMV) proposed by Zeng et al. [58] is able to recognize the area of occluders in the previous and the current frame together (more details can be found in supplementary), as shown in Figure 3 (e). To recognize the motion-unreliable region, we first subtract DMV with TMV and perform a binarization, which sets the pixel as 0 when its value is smaller than a threshold (0.1 in practice), otherwise set as 1. Then, we get a motion mask, which accurately marks the previously occluded regions, as shown in Figure 3 (f).

Gated Convolution. Then, we use the motion mask in our network with a gated convolution [57] to enable a learnable feature selection mechanism. The network can better identify unreliable regions and use less temporal information in these regions. Specifically, the warped LR lighting component and G-buffers are concatenated with the motion mask to form a reliable-aware input. Then a dynamic gating map is learned from the reliable-aware input through a convolution. The valid and invalid pixels in the reliable-aware features extracted from a convolution can be treated differently using this dynamic gating map. It is formulated as:

$$\begin{aligned} X_{\text{gate}} &= \text{Conv}(W_g, X_{\text{in}}), \\ X_{\text{feat}} &= \text{Conv}(W_f, X_{\text{in}}), \\ X_{\text{out}} &= \phi(X_{\text{feat}}) \odot \sigma(X_{\text{gate}}). \end{aligned} \quad (5)$$

where X_{in} is the reliable-aware input, X_{gate} is the dynamic gating map, X_{feat} is reliable-aware features, and X_{out} is the output feature map from the reliable warping module. W_g and W_f are two convolutional filters, $\odot$ denotes element-wise multiplication, ϕ and σ indicate the LeakyReLU activation function and sigmoid function respectively.

3.3. Frame-Recurrent Super-resolution

Until now, we already have the features extracted from the current and previous frames, which can enable a super-resolution. However, we notice a temporal-unstable issue. Inspired by previous works [5, 9, 13, 16, 40, 54], the recurrent neural networks can reduce flickers between frames and produce a relatively stable video by using the reconstructed image from the previous frame. Thus, we also integrate the recurrent framework into our network.

In our recurrent framework, we not only introduce the previously reconstructed image but also exploit a convolution LSTM (ConvLSTM) [43] structure for better spatial-temporal feature extraction. The ConvLSTM can effectively utilize the state of intermediate features to obtain more spatial-temporal correlation between adjacent frames, so as to improve the reconstruction quality. Later, we use a U-shaped reconstruction module for residual channel attention blocks [60] connection to reduce network computation.

Now, we show the entire frame-recurrent reconstruction module. Firstly, we warp the reconstructed lighting component from the previous frame to the current frame, perform a pixel-unshuffle [42] operation, which maps the HR input to LR space, and extract its features and concatenate the shallow features of the other two modules (radiance demodulation and reliable warping module). Then the concatenated features are fed into the ConvLSTM together with the states of the previous intermediate features, and then fed into a U-shaped reconstruction module to reconstruct the HR lighting component of the current frame. The structure of the reconstruction module can be found in the supplementary.

Our loss function consists of three parts: a smooth L_1 loss [10] (L_1^{s}), a structural similarity loss [50] (L_{SSIM}) and a temporal loss (L_t). The temporal loss can better ensure the coherence of adjacent reconstructed frames. The final loss function is defined as:

$$\begin{aligned} L_{\text{final}}(\hat{I}_{i-1}^{\text{SR}}, I_i^{\text{SR}}, I_i^{\text{HR}}) = w_1 \cdot L_1^{\text{s}}(I_i^{\text{SR}}, I_i^{\text{HR}}) \\ + w_2 \cdot (1 - L_{\text{SSIM}}(I_i^{\text{SR}}, I_i^{\text{HR}})) \\ + w_3 \cdot L_t(\hat{I}_{i-1}^{\text{SR}}, I_i^{\text{SR}}), \end{aligned} \tag{6}$$

where L_t is defined as:

$$L_t(\hat{I}_{i-1}^{\text{SR}}, I_i^{\text{SR}}) = L_1^{\text{s}}(M_{i-1} \odot \hat{I}_{i-1}^{\text{SR}}, M_{i-1} \odot I_i^{\text{SR}}). \tag{7}$$

$\hat{I}_{i-1}^{\text{SR}}$ represents the warped reconstructed lighting component of frame $i-1$. I_i^{SR} is the network output (lighting component of frame i) and I_i^{HR} is the HR reference for the lighting component of frame i. M_{i-1} is the inversion value of the HR motion mask at frame $i-1$, which is generated by using the bilinear upsampling of the traditional and dual motion vectors. The weights w_1, w_2 and w_3 are set to 1:1:1.

4. Experiments

Unless specified in the experiment, the following experiments train a network separately on each scene for a better reconstruction quality. And by default the SR factor is set as 4×4, and the target resolution is 1920×1080. The best and second-best quality or performance is shown in **bold** and underlined, respectively. All results are calculated on RGB-channel. More experimental details and results can be found in the supplementary.

4.1. Datasets and Implementation

Our dataset consists of seven representative scenes rendered with Unity [48] engine, covering typical challenging scenarios in real-time rendering: complex textures and geometries (e.g., Bistro [33], Square [36], San_M [34]), glossy reflections (e.g., Bar [33] and ZeroDay [51]) and fast-moving objects (e.g., Airplane and Pica scene). For rendering, we use the Disney material model [2] and the ray-traced global illumination method in Unity. Similar to previous work [53], we set up different fast-moving cameras in each scene to generate multiple sequences (100 frames each). Each sequence includes different objects and materials to enhance diversity. Then, we randomly divide the training, validation and testing datasets from these sequences.

For the training data, we generate the LR lighting component, traditional motion vector, dual motion vector and G-buffers (normal, depth) as the inputs and then generate an HR lighting component as the ground truth (GT). For that, we first render it at 3840×2160 with $8\times$ MSAA and then downscale the image to 1920×1080 with a 2×2 box filter to reduce aliasing. For the testing data, we generate the LR input data and the HR material component.

We also perform data augmentation on the training dataset by randomly cropping different regions with size 96×96 at each LR frame, bringing the total training data to at least 5000 patches per scene. The test data keeps the original LR size without any cropping.

Our network is implemented in the PyTorch [37] framework with the Adam optimizer [23]. The total number of training epochs is 200, and the initial learning rate is set to $5e^{-4}$, which is halved every 100 epochs. The training samples are fed into the network in a batch size of 8. Training takes about 24 hours on a single NVIDIA RTX 3090 GPU per scene.

4.2. Quality Evaluation

We use four quality metrics to measure the quality of the reconstructed image: peak signal-to-noise ratio (PSNR),

Table 1. Quality and performance comparisons between our method and the other six methods on seven scenes.

		FRVSR	TecoGAN	NSRR	BasicVSR++	TTVSR	RVRT	Ours
PSNR / SSIM	Bistro	24.24 / 0.7648	23.02 / 0.7264	24.62 / 0.7975	25.31 / 0.8085	25.37 / 0.8108	25.42 / 0.8145	**26.43 / 0.8739**
	San_M	27.57 / 0.8422	26.61 / 0.8141	27.52 / 0.8598	29.02 / 0.8752	29.20 / 0.8754	28.58 / 0.8608	**30.37 / 0.9426**
	Square	20.61 / 0.5880	19.32 / 0.5230	20.66 / 0.6009	21.13 / 0.6276	21.19 / 0.6253	20.95 / 0.6120	**21.58 / 0.7306**
	Bar	23.80 / 0.7800	23.48 / 0.7617	25.51 / 0.8445	26.20 / 0.8513	26.02 / 0.8469	26.22 / 0.8473	**27.16 / 0.9202**
	ZeroDay	21.53 / 0.7735	21.04 / 0.7624	21.82 / 0.7922	22.28 / 0.8125	22.60 / 0.8159	22.57 / 0.8142	**23.63 / 0.8680**
	Airplane	33.03 / 0.9450	32.27 / 0.9347	31.75 / 0.9429	33.77 / 0.9534	33.94 / 0.9550	33.98 / 0.9536	**34.09 / 0.9643**
	Pica	32.73 / 0.9621	31.34 / 0.9510	32.57 / 0.9630	35.19 / 0.9773	36.33 / 0.9818	**37.09** / 0.9821	37.03 / **0.9828**
↓ LPIPS / VMAF	Bistro	0.326 / 35.97	0.234 / 35.20	0.281 / 43.16	0.282 / 45.55	0.281 / 47.28	0.278 / 48.87	**0.141 / 53.82**
	San_M	0.276 / 46.21	0.214 / 37.44	0.242 / 54.62	0.221 / 58.77	0.222 / 59.66	0.244 / 54.13	**0.075 / 73.50**
	Square	0.443 / 17.57	0.347 / 19.34	0.433 / 19.70	0.403 / 19.34	0.408 / 23.26	0.418 / 21.65	**0.227 / 26.26**
	Bar	0.321 / 34.13	0.279 / 31.15	0.318 / 34.66	0.311 / 39.44	0.314 / 40.83	0.319 / 40.01	**0.087 / 55.22**
	ZeroDay	0.301 / 21.43	0.282 / 19.85	0.291 / 27.54	0.285 / 35.61	0.278 / 37.32	0.282 / 36.92	**0.154 / 53.38**
	Airplane	0.186 / 64.43	0.153 / 65.77	0.186 / 56.99	0.148 / 70.16	0.142 / **71.46**	0.144 / 70.85	**0.076** / 70.92
	Pica	0.078 / 66.67	0.054 / 63.09	0.064 / 67.31	0.046 / 78.92	0.039 / 83.02	0.030 / **85.92**	**0.029** / 85.43
	Params (M)	2.59	2.59	**0.53**	7.32	6.77	10.78	1.61
	Runtime (ms)	14.97	14.97	23.94	98.69	>100	>100	**12.41**

structural similarity index (SSIM) [50], learned perceptual image patch similarity (LPIPS) [59] and video multi-method assessment fusion (VMAF) [27], where VMAF considers temporal stability.

We first compare our method with six previous deep-learning-based VSR and RRSR methods: FRVSR [40], TecoGAN [6], BasicVSR++ [5], TTVSR [32], RVRT [29] and NSRR [53]. Since NSRR does not have open-source code, we reproduce it and set the number of previous frames as 2, the same as our method. We retrain and test the other models on our dataset using the released code. The reconstructed results of our method and the other six methods on five scenes are in Figures 1 and 4. By comparison, our method can preserve more texture details, while all the other methods produce blurry results, such as the blackboard in the Bistro scene and painting in the Square scene, etc. Moreover, noticeable ghosting artifacts are shown in the results of NSRR (pointed by the red arrow), while our method is ghosting-free. We report the average quality metrics across all test data on seven scenes in Table 1. Our method outperforms other methods on almost all scenes.

We also compare our method with DLSS 2.0 [7] and FSR 2.0 [8] in Figure 5 and supplementary. The SR factor is set as 2×2, since they officially do not support 4×4. By comparison, DLSS 2.0 has severe aliasing, and FSR 2.0 suffers from over-blur. Our method preserves more details while anti-aliasing, and outperforms them both qualitatively and quantitatively.

4.3. Performance Measurement

We compare our method and others in terms of parameter count and running time at the bottom of Table 1. We use Nvidia TensorRT [46] with 16-bit precision for acceleration on all models. Our running time are much lower than other methods. NSRR has the fewest parameters but with a longer running time than ours. The main reason is that its upsampling is performed before reconstruction, leading to a larger feature size and higher computational cost, while our method performs upsampling after the reconstruction, significantly reducing the computation.

We further analyze the runtime cost of our model in Table 2. The total time cost for our method is about 14.0 ms, which meets the real-time requirements and can be further shortened with some hardware acceleration. Note that generating the HR material component only costs 0.8 ms, which is negligible.

Our SR method has improved the efficiency of real-time renderings significantly. For example, in the ZeroDay scene, rendering HR radiance directly with ray-traced GI costs about 89.6ms. In contrast, it only costs 29.3ms using our method, including 15.3ms for the LR lighting component rendering and 14.0ms for the SR process. This leads to an over $3\times$ performance improvement while maintaining high-fidelity results.

4.4. Generalization Ability

Besides training each scene individually, our method can also be trained on several scenes and generalized to unseen scenes, trading the quality for generalization. To demonstrate the generalization ability of our model, we compare it with FRVSR, TecoGAN and NSRR in Table 3 and the supplementary. We randomly select 120 sequences (12,000 frames in total) from five scenes, excluding the Bistro and Bar scenes, retrain all the methods and test them on the Bistro and Bar scenes. By comparison, our method produces higher quality than the other three methods, which indicates that our method has a generalization ability.

4.5. Ablation Studies

Radiance Demodulation. To validate the impact of the radiance demodulation module, we provide a comparison

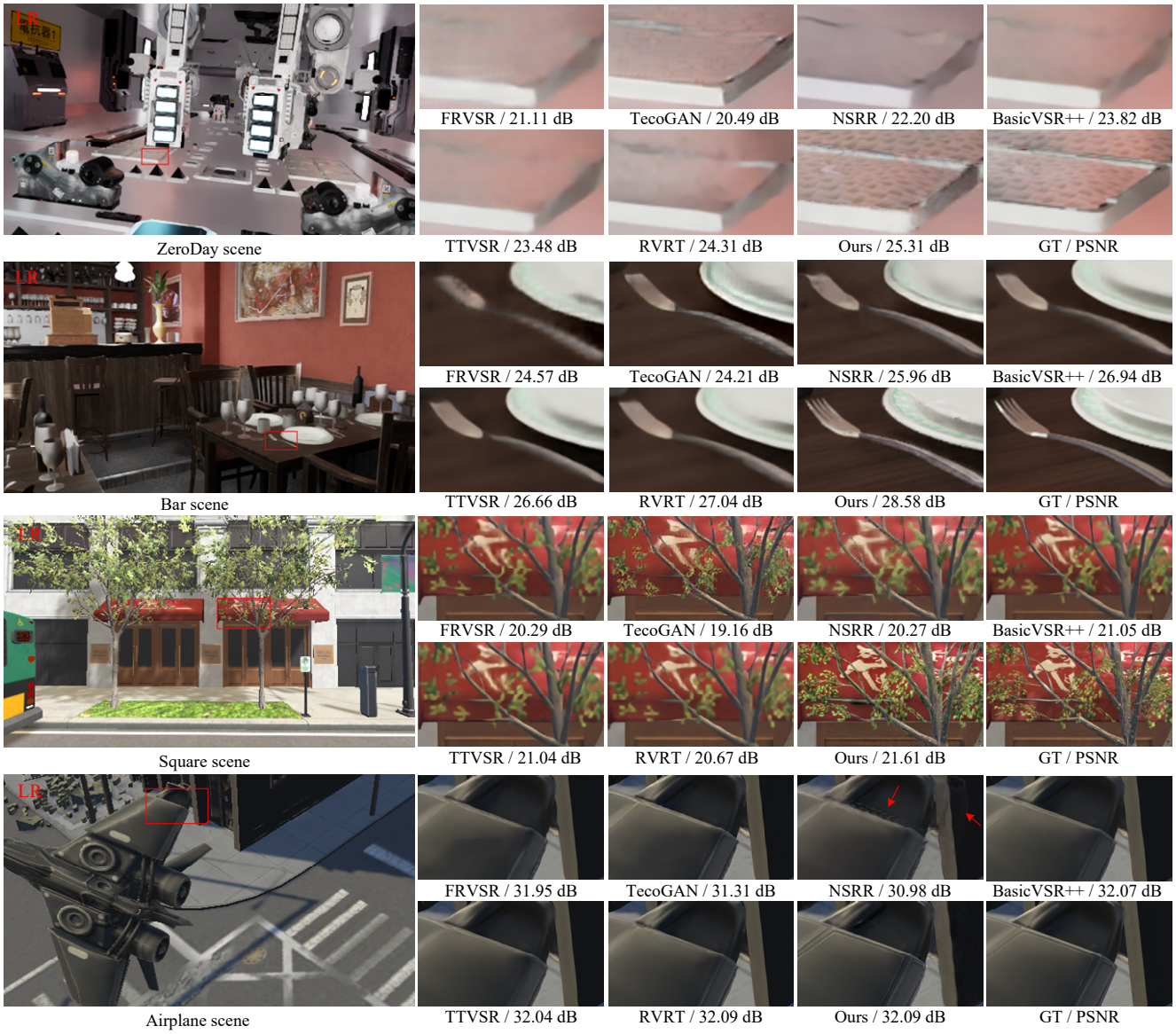


Figure 4. Comparison among our method, FRVSR [40], TecoGAN [6], NSRR [53], BasicVSR++ [5], TTVSR [32] and RVRT [29].

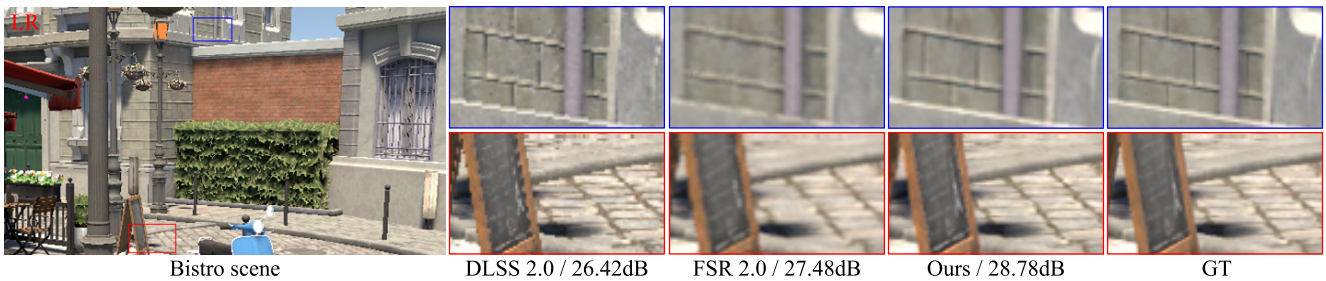


Figure 5. Comparison among our method, DLSS 2.0 [7] and FSR 2.0 [8] on the Bistro scene. The target resolution is set as 1920×1080 and the SR factor is set as 2×2.

Table 2. Time cost breakdown of our method. *HR_M* and *LR_G* represent the generation of HR material component and LR G-buffer respectively, and the *Inference* means network inference.

	HR_M	LR_G	Warping	Inference	Total
Time (ms)	0.84	0.35	0.43	12.41	14.03

Table 3. Generalization ability comparisons with other three methods on Bistro and Bar scenes.

	Scene	FRVSR	TecoGAN	NSRR	Ours
PSNR (dB)	Bistro	22.84	22.24	22.81	**23.29**
	Bar	22.13	22.02	22.07	**22.16**
SSIM	Bistro	0.7232	0.6964	0.7267	**0.8021**
	Bar	0.7488	0.7278	0.7352	**0.7836**

Table 4. The impact of the radiance demodulation for NSRR and unidirectional BasicVSR++ (BVSR++_Unidir.) on the Bistro scene. The quality metric is PSNR (dB).

	+GBuffer	+Demod.	Params (M)	Time (ms)
NSRR	24.78	25.45	**0.54**	24.35
BVSR++_Unidir.	25.03	**26.50**	4.87	75.21
Ours	24.78	26.43	1.61	**12.41**

in Figure 6. From the results, the radiance demodulation preserves more details on the blackboard and significantly improves the reconstruction quality (1.65dB in PSNR).

Both G-buffer and radiance demodulation can benefit other methods (e.g., NSRR and BasicVSR++). We compare these two methods with two modified versions against our method in Table 4. Note that we only use the forward modules in BasicVSR++ (i.e., BVSR++_Unidir.) for a fair comparison, since the bidirectional temporal feature propagation is infeasible in the real-time rendering applications. Introducing radiance demodulation improves moderately for NSRR (0.67dB in PSNR). However, it still has a lower (0.98dB in PSNR) quality than ours. BVSR++_Unidir can achieve comparable quality (26.50dB vs. 26.43dB in PSNR) with ours, at the cost of 3× parameters count and 6× running time. It indicates that our method's lightweight network is tailored for real-time rendering.

Recurrent Framework and Temporal Loss. We study the impacts of our recurrent framework and temporal loss in Table 5. To measure the temporal stability, we introduce a warping error metric [25], which is computed based on the flow warping error between adjacent frames. Combining the frame-recurrent framework and the temporal loss produces the best results. With frame-recurrent structure only, the PSNR and SSIM are better than the naive network, while the warping error is worse. Thus, the frame-recurrent framework improves the reconstruction quality, and the temporal loss improves the temporal stability.

Motion Mask. The impact of the motion mask is shown

Table 5. Ablation experiment for the recurrent framework and temporal loss on the Bistro scene.

	(A)	(B)	Ours
Recurrent Framework		✓	✓
Temporal Loss			✓
Warping Error ↓	3.5777	3.9011	2.9315
PSNR(dB)	26.05	26.20	26.43
SSIM	0.8662	0.8711	0.8739

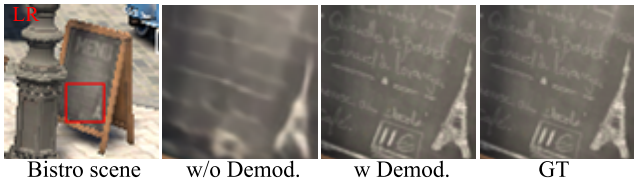


Figure 6. Ablation study of the radiance demodulation.

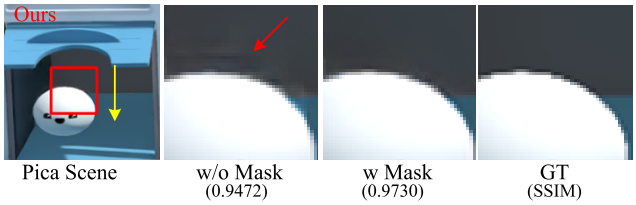


Figure 7. Ablation experiment for the motion mask (Mask). The yellow arrow indicates the direction of the ball's movement, the red arrow points out the ball's white ghosting.

in Figure 7. Without the motion mask, a translucent ghosting of the ball can be seen from the red arrow, while the ghosting artifacts disappear after using the motion mask.

5. Conclusion

In this paper, we have presented a novel lightweight super-resolution method for real-time rendering. By introducing radiance demodulation into super-resolution, the reconstructed quality is improved significantly. We also proposed a new approach to detect the motion-unreliable region, which serves as a mask to aid reconstruction and reduces the ghosting artifacts. Furthermore, a new frame-recurrent-based neural network is utilized to improve temporal stability while ensuring the reconstruction quality. Our method outperforms the existing state-of-the-art methods by a large margin. We believe it will benefit many applications like the interactive design or even video games, with further improvements in both quality and performance.

6. Acknowledgments

We thank the reviewers and Jin Xie for the valuable comments. This work has been partially supported by the National Natural Science Foundation of China under grants No.62272275 and No.62172220.

References

[1] Steve Bako, Thijs Vogels, Brian McWilliams, Mark Meyer, Jan Novák, Alex Harvill, Pradeep Sen, Tony Derose, and Fabrice Rousselle. Kernel-predicting convolutional networks for denoising monte carlo renderings. *ACM Trans. Graph.*, 36(4):97–1, 2017. 4

[2] Brent Burley and Walt Disney Animation Studios. Physically-based shading at disney. In *ACM SIGGRAPH*, volume 2012, pages 1–7. vol. 2012, 2012. 5

[3] Jose Caballero, Christian Ledig, Andrew P. Aitken, Alejandro Acosta, Johannes Totz, Zehan Wang, and Wenzhe Shi. Real-time video super-resolution with spatio-temporal networks and motion compensation. In *2017 IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2017, Honolulu, HI, USA, July 21-26, 2017*, pages 2848–2857. IEEE Computer Society, 2017. 2

[4] Kelvin CK Chan, Xintao Wang, Ke Yu, Chao Dong, and Chen Change Loy. Basicvsr: The search for essential components in video super-resolution and beyond. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 4947–4956, 2021. 2

[5] Kelvin CK Chan, Shangchen Zhou, Xiangyu Xu, and Chen Change Loy. Basicvsr++: Improving video super-resolution with enhanced propagation and alignment. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 5972–5981, 2022. 1, 2, 5, 6, 7

[6] Mengyu Chu, You Xie, Jonas Mayer, Laura Leal-Taixé, and Nils Thuerey. Learning temporal coherence via self-supervision for gan-based video generation. *ACM Transactions on Graphics (TOG)*, 39(4):75–1, 2020. 2, 6, 7

[7] Andrew Edelsten, Paula Jukarainen, and Anjul Patney. Truly next-gen: Adding deep learning to games and graphics. In *In NVIDIA Sponsored Sessions (Game Developers Conference)*, 2019. 2, 6, 7

[8] Amd fidelityfx super resolution, 2022. 2, 6, 7

[9] Dario Fuoli, Shuhang Gu, and Radu Timofte. Efficient video super-resolution through recurrent latent space propagation. In *2019 IEEE/CVF International Conference on Computer Vision Workshop (ICCVW)*, pages 3476–3485. IEEE, 2019. 2, 5

[10] Ross Girshick. Fast r-cnn. In *Proceedings of the IEEE international conference on computer vision*, pages 1440–1448, 2015. 5

[11] Jinjin Gu, Haoming Cai, Chenyu Dong, Ruofan Zhang, Yulun Zhang, Wenming Yang, and Chun Yuan. Super-resolution by predicting offsets: An ultra-efficient super-resolution network for rasterized images. In Shai Avidan, Gabriel Brostow, Moustapha Cissé, Giovanni Maria Farinella, and Tal Hassner, editors, *Computer Vision – ECCV 2022*, pages 583–598, Cham, 2022. Springer Nature Switzerland. 2, 3

[12] Yu-Xiao Guo, Guojun Chen, Yue Dong, and Xin Tong. Classifier guided temporal supersampling for real-time rendering. In *Computer Graphics Forum*, volume 41, pages 237–246. Wiley Online Library, 2022. 2, 3

[13] Muhammad Haris, Gregory Shakhnarovich, and Norimichi Ukita. Recurrent back-projection network for video super-resolution. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 3897–3906, 2019. 2, 5

[14] Yan Huang, Wei Wang, and Liang Wang. Bidirectional recurrent convolutional networks for multi-frame super-resolution. *Advances in neural information processing systems*, 28, 2015. 2

[15] Yan Huang, Wei Wang, and Liang Wang. Bidirectional recurrent convolutional networks for multi-frame super-resolution. *Advances in neural information processing systems*, 28, 2015. 2

[16] Takashi Isobe, Xu Jia, Shuhang Gu, Songjiang Li, Shengjin Wang, and Qi Tian. Video super-resolution with recurrent structure-detail network. In *European Conference on Computer Vision*, pages 645–660. Springer, 2020. 2, 5

[17] Takashi Isobe, Songjiang Li, Xu Jia, Shanxin Yuan, Gregory Slabaugh, Chunjing Xu, Ya-Li Li, Shengjin Wang, and Qi Tian. Video super-resolution with temporal group attention. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 8008–8017, 2020. 2

[18] Takashi Isobe, Fang Zhu, Xu Jia, and Shengjin Wang. Revisiting temporal modeling for video super-resolution. *arXiv preprint arXiv:2008.05765*, 2020. 2

[19] Takashi Isobe, Fang Zhu, Xu Jia, and Shengjin Wang. Revisiting temporal modeling for video super-resolution. *arXiv preprint arXiv:2008.05765*, 2020. 2

[20] Younghyun Jo, Seoung Wug Oh, Jaeyeon Kang, and Seon Joo Kim. Deep video super-resolution network using dynamic upsampling filters without explicit motion compensation. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 3224–3232, 2018. 2

[21] James T. Kajiya. The rendering equation. In David C. Evans and Russell J. Athay, editors, *Proceedings of the 13th Annual Conference on Computer Graphics and Interactive Techniques, SIGGRAPH 1986, Dallas, Texas, USA, August 18-22, 1986*, pages 143–150. ACM, 1986. 4

[22] Armin Kappeler, Seunghwan Yoo, Qiqin Dai, and Aggelos K Katsaggelos. Video super-resolution with convolutional neural networks. *IEEE transactions on computational imaging*, 2(2):109–122, 2016. 2

[23] Diederik P Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014. 5

[24] Janne Kontkanen, Jussi Räsänen, and Alexander Keller. Irradiance filtering for monte carlo ray tracing. In *Monte Carlo and Quasi-Monte Carlo Methods 2004*, pages 259–272. Springer, 2004. 4

[25] Wei-Sheng Lai, Jia-Bin Huang, Oliver Wang, Eli Shechtman, Ersin Yumer, and Ming-Hsuan Yang. Learning blind video temporal consistency. In *Proceedings of the European conference on computer vision (ECCV)*, pages 170–185, 2018. 8

[26] Wenbo Li, Xin Tao, Taian Guo, Lu Qi, Jiangbo Lu, and Jiaya Jia. Mucan: Multi-correspondence aggregation network for video super-resolution. In *European conference on computer vision*, pages 335–351. Springer, 2020. 2

[27] Zhi Li, Anne Aaron, Ioannis Katsavounidis, Anush Moorthy, and Megha Manohara. Toward a practical perceptual video quality metric. *The Netflix Tech Blog*, 6(2), 2016. 6

[28] Jingyun Liang, Jiezhang Cao, Yuchen Fan, Kai Zhang, Rakesh Ranjan, Yawei Li, Radu Timofte, and Luc Van Gool. Vrt: A video restoration transformer. *arXiv preprint arXiv:2201.12288*, 2022. 2

[29] Jingyun Liang, Yuchen Fan, Xiaoyu Xiang, Rakesh Ranjan, Eddy Ilg, Simon Green, Jiezhang Cao, Kai Zhang, Radu Timofte, and Luc V Gool. Recurrent video restoration transformer with guided deformable attention. *Advances in Neural Information Processing Systems*, 35:378–393, 2022. 1, 2, 6, 7

[30] Renjie Liao, Xin Tao, Ruiyu Li, Ziyang Ma, and Jiaya Jia. Video super-resolution via deep draft-ensemble learning. In *Proceedings of the IEEE international conference on computer vision*, pages 531–539, 2015. 2

[31] Jiayi Lin, Yan Huang, and Liang Wang. Fdan: Flow-guided deformable alignment network for video super-resolution. *arXiv preprint arXiv:2105.05640*, 2021. 2

[32] Chengxu Liu, Huan Yang, Jianlong Fu, and Xueming Qian. Learning trajectory-aware transformer for video super-resolution. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 5687–5696, 2022. 1, 2, 6, 7

[33] Amazon Lumberyard. Amazon lumberyard bistro, July 2017. 5

[34] Morgan McGuire. Computer graphics archive, July 2017. `https://casual-effects.com/data`. 5

[35] Antoine Mercier, Ruan Erasmus, Yashesh Savani, Manik Dhingra, Fatih Porikli, and Guillaume Berger. Efficient neural supersampling on a novel gaming dataset. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 296–306, 2023. 2, 3

[36] Kate Anderson Nicholas Hull and Nir Benty. Nvidia emerald square, July 2017. 5

[37] pytorch, 2022. 5

[38] Anurag Ranjan and Michael J Black. Optical flow estimation using a spatial pyramid network. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 4161–4170, 2017. 2

[39] Olaf Ronneberger, Philipp Fischer, and Thomas Brox. U-net: Convolutional networks for biomedical image segmentation. In *International Conference on Medical image computing and computer-assisted intervention*, pages 234–241. Springer, 2015. 3

[40] Mehdi SM Sajjadi, Raviteja Vemulapalli, and Matthew Brown. Frame-recurrent video super-resolution. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 6626–6634, 2018. 2, 5, 6, 7

[41] Christoph Schied, Anton Kaplanyan, Chris Wyman, Anjul Patney, Chakravarty R Alla Chaitanya, John Burgess, Shiqiu Liu, Carsten Dachsbacher, Aaron Lefohn, and Marco Salvi. Spatiotemporal variance-guided filtering: real-time reconstruction for path-traced global illumination. In *Proceedings of High Performance Graphics*, pages 1–12. 2017. 4

[42] Wenzhe Shi, Jose Caballero, Ferenc Huszar, Johannes Totz, Andrew P. Aitken, Rob Bishop, Daniel Rueckert, and Zehan Wang. Real-time single image and video super-resolution using an efficient sub-pixel convolutional neural network. In *2016 IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2016, Las Vegas, NV, USA, June 27-30, 2016*, pages 1874–1883. IEEE Computer Society, 2016. 5

[43] Xingjian Shi, Zhourong Chen, Hao Wang, Dit-Yan Yeung, Wai-Kin Wong, and Wang-chun Woo. Convolutional lstm network: A machine learning approach for precipitation nowcasting. *Advances in neural information processing systems*, 28, 2015. 5

[44] Taau, 2020. 2

[45] Natasha Tatarchuk, Brian Karis, Michal Drobot, Nicolas Schulz, Jerome Charles, and Theodor Mader. Advances in real-time rendering in games, part I. In *Special Interest Group on Computer Graphics and Interactive Techniques Conference, SIGGRAPH '14, Vancouver, Canada, August 10-14, 2014, Courses*, page 10:1. ACM, 2014. 2, 4

[46] Nvidia tensorrt, 2022. 6

[47] Yapeng Tian, Yulun Zhang, Yun Fu, and Chenliang Xu. Tdan: Temporally-deformable alignment network for video super-resolution. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 3360–3369, 2020. 2

[48] Unity, 2022. 5

[49] Xintao Wang, Kelvin CK Chan, Ke Yu, Chao Dong, and Chen Change Loy. Edvr: Video restoration with enhanced deformable convolutional networks. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops*, pages 0–0, 2019. 2

[50] Zhou Wang, Alan C Bovik, Hamid R Sheikh, and Eero P Simoncelli. Image quality assessment: from error visibility to structural similarity. *IEEE transactions on image processing*, 13(4):600–612, 2004. 5, 6

[51] Mike Winkelmann. Zero-day, open research content archive (orca), November 2019. 5

[52] 2, 3

[53] Lei Xiao, Salah Nouri, Matt Chapman, Alexander Fix, Douglas Lanman, and Anton Kaplanyan. Neural supersampling for real-time rendering. *ACM Trans. Graph.*, 39(4), jul 2020. 1, 2, 3, 5, 6, 7

[54] Bo Yan, Chuming Lin, and Weimin Tan. Frame and feature-context video super-resolution. In *Proceedings of the AAAI conference on artificial intelligence*, volume 33, pages 5597–5604, 2019. 5

[55] Sipeng Yang, Yunlu Zhao, Yuzhe Luo, He Wang, Hongyu Sun, Chen Li, Binghuang Cai, and Xiaogang Jin. Mnss: Neural supersampling framework for real-time rendering on mobile devices. *IEEE Transactions on Visualization and Computer Graphics*, 2023. 3

[56] Peng Yi, Zhongyuan Wang, Kui Jiang, Junjun Jiang, and Jiayi Ma. Progressive fusion video super-resolution network via exploiting non-local spatio-temporal correlations. In *Proceedings of the IEEE/CVF international conference on computer vision*, pages 3106–3115, 2019. 2

[57] Jiahui Yu, Zhe Lin, Jimei Yang, Xiaohui Shen, Xin Lu, and Thomas S. Huang. Free-form image inpainting with gated convolution. In *2019 IEEE/CVF International Conference on Computer Vision, ICCV 2019, Seoul, Korea (South), October 27 - November 2, 2019*, pages 4470–4479. IEEE, 2019. 4

[58] Zheng Zeng, Shiqiu Liu, Jinglei Yang, Lu Wang, and Ling-Qi Yan. Temporally reliable motion vectors for real-time ray tracing. *Computer Graphics Forum*, 40(2):79–90, 2021. 4

[59] Richard Zhang, Phillip Isola, Alexei A Efros, Eli Shechtman, and Oliver Wang. The unreasonable effectiveness of deep features as a perceptual metric. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 586–595, 2018. 6

[60] Yulun Zhang, Kunpeng Li, Kai Li, Lichen Wang, Bineng Zhong, and Yun Fu. Image super-resolution using very deep residual channel attention networks. In Vittorio Ferrari, Martial Hebert, Cristian Sminchisescu, and Yair Weiss, editors, *Computer Vision - ECCV 2018 - 15th European Conference, Munich, Germany, September 8-14, 2018, Proceedings, Part VII*, volume 11211 of *Lecture Notes in Computer Science*, pages 294–310. Springer, 2018. 5

[61] Zhihua Zhong, Jingsen Zhu, Yuxin Dai, Chuankun Zheng, Guanlin Chen, Yuchi Huo, Hujun Bao, and Rui Wang. Fusesr: Super resolution for real-time rendering through efficient multi-resolution fusion. In *SIGGRAPH Asia 2023 Conference Papers*, pages 1–10, 2023. 3

[62] Tao Zhuang, Pengfei Shen, Beibei Wang, and Ligang Liu. Real-time denoising using brdf pre-integration factorization. In *Computer Graphics Forum*, volume 40, pages 173–180. Wiley Online Library, 2021. 2, 4