Frame Interpolation with Consecutive Brownian Bridge Diffusion

Zonglin Lyu[1], Ming Li[2], Jianbo Jiao[3], and Chen Chen[2]
u1519979@umail.utah.edu, mingli@ucf.edu, j.jiao@bham.ac.uk, chen.chen@crcv.ucf.edu
[1]University of Utah, [2]Center for Research in Computer Vision, University of Central Florida, [3] University of Birmingham
Project Page: zonglinl.github.io/videointerp/

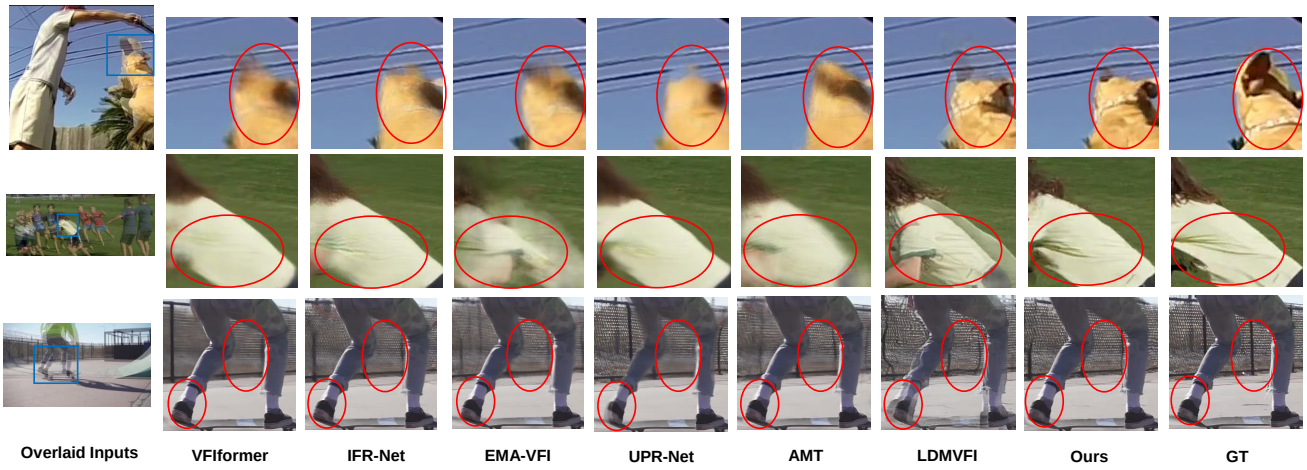

Figure 1: Qualitative Comparison of our proposed method and SOTAs. Our method generates clear interpolated images, while recent SOTAs generate blurred or overlaid results. In our method, the generated images have clearer dog skins (first row), clearer cloth with folds (second row), and clearer fences with nets and high-quality shoes (third row). Images within blue boxes are displayed for detailed comparisons, and red circles highlight our performance. Examples are chosen from SNU-FILM [7] extreme subset. More visualization results are shown in Appendix C.2

ABSTRACT

Recent work in Video Frame Interpolation (VFI) tries to formulate VFI as a diffusion-based conditional image generation problem, synthesizing the intermediate frame given a random noise and neighboring frames. Due to the relatively high resolution of videos, Latent Diffusion Models (LDMs) are employed to run diffusion models in latent space efficiently. Such a formulation poses a crucial challenge: VFI expects that the output is *deterministically* equal to the ground truth intermediate frame, but LDMs *randomly* generate a diverse set of different images when the model runs multiple times. The diversity is due to the large cumulative variance (variance accumulated at each generation step) of generated latent representations in LDMs, making the sampling trajectory random. To address this problem, we propose our unique solution: Frame Interpolation with Consecutive Brownian Bridge Diffusion. Specifically, we propose consecutive Brownian Bridge diffusion that takes a deterministic initial value as input, resulting in a much smaller cumulative variance of generated latent representations. Our experiments suggest that our method can improve together with the improvement of the autoencoder and achieve state-of-the-art performance in VFI, leaving strong potential for further enhancement. Our code is available at https://github.com/ZonglinL/ConsecutiveBrownianBridge.

CCS CONCEPTS

• **Computing methodologies** → **Computer vision**.

KEYWORDS

Video Frame Interpolation, Diffusion Models, Brownian Bridge

1 INTRODUCTION

Video Frame Interpolation (VFI) aims to generate high frame-per-second (fps) videos from low fps videos by estimating the intermediate frame given its neighboring frames. High-quality frame interpolation contributes to other practical applications such as novel view synthesis [13], video compression [56], and high-fps cartoon synthesis [45].

Current works in VFI can be divided into two folds in terms of methodologies: flow-based methods [1, 6, 12, 17, 19, 23, 28, 31, 32, 37, 40, 45, 58] and kernel-based methods [4, 5, 8, 26, 34, 35, 44]. Flow-based methods compute flows in the neighboring frames and forward warp neighboring images and features [17, 23, 32, 33, 45] or

MM '24, October 28-November 1, 2024, Melbourne, VIC, Australia

ACM ISBN 979-8-4007-0686-8/24/10
https://doi.org/10.1145/3664647.368096

estimate flows from the intermediate frame to neighboring frames and backward warp neighboring frames and features [1, 6, 12, 19, 28, 31, 37, 40, 58]. Instead of relying on optical flows, kernel-based methods predict convolution kernels for pixels in the neighboring frames. Recent advances in flow estimation [18, 20–22, 49, 50, 55] make it more popular to adopt flow-based methods in VFI.

Other than these two folds of methods, MCVD [53] and LDMVFI [10] start formulating VFI as a diffusion-based image generation problem, where LDMVFI takes advantage of LDM [41] for better efficiency. Though diffusion models achieve excellent performance in image generation, there remain challenges in applying them to VFI.

(1) The formulation of diffusion models results in a large cumulative variance (the variance accumulated during sampling) of generated latent representations. The sampling process starts with standard Gaussian noise and adds small Gaussian noise to the denoised output at each step. Noises are added up to a large cumulative variance when images are generated. Though such a variance is beneficial to diversity (i.e. repeated sampling results in *different* outputs), VFI requires that repeated sampling returns *identical* results, which is the ground truth intermediate frame. Therefore, a small cumulative variance is preferred in VFI. The relation of the cumulative variance and diversity is supported by the fact that DDIM [46] tends to generate relatively deterministic images than DDPM [16] because DDIM removes small noises at each sampling step. LDMVFI [10] uses conditional generation as guidance, but this does not change the nature of large cumulative variance. In Section 3.4, we show that our method has a much lower cumulative variance.
(2) Videos usually have high resolution, which can be up to 4K [39], resulting in practical constraints to apply diffusion models [16] in pixel spaces. It is natural to apply Latent Diffusion Models (LDMs) [41] for better efficiency, but this does not take advantage of neighboring frames, which can be a good guide to reconstruction. LDMVFI[10] designs reconstruction models that leverage neighboring frames, but it tends to reconstruct overlaid images when there is a relatively large motion between neighboring frames, possibly due to the cross-attention with features of neighboring frames, which is shown in Figure 1.

To tackle these challenges, we propose a consecutive Brownian Bridge Diffusion (in latent space) that transits among three deterministic endpoints for VFI. This method results in a much smaller cumulative variance, achieving a better estimation of the ground truth inputs. We also provide a novel method to analyze the LDM-based VFI methods: by analyzing the gap between quantitative metrics of the outputs from the autoencoder and the outputs from the diffusion model + decoder (we name this as ground truth estimation), it is easier to figure out the specific directions of improvement: whether to improve the autoencoder or the diffusion model. Moreover, we take advantage of flow estimation and refinement methods in recent literature [31] to improve the autoencoder. The feature pyramids from neighboring frames are warped based on estimated optical flows, aiming to alleviate the issues of reconstructing overlaid images. In experiments, our method improves by a large margin when the autoencoder is improved and achieves state-of-the-art performance. Our contribution can be summarized in three parts:

- We propose a novel consecutive Brownian Bridge diffusion model for VFI and justify its advantages over traditional diffusion models: lower cumulative variance and better ground truth estimation capability. Additionally, we provide a cleaner formulation of Brownian Bridges and also propose the loss weighting for our Consecutive Brownian Bridges.
- We provide a novel method to analyze LDM-based VFI. With our methods of analysis, researchers can have specific directions for further improvements.
- Through extensive experiments, we validate the effectiveness of our method. Our method estimates the ground truth better than traditional diffusion with conditional generation (LDMVFI [10]). Moreover, the performance of our method improves when the autoencoder improves and achieves state-of-the-art performance with a simple yet effective autoencoder, indicating its strong potential in VFI.

2 RELATED WORKS

2.1 Video Frame Interpolation

Video Frame Interpolation can be roughly divided into two categories in terms of methodologies: flow-based methods [1, 6, 12, 17, 19, 23, 28, 31, 32, 37, 40, 45, 58] and kernel-based methods [4, 5, 8, 26, 34, 35, 44]. Flow-based methods assume certain motion types, where a few works assume non-linear types [6, 12] while others assume linear. Via such assumptions, flow-based methods estimate flows in two ways. Some estimate flows from the intermediate frame to neighboring frames (or the reverse way) and apply backward warping to neighboring frames and their features [1, 6, 12, 19, 28, 31, 37, 40, 58]. Others compute flows among the neighboring frames and apply forward splatting [17, 23, 32, 33, 45]. In addition to the basic framework, advanced details such as recurrence of inputs with different resolution level [23], cross-frame attention [58], and 4D-correlations [28] are proposed to improve performance. Kernel-based methods, introduced by [34], aim to predict the convolution kernel applied to neighboring frames to generate the intermediate frame, but it has difficulty in dealing with large displacement. Following works [5, 8, 26] alleviate such issues by introducing deformable convolution. LDMVFI [10] recently introduced a method based on Latent Diffusion Models (LDMs) [41], formulating VFI as a conditional generation task. LDMVFI uses an autoencoder introduced by LDMs to compress images into latent representations, efficiently run the diffusion process, and then reconstruct images from latent space. Instead of directly predicting image pixels during reconstruction, it takes upsampled latent representations in the autoencoder as inputs to predict convolution kernels in kernel-based methods to complete the VFI task.

2.2 Diffusion Models

The diffusion model is introduced by DDPM [16] to image generation task and achieves excellent performance in image generation. The whole diffusion model can be split into a forward diffusion process and a backward sampling process. The forward diffusion process is defined as a Markov Chain with steps $t = 1, ..., T$, and the backward sampling process aims to estimate the distribution of the

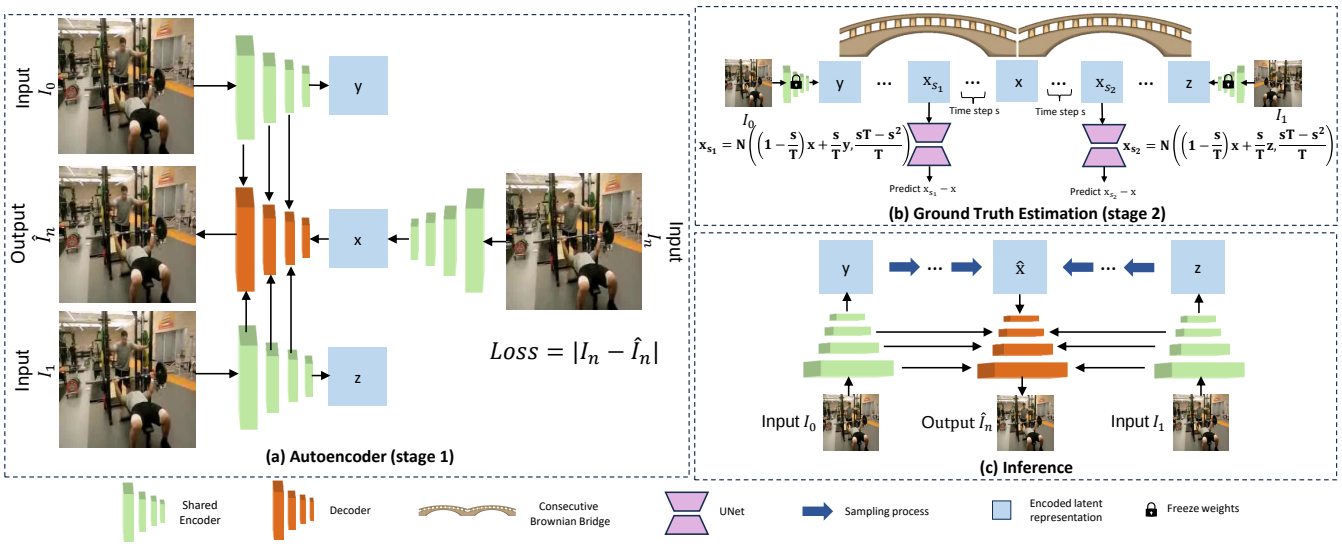

Figure 2: The illustration of our two-stage method. The encoder is shared for all frames. (a) The autoencoder stage. In this stage, previous frame I_0, intermediate frame I_n, and next frame I_1 are encoded by the encoder to y, x, z respectively. Then x is fed to the decoder, together with the encoder feature of I_0, I_1 at different down-sampling factors. The decoder predicts the intermediate frame as $\hat{I}_n$. The encoder and decoder are trained in this stage. (b) The ground truth estimation stage. In this stage, y, x, z will be fed to the consecutive Brownian Bridge diffusion as three endpoints, where we sample two states that move time step s from x in both directions. The UNet predicts the difference between the current state and x. The autoencoder is well-trained and frozen in this stage. (c) Inference. $\hat{x}$ is sampled from y, z to estimate x (details in Section 3.4). The decoder receives $\hat{x}$ and encoder features of I_0, I_1 at different down-sampling factors to interpolate the intermediate frame.

reversed Markov chain. The variance of the reversed Markov chain has a closed-form solution, and the expected value is estimated with a deep neural network. Though achieving strong performance in image generation tasks, DDPM [16] requires $T = 1000$ iterative steps to generate images, resulting in inefficient generation. Sampling steps cannot be skipped without largely degrading performance due to its Markov chain property. To enable efficient and high-quality generation, DDIM [46] proposes a non-Markov formulation of diffusion models, where the conditional distribution at time $t-k$ ($k > 0$) can be directly computed with the conditional distribution at time t. Therefore, skipping steps does not largely degrade performance. Score-based SDEs [3, 47, 60] are also proposed as an alternative formulation of diffusion models by writing the diffusion process in terms of Stochastic Differential Equations [36], where the reversed process has a closed-form continuous time formulation and can be solved with Eluer's method with a few steps [47]. In addition, Probability Flow ODE is proposed as the deterministic process that shares the same marginal distribution with the reversed SDE [47]. Following score-based SDEs, some works propose efficient methods to estimate the solution Probability Flow ODE [29, 30]. Other than using the diffusion process to connect data distribution and Gaussian distribution, diffusion bridges [11, 27, 43, 60] are proposed to connect arbitrary distributions such as two different data distributions. Instead of working on the diffusion process, the Latent Diffusion Model [41] proposes autoencoders with KL-regularized (VAE) and VQ-regularized (VQ Layer) that compress and reconstruct images, and diffusion models run with compressed images. With such autoencoders, high-resolution images can be generated efficiently. In our work, the Vector Quantized (VQ) version is deployed.

3 METHODOLOGY

In this section, we will first go through preliminaries on the Diffusion Model (DDPM) [16] and Brownian Bridge Diffusion Model (BBDM) [27] and introduce the overview of the two-stage formulation: autoencoder and ground truth estimation (with consecutive Brownian Bridge diffusion). Then, we will discuss the details of our autoencoder method. Finally, we propose our solution to the frame interpolation task: consecutive Brownian Bridge diffusion.

3.1 Preliminaries

Diffusion Model. The forward diffusion process of Diffsuion Model [16] is defined as:

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

When $t = 1$, $\mathbf{x}_{t-1} = \mathbf{x}_0$ is a sampled from the data (images). By iterating Eq. (1), we get the conditional marginal distribution [16]:

$$q(\mathbf{x}_t|x_0) = \mathcal{N}(x_t; \sqrt{\alpha}_t\mathbf{x}_0, (1-\alpha_t)\mathbf{I}), \tag{2}$$

$$\text{where } \alpha_t = \prod_{s=1}^{t}(1-\beta_s).$$

The sampling process is derived with the Bayes' theorem [16]:

$$p_\theta(\mathbf{x}_{t-1}|\mathbf{x}_t) = q(\mathbf{x}_{t-1}|\mathbf{x}_0, \mathbf{x}_t) = \mathcal{N}(x_{t-1}; \tilde{\mu}_t, \tilde{\beta}_t), \tag{3}$$

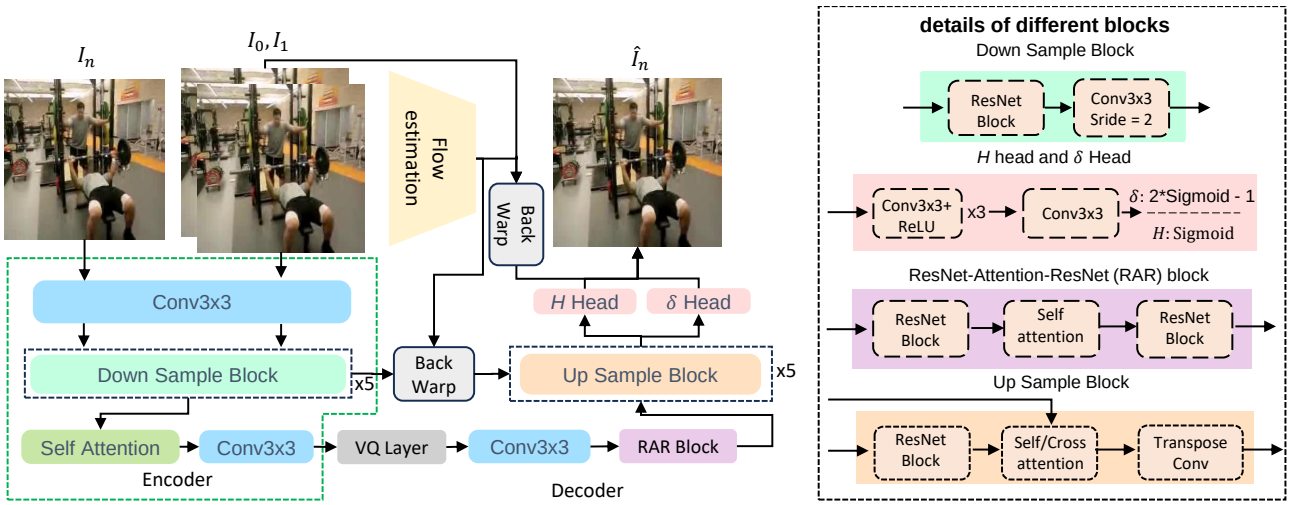

Figure 3: Architecture of the autoencoder. The encoder is in green dashed boxes, and the decoder contains all remaining parts. The output of consecutive Brownian Bridge diffusion will be fed to the VQ layer. The features of I_0, I_1 at different down-sampling rate will be sent to the cross-attention module at Up Sample Block in the Decoder.

$$\text{where } \tilde{\mu}_t = \frac{\sqrt{\alpha_{t-1}}\beta_t}{1-\alpha_t}\mathbf{x}_0 + \frac{\sqrt{1-\beta_t}(1-\alpha_{t-1})}{1-\alpha_t}\mathbf{x}_t, \tag{4}$$

$$\text{and } \tilde{\beta}_t = \frac{1-\alpha_{t-1}}{1-\alpha_t}\beta_t. \tag{5}$$

Eq. (4) can be rewritten with Eq. (2) via reparameterization:

$$\tilde{\mu}_t = \frac{1}{1-\beta_t}\left(\mathbf{x}_t - \frac{\beta_t}{\sqrt{1-\alpha_t}}\epsilon\right), \text{where } \epsilon \sim \mathcal{N}(0, \mathbf{I}). \tag{6}$$

By Eq. (4) and (6), we only need to estimate ϵ to estimate $p_\theta(\mathbf{x}_{t-1}|\mathbf{x}_t)$. Therefore, the training objective is:

$$\mathbb{E}_{\mathbf{x}_0,\epsilon}\left[||\epsilon_\theta(\mathbf{x}_t, t) - \epsilon||_2^2\right]. \tag{7}$$

It suffices to train a neural network $\epsilon_\theta(\mathbf{x}_t, t)$ predicting ϵ.

Brownian Bridge Diffusion Model. Brownian Bridge [42] is a stochastic process that transits between two fixed endpoints, which is formulated as $X_t = W_t|(W_{t_1}, W_{t_2})$, where W_t is a standard Wiener process with distribution $\mathcal{N}(0, t)$. We can write a Brownian Bridge as $X_t = W_t|(W_0, W_T)$ to define a diffusion process. When $W_0 = a, W_T = b$, we have:

$$X_t \sim \mathcal{N}\left((1-\frac{t}{T})a + \frac{t}{T}b, \frac{tT-t^2}{T}\right). \tag{8}$$

BBDM [27] develops an image-to-image translation method based on the Brownian Bridge process by treating a and b as two images. The forward diffusion process is defined as:

$$q(\mathbf{x}_t|\mathbf{x}_0, \mathbf{y}) = \mathcal{N}\left(\mathbf{x}_t; (1-m_t)\mathbf{x}_0 + m_t\mathbf{y}, \delta_t\right), \tag{9}$$

$$\text{where } m_t = \frac{t}{T} \text{ and } \delta_t = 2s(m_t - m_t^2). \tag{10}$$

$\mathbf{x}_0$ and $\mathbf{y}$ are two images, and s is a constant that controls the maximum variance in the Brownian Bridge. The sampling process is derived based on Bayes' theorem [27]:

$$\begin{aligned} p_\theta(\mathbf{x}_{t-1}|\mathbf{x}_t, \mathbf{y}) &= q(\mathbf{x}_{t-1}|\mathbf{x}_0, \mathbf{x}_t, \mathbf{y}) \\ &= \frac{q(\mathbf{x}_t|\mathbf{x}_{t-1}, \mathbf{y})q(\mathbf{x}_{t-1}|\mathbf{x}_0, \mathbf{y})}{q(\mathbf{x}_t|\mathbf{x}_0, \mathbf{y})} \\ &= \mathcal{N}(\tilde{\mu}_t, \tilde{\delta}_t\mathbf{I}). \end{aligned} \tag{11}$$

$$\text{where } \tilde{\mu}_t = c_{xt}\mathbf{x}_t + c_{yt}y + c_{\epsilon t}(m_t(\mathbf{y}-\mathbf{x}_0) + \sqrt{\delta_t}\epsilon),$$

$$c_{xt} = \frac{\delta_{t-1}}{\delta_t}\frac{1-m_t}{1-m_{t-1}} + \frac{\delta_{t|t-1}}{\delta_t}(1-m_t),$$

$$c_{yt} = m_{t-1} - m_t\frac{1-m_t}{1-m_{t-1}}\frac{\delta_{t-1}}{\delta_t},$$

$$c_{\epsilon t} = (1-m_{t-1})\frac{\delta_{t|t-1}}{\delta_t},$$

$$\delta_{t|t-1} = \delta_t - \delta_{t-1}\frac{(1-m_t)^2}{(1-m_{t-1})^2}.$$

It suffices to train a deep neural network ϵ_θ to estimate the term $c_{\epsilon t}(m_t(\mathbf{y}-\mathbf{x}_0) + \sqrt{\delta_t}\epsilon)$, and therefore the training objective is $\mathbb{E}_{\mathbf{x}_0,\mathbf{y},\epsilon}[c_{\epsilon t}||m_t(\mathbf{y}-\mathbf{x}_0) + \sqrt{\delta_t}\epsilon - \epsilon_\theta(\mathbf{x}_t, t)||_2^2]$.

3.2 Formulation of Diffusion-based VFI

The goal of video frame interpolation is to estimate the intermediate frame I_n given the previous frame I_0 and the next frame I_1. n is set to 0.5 to interpolate the frame in the middle of I_0 and I_1. In latent diffusion models [41], there is an autoencoder that encodes images to latent representations and decodes images from latent representations. The diffusion process denoises a latent representation, and the decoder reconstruct it back to an image. Since the initial noise is random, the decoded images are diverse images when they are sampled repetitively with the same conditions such as poses. Instead of

diversity, VFI looks for a deterministic ground truth, which is the intermediate frame. To estimate the ground truth intermediate frame, we only need to estimate the corresponding latent representation in the LDM-based framework. Therefore, LDM-based VFI can be split into two stages: autoencoder and ground truth estimation. The two stages are defined as:

(1) **Autoencoder**. The primary function of the autoencoder is similar to image compression: compressing images to latent representations so that the diffusion model can be efficiently implemented. We denote $\mathbf{x}, \mathbf{y}, \mathbf{z}$ as encoded latent representations of I_n, I_0, I_1. In this stage, the goal is to compress I_n to $\mathbf{x}$ with an encoder and then reconstruct I_n from $\mathbf{x}$ with a decoder. $\mathbf{x}$ is provided to the decoder together with neighboring frames I_0, I_1 and their features in the encoder at different down-sampling factors. The overview of this stage is shown in Figure 2 (a). However, to interpolate the intermediate frame, $\mathbf{x}$ is unknown, so we need to estimate this ground truth.
(2) **Ground truth estimation**. In this stage, the goal is to accurately estimate $\mathbf{x}$ with a diffusion model. The diffusion model converts $\mathbf{x}$ to $\mathbf{y}, \mathbf{z}$ with the diffusion process, and we train a UNet to predict the difference between the current diffusion state and $\mathbf{x}$, shown in Figure 2 (b). The sampling process of the diffusion model will convert $\mathbf{y}, \mathbf{z}$ to $\mathbf{x}$ with the UNet output.

The autoencoder is modeled with VQModel [41] in Section 3.3, and the ground truth estimation is accomplished by our consecutive Brownian Bridge Diffusion in Section 3.4. During inference, both stages are combined as shown in Figure 2 (c), where we decode diffusion-generated latent representation $\hat{\mathbf{x}}$. Via such formulation, we have a novel method to analyze the LDM-based VFI method. If images decoded from $\mathbf{x}$ (Figure 2 (a)) have similar visual quality to images decoded from $\hat{\mathbf{x}}$ (Figure 2 (c)), then the diffusion model achieves a strong performance in ground truth estimation, so it will be good to develop a good autoencoder. On the other way round, the performance of ground truth estimation can be potentially improved by redesigning the diffusion model.

3.3 Autoencoder

Diffusion models running in pixel space are extremely inefficient in video interpolation because videos can be up to 4K in real life [39]. Therefore, we can encode images into a latent space with encoder $\mathcal{E}$ and decode images from the latent space with decoder $\mathcal{D}$. Features of I_0, I_1 are included because detailed information may be lost when images are encoded to latent representations [10]. We incorporate feature pyramids of neighboring frames into the decoder stage as guidance because neighboring frames contain a large number of shared details. Given I_n, I_0, I_1, the encoder $\mathcal{E}$ will output encoded latent representation $\mathbf{x}, \mathbf{y}, \mathbf{z}$ for diffusion models and feature pyramids of I_0, I_1 in different down-sampling rates, denoted $\{f_y^k\}, \{f_z^k\}$, where k is down-sampling factor. When $k = 1$, $\{f_y^k\}$ and $\{f_z^k\}$ represent original images. The decoder $\mathcal{D}$ will take sampled latent representation $\hat{\mathbf{x}}$ (output of diffusion model that estimates $\mathbf{x}$) and feature pyramids $\{f_y^k\}, \{f_z^k\}$ to reconstruct I_n. In lines of equations, we have:

$$\begin{aligned} \mathbf{x}, \mathbf{y}, \{f_y^k\}, \mathbf{z}, \{f_z^k\} &= \mathcal{E}(I_n, I_0, I_1), \\ \hat{I}_n &= \mathcal{D}\left(\mathbf{x}, \{f_y^k\}, \{f_z^k\}\right). \end{aligned} \tag{12}$$

Our encoder shares an identical structure with that in LDMVFI [10], and we slightly modify the decoder to better fit the VFI task.

Decoding with Warped Features. LDMVFI [10] apply cross-attention [52] to up-sampled $\hat{\mathbf{x}}$ and f_x^k, f_y^k. However, this does not explicitly deal with motion changes, and therefore LDMVFI usually produces overlaid results as shown in Figure 1. Therefore, we estimate optical flows from I_n to I_0, I_1 and apply backward warping to the feature pyramids to tackle this problem. Suppose $\hat{x}$ is generated by our consecutive Brownian Bridge diffusion, and it is up-sampled to h^k where k denotes the down-sampling factor compared to the original image. Then, we apply $CA\left(h^k, Cat(w(f_y^k), w(f_z^k))\right)$ for $k > 1$ to fuse the latent representation h^k and feature pyramids f_y^k and f_z^k, where $CA(\cdot,\cdot)$, $Cat(\cdot,\cdot)$, and $w(\cdot)$ denotes cross attention, channel-wise concatenation, and backward warping with estimated optical flows respectively. Finally, we apply convolution layers to h^1 to predict soft mask H and residual δ. The interpolation output is $\hat{I}_n = H * w(I_0) + (1 - H) * w(I_1) + \delta$, where $*$ holds for Hadamard product, and $\hat{I}_n$ is the reconstructed image. The detailed illustration of the architecture is shown in Figure 3. The VQ layer is connected with the encoder during training, but it is disconnected from the encoder and receives the sampled latent representation from the diffusion model.

3.4 Consecutive Brownian Bridge Diffusion

Brownian Bridge diffusion model (BBDM) [27] is designed for translation between image pairs, connecting two deterministic points, which seems to be a good solution to estimate the ground truth intermediate frame. However, it does not fit the VFI task. In VFI, images are provided as triplets because we aim to reconstruct intermediate frames giving neighboring frames, resulting in three points that need to be connected. If we construct a Brownian Bridge between the intermediate frame and the next frame, then the previous frame is ignored, and so is the other way round. This is problematic because we do not know what "intermediate" is if we lose one of its neighbors. Therefore, we need a process that transits among three images. Given two neighboring images I_0, I_1, we aim to construct a Brownian Bridge process with endpoints I_0, I_1 and additionally condition its middle stage on the intermediate frame I_n ($n = 0.5$ for $2\times$ interpolation). To achieve this, the process starts at $t = 0$ with value $\mathbf{y}$, passes $t = T$ with value $\mathbf{x}$, and ends at $t = 2T$ with value $\mathbf{z}$. To be consistent with the notation in diffusion models, $\mathbf{x}, \mathbf{y}, \mathbf{z}$ are used to represent latent representations of I_n, I_0, I_1 respectively. It is therefore defined as $X_t = W_t | W_0 = \mathbf{y}, W_T = \mathbf{x}, W_{2T} = \mathbf{z}$. The sampling process starts from time 0 and $2T$ and goes to time T. Such a process indeed consists of two Brownian Bridges, where the first one ends at $\mathbf{x}$ and the second one starts at $\mathbf{x}$. We can easily verify that for $0 < t < h$:

$$W_s | (W_0, W_t, W_h) = \begin{cases} W_s | (W_0, W_t) & \text{if } s < t \\ W_s | (W_t, W_h) & \text{if } s > t \end{cases}. \tag{13}$$

According to Eq. (13), we can derive the distribution of our consecutive Brownian Bridge diffusion (details shown in Appendix A.1):

Algorithm 1 Training

repeat
  sample triplet $\mathbf{x}, \mathbf{y}, \mathbf{z}$ from dataset
  $s \leftarrow Uniform(0, T)$
  $w_s \leftarrow min\{\frac{1}{\delta_t}, \gamma\}$ ▷ γ is a pre-defined constant
  $\epsilon \leftarrow \mathcal{N}(\mathbf{0}, \mathbf{I})$
  $\mathbf{x}_{\mathbf{s}_1} \leftarrow \frac{s}{T}\mathbf{x} + (1 - \frac{s}{T})\mathbf{y} + \sqrt{\frac{s(T-s)}{T}}\epsilon$
  $\mathbf{x}_{\mathbf{s}_2} \leftarrow \frac{s}{T}\mathbf{x} + (1 - \frac{s}{T})\mathbf{z} + \sqrt{\frac{s(T-s)}{T}}\epsilon$
  $\mathrm{r} \leftarrow Uniform(0, 1)$
  if r < 0.5 **then** take a gradient step on
    $\nabla_\theta ||\epsilon_\theta(\mathbf{x}_{s_1}, T - s, \mathbf{y}, \mathbf{z}) - (\mathbf{x}_{s_1} - \mathbf{x})||_2^2$
  else take a gradient step on
    $\nabla_\theta ||\epsilon_\theta(\mathbf{x}_{s_2}, T + s, \mathbf{y}, \mathbf{z}) - (\mathbf{x}_{s_2} - \mathbf{x})||_2^2$
  end if
until convergence

Algorithm 2 Sampling

$t_1, t_2 \leftarrow T, \Delta_t \leftarrow \frac{T}{\text{sampling steps}}, \mathbf{x}_{T_1} = \mathbf{y}, \mathbf{x}_{T_2} = \mathbf{z}$
repeat
  $s_1, s_2 \leftarrow t_1 - \Delta_t, t_2 - \Delta_t$
  $\epsilon \leftarrow \mathcal{N}(\mathbf{0}, \mathbf{I})$
  $\mathbf{x}_{\mathbf{s}_1} \leftarrow x_{t_1} - \frac{\Delta_t}{t_1}\epsilon_\theta(x_{t_1}, T - t_1, \mathbf{y}, \mathbf{z}) + \sqrt{\frac{s_1\Delta_t}{t_1}}\epsilon$
  $\mathbf{x}_{\mathbf{s}_2} \leftarrow x_{t_2} - \frac{\Delta_t}{t_2}\epsilon_\theta(x_{t_2}, T - t_2, \mathbf{y}, \mathbf{z}) + \sqrt{\frac{s_2\Delta_t}{t_2}}\epsilon$
  $t_1, t_2 \leftarrow s_1, s_2$
until $t_1, t_2 = 0$

$$q(\mathbf{x}_t|\mathbf{y}, \mathbf{x}, \mathbf{z}) = \begin{cases} \mathcal{N}(\frac{s}{T}\mathbf{x} + (1 - \frac{s}{T})\mathbf{y}, \frac{s(T-s)}{T}\mathbf{I}) \; s = T - t, t < T \\ \mathcal{N}(\frac{s}{T}\mathbf{x} + (1 - \frac{s}{T})\mathbf{z}, \frac{s(T-s)}{T}\mathbf{I}) \; s = t - T, t > T \end{cases}. \tag{14}$$

Cleaner Formulation. Eq. (11) is in a discrete setup, and the sampling process is derived via Bayes' theorem, resulting in a complicated formulation. To preserve the maximum variance, it suffices to have $T = 2s$ in Eq. (8) with a continuous formulation and discretize it for training and sampling. Our forward diffusion is defined as Eq. (14). To sample at time s from t ($s < t$), we rewrite Eq. (11) according to Eq. (13):

$$\begin{aligned} p_\theta(\mathbf{x}_s|\mathbf{x}_t, \mathbf{y}) &= q(\mathbf{x}_s|\mathbf{x}, \mathbf{x}_t, \mathbf{y}) = q(\mathbf{x}_s|\mathbf{x}, \mathbf{x}_t) \\ &= \mathcal{N}\left(\mathbf{x}_s; \frac{s}{t}\mathbf{x}_t + (1 - \frac{s}{t})\mathbf{x}, \frac{s(t-s)}{t}\mathbf{I}\right) \\ &= \mathcal{N}\left(\mathbf{x}_s; \mathbf{x}_t - \frac{t-s}{t}(\mathbf{x}_t - \mathbf{x}), \frac{s(t-s)}{t}\mathbf{I}\right). \end{aligned} \tag{15}$$

Note that $\mathbf{x}_0$ in Eq. (11) and $\mathbf{x}$ in our formulation both represent the image. This formulation can be solved with a few steps without DDIM [46] similar to Euler's method.

Training and Sampling. According to Eq. (15), it suffices to have a neural network ϵ_θ estimating $\mathbf{x}_t - \mathbf{x}_0$. Moreover, based on Eq. (14), we can sample s from $Uniform(0, T)$ and compute $t = T \pm s$ for $t > T$ and $T < t$. With one sample of s, we can obtain two samples at each side of our consecutive Brownian bridge diffusion symmetric at T. $\mathbf{y}, \mathbf{z}$ are added to the denoising UNet as extra conditions. Therefore, the training objective becomes:

$$\begin{aligned} &\mathbb{E}_{\{\mathbf{y},\mathbf{x},\mathbf{z}\},\epsilon}[||\epsilon_\theta(\mathbf{x}_{s_1}, T - s, \mathbf{y}, \mathbf{z}) - (\mathbf{x}_{s_1} - \mathbf{x})||_2^2] \\ &+ \mathbb{E}_{\{\mathbf{y},\mathbf{x},\mathbf{z}\},\epsilon}[||\epsilon_\theta(\mathbf{x}_{s_2}, T + s, \mathbf{y}, \mathbf{z}) - (\mathbf{x}_{s_2} - \mathbf{x})||_2^2]. \end{aligned} \tag{16}$$

$$\begin{aligned} \text{where } \mathbf{x}_{\mathbf{s}_1} &= \frac{s}{T}\mathbf{x} + (1 - \frac{s}{T})\mathbf{y} + \sqrt{\frac{s(T-s)}{T}}\epsilon, \\ \mathbf{x}_{\mathbf{s}_2} &= \frac{s}{T}\mathbf{x} + (1 - \frac{s}{T})\mathbf{z} + \sqrt{\frac{s(T-s)}{T}}\epsilon, \\ \epsilon &\sim \mathcal{N}(\mathbf{0}, \mathbf{I}). \end{aligned} \tag{17}$$

Optimizing Eq. (16) requires two forward calls of UNet. For efficiency, we randomly select one of them to optimize during training. Moreover, [14] proposes $min - SNR - \gamma$ loss weighting for different time steps based on the signal-to-noise ratio, defined as $min\{SNR(t), \gamma\}$. In DDPM [16], we have $SNR(t) = \frac{\alpha_t}{1-\alpha_t}$ because the mean and standard deviation are scaled by $\sqrt{\alpha_t}$ and $\sqrt{1-\alpha_t}$ respectively in the diffusion process. In our formulation, the expected values are not scaled down: neighboring frames share almost identical expected values. Therefore, the SNR is defined as $\frac{1}{\delta_t}$, where δ_t is the standard deviation of the diffusion process at time t. The weighting is defined as $w_t = min\{\frac{1}{\delta_t}, \gamma\}$.

The training algorithm is shown in Algorithm 1. To sample from neighboring frames, we sample from either of the two endpoints $\mathbf{y}, \mathbf{z}$ with Eq. (14) and (15), shown in Algorithm 2. After sampling, we replace $\mathbf{x}$ in Eq (12) with the sampled latent representations to decode the interpolated frame.

Cumulative Variance. As we claimed, diffusion model [16] with conditional generation has a large cumulative variance while ours is much smaller. The cumulative variance for traditional conditional generation is larger than $1 + \sum_t \hat{\beta}_t$, which corresponds to 11.036 in experiments. However, in our method, such a cumulative variance is smaller than $T = 2$ in our experiments, resulting in a more deterministic estimation of the ground truth latent representations. Detailed justification is shown in Appendix A.2

4 EXPERIMENTS

4.1 Implementations

Autoencoder. The down-sampling factor is set to be $f = 32$ for our autoencoder, which follows the setup of LDMVFI [10]. The flow estimation and refinement modules are initialized from pretrained VFIformer [31] and frozen for better efficiency. The codebook size and embedding dimension of the VQ Layer are set to 16384 and 3 respectively. The number of channels in the latent space (encoder output) is set to 8. A self-attention [52] is applied at 32× down-sampling latent representation (both encoder and decoder), and cross attentions [52] with warped features are applied on the 2× to 32× down-sampling factors in the decoder. Following LDMVFI, max-attention [51] is applied for better efficiency. The model is trained with Adam optimizer [24] with a learning rate of 10^{-5} for 100 epochs with a batch size of 16.

Consecutive Brownian Bridge Diffusion. We set $T = 2$ (corresponding to maximum variance $\frac{1}{2}$) and discretize 1000 steps for training and 50 steps for sampling. The denoising UNet takes the

Table 1: Quantitative results (LPIPS/FloLPIPS/FID, the lower the better) on test datasets. † means we evaluate our consecutive Brownian Bridge diffusion (trained on Vimeo 90K triplets [57]) with autoencoder provided by LDMVFI [10]. The best performances are boldfaced, and the second best performances are underlined.

Methods	Middlebury	UCF-101	DAVIS	SNU-FILM			
				easy	medium	hard	extreme
	LPIPS/FloLPIPS/FID	LPIPS/FloLPIPS/FID	LPIPS/FloLPIPS/FID	LPIPS/FloLPIPS/FID	LPIPS/FloLPIPS/FID	LPIPS/FloLPIPS/FID	LPIPS/FloLPIPS/FID
ABME'21 [38]	0.027/0.040/11.393	0.058/0.069/37.066	0.151/0.209/16.931	0.022/0.034/6.363	0.042/0.076/15.159	0.092/0.168/34.236	0.182/0.300/63.561
MCVD'22 [53]	0.123/0.138/41.053	0.155/0.169/102.054	0.247/0.293/28.002	0.199/0.230/32.246	0.213/0.243/37.474	0.250/0.292/51.529	0.320/0.385/83.156
VFIformer'22 [31]	0.015/0.024/9.439	0.033/0.040/22.513	0.127/0.184/14.407	0.018/0.029/5.918	0.033/0.053/11.271	0.061/0.100/22.775	0.119/0.185/40.586
IFRNet'22 [25]	0.015/0.030/10.029	0.029/0.034/20.589	0.106/0.156/12.422	0.021/0.031/6.863	0.034/0.050/12.197	0.059/0.093/23.254	0.116/0.182/42.824
AMT'23 [28]	0.015/0.023/7.895	0.032/0.039/21.915	0.109/0.145/13.018	0.022/0.034/6.139	0.035/0.055/11.039	0.060/0.092/20.810	0.112/0.177/40.075
UPR-Net'23 [23]	0.015/0.024/7.935	0.032/0.039/21.970	0.134/0.172/15.002	0.018/0.029/5.669	0.034/0.052/10.983	0.062/0.097/22.127	0.112/0.176/40.098
EMA-VFI'23 [58]	0.015/0.025/8.358	0.032/0.038/21.395	0.132/0.166/15.186	0.019/0.038/5.882	0.033/0.053/11.051	0.060/0.091/20.679	0.114/0.170/39.051
LDMVFI'24 [10]	0.019/0.044/16.167	0.026/0.035/26.301	0.107 0.153/12.554	0.014/0.024/5.752	0.028/0.053/12.485	0.060/0.114/26.520	0.123/0.204/47.042
Ours†	0.017/0.040/14.447	0.024/0.034/15.335	0.102/0.150/12.623	0.013/0.022/5.737	0.028/0.050/12.569	0.058/0.110/25.567	0.118/0.197/46.088
Ours	**0.009/0.018/7.470**	**0.021/0.032/14.000**	**0.092/0.136/9.220**	**0.012/0.019/4.791**	**0.022/0.039/9.039**	**0.047/0.091/18.589**	**0.104**/0.184/**36.631**

concatenation of $\mathbf{x}_t, \mathbf{y}, \mathbf{z}$ as input and is trained with Adam optimizer [24] with 10^{-4} learning rate for 30 epochs with a batch size of 64. γ is set to be 5 in the $min - SNR - \gamma$ weighting.

4.2 Datasets and Evaluation Metrics

Training Sets. To ensure a fair comparison with most recent works [1, 6, 12, 17, 19, 23, 31, 32, 40, 45], we train our models in Vimeo 90K triplets dataset [57], which contains 51,312 triplets. We apply random flipping, random cropping to 256×256, temporal order reversing, and random rotation with multiples of 90 degrees as data augmentation.

Test Sets. We select UCF-101 [48], DAVIS [39], SNU-FILM [7], and Middlebury [2] to evaluate our method. UCF-101 and Middlebury consist of relatively low-resolution videos (less than 1K), whereas DAVIS and SNU-FILM consist of relatively high-resolution videos (up to 4K). SNU-FILM consists of four categories with increasing levels of difficulties (i.e. larger motion changes): easy, medium, hard, and extreme.

Evaluation Metrics. Recent works [9, 10, 59] reveal that PSNR and SSIM [54] are sometimes unreliable because they have relatively lower correlation with humans' visual judgments. However, learning-based metrics such as FID [15], LPIPS [59], and FloLPIPS [9] are shown to have a higher correlation with humans' visual judgments in [10, 59]. Moreover, we also experimentally find such inconsistencies between PSNR/SSIM and visual quality, which will be discussed in Section 4.3. Therefore, we select FID, LPIPS, and FloLPIPS as our main evaluation metrics. LPIPS and FID measure similarities or distances in the latent space of deep learning models. FloLPIPS is based on LPIPS but takes the motion change among three frames into consideration. The results in PSNR/SSIM are included in Appendix C.1.

4.3 Experimental Results

Quantitative Results. Our method is compared with recent open-source state-of-the-art VFI methods, such as ABME [38], MCVD [53], VFIformer [31], IFRNet [25], AMT [28], UPR-Net [23], EMA-VFI [58], and LDMVFI [10]. The evaluation is reported in LPIPS/FloLPIPS/FID (lower the better), shown in Table 1. We evaluate VFIformer, IFRNet, AMT, UPR-Net, and EMA-VFI with their provided weights, and other results are from the appendix of LDMVFI [10]. Models with different versions in the number of parameters are chosen to be the largest ones. With the same autoencoder as LDMVFI [10], our method (ours†) achieves better performance than LDMVFI, indicating the effectiveness of our consecutive Brownian Bridge Diffusion. Moreover, with an improved autoencoder, our method (denoted as ours) achieves state-of-the-art performance. It is important to note that we achieve much better FloLPIPS than other SOTAs, indicating our interpolated results achieve stronger motion consistency.

Qualitative Results. In Table 1, our consecutive Brownian Bridge diffusion with the autoencoder in LDMVFI [10] (denoted as our†) generally achieves better quantitative results than LDMVFI, showing our method is effective. We include qualitative visualization in Figure 5 to support this result. Moreover, as mentioned in Section 1, we find that the autoencoder in [10] usually reconstructs overlaid images, and therefore we propose a new method of reconstruction. We provide examples to visualize the reconstruction results with our autoencoder and LDMVFI's autoencoder for comparison, shown in Figure 4. All examples are from SNU-FILM extreme [7], which contains relatively large motion changes in neighboring frames.

We have provided some visual comparisons of our method and recent SOTAs in Figure 1. Our method achieves better visual quality because we have clearer details such as dog skins, cloth with folds, and fences with nets. However, UPR-Net [23] achieves better PSNR/SSIM in all the cropped regions ($5-10\%$ better) than ours, which is highly inconsistent with the visual quality.

4.4 Ablation Studies

As we discussed in Section 3.2, latent-diffusion-based VFI is broken down into two stages, so we have a novel method to analyze the entire model. We conduct an ablation study on the ground truth estimation capability of our consecutive Brownian Bridge diffusion. We compare the evaluation results of decoded images with diffusion-generated latent representation $\hat{\mathbf{x}}$ and ground truth $\mathbf{x}$, which is encoded I_n. The results are shown in Table 2. It is important to note that, fixing inputs as the ground truth, our autoencoder achieves a stronger performance than the autoencoder in LDMVFI [10], indicating the effectiveness of our autoencoder. Also, fixing the autoencoder, our consecutive Brownian Bridge diffusion achieves almost identical performance with the ground truth, indicating its strong capability of ground truth estimation.

Table 2: Ablation studies of autoencoder and ground truth estimation. + GT means we input ground truth x to the decoder part of autoencoder. + BB indicates our consecutive Brownian Bridge diffusion trained with autoencoder of LDMVFI. With our consecutive Brownian Bridge diffusion, the interpolated frame has almost the same performance as the interpolated frame with ground truth latent representation, indicating the strong ground truth estimation capability Our autoencoder also has better performance than LDMVFI [10].

Methods	Middlebury	UCF-101	DAVIS	SNU-FILM			
				easy	medium	hard	extreme
	LPIPS/FloLPIPS/FID	LPIPS/FloLPIPS/FID	LPIPS/FloLPIPS/FID	LPIPS/FloLPIPS/FID	LPIPS/FloLPIPS/FID	LPIPS/FloLPIPS/FID	LPIPS/FloLPIPS/FID
LDMVFI'24 [10]	0.019/0.044/16.167	0.026/0.035/26.301	0.107 0.153/12.554	0.014/0.024/5.752	0.028/0.053/12.485	0.060/0.114/26.520	0.123 0.204/47.042
LDMVFI'24 [10] + BB	0.017/0.040/14.447	0.024/0.034/15.335	0.102/0.150/12.623	0.013/0.022/5.737	0.028/0.050/12.569	0.058/0.110/25.567	0.118/0.197/46.088
LDMVFI'24 [10] + GT	0.017/0.040/14.447	0.024/0.034/15.335	0.102/0.150/12.625	0.013/0.022/5.739	0.028/0.050/12.563	0.058/0.110/25.565	0.118/0.197/46.080
Ours	0.009/0.018/7.470	0.021/0.032/14.000	0.092/0.0136/9.220	0.012/0.019/4.791	0.022/0.039/9.039	0.047/0.091/18.589	0.104/0.184/36.631
Ours + GT	0.009/0.018/7.468	0.021/0.032/14.000	0.092/0.136/9.220	0.012/0.019/4.791	0.022/0.039/9.039	0.047/0.091/18.591	0.104/0.184/36.633

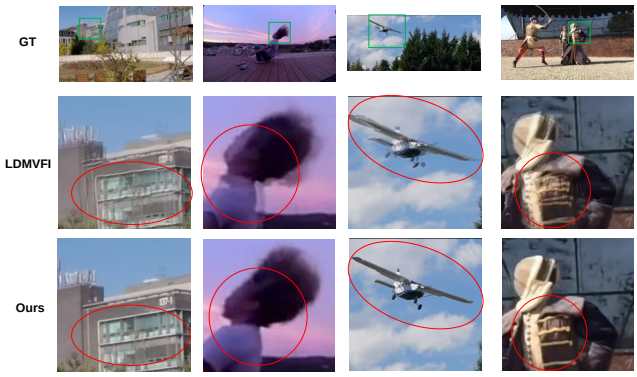

Figure 4: The reconstruction quality of our autoencoder and LDMVFI's autoencoder (decoding with ground truth latent representation x). Images are cropped with green boxes for detailed comparisons. Red circles highlight the details where our method achieves better performance. LDMVFI usually outputs overlaid images while our method does not.

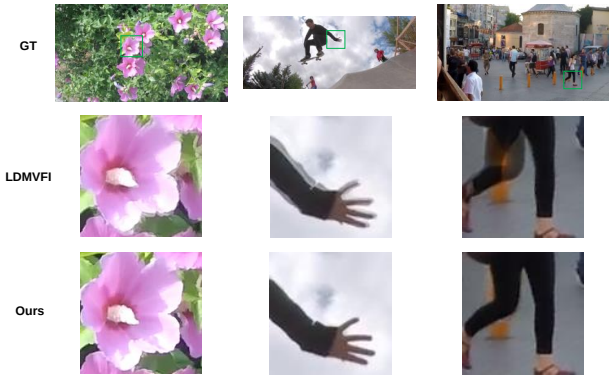

Figure 5: The visual comparison of interpolated results of LDMVFI [10] vs our method with the same autoencoder in LDMVFI (LDMVFI vs our† in Table 1). With the same autoencoder, our method can still achieve better visual quality than LDMVFI, demonstrating the superiority of our proposed consecutive Brownian Bridge diffusion.

However, the conditional generation model in LDMVFI [10] usually underperforms the autoencoder with ground truth inputs. Therefore, our method has a stronger ability in both the autoencoder and ground truth estimation stages. More ablation studies are provided in Appendix C.3.

5 CONCLUSION

In this study, we propose our consecutive Brownian Bridge diffusion Model that better estimates the ground truth latent representation due to its low cumulative variance. We justify its effectiveness with extensive experiments in a wide range of datasets, though we do acknowledge that it requires larger GPU memory (18.55G for one 1080×720 image) than recent non-diffusion VFI methods such as UPR-Net [23](3G). Our method improves when the autoencoder is improved and achieves state-of-the-art performance with a simple yet effective design of the autoencoder, demonstrating its strong potential in the VFI task as a carefully designed autoencoder could potentially boost the performance by a large margin. In addition, we propose a novel method to analyze LDM-based VFI, providing insights for future research: whether future research could be conducted on autoencoder or diffusion model. Therefore, we believe our work will provide unique research directions and insights for diffusion-based video frame interpolation.

REFERENCES

[1] Dawit Mureja Argaw and In So Kweon. 2022. Long-term video frame interpolation via feature propagation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition.*
[2] Simon Baker, Daniel Scharstein, James P Lewis, Stefan Roth, Michael J Black, and Richard Szeliski. 2011. A database and evaluation methodology for optical flow. *International journal of computer vision* (2011).
[3] Georgios Batzolis, Jan Stanczuk, Carola-Bibiane Schönlieb, and Christian Etmann. 2021. Conditional image generation with score-based diffusion models. *arXiv preprint arXiv:2111.13606* (2021).
[4] Zhiqi Chen, Ran Wang, Haojie Liu, and Yao Wang. 2021. PDWN: Pyramid deformable warping network for video interpolation. *IEEE Open Journal of Signal Processing* (2021).
[5] Xianhang Cheng and Zhenzhong Chen. 2020. Video frame interpolation via deformable separable convolution. In *Proceedings of the AAAI Conference on Artificial Intelligence.*

[6] Jinsoo Choi, Jaesik Park, and In So Kweon. 2021. High-quality frame interpolation via tridirectional inference. In *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision*.

[7] Myungsub Choi, Heewon Kim, Bohyung Han, Ning Xu, and Kyoung Mu Lee. 2020. Channel attention is all you need for video frame interpolation. In *Proceedings of the AAAI Conference on Artificial Intelligence*.

[8] Jifeng Dai, Haozhi Qi, Yuwen Xiong, Yi Li, Guodong Zhang, Han Hu, and Yichen Wei. 2017. Deformable convolutional networks. In *Proceedings of the IEEE international conference on computer vision*.

[9] Duolikun Danier, Fan Zhang, and David Bull. 2022. FloLPIPS: A bespoke video quality metric for frame interpolation. In *2022 Picture Coding Symposium (PCS)*. IEEE.

[10] Duolikun Danier, Fan Zhang, and David R. Bull. 2024. LDMVFI: Video Frame Interpolation with Latent Diffusion Models. In *AAAI Conference on Artificial Intelligence*.

[11] Valentin De Bortoli, James Thornton, Jeremy Heng, and Arnaud Doucet. 2021. Diffusion schrödinger bridge with applications to score-based generative modeling. *Advances in Neural Information Processing Systems* (2021).

[12] Saikat Dutta, Arulkumar Subramaniam, and Anurag Mittal. 2022. Non-linear motion estimation for video frame interpolation using space-time convolutions. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*.

[13] John Flynn, Ivan Neulander, James Philbin, and Noah Snavely. 2016. Deepstereo: Learning to predict new views from the world's imagery. In *Proceedings of the IEEE conference on computer vision and pattern recognition*.

[14] Tiankai Hang, Shuyang Gu, Chen Li, Jianmin Bao, Dong Chen, Han Hu, Xin Geng, and Baining Guo. 2023. Efficient diffusion training via min-snr weighting strategy. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*.

[15] Martin Heusel, Hubert Ramsauer, Thomas Unterthiner, Bernhard Nessler, and Sepp Hochreiter. 2017. Gans trained by a two time-scale update rule converge to a local nash equilibrium. *Advances in neural information processing systems* (2017).

[16] Jonathan Ho, Ajay Jain, and Pieter Abbeel. 2020. Denoising diffusion probabilistic models. *Advances in neural information processing systems* (2020).

[17] Ping Hu, Simon Niklaus, Stan Sclaroff, and Kate Saenko. 2022. Many-to-many splatting for efficient video frame interpolation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*.

[18] Zhaoyang Huang, Xiaoyu Shi, Chao Zhang, Qiang Wang, Ka Chun Cheung, Hongwei Qin, Jifeng Dai, and Hongsheng Li. 2022. Flowformer: A transformer architecture for optical flow. In *European conference on computer vision*.

[19] Zhewei Huang, Tianyuan Zhang, Wen Heng, Boxin Shi, and Shuchang Zhou. 2022. Real-time intermediate flow estimation for video frame interpolation. In *European Conference on Computer Vision*.

[20] Zhewei Huang, Tianyuan Zhang, Wen Heng, Boxin Shi, and Shuchang Zhou. 2022. Real-Time Intermediate Flow Estimation for Video Frame Interpolation. In *Proceedings of the European Conference on Computer Vision (ECCV)*.

[21] Tak-Wai Hui, Xiaoou Tang, and Chen Change Loy. 2018. Liteflownet: A lightweight convolutional neural network for optical flow estimation. In *Proceedings of the IEEE conference on computer vision and pattern recognition*.

[22] Eddy Ilg, Nikolaus Mayer, Tonmoy Saikia, Margret Keuper, Alexey Dosovitskiy, and Thomas Brox. 2017. Flownet 2.0: Evolution of optical flow estimation with deep networks. In *Proceedings of the IEEE conference on computer vision and pattern recognition*.

[23] Xin Jin, Longhai Wu, Jie Chen, Youxin Chen, Jayoon Koo, and Cheul-hee Hahm. 2023. A unified pyramid recurrent network for video frame interpolation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*.

[24] Diederik P. Kingma and Jimmy Ba. 2015. Adam: A Method for Stochastic Optimization. In *International Conference on Learning Representations*.

[25] Lingtong Kong, Boyuan Jiang, Donghao Luo, Wenqing Chu, Xiaoming Huang, Ying Tai, Chengjie Wang, and Jie Yang. 2022. IFRNet: Intermediate Feature Refine Network for Efficient Frame Interpolation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*.

[26] Hyeongmin Lee, Taeoh Kim, Tae-young Chung, Daehyun Pak, Yuseok Ban, and Sangyoun Lee. 2020. Adacof: Adaptive collaboration of flows for video frame interpolation. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*.

[27] Bo Li, Kaitao Xue, Bin Liu, and Yu-Kun Lai. 2023. BBDM: Image-to-image translation with Brownian bridge diffusion models. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*.

[28] Zhen Li, Zuo-Liang Zhu, Ling-Hao Han, Qibin Hou, Chun-Le Guo, and Ming-Ming Cheng. 2023. AMT: All-Pairs Multi-Field Transforms for Efficient Frame Interpolation. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*.

[29] Cheng Lu, Yuhao Zhou, Fan Bao, Jianfei Chen, Chongxuan Li, and Jun Zhu. 2022. Dpm-solver: A fast ode solver for diffusion probabilistic model sampling in around 10 steps. *Advances in Neural Information Processing Systems* (2022).

[30] Cheng Lu, Yuhao Zhou, Fan Bao, Jianfei Chen, Chongxuan Li, and Jun Zhu. 2022. Dpm-solver++: Fast solver for guided sampling of diffusion probabilistic models. *arXiv preprint arXiv:2211.01095* (2022).

[31] Liying Lu, Ruizheng Wu, Huaijia Lin, Jiangbo Lu, and Jiaya Jia. 2022. Video frame interpolation with transformer. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*.

[32] Simon Niklaus and Feng Liu. 2018. Context-aware synthesis for video frame interpolation. In *Proceedings of the IEEE conference on computer vision and pattern recognition*.

[33] Simon Niklaus and Feng Liu. 2020. Softmax splatting for video frame interpolation. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*.

[34] Simon Niklaus, Long Mai, and Feng Liu. 2017. Video frame interpolation via adaptive convolution. In *Proceedings of the IEEE conference on computer vision and pattern recognition*.

[35] Simon Niklaus, Long Mai, and Feng Liu. 2017. Video frame interpolation via adaptive separable convolution. In *Proceedings of the IEEE international conference on computer vision*.

[36] Bernt Oksendal. 2013. *Stochastic differential equations: an introduction with applications*. Springer Science & Business Media.

[37] Junheum Park, Jintae Kim, and Chang-Su Kim. 2023. BiFormer: Learning Bilateral Motion Estimation via Bilateral Transformer for 4K Video Frame Interpolation. In *Computer Vision and Pattern Recognition*.

[38] Junheum Park, Chul Lee, and Chang-Su Kim. 2021. Asymmetric Bilateral Motion Estimation for Video Frame Interpolation. In *International Conference on Computer Vision*.

[39] Federico Perazzi, Jordi Pont-Tuset, Brian McWilliams, Luc Van Gool, Markus Gross, and Alexander Sorkine-Hornung. 2016. A Benchmark Dataset and Evaluation Methodology for Video Object Segmentation. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*.

[40] Markus Plack, Karlis Martins Briedis, Abdelaziz Djelouah, Matthias B Hullin, Markus Gross, and Christopher Schroers. 2023. Frame Interpolation Transformer and Uncertainty Guidance. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*.

[41] Robin Rombach, Andreas Blattmann, Dominik Lorenz, Patrick Esser, and Björn Ommer. 2022. High-resolution image synthesis with latent diffusion models. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*.

[42] Sheldon M Ross. 1995. *Stochastic processes*.

[43] Yuyang Shi, Valentin De Bortoli, Andrew Campbell, and Arnaud Doucet. 2024. Diffusion Schrödinger bridge matching. *Advances in Neural Information Processing Systems* (2024).

[44] Zhihao Shi, Xiaohong Liu, Kangdi Shi, Linhui Dai, and Jun Chen. 2021. Video frame interpolation via generalized deformable convolution. *IEEE transactions on multimedia* (2021).

[45] Li Siyao, Shiyu Zhao, Weijiang Yu, Wenxiu Sun, Dimitris Metaxas, Chen Change Loy, and Ziwei Liu. 2021. Deep animation video interpolation in the wild. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*.

[46] Jiaming Song, Chenlin Meng, and Stefano Ermon. 2021. Denoising Diffusion Implicit Models. In *International Conference on Learning Representations*.

[47] Yang Song, Jascha Sohl-Dickstein, Diederik P Kingma, Abhishek Kumar, Stefano Ermon, and Ben Poole. 2021. Score-Based Generative Modeling through Stochastic Differential Equations. In *International Conference on Learning Representations*.

[48] Khurram Soomro, Amir Roshan Zamir, and Mubarak Shah. 2012. UCF101: A dataset of 101 human actions classes from videos in the wild. *arXiv preprint arXiv:1212.0402* (2012).

[49] Deqing Sun, Xiaodong Yang, Ming-Yu Liu, and Jan Kautz. 2018. Pwc-net: Cnns for optical flow using pyramid, warping, and cost volume. In *Proceedings of the IEEE conference on computer vision and pattern recognition*.

[50] Zachary Teed and Jia Deng. 2020. Raft: Recurrent all-pairs field transforms for optical flow. In *European Conference on Computer Vision*.

[51] Zhengzhong Tu, Hossein Talebi, Han Zhang, Feng Yang, Peyman Milanfar, Alan Bovik, and Yinxiao Li. 2022. Maxvit: Multi-axis vision transformer. In *European conference on computer vision*.

[52] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. *Advances in neural information processing systems* (2017).

[53] Vikram Voleti, Alexia Jolicoeur-Martineau, and Chris Pal. 2022. Mcvd-masked conditional video diffusion for prediction, generation, and interpolation. *Advances in neural information processing systems* (2022).

[54] Zhou Wang, Alan C Bovik, Hamid R Sheikh, and Eero P Simoncelli. 2004. Image quality assessment: from error visibility to structural similarity. *IEEE transactions on image processing* (2004).

[55] Philippe Weinzaepfel, Thomas Lucas, Vincent Leroy, Yohann Cabon, Vaibhav Arora, Romain Brégier, Gabriela Csurka, Leonid Antsfeld, Boris Chidlovskii, and Jérôme Revaud. 2023. CroCo v2: Improved Cross-view Completion Pre-training for Stereo Matching and Optical Flow. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*.

[56] Chao-Yuan Wu, Nayan Singhal, and Philipp Krahenbuhl. 2018. Video compression through image interpolation. In *Proceedings of the European conference on*

computer vision (ECCV).

[57] Tianfan Xue, Baian Chen, Jiajun Wu, Donglai Wei, and William T Freeman. 2019. Video Enhancement with Task-Oriented Flow. *International Journal of Computer Vision (IJCV)* (2019).

[58] Guozhen Zhang, Yuhan Zhu, Haonan Wang, Youxin Chen, Gangshan Wu, and Limin Wang. 2023. Extracting motion and appearance via inter-frame attention for efficient video frame interpolation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*.

[59] Richard Zhang, Phillip Isola, Alexei A Efros, Eli Shechtman, and Oliver Wang. 2018. The Unreasonable Effectiveness of Deep Features as a Perceptual Metric. In *CVPR*.

[60] Linqi Zhou, Aaron Lou, Samar Khanna, and Stefano Ermon. 2024. Denoising Diffusion Bridge Models. In *The Twelfth International Conference on Learning Representations*.

A FORMULA DERIVATION

A.1 Consecutive Brownian Bridge

For $0 < t < h$, if we have $s > t$, then the Markov property of the Wiener process produces:

$$W_s|(W_0, W_t, W_h) = W_s|(W_t, W_h)$$

Applying in our setting, this becomes: $W_t|W_T = \mathbf{x}, W_{2T} = \mathbf{z}$ for $t > T$. Note that only the variance of the Wiener process is related to time, and the variance of general Brownian Bridge $W_t|(W_{t_1}, W_{t_2})$ is $\frac{(t_2-t)(t-t_1)}{t_2-t_1}$. If we add any value simultaneously to t_1, t_2, t, the variance is unchanged. Therefore, we can subtract T in time to get $W_s|W_0 = \mathbf{x}, W_T = \mathbf{z}$, where $s = t - T$.

If we have $s < t$, then it is important to know that $tW_{t^{-1}}$ is a Wiener process with the same distribution with W_t [36]. We can add a small ϵ to time and use such transformation to obtain:

$$\begin{aligned}
& W_s|(W_0, W_t, W_h) \\
= \; & W_{s+\epsilon}|(W_\epsilon, W_{t+\epsilon}, W_{h+\epsilon}) \\
= \; & (s+\epsilon)W_{(s+\epsilon)^{-1}}|\epsilon W_{\epsilon^{-1}}, (t+\epsilon)W_{(t+\epsilon)^{-1}}, (h+\epsilon)W_{(h+\epsilon)^{-1}} \\
= \; & (s+\epsilon)W_{(s+\epsilon)^{-1}}|\epsilon W_{\epsilon^{-1}}, (t+\epsilon)W_{(t+\epsilon)^{-1}} \\
= \; & W_s|(W_0, W_t)
\end{aligned}$$

In our method, this becomes $W_t|W_0 = \mathbf{y}, W_T = \mathbf{x}$. The distribution is $\mathcal{N}(\frac{t}{T}\mathbf{y} + (1 - \frac{t}{T})\mathbf{x}), \frac{t(T-t)}{T}\mathbf{I})$. Now, let's consider another process defined as $W_s|W_0 = \mathbf{x}, W_T = \mathbf{y}$. The distribution is easy to derive: $\mathcal{N}(\frac{s}{T}\mathbf{x} + (1 - \frac{s}{T})\mathbf{y}), \frac{s(T-s)}{T}\mathbf{I})$. With simple algebra, we can find that when $s = T - t$, the two distributions are equal. Thus, we finish the derivation of the distribution of consecutive Brownian Bridge.

A.2 Cumulative Variance

We denote $\mathbf{z}$ as standard Gaussian distribution. In DDPM [16], $\mathbf{x}_{t-1} = \frac{1}{\sqrt{1-\beta_t}}\left(\mathbf{x}_t - \frac{\beta_t}{\sqrt{1-\alpha_t}}\epsilon_\theta\right) + \sqrt{\hat{\beta}_t}\mathbf{z}$. At the first step of generation, since $\mathbf{x}_T \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$ and $0 < \beta_t < 1$, we have:

$$\begin{aligned}
Var(\mathbf{x}_{T-1}) &= Var\left(\frac{1}{\sqrt{1-\beta_t}}\left(\mathbf{x}_T - \frac{\beta_t}{\sqrt{1-\alpha_t}}\epsilon_\theta\right) + \sqrt{\hat{\beta}_t}\mathbf{z}\right) \\
&> Var\left(\frac{1}{\sqrt{1-\beta_t}}\mathbf{x}_T + \sqrt{\hat{\beta}_t}\mathbf{z}\right) \\
&> 1 + \hat{\beta}_t
\end{aligned}$$

Since ϵ_θ takes random input, it has a positive variance. The following sampling steps have fixed inputs x_t, so the variance only contains $\hat{\beta}_t$. Therefore, the cumulative variance is larger than $1 + \sum_t \hat{\beta}_t$, corresponding to **11.036** in real experiments. However, in our method, we have $\mathbf{x}_{t-\Delta_t} = \mathbf{x}_t - \frac{\Delta_t}{t}\epsilon_\theta + \sqrt{\frac{(t-\Delta_t)\Delta_t}{t}}\mathbf{z}$, and $\mathbf{x}_T$ is deterministic, we have:

$$\begin{aligned}
Var(x_{t-\Delta_t}) &= Var\left(\mathbf{x}_t - \frac{\Delta_t}{t}\epsilon_\theta + \sqrt{\frac{(t-\Delta_t)\Delta_t}{t}}\mathbf{z}\right) \\
&= Var\left(\sqrt{\frac{(t-\Delta_t)\Delta_t}{t}}\mathbf{z}\right) \\
&< \Delta_t
\end{aligned}$$

Since ϵ_θ takes fixed inputs, it has no variance. The cumulative variance is smaller than $\sum_t \Delta_t = T$, corresponding to **2** in our experiments. We mentioned this result in Section 3.4 in our main paper.

B CONNECTION WITH DIFFUSION SDES

Our method can be easily written in score-based SDE [3, 47, 60]. The forward process of score-based SDEs is defined as:

$$d\mathbf{x} = f(x,t)dt + g(t)d\mathbf{w}. \tag{18}$$

$f(x,t)$ is the drift term, and $g(t)$ is the dispersion term. $\mathbf{w}$ denotes the standard Wiener process. The corresponding reversed SDE is defined as:

$$d\mathbf{x} = \left[f(x,t) - g(t)^2\nabla_{\mathbf{x}} log p_t(\mathbf{x})\right]dt + g(t)d\bar{\mathbf{w}}. \tag{19}$$

The conditional generation counterpart is defined as:

$$d\mathbf{x} = \left\{f(x,t) - g(t)^2\nabla_{\mathbf{x}}[log p_t(\mathbf{x}) + log p_t(\mathbf{y}|\mathbf{x})]\right\}dt + g(t)d\bar{\mathbf{w}}. \tag{20}$$

The term $\mathbf{y}$ is the conditional control for generation. Moreover, there exists a deterministic ODE trajectory (probability flow ODE) with the same marginal distribution $p_t(x)$ with Eq. (19) [47]:

$$d\mathbf{x} = \left[f(x,t) - \frac{1}{2}g(t)^2\nabla_{\mathbf{x}} log p_t(\mathbf{x})\right]dt. \tag{21}$$

Therefore, the it suffices to train a neural network s_θ estimating $\nabla_{\mathbf{x}} log p_t(\mathbf{x})$ [47]. Indeed, Brownian Bridge can be written in SDE form by [36]:

$$d\mathbf{x} = \frac{\mathbf{y} - \mathbf{x}_t}{T - t}dt + d\mathbf{w}. \tag{22}$$

$\mathbf{y}$ is another endpoint of the Brownian Bridge. The reversed SDE is defined as:

$$d\mathbf{x} = \left[\frac{\mathbf{y} - \mathbf{x}_t}{T - t} - \nabla_{\mathbf{x}} log p_t(\mathbf{x})\right]dt + d\bar{\mathbf{w}}. \tag{23}$$

By our formulation, our proposed method is compatible with score-based SDEs. Moreover, compared with conditional SDEs in Eq. (20), this formulation does not include $log p_t(\mathbf{y}|\mathbf{x})$ which needs estimation.

C ADDITIONAL RESULTS

C.1 Quantitative Results

We provide the evaluation results in PSNR/SSIM in Table 4. Though our method does not have state-of-the-art (but still comparable with SOTAs) performance in PSNR/SSIM, it is due to the **inconsistency** between PSNR/SSIM and visual quality (see Section C.2 and Figure 6). Therefore, we choose LPIPS/FloLPIPS/FID as our main evaluation metrics.

C.2 Qualitative Reults

Inconsistency Between PSNR/SSIM and Visual Quality. We provide some examples to demonstrate the inconsistency between PSNR/SSIM and visual quality, as shown in Figure 6. Our method achieves better visual quality than UPR-Net [23] such as clearer dog skins, clearer cloth with folds, and clearer shoes and fences with nets. However, we did not achieve a satisfactory PSNR/SSIM, which is 5-10% lower than that of UPR-Net.

Figure 6: Visual illustration of the inconsistency between PSNR/SSIM and visual quality. Only images cropped within blue boxes are evaluated with PSNR/SSIM. The red circles highlight our visual quality. Our method generates images with better visual quality, but the PSNR/SSIM is much lower.

Figure 7: Visual comparison between our sampling and DDIM sampling with 5 steps generation. They achieve almost identical results (with very large PSNR). The residual is the absolute difference between the two images. Black means 0 difference, and almost everywhere is black.

Additional Qualitative Comparisons. In addition, we provide more qualitative comparisons between our method and LDMVFI [10] in Figure 8 and qualitative comparisons between our method and recent SOTAs in Figure 9. All examples are selected from SNU-FILM extreme [7].

Multi-frame Interpolation. We provide qualitative results of multi-frame interpolation of our methods and LDMVFI [10]. Multi-frame interpolation is achieved in a bisection manner. We first interpolate $I_{0.5}$ with I_0, I_1, and then we interpolate $I_{0.25}$ with $I_0, I_{0.5}$ and $I_{0.75}$ with $I_{0.5}, I_1$. More frames can be interpolated in this manner. We interpolate 7 frames between two I_0, I_1, and the visual comparisons are presented in Figure 10. All examples are selected from SNU-FILM hard [7]. Additional video demos are shown on our GitHub page: https://zonglinl.github.io/videointerp. Due to the bisection-like multi-frame interpolation method, the multi-frame interpolation results largely depends on the first step of interpolation ($I_{0.5}$). If $I_{0.5}$ achieves good quality, then the relative motion in the second step (interpolating $I_{0.25}, I_{0.75}$) is easy to achieve high quality because the motion changes become smaller. However, if the interpolation quality is not good at the first step, then later steps will not achieve good quality because such an unsatisfactory quality will be transmitted. LDMVFI 8 tends to generate overlaid or distorted $I_{0.5}$, resulting in unsatisfactory multi-frame interpolation results. We largely alleviate this problem, resulting in much better and more realistic interpolated videos.

Table 3: Ablation study on the number of sampling steps. This experiment is conducted on SNU-FILM extreme subset [7].

Number of steps	LPIPS	FloLPIPS	FID
200	0.110	0.184	36.632
100	0.110	0.184	36.631
50	0.110	0.184	36.631
20	0.110	0.184	36.632
5	0.110	0.184	36.632

Inference Time. With one Nvidia RTX 8000 GPU, our method generates a 720×1280 image with approximately 1.2 seconds with 18G GPU memory. The inference speed is similar to recent SOTAs such as LDMVFI [10] (1.2s) and UPR-Net-Large [23] (1.15s), but diffusion-based methods require much more memory (LDMVFI requires 22G while UPR-Net requires 3G).

C.3 Ablation Studies

Number of Sampling Steps. We investigate how the number of sampling steps will impact the performance. This ablation study is conducted on SNU-FILM extreme subset [7], shown in Table 3. We observe that the performance remains almost identical. The reason could be the relatively small differences between neighboring frames. Our method does not convert random noise to images like DDPM [16]. Instead, we convert one image to its neighboring frames, so we do not need to generate details from random noises. Instead, we change details from existing details, and therefore it may not need many steps to generate.

DDIM Sampling. As we claimed, our formulation does not need DDIM [46] sampling to accelerate. We compare our sampling with DDIM sampling with $\eta = 0$ in 5 sampling steps for comparison. The visual result is shown in Figure 7. There is almost no difference between the output of our sampling method and DDIM sampling, indicating that we do not require such a method to accelerate sampling.

Table 4: Quantitative results (PSNR/SSIM) on test datasets (the higher the better). † means we evaluate our consecutive Brownian Bridge diffusion (trained on Vimeo 90K [57]) with autoencoder provided by LDMVFI [10].

Methods	Middlebury	UCF-101	DAVIS	SNU-FILM			
				easy	medium	hard	extreme
	PSNR/SSIM	PSNR/SSIM	PSNR/SSIM	PSNR/SSIM	PSNR/SSIM	PSNR/SSIM	PSNR/SSIM
ABME'21 [38]	37.639/0.986	35.380/0.970	26.861/0.865	39.590/0.990	35.770/0.979	30.580/0.936	25.430/0.864
MCVD'22 [53]	20.539/0.820	18.775/0.710	18.946/0.705	22.201/0.828	21.488/0.812	20.314/0.766	18.464/0.694
VFIformer'22 [31]	38.438/0.987	35.430/0.970	26.241/0.850	40.130/0.991	36.090/0.980	30.670/0.938	25.430/0.864
IFRNet'22 [25]	36.368/0.983	35.420/0.967	27.313/0.877	40.100/0.991	36.120/0.980	30.630/0.937	25.270/0.861
AMT'23 [28]	38.395/0.988	35.450/0.970	27.234/0.877	39.880/0.991	36.120/0.981	30.780/0.939	25.430/0.865
UPR-Net'23 [23]	38.065/0.986	35.470/0.970	26.894/0.870	40.440/0.991	36.290/0.980	30.860/0.938	25.630/0.864
EMA-VFI'23 [58]	38.526/0.988	35.480/0.970	27.111/0.871	39.980/0.991	36.090/0.980	30.940/0.939	25.690/0.866
LDMVFI'24 [10]	34.230/0.974	32.160/0.964	25.073/0.819	38.890 0.988	33.975/0.971	29.144/0.911	23.349 0.827
Ours†	34.057/0.970	34.730/0.965	25.446/0.837	38.720/0.988	34.016/0.971	28.556/0.918	23.931/0.837
Ours	36.852/0.983	35.151/0.968	26.391/0.858	39.637/0.990	34.886/0.974	29.615/0.929	24.376/0.848

Figure 8: Additional Qualitative Comparison of our methods and LDMVFI. Images cropped with blue boxes are shown for better-detailed comparison. Our method steadily achieves better visual quality.

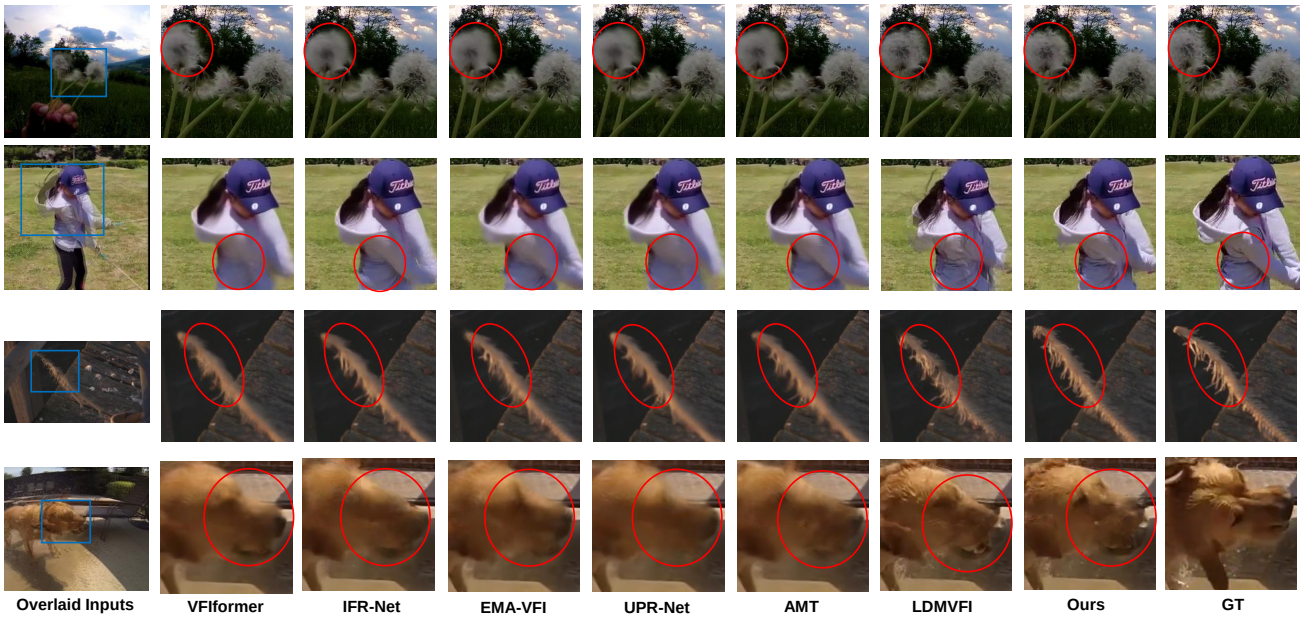

Figure 9: Additional Qualitative Comparison of our methods and recent SOTAs. Only images within the blue box are displayed for better-detailed comparison.

Figure 10: Multi-frame interpolation results. LDMVFI usually interpolates distorted or overlaid images while ours does not. Images with red and blue borders are displayed to show details. Our method corresponds to the blue border while LDMVFI corresponds to the red. Green circles highlight the detail where our performance is better.