VideoMark: A Distortion-Free Robust Watermarking Framework for Video Diffusion Models

Xuming Hu[1*], **Hanqian Li**[1*], **Jungang Li**[1*], **Yu Huang**[1], **Aiwei Liu** [2†]
[1] AI Thrust, Hong Kong University of Science and Technology (Guangzhou), China
[2]School of Software, BNRist, Tsinghua University, China

Abstract

This work introduces **VideoMark**, a distortion-free robust watermarking framework for video diffusion models. As diffusion models excel in generating realistic videos, reliable content attribution is increasingly critical. However, existing video watermarking methods often introduce distortion by altering the initial distribution of diffusion variables and are vulnerable to temporal attacks, such as frame deletion, due to variable video lengths. VideoMark addresses these challenges by employing a **pure pseudorandom initialization** to embed watermarks, avoiding distortion while ensuring uniform noise distribution in the latent space to preserve generation quality. To enhance robustness, we adopt a frame-wise watermarking strategy with pseudorandom error correction (PRC) codes, using a fixed watermark sequence with randomly selected starting indices for each video. For watermark extraction, we propose a Temporal Matching Module (TMM) that leverages edit distance to align decoded messages with the original watermark sequence, ensuring resilience against temporal attacks. Experimental results show that VideoMark achieves higher decoding accuracy than existing methods while maintaining video quality comparable to watermark-free generation. The watermark remains imperceptible to attackers without the secret key, offering superior invisibility compared to other frameworks. VideoMark provides a practical, training-free solution for content attribution in diffusion-based video generation. Code and data are available at Project Page.

1 Introduction

In recent years, diffusion models have revolutionized the landscape of AI-generated content, emerging as the state-of-the-art technology for image and video generation [8, 9, 20]. These models can create highly realistic content that is increasingly indistinguishable from human-created media [19]. The rapid advancement in generation quality has created an urgent need to track and attribute AI-generated content, particularly given growing concerns about copyright infringement and potential misuse [1, 27]. To address these challenges, watermarking techniques have emerged as a crucial solution, providing a reliable mechanism for content traceability and authentication in the era of AI-generated media.

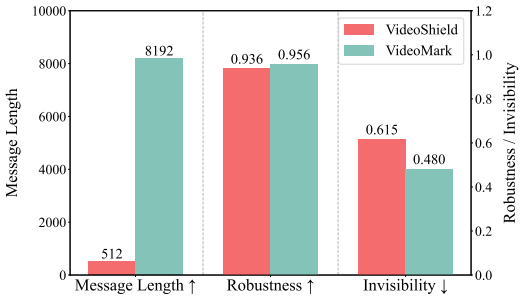

Figure 1: VideoMark outperforms VideoShield across three key metrics: message length, robustness, and invisibility.

[*]Equal Contribution.
[†]Corresponding author.

Preprint. Under review.

Traditional watermarking methods for both images and videos typically operate as **post-processing techniques**, where watermarks are embedded after content generation [16, 28]. These methods not only require additional computational overhead but also suffer from limited generalization capabilities. Recent research has shifted towards embedding watermarks during the generation process itself. Leveraging the **reversibility of DDIM** [21] in diffusion models, several methods have achieved notable success in the image domain by manipulating the initial Gaussian noise—such as Tree-Ring [25]—or by embedding messages directly into the noise distribution, as demonstrated by approaches like Gaussian Shading [26] and PRC-Watermark [7].

However, directly adapting image watermarking techniques to the video domain presents unique challenges. First, video DDIM inversion yields **lower accuracy** than image-based methods. And as a result, methods like VideoShield [10] repeat watermark patterns in initial noise to enhance detection, but these **compromise video quality and watermark invisibility**. Second, watermark robustness suffers against temporal attacks like frame deletion or reordering, because treating videos as a single entity fails to localize watermarks temporally. Third, variable video lengths pose difficulties for algorithms relying on fixed noise initialization, limiting scalability.

To address video watermarking challenges, we first define essential characteristics for our proposed watermark. Primarily, the watermark embedded within the initial latent noise must cause negligible perturbation to the original noise space. Secondly, our approach involves inserting unique watermarks into individual frames. Beyond this per-frame embedding, we must also establish a temporal relationship for these watermarks across consecutive frames to improve robustness.

With these needs in mind, we introduce **VideoMark**, a distortion free and unbiased watermarking framework designed for video diffusion models. To achieve an imperceptible watermark that preserves the original noise characteristics, VideoMark utilizes pseudorandom error correction (PRC) codes [4]. These codes map the watermark bits directly onto the initialized Gaussian noise for every frame. This specific design ensures the watermark integrates seamlessly, thus fulfilling our first design goal. To enable frame-specific watermarking, VideoMark handles each frame's watermark independently while embedding a sequential logic. Specifically, we generate an extended watermark message sequence. For each video, a random starting position within this master sequence initializes the first frame's watermark, and subsequent frames derive their watermarks sequentially. This aligns with our second design objective, facilitating both individualized frame watermarking and temporal coherence.

To accurately extract the watermark, we propose a temporal matching module (TMM), which uses edit distance to align the decoded message with the embedded watermark sequence, thereby improving decoding accuracy. Even when videos undergo temporal attacks such as frame deletion, TMM ensures the robustness and reliability of the watermark.

In our experiments, we evaluate the effectiveness of our watermarking framework across different video diffusion models, demonstrating high decoding accuracy, high-quality generated videos, and strong invisibility. Our watermark achieves higher decoding accuracy compared to VideoShield, which is currently the state-of-the-art watermarking approach for video diffusion models. Additionally, our watermark achieves the best video quality on both the objective video evaluation benchmark VBench[11] and subjective assessments, maintaining parity with watermark-free videos. Importantly, our watermark remains undetectable to attackers without the key, ensuring strong imperceptibility compared to other watermarking frameworks.

In summary, the contributions of this work are summarized as follows:

- We propose VideoMark, which leverages pseudo-random Gaussian space initialization to achieve undetectable watermarking in video diffusion models.
- We introduce a frame-wise watermarking strategy with extended message sequences, solving the challenge of variable-length videos and temporal attacks.
- Our extensive experiments demonstrate that VideoMark achieves higher decoding accuracy than existing methods while maintaining video quality on par with watermark-free generation across various video diffusion models and attack scenarios.

2 Preliminaries

2.1 Diffusion Models

Diffusion models generate content through an iterative denoising process. Given a noise schedule $\beta_t t = 1^T$, the forward process gradually adds noise to data $\mathbf{x_0}$:

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

The reverse process is learned to gradually denoise from $\mathbf{x}T \sim \mathcal{N}(0, \mathbf{I})$ to generate content. While DDPM [8] introduces stochasticity in each denoising step, DDIM [21] provides an approximately invertible deterministic sampling process:

$$\mathbf{x}_{t-1} = \sqrt{\alpha_{t-1}}\left(\frac{\mathbf{x}_t - \sqrt{1-\alpha_t}\boldsymbol{\epsilon}_\theta(\mathbf{x}_t, t)}{\sqrt{\alpha_t}}\right) + \sqrt{1-\alpha_{t-1}}\boldsymbol{\epsilon}_\theta(\mathbf{x}_t, t) \tag{2}$$

This deterministic reversibility enables control over the generation process through manipulation of the initial noise.

2.2 Pseudorandom Codes (PRC)

A PRC is a coding scheme that maps messages to statistically random-looking codewords. We adopt the construction from Christ and Gunn [4], which provides security based on the hardness of the Learning Parity with Noise (LPN) problem.

The PRC framework consists of three core algorithms:

- KeyGen$(n, m, \text{fpr}, t) \to$ key: Generates a key for encoding m-bit messages into n-bit codewords with sparsity parameter t
- Encode$(\text{key}, \mathbf{m}) \to \mathbf{c}$: Maps message $\mathbf{m}$ to codeword $\mathbf{c} \in \{-1, 1\}^n$
- Decode$(\text{key}, \mathbf{s}) \to \mathbf{m}$ or $\emptyset$: Recovers message from potentially corrupted signal $\mathbf{s} \in [-1, 1]^n$

Our implementation supports soft decisions on recovered bits, optimized for robust watermarking (see Appendix A.1 for details).

2.3 Generative Video Watermarking and Threat Model

Diffusion-based video watermarking involves three key functions in the watermarking process:

1. **Generation**: $V = \mathcal{G}(m, k)$, where $\mathcal{G}$ generates a watermarked video V by embedding message m using secret key k during the diffusion process.

2. **Decoding**: $\hat{m} = \mathcal{D}(V)$, where $\mathcal{D}$ extracts the watermark message $\hat{m}$ from the given video V.

3. **Detection**: $\{p, d\} = \mathcal{T}(m, \hat{m})$, where $\mathcal{T}$ compares the original message m with the decoded message $\hat{m}$. This function outputs a p-value p and a boolean decision d indicating whether the distance between m and $\hat{m}$ is significantly smaller than that between m and a random message.

We consider active adversaries who may perform various modifications on the watermarked video to remove or corrupt the embedded watermark. These include:

- **Temporal Attacks**: Frame drop, insert, or swap, which disrupts the temporal structure.
- **Spatial Attacks**: Frame-wise manipulations such as Gaussian blurring, colour jittering, and resolution compression, which aim to distort the watermark signal by degrading the visual content of individual frames.

Our framework aims to be robust against these attacks while ensuring the watermark remains imperceptible and the video quality is preserved.

3 Proposed Method

In this section, we provide a detailed explanation of the proposed unbiased watermarking method in video diffusion models. Specifically, in Section 3.1, we detail the process the watermark generation. In Section 3.2, we introduce the watermark extraction process.

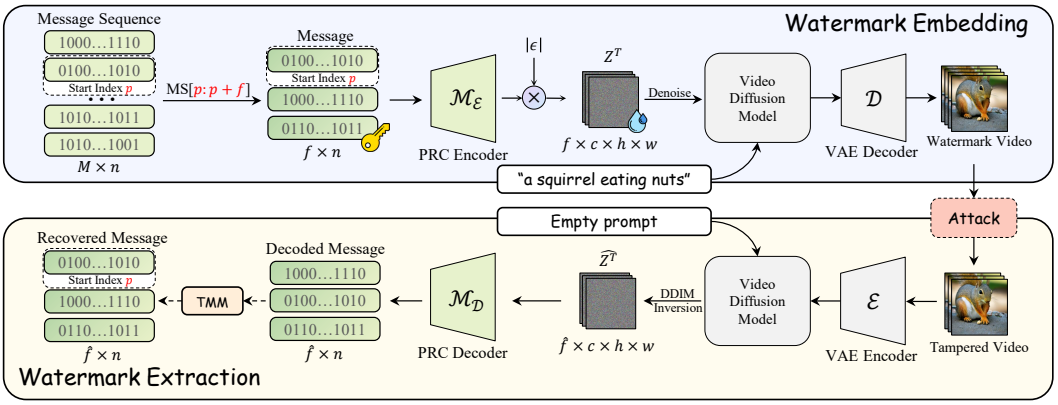

Figure 2: The overall framework of VideoMark. In the watermark embedding phase, ϵ is the standard Gaussian noise sampled randomly. In the I2V task, during the watermark extraction phase, we use the first frame of the video as the prompt to predict the initialization noise.

Algorithm 1 Watermarked Video Generation

Input: PRC-key k, frames f, channels c, height h, width w, message m, video diffusion model $\mathcal{M}$, variational autoencoder (VAE) decoder $\mathcal{D}$
Output: watermarked video V
Generate extended message sequence M longer than maximum supported length
Randomly select starting position p in M
for $i \in 1, ..., f$ **do**
    Get frame message m_i from M starting from position $p + i$
    Applying **Encode** to sample a PRC codeword $c_i \in \{-1, 1\}^{c \times h \times w}$ using m_i
    Sample $\epsilon_i \sim \mathcal{N}(0, 1) \in \mathbb{R}^{c \times h \times w}$
    Compute $\hat{\epsilon_i} \leftarrow c_i \cdot |\epsilon_i|$
end for
Generate watermarked video V by denoising latent space $\hat{\epsilon}$ with diffusion model $\mathcal{M}$ and decoding it with VAE decoder $\mathcal{D}$

3.1 Watermark Generation

In this section, we introduce the watermark generation process. VideoMark achieves high invisibility and video quality in diffusion-based watermarking by initializing each frame with pseudo-random Gaussian noise using PRC, followed by DDIM denoising[21] and VAE decoding [13]. To enhance diversity and adapt to varying video lengths, we employ an extended message list with a random start index.

Prior watermarking methods (e.g. VideoShield [10]) often repeat identical noise patterns, compromising pseudo-randomness, reducing watermark bit capacity, and degrading invisibility and video quality. VideoMark addresses this by generating frame-specific pseudo-random initializations. For a video with f frames, dimensions $c \times h \times w$ (channels, height, width), and a message bit m'_i per frame, the process is as follows. For each frame $i \in \{1, \ldots, f\}$, we sample Gaussian noise $\boldsymbol{\epsilon}_i \sim \mathcal{N}(0, \mathbf{I}) \in \mathbb{R}^{c \times h \times w}$. Using a PRC key k, we encode m'_i to obtain a codeword $\mathbf{c}_i = \text{Encode}(k, m'_i) \in \{-1, 1\}^{c \times h \times w}$. The watermarked noise is:

$$\hat{\boldsymbol{\epsilon}}_i = \mathbf{c}_i \cdot |\boldsymbol{\epsilon}_i|,$$

where $\mathbf{c}_i$ modulates the sign of $\boldsymbol{\epsilon}_i$, preserving its magnitude. The noise sequence $\hat{\boldsymbol{\epsilon}} = [\hat{\boldsymbol{\epsilon}}_1, \ldots, \hat{\boldsymbol{\epsilon}}_f]$ is denoised using a DDIM diffusion model $\mathcal{M}$. For each frame, DDIM iterates over T steps:

$$\hat{\boldsymbol{z}}_i^{(t-1)} = \sqrt{\alpha_{t-1}} \left(\frac{\hat{\boldsymbol{z}}_i^{(t)} - \sqrt{1 - \alpha_t} \boldsymbol{\epsilon}_\theta(\hat{\boldsymbol{z}}_i^{(t)}, t)}{\sqrt{\alpha_t}} \right) + \sqrt{1 - \alpha_{t-1}} \boldsymbol{\epsilon}_\theta(\hat{\boldsymbol{z}}_i^{(t)}, t), \tag{3}$$

with $\hat{\boldsymbol{z}}_i^{(T)} = \hat{\boldsymbol{\epsilon}}_i$, producing latent $\hat{\boldsymbol{z}}_i^{(0)}$. The VAE decoder $\mathcal{D}$ generates the watermarked video $V = [\mathcal{D}(\hat{\boldsymbol{z}}_1^{(0)}), \ldots, \mathcal{D}(\hat{\boldsymbol{z}}_f^{(0)})]$.

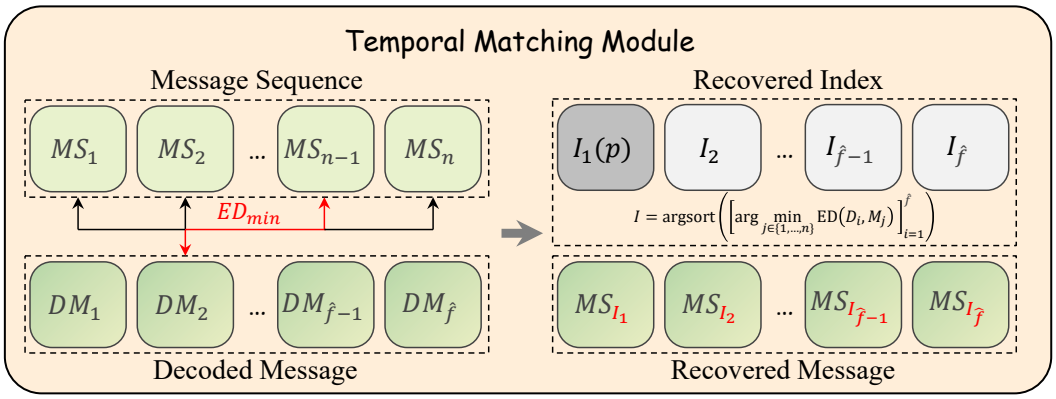

Figure 3: The framework of Temporal Matching Module. ED$(\cdot)$ denotes the edit distance function. In the Recovered Index list, I_1 represents the selected starting position p.

To adapt to videos of varying lengths and increase diversity, we generate an extended message list $M = [m_1, \ldots, m_L]$, where $L > f_{\text{max}}$ and f_{max} is the maximum supported frame count. For each video, we sample a start index $p \sim \text{Uniform}(0, L - f)$, selecting messages $m'_i = M[p + i]$ for $i \in \{1, \ldots, f\}$. These are encoded via PRC to produce $\mathbf{c}_i$. The random start index ensures diverse initializations across videos, improving security and reducing detectable patterns, while supporting arbitrary video lengths. This frame-wise approach resists temporal and spatial attacks.

The pipeline is shown in Figure 2 and detailed in Algorithm 1.

3.2 Watermark Extraction

In this section, we present our watermark extraction process which consists of three key functions: decoding, detection, and recovery. This approach effectively handles various attacks that may disrupt the video structure.

Decoding Function $\hat{m} = \mathcal{D}(V)$ extracts the embedded message from a watermarked video V with f frames. We first recover the approximate initial noise for each frame using the DDIM inverse process:

$$\tilde{\boldsymbol{\epsilon}}_i = \mathcal{M}^{-1}(V_i) \quad \text{for } i \in \{1, ..., f\} \tag{4}$$

Then, we decode each frame's message bit using the sign pattern of the recovered noise:

$$\hat{m}_i = \text{PRC.Decode}(k, \text{sign}(\tilde{\boldsymbol{\epsilon}}_i)) \tag{5}$$

where the Decode function extracts the message bit encoded in the sign pattern using the PRC key k (details of the PRC algorithm can be found in Appendix A.1), matching the encoding process where $\hat{\boldsymbol{\epsilon}}_i = \mathbf{c}_i \cdot |\boldsymbol{\epsilon}_i|$ and $\mathbf{c}_i \in \{-1, 1\}^{c \times h \times w}$. The complete decoded sequence is returned as $\hat{m} = [\hat{m}_1, \ldots, \hat{m}_f]$.

Detection Function $\{p, d\} = \mathcal{T}(m, \hat{m})$ determines whether the decoded message $\hat{m}$ contains the watermark message m. We compute the **edit distance** between these sequences, where the cost of insertion, deletion, and replacement operations is 1. To assess statistical significance, we generate N random sequences $\{r^1, r^2, ..., r^N\}$ and compute their edit distances with $\hat{m}$. The p-value is:

$$p = \text{rank}(d_{\text{edit}}(m, \hat{m}))/N \tag{6}$$

where rank is the position of $d_{\text{edit}}(m, \hat{m})$ among all distances. The detection result is $d = \mathbf{1}_{p<\tau}$ with threshold τ. If the p-value is less than τ, there is a watermark with m.

The edit distance calculation incorporates frame-wise Hamming distance, defined as:

$$d_H(m_i, \hat{m}_j) = \frac{1}{|m_i|} \sum_{k=1}^{|m_i|} \mathbf{1}_{m_i[k] \neq \hat{m}_j[k]} \tag{7}$$

This distance is normalized through a continuous mapping:

$$d_N(m_i, \hat{m}_j) = \begin{cases} 0 & \text{if } d_H(m_i, \hat{m}_j) < 0.5 \\ 2(d_H(m_i, \hat{m}_j) - 0.5) & \text{if } d_H(m_i, \hat{m}_j) \geq 0.5 \end{cases} \tag{8}$$

This transformation maps distances below 0.5 to 0, while linearly scaling distances from 0.5 and above to the range [0,1], enhancing the sensitivity of our detection mechanism.

Recovery Function $m' = \mathcal{R}(m, \hat{m})$ is applied when detection succeeds. During the edit distance calculation, we find both the starting index s and the optimal alignment path between m and $\hat{m}$:

$$s, \mathcal{P} = \text{argmin}_i\{d_{\text{edit}}(m[i:], \hat{m}), \text{Path}(m[i:], \hat{m})\} \tag{9}$$

where $\mathcal{P}$ represents the sequence of operations (match, insert, delete, substitute) that transforms $m[s:]$ into $\hat{m}$ with minimal cost. Using this path, we recover the original message by extracting the corresponding subsequence from m that aligns with $\hat{m}$:

$$m' = \{m[s+j] \mid \mathcal{P}_j \text{ is a match or substitute operation}\} \tag{10}$$

This extracts precisely the elements from the original message that correspond to the decoded sequence after accounting for any frame manipulations.

4 Experiments

4.1 Experimental Setting

Implementation details. In our primary experiments, we explore both text-to-video (T2V) and image-to-video (I2V) generation tasks, employing ModelScope (MS) [24] for T2V synthesis and I2VGen-XL [29] for I2V generation. The generated videos consist of 16 frames, each with a resolution of 512 $\times$ 512. The inference and inversion steps are set to their default values of 25 and 50, respectively. Watermarks of 512 bits are embedded into each generated frame of the two models. As described in the Section 3.2, we leverage DDIM inversion to obtain predicted initial noise. The threshold τ is set to 0.005. The number of random sequences N is set to 1000. All experiments are conducted using the NVIDIA Tesla A800 80G GPU.

Baseline. We selected four watermarking methods as baselines for comparison: RivaGAN[28], REVMark[30], VideoSeal[6], and VideoShield[10]. All selected methods are open-source and specifically designed to embed multi-bit strings within a video. Specifically, we set 32 bits for RivaGAN, 96 bits for REVMark, 96 bits for VideoSeal and 512 bits for VideoShield. Among these methods, VideoShield is the only in-generation approach, whereas the others are post-processing techniques that necessitate training new models for watermark embedding.

Datasets. We select 50 prompts from the test set of VBench[11], covering five categories: Animal, Human, Plant, Scenery, and Vehicles, with 10 prompts per category. For the T2V task, we generate four videos for each prompt for evaluation, ensuring diversity in outputs while maintaining consistency in prompt interpretation. For the T2V task, we first leverage a text-to-image model Stable Diffusion 2.1[18], to generate images corresponding to the selected prompts. These generated images are subsequently utilized to create videos. Overall, we generate a total of 200 videos for both tasks for the primary experiments. Additionally, for each prompt category in VBench, we generate 10 watermarked and 10 non-watermarked videos, resulting in a total of 8,000 watermarked and 8,000 non-watermarked videos for the watermark learnability comparison experiment.

Metric. We leverage Bit Accuracy to evaluate the ratio of correctly extracted watermark bits. To evaluate the quality of the generated videos, we conducted both objective and subjective assessments. For the objective evaluation, we leverage the metrics Subject Consistency, Background Consistency, Motion Smoothness, and Image Quality from VBench (see Appendix A.2 for details). For the subjective evaluation, we meticulously designed a pipeline that leverages GPT-4o to evaluate and score the generated videos (see Appendix A.3 for details).

4.2 Main Results

In Table 1, we present the main experimental results of VideoMark, including extraction accuracy, video quality, and both temporal and spatial robustness.

Extraction. The "Extraction" columns present the watermark bit length and bit accuracy of VideoMark in comparison with the baseline methods. For I2V, due to the accumulation of significant errors in the first frame during the inversion stage, we embed the watermark in all frames except the first frame. VideoMark achieves bit accuracies of 1.000 and 0.997 on the two models while

Table 1: The main experimental results of VideoMark. Except for the **Video Quality** columns, all other columns report bit accuracy metrics.

Model	Method	Extraction		Video Quality					Temporal Tampering(Acc.)				Spatial Tampering(Acc.)			
		Bit Len.	Acc.	SC	BC	MS	IQ	Avg.	Swap	Insert	Drop	Avg.	G. Blur	C. Jitter	R. Comp.	Avg.
MS	RivaGAN	32	0.994	0.922	0.951	0.960	0.648	0.870	0.930	0.919	0.930	0.926	0.919	0.939	0.783	0.880
	REVMark	96	0.996	0.943	0.960	0.972	0.450	0.831	0.992	-	-	-	0.987	0.765	0.508	0.753
	VideoSeal	96	0.964	0.950	0.959	**0.977**	0.679	0.891	0.960	0.960	0.961	0.961	0.964	0.964	0.565	0.831
	VideoShield	512	**1.000**	0.949	**0.962**	**0.977**	0.689	0.894	**1.000**	-	-	-	**1.000**	**1.000**	0.999	1.000
	VideoMark	512× 16	**1.000**	**0.951**	0.961	**0.977**	**0.692**	**0.895**	**1.000**	**1.000**	**1.000**	**1.000**	**1.000**	**1.000**	**1.000**	**1.000**
I2V	RivaGAN	32	0.942	0.858	0.912	0.927	0.561	0.815	0.919	0.909	0.919	0.916	0.886	0.893	0.781	0.853
	REVMark	96	0.975	0.853	0.900	0.918	0.500	0.793	0.967	-	-	-	0.928	0.713	0.518	0.720
	VideoSeal	96	0.982	0.859	0.915	0.928	0.573	0.819	0.980	0.980	0.981	0.981	0.980	**0.981**	0.633	0.865
	VideoShield	512	0.990	0.811	0.892	0.913	0.530	0.787	0.990	-	-	-	**0.990**	0.849	0.777	0.872
	VideoMark	512× 15	**0.997**	**0.864**	**0.917**	**0.930**	**0.581**	**0.823**	**0.997**	**0.991**	**0.997**	**0.995**	0.857	0.955	**0.921**	**0.911**

embedding 512×16 and 512×15 watermark bits, respectively, demonstrating superior extraction performance and confirming the effectiveness of our approach. This performance is comparable to the state-of-the-art watermarking algorithm VideoShield, prominently surpassing other watermarking algorithms, including VideoSeal and REVMark.

Quality. The "Video Quality" columns present the objective experimental results of various watermarking methods on the VBench benchmark. VideoMark consistently achieves state-of-the-art performance across all four metrics in both tasks. In the I2V task, it surpasses the best post-processing method, VideoSeal, by 0.004, and outperforms the leading in-processing method, VideoShield, by 0.036. Notably, in terms of Image Quality (IQ), our method achieves scores of 0.895 in the T2V task and 0.581 in the I2V task.

In addition to the objective evaluation metrics, Table 2 presents the voting results of subjective video quality assessments conducted using GPT-4o. The results show that VideoMark received the most votes for video quality in both tasks, with 260 and 245 votes, respectively, 29 and 22 more than the second-best method, RivaGAN. The visual results of different methods are provided in Appendix A.4 (Figure 9).

Table 2: Results of GPT-4o voting on watermarked video samples.

Model	RivaGAN	REVMark	VideoSeal	VideoShield	*VideoMark*
MS	231	47	194	240	**260**
I2V	232	111	205	227	**245**
Total Votes	463	158	399	467	**505**

Robustness. The "Temporal Tampering" and "Spatial Tampering" columns show robustness results under temporal and spatial attacks, respectively (detailed experimental settings are in A.5 and A.6).

Table 3: VideoMark robustness under temporal tampering attacks, reported as p-values.

ModelScope			**I2VGen-XL**		
Swap	Insert	Drop	Swap	Insert	Drop
0.001	0.001	0.001	0.001	0.001	0.001

Table 4: Comparison of temporal tampering robustness between VideoShield and VideoMark, using matching accuracy as the evaluation metric.

Method	**ModelScope**			**I2VGen-XL**		
	Swap	Insert	Drop	Swap	Insert	Drop
VideoShield	1.000	1.000	1.000	0.983	0.983	0.981
VideoMark	1.000	1.000	1.000	**0.996**	**0.989**	**0.996**

For temporal tampering, we show the bit accuracy between the decoded and embedded message. As REVMark does not release the necessary model files, and VideoShield cannot handle videos with variable frames during decoding, we omit their results from this evaluation. The results show that VideoMark maintains a perfect bit accuracy of 1.000 in the T2V task. In the I2V task, it achieves an average bit accuracy of 0.996 and retains a strong performance of 0.991 even under the most challenging frame insertion attack. These findings further suggest that Videomark can reliably decode the embedded message even under temporal tampering, thereby ensuring that the watermark is robustly distributed across frames.

In addition, in Table 3, we present the p-values of VideoMark's detection results under temporal tampering attacks. Both models exhibit a p-value of 0.001 in detecting temporal tampering, which indicates strong statistical significance. In Table 4, we compare the frame matching accuracy between VideoMark and VideoShield. The results indicate that VideoMark maintains a high frame matching accuracy of up to 0.996 in the I2V task, demonstrating the practical viability of the TMM module in reliably reconstructing the original temporal order.

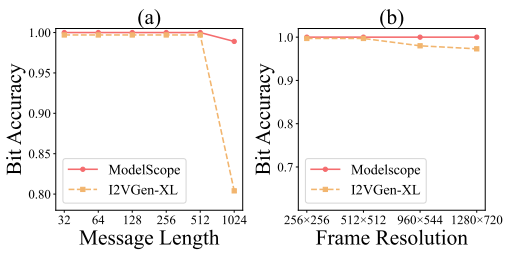

Figure 6: A Comparison of Bit Accuracy Performance Across Different Message Lengths and Resolutions

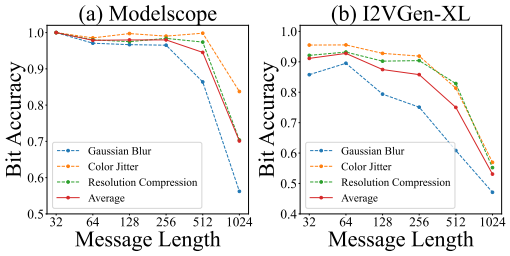

Figure 7: The robustness of VideoMark against spatial tampering under different message lengths.

For spatial tampering, we evaluate the performance of various methods under different types of spatial attacks (see Appendix A.1 for details). In this setting, VideoMark embeds 32 message bits per frame. The results show that VideoMark achieves perfect bit accuracy (1.000) on the T2V task. In the I2V task, while its bit accuracy under the Gaussian Blur attack is relatively lower (0.857), it still achieves the highest average bit accuracy of 0.911 across all three attack types.

Invisibility. To evaluate detectability, we leverage VideoMAE [23] as the backbone and train it with 100 epochs on a dataset consisting of 8,000 watermark-free videos and 8,000 watermarked videos to perform binary classification. The results in Figure 4, show that the network's classification accuracy is notably low for videos watermarked with VideoMark, achieving 54.07% on the training set and 48.02% on the validation set. In contrast, for other watermarking methods, the network performs similarly on the validation and training sets, suggesting that their watermark patterns are easier to learn and thus more easily detected.

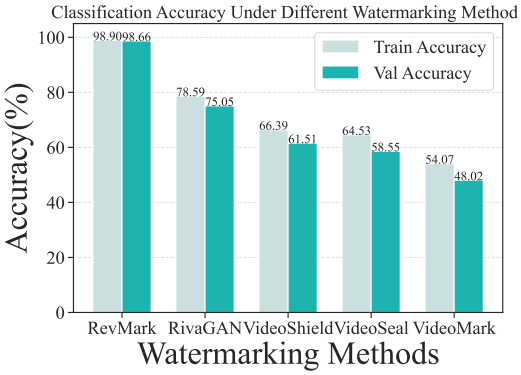

Figure 4: The binary classification results under different watermarking algorithms.

4.3 Analysis

Impact of message length. In the left part of Figure 6, we present the extraction accuracy for different watermark lengths: $[32, 64, 128, 256, 512, 1024]$ bits. The results show that VideoMark maintains high extraction accuracy when the number of embedded watermark bits is below 512. However, a noticeable drop in performance occurs at 1024 bits for both tasks, with the decline being especially significant in the I2V setting. This suggests that overly large watermarks may exceed the model's embedding capacity, resulting in reduced extraction accuracy. Based on these findings, we fix the message length at 512 bits for all subsequent extraction experiments.

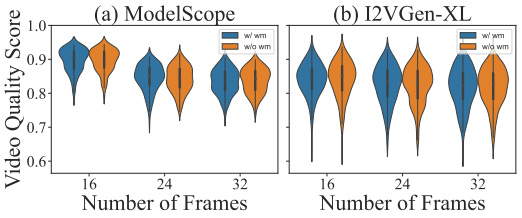

Figure 5: Video quality evaluation under varying numbers of frames, comparing watermarked (w/ wm) and non-watermarked (w/o wm) outputs.

As illustrated in Figure 7, in the T2V task, the highest robustness is observed when 32 watermark bits are embedded, with a clear degradation as the number of bits increases. In contrast, for the I2V task, robustness peaks at 64 bits, which is slightly higher than that achieved with 32 bits. We hypothesize that the observed differences are attributable to the generative complexity inherent in T2V versus I2V

models. The higher variability in T2V outputs may increase the likelihood of spatial perturbation during generation, thus making robustness more sensitive to the embedding payload.

Impact of frame resolution. In the right part of Figure 6, we present the bit accuracy of watermark extraction across different resolutions. We observe that in the T2V task, the bit accuracy remains at 1.0 across all resolutions. In contrast, for the I2V task, the bit accuracy degrades at higher resolutions. We attribute this to the accumulation of inversion errors in DDIM under high-resolution settings in the I2V task, which hinders effective watermark extraction. In the T2V task, the relatively low inversion error likely contributes to the successful extraction of watermarks, even under high-resolution conditions.

Impact of the number of generated frames. To further assess the influence of the number of generated frames on video quality under watermarking, Figure 5 compares watermarked and clean videos at different generation lengths. We observe that, across all three settings, the quality distribution of videos generated with VideoMark closely aligns with that of non-watermarked videos. This indicates that VideoMark introduces minimal perceptual distortion, regardless of video length. The consistency suggests that the watermarking process is well-aligned with the generative model's latent space and does not interfere with content realism.

5 Related Work

5.1 Video Diffusion Models

Diffusion models [20] map the data distribution to a Gaussian prior by progressively adding noise and recover the original data through an iterative denoising process. Video diffusion models [9] adopt a 3D U-Net architecture with interleaved spatial and temporal attention to generate high-quality frames while ensuring temporal consistency. Based on latent diffusion models [19], SVD [2] learns a multi-dimensional latent space for high-resolution frame synthesis. During the generation phase, DDIM sampling [22], a widely used efficient sampling algorithm, reduces the number of required sampling steps while maintaining high video quality, compared to DDPM sampling [8].

5.2 Video Watermark

Video watermarking technology embeds imperceptible patterns within visual content and employs specialized detection techniques to ascertain the presence of a watermark[15]. Watermarking methods are typically categorized into two paradigms: post-processing and in-processing schemes.

Post-processing schemes introduce perturbations to the video content with minimal visual distortion, typically at the pixel level. Recent works[6, 16, 28, 30] primarily focus on training watermark embedding networks by optimizing the discrepancy between watermarked and original videos, as well as the differences in encoding and decoding information. These methods may fail to address the trade-offs in optimization objectives, such as video quality and the robustness of watermark decoding.

In contrast to post-processing methods, in-processing schemes integrate the watermarking into the video generation process of current generative video models to better utilize their capabilities. For instance, Videoshield[10] extends the Gaussian Shading technique[26] from the image domain to the video domain, demonstrating enhanced robustness. However, its initialization strategy of repeating watermark bits introduces fixed patterns in the latent space, ultimately degrading the quality of the generated videos. Currently, there is only one existing approach for in-processing video watermarking. Therefore, previous methods struggle to achieve both watermark robustness and video quality. We propose an undetectable video watermarking scheme to address this trade-off.

6 Conclusion

In this work, we propose a training-free, undetectable watermarking framework for video diffusion models. Through extensive experiments, we demonstrate that the generated videos retain high visual quality and exhibit no perceptible artifacts attributable to the embedded watermark. However, the current framework relies on approximate inversion techniques, which may limit extraction accuracy in certain scenarios. For future improvements, we suggest exploring more advanced or robust inversion algorithms to enhance the reliability and effectiveness of the watermark retrieval process.

References

[1] Zaynab Almutairi and Hebah Elgibreen. A review of modern audio deepfake detection methods: challenges and future directions. *Algorithms*, 15(5):155, 2022.

[2] Andreas Blattmann, Tim Dockhorn, Sumith Kulal, Daniel Mendelevitch, Maciej Kilian, Dominik Lorenz, Yam Levi, Zion English, Vikram Voleti, Adam Letts, et al. Stable video diffusion: Scaling latent video diffusion models to large datasets. *arXiv preprint arXiv:2311.15127*, 2023.

[3] Mathilde Caron, Hugo Touvron, Ishan Misra, Hervé Jégou, Julien Mairal, Piotr Bojanowski, and Armand Joulin. Emerging properties in self-supervised vision transformers. In *Proceedings of the IEEE/CVF international conference on computer vision*, pages 9650–9660, 2021.

[4] Miranda Christ and Sam Gunn. Pseudorandom error-correcting codes. In *Annual International Cryptology Conference*, pages 325–347. Springer, 2024.

[5] Yuming Fang, Hanwei Zhu, Yan Zeng, Kede Ma, and Zhou Wang. Perceptual quality assessment of smartphone photography. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 3677–3686, 2020.

[6] Pierre Fernandez, Hady Elsahar, I Zeki Yalniz, and Alexandre Mourachko. Video seal: Open and efficient video watermarking. *arXiv preprint arXiv:2412.09492*, 2024.

[7] Sam Gunn, Xuandong Zhao, and Dawn Song. An undetectable watermark for generative image models. *arXiv preprint arXiv:2410.07369*, 2024.

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

[9] Jonathan Ho, Tim Salimans, Alexey Gritsenko, William Chan, Mohammad Norouzi, and David J Fleet. Video diffusion models. *Advances in Neural Information Processing Systems*, 35:8633–8646, 2022.

[10] Runyi Hu, Jie Zhang, Yiming Li, Jiwei Li, Qing Guo, Han Qiu, and Tianwei Zhang. Videoshield: Regulating diffusion-based video generation models via watermarking. In *The Thirteenth International Conference on Learning Representations*, 2025.

[11] Ziqi Huang, Yinan He, Jiashuo Yu, Fan Zhang, Chenyang Si, Yuming Jiang, Yuanhan Zhang, Tianxing Wu, Qingyang Jin, Nattapol Chanpaisit, et al. Vbench: Comprehensive benchmark suite for video generative models. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 21807–21818, 2024.

[12] Junjie Ke, Qifei Wang, Yilin Wang, Peyman Milanfar, and Feng Yang. Musiq: Multi-scale image quality transformer. In *Proceedings of the IEEE/CVF international conference on computer vision*, pages 5148–5157, 2021.

[13] Diederik P Kingma and Max Welling. Auto-encoding variational bayes. *arXiv preprint arXiv:1312.6114*, 2013.

[14] Zhen Li, Zuo-Liang Zhu, Ling-Hao Han, Qibin Hou, Chun-Le Guo, and Ming-Ming Cheng. Amt: All-pairs multi-field transforms for efficient frame interpolation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 9801–9810, 2023.

[15] Aiwei Liu, Leyi Pan, Yijian Lu, Jingjing Li, Xuming Hu, Xi Zhang, Lijie Wen, Irwin King, Hui Xiong, and Philip Yu. A survey of text watermarking in the era of large language models. *ACM Computing Surveys*, 57(2):1–36, 2024.

[16] Xiyang Luo, Yinxiao Li, Huiwen Chang, Ce Liu, Peyman Milanfar, and Feng Yang. Dvmark: a deep multiscale framework for video watermarking. *IEEE Transactions on Image Processing*, 2023.

[17] Alec Radford, Jong Wook Kim, Chris Hallacy, Aditya Ramesh, Gabriel Goh, Sandhini Agarwal, Girish Sastry, Amanda Askell, Pamela Mishkin, Jack Clark, et al. Learning transferable visual models from natural language supervision. In *International conference on machine learning*, pages 8748–8763. PmLR, 2021.

[18] Robin Rombach, Andreas Blattmann, Dominik Lorenz, Patrick Esser, and Björn Ommer. High-resolution image synthesis with latent diffusion models. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 10684–10695, June 2022.

[19] Robin Rombach, Andreas Blattmann, Dominik Lorenz, Patrick Esser, and Björn Ommer. High-resolution image synthesis with latent diffusion models. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 10684–10695, 2022.

[20] Jascha Sohl-Dickstein, Eric Weiss, Niru Maheswaranathan, and Surya Ganguli. Deep unsupervised learning using nonequilibrium thermodynamics. In *International conference on machine learning*, pages 2256–2265. pmlr, 2015.

[21] Jiaming Song, Chenlin Meng, and Stefano Ermon. Denoising diffusion implicit models. *arXiv preprint arXiv:2010.02502*, 2020.

[22] Jiaming Song, Chenlin Meng, and Stefano Ermon. Denoising diffusion implicit models. In *International Conference on Learning Representations*, 2021.

[23] Zhan Tong, Yibing Song, Jue Wang, and Limin Wang. Videomae: Masked autoencoders are data-efficient learners for self-supervised video pre-training. *Advances in neural information processing systems*, 35:10078–10093, 2022.

[24] Jiuniu Wang, Hangjie Yuan, Dayou Chen, Yingya Zhang, Xiang Wang, and Shiwei Zhang. Modelscope text-to-video technical report. *arXiv preprint arXiv:2308.06571*, 2023.

[25] Yuxin Wen, John Kirchenbauer, Jonas Geiping, and Tom Goldstein. Tree-ring watermarks: Fingerprints for diffusion images that are invisible and robust. *arXiv preprint arXiv:2305.20030*, 2023.

[26] Zijin Yang, Kai Zeng, Kejiang Chen, Han Fang, Weiming Zhang, and Nenghai Yu. Gaussian shading: Provable performance-lossless image watermarking for diffusion models. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 12162–12171, 2024.

[27] Junyan Zhang, Shuliang Liu, Aiwei Liu, Yubo Gao, Jungang Li, Xiaojie Gu, and Xuming Hu. Cohemark: A novel sentence-level watermark for enhanced text quality. In *The 1st Workshop on GenAI Watermarking*, 2025.

[28] Kevin Alex Zhang, Lei Xu, Alfredo Cuesta-Infante, and Kalyan Veeramachaneni. Robust invisible video watermarking with attention. *arXiv preprint arXiv:1909.01285*, 2019.

[29] Shiwei Zhang, Jiayu Wang, Yingya Zhang, Kang Zhao, Hangjie Yuan, Zhiwu Qing, Xiang Wang, Deli Zhao, and Jingren Zhou. I2vgen-xl: High-quality image-to-video synthesis via cascaded diffusion models. 2023.

[30] Yulin Zhang, Jiangqun Ni, Wenkang Su, and Xin Liao. A novel deep video watermarking framework with enhanced robustness to h. 264/avc compression. In *Proceedings of the 31st ACM International Conference on Multimedia*, pages 8095–8104, 2023.

A Appendix / supplemental material

A.1 PRC Watermark

Generation phase. We utilize **PRC.Encode** to transform the message intended for embedding into the binary codeword $c \in \{-1, 1\}^{c \times h \times w}$, while guaranteeing a uniform distribution, such that both -1 and 1 occur with equal probability. Let $\epsilon \sim \mathcal{N}(0, 1)$ and $c \in \{-1, 1\}$ be a Rademacher variable such that $\mathbb{P}(c = 1) = \mathbb{P}(c = -1) = 0.5$. Define the transformation:

$$\hat{\epsilon} = |\epsilon| \cdot c$$

We aim to compute the marginal distribution of $\hat{\epsilon}$ and show that $\hat{\epsilon} \sim \mathcal{N}(0, 1)$, thereby ensuring that the initialization noise remains unbiased and preserves the statistical properties of the standard Gaussian distribution.

The marginal probability density function of $\hat{\epsilon}$ can be obtained by marginalizing over the discrete variable c:

$$f_{\hat{\epsilon}}(x) = \sum_{c \in \{-1,1\}} f_{\hat{\epsilon}|C}(x \mid c) \cdot \mathbb{P}(C = c)$$

Since $|\epsilon|$ follows the half-normal distribution:

$$f_{|\epsilon|}(u) = \frac{2}{\sqrt{2\pi}} e^{-u^2/2}, \quad u \geq 0$$

We compute the conditional distributions:

$$f_{\hat{\epsilon}|C}(x \mid c) = \begin{cases} f_{|\epsilon|}(x), & \text{if } c = 1, x \geq 0 \\ f_{|\epsilon|}(-x), & \text{if } c = -1, x < 0 \\ 0, & \text{otherwise} \end{cases}$$

Therefore, the marginal becomes:

$$f_{\hat{\epsilon}}(x) = \begin{cases} \frac{1}{2} f_{|\epsilon|}(x) = \frac{1}{\sqrt{2\pi}} e^{-x^2/2}, & x \geq 0 \\ \frac{1}{2} f_{|\epsilon|}(-x) = \frac{1}{\sqrt{2\pi}} e^{-x^2/2}, & x < 0 \end{cases}$$

Thus,

$$f_{\hat{\epsilon}}(x) = \frac{1}{\sqrt{2\pi}} e^{-x^2/2}, \quad \forall x \in \mathbb{R}$$

which implies $\hat{\epsilon} \sim \mathcal{N}(0, 1)$. This indicates that the encrypted noise remains unbiased in its distribution, making it infeasible for users without the key to perceive the embedded watermark.

Detection phase. PRC.Detect accepts as input a vector $\mathbf{s}$ with entries in the interval $[-1, 1]$ rather than a bit-string. Given the PRC codeword $\mathbf{c}$, each component s_i is intended to approximate the conditional expectation of $(-1)^{c_i}$ based on the user's observation. The full procedure for **PRC.Detect** is presented in Algorithm 2.

Algorithm 2 PRC.Detect

Input: PRC-key k, s
Output: Detection result True or False
Parse

$$(n, \text{message\_length}, F, t, \lambda, \eta, \text{num\_test\_bits}, k, r, \text{max\_bp\_iter}, \text{otp}, \text{testbits}, G, P) \leftarrow k$$

For $i \in [n]$, $s_i \leftarrow (-1)^{\text{otp}_i} \cdot (1-2\eta) \cdot s_i$
For each parity check $w \in P$, let $\hat{s}_w \leftarrow \prod_{i \in w} s_i$
$C \leftarrow \frac{1}{2} \sum_{w \in P} \log^2 \left(\frac{1+\hat{s}_w}{1-\hat{s}_w} \right)$
if

$$\sum_{w \in P} \log \left(\frac{1+\hat{s}_w}{2} \right) \geq \sqrt{C \log(1/F)} + \frac{1}{2} \sum_{w \in P} \log \left(\frac{1-\hat{s}_w^2}{4} \right)$$

then
    return True
else
    return False
end if

Decode phase. PRC.Decode accepts as input a vector $\mathbf{s}$ with entries in the interval $[-1, 1]$, and returns a decoded binary message. The full procedure for **PRC.Decode** is presented in Algorithm 2.

Algorithm 3 PRC.Decode

Input: PRC-key k, s
Output: Decoded message $m \in \{0,1\}^k$ or None
Parse

$$(n, \text{message\_length}, F, t, \lambda, \eta, \text{num\_test\_bits}, k, r, \text{max\_bp\_iter}, \text{otp}, \text{testbits}, G, P) \leftarrow k$$

for $i \in [n]$ **do**
    $s_i \leftarrow (-1)^{\text{otp}_i} \cdot (1-2\eta) \cdot s_i$
end for
$y \leftarrow \text{BP-OSD}(G, P, s)$
Parse $(\text{testbits}', r, m) \leftarrow y$
return m

Algorithm 2 performs coarse-grained detection on each video frame to preliminarily identify potential targets or anomalies, serving as a foundation for the overall detection of the video.

In contrast, Algorithm 3 provides fine-grained analysis by attempting to decode a binary message from the input signal, enabling precise verification of target presence and integrity.

A.2 VBench for Video Quality

In this section, we provide a detailed overview of the video quality metrics in VBench, including Subject Consistency, Background Consistency, Motion Smoothness, and Imaging Quality. Additional details can be found in the original paper.

- **Subject Consistency** is measured by computing the similarity of DINO features[3] across frames:

$$S_{\text{subject}} = \frac{1}{T-1} \sum_{t=2}^{T} \frac{1}{2} \left(\langle d_1, d_t \rangle + \langle d_{t-1}, d_t \rangle \right), \tag{11}$$

  where d_i denotes the DINO image feature of the i^{th} frame, normalized to unit length, and $\langle \cdot, \cdot \rangle$ represents the dot product, corresponding to cosine similarity.

- **Background Consistency** assesses the temporal coherence of background scenes by computing CLIP feature similarity across frames [17]:

$$S_{\text{background}} = \frac{1}{T-1} \sum_{t=2}^{T} \frac{1}{2} \left(\langle c_1, c_t \rangle + \langle c_{t-1}, c_t \rangle \right), \tag{12}$$

  where c_i denotes the CLIP image feature of the i^{th} frame, normalized to unit length.

- **Motion Smoothness** is assessed using a frame-by-frame motion prior inspired by video frame interpolation models [14]. Given a generated video with frames $[f_0, f_1, f_2, f_3, f_4, \ldots, f_{2n-2}, f_{2n-1}, f_{2n}]$, the odd-indexed frames are manually removed to create a low-frame-rate sequence $[f_0, f_2, f_4, \ldots, f_{2n-2}, f_{2n}]$. A video frame interpolation model [14] is then applied to reconstruct the dropped frames $[\hat{f}_1, \hat{f}_3, \ldots, \hat{f}_{2n-1}]$. The Mean Absolute Error (MAE) between the reconstructed and original frames is calculated and normalized to the range $[0, 1]$, where a higher value indicates smoother motion.

- **Imaging Quality** is evaluated using the MUSIQ image quality predictor [12], which is trained on the SPAQ dataset [5] and is capable of handling images with varying aspect ratios and resolutions. Each frame is assigned a quality score by MUSIQ, which is linearly normalized to the range $[0, 1]$ by dividing by 100. The final imaging quality score is computed by averaging the normalized scores across all frames in the video.

In Table 5, we present a comparison between the video quality with and without watermarking under the VBench benchmark. It can be observed that VideoMark preserves video quality comparable to that of watermark-free videos.

Table 5: Video quality between different models.

Model	Method	Subject Consistency	Background Consistency	Motion Smoothness	Imaging Quality
MS	RivaGAN	0.922	0.951	0.960	0.648
	REVMark	0.943	0.960	0.972	0.450
	VideoSeal	0.950	0.959	**0.977**	0.679
	VideoShield	0.949	**0.962**	**0.977**	0.689
	VideoMark (ours)	**0.951**	0.961	**0.977**	**0.692**
	w/o watermark	**0.951**	**0.962**	**0.977**	0.691
I2V	RivaGAN	0.858	0.912	0.927	0.561
	REVMark	0.853	0.900	0.918	0.500
	VideoSeal	0.859	0.915	0.928	0.573
	VideoShield	0.811	0.892	0.913	0.530
	VideoMark (ours)	**0.864**	**0.917**	**0.930**	**0.581**
	w/o watermark	0.861	0.913	0.928	0.578

A.3 MLLM-as-a-Judge for Video Quality

Figure 8 shows the specific evaluation prompt used with GPT-4o for video quality assessment. The specific results are presented in Table 6, where each evaluation metric demonstrates that the proposed Videomark method effectively preserves the visual quality of the videos while embedding the watermark. Although there is a slight gap in subject consistency and motion smoothness compared to RivaGAN, the overall performance is comparable to that of the other methods.

Table 6: Detailed performance evaluation of videos assessed by GPT-4o.

Method	ModelScope				Total	I2VGen-XL				Total
	SC	BC	MS	IQ		SC	BC	MS	IQ	
RivaGAN	**7.88**	8.85	**4.75**	6.90	**28.38**	**7.69**	7.67	**6.32**	7.14	**28.82**
REVMark	6.52	7.95	4.41	3.95	22.83	7.62	**7.73**	5.94	5.57	26.86
VideoSeal	7.82	8.86	4.65	6.90	28.23	7.63	7.62	6.19	7.10	28.54
VideoShield	7.77	8.86	4.37	6.85	27.85	7.58	7.57	6.17	7.04	28.36
VideoMark (ours)	7.86	**8.93**	4.55	**6.98**	28.32	7.66	7.70	6.10	**7.19**	28.65
w/o watermark	7.84	8.84	4.67	6.95	28.31	7.67	7.61	6.28	7.17	28.73

SC: Subject Consistency BC: Background Consistency MS: Motion Smoothness
IQ: Imaging Quality

Prompts for GPT-4o Scoring Video Quality

Please perform a thorough and objective evaluation of the video result for sample `'sample_name'`. Using the provided 16 sampled frames as a reference, carefully assess the video on the following criteria:

1. Subject Consistency: Evaluate how consistently and accurately the subject is represented across all frames.
2. Background Consistency: Assess the uniformity and coherence of the background elements throughout the video.
3. Motion Smoothness: Judge the fluidity and natural progression of motion, ensuring transitions are smooth and realistic.
4. Imaging Quality: Evaluate the overall visual quality, including clarity, sharpness, color fidelity, and detail.

For each criterion, assign a numeric score between 0 (lowest) and 10 (highest) based solely on the visual data. Compute the total score as the sum of these individual scores. Your evaluation should be objective, data-driven, and free from any additional commentary or explanation. Return your response strictly as a JSON object formatted exactly as shown below without any extra text:

```
{
"subject_consistency": <number>,
"background_consistency": <number>,
"motion_smoothness": <number>,
"imaging_quality": <number>,
"total_score": <number>
}
```

Figure 8: Prompt used for the subjective evaluation of video quality.

A.4 Visualization of Generated Videos

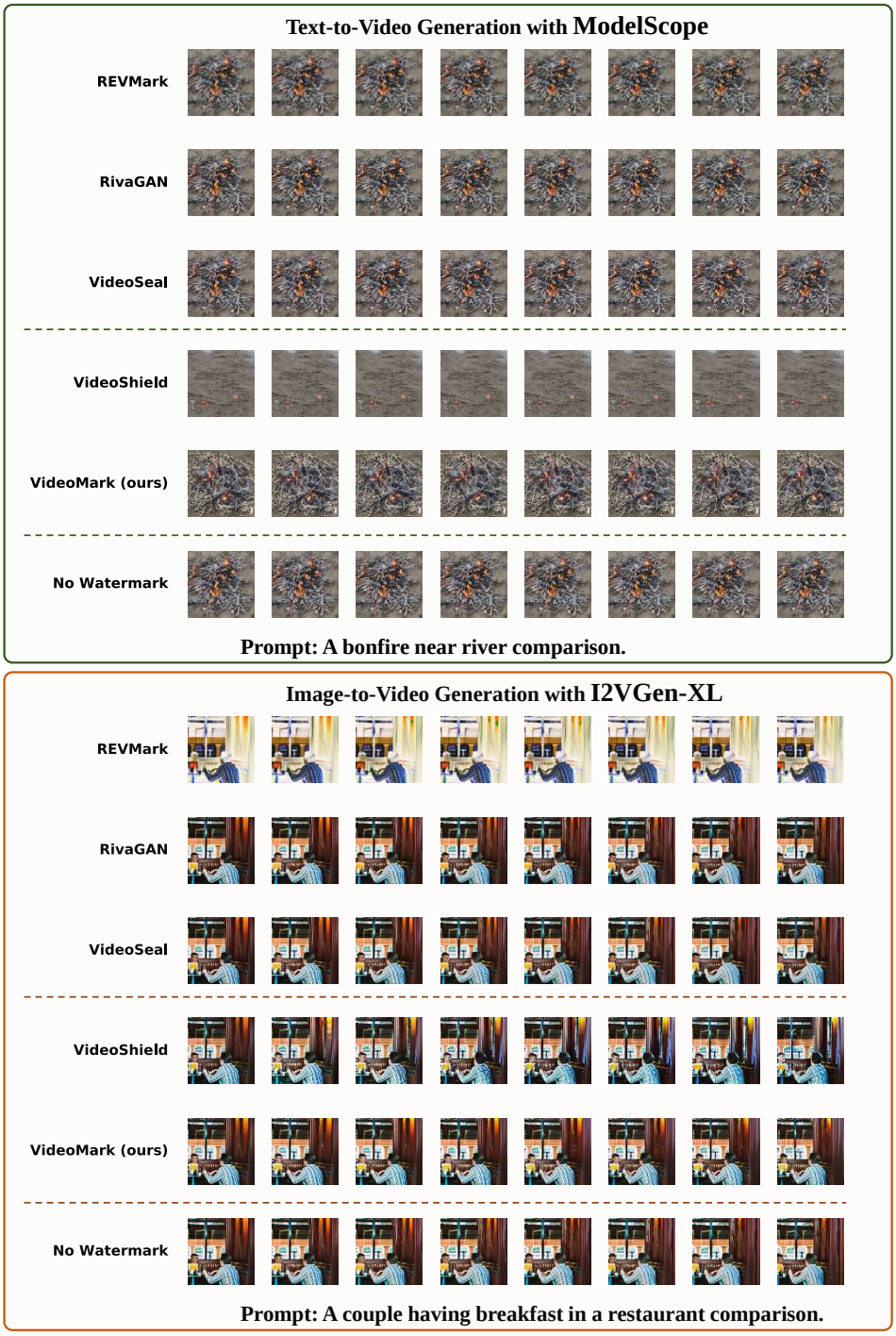

Figure 9: Visualization results comparing VideoMark with existing approaches on both T2V and I2V tasks. The visualizations for REVMarK, RivaGAN, and VideoSeal correspond to the content embedded into the watermark-free videos displayed in the bottom row (No Watermark).

A.5 Temporal Tampering Implement

We provide the details of the temporal tampering in this section. We consider three types of temporal tampering: **Frame Drop**, **Frame Insert**, and **Frame Swap**. The first two tampering methods alter the total number of frames in the video, whereas the last method only alters the frame order. The implementations and formulations are described as follows. Given a video sequence with f frames:

- **Frame Drop** removes a frame at a randomly selected temporal index p, where $p \in \{0, 1, \ldots, f-1\}$. The tampered sequence is defined as:

$$\text{Drop}(\text{frame}_p) \rightarrow \{\text{frame}_0, \ldots, \text{frame}_{p-1}, \text{frame}_{p+1}, \ldots, \text{frame}_{f-1}\}.$$

- **Frame Insert** duplicates a neighboring frame and inserts it at a randomly selected index $p \in \{1, 2, \ldots, f-1\}$. Assuming the duplicated frame is the preceding frame, the tampered sequence becomes:

$$\text{Insert}(\text{frame}_{p-1}) \rightarrow \{\text{frame}_0, \ldots, \text{frame}_{p-1}, \text{frame}_{p-1}, \text{frame}_p, \ldots, \text{frame}_{f-1}\}.$$

- **Frame Swap** exchanges the positions of temporally adjacent frames at regular intervals. Specifically, for

$$p = 2 + 4k, \quad k \sim \left\{0, 1, 2, \ldots, \left\lfloor \frac{f-3}{4} \right\rfloor\right\},$$

the swap operation is defined as:

$$\text{Swap}(\text{frame}_p, \text{frame}_{p+1}) \rightarrow \{\ldots, \text{frame}_{p+1}, \text{frame}_p, \ldots\}.$$

A.6 Spatial Tampering Implement

We provide the details of the spatial tampering in this section. We consider three types of temporal tampering: **Gaussian Blur**, **Colour Jitter**, **Resolution Compression**. These tampering operations are applied to every frame of the video. The implementations and formulations are described as follows. Given a video frame $\mathbf{F}$ with height H and width W:

- **Gaussian Blur** applies a convolution with a Gaussian kernel K of size $k \times k$, where k is a positive odd integer. The blurred frame $\mathbf{F}'$ is computed as:

$$\mathbf{F}' = \mathbf{F} * K,$$

where $*$ denotes the convolution operation. We use a kernel size of 11×11 in our experiments.

- **Colour Jitter** perturbs the brightness and contrast of the frame. The transformed frame $\mathbf{F}'$ is computed as:

$$\mathbf{F}' = \alpha\mathbf{F} + \beta,$$

where $\alpha \sim \mathcal{U}(1-s, 1+s)$ controls contrast, $\beta \sim \mathcal{U}(-s, s)$ controls brightness, and s is the jitter strength hyperparameter. We set $s = 0.1$ in all experiments.

- **Resolution Compression** simulates compression artifacts by downscaling and then upscaling the frame. Given a downscale factor $r \in (0, 1)$, the frame $\mathbf{F}'$ is obtained as:

$$\mathbf{F}' = \text{Upscale}(\text{Downscale}(\mathbf{F}, r), (H, W)),$$

where $\text{Downscale}(\cdot, r)$ resizes the frame to dimensions (rH, rW) and $\text{Upscale}(\cdot, (H, W))$ resizes it back to the original size using bilinear interpolation. We set a downscale factor of $r = 0.5$.