# FLOW-SYNDROME: KEYED LATENT-CARRIER WATERMARKING FOR VIDEO GENERATION

**Anonymous authors**
Paper under double-blind review

## ABSTRACT

The growing accessibility of realistic video generation creates an urgent need to preserve provenance through distribution and editing. Existing video watermarking methods encode information in output-space perturbations, generation noise, latent representations, or model parameters, each introducing different dependencies on generator access, inversion, or synchronization under temporal edits. We propose *Flow-Syndrome*, a generation-time watermarking framework that writes keyed code constraints into the latent representation of a frozen video generator and recovers them with a key-independent detector. A secret key determines a permutation and polarity map over detector carriers together with the keyed payload representation, while the detector itself is trained once per generator family and does not depend on the deployment key. During generation, bounded latent guidance drives detector scores toward the target carrier signs while fidelity and temporal-consistency terms limit perceptual distortion. At detection, the received video is re-encoded with the frozen VAE, carrier scores are aggregated over temporal groups, and the keyed mapping is inverted to recover the payload without access to the original video or inversion of the generative sampling process. Training through decode–transform–encode round trips, together with repeated encoding and temporal aggregation, improves robustness to common spatial, compression, and temporal edits. Experiments on both text-to-video and image-to-video models show strong payload recovery under diverse transformations while preserving video quality.

## 1 INTRODUCTION

Recent advances in text-to-video (T2V) and image-to-video (I2V) generation have made realistic synthetic videos increasingly accessible. Once released, a generated video may be compressed, resized, temporally edited, or incorporated into other content. These operations can separate a video from its original provenance record while leaving its content visually convincing. Video watermarking addresses this problem by embedding an identifier into the generated content for subsequent recovery. A practical system should preserve visual fidelity, retain sufficient evidence after editing, and support extraction from the distributed video without requiring its unmodified counterpart.

Existing methods realize these goals through several different watermark carriers. Output-space methods, such as VideoSeal Fernandez et al. (2024b), learn perturbations to generated frames and recover messages using a dedicated extractor. In-generation approaches can encode information in generation noise or other internal representations and may rely on generator inversion during extraction Hu et al. (2025a;b). Other methods modify model or decoder parameters to create persistent signatures in generated content Jang et al. (2024); Fares et al. (2026). These approaches have substantially improved watermark robustness and fidelity, including through spatiotemporal feature learning, temporal propagation, and explicit synchronization or matching mechanisms.

Their design choices, however, introduce different deployment dependencies. Output-space perturbations must remain small enough to preserve quality while surviving subsequent processing. Noise- or inversion-based methods inherit the approximation error and model dependence of the inversion procedure. Parameter-based methods require access to, and modification of, generator components. Temporal edits introduce an additional difficulty because they can change the correspondence between embedded evidence and the frames observed at detection time. Recent systems address this



Reviewers: please read the Reviewer Guidelines (iclr.cc/Conferences/2027/ReviewerGuidelines) and the AI Policy for Reviewers (iclr.cc/Conferences/2027/AIPolicyForReviewers). If you used AI to expand, edit, or polish your review, please provide the input text to the LLM. Better still, consider skipping the LLM and submitting your original text: we, and the authors, are much more interested in your unedited thoughts than in what an LLM has to say. AI-assisted or not, you are putting your name and reputation behind your review: LLM-generated falsehoods, hallucinations or misrepresentations are subject to disciplinary action, which may include desk-rejecting all papers you have authored.

challenge with explicit temporal matching, frame-group recovery, or transformation-aware training Hu et al. (2025b); Zhu et al. (2026); Jeong et al. (2025), but robustness can still depend strongly on the transformations and model families encountered during development.

We study a complementary design point: use the generator's latent representation as the watermark carrier, while separating the reusable detector from the secret key. Modern video generators expose a latent tensor throughout the sampling process and decode that tensor through a frozen video VAE. This makes the latent attractive for generation-time watermarking: a small guided change can be introduced without modifying the generator parameters, and the received video can later be mapped back to the same representation through the corresponding VAE encoder. The key technical question is how to turn this latent into a reusable collection of watermark carriers whose semantics remain hidden without the key and whose measurements remain sufficiently stable after common video transformations.

To this end, we introduce *Flow-Syndrome*. A compact 3D convolutional detector, trained once per generator family, maps a latent tensor to a set of carrier scores over several temporal groups. The detector is key-independent. A pseudorandom function of the secret key derives a permutation and polarity map over these carriers, together with a keyed payload representation. Thus the same detector can serve many independent keys: the key determines which carrier corresponds to which code bit and which sign represents each bit, while the detector weights themselves need not be secret.

Embedding is performed during generation by bounded guidance on the current latent. A watermark loss drives carrier scores toward the key-dependent target signs, while fidelity and temporal-consistency terms keep the guided latent close to the sampler's own trajectory and discourage visible flicker. At detection, the verifier receives only a possibly transformed video, re-encodes it with the frozen VAE encoder, evaluates the same detector, aggregates carrier scores over temporal groups, and applies the keyed inverse map. Repeated encoding and temporal aggregation provide tolerance to corrupted carrier measurements. The detector is trained through decode–transform–encode round trips so that the learned carrier measurements emphasize information that survives common deployment transformations.

This design has three useful consequences. First, watermark extraction is reference-free: it requires neither the unwatermarked video nor inversion of the generative sampling trajectory. Second, one key-independent detector supports many deployment keys, enabling wrong-key controls and empirical calibration without retraining the detector. Third, the watermark is embedded directly into the generator's latent during sampling, so no generator parameter modification is required.

Experiments on both T2V and I2V models evaluate payload recovery under spatial processing, recompression, screen recording, denoising, and multiple temporal manipulations such as frame dropping, swapping, insertion, and trimming, while also measuring video fidelity. Our main contributions are:

- We introduce a *keyed latent-carrier representation* for video generation watermarking in which a key-independent detector is combined with a secret-key-derived carrier permutation, polarity map, and payload representation.
- We develop a *generation-time latent-guidance* procedure that embeds target carrier signs into the frozen generator's latent while constraining perceptual and temporal distortion, together with blind extraction from the received video through VAE re-encoding.
- We develop *transformation-aware carrier recovery* through decode–transform–encode detector training, repeated payload encoding, and temporal-group aggregation, and evaluate the resulting robustness and fidelity across both T2V and I2V settings.

## 2 Related Work

**Output-space watermarking.** VideoSeal Fernandez et al. (2024b) embeds a learned signal in decoded video and uses a dedicated extractor. This design is generator-agnostic at deployment, but the watermark must survive all subsequent pixel-space processing while remaining visually unobtrusive. Flow-Syndrome instead embeds during generation and reads a learned latent carrier after VAE re-encoding.

**In-generation and latent/noise watermarking.** VideoShield Hu et al. (2025a) and VideoMark Hu et al. (2025b) place watermark evidence inside the generative process, while LVMark Jang et al. (2024) studies watermarking for latent video diffusion models. These works establish that internal generative representations can provide strong fidelity and robustness. Our intended distinction is narrower: Flow-Syndrome trains a reusable, key-independent latent detector and moves deployment-specific secrecy into a keyed carrier map and payload representation. Embedding then optimizes the current latent against that fixed detector rather than retraining a detector or modifying generator parameters for each key.

**Temporal robustness and synchronization.** Temporal editing is a distinct challenge for video watermarking because frame dropping, insertion, reordering, and trimming can invalidate a fixed correspondence between embedded evidence and received frames. VideoMark Hu et al. (2025b), SIGMark Zhu et al. (2026), and WaTeRFlow Jeong et al. (2025) explicitly address temporal robustness through matching, group recovery, or flow-consistency mechanisms. We therefore do not claim that temporal robustness itself is new. Our design instead asks whether temporal robustness can emerge from a key-independent latent detector trained through transformed round trips, combined with temporal-group replication and aggregation, without requiring optical-flow features or an explicit frame-to-symbol synchronization stage at detection.

The contribution of the proposed Flow-Syndrome is thus the combination of three design choices: (i) a learned latent carrier exposed by a frozen generator/VAE pipeline, (ii) a detector whose weights are independent of the secret deployment key, with keying implemented by carrier permutation and polarity, and (iii) detector-guided latent optimization at generation time. The syndrome-style representation is a keyed organization of payload and anchor coordinates; in the present construction, repetition and temporal aggregation—not syndrome decoding—provide the main error tolerance.

## 3 METHOD

We cast video generation watermarking as keyed generation-time embedding followed by blind detection under content-preserving transformations. A text-to-video generator $G_\theta$ produces a video $X = G_\theta(p, z) \in \mathbb{R}^{T\times C\times H\times W}$, where $p$ is the prompt, $z$ is the sampling noise or latent seed, $T$ is the number of frames, $C$ the number of channels, and $H \times W$ the spatial resolution. Flow-Syndrome is a generation-time scheme: the keyed embedder $E_K$ acts on the sampling process of the frozen generator, producing $\widetilde{X} = E_K(G_\theta; p, z, m)$, a video that carries the watermark when it is decoded. The unwatermarked sample $X$ is used only as a fidelity reference in our analysis and experiments. A reference-free detector receives a possibly transformed video $Y = \tau(\widetilde{X})$ and returns a payload estimate and a presence score, $D_K(Y) \to (\widehat{m}, \rho)$, without access to the unwatermarked reference or inversion of the generative sampling process. The payload $m \in \{0, 1\}^L$ can carry provenance information such as model identity, service identity, generation time, a session nonce, or a user fingerprint digest.

A practical scheme should satisfy four requirements. *Fidelity:* $d(X, \widetilde{X}) \le \epsilon$ for a perceptual distance $d$. *Robustness:* $\widehat{m} = m$, with $\rho$ above a calibrated acceptance threshold $\rho_0$, under the target family of transformations $\tau$. *Low false positives:* the presence check should rarely fire on unwatermarked video; in practice we calibrate this behavior empirically using clean-video and wrong-key controls. *Key privacy:* without $K$, the assignment of detector carriers to code bits and their polarities is hidden. Provenance authenticity can be provided separately by authenticating the recovered payload, for example with a digital signature or message authentication code.

Existing watermarks for generated video may reside in pixels, post-hoc perturbations, model parameters, decoder artifacts, generation noise, or other internal representations. We instead write a keyed code constraint into the generator's own latent and read it back with a compact detector trained once per generator family. The term *Flow* refers here to the flow-matching video-generation setting in which this latent is sampled. Roles of different parts in Flow-Syndrome are listed in Table 4 in supplement.

### 3.1 SCOPE AND THREAT MODEL

We separate *watermark recovery* from *provenance authentication*. The watermarking layer considered here aims to preserve a keyed payload through a specified family of content-preserving

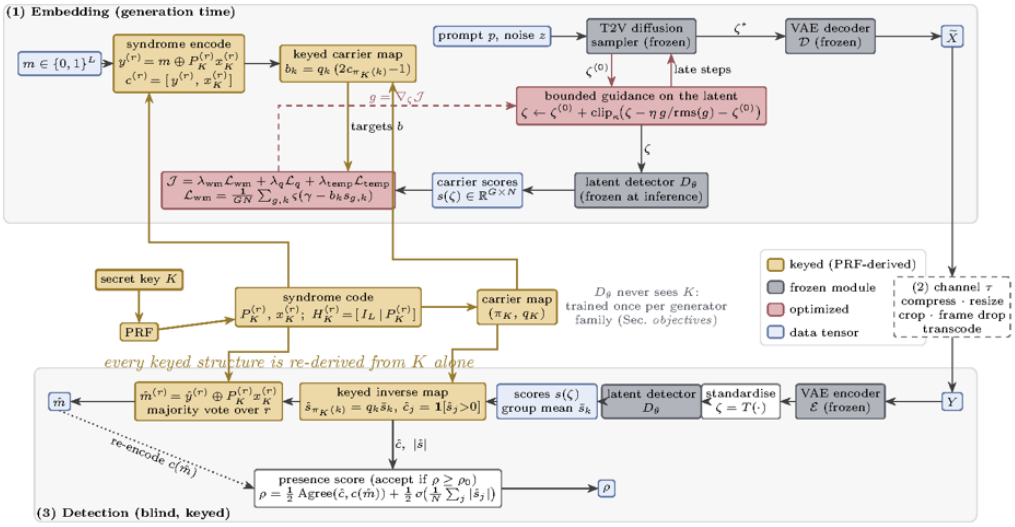



Figure 1: Flow-Syndrome Watermarking pipeline. A pseudorandom function (PRF) of the secret key $K$ derives the syndrome code $P_K^{(r)}, x_K^{(r)}$ (with $H_K^{(r)} = [I_L \mid P_K^{(r)}]$) and the carrier map $(\pi_K, q_K)$ that assigns code bits and polarities to the $N = RM$ carrier channels of the detector. The latent detector $D_\theta$ is trained once per generator family and never sees the key. During embedding, bounded guidance drives carrier scores toward target signs while fidelity terms keep the latent close to the sampler iterate. During detection, the received video is re-encoded with the same VAE, carrier scores are averaged over temporal groups, the keyed inverse map is applied, repeats are decoded, and the payload is recovered by majority vote.

transformations while maintaining video fidelity. The algorithms and detector architecture may be public and the deployment key $K$ is secret. A verifier possessing $K$ attempts to recover $m$ and decide whether the observed video contains a watermark consistent with that key.

We consider three qualitatively different failure modes. A *removal attack* transforms a marked video so that recovery fails while attempting to preserve useful visual content. A *false-positive attack* or accidental false positive causes an unwatermarked video to be accepted under a key. A *payload-substitution/forgery attack* attempts to make a video decode to an attacker-chosen provenance statement. Our experiments primarily measure robustness to non-adaptive removal transformations and false-positive behavior. They do not establish cryptographic unforgeability against an adaptive attacker with watermarking or detector queries. When cryptographic provenance authenticity is required, the payload should carry an independently authenticated statement, e.g., $m_0 \| \operatorname{Sig}_{sk}(m_0)$ or $m_0 \| \operatorname{MAC}_{K_a}(m_0)$, and the verifier should authenticate it after watermark recovery.

The key-derived permutation and polarity therefore provide *keyed carrier semantics*. In particular, we make no claim that the public detector is secure against arbitrary white-box, gradient-based, collusion, or chosen-watermark attacks. Evaluating such adaptive attacks is an important extension beyond the robustness channel studied here.

### 3.2 The Generator's Latent as the Watermark Carrier

Modern video generators sample in the latent space of a frozen video VAE $(E, D)$: a clip of $T$ frames at $H \times W$ corresponds to a latent of $C_\ell = 16$ channels, $T_\ell = \lfloor (T-1)/4 \rfloor + 1$ latent frames and $H/8 \times W/8$ spatial positions for the generators we use. This latent per channel is represented as $\zeta = \mathcal{T}(\xi) \in \mathbb{R}^{C_\ell \times T_\ell \times H_\ell \times W_\ell}$. The latent $\zeta$ is a good carrier because it is the quantity the sampler exposes at every step, so a watermark written into it needs no change to the generator and is present in the decoded video by construction. The same VAE encoder is available at detection time, so a detector can read the carrier through the encoder rather than from pixels, which lets it be trained end to end against the decode–encode round trip the deployed watermark must survive.

### 3.3 KEYED CARRIER MAP AND TRAINED DETECTOR

The carrier is read by a compact 3D convolutional detector $D_\theta$:

$$s(\zeta) = D_\theta(\zeta) \in \mathbb{R}^{G\times N}. \tag{1}$$

The key acts through the carrier map: a pseudorandom function of $K$ draws a permutation $\pi_K$ of $\{1,\dots,N\}$ and polarities $q_K \in \{-1,+1\}^N$, and carrier $k$ is assigned the target sign

$$b_k(m,K) = q_k\big(2c_{\pi_K(k)}(m,K)-1\big), \qquad k=1,\dots,N, \tag{2}$$

where $c(m,K) \in \{0,1\}^N$ is the keyed codeword of Sec. 3.3. A carrier score therefore carries a noisy code bit in its sign and an evidence weight in its magnitude, but which bit, and with which polarity, is hidden without $K$. Because the detector is key-independent, one detector evaluation serves every key: the wrong-key null distribution is available at no additional cost, and the detector itself need not be kept secret.

The detector is trained on real video latents watermarked by a parameter-free carrier embedder that plants sparse sign spikes: for each group it selects a fixed, publicly seeded set of $\nu$ latent elements per carrier ($\nu = 16$) and overwrites each with $b_k \max(|\zeta|, m_0)$, $m_0 = 0.5$. These spikes teach the detector where and how to read.

**Detector training.** For a clean latent $\zeta_{\mathrm{cl}}$ and payload $m$, let $\zeta_{\mathrm{mk}}$ be the marked latent, and let $\zeta_{\mathrm{rt}} = E(A(D(\zeta_{\mathrm{mk}})))$ be its round-trip counterpart, where $D$ and $E$ are the generator's frozen VAE decoder and encoder and $A$ is one transformation drawn uniformly from a five-way augmentation suite (identity, resize round trip, centre crop and resize, periodic frame drop-and-duplicate, social-media-style recompression). A cheap latent-domain corruption is applied on top,

$$\kappa(\zeta) = (\zeta + \sigma\varepsilon) \odot M_{\mathrm{ch}} \odot M_{\mathrm{tm}}, \qquad \varepsilon \sim \mathcal{N}(0, I), \tag{3}$$

with $\sigma = 0.01$ and $M_{\mathrm{ch}}, M_{\mathrm{tm}}$ independent Bernoulli keep-masks that zero an entire channel, respectively an entire latent frame, with probability $0.05$. The training objective is

$$\mathcal{L}_{\mathrm{det}}(\theta) = w_{\mathrm{raw}}\mathcal{L}_{\mathrm{wm}}(\zeta_{\mathrm{mk}}) + w_{\mathrm{rt}}\mathcal{L}_{\mathrm{wm}}(\kappa(\zeta_{\mathrm{rt}})) + w_{\mathrm{cn}}\mathcal{L}_{\mathrm{cn}} + w_{\mathrm{ng}}\mathcal{L}_{\mathrm{ng}}, \tag{4}$$

with

$$\mathcal{L}_{\mathrm{cn}} = \frac{1}{N}\|\bar{s}(\zeta_{\mathrm{mk}}) - \bar{s}(\kappa(\zeta_{\mathrm{rt}}))\|_2^2, \quad \bar{s}(\cdot) = \frac{1}{G}\sum_g s_{g,\cdot}(\cdot), \quad \mathcal{L}_{\mathrm{ng}} = \frac{1}{GN}\sum_{g,i} s_{g,i}(\zeta_{\mathrm{cl}})^2 \tag{5}$$

The consistency term ties the group-averaged score vectors of the raw and round-trip views together; the negative term supplies the clean-video class and makes the presence score meaningful. We use $w_{\mathrm{raw}} = 0.5$, $w_{\mathrm{rt}} = 1.0$, and $w_{\mathrm{cn}} = w_{\mathrm{ng}} = 0.1$.

During training we derive fresh sample-specific carrier assignments from a root key and a public sample nonce, e.g., $K_s = \mathrm{SHA256}(K\|\mathrm{nonce})$. The neural detector $D_\theta$ itself is key-independent; full watermark detection remains keyed because the verifier uses $K$ (and the public nonce, when sample-specific derivation is enabled) to reconstruct the carrier map and payload representation.

**Mark generation during training.** The detector trained purely on carrier marks is mismatched with deployment: carriers are sparse $\pm 0.5$ spikes, whereas guidance produces dense, temporally smooth perturbations bounded by $10^{-2}$. We therefore embed a fraction of the training positives with the guidance operator itself, run against a frozen snapshot $\bar{\theta}$ of the current detector that is refreshed every optimiser step. Fidelity enters where it acts, namely on generation of the mark.

### 3.4 SYNDROME PAYLOAD ENCODING

Because any single carrier sign is unreliable, we spread the payload over a keyed linear representation; arithmetic is over $\mathrm{GF}(2)$, with $\oplus$ denoting addition modulo two. Fix a codeword length $M > L$ and an odd repeat count $R$. For each repeat $r \in \{1,\dots,R\}$, the key generates a sparse binary matrix $P_K^{(r)} \in \{0,1\}^{L\times(M-L)}$ with a fixed small column weight (three in our implementation), the systematic parity-check matrix $H_K^{(r)} = [I_L \mid P_K^{(r)}]$, and keyed anchor bits $x_K^{(r)} \in \{0,1\}^{M-L}$.

The codeword is $c^{(r)} = [y^{(r)}, x_K^{(r)}] \in \{0,1\}^M$ with $y^{(r)} = m \oplus P_K^{(r)} x_K^{(r)}$, so that $H_K^{(r)} c^{(r)} = m$. Concatenating the repeats gives $c = [c^{(1)}, \ldots, c^{(R)}] \in \{0,1\}^N$ with $N = RM$, matched one-to-one with the detector's carrier channels through the carrier map. In the current decoder, the keyed anchor bits are regenerated from $K$ when recovering the payload, while their noisy observations are used only by the presence test. Consequently, payload robustness is provided primarily by repeated encoding and aggregation across independently measured carriers; the sparse keyed matrix defines the payload representation and couples the payload-bearing and anchor coordinates used for consistency testing. Our default configuration uses $L = 32$, $M = 96$, and $R = 3$, hence $N = 288$.

**What the syndrome representation does and does not do.** Because the verifier regenerates $x_K^{(r)}$, payload recovery from a hard observation $\widehat{y}^{(r)}$ is

$$\widehat{m}^{(r)} = \widehat{y}^{(r)} \oplus P_K^{(r)} x_K^{(r)}. \tag{6}$$

Hence errors in the observed anchor coordinates $\widehat{x}^{(r)}$ do *not* propagate into $\widehat{m}^{(r)}$ in the present decoder. The matrix $P_K^{(r)}$ should therefore not be interpreted as an error-correcting parity-check matrix whose sparse columns correct noisy anchors. Rather, $H_K^{(r)} c^{(r)} = m$ gives a convenient keyed systematic syndrome representation: the payload-bearing coordinates are masked by a key-derived linear function of the anchors, while the observed anchors provide held-out key-consistency evidence for presence testing. Error tolerance comes from repeating the representation across $R$ blocks and $G$ temporal groups and aggregating their measurements.

This distinction matters for both interpretation and evaluation. A stronger future variant could use genuine coset/soft decoding in which the received anchor coordinates participate in decoding; that would be a different construction and is not assumed by the results reported here.

### 3.5 Watermark Embedding Objective

Embedding is performed at generation time. The watermark embedding optimization objective is the softplus margin objective

$$\mathcal{J}_{\mathrm{wm}}(\zeta) = \frac{1}{GN} \sum_{g=1}^{G} \sum_{k=1}^{N} \varsigma\big(\gamma - b_k s_{g,k}(\zeta)\big), \qquad \varsigma(x) = \log(1 + e^x), \tag{7}$$

where $\gamma > 0$ is the desired detection margin. At late denoising steps, $\mathcal{J}_{\mathrm{wm}}$ is combined with two fidelity terms anchored at the sampler's own pre-guidance latent $a$,

$$J(\zeta; a) = \lambda_{\mathrm{wm}} \mathcal{J}_{\mathrm{wm}}(\zeta) + \lambda_q \mathcal{J}_q(\zeta, a) + \lambda_{\mathrm{temp}} \mathcal{J}_{\mathrm{temp}}(\zeta, a), \tag{8}$$

where $\mathcal{J}_q(\zeta, a) = n^{-1} \|\zeta - a\|_2^2$ and $\mathcal{J}_{\mathrm{temp}}(\zeta, a) = n_\Delta^{-1} \|\Delta\zeta - \Delta a\|_1$. Each guided step applies a few iterations of RMS-normalised gradient descent, with cumulative displacement clipped element-wise to $[-\kappa, \kappa]$: $\zeta \leftarrow \zeta^{(0)} + \mathrm{clip}_\kappa\left(\zeta - \eta \frac{g}{\mathrm{rms}(g)} - \zeta^{(0)}\right), \qquad g = \nabla_\zeta J(\zeta; a)$. Restricting guidance to late denoising steps steers fine detail while leaving content and layout intact. The complete schedule, normalisation regimes, final refinement, and hyperparameters are given in the supplementary materials.

### 3.6 Reference-Free Detection

Detection is blind and keyed: every keyed structure used at embedding time is re-derived from $K$, and the detector weights are the same fixed $D_\theta$. A suspect video $Y$ is encoded by the frozen VAE encoder and standardised, $\zeta_Y = \mathcal{T}(E(Y))$. The detector emits $s(\zeta_Y)$, which is averaged over temporal groups and mapped back to code order,

$$\bar{s}_k = \frac{1}{G} \sum_{g=1}^{G} s_{g,k}(\zeta_Y), \qquad \widehat{s}_{\pi_K(k)} = q_k \bar{s}_k, \qquad \widehat{c}_j = \mathbf{1}[\widehat{s}_j > 0]. \tag{9}$$

The hard bits are partitioned into $R$ blocks $\widehat{c}^{(r)} = [\widehat{y}^{(r)}, \widehat{x}^{(r)}]$. Each repeat is decoded with anchors regenerated from the key,

$$\widehat{m}^{(r)} = \widehat{y}^{(r)} \oplus P_K^{(r)} x_K^{(r)}, \qquad r = 1, \ldots, R, \tag{10}$$

and the final payload is the per-bit majority vote. The observed $\widehat{x}^{(r)}$ contributes to the presence check.

The presence score combines codeword-level agreement with score magnitudes. Let $c(\widehat{m})$ be the re-encoding of the decoded payload under $K$ and $\mathrm{Agree}(\widehat{c}, c(\widehat{m})) = \frac{1}{N}\sum_{j=1}^{N} \mathbf{1}[\widehat{c}_j = c_j(\widehat{m})]$. Then $\rho = \frac{1}{2}\,\mathrm{Agree}(\widehat{c}, c(\widehat{m})) + \frac{1}{2}\sigma\left(\frac{1}{N}\sum_{j=1}^{N} |\widehat{s}_j|\right)$, where $\sigma$ is the logistic function. Since $\widehat{m}$ is fit to the observed bits, the payload part of the agreement is optimistically biased. The anchor part, which matches only at chance on clean video, helps make $\rho$ discriminative together with the magnitude term. In deployment, the recovered payload can additionally be authenticated, e.g., by verifying an embedded digital signature or message authentication code.

### 3.7 Robustness to Video Transformations

Robustness is built into the detector rather than into the embedding. The deployed watermark must survive the VAE decode, ordinary video processing, container storage, and VAE re-encoding before the detector sees it, so the detector is trained on exactly that round trip: every training positive is decoded, passed through one transformation drawn from an augmentation suite (resize round trips, centre cropping with rescaling, periodic frame dropping and duplication, and a composed social-media-style transform), re-encoded, and further corrupted in the latent domain before its loss is computed. The detector thus learns to read the component of the mark that survives the channel, and the embedding objective, which differentiates through that detector, is steered toward that component automatically. Temporal redundancy comes from the $G$ groups: each carries a full copy of the code, and group averaging lets intact groups outvote damaged ones. Transformations that change the frame count are passed to the encoder exactly as they leave the video, without resampling back to the original length.

## 4 Experiments

### 4.1 Experimental Questions and Datasets

Our evaluation is organized around four questions. **Q1 (recovery):** how accurately can the payload be recovered before and after common video transformations? **Q2 (temporal edits):** how does recovery behave under frame dropping, swapping, insertion, and trimming? **Q3 (fidelity):** how much does latent guidance change perceptual video quality? **Q4 (capacity):** how does increasing payload length affect recovery and fidelity? We evaluate both T2V and I2V generation. The supplied results use Wan2.2-T2V for the attack benchmark and Wan2.2-I2V for the I2V fidelity/capacity study. Datasets for detection training, evaluation prompts are separate. Details on those datasets are presented in the supplement.

### 4.2 Evaluation Metrics

For evaluating robustness and video quality, different metrics are employed. Details can be found in the supplement.

### 4.3 Experimental Results

**Robustness on Wan2.2-T2V.** Tab. 1 reports bit accuracy under all 16 attacks. Our 16-bit model recovers the payload perfectly on unattacked clips and averages 0.918 over the attack suite, with 1.000 on blur, rescaling, colour jitter, adjacent-frame swap, frame insertion and subtitle overlay. Its profile is strongest on temporal edits (0.965 average over category II, including 0.959 under frame drops and 0.978 under trimming), which the temporal grouping of the detector is designed for: each group carries a full copy of the code and the intact groups outvote the damaged ones. It remains high under real-world post-processing (0.936 over category III; 0.950 under screen recording, 0.912 under denoising) and is weakest under geometric transformations (0.738 cropping, 0.644 rotation) and Gaussian noise (0.800).

**Effect of payload size.** Robustness falls as capacity rises. From 16 to 128 bits the attack-suite average moves from 0.918 to 0.816 while unattacked accuracy stays at or above 0.993; the 32-bit

| attack | cat | ours-16 | ours-32 | ours-48 | ours-64 | ours-128 | ours-256 | ours-512 | VideoSeal | VidSig | VideoShield |
|---|---|---|---|---|---|---|---|---|---|---|---|
| clean | – | **1.000** | 1.000 | 1.000 | 0.997 | 0.993 | 0.892 | 0.524 | 0.999 | 0.841 | 0.999 |
| gauss | I | 0.800 | 0.680 | 0.744 | 0.787 | 0.803 | 0.686 | 0.515 | 0.982 | 0.797 | **0.992** |
| blur | I | **1.000** | 1.000 | 1.000 | 0.991 | 0.979 | 0.868 | 0.523 | 0.998 | 0.505 | 0.997 |
| crop | I | 0.738 | 0.473 | 0.493 | 0.552 | 0.580 | 0.536 | 0.509 | **0.987** | 0.835 | 0.793 |
| rotate | I | 0.644 | 0.438 | 0.466 | 0.532 | 0.566 | 0.537 | 0.513 | 0.732 | **0.746** | 0.569 |
| rescale | I | **1.000** | 1.000 | 1.000 | 0.995 | 0.989 | 0.889 | 0.526 | 0.999 | 0.585 | 0.997 |
| colorjit | I | **1.000** | 0.986 | 0.995 | 0.974 | 0.950 | 0.851 | 0.517 | 0.999 | 0.841 | 0.998 |
| drop | II | 0.959 | 0.872 | 0.910 | 0.866 | 0.846 | 0.741 | 0.512 | **0.999** | 0.839 | 0.885 |
| swap | II | 0.887 | 0.673 | 0.813 | 0.750 | 0.728 | 0.645 | 0.512 | **0.999** | 0.841 | 0.933 |
| swap_adjacent | II | **1.000** | 0.948 | 0.998 | 0.967 | 0.950 | 0.846 | 0.527 | 0.999 | 0.841 | 0.992 |
| insert | II | **1.000** | 0.975 | 0.981 | 0.952 | 0.935 | 0.822 | 0.525 | 0.999 | 0.843 | 0.993 |
| trim | II | 0.978 | 0.697 | 0.923 | 0.834 | 0.856 | 0.649 | 0.511 | **0.998** | 0.844 | 0.851 |
| recomp | III | 0.828 | 0.581 | 0.600 | 0.671 | 0.671 | 0.574 | 0.513 | 0.924 | 0.765 | **0.963** |
| screenrec | III | 0.950 | 0.645 | 0.681 | 0.648 | 0.685 | 0.566 | 0.506 | 0.907 | 0.495 | **0.976** |
| denoise | III | 0.912 | 0.744 | 0.839 | 0.762 | 0.791 | 0.625 | 0.518 | **0.998** | 0.654 | 0.971 |
| subtitle | III | **1.000** | 1.000 | 0.999 | 0.994 | 0.989 | 0.885 | 0.526 | 0.997 | 0.836 | 0.994 |
| crop_drop | III | 0.988 | 0.725 | 0.795 | 0.727 | 0.736 | 0.611 | 0.496 | **0.989** | 0.816 | 0.985 |
| **all 16 attacks** | | 0.918 | 0.777 | 0.827 | 0.813 | 0.816 | 0.708 | 0.516 | **0.969** | 0.755 | 0.931 |

Table 1: Robustness under different video attacks with Wan2.2-T2V model. Best results are highlighted in bold and second best results are underlined.

model, at 0.777, is an outlier below the 48–128-bit models rather than part of the trend. Beyond 128 bits the carrier saturates under the fixed perturbation budget: the 256-bit model reads 0.892 unattacked and 0.708 under attack, and at 512 bits the detector cannot recover the payload even from clean video (0.524, chance level). The same latent budget must carry every additional bit, so each bit is defended by proportionally less signal; in this configuration the usable capacity lies between 128 and 256 bits, and 16 bits is the operating point that maximises robustness.

**Comparison with baselines.** VideoSeal is the most robust method overall (0.969) and VideoShield follows (0.931), with our 16-bit model at 0.918 and VidSig at 0.755. Two caveats frame this ranking. VideoSeal carries 256 bits and is post-hoc: its CNN extractor resizes every frame to a fixed $256 \times 256$ grid and was trained with geometric augmentation, which makes it nearly invariant to cropping (0.987) where every in-generation method, ours (0.738), VideoShield (0.793) and VidSig (0.835), degrades, and on rotation all methods drop to 0.57–0.75. Against VideoSeal our 16-bit model matches or exceeds it on eight of the seventeen conditions (clean, blur, rescaling, colour jitter, adjacent swap, insertion, subtitle and screen recording, 0.950 vs. 0.907), and the deficit is concentrated in cropping, Gaussian noise, frame swapping and recompression. VideoShield and our method have complementary profiles: VideoShield, whose mark occupies the entire initial noise tensor, is the strongest method under real-world post-processing (0.963 recompression, 0.976 screen recording, 0.971 denoising) but the weakest under temporal edits among the strong methods (0.931 over category II vs. our 0.965; 0.851 under trimming, 0.885 under frame drops) and under rotation (0.569); it also generates at $256 \times 256$ with 16 frames, where an attack removes a larger fraction of the content than at our resolution. VidSig is capped by its own clean accuracy (0.841 after MP4 export) and collapses under blur (0.505), rescaling (0.585) and screen recording (0.495). Among in-generation methods on a modern high-resolution backbone, ours is therefore the most robust to temporal manipulation and the most capacity-flexible, at the cost of geometric attacks, which we discuss as future work.

**Video quality across generators.** Tab. 2 reports VBench scores of watermarked videos on all four generators (the HunyuanVideo rows use the 32-bit model). Read across rows, the scores track the backbone: HunyuanVideo-T2V attains the highest subject and background consistency and motion smoothness (0.9776/0.9789/0.9927), Wan2.2-I2V the highest imaging quality (0.7219), and VideoShield's $256 \times 256$ ModelScope clips the lowest values on every dimension. Read against the corresponding unwatermarked generations, which is the comparison that isolates the watermark, every configuration of our method is within 0.008 of its clean counterpart on all four dimensions, and

| set | SC | BC | MS | IQ |
|---|---|---|---|---|
| VideoSeal | 0.9681 | 0.9680 | 0.9869 | 0.6884 |
| VidSig | <u>0.9749</u> | 0.9740 | 0.9832 | <u>0.6903</u> |
| VideoShield | 0.9344 | 0.9608 | 0.9667 | 0.6387 |
| Wan2.2 T2V-16 | 0.9673 | <u>0.9756</u> | 0.9867 | 0.6873 |
| Wan2.2 T2V-128 | 0.9642 | 0.9700 | 0.9867 | 0.6872 |
| Wan2.2 I2V-16 | 0.9713 | 0.9659 | 0.9815 | **0.7219** |
| Hunyuan-T2V | **0.9776** | **0.9789** | **0.9927** | 0.6291 |
| Hunyuan-I2V | 0.9668 | 0.9625 | <u>0.9870</u> | 0.6813 |

Table 2: VBench scores with Wan2.2-T2V, Wan2.2-I2V Hunyuan-T2V and Hunyuan-I2V models. Best results are highlighted in bold and second best results are underlined.

| Set | Generator | Res. | Frames | Bits | SC | BC | MS | IQ |
|---|---|---|---|---|---|---|---|---|
| Ours-16 | Wan2.2-I2V | 832×464 | 81 | 16 | <u>0.9713</u> | **0.9659** | 0.9815 | 0.7219 |
| Ours-32 | Wan2.2-I2V | 832×464 | 81 | 32 | 0.9673 | 0.9631 | 0.9816 | <u>0.7221</u> |
| Ours-48 | Wan2.2-I2V | 832×464 | 81 | 48 | 0.9688 | 0.9620 | <u>0.9817</u> | 0.7219 |
| Ours-64 | Wan2.2-I2V | 832×464 | 81 | 64 | 0.9673 | 0.9623 | 0.9815 | 0.7218 |
| Ours-128 | Wan2.2-I2V | 832×464 | 81 | 128 | 0.9661 | 0.9610 | 0.9816 | 0.7220 |
| VideoSeal | Wan2.2-I2V (our clips) | 832×464 | 81 | 256 | **0.9738** | <u>0.9653</u> | **0.9825** | **0.7224** |
| VidSig | SVD-XT | 512×512 | 16 | 48 | 0.9410 | 0.9486 | 0.9455 | 0.6850 |
| VideoShield | SVD-XT | 512×512 | 16 | 512 | 0.9576 | 0.9451 | 0.9506 | 0.6899 |

Table 3: VBench scores on watermarked sets (20 clips per set) with the Wan2.2-I2V model. Best results are highlighted in bold and second best results are underlined.

within 0.001 on motion smoothness and imaging quality; on HunyuanVideo the largest deviation is 0.002 (background consistency, I2V). The watermark is thus invisible to VBench on both backbones and both tasks, and the ordering of Tab. 2 should be attributed to the generators .

**Video quality on Wan2.2-I2V.** Tab. 3 isolates the effect of payload size and of the baselines on one backbone and one clip set. Across 16 to 128 bits our scores are flat: subject consistency spans 0.9661–0.9713, background consistency 0.9610–0.9659, and motion smoothness and imaging quality vary by at most 0.0003, so the perceptual cost of the watermark does not grow with the payload even though robustness does fall. VideoSeal, which marks the identical clean clips, and our 16-bit model are within 0.003 of each other on every dimension (0.9738 vs. 0.9713 SC, 0.9653 vs. 0.9659 BC, 0.9825 vs. 0.9815 MS, 0.7224 vs. 0.7219 IQ), and both sit within 0.007 of the unwatermarked clips (0.9741/0.9714/0.9813/0.7219): the two methods are equally transparent to VBench, and neither is distinguishable from the clean generation by these metrics. VidSig and VideoShield score lower on every dimension, but their own unwatermarked SVD-XT generations score the same, so the gap measures the $512 \times 512$, 16-frame backbone rather than the watermarks. Taken with Tab. 3, the results show that our method reaches the robustness regime of the strongest dedicated baselines at a perceptual cost that VBench cannot detect, while requiring no change to the generator and no per-generator embedder beyond a 1.9M-parameter detector.

## 5 Conclusion

We presented Flow-Syndrome, a generation-time video watermarking framework built around a key-independent latent detector, a secret-key-derived carrier map, and bounded detector-guided optimization of the generator latent. The design supports blind recovery from the received video without the unwatermarked reference or inversion of the sampling trajectory, while allowing one detector to serve many deployment keys. Experiment results demonstrate that short payloads can survive a range of spatial, compression, and especially temporal transformations with small changes in the reported VBench quality dimensions. Looking ahead, we will focus on designing stronger empirical false-positive calibration, component ablations, held-out transformation tests, and evaluation against adaptive watermark-removal and forgery attacks.

## AI DISCLOSURE

In this work, we used generative AI tools to (1) aid or polish writing, (2) find related work. We have reviewed all AI-assisted work. We take responsibility for the final content of this work, including text, claims or artifacts produced with the aid of generative AI.

## REFERENCES

Samar Fares, Nurbek Tastan, and Karthik Nandakumar. SPDMark: Selective parameter displacement for robust video watermarking. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 10303–10312, 2026. URL https://openaccess.thecvf.com/content/CVPR2026/html/Fares_SPDMark_Selective_Parameter_Displacement_for_Robust_Video_Watermarking_CVPR_2026_paper.html.

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

Pierre Fernandez, Hady Elsahar, I. Zeki Yalniz, and Alexandre Mourachko. Video Seal: Open and efficient video watermarking. *arXiv preprint arXiv:2412.09492*, 2024b. URL https://arxiv.org/abs/2412.09492.

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 *International Conference on Learning Representations*, 2025a. URL https://arxiv.org/abs/2501.14195.

Xuming Hu, Hanqian Li, Jungang Li, Yu Huang, Shuliang Liu, Qi Zheng, Junhao Chen, and Aiwei Liu. VideoMark: A distortion-free robust watermarking framework for video diffusion models. *arXiv preprint arXiv:2504.16359*, 2025b. URL https://arxiv.org/abs/2504.16359.

Youngdong Jang, MinHyuk Jang, JaeHyeok Lee, Feng Yang, Gyeongrok Oh, Jongheon Jeong, and Sangpil Kim. LVMark: Robust watermark for latent video diffusion models. *arXiv preprint arXiv:2412.09122*, 2024. URL https://arxiv.org/abs/2412.09122v4. Version 4, revised July 2026.

Utae Jeong, Sumin In, Hyunju Ryu, Jaewan Choi, Feng Yang, Jongheon Jeong, Seungryong Kim, and Sangpil Kim. WaTeRFlow: Watermark temporal robustness via flow consistency. *arXiv preprint arXiv:2512.19048*, 2025. URL https://arxiv.org/abs/2512.19048.

Kepan Nan, Rui Xie, Penghao Zhou, Tiehan Fan, Zhenheng Yang, Zhijie Chen, Xiang Li, Jian Yang, and Ying Tai. Openvid-1m: A large-scale high-quality dataset for text-to-video generation. *arXiv preprint arXiv:2407.02371*, 2024.

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)*, pp. 10684–10695, June 2022.

Tencent Hunyuan Foundation Model Team. Hunyuanvideo 1.5 technical report, 2025. URL https://arxiv.org/abs/2511.18870.

Team Wan, Ang Wang, Baole Ai, Bin Wen, Chaojie Mao, Chen-Wei Xie, Di Chen, Feiwu Yu, Haiming Zhao, Jianxiao Yang, Jianyuan Zeng, Jiayu Wang, Jingfeng Zhang, Jingren Zhou, Jinkai Wang, Jixuan Chen, Kai Zhu, Kang Zhao, Keyu Yan, Lianghua Huang, Mengyang Feng, Ningyi Zhang, Pandeng Li, Pingyu Wu, Ruihang Chu, Ruili Feng, Shiwei Zhang, Siyang Sun, Tao Fang, Tianxing Wang, Tianyi Gui, Tingyu Weng, Tong Shen, Wei Lin, Wei Wang, Wei Wang, Wenmeng Zhou, Wente Wang, Wenting Shen, Wenyuan Yu, Xianzhong Shi, Xiaoming Huang, Xin Xu, Yan Kou, Yangyu Lv, Yifei Li, Yijing Liu, Yiming Wang, Yingya Zhang, Yitong Huang, Yong Li, You Wu, Yu Liu, Yulin Pan, Yun Zheng, Yuntao Hong, Yupeng Shi, Yutong Feng, Zeyinzi Jiang, Zhen Han, Zhi-Fan Wu, and Ziyu Liu. Wan: Open and advanced large-scale video generative models. *arXiv preprint arXiv:2503.20314*, 2025.

Dian Zheng, Ziqi Huang, Hongbo Liu, Kai Zou, Yinan He, Fan Zhang, Lulu Gu, Yuanhan Zhang, Jingwen He, Wei-Shi Zheng, et al. Vbench-2.0: Advancing video generation benchmark suite for intrinsic faithfulness. *arXiv preprint arXiv:2503.21755*, 2025.

Xinjie Zhu, Zijing Zhao, Hui Jin, Qingxiao Guo, Yilong Ma, Yunhao Wang, Xiaobing Guo, and Weifeng Zhang. SIGMark: Scalable in-generation watermark with blind extraction for video diffusion. In *International Conference on Learning Representations*, 2026. URL https://arxiv.org/abs/2603.02882.

## A More Implementation Details

Detector $D_\theta$ maps a standardised latent of any length and size to $N$ carrier scores for each of $G$ temporal groups: a 3D convolutional stem, residual 3D convolution stages with group normalisation that downsample spatially while preserving temporal resolution, adaptive average pooling to $G \times 4 \times 4$ cells, and a per-group linear head. Adaptive pooling makes the same weights applicable to the training clips and to the longer, larger generation latents, and it partitions the latent into $G$ contiguous temporal groups ($G = 7$), each of which carries an independent copy of all $N$ carriers; detection averages over groups, so a temporal attack that damages some groups is outvoted by the others. The detector has $1.9$M parameters and is the only trained component of the system.

## B Roles of different parts in Flow-Syndrome

| Component | Role |
|---|---|
| Latent detector $D_\theta$ | Public/key-independent measurement map trained once per generator family. |
| Carrier map $(\pi_K, q_K)$ | Secret-key-derived assignment of code coordinates and sign semantics to detector carriers. |
| Syndrome representation | Keyed organization/masking of payload-bearing and anchor coordinates; anchors contribute to presence testing. |
| Repetition + temporal groups | Primary redundancy mechanisms for payload recovery under corrupted measurements. |
| Latent guidance | Generation-time optimization that moves detector scores toward target signs while constraining distortion. |
| Authenticated payload (optional) | Separate cryptographic layer for provenance authenticity; not implied by carrier keying alone. |

Table 4: Separation of roles in Flow-Syndrome. This distinction is important: keying, error tolerance, presence testing, and provenance authentication are different functions.

## C Detection when the payload is known.

When the verifier knows the expected payload, as in a robustness benchmark, we score agreement with the target codeword under the claimed key. If the hard carrier decisions were independent fair bits under the null, the number of matches would follow a $\mathrm{Binomial}(N, 1/2)$ distribution; we use this only as an idealized reference because carrier outputs produced by a common neural detector need not be independent. For reported operating points, the acceptance threshold should therefore be calibrated empirically from unwatermarked and wrong-key controls under the corresponding transformation. Because $D_\theta$ is key-independent, the same detector output can be rescored under many wrong keys at negligible additional cost, providing a convenient empirical null distribution.

## D Details on Datasets

**Detector training.** The latent detector is trained on 10,000 videos from OpenVid-1M Nan et al. (2024), a corpus of real, high-resolution clips with no overlap with the evaluation prompts. Each video contributes one clip of 81 frames sampled uniformly over its duration and resized to $256 \times 448$, which the frozen VAE of the target generator encodes once into a cached latent; all training views (marked, round-trip, clean) are derived from these latents. Training never sees a generated video: the detector learns to read the carrier from real-video latents, and the generation-time embedding of Sec. 3.5 is then applied zero-shot to the generator's own samples.

**Evaluation prompts.** Following SPDMark Fares et al. (2026), text-to-video (T2V) evaluation uses 50 prompts from the VBench Zheng et al. (2025) prompt suite, ten from each of five categories (animal, human, plant, scenery, vehicles), sampled with four fixed seeds, i.e. 200 videos per generator and configuration. Robustness (Tab. 1) is measured on a fixed subset of 20 videos, four prompts per category at one seed, so that all methods, including the baselines that generate on a different

backbone, are evaluated on identical prompts. The VBench sets of Tab. 2 use the same 20 clips. The HunyuanVideo rows of Tab. 2 are computed on all 200 videos.

**Image-to-video.** For image-to-video (I2V) we follow the protocol of SPDMark, Sec. 4.1: the same 50 prompts (four seeds each) are first rendered into 200 conditioning images with Stable Diffusion v1.5 Rombach et al. (2022) ($768 \times 432$, 50 steps, guidance 7.5), and every I2V method is conditioned on the same images. Our method and VideoSeal use Wan2.2-I2V-A14B Wan et al. (2025). VidSig and VideoShield are bound to their own backbones.

## E Evaluation Metrics

**Robustness.** We adopt the 16 post-processing attacks of the SPDMark protocol in its three categories: (I) photometric and spatial, Gaussian noise, blur, cropping, rotation, rescaling and colour jitter; (II) temporal, frame dropping, frame swapping, adjacent-frame swapping, frame insertion and trimming; (III) real-world, H.264 recompression, screen recording, denoising, subtitle overlay and combined crop-and-drop. Attack parameters follow the SPDMark release; its seventeenth attack, STTN inpainting, is omitted because the required inpainting weights are not distributed. For every attacked video we report *bit accuracy*, the fraction of payload bits recovered by blind detection from the attacked MP4 alone, averaged over the evaluation clips. Each method is evaluated at its native payload (ours 16–512 bits, VideoSeal 256, VidSig 48, VideoShield 512), which is stated alongside every number, since the same accuracy carries different evidence at different code lengths. For our method a wrong-key control is available at no extra cost because the detector is key-independent; under a wrong key the recovered bits are at chance, and a binomial test on the codeword bits at a false-positive rate of $10^{-3}$ raised no false alarm on any attacked clip.

**Video quality.** Perceptual quality is measured with the four VBench Zheng et al. (2025) dimensions SPDMark reports, computed with the released VBench evaluator: Subject Consistency (SC; DINO feature similarity across frames), Background Consistency (BC; CLIP feature similarity), Motion Smoothness (MS; frame-interpolation error under AMT) and Imaging Quality (IQ; the no-reference predictor MUSIQ). Absolute VBench scores are dominated by the generator rather than the watermark, so we read them relative to the corresponding unwatermarked generations and only compare methods that share a backbone.

## F Implementation Details

**Generators.** All generators are frozen and used with their released weights and schedulers. Wan2.2-T2V-A14B and Wan2.2-I2V-A14B Wan et al. (2025) sample 81 frames at $832 \times 480$ (T2V) or $832 \times 464$ (I2V, preserving the 16:9 conditioning image) with 40 UniPC steps and classifier-free guidance scales of 3 and 4 for the low- and high-noise experts. HunyuanVideo and HunyuanVideo-I2V Team (2025) sample 81 frames at $832 \times 480$ and $832 \times 464$ with 40 steps and guidance scale 6. Videos are stored as H.264 MP4 at 16 fps with the generator's default writer settings; all detection, including for the baselines, starts from these files.

**Detector and code.** The detector is the 1.9M-parameter 3D convolutional network of Sec. 3.3 with $G = 7$ temporal groups and a $4 \times 4$ pooled spatial grid. A payload of $L$ bits is encoded into $R = 3$ repeats of an $M = 3L$-bit keyed syndrome codeword ($N = 9L$ carriers and 288 for $L = 32$). The carrier embedder uses 16 latent elements per carrier for $L \leq 128$, 12 for $L = 256$ and 6 for $L = 512$, the largest count that fits in one temporal group at the training resolution. HunyuanVideo's VAE provides only a global latent scale, so its latents are standardised per channel with statistics measured on 200 training videos before they reach the detector.

**Training.** Detectors are trained with the objective of Sec. 3.3 (loss weights 0.5/1.0/0.1/0.1, margin 1) using AdamW (learning rate $10^{-4}$, weight decay $10^{-4}$, gradient clipping 1.0) and a batch of four clips per GPU on two NVIDIA H100 GPUs. Every positive is passed through the frozen VAE decoder, one randomly drawn video transformation and the VAE encoder before its loss is computed. A base 32-bit Wan2.2 detector is trained for 50 epochs on carrier marks. Each payload size in Tab. 1 is then obtained by fine-tuning that detector for 7 epochs (2 carrier-only warm-up epochs, then half of the positives embedded by the guidance operator itself against a frozen copy of the detector refreshed every step) on a 6,000-clip subset, with the output projection re-initialised

at a $10\times$ learning rate because its width changes with $L$. The HunyuanVideo detector is initialised from the Wan2.2 32-bit detector, whose convolutional body transfers unchanged, and trained for 50 epochs under the same recipe; model selection uses validation accuracy on guided positives after the VAE round trip.

**Generation-time embedding.** Guidance follows with the deployment settings of Tab. 3: it is applied at every sampler step in the last 40% of the trajectory with four inner steps ($\eta_0 = 0.02$, decay 0.5, element-wise bound $\kappa = 0.01$), fidelity weights $\lambda_q = 2000$ and $\lambda_{\text{temp}} = 20$, followed by four refinement steps on the final latent bounded to an RMS displacement of 0.03. The same settings are used for every generator and payload size; the overhead is 84 detector passes per video, below 2% of generation time.

**Baselines.** We compare against the three baselines of SPDMark with their official code and released weights. VideoSeal Fernandez et al. (2024a) is post-hoc and watermarks our own unwatermarked generations with its 256-bit model, so its rows are a same-content comparison; VidSig **?** (48-bit message, fine-tuned latent decoder) and VideoShield Hu et al. (2025b) (512 template bits written into the initial noise and read by DDIM inversion) are in-generation methods tied to their released backbones and generate their own clips, on ModelScope for T2V ($512 \times 512$ and $256 \times 256$, 16 frames) and on SVD-XT for I2V ($512 \times 512$, 16 frames). Attacked videos are written once, losslessly, and detected by each method's own detector; VideoShield's detector requires exactly 16 frames, so for the three frame-count-changing attacks its input is resampled onto 16 frames. SPDMark itself is not included because its release ships no pretrained decoder or extractor.