PyraTok: Language-Aligned Pyramidal Tokenizer for Video Understanding and Generation

Supplementary Material

Table of Contents:

A. Theoretical Analysis of Language-aligned Pyramidal Quantization (LaPQ)

We analyze the behavior of the LaPQ objective and the conditions under which the model avoids posterior collapse. Let θ denote all trainable parameters. The LaPQ objective in Eq. (4) is composed of smooth losses (reconstruction, codebook, autoregressive, and drift), each of which is an expectation over the training distribution $\mathcal{D}$ of video–text pairs $(\mathbf{X}, t)$, *i.e.*, $\mathcal{L}_\theta = \mathbb{E}_{(\mathbf{X},t)\sim\mathcal{D}}\big[\ell(\theta; \mathbf{X}, t)\big]$. All LaPQ modules (LoRA layers, AR head, and LFQ quantizers with stop-gradient) use differentiable operations, so $\mathcal{L}_\theta$ is a smooth, lower-bounded deep-network objective.

Why LaPQ Mitigates Posterior Collapse. At LaPQ level l, let $\mathbf{q}^{(l)} = \mathcal{Q}_l(\mathbf{q}^{(l-1)}, \mathbf{F}^{(l)}, \mathbf{e_t})$ be the (soft) assignment distribution, where $\mathbf{F}^{(l)}$ are encoder features and $\mathbf{e_t}$ is the text embedding extracted from the text t. We say LaPQ at level l is *collapsed* if there exists a fixed distribution $\bar{\mathbf{q}}^{(l)}$ such that $\mathbf{q}^{(l)} \equiv \bar{\mathbf{q}}^{(l)}$ for all $(\mathbf{X}, t) \sim \mathcal{D}$. A *fully collapsed* LaPQ posterior satisfies this for all levels $l = 1, \ldots, L$. Assume the following natural conditions:

1. **Data non-degeneracy:** The data distribution $\mathcal{D}$ is non-degenerate, *i.e.*, there exist $(\mathbf{X}, t)$ and $(\mathbf{X}', t')$ s.t. the corresponding optimal reconstruction outputs under reconstruction loss $\mathcal{L}_{\text{recon}}$ differ.
2. **Decoder injectivity on code sequences:** For any two distinct latent code sequences $\mathbf{q} \neq \mathbf{q}'$, the decoder produces distinct reconstructions $\mathcal{D}e(\mathbf{q}) \neq \mathcal{D}e(\mathbf{q}')$.
3. **Model expressiveness:** For any measurable mapping $(\mathbf{X}, t) \mapsto \mathbf{q}^{(1:L)}$, realizable via encoder features $\mathbf{F}^{(l)}$ and text embedding $\mathbf{e_t}$, there exists a parameter vector θ that realizes it to arbitrary precision.

Proposition 1 (Non-optimality of Collapsed LaPQ Posteriors)**.** *Any fully collapsed LaPQ posterior $\mathbf{q}^{(l)} \equiv \bar{\mathbf{q}}^{(l)}$ cannot minimize the LaPQ objective.*

Proof. Consider any parameter vector θ_{c} that yields a fully collapsed posterior. Then, by definition, every quantizer output distribution $\mathbf{q}^{(l)}$ is constant across inputs, hence the decoder input (the discrete code sequence $\mathbf{q}_c$) is also constant. Hence, all reconstructions are equal to $\hat{\mathbf{X}}_{\text{c}} = \mathcal{D}e(\mathbf{q}_{\text{c}})$. Then, the reconstruction loss $\mathcal{L}_{\text{recon}}(\theta_{\text{c}})$ is the expected reconstruction loss under a *constant* prediction:

$$\mathcal{L}_{\text{recon}}(\theta_{\text{c}}) = \mathbb{E}_{(\mathbf{X},t)\sim\mathcal{D}}\big[\ell_{\text{recon}}(\hat{\mathbf{X}}_{\text{c}}, \mathbf{X})\big]. \tag{5}$$

By the non-degeneracy of $\mathcal{D}$ and standard properties of L_1/SSIM/LPIPS reconstructions, there exists a non-constant mapping $\mathbf{X} \mapsto \hat{\mathbf{X}}(\mathbf{X})$ that achieves strictly lower expected reconstruction error than any constant prediction. Using the model expressiveness assumption, we can approximate such a mapping with some parameter vector θ_{nc} that yields non-collapsed assignments $\mathbf{q}^{(l)}$ and reconstructions $\hat{\mathbf{X}}(\mathbf{X})$. Therefore $\mathcal{L}_{\text{recon}}(\theta_{\text{nc}}) < \mathcal{L}_{\text{recon}}(\theta_{\text{c}})$.

We now inspect the remaining terms in the objective.

(i) Hierarchical KL and entropy terms. For a fully collapsed posterior, the hierarchical KL terms $\mathrm{D}_{\mathrm{KL}}(\mathbf{q}^{(l)} \| \mathbf{q}^{(l-1)})$ vanish only if all levels share exactly the same constant distribution; otherwise, they incur a positive penalty. Moreover, the entropy term $\mathbb{E}\big[-\mathbf{q}^{(l)} \log \mathbf{q}^{(l)}\big]$ is minimized by near one-hot distributions. A fully collapsed solution that is both constant and sharply peaked is incompatible with representing the variability in $\mathbf{X}$ and induces suboptimal hierarchical penalties.

(ii) Text-conditioned and AR terms. For a collapsed posterior, assignments $\mathbf{q}^{(l)}$ are independent of the text embedding $\mathbf{e_t}$, *i.e.*, if $\mathbf{q}^{(l)}$ is constant, it cannot match varying text embeddings. Consequently, the text-conditioned KL terms $\mathrm{D}_{\mathrm{KL}}(\mathbf{q}_i \,\|\, \mathrm{sg}(\mathbf{e_t}))$ for $\mathbf{q}_i \in \mathbf{q}^{(l)}$ and the global text–codebook alignment terms cannot be minimized across distinct texts. Similarly, the autoregressive loss $\mathcal{L}_{\text{AR}}$ cannot exploit visual or textual information because the discrete tokens do not depend on $(\mathbf{X}, t)$. By contrast, a non-collapsed posterior can strictly reduce both.

Combining all pieces, we obtain $\mathcal{L}(\theta_{\text{nc}}) < \mathcal{L}(\theta_{\text{c}})$ since $\mathcal{L}_{\text{recon}}$ is strictly lower and the remaining terms can be made no worse, and typically strictly better, by making assignments depend on $(\mathbf{X}, t)$ while respecting the regularizers. Thus θ_{c} cannot be a global minimizer of $\mathcal{L}$. □

Proposition 1 states that any fully collapsed LaPQ posterior is suboptimal under the proposed objective, provided natural structural assumptions on the data and model capacity. Therefore, gradient-based training of LaPQ is driven toward stationary points that preserve dependent discrete representations, thereby mitigating posterior collapse and encouraging high-utilization codebooks.

B. Implementation Details

PyraTok is implemented using the pretrained Wan 2.2L [53] video VAE as the backbone to ensure high-fidelity visual reconstruction. We initialize the encoder with pretrained WAN-2.2 weights, while the LaPQ module and decoder are randomly initialized. Both the encoder and decoder of Wan 2.2L are kept frozen to preserve the pretrained visual quality. To encourage the model to capture long-range temporal dependencies and motion continuity, we temporally mask 30% of frames and apply cosine-based spatial masking on each frame following [14]. To enable efficient adaptation to our multi-scale semantic learning objective without full fine-tuning, we incorporate LoRA adapters [15] with rank 16 and alpha 32 into all encoder blocks. These adapters provide lightweight parameterization while preserving the representational capacity of the backbone. For text conditioning, we employ the Qwen2.5-VL (3B) [1], referred as pretrained VLM in main paper, to extract semantically rich textual embeddings that guide both the quantization and the multimodal semantic alignment. Loss weights are set to λ_{recon}=2.5, $\lambda_{\text{codebook}}$=2.5, λ_{AR}=1.5, and λ_{drift}=0.6. To reduce memory footprint and accelerate training, we apply VAE tiling for latent-space tokenization and quantize the alignment VLM to AWQ INT-4 [30]. In PyraTok, En(·) refers to the frozen DINOv3 [40] encoder, which serves as a strong pretrained visual encoder. It is used to provide stable, high-quality visual features that anchor adaptation and prevent drift from the pretrained visual manifold.

All baselines are trained under identical dataset settings to ensure fair comparison. The average prompt length during training is ∼60 tokens. Training is conducted in three progressive stages, each designed to incrementally strengthen multimodal alignment and visual–temporal consistency.

Stage 1 — Self-Supervised Pretraining. In the first stage, we perform self-supervised pretraining focused on language alignment. Input spatial resolutions vary from 512×512 up to 2048×2048, and we train across multiple aspect ratios, including 1:1, 4:3, 3:2, 16:9, and 2:1. For temporal modeling, the number of frames ranges from $16{+}1$ to $96{+}1$, where the additional frame denotes the conditioning key frame. This stage establishes robust cross-modal grounding and spatial-temporal coherence.

Stage 2 — Text–Visual Token Alignment. The second stage incorporates text–visual token alignment through the pretrained Qwen 2.5 VL (3B) backbone. We maintain the same spatial and temporal configurations as Stage 1 for training stability. This stage refines the alignment between linguistic tokens and visual embeddings, enhancing the semantic consistency of multimodal representations.

Stage 3 — Full-Scale Fine-Tuning. In the final stage, the model is exposed to multi-resolution and multi–aspect-ratio inputs, ranging from 128×128 to 4096×4096, covering the same aspect ratios (1:1, 4:3, 3:2, 16:9, 2:1). The number of frames is kept consistent with previous stages. Due to increased resolution and GPU memory demands, the batch size is reduced from $4 \rightarrow 2$ per GPU. This stage optimizes the model with both alignment loss and a frame-level retention loss computed using DINOv3 [40], ensuring long-range temporal retention and fine-grained visual correspondence.

Figure 11. **Reconstruction loss.** The best PSNR configuration (VQ-Blocks: 4) converges at a loss of 0.12, while other ablation variants stabilize above 0.25.

All training stages are optimized using AdamW [32] with an initial learning rate of 1×10^{-5} and a cosine annealing scheduler. Gradient accumulation steps are kept constant across stages. We train on a cluster of 128×NVIDIA A100 (80 GB) GPUs. The total number of optimization steps is 30K for Stage 1, 60K for Stage 2, and 180K for Stage 3.

C. Datasets

To comprehensively train, validate, and evaluate our PyraTok model, we employ a diverse collection of large-scale video–text datasets spanning various resolutions, domains, and task-specific settings. The dataset design emphasizes both scale and diversity, ensuring robust generalization across video understanding, retrieval, and generation tasks.

C.1. Training Datasets

Droplet-10M [73] (Subset). We curate a subset of the Droplet-10M dataset, consisting of approximately 4–5 million HD videos (720p). This subset serves as the foundation for pretraining, providing broad coverage of human activities, natural scenes, and diverse motion patterns, and a dense caption distribution and consistent temporal dynamics, crucial for learning fine-grained video–text alignment. To ensure data quality and maintain high spatial fidelity, only videos at 720p or higher resolution are retained.

OpenVid-1M [35] (300K Subset). We supplement training with 300K high-quality video–caption pairs sampled from OpenVid-1M. Only HD videos are selected to maintain visual consistency. This subset contributes to expanding linguistic diversity and contextual variation, improving open-domain caption understanding and cross-modal reasoning.

Table 7. **Reconstruction performance on UltraVideo-test.**

Methods	PSNR (↑)	SSIM (↑)	LPIPS (↓)
LARP [55]	24.23	0.811	15.32
VAE-CogVideoX [67]	25.34	0.799	16.17
3D-MBQ-VAE [44]	26.78	0.838	13.23
WAN 2.2 [53]	27.83	0.858	11.06
PyraTok	**31.44**	**0.892**	**8.82**

Table 8. **Video compression at 0.034 bitrate on MCL-JCV.**

Methods	PSNR (↑)	SSIM (↑)	LPIPS (↓)
HEVC [43]	30.10	0.943	0.199
VCC [6]	32.65	0.966	0.153
MAGVIT [69]	23.70	0.846	0.144
MAGVIT-v2 [70]	26.18	0.894	0.104
3D-MBQ-VAE [44]	29.09	0.922	0.089
PyraTok (Ours)	**29.82**	**0.942**	**0.068**

UltraVideo [64] (40K with Reconstructed Captions). To enrich representation at extreme resolutions, we incorporate 40K ultra–high-definition videos (4K and 8K) from the UltraVideo dataset. Since many of these videos lack high-quality textual descriptions, we generate captions using a multimodal LLM pipeline. This enables the model to learn from high-fidelity visual data and supports scalability to higher-resolution downstream applications.

C.2. Testing and Validation Datasets

OpenVid-1M [35] (Test Split). We employ 100K samples from the OpenVid-1M test split for evaluating generalization to unseen open-domain video–text pairs. This ensures consistency with the distribution of the training data while validating model generalization under identical data conditions. **WebVid-10M [2] (Validation) and COCO [31] (Validation).** For generative evaluation, we follow the WebVid-10M and COCO-Val validation protocols. For class-guided video generation, we further evaluate on the **UCF-101** [42] dataset. **MCL-JCV [54] (Compression Validation).** To assess the effectiveness of our video compression and reconstruction, we employ the MCL-JCV benchmark. This dataset provides a controlled setup for evaluating perceptual quality and rate–distortion tradeoffs under varying compression levels.

To evaluate generalization beyond supervised training, we test the model under zero-shot conditions across diverse downstream video understanding tasks. For zero-shot action localization, we evaluate on **ActivityNet** [7] and **THUMOS14** [19], which contain diverse and complex activities. For zero-shot video segmentation, we benchmark on **YouTube-VIS 2021** [66] and **OVIS** [36]. Both datasets present challenging dynamic scenes with multiple interacting objects and frequent occlusions. For video classification, we utilize **Kinetics** [22], while for VideoQA, we adopt

Table 9. **Class-guided video generation results on UCF-101.**

Tokenizer	Type	#Tokens	#Params (Gen.)	gFVD (↓)
MAGVIT [69]	AR	1024	306M	265
MAGVIT-V2 [70]	AR	1280	307M	109
MAGVIT [69]	MLLM	1024	306M	76
MAGVIT-V2 [70]	MLLM	1280	307M	58
LARP-L [55]	AR	1024	632M	57
CogVideoX [67]	AR	6800	9.4B	626
TATS [12]	AR	4096	321M	332
Video-LaVIT [21]	AR	512	7B	280
OmniTok [57]	AR	5120	650M	191
LARP-L [55]	AR	1024	632M	99
SweetTok [46]	AR	1280	1.9B	65
PyraTok	AR	1024	2.3B	**51**

MVBench [25], a comprehensive multi-task benchmark covering spatiotemporal reasoning, action understanding, and commonsense interpretation.

D. Ablation Studies and Additional Analyses

D.1. Reconstruction Results

PyraTok achieves substantial improvements over prior video VAE tokenizers on the UltraVideo-test benchmark (Table 7). Compared to the strongest baseline, WAN 2.2, our method improves reconstruction quality by +3.61 PSNR and +0.034 SSIM while reducing perceptual error by 2.24 LPIPS. These gains indicate that the hierarchical design of LaPQ more effectively preserves both low-level spatial details and high-level semantic structure during compression. By aligning visual tokens with language semantics across multiple pyramid levels and utilizing a large-capacity codebook, PyraTok captures richer spatiotemporal representations than prior discrete VAEs that rely on single-scale quantization and smaller vocabularies. As a result, the reconstructed videos exhibit sharper textures, improved structural consistency, and better perceptual fidelity, demonstrating the effectiveness of our language-aligned pyramidal quantization framework for high-quality video reconstruction.

D.2. Video Compression

As reported in Table 8, PyraTok attains the lowest LPIPS (0.068) and competitive PSNR/SSIM on MCL-JCV [54] at a bitrate of 0.034, surpassing traditional codecs like HEVC [43] and VCC [6] in perceptual fidelity (LPIPS) by preserving fine texture and temporal coherence through semantically guided quantization.

D.3. Video Generation

We evaluate our tokenizer and generator on class-guided video generation using the UCF-101 [42] dataset. Given a target action class, the model is conditioned on the class label and asked to synthesize a short video clip from scratch. This setting measures not only low-level visual fidelity

Table 10. **Ablation study of different quantization techniques in PyraTok.** Each method is specified by its quantization type, codebook vocabulary size, and embedding dimensionality, evaluated on COCO-Val and WebVid-10M.

Quantization	Vocab	Dim	COCO-Val			WebVid-10M			Inf. Time
			PSNR (↑)	SSIM (↑)	LPIPS (↓)	PSNR (↑)	SSIM (↑)	LPIPS (↓)	
VQ [52]	4096	256	31.45	0.825	0.093	32.91	0.838	0.092	409
GVQ [20]	4096	256	32.25	0.836	0.089	33.34	0.842	0.089	438
LFQ [70]	32800	16	34.22	0.842	0.084	33.92	0.855	0.085	419
RVQ [24]	8000	512	33.92	0.849	0.078	34.22	0.865	0.079	489
LaPQ	8000	512	34.45	0.855	0.073	34.98	0.871	0.076	503
RVQ [24]	32800	16	34.78	0.869	0.076	35.27	0.879	0.074	488
LaPQ (ours)	48000	16	**35.72**	**0.879**	**0.066**	**36.05**	**0.885**	**0.071**	**492**

(appearance, motion smoothness) but also whether the generated sequence is semantically consistent with the requested action category.

We compare PyraTok against a broad set of video generative models that rely on different tokenizers and generator architectures, including MAGVIT/MAGVIT-V2, LARP-L, CogVideo, TATS, Video-LaVIT, OmniTok, and SweetTok. For all methods, we report the generative Fréchet Video Distance (gFVD), where lower values indicate better alignment with the distribution of real videos. As shown in Table 9, our method achieves the lowest gFVD on UCF-101, improving upon the strongest prior tokenizer by a substantial margin. These results indicate that our representation is better suited for high-quality, temporally coherent video synthesis, and that scaling the generator on top of our tokens directly translates into stronger video generation performance.

D.4. Ablation on VQ Techniques

Table 10 presents an ablation of the quantization module in PyraTok, where each row corresponds to a different way of discretizing the encoder features, specified by its quantization type, vocabulary size, and embedding dimensionality. The simple single-codebook baseline VQ [52] (4096/256), with vocal size of 4096 and a dimension of 256, yields the weakest reconstruction quality on both COCO [31] and WebVid [2], confirming that a single global codebook is insufficient to capture the variability of natural image–video data. Introducing a group structure in GVQ [20] (4096/256) slightly improves PSNR and SSIM, and reduces LPIPS; however, the gains are modest because each group still operates with a relatively small shared codebook. The lookup-free single-block variant, LFQ [70] (32800/16), increases the effective vocabulary while reducing the per-code dimension, resulting in a clear improvement in PSNR and SSIM, and a lower LPIPS, indicating that finer local code assignment is beneficial.

Residual quantization with a higher-dimensional code space, RVQ [24] (8000 / 512), further reduces distortion over vanilla VQ, and replacing the residual codebook with our latent product quantizer, LaPQ (8000 / 512), yields another consistent improvement, showing that decomposing the latent channels into product codebooks makes better use of the same vocabulary size. When we combine residual modeling with LFQ-style factorization, RVQ (32800 / 16) achieves even better performance, but our full LaPQ (Ours, 48000 / 16) achieves the best performance across all metrics on both validation sets, with the highest PSNR/SSIM and lowest LPIPS, while incurring only a small increase in inference time compared to simpler schemes. Overall, results demonstrate that LaPQ's combination of lookup-free factorization and product–residual coding provides a significantly more expressive and distortion-resilient discrete representation than standard VQ, GVQ, LFQ, or RVQ under comparable computational budgets.

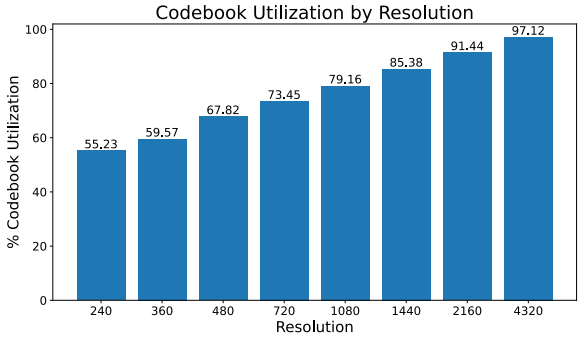


Figure 12. **Codebook utilization as a function of input resolution.** Higher resolutions activate a larger fraction of the vocabulary, indicating that PyraTok effectively exploits the increased spatial support to encode more diverse semantics.

D.5. Codebook Utilization vs. Resolution

We further analyze how the effectiveness of our tokenizer scales with input resolution by measuring the percentage of active codewords at different spatial resolutions (Fig. 12). As the resolution increases from 240p to 4320p, codebook utilization rises monotonically from 55.23% to 97.12%, indicating that higher-resolution inputs naturally excite a richer subset of the learned vocabulary rather than collapsing to a small set of frequently used tokens. In particular, utilization already exceeds 79% at 1080p and surpasses 90% in the 4K regime (2160p and 4320p), suggesting that the proposed

Table 11. **Ablation on losses for understanding tasks.**

	THUMOS14	ActivityNet	MVBench
✗ $\mathcal{L}_{\text{dift}}$	31.27	27.62	83.32
✗ $\mathcal{L}_{\text{AR}}$	32.45	27.98	79.45
✗ $\mathcal{L}_{\text{dino}}$ & $\mathcal{L}_{\text{AR}}$	29.29	26.78	81.57
✗ $\mathcal{L}_{\text{text-cond. alignment}}$	30.22	27.55	83.56
✗ $\mathcal{L}_{\text{vision-commitment}}$	32.67	28.21	84.23
✗ $\mathcal{L}_{\text{text-codebook alignment}}$	31.11	27.07	83.91
✓ **All losses**	**33.17**	**29.11**	**86.03**

pyramidal design can effectively exploit the larger spatial support to express more diverse and fine-grained semantics. This trend confirms that our discrete latent space remains expressive and well-populated as we scale to high-resolution video, which is critical for both faithful reconstruction and downstream video-language understanding tasks.

D.6. Ablation on Losses for Video Understanding

We ablate each component of the training objective on three video understanding benchmarks: THUMOS14 [19] and ActivityNet v1.3 [7] for temporal action localization, and MVBench for video question answering (Table 11). With the full objective, our model achieves $33.17/29.11$ Avg. mAP on THUMOS14/ActivityNet and 86.03 mAP on MVBench. Removing the drift regularizer $\mathcal{L}_{\text{drift}}$, which anchors the adapted encoder to the pretrained VAE manifold, degrades performance by $1.90/1.49$ mAP on THUMOS14/ActivityNet and by 2.71 points on MVBench, indicating that maintaining a stable latent space is important for robust transfer across both localization and QA.

The autoregressive alignment loss $\mathcal{L}_{\text{AR}}$ has a different effect: dropping it leads to a relatively small drop on temporal localization ($0.72/1.13$ mAP), but causes a pronounced 6.58-point decline on MVBench. This suggests that sequence-level token modeling is especially critical for high-level video reasoning, where the model must integrate information over longer temporal horizons. When we remove both the DINO-guided visual loss and the autoregressive loss ($\mathcal{L}_{\text{dino}} + \mathcal{L}_{\text{AR}}$), performance drops most severely on TAL (by 3.88 and 2.33 mAP on THUMOS14 and ActivityNet, respectively) and by 4.46 points on MVBench, highlighting the complementarity between discriminative visual supervision and global token prediction.

We further study the codebook-related objectives, as described in Equation (3). Ablating the text-conditioned alignment term $\mathcal{L}_{\text{text-cond.}}$ reduces performance by $2.95/1.56$ mAP on THUMOS14/ActivityNet and by 2.47 points on MVBench, while removing the text–codebook alignment $\mathcal{L}_{\text{text-codebook}}$ yields a similar degradation ($2.06/2.04$ mAP and 2.12 points). These results confirm that both local token–text alignment and global codeword–text alignment are necessary to maintain semantically structured latents that generalize well across detection and QA tasks. In contrast, dropping the vision-comment loss $\mathcal{L}_{\text{vision-commitment}}$ produces the smallest degradation (at most $0.50/0.90$ mAP on THUMOS14/ActivityNet and 1.80 points on MVBench), suggesting that, for downstream understanding, the semantic shaping of the codebook is more critical than the pure vision commitment penalty. Overall, the complete loss formulation is consistently superior, validating our multi-part objective for unified video understanding.

D.7. Ablation on Encoder Training Strategy

Table 12 analyzes the impact of different encoder training strategies on reconstruction quality. Fully training the encoder yields reasonable performance but does not fully leverage the strong priors of pretrained video representations. Freezing the encoder slightly improves stability but limits adaptability to the tokenizer. Training the encoder from scratch achieves competitive reconstruction quality but requires substantially higher computational cost (798 GPU hours), making it impractical for large-scale training. In contrast, adapting the pretrained encoder using lightweight LoRA modules provides the best trade-off between performance and efficiency. Our LoRA-based approach achieves the highest reconstruction fidelity on both COCO and WebVid-Val, improving PSNR and SSIM while significantly reducing LPIPS compared to all baselines. These results suggest that parameter-efficient adaptation effectively preserves the pretrained encoder's semantic priors while allowing the model to specialize for pyramidal quantization, resulting in better visual fidelity with substantially lower training cost than training from scratch.

D.8. Token Representation Analysis

Figure 13 visualizes the distribution of learned video tokens using t-SNE on the WebVid10M dataset. The progressive stages of our Language-aligned Pyramidal Quantization (LaPQ) demonstrate increasingly structured and semantically coherent token representations. Early pyramid levels (LaPQ1 and LaPQ2) exhibit partially mixed clusters, reflecting low-level visual features and coarse semantic grouping. As quantization proceeds to deeper levels (LaPQ3 and LaPQ4), the token embeddings become significantly more compact and well-separated, indicating improved semantic disentanglement and higher representational clarity. In contrast, the tokens produced by WAN 2.2 show noticeably higher overlap and less distinct cluster boundaries, suggesting weaker semantic organization in the learned latent space. These results highlight how the hierarchical design of LaPQ progressively refines token representations, enabling clearer semantic separation and more expressive video tokenization.

E. Additional Qualitative Results

We present additional qualitative examples for video generation and understanding tasks.

Table 12. **Effect of encoder training strategy on reconstruction quality.**

Method	Coco Val			WebVid-Val			Training Time (GPU Hours)
	PSNR↑	SSIM↑	LPIPS↓	PSNR↑	SSIM↑	LPIPS↓	
Encoder Fully Trainable	33.27	0.852	0.081	33.75	0.843	0.083	267
Encoder Frozen	34.82	0.855	0.079	34.78	0.844	0.075	276
Encoder Scratch (Random Init)	34.11	0.865	0.075	35.76	0.869	0.078	798
Encoder + LoRA (PyraTok)	**35.72**	**0.879**	**0.066**	**36.05**	**0.885**	**0.071**	348

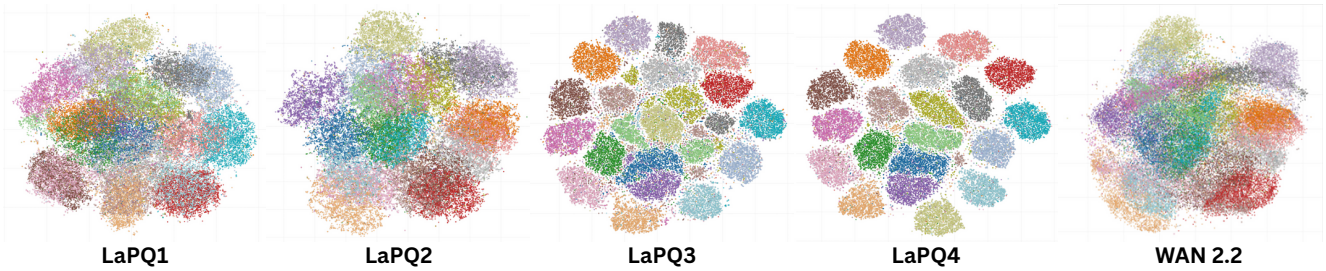


Figure 13. **t-SNE visualization** of learned token embeddings on WebVid10M.

E.1. Zero-shot Video Segmentation

Given an input video and a natural language text t, we leverage the language-aligned discrete representation produced by PyraTok to obtain zero-shot, text-guided spatio-temporal masks. Specifically, we first pass the video through the frozen PyraTok encoder and its Language-aligned Pyramidal Quantization (LaPQ) hierarchy and extract the quantized features from the last quantization block, denoted by $\mathbf{q}^{(L)} \in \mathbb{R}^{T' \times H' \times W' \times d}$, which capture high-level, text-aligned semantics at a compressed spatio-temporal resolution. We then decompose the input text into a set of semantic units (typically content words or short phrases), $\{w_1, \ldots, w_K\}$, and obtain a language embedding $\mathbf{e}_{w_k}$ for each unit using the same vision–language model employed during PyraTok training. For every semantic unit w_k, we compute a similarity score between $\mathbf{e}_{w_k}$ and each token in $\mathbf{q}^{(L)}$ (*e.g.*, via cosine similarity in the shared embedding space), yielding a token-level relevance map $\mathbf{S}^{\text{tok}}_{w_k}(t', h', w')$. This relevance map is then upsampled to the original video resolution, following the encoder downsampling pattern (or via decoder-aligned projection), to produce a dense per-pixel score volume $\mathbf{S}_{w_k}(x, y, t)$ for each semantic unit. We treat these volumes as unary potentials in a fully connected 3D Conditional Random Field (CRF) defined over the spatio-temporal lattice (x, y, t), with pairwise terms encouraging spatial smoothness aligned to image edges and temporal consistency across adjacent frames. Running mean-field inference in this 3D-CRF refines the raw scores into a binary segmentation mask $\mathbf{M}_{w_k}(x, y, t) \in \{0, 1\}$ that assigns each pixel in each frame to semantic unit w_k. Repeating this procedure iteratively over all semantic units in the prompt yields a set of word-level, zero-shot, text-guided segmentation masks that are both spatially precise and temporally coherent across the video.

We compare our language-guided tokenizer with OmniTokenizer [57], LARP [55], and VideoVAE+ [63], on diverse scenes in Fig. 14 and novel-category examples in Fig. 15. Existing tokenizers often yield coarse, blob-like masks with strong label confusion: OmniTokenizer and LARP tend to over-smooth object boundaries and merge adjacent instances (*e.g.*, bus and road, trees and background), while VideoVAE+ frequently misses thin structures such as bike frames, surfboards, and traffic signs, or hallucinates spurious regions in uniform areas. These methods also struggle with rare or fine-grained concepts, leading to incomplete segmentation of small objects (*e.g.*, cat ears, surfboard tips) and inconsistent labeling across the image. In contrast, PyraTok produces masks that are both sharper and more semantically aligned with the ground truth, accurately separating foreground from background and preserving thin structures. In Fig. 15, our model demonstrates strong zero-shot generalization: it cleanly segments unseen categories such as millennium falcon, tordelli, golden retriever, Pikachu, and axolotl, and simultaneously grounds multiple text queries (*e.g.*, "golden retriever / puppy / grass field / vegetation") in the correct regions, highlighting that our language-aligned tokens carry richer semantic information than prior VAE-based tokenizers.

E.2. Video Question Answering

For all question answering results, we adopt Qwen2.5 VL(3B) [1] as the default vision–language (VLM) backbone to generate open-ended answers from our video representations. Given an input clip, we first encode the video with our proposed PyraTok VAE and extract the discrete representations from *all* quantization blocks. These multi-scale features are projected into the language embedding space and prepended to the question tokens, yielding a unified conditioning sequence for the autoregressive decoder. The Qwen2.5 VL 3B model then performs conditional text generation to produce the final answer. All VQA inferences

are executed using the Text Generation Inference (TGI) pipeline from HuggingFace [17], which provides a stable and reproducible deployment for our qualitative analysis.

Furthermore, across Figs. 16 to 18, we compare our PyraTok model against Qwen 2.5 VL (3B), VideoVAE+, OmniTokenizer, and LARP on diverse video scenarios, including action sequences (helicopter crash, motorcycle chase, aircraft destruction), transformation events (monster emergence, firetruck-to-robot), and emotional interactions (a surprise proposal). The lower-capacity baselines (Qwen2.5-3B and VideoVAE+) often produce vague or partially incorrect explanations, while OmniTokenizer and LARP capture events more reliably but still miss finer details. PyraTok consistently provides the most accurate, complete, and context-aware interpretations across all scenarios, demonstrating stronger temporal reasoning, causal understanding, and fine-grained visual grounding compared to the competing models.

E.3. Action Localization

We tackle temporal action localization in long, untrimmed videos by directly operating in the discrete latent space of PyraTok. Given a video of N RGB frames and a textual description of the target action, we first encode every frame with our pyramidal tokenizer. Empirically, we observe that $\mathbf{q}^{(1)}$ offers the best trade-off between semantic expressiveness and temporal resolution: it preserves subtle motion cues (*e.g.*, arm swing before an arrow release, the instant of impact in a punch in Fig. 19) that are strongly smoothed out in deeper levels. We therefore use $q^{(1)}$ as our frame-level features. For each frame t, we spatially pool the tokens $\mathbf{q}_t^{(1)}$ (mean-pooling over space) to obtain a compact frame descriptor $\mathbf{v}_t \in \mathbb{R}^d$. The textual query is encoded by the same language backbone used for PyraTok's cross-modal training, producing a normalized embedding $\mathbf{z} \in \mathbb{R}^d$. We compute cosine similarity scores $s_t = \langle \mathbf{v}_t, \mathbf{z} \rangle$ for all frames, which yield a dense text–video alignment signal over time.

To robustly localize an action interval, we evaluate similarities in a sliding-window fashion. The video is partitioned into overlapping chunks $(t, t+K-1)$ of length K=25 frames (with stride 1 in all experiments). For each chunk we aggregate the frame scores, $S_t = \frac{1}{K} \sum_{i=t}^{t+K-1} s_i$, resulting in a 1D confidence trajectory $\{S_t\}_{t=1}^{N-K+1}$ that reflects how strongly the query is grounded in each temporal neighborhood. We then decode this trajectory into contiguous segments using a longest-connected-sequence algorithm: (i) we threshold S_t at a fixed confidence τ to obtain a binary sequence; (ii) identify all maximally connected high-confidence segments; and (iii) select the segment with the highest average score as the predicted action interval. For multi-action queries, we iteratively remove the selected interval and repeat, merging overlapping segments when necessary. The resulting segments define our temporal action predictions.

Fig. 19 and Fig. 20 visualize localized action segments for different tokenizers on several challenging examples. For each text query, the ground-truth (GT) segment is shown in blue, and the predictions obtained from VideoVAE [63],+, SweetTok [46], LARP [55], and PyraTok are displayed as colored bars beneath. The baselines consistently exhibit temporally diffuse and fragmented activations: their similarity signals tend to fire on visually similar but semantically off-target frames, producing multiple short segments or systematically shifted intervals. For instance, in Fig. 19, in the clip "A girl shoots an arrow", both VideoVAE + and SweetTok activate broadly over the whole sequence and fail to concentrate probability on the actual release moment, while LARP on several disjoint intervals before and after the shot. In contrast, PyraTok yields a single, compact segment that tightly aligns with the GT span around the arrow release. A similar pattern appears for text query "A person fires a shotgun", where baseline tokenizers localize earlier or later segments, whereas PyraTok localizes correctly.

The advantages of our fine-grained features are even more evident for actions with multiple sub-events. In Fig. 19 example "An MMA fighter knocks down his opponent with a punch to the face" and "... with a kick to the face", the motion unfolds rapidly and is preceded by visually similar feints. VideoVAE + and SweetTok tend to spread confidence over the entire exchange, leading to overly long or misaligned segments, while LARP often localizes only part of the motion (*e.g.*, the wind-up but not the impact). PyraTok, by contrast, localizes a concise window centered around the decisive contact, closely matching the GT. In Fig. 20, for "A person performs two overhead presses", our method produces two high-confidence video segments that track both overhead press repetitions, whereas baselines either miss the second repetition or merge the two into one coarse interval. For complex, extended actions such as "A man and a woman engage in sword fighting" and "Three missiles are launched from a desert", baseline tokenizers again show scattered activations, localizing short segments around high-motion frames or transient explosions, and resulting in under-coverage of the GT. In contrast, PyraTok yields more accurate localization.

E.4. Text-2-Video Generation

To assess the usefulness of our tokens for generative modeling, we couple PyraTok with a conditional video decoder built on Qwen 2.5 VL [1]. Concretely, we treat the text encoder of Qwen 2.5 VL as a frozen condition network and fine-tune its video decoder to autoregressively predict PyraTok codes. Given a textual prompt, we first encode the prompt into language features, which are injected into a transformer-based decoder that models the joint distribution over all spatio–temporal tokens from our four quantizers. The decoder predicts the next token conditioned on the text and all previously generated tokens, until a full sequence

of discrete video codes is obtained. These codes are then passed through the PyraTok VAE decoder to synthesize the final video. Thanks to PyraTok's compact yet expressive representation, this pipeline can generate videos at 20 FPS with resolutions up to 4K.

Fig. 21 and Fig. 22 show qualitative comparisons on text-to-video generation where we keep the Qwen 2.5 VL decoder architecture fixed and only swap the underlying tokenizer. OmniTokenizer and LARP tend to under-utilize fine-grained textual cues, often missing localized attributes such as the "two egg halves" in the ramen bowl or the "motion blur on pedestrians" in the neon street scene, and producing over-smoothed or distorted structures in complex compositions like the tree city and Mars spaceport. SweetTok better preserves global layout but still struggles with high-frequency details and subtle style descriptors (*e.g.*, HDR interior lighting, crisp spray around the polar bear), leading to muted textures and inconsistent object shapes. In contrast, PyraTok yields samples that more faithfully reflect both global scene descriptions and fine-grained phrases in the prompts. The additional objects specified in the text appear at the correct locations, motion-related cues are rendered more plausibly, and material and lighting properties (glossy chocolate surface, bioluminescent foliage, cinematic city glow) are captured with higher fidelity. Fig. 23 further illustrates 4K text-to-video generation for a 3-second clip, where PyraTok renders fine-grained details and maintains sharp structures, demonstrating that our multi-scale quantization supports high-resolution, text-aligned video synthesis.

E.5. High-resolution Frame Reconstruction

We further evaluate our method on 4K frame reconstruction in Fig. 24. At this resolution, prior tokenizers struggle to preserve fine structures and high-frequency textures. VideoVAE+ [63] produces strong over-smoothing: the coral branches, tree leaves, and fur on the buffalo become noticeably blurred, and small objects such as distant boats and fire lamps nearly vanish in the zoomed-in crops. OmniTokenizer [57] improves sharpness but introduces ringing and aliasing along high-contrast boundaries (*e.g.*, the product watch edges and mountain silhouettes), and often exhibits color bleeding in specular regions. SweetTok [46] and LARP [55] retain more detail yet still suffer from blocky artifacts on repetitive textures (grass, foliage, brick walls) and inconsistent reconstruction of tiny highlights, such as reflections on the watch bezel and lights on the night harbor. In contrast, our PyraTok reconstructions remain consistently crisp and coherent. Objects across all scenes—from coral polyps and reef fish to product shots and distant architectural details—retain sharp contours and clean textures without haloing. Fine-grained elements such as fur strands, leaf veins, and small fruits are faithfully preserved, demonstrating that our pyramidal tokenization scales effectively to ultra-high resolutions while avoiding the blurring and aliasing present in prior methods.

In qualitative video reconstruction comparisons (Fig. 25–Fig. 28), existing tokenizers show consistent limitations across diverse scenes. TokenFlow [37] and SweetTok often oversmooth high-frequency content, causing foliage, clothing textures, and facial details to blur, and small or thin structures to distort or disappear; they also introduce blocky artifacts under large motion. LARP better preserves local contrast but frequently produces ringing around boundaries and unstable illumination, leading to flickering highlights and shadows. MotionAura [44] improves temporal smoothness yet still suffers from identity drift in talking-head sequences and ghosting around fast movements, reducing perceptual realism. Moreover, as previous methods were trained on low-resolution data, their high-resolution reconstructions exhibit temporal artifacts such as frame stuttering. In contrast, our 4K-trained PyraTok preserves high-frequency detail and temporal coherence, producing smooth and stable video.

E.6. Adapting Pretrained T2V Priors with PyraTok

We further study whether PyraTok can serve as a drop-in tokenizer for existing text-to-video priors. To this end, we replace the original VAE/tokenizer in three pretrained models, *i.e.*, MAGVIT-V2 [33, 70], OmniGenV2 [60] (autoregressive priors), and MotionAura [44] (discrete diffusion prior), and fine-tune only the prior on 10k clips from OpenVid-1M [35] so that it models PyraTok codes. Under identical prompts and sampling hyper-parameters, and across all architectures, using the native tokenizer leads to typical failure modes: colors and exposure drift over time, geometry "breathes" (*e.g.*, wobbling backgrounds and object contours), high-frequency details such as dough surface texture or water droplets quickly collapse into smooth blobs, and object semantics are weakly preserved (*e.g.*, inconsistent shape of the claw-machine robot or citrus slices). After swapping in PyraTok, the same priors produce videos that are both more semantically aligned with the prompts and markedly more temporally consistent. In Fig. 29, MAGVITv2+PyraTok maintains stable neon lighting in the arcade, preserves the dough's volume and hand pose across frames, and keeps the boiling dumplings sharp with coherent bubble motion. In Fig. 30, OmniGenV2+PyraTok yields crisp tree trunks and facial details with reduced frame-to-frame jitter, while the splashing juice exhibits smoother, physically plausible trajectories. Similarly, MotionAura+PyraTok in Fig. 31 suppresses flicker in backgrounds. These improvements indicate that PyraTok's multi-scale discrete representation reduces artifacts and exposes a more structured latent space, making it easier for both autoregressive and diffusion priors to model long-range spatio-temporal dependencies and maintain object identity over time, even with minimal fine-tuning.

Figure 14. **Zero-shot semantic segmentation comparison across various scenes.** Results illustrate PyraTok's ability to recover fine object boundaries, preserve small structures, and produce semantically coherent segmentations across diverse domains. Details in Appendix E.1.

Figure 15. **Zero-shot semantic segmentation examples produced by PyraTok using only text prompts.** Each column shows an input image, the corresponding segmentation mask predicted by PyraTok, and the set of text labels used. Results span diverse object types, demonstrating PyraTok's ability to segment both rare and common entities without task-specific training. Details in Appendix E.1.

Question: How does the copter in the video crash?

Qwen2.5-3B : The helicopter seems to lose control after clipping part of a building structure, which sends it spinning before it crashes.

VideoVAE+ : The helicopter hits a wire hanging between the buildings and can't stabilize afterward, leading to the crash.

OmniTokenizer : The helicopter crashes after making contact with an overhead line stretched across the street.

LARP : It collides with a tight cable strung between the buildings. The wire snags the helicopter, throwing it off balance and causing it to crash.

PyraTok : The helicopter gets tangled in the steel wire stretched between the buildings, loses control instantly, and crashes onto the street

Question: From where did the big green monster come into the glass cage?

Qwen2.5-3B : The green monster looks like it arrives from another area of the facility and ends up inside the glass cage during the commotion.

VideoVAE+ : The monster spawns inside the glass chamber after the man disappears completely.

OmniTokenizer : The creature is actually the man in the cell changing form. He turns into the green monster inside the glass cage.

LARP : The monster comes from the man inside the cage. He transforms into the creature while still being held in the glass containment unit.

PyraTok : The big green monster appears after the man inside the cage transforms into his creature form. He morphs from a normal human into the monster while still inside the glass chamber.

Figure 16. **Qualitative comparison of video understanding across two challenging temporal-reasoning questions.** For each example, we show representative frames from the input video and the corresponding answers produced by different models. Baselines typically provide generic or partially plausible descriptions that miss key causal events. In contrast, PyraTok produces precise, temporally grounded explanations that correctly capture fine-grained interactions, entity tracking, and scene dynamics. These examples demonstrate PyraTok's ability to support detailed video comprehension and causal reasoning from short video sequences. Discussion in Appendix E.2.

Question: What is happening in the video?

Qwen2.5-3B : A man on a motorcycle is racing across an open area as the plane takes off, and it looks like he's trying to catch up after being left behind.

VideoVAE+ : A man is riding a motorcycle near an aircraft preparing to leave. He appears to be trying to reach the plane while something is chasing him.

OmniTokenizer : The man rides a motorcycle in the desert with dinosaurs chasing him. He heads for a plane that's starting to take off and drives into it from the back ramp

LARP : A man on a motorcycle is being chased by dinosaurs. He tries to escape by reaching a transport plane that is lifting off and drives up the rear ramp to get inside.

PyraTok : The man is speeding on a motorcycle while being chased by several dinosaurs. He races toward a military-style plane as it begins taking off and manages to board through the rear ramp while still riding the bike.

Question: Why does the woman in the video seem surprised?

Qwen2.5-3B : The woman seems surprised because the man suddenly shows her something in his hand, catching her off guard.

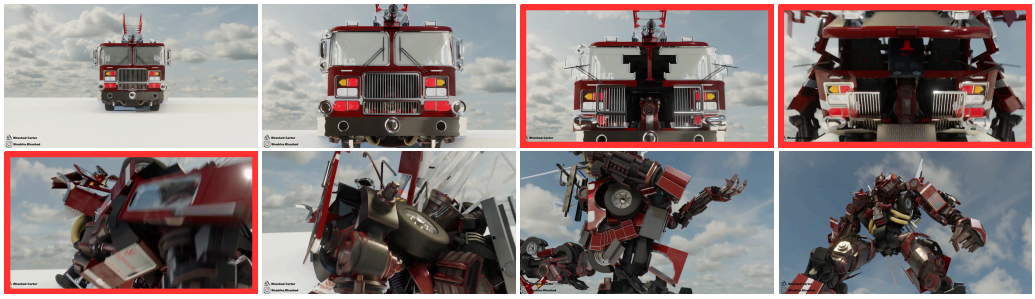
VideoVAE+ : She appears surprised because the man gives her something unexpected during their beach picnic.

OmniTokenizer : She seems surprised because the man takes out a ring and proposes, which she wasn't fully prepared for.

LARP : She looks surprised because the man is proposing to her, and she wasn't expecting it. His gesture catches her off guard emotionally.

PyraTok : The woman reacts with surprise because the man suddenly proposes, and she's overwhelmed by the unexpected romantic moment.

Figure 17. **Qualitative comparison of video reasoning ability across models.** Models describe major actions in two dynamic scenes (a motorcycle escape from dinosaurs and a surprise beach proposal). PyraTok delivers the most precise and context-aware answers across both scenarios. Discussion in Appendix E.2.



Question: Where did the red giant robot come from?

Qwen2.5-3B : The giant robot seems to come from inside the fire truck when the vehicle breaks apart and reveals the machine.

VideoVAE+ : The robot appears to rise out of the fire truck after it opens up and shifts its structure.

OmniTokenizer : It comes from the fire truck, which transforms piece by piece into the robot.

LARP : The robot originates from the fire truck. The vehicle folds and reshapes itself until it becomes the large red robot.

PyraTok : The red giant robot emerges by transforming from the fire truck. The truck's parts shift and reassemble into the robot form.

Question: How does the aircraft get destroyed?

Qwen2.5-3B : The plane crashes in flames after something impacts it on the runway as it's trying to take off.

VideoVAE+ : The aircraft is blown apart after being struck by an explosive attack from the runway area during its takeoff roll.

OmniTokenizer : It gets destroyed because a missile hits it just as it's lifting off, triggering an explosion that breaks the jet apart.

LARP : The jet is destroyed by a missile launched from the portable launcher. The explosion hits it during takeoff and tears the aircraft apart.

PyraTok : The aircraft is destroyed when the person in black fires a missile at it right as it begins to take off. The missile strikes the jet, causing a large explosion that rips it apart

Figure 18. **Qualitative comparison of video understanding on two transformation- and action-level reasoning tasks.** Baseline methods provide generic or underspecified descriptions (*e.g.*, stating that the robot "comes from the fire truck"), often missing key causal events, responsible agents, and transformation mechanics. In contrast, PyraTok produces precise, temporally grounded explanations that correctly identify object transformations, causal triggers, and scene dynamics, such as the fire truck's parts reassembling into the robot or the person in black firing the missile that destroys the aircraft. Discussion in Appendix E.2.

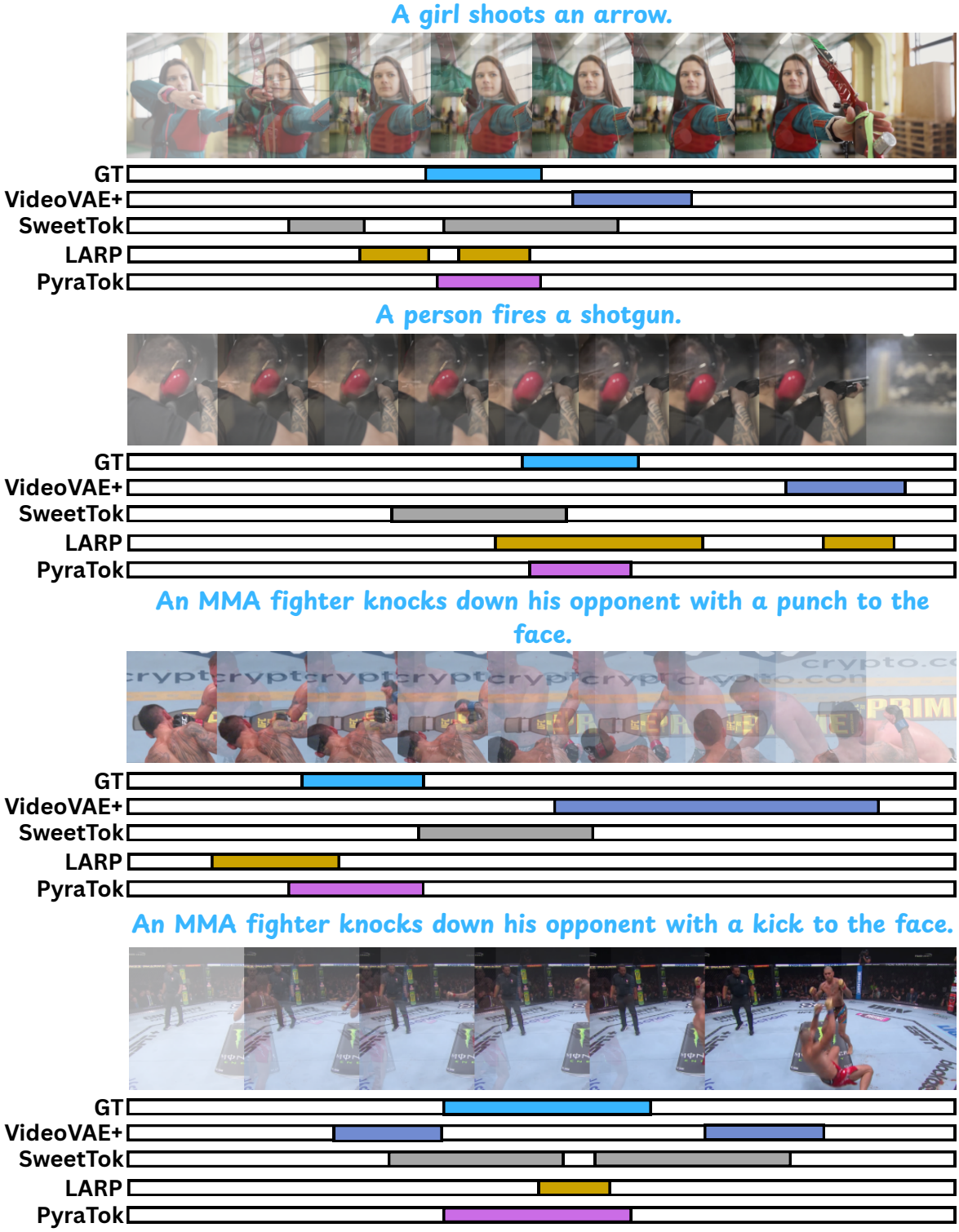


Figure 19. **Action localization results comparing PyraTok with several baselines across different video scenarios.** For each prompt, the top row shows sampled video frames, followed by temporal action segments for the ground truth and predictions from each method. PyraTok produces action intervals that align more closely with the ground-truth boundaries, demonstrating improved temporal precision and robustness across diverse actions. Details in Appendix E.3.

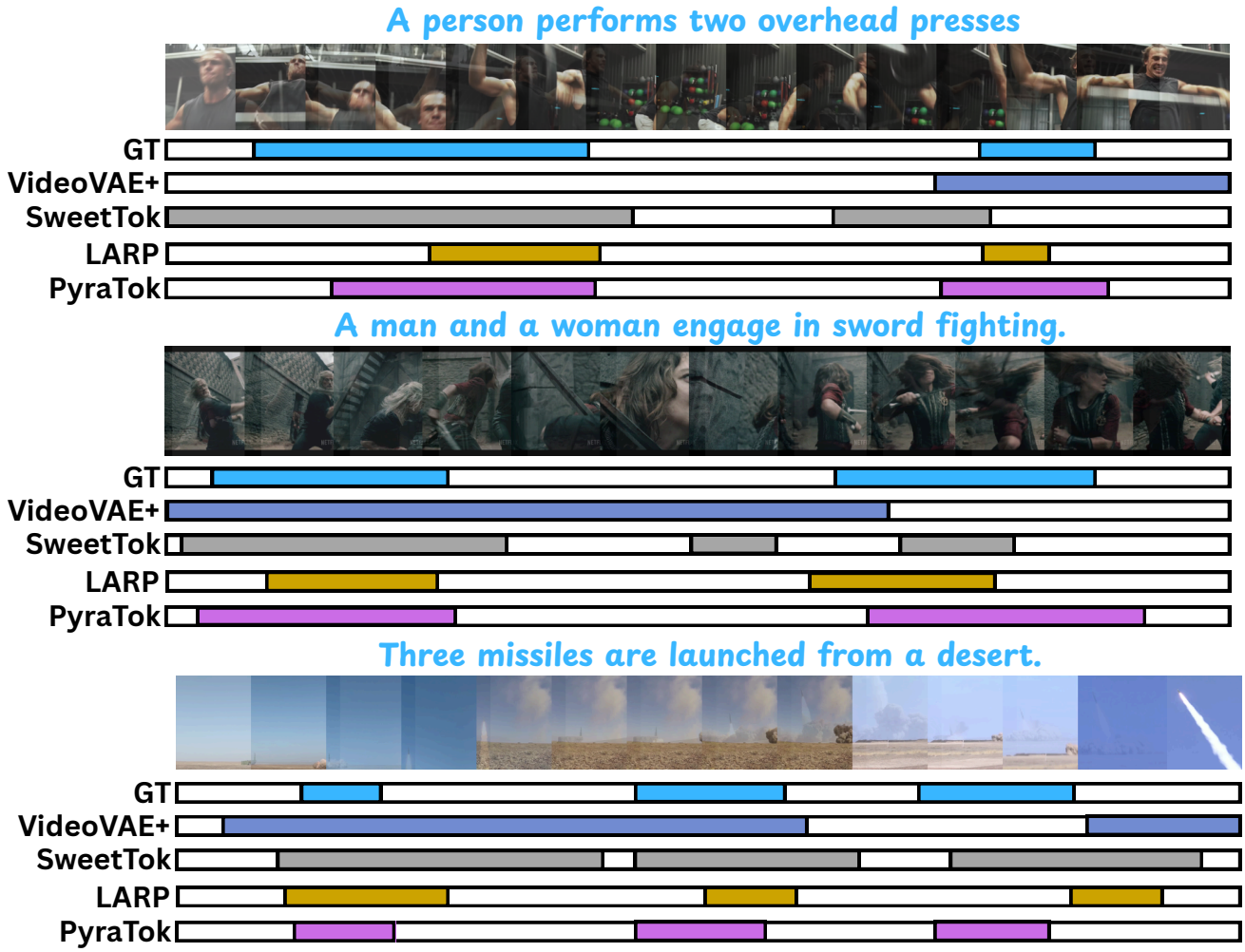


Figure 20. **Additional action localization comparisons across diverse scenarios.** Each example shows sampled frames followed by ground-truth action intervals and model predictions. PyraTok consistently yields temporally aligned and coherent action segments, reducing fragmentation and improving boundary accuracy compared to prior baselines. Details in Appendix E.3.

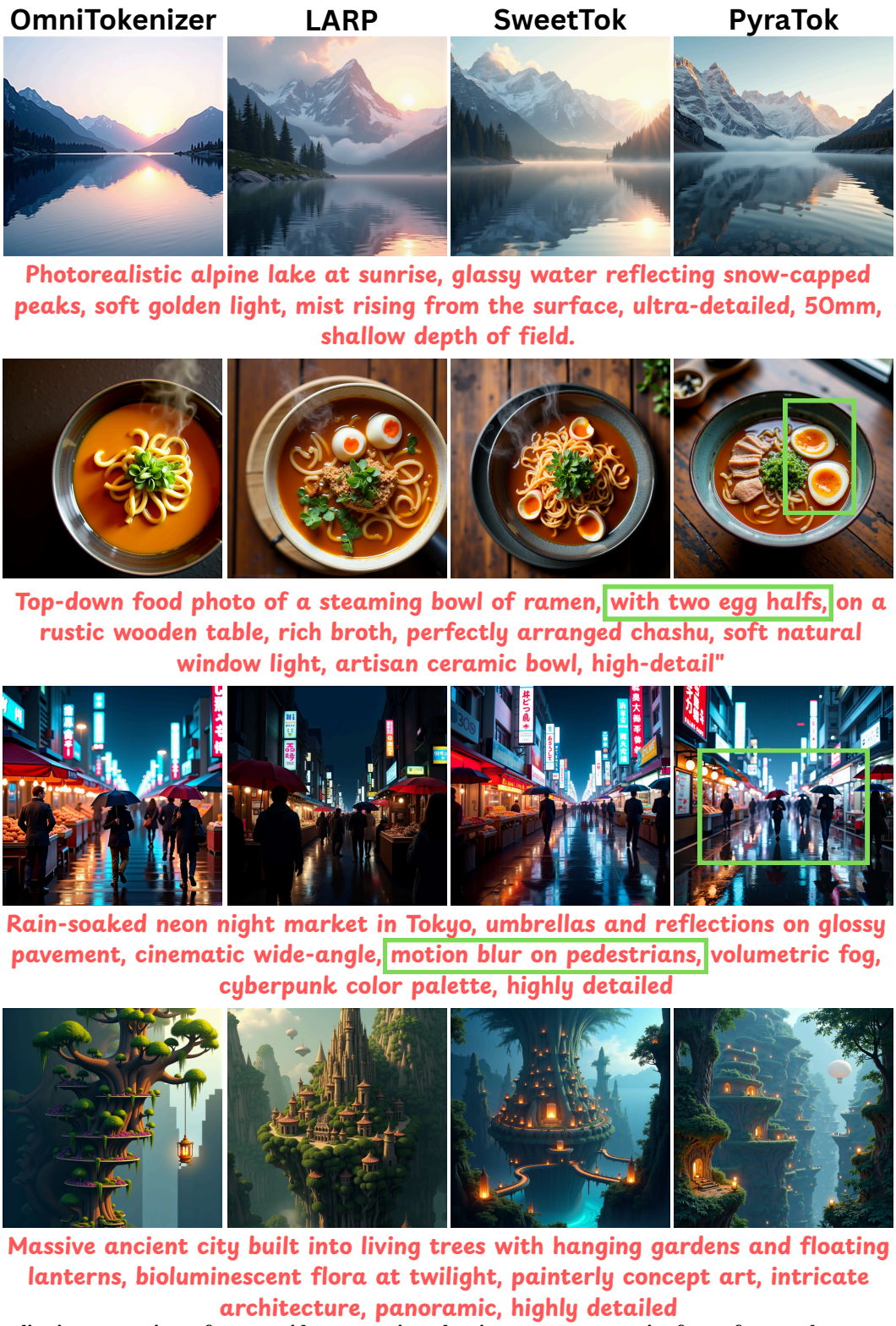


Figure 21. **Qualitative comparison of text-to-video generation, showing one representative frame from each generated clip across diverse prompt categories**, including photorealistic landscapes, detailed food scenes, night-market environments, and stylized concept art. Although only a single frame per video is shown, the green boxes highlight fine-grained details faithfully produced by PyraTok, such as the correct depiction of "two half eggs" in the ramen scene and realistic motion blur on pedestrians in the night-market prompt, illustrating PyraTok's ability to accurately interpret and render subtle textual attributes in text-to-video generation. Discussion in Appendix E.4.

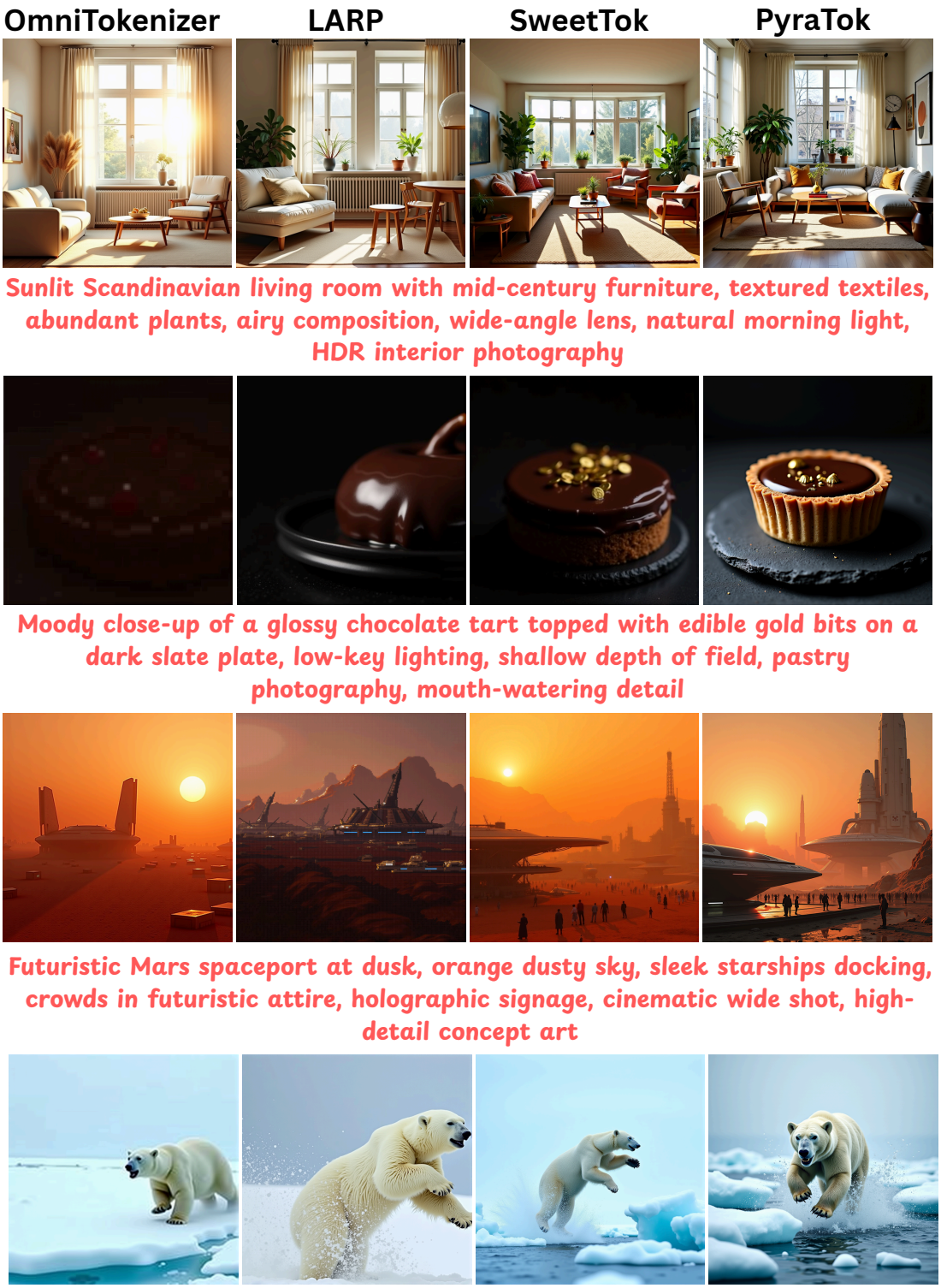


Figure 22. **Single-frame text-to-video generation comparisons across diverse prompts**, including interior scenes, food close-ups, sci-fi concept art, and wildlife action. PyraTok consistently captures fine-grained details, accurate lighting, textures, object geometry, and scene composition, demonstrating strong prompt alignment and high-fidelity generation across varied visual domains. Discussion in Appendix E.4.

A 4k resolution drone video of a futuristic city skyline at night, flying between tall neon-lit skyscrapers with cars moving far below. Aerial shot, high angle, bird's-eye view. The camera glides steadily across the city, revealing illuminated buildings and busy highways. Long-exposure lighting, cinematic neon glow.

A 4k resolution cinematic video of a lone traveler wearing a cap walking through a foggy forest at dawn. Captured in a high-angle shot with iridescent lighting, the slow panning camera reveals trees swaying gently and birds gliding across the misty sky.

Figure 23. **Text-to-video generation PyraTok results at 4K resolution**. The first example depicts a futuristic neon-lit city captured by a flying drone, where PyraTok maintains crisp details, stable long-exposure lighting, and smooth camera motion. The second example illustrates a foggy forest at dawn featuring "a lone traveler wearing a cap." Even though the person occupies only a tiny fraction of the scene, PyraTok accurately renders fine-grained details, such as the cap on the traveler's head, demonstrating strong text-alignment and high-resolution consistency in large-scale, wide-angle video generation. Discussion in Appendix E.4.

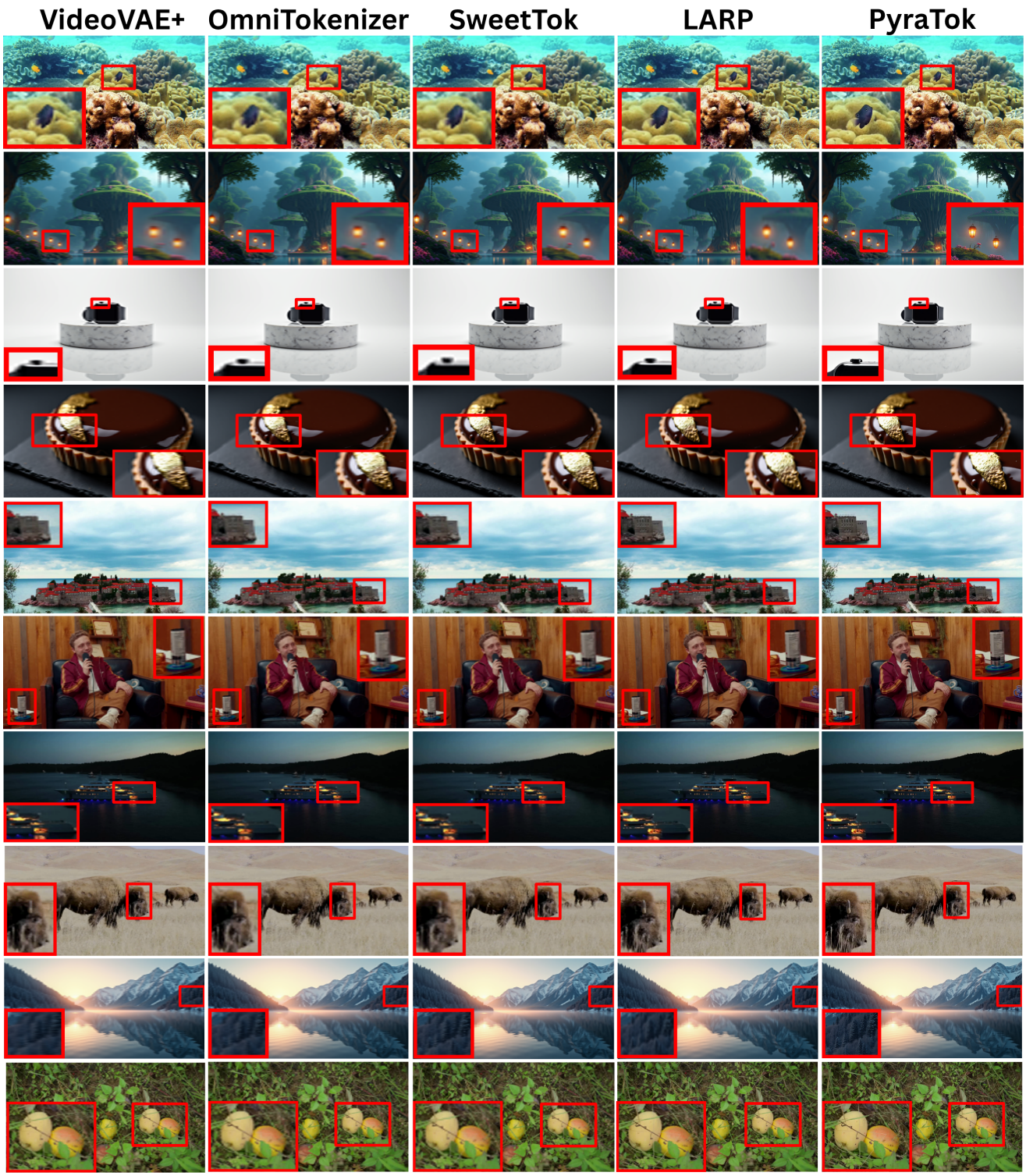


Figure 24. **Qualitative comparison of single-frame reconstruction across diverse scenes**, including underwater environments, fantasy landscapes, product renders, food close-ups, urban views, interview settings, night scenes, wildlife, mountain vistas, and natural textures. Each row shows outputs from one method for the same input frame, with red boxes highlighting fine details, such as small objects, textures, reflections, and thin structures, used to compare reconstruction fidelity, sharpness, and color consistency. PyraTok preserves fine details reliably and delivers consistent, high-quality reconstructions across all scene types. Discussion in Appendix E.5.

Figure 25. **Qualitative comparison of video reconstruction methods on a fast-moving dinosaur sequence** containing dense foliage, small background animals, and detailed facial motions. Each row shows outputs from one method on the same frames. Red boxes highlight challenging regions such as vegetation, moving creatures, and fine facial details, where differences in sharpness, temporal coherence, and motion fidelity are most apparent. Discussion in Appendix E.5.

Figure 26. **Qualitative comparison of PyraTok with other video reconstruction methods on a dynamic café scene** containing multiple people, complex indoor lighting, and detailed textures. Each row shows outputs from one method on the same frames. The scene highlights challenges such as preserving facial details, clothing patterns, reflections, and background structures, allowing visual comparison of reconstruction sharpness and temporal consistency. Details in Appendix E.5.

Figure 27. **Qualitative comparison of PyraTok with other video reconstruction methods** on an indoor interview scene featuring expressive hand motions, detailed facial appearance, and complex background textures, revealing differences in preserving facial clarity, hand motion coherence, and fine background details such as books, fabrics, and stone textures. Details in Appendix E.5.

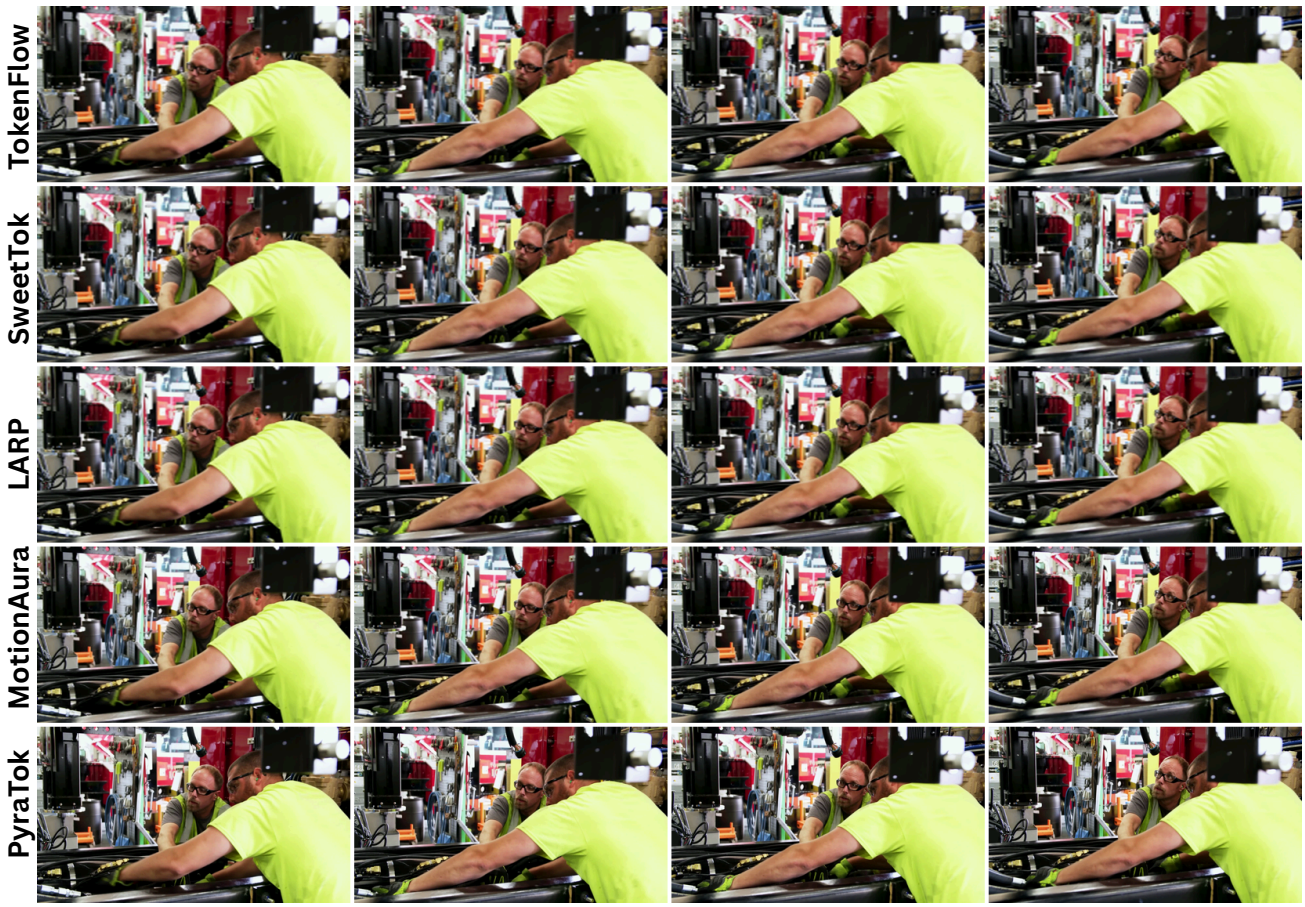


Figure 28. **Qualitative comparison of PyraTok with other video reconstruction methods on a workshop scene** involving fast arm movements, reflective machinery, and detailed background clutter, highlighting differences in preserving motion clarity, fine textures on tools and equipment, and the stability of subtle visual details under rapid motion. Details in Appendix E.5.

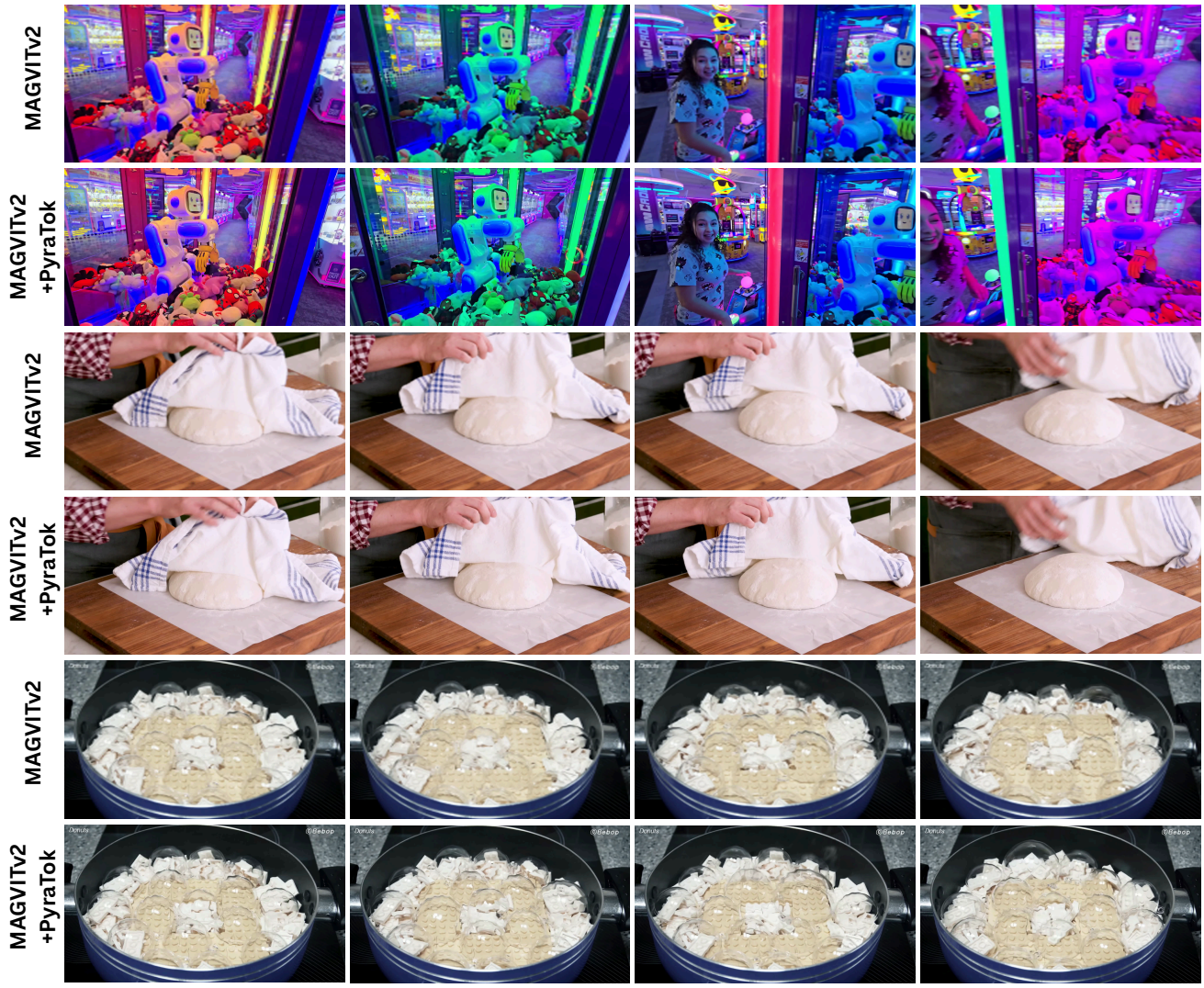


Figure 29. **Comparison of video reconstruction quality when replacing the default MAGVIT-V2 [70] VAE with our PyraTok VAE.** Each pair of rows shows frames generated by the original MAGVIT-V2 (top) and the enhanced MAGVIT-V2 + PyraTok configuration (bottom). Across diverse scenes, including arcade environments with complex lighting, close-up dough preparation, and detailed cooking sequences, PyraTok improves visual sharpness, color consistency, and fine-detail preservation, demonstrating its effectiveness as a drop-in VAE replacement for higher-quality video generation. Discussion in Appendix E.6.

Figure 30. **Comparison of video generation quality when replacing the default VAE of OmniGen-V2 [60] with our PyraTok VAE.** For each scene, the top row shows frames produced by the original OmniGen-V2, while the bottom row shows frames from OmniGen-V2 + PyraTok. PyraTok improves texture sharpness, color fidelity, and fine-detail preservation, demonstrating its effectiveness as a universal, high-quality VAE substitute for diverse video generation pipelines. Discussion in Appendix E.6.

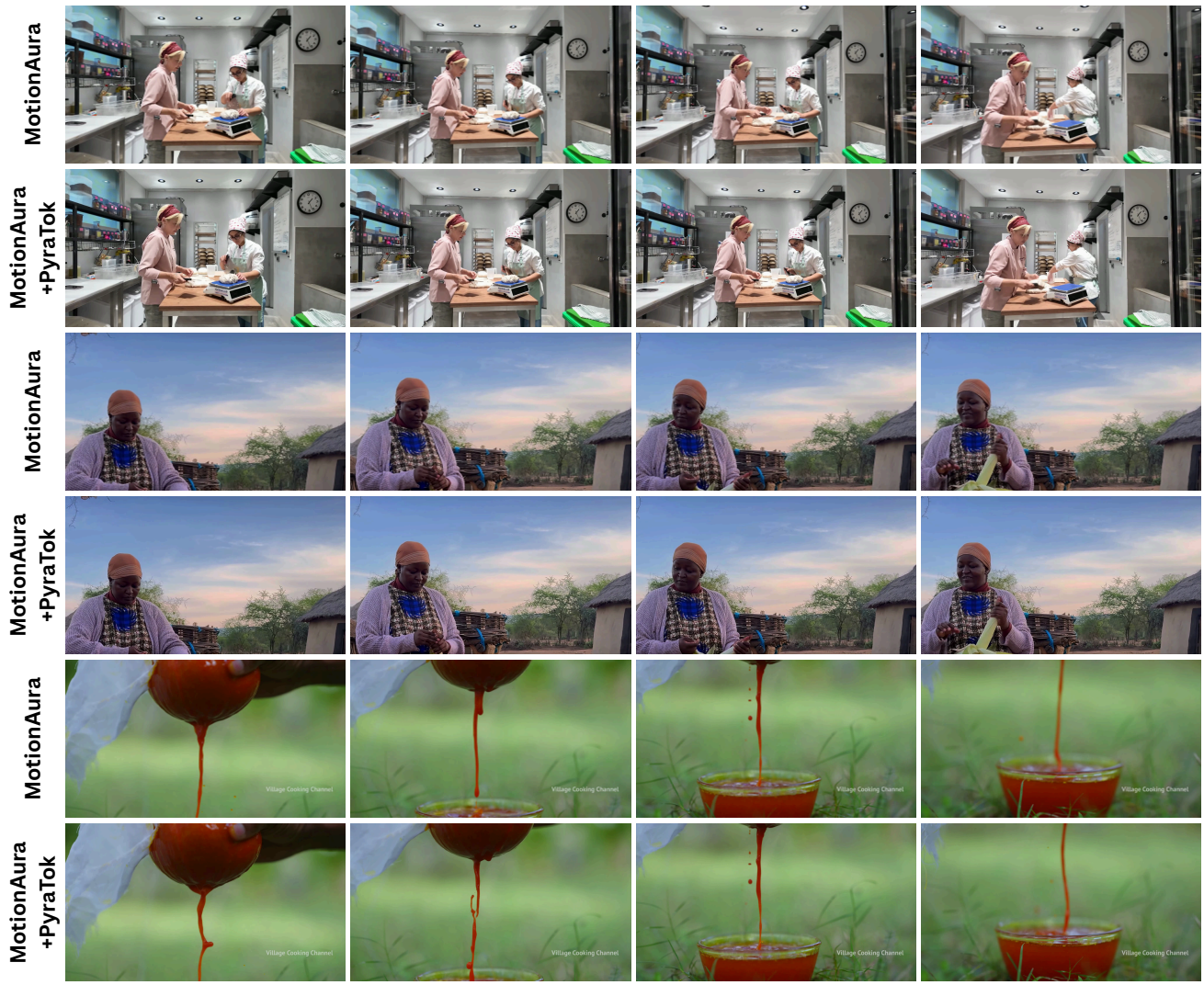


Figure 31. **Comparison of video generation quality** when substituting the default VAE in MotionAura [44] with our PyraTok VAE. For each example, the top row shows frames produced by the original MotionAura, while the bottom row shows results from MotionAura + PyraTok. Across kitchen scenes, outdoor human activity, and close-up liquid motion, PyraTok enhances sharpness, preserves fine textures, and improves temporal consistency—demonstrating its effectiveness as a high-quality VAE replacement for improving realism and detail in MotionAura-generated videos. Discussion in Appendix E.6.