ICML
International Conference
On Machine Learning

# FAST-AR

## Fast Autoregressive Video Diffusion & World Models with *Temporal Cache Compression and Sparse Attention*

Dvir Samuel, Issar Tzachor, Matan Levy,
Michael Green, Gal Chechik, Rami Ben-Ari

*Streaming video from an autoregressive diffusion model rendered **~10× faster with FAST-AR***

OriginAI

THE HEBREW UNIVERSITY OF JERUSALEM

NVIDIA

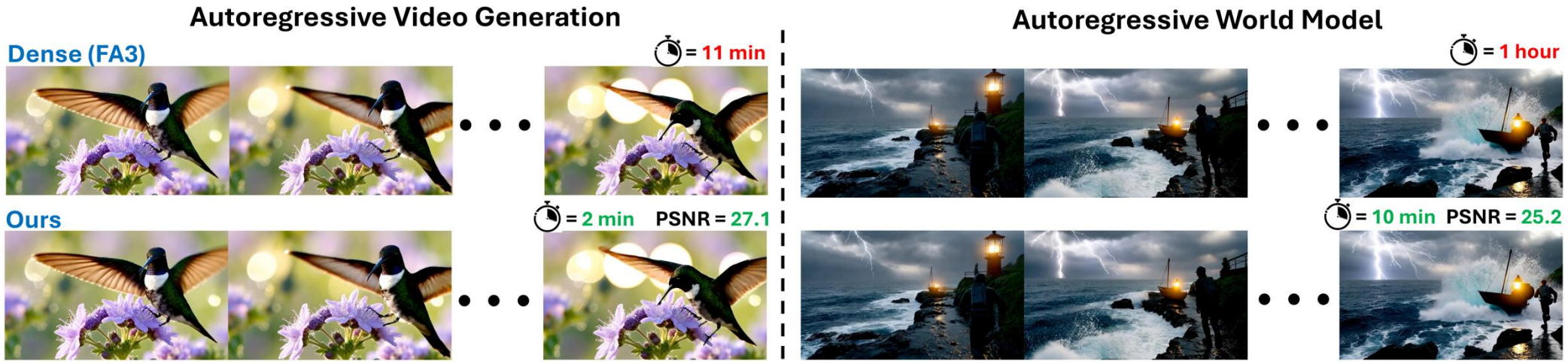
THE PROBLEM

# Streaming generation hits an attention wall

**Growing cache.** Every new frame attends to all previous frames, kept in a KV cache that never stops expanding.

**Compounding cost.** Per-step attention grows linearly; cumulative cost over a video grows as $O(T^2)$.

**Capped context.** To fit memory, models shorten the context window — which breaks long-range consistency.




*Same quality, a fraction of the time — **11 min → 2 min** on an H100 (video diffusion, left; world model, right).*

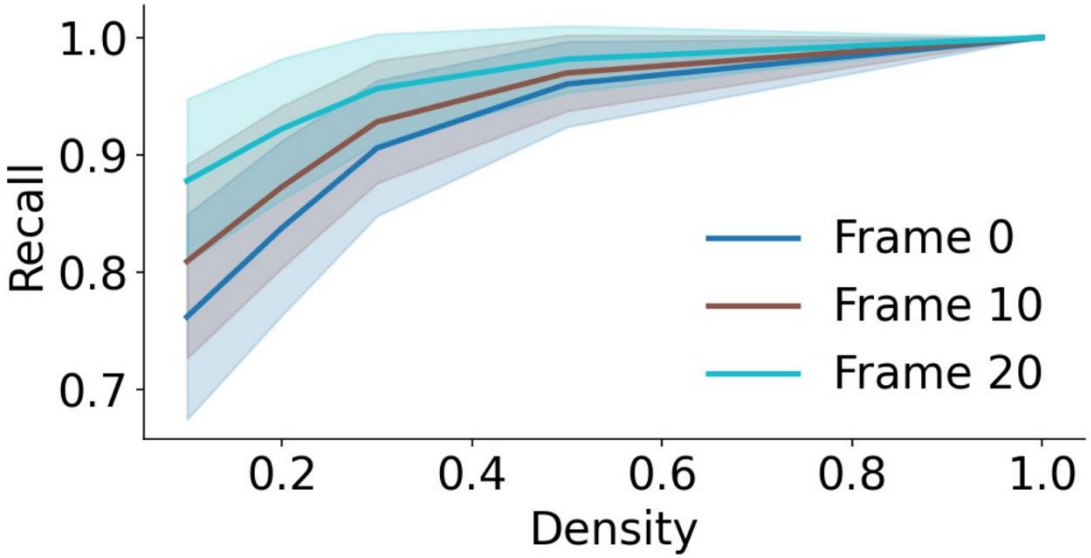
WHY IT'S POSSIBLE

# Attention here is massively redundant

**Duplicate keys** reappear, nearly unchanged, across frames → `TempCache`

**Q / K evolve slowly** and cluster by meaning, so most score computations are wasted → `AnnSA`

**Only a few prompt tokens** matter for any single frame → `AnnCA`




**Keep just 30% of attention → preserve ~85% of the mass**

*(holds across layers and frames).*

METHOD

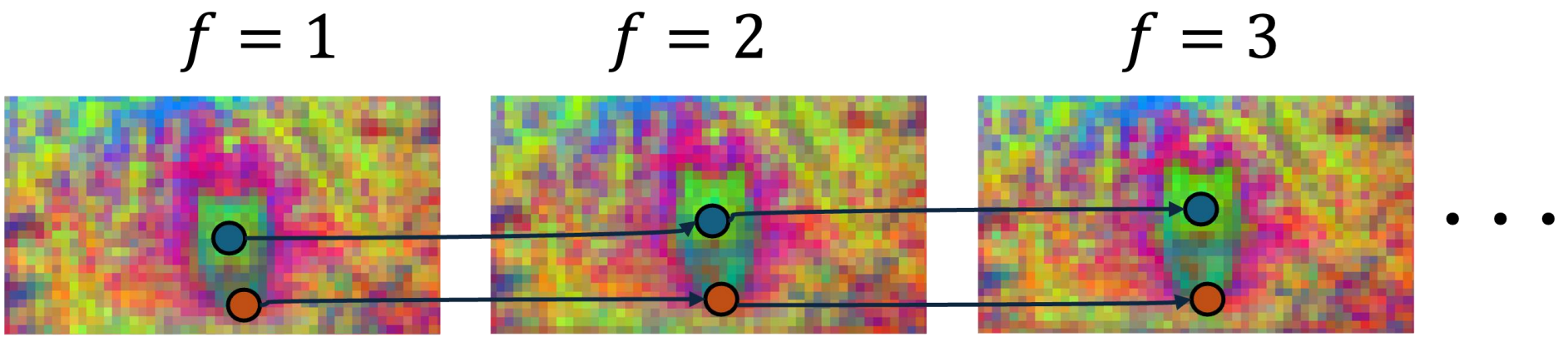
# FAST-AR: attention as nearest-neighbor search

**Core idea.** Only a few keys matter per query, retrieve them with a fast ANN index (LSH or quantization), then attend only over that small set. No retraining, no custom kernels.

`TempCache` merges keys that recur across frames → bounds the KV cache.

`AnnSA` limits each query to its semantic bucket → cost independent of context length.

`AnnCA` keeps only frame-relevant prompt tokens → cheap cross-attention.




*TempCache matches keys across frames via temporal correspondence.*

RESULTS · SPEED

# Real-time generation — Ours finishes first

**Real-time generation simulation — 2-min video · 2880 frames @ 24 FPS**

**02:25.4**
elapsed (time-lapse)

**Dense (FlashAttention-3)**

**FPS 5.9** 1372 / 2880

**Ours (FAST-AR)**


**FPS 19.8** 2880 / 2880

**FlowCache + RadialAttn**

**FPS 7.2** 1214 / 2880

In the time **Ours** generates the full 2-min video (2,880 frames), Dense has produced under half the frames and FlowCache+RadialAttn fewer still — and both keep slowing as the KV cache grows.

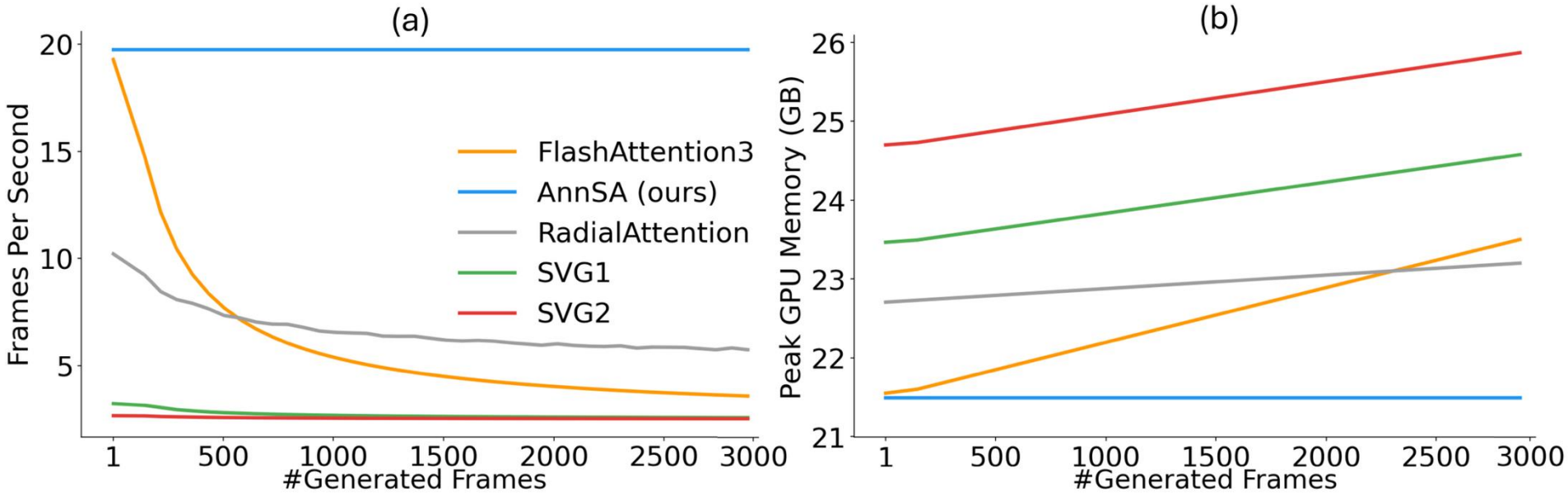
RESULTS · Efficiency

# Up to 10× faster at constant memory

**Quality unchanged** — VBench 84.0 vs 84.1 (dense).

**KV cache → ~16% density**, ~90% attention recall.

**Best baseline:** only ×4.4 — and quality collapses to 45.




*To 3,000 frames: ours holds FPS (left) and peak memory (right) flat; dense and offline-sparse baselines slow down and grow.*

RESULTS · QUALITY

# We match dense quality; baselines drift


Same prompt, panels left→right: Dense FA3, **Ours**, RadialAttn, SVG2, SVG. Over a 2-min rollout, ours stays sharp and identity-stable — indistinguishable from dense — while the sparse baselines blur and drift.

FAST-AR · ICML 2026

# Training-free acceleration for autoregressive video diffusion

**TempCache** — bounds the KV cache.

**AnnSA** — sparsifies self-attention.

**AnnCA** — sparsifies cross-attention.

· Plug-and-play on video diffusion and world models
· 5–10× faster
· constant GPU memory
· near-identical quality.

**Project Page**

dvirsamuel.github.io/fast-auto-regressive-video

*Thank you!*