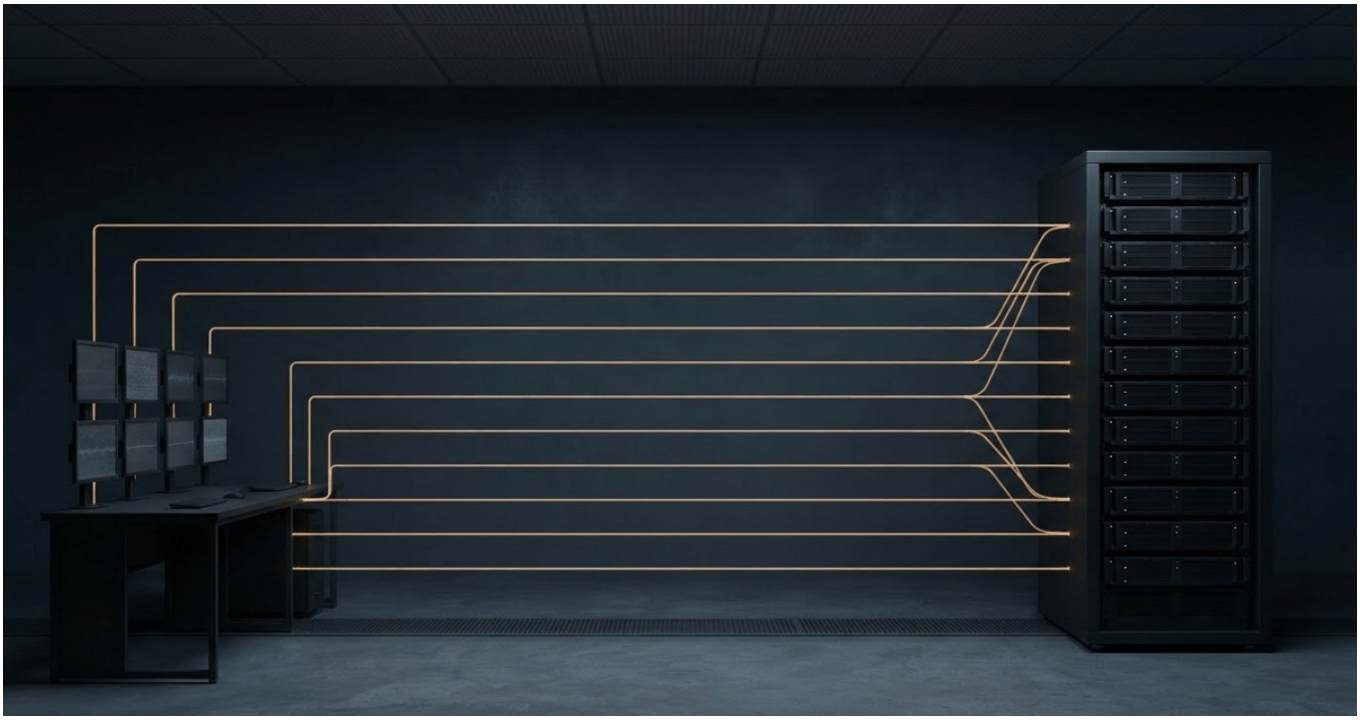
Real-Time AI Streaming Is a Product Trend, Not a Benchmark

Julien Laurenceau | Published 2026-09-01 | Updated 2026-09-30

https://pepitedata.com/real-time-ai-streaming-product-trend/

PepiteData | LinkedIn: Julien Laurenceau | X: @Julien_Lau



Update, September 2026: the model in the production path is only one layer. I mapped the supply chain around it, from routers and prompt logs to provenance, drift and pipeline sizing: the AI supply chain as a production dependency.

Last week, a text model shipped with a headline figure of 1,107 tokens per second. I can read about 250 words per minute. At its claimed best case it produces text roughly two hundred times faster than I can read. Even the median production figure below is faster than any human reader by more than an order of magnitude. The number that should interest you is different: on OpenRouter's own production panel for the same model, the median observed throughput was about 170 tokens per second. Both numbers are true. Understanding why both matter, and which one you should put in a capacity plan, is the subject of this article.

Because something changed this summer. AI products are moving from "the answer arrives" to "the answer flows". That is not a benchmark improvement. It is a change in what the product is, and it lands on whoever sizes the infrastructure.

What actually changed

What changed is the delivery unit: AI output moved from a completed response you wait for to a stream you watch while it is produced. For most of the last three years, an AI feature behaved like a batch job: the user typed, waited several seconds, and received a complete response. Latency was a waiting problem: make the model faster, the wait shrinks, the user is happier. Simple.

Three recent developments break that model:

- **Text.** Inception Labs' Mercury 2.5 Preview went live on OpenRouter on August 31, 2026. It is a diffusion language model: instead of generating tokens one after another, it produces and refines multiple tokens in parallel, which is how it reaches its 1,107 tokens/sec on standard GPUs, a figure Inception Labs claims and OpenRouter repeats on the model page. OpenRouter announced it on X: "The fastest reasoning LLM is now live exclusively on OpenRouter". (OpenRouter model page)
- **Video.** fal's H3 Max generates video faster than playback speed: a 5-second 768p clip with audio in under 3 seconds, more than 50 times faster than the base model by Design Arena's comparison. Engineers wired it to a continuous 24/7 livestream where chat directs every scene, and Pieter Levels shipped his own chat-directed stream, "Infinite Slop", a day later. fal announced it on X: "Video generation is now faster than real time. An infinite broadcast where every frame is generated on the fly". (fal model page, SaaSCity's write-up, Infinite Slop)
- **Interactive worlds.** Google DeepMind's Genie 3 generates navigable 3D worlds at 24 frames per second and 720p, in real time, for a few minutes at a time. (DeepMind announcement)

The common thread is not raw speed. It is the delivery unit. When generation runs at or beyond playback speed, output stops being a file you download after processing and becomes a stream you watch while it is produced. The user's role changes too: in the fal stream, the audience types prompts and steers the broadcast live. The audience became an input.

The three numbers that now define quality

The three numbers that now define streaming quality are time to first token, token throughput and inter-token latency, and concurrent streams. If your monitoring, your SLAs and your vendor conversations are built around mean request latency, you are measuring the wrong regime. Three metrics define the streaming experience:

Time to first token (TTFT). How long until something appears on screen. A response can have excellent average latency and still feel dead if the first two seconds are silent. Interfaces are judged on this number before anything else.

Token throughput and inter-token latency. Once the stream starts, how fast does it flow? Above roughly human reading speed, extra tokens per second buy nothing for the individual user. This is the counterintuitive part: beyond the reading threshold, speed stops being a user experience feature and becomes a capacity feature. The same GPU that serves one user instantly can serve more concurrent streams.

Concurrent streams. This is the number most capacity plans are missing. In the streaming regime, the unit of load is a session that holds a GPU connection open for the full duration of generation. Ten thousand requests per second of short completions and two thousand simultaneous long generations are very different hardware problems, and request-rate numbers do not tell them apart.

The OpenRouter data makes the point better than any argument I could construct. The panel reports a round-trip latency P50 of roughly 1.3 seconds, but a P90 of about 8.4 seconds (these percentiles average across production traffic of mixed shapes; they are neither a single stream's inter-token rhythm nor a ceiling). A capacity decision built on the median would undersize for the tail. Streaming makes the tail a first-class product property: a stalled stream is visible to the user in a way a slow batch response never was.

Why this is a product trend, not a benchmark story

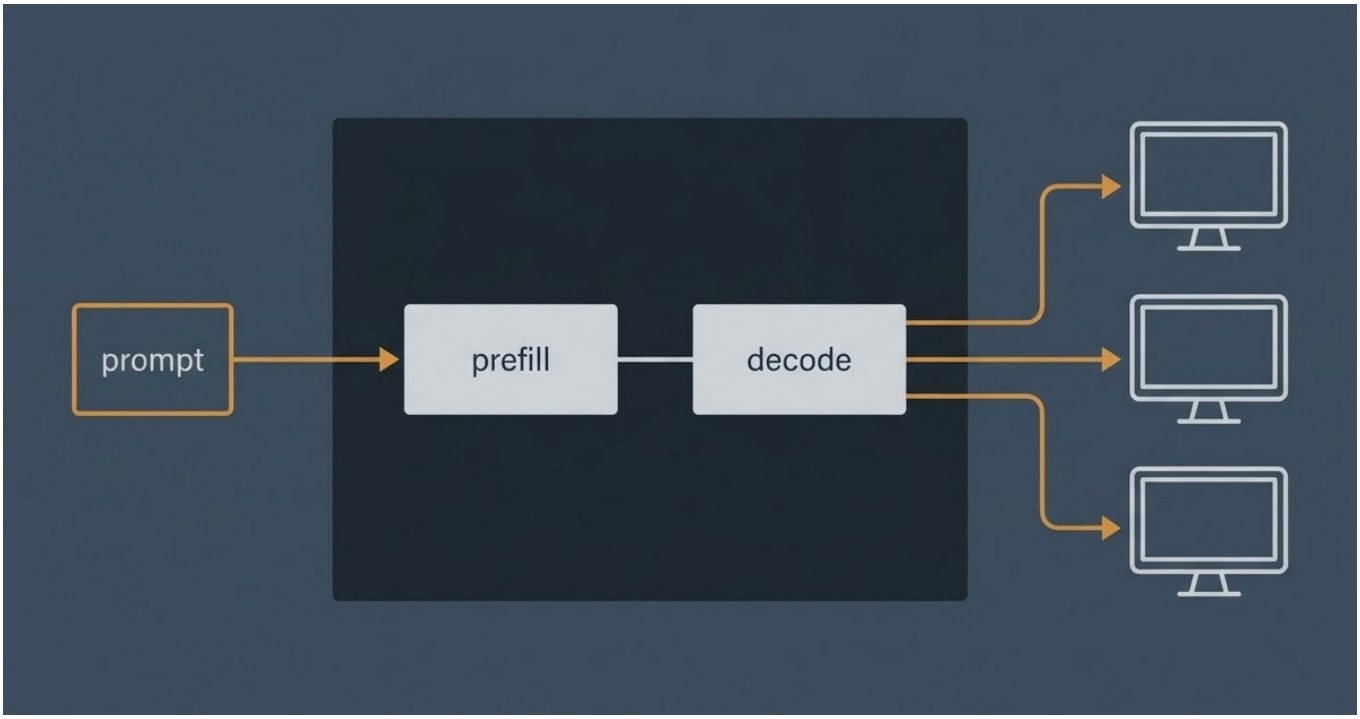
This is a product trend because once a UX capability becomes the default expectation across categories, it stops being a differentiator and becomes an infrastructure requirement. Benchmarks get faster every quarter and almost none of them change how products are built. So why do I think this one will?

Because the shift follows the streaming-video trajectory. Once generation and delivery ran at playback speed, streaming stopped being a feature and became an architectural constraint. It reshaped CDNs, buffer sizing, encoding pipelines and the capacity economics of the industry. Nobody asks anymore whether a video site "supports streaming". It is the water.

The same expectation cascade is visible across AI product categories:

- **Chat and assistants** are already streaming, and in my experience users increasingly read a slow non-streaming interface as broken.
- **Code assistants** compete on how fast completions appear while you type. The product category is defined by the streaming experience.
- **Agents** running tool loops, I would bet, need each step back fast enough that the user keeps watching rather than switching tabs.
- **Generative media** is the newest entrant: faster-than-real-time video and playable generative worlds turn generation into something watched, steered and inhabited live.

When a UX capability becomes the default expectation across every major category, it stops being a differentiator and becomes an infrastructure requirement. That is the trend I am claiming, and it is why I would not file the Mercury and fal announcements under "benchmarks".



Prefill and decode scale differently; the decode phase is where concurrent streaming sessions accumulate cost.

What it breaks in capacity planning

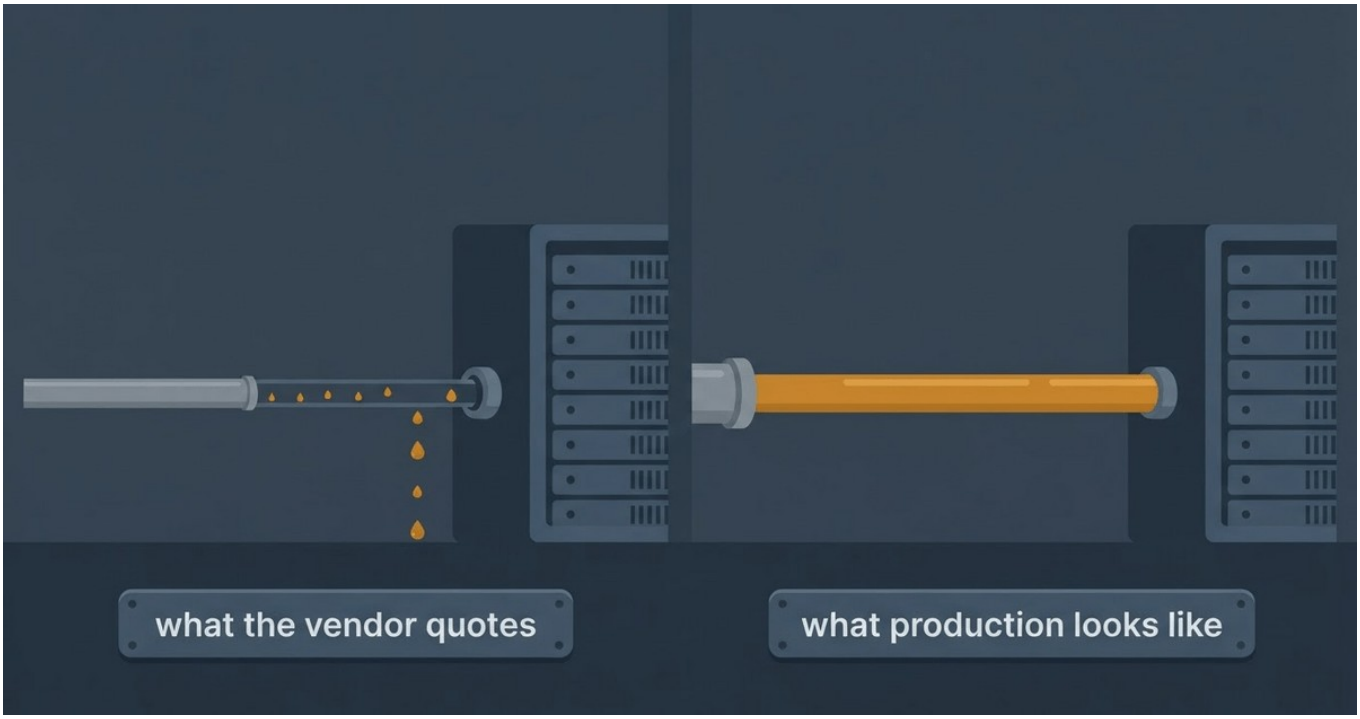
The streaming regime changes the arithmetic underneath AI infrastructure in three places: prefill and decode scale differently, hardware becomes output-token-bound, and the gap between best case and median output throughput separates a vendor claim from production reality. A capacity plan built for batch request rates misses all three.

Prefill and decode scale differently. Every request has a prefill phase (processing the prompt, compute-bound, parallelizable across the input) and a decode phase (producing output, memory-bandwidth-bound, sequential in classic architectures). Batch sizing treats these as one number. Streaming makes decode the dominant cost, because the user is connected for the entire decode. A vendor quote that prices in tokens processed, without separating prefill from decode, cannot tell you what a concurrent streaming session actually costs you.

Output-token-bound hardware. When output length dominates, the sizing question shifts from "how big is the model" to "how many output tokens per second per GPU, at what batch concurrency". Diffusion LMs change this arithmetic again: they generate tokens in parallel, which is why their throughput ceiling differs, and why their best-case and production numbers diverge so sharply.

The gap between best case and median. Here is the conflict every buyer should internalize. Mercury 2.5's 1,107 tokens/sec is a first-party claim about what the architecture achieves on standard GPUs, conditions not specified. OpenRouter's own live panel for that exact model, over a three-day production window, shows a median of about 170 tokens/sec, a P90 near 520 and a P99 near 767. Neither number is wrong. One describes the model; the other describes your production reality, with mixed request shapes, concurrent load and network in the path. Size the hardware on the second number, negotiate against the first.

Vendor quotes today mostly still size for batch or request rate. If you sign a capacity commitment without asking how it behaves under sustained concurrent streaming load, you will over-provision for the wrong dimension, or discover the tail latency in production.



Vendor quotes price a trickle of requests; production streaming holds wide open connections for whole sessions.

What to ask before your next commitment

Before signing a capacity commitment, ask for measured production numbers under sustained streaming load rather than headline figures. The six questions below cover time to first token under load, sustained output-token throughput per GPU at real concurrency, the split of prefill and decode costs, session limits per node, per-stream observability, and the provider's own production panel. Carry this list into your next vendor conversation or internal review:

1. **What is the time to first token at P50 and P90, under load, not in isolation?**
2. **What is sustained output-token throughput per GPU at the concurrency you are actually selling me, and what measured production distribution sits behind that number?**
3. **How do prefill and decode costs appear separately in the pricing and in the capacity model?**
4. **What is the maximum number of concurrent streaming sessions per node, and what degrades first when that limit is approached: inter-token latency, TTFT, or admission?**
5. **Does the observability layer expose per-stream token inter-arrival time, or only request-level latency histograms?**
6. **What did the provider's own live production panel show last week, and can they reconcile it with the headline figure?** (OpenRouter publishes these panels per model. It is the single most honest artifact in this space.)

If a vendor cannot answer 2, 4 and 6 with measured numbers, the quote is sized for a regime the product is leaving.

The honest caveats

These announcements have four honest caveats: the 1,107 tokens/sec figure is first-party and best-case, H3 Max clips hold coherence for roughly 15 seconds, Genie 3 worlds last a few minutes behind a research preview, and real-time here is not hard real-time. I want to be precise about what these developments do not establish:

- The 1,107 tokens/sec figure is first-party and best-case. No independent benchmark of Mercury 2.5 was available when I wrote this; the vendor's quality comparison to frontier models is also self-asserted.
- fal's H3 Max is genuinely faster than playback, but its clips hold coherence for roughly 15 seconds; longer content is stitched, and character consistency drifts.
- Genie 3 runs at 24 FPS 720p "for a few minutes" behind a restricted research preview. Real-time playable worlds exist; durable ones do not yet.
- "Real-time" here means fast enough that generation finishes within the reader's or viewer's attention window. It is not hard real-time in the systems sense, and no one in this chain offers latency guarantees.

The trend is real but narrower than the headlines. Two of the three signals landed within the same week at the end of August 2026, which is why the shift feels sudden. The capability underneath it has been building for over a year.

FAQ

What is time to first token (TTFT)?

Time to first token is how long a user waits between submitting a request and seeing the first piece of generated output. In streaming AI products it is the metric users judge first, because it defines whether the interface feels alive or frozen, and it can be poor even when average request latency looks fine.

Why does token throughput stop mattering beyond a certain speed?

Once output flows faster than a person can read or watch, additional speed buys nothing for that individual user. The value moves to capacity: a GPU producing tokens faster than the reading threshold can serve more concurrent streams on the same hardware. Speed shifts from a user experience feature to a session economics feature.

How is streaming inference different from batch inference?

Batch inference returns the complete output after the model finishes; the user waits for the whole result. Streaming inference delivers tokens or frames as they are generated, so the user consumes the output while it is being produced. The two regimes need different metrics, different SLAs and different capacity models: request latency for batch, TTFT, inter-token latency and concurrent streams for streaming.

What are prefill and decode in LLM inference?

Prefill is the phase where the model processes the input prompt; it is compute-bound and parallelizes across the input. Decode is the phase where the model produces output tokens; it is largely sequential and memory-bandwidth-bound. They scale differently, so a capacity plan that prices them as one number cannot predict what a concurrent streaming workload costs.

What should I ask a vendor about streaming capacity?

Ask for sustained output-token throughput per GPU at realistic concurrency, the measured production distribution behind that number, the maximum concurrent streaming sessions per node, TTFT at P50 and P90 under load, and per-stream token timing in the observability layer. If the quote is built on request rate or tokens processed, it is sized for the wrong regime.

Book a 15-min intro call

Book a free 15-min intro call and we will look at your platform and which service fits. If you are sizing AI infrastructure for a streaming product, or reviewing a vendor quote that still prices in requests per second, that is exactly the kind of question I work on.

Sources

1. OpenRouter, Mercury 2.5 Preview model page, Aug 31 2026. https://openrouter.ai/inception/mercury-2.5-preview and announcement on X
2. Inception Labs, "Introducing Mercury 2". https://www.inceptionlabs.ai/blog/introducing-mercury-2
3. SaaSCity, "AI Video Faster Than Real Time: Inside MiniMax H3 Max", Aug 2026. https://saascity.io/blog/ai-video-faster-than-real-time-minimax-h3-max-live
4. fal, H3 Max model page. https://fal.ai/minimax-h3-max and announcement on X
5. Pieter Levels, "Infinite Slop". https://levels.io/infinite-slop
6. Latent Space, "Fal's H3 Max Live breaks the infinite videogen barrier". https://www.latent.space/p/ainews-fals-h3-max-live-breaks-the
7. Google DeepMind, "Genie 3: A new frontier for world models", Aug 5 2025. https://deepmind.google/blog/genie-3-a-new-frontier-for-world-models/

No affiliation: the author has no relationship with OpenRouter, Inception Labs, fal, MiniMax, Pieter Levels or Google DeepMind. Nothing above is an endorsement.

Update 2026-09-29. This post is now part of AI Infrastructure and SRE: Running Models in Production, the field guide to running models in production.

Related posts

- AI Infrastructure and SRE: Running Models in Production

PDF snapshot of the public article. The online version is authoritative; links remain clickable.