Where Does a Robot Think – On-Device vs Datacenter Inference

机器人“思考”在哪里——设备端推理与数据中心推理

The Embodiment Problem, Planning vs Action Layers, Glass-To-Glass Budgets, Wafers & DRAM Constraints, One B300 vs 56 Thors TCO, Factories To Caves

具身问题、规划层与行动层、从玻璃到玻璃的预算、晶圆与 DRAM 限制、一块 B300 与 56 台 Thor 的总拥有成本对比、从工厂到洞穴

VAN CHIAM, ZANE FONG, BRYAN SHAN, AND 4 OTHERS

VAN CHIAM、ZANE FONG、BRYAN SHAN 及其他 4 人

SEP 10, 2026 · PAID · 付费

49 1 Share 分享

For most of its short history, AI lived behind a screen. That’s starting to change.

在其短暂的历史中，AI 大多生活在屏幕之后。这种情况正在开始改变。

First came chatbots, good for answering a question or drafting an email. Then agentic AI, models that don’t just respond but do real work on a computer: navigating software, calling tools, finishing multi-step tasks on their own. Now the frontier is physical AI, intelligence that reaches past the screen to perceive the world and act on it. The biggest piece is robots, and it is still early: the hardware, the models, and the economics are all being worked out right now.

先是聊天机器人，擅长回答问题或起草邮件。然后是代理式 AI，这些模型不仅能回应，还能在计算机上实际执行任务：操作软件、调用工具、自主完成多步骤任务。如今，前沿是物理 AI，即超越屏幕去感知世界并对其采取行动的智能。其中最大的部分是机器人，而这仍处于早期阶段：硬件、模型和经济性都正在摸索之中。

The Embodiment Problem 具身问题

With LLMs, the hardware bends to the model. For hyperscalers and AI labs, there’s no hard latency requirement and a very high spending ceiling, so the process is simple:

pour in as much data and compute as possible at training, then figure out how to serve the model that comes out. The model comes first; the hardware serves it. Robotics inverts this. Tighter constraints flip the order of design: you first work out what it takes to serve a model on the robot, then build a model that fits. Two constraints drive this.

对于 LLM 而言，硬件屈从于模型。对于超大规模云服务商和 AI 实验室来说，没有严格的延迟要求，且支出上限很高，因此过程很简单：在训练时尽可能多地投入数据和算力，然后想办法服务训练出的模型。模型优先；硬件为其服务。机器人技术则颠覆了这一点。更严格的约束颠倒了设计顺序：你首先要弄清楚在机器人上服务模型需要什么，然后构建一个与之匹配的模型。有两个约束驱动着这一过程。

The first is time. A robot runs real-time control loops and can't miss a deadline. An LLM can be slow without affecting the final output. If a robot is slow, the world changes around it, and then the action is obsolete by the time it needs to enact it.

首先是时间。机器人运行实时控制回路，不能错过任何截止时间。而 LLM 可以慢一些，不影响最终输出。如果机器人反应迟缓，周围的世界会随之变化，等到它需要执行动作时，该动作可能已经过时了。

The second is cost. When you use an LLM, it lives behind a screen provided by the user. In robotics, the manufacturer has to build both the compute and the robot itself and pay for it upfront, on every unit, and the capital expenditure can be immense. At scale, this upfront cost can be billions and even in the trillions if we do eventually get to a billion robots.

其次是成本。当你使用 LLM 时，它存在于用户提供的屏幕背后。而在机器人领域，制造商必须同时构建计算设备和机器人本身，并在每一台设备上预先支付费用，资本支出可能非常庞大。在规模化生产时，这种前期成本可能达到数十亿，如果我们最终达到十亿台机器人，甚至可能高达数万亿。

Because of these two constraints, in robotics the on-robot hardware is fixed, and the model is designed to fit it. Even at hyperscaler funding levels, most of the capital will have to go into the robots and factories themselves, not into compute. Model capability is therefore capped by what can run on affordable, real-time hardware on the robot, and intelligence becomes something you ration against latency and unit

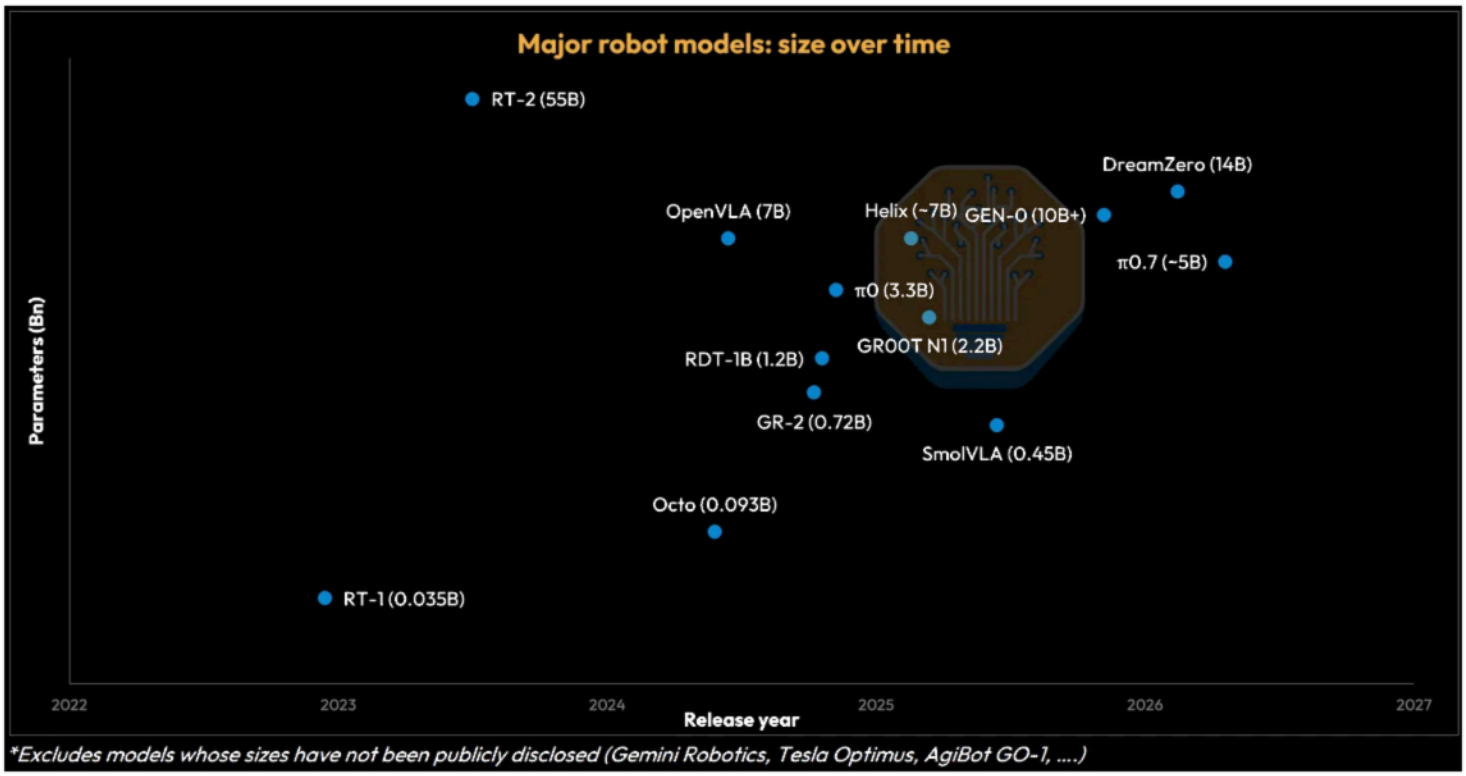
economics.

由于这两个限制，在机器人领域，机器人上的硬件是固定的，模型需要设计得与之匹配。即使有超大规模的资金支持，大部分资本也必须投入到机器人和工厂本身，而非计算设备。因此，模型能力受限于能在机器人上以可负担的实时硬件运行的范围，智能变成了一种需要根据延迟和单位经济性来权衡的资源。

Robot Models Are Still Small...
机器人模型仍然很小……

This is why frontier robot models remain far smaller than frontier LLMs, which have already pushed into the trillions of parameters. Today most frontier robot models sit between roughly 5 and 14 billion parameters: Physical Intelligence's π0.7 at 5 billion, and NVIDIA's DreamZero at 14 billion.

这就是为什么前沿机器人模型仍然远小于前沿 LLM，后者已经进入了万亿参数级别。如今，大多数前沿机器人模型的参数量大约在 50 亿到 140 亿之间：Physical Intelligence 的π0.7 为 50 亿参数，而 NVIDIA 的 DreamZero 为 140 亿参数。



Source: SemiAnalysis 来源：SemiAnalysis

In LLMs, scaling laws have held, scaling has consistently broadened the range of problems a single model can solve. Early results show robotics might work the same

way. Data has been a constraint but that is now scaling with World Models and Embodied human data collection. Embodied AI faces limits language models never did, like scarce data and the cost of real-world interaction, so whether scaling pays off the same way is genuinely an open question.

在 LLM 中，扩展定律一直成立，扩展持续拓宽了单个模型能解决的问题范围。早期结果表明，机器人技术可能遵循同样的规律。数据一直是一个制约因素，但现在随着世界模型和具身人类数据收集的发展，这一制约正在得到缓解。具身 AI 面临着语言模型从未遇到过的限制，比如数据稀缺和现实世界交互的成本，因此扩展是否能以同样的方式带来回报，确实是一个悬而未决的问题。

If robot models do continue to scale, the question becomes where should all that compute run, given the latency, cost and model limits mentioned above. The field is genuinely split, with different companies making diverging bets. Figure runs its in-house Helix model entirely onboard the robot. Physical Intelligence's π0.7 model goes the other way and runs its policy off-robot on a single H100. NVIDIA's DreamZero, a 14-billion-parameter "world action model" built on a video-diffusion backbone, needs two GB200 GPUs off-robot just to run in real time.

如果机器人模型确实会继续扩大规模，那么问题就变成了：考虑到上述提到的延迟、成本和模型限制，所有这些计算应该在哪里运行。这一领域确实存在分歧，不同公司做出了截然不同的押注。Figure 公司将其自研的 Helix 模型完全部署在机器人本体上运行。Physical Intelligence 的π0.7 模型则走另一条路线，将其策略在机器人之外的单个 H100 上运行。NVIDIA 的 DreamZero——一个基于视频扩散架构构建的 140 亿参数"世界动作模型"——需要机器人之外的两块 GB200 GPU 才能实时运行。

Our view is that a cascade is inevitable. Some robots will run cognition on-board, while others will offload it to GPUs sitting in a datacenter. The biggest robot model builders today, like NVIDIA and Google DeepMind, will lean toward this cloud approach, running the heavy cognition on cloud GPUs, with only a small edge model on the robot for low-level control and safety.

我们的观点是，分层架构不可避免。一些机器人将在本体上运行认知功能，而另一些则会将其卸载到数据中心内的 GPU 上。如今最大的机器人模型构建者，如 NVIDIA 和 Google DeepMind，将倾向于这种云端方法，在云端 GPU 上运行重型认知任务，仅在机器人上保留一个小型边缘模型用于低级控制和安全性。

Smaller builders, such as Dyna and Sunday, may initially favor models that fit on a Jetson or consumer GPUs because their early deployments operate within narrow, controllable environments where onboard inference is sufficient. As deployments scale, however, they will converge on whichever mix of onboard and off-robot compute delivers the best performance and economics for their business model.

像 Dyna 和 Sunday 这样规模较小的制造商，初期可能会倾向于选择能在 Jetson 或消费级 GPU 上运行的模型，因为他们的早期部署环境范围狭窄且可控，机载推理已足够应对。然而，随着部署规模扩大，他们将根据自身商业模式，选择机载与机器人外部计算的最佳组合，以实现最优性能和经济效益。

Moving the GPU off the robot brings real benefits, but also real challenges. Chief among them is latency, and it doesn't work everywhere. In environments with poor connectivity, cloud deployment isn't an option at all. Despite these difficulties, off-robot compute remains beneficial, and in some cases necessary, as the following sections show.

将 GPU 移出机器人确实带来了实际好处，但也伴随着真正的挑战。其中最主要的是延迟问题，而且这种方法并非在所有环境下都适用。在连接性差的环境中，云端部署根本不可行。尽管存在这些困难，机器人外部计算仍然是有益的，在某些情况下甚至是必要的，如下文所述。

Models in Robots 机器人中的模型

Before we get into why the robot model should run in an off-robot datacenter, here's a quick primer on how robot models work, and which ones we're talking about.

在我们深入探讨为什么机器人模型应在机器人外的数据中心运行之前，先快速了解一下机器人模型的工作原理，以及我们具体指的是哪些模型。

Robot control is a broad space. Hard-coded control, where engineers write out the rules and math for every movement, has run factories for decades on embedded compute, and small task-specific policies fit onboard just fine. That's not our subject. We're talking about data-driven generalist robotics, robots run by large foundation

models, because that's where the frontier is.

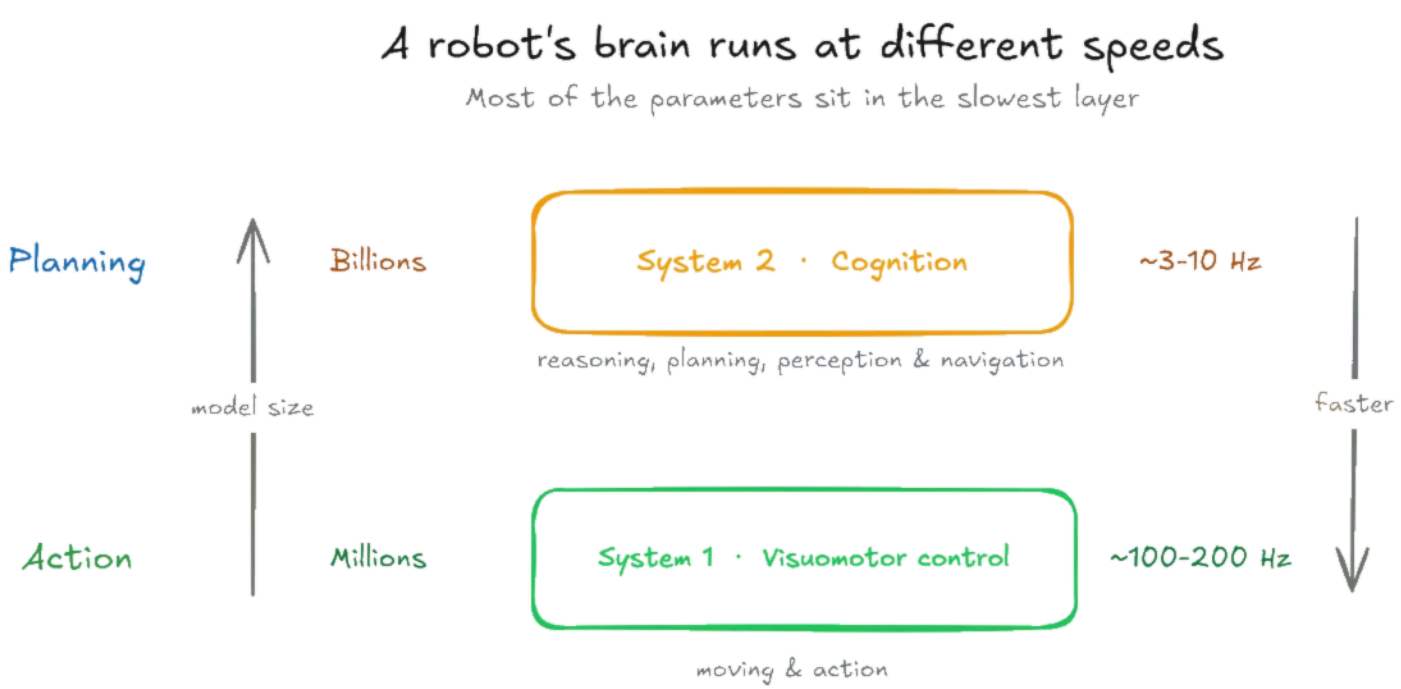
机器人控制是一个广阔的领域。硬编码控制——即工程师为每一个动作编写规则和数学公式——已在嵌入式计算上运行工厂数十年，而小型任务专用策略也能很好地适配机载设备。这不是我们讨论的主题。我们谈论的是数据驱动的通用机器人，即由大型基础模型驱动的机器人，因为那才是前沿所在。

Even within data-driven generalist robot models, architectures vary. Some are monolithic: one model that goes straight from what the camera sees to how the motors move. Many are hierarchical: a big, slow model decides what to do, and a small, fast one turns that into movement, hundreds of times a second. Our argument for off-robot datacenter compute is about these hierarchical models, where the brain separates into two or more layers, most often a planning layer and an action layer.

即便在数据驱动的通用机器人模型中，架构也各不相同。有些是单体式的：一个模型直接从摄像头所见映射到电机如何运动。许多是层级式的：一个大型、缓慢的模型决定要做什么，而一个小型、快速的模型将其转化为每秒数百次的运动。我们关于机器人外数据中心计算的论点，正是针对这些层级式模型，其中"大脑"分为两层或多层，最常见的是规划层和动作层。

The planning layer is the slow, high-level one: it perceives the scene, reasons about space, and plans the task. The action layer is the fast, low-level one: it turns those plans into motor commands. The planning layer, which is also the heavy part of the model, is the more forgiving: it updates only a few times a second and runs asynchronously from the control loop, so the action layer keeps acting on the most recent plan while the next one is still in flight. That decoupling lets planning absorb the delay, and even the jitter, of a wireless round trip, while the action layer runs at hundreds of Hz and can tolerate none of it. So planning is the natural candidate to move into the datacenter, while the fast control loop stays local.

规划层是缓慢而高层的部分：它感知场景、推理空间并规划任务。动作层则是快速而低层的部分：它将那些计划转化为运动指令。规划层同时也是模型中较重的部分，它更为宽容：每秒仅更新几次，并与控制回路异步运行，因此动作层会在下一个计划仍在生成时，持续依据最新的计划执行动作。这种解耦让规划层能够吸收无线往返带来的延迟甚至抖动，而动作层则以数百赫兹的频率运行，无法容忍任何延迟。因此，规划层是移入数据中心的自然候选，而快速控制回路则留在本地。



Source: SemiAnalysis 来源：SemiAnalysis

The amount of planning scales with how general the robot's job is. A generalist built to handle open-ended tasks needs a large planning model doing genuine reasoning, and those models are heavy. That reasoning is exactly what gets pushed to the datacenter. A specialist is the opposite case: a narrow policy doing one repetitive job carries little high-level planning, so the whole model is small enough to stay on the robot.

规划的量取决于机器人任务的通用程度。一个为处理开放式任务而构建的通用型机器人需要大型规划模型进行真正的推理，而这些模型很重。这种推理正是被推送到数据中心的部分。专用型机器人则相反：执行单一重复性工作的窄策略携带的高层规划很少，因此整个模型足够小，可以留在机器人上。

Why DC GPUs are better suited...

为何数据中心 GPU 更合适……

Specialist robots are for the most part solved: factories have run on them for many years now using hard-coded automation, but not modern AI. But a specialist is rigid, one task at one station, idle when the bottleneck moves. Generalists are the frontier because one flexible robot can cover many jobs, redeploy to wherever is bottlenecked, a changing product mix, and the long tail of unstructured work. This aligns with some of the many lessons of the Toyota Production System, where we learned that multi-

skilled workers can reduce waste and improve economic efficiency. As a result, the TAM for generalist robotics is huge, not just in manufacturing and warehousing but also in homes and other places.

专业机器人大多已得到解决：多年来，工厂一直依靠硬编码自动化运行它们，而非现代 AI。但专业机器人是僵化的，一个工位执行一项任务，当瓶颈转移时便闲置。通用型机器人是前沿方向，因为一个灵活的机器人可以覆盖多种工作，重新部署到任何出现瓶颈的地方，适应不断变化的产品组合，以及处理长尾的非结构化工作。这与丰田生产系统中的许多经验相契合，我们从中了解到，多技能工人可以减少浪费并提高经济效率。因此，通用型机器人技术的 TAM（总可寻址市场）巨大，不仅限于制造业和仓储业，还包括家庭及其他场所。

The more generality you demand, the more you lean toward FLOP-hungry foundation models and datacenter-scale compute. Not every robot needs maximal generality, and many tasks are served well by smaller models, but the frontier of flexible, general-purpose robots is where datacenter-scale compute becomes necessary.

你对通用性的要求越高，就越倾向于依赖高算力消耗的基础模型和数据中心级计算。并非每个机器人都需要最大程度的通用性，许多任务由较小模型即可良好完成，但灵活、通用型机器人的前沿领域正是数据中心级计算变得不可或缺之处。

LLMs vs Robot Models Token Flow
LLM 与机器人模型的 Token 流

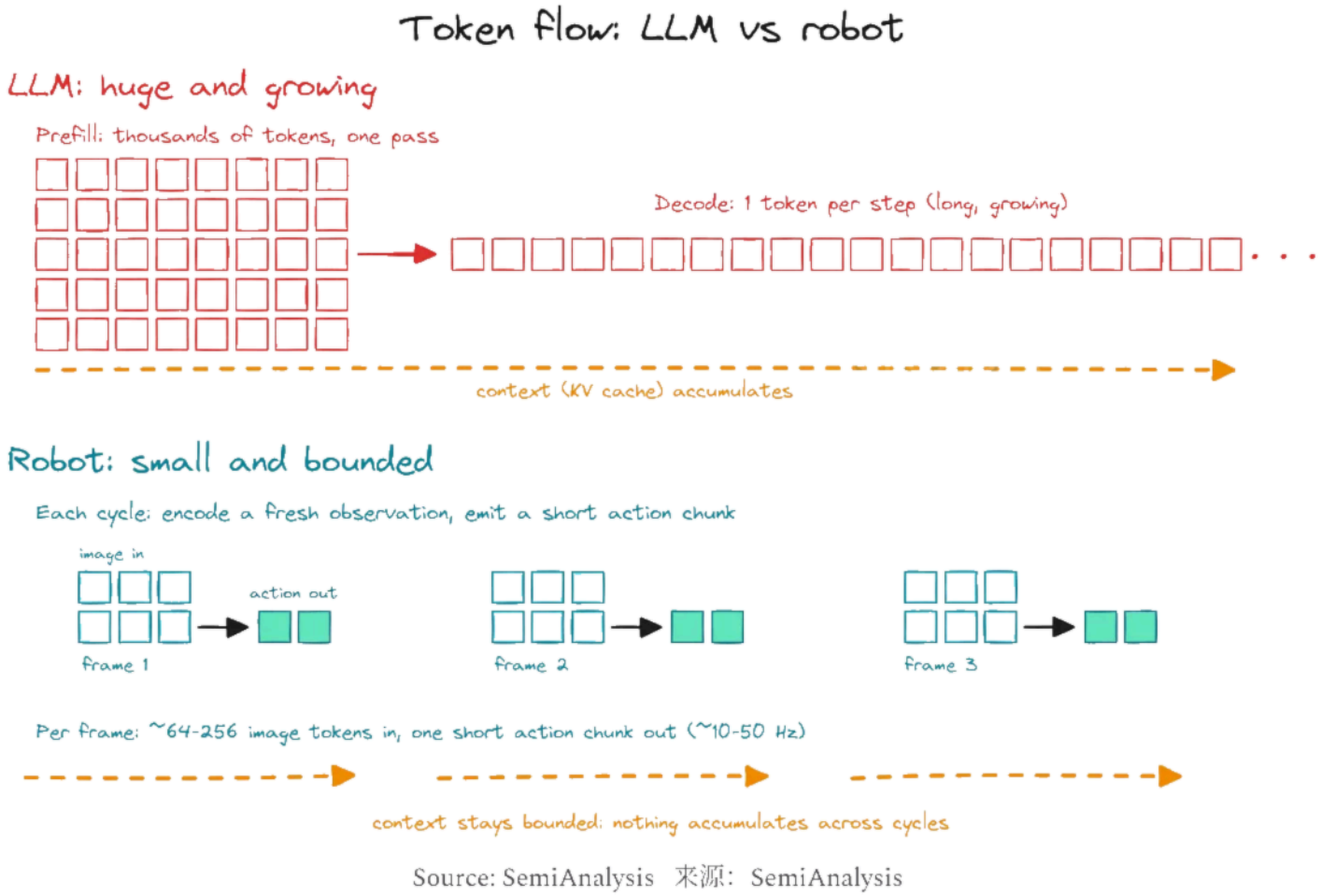
An LLM ingests and emits huge, bursty token counts: a long prompt to prefill, then a long generation, with a KV cache that grows the whole time. That is the memory wall, the capacity to hold that cache and the bandwidth to stream the weights and cache on every token, and it makes LLM serving bandwidth-bound.

LLM 摄取并产生海量、突发性的 token 数量：先是长提示词进行预填充，然后是一次长生成，期间 KV 缓存不断增长。这就是内存墙——容纳该缓存的容量以及每个 token 上流式传输权重和缓存的带宽，这使得 LLM 服务受带宽限制。

A robot model is the opposite, a smaller steady stream, more metronome-like rather than a burst. It never stops, because a robot is not answering one question and halting; it is locked in a closed loop with a moving world, taking in a fresh frame and emitting

a short action chunk a few times a second. Each frame is compressed in dedicated hardware (the ISP and video-codec ASICs) into a fixed, compact token budget of tens to a few hundred tokens. So robot models are not as memory capacity and bandwidth bound in the same way that LLMs are.

机器人模型则相反，是更小的稳定流，更像节拍器而非突发。它从不停止，因为机器人不是回答一个问题就结束；它锁定在与动态世界的闭环中，每秒数次摄取新帧并输出短动作片段。每帧在专用硬件（ISP 和视频编解码 ASIC）中被压缩为固定、紧凑的 token 预算，从几十到几百个 token 不等。因此，机器人模型不像 LLM 那样受内存容量和带宽限制。



Source: SemiAnalysis 来源：SemiAnalysis

Frontier Robot Models are FLOPs hungry
前沿机器人模型渴求 FLOPs

What binds a frontier robot model sometimes memory, but often it is the FLOPs to run it.

束缚前沿机器人模型的有时是内存，但往往是运行所需的 FLOPs。

These are FLOP-hungry workloads: a VLA (Vision Language Action model) is a multi-billion-parameter model, orders of magnitude larger than the specialist policies before it, and its vision and language stages, which are the bulk of the work, are compute-bound on any capable GPU. A WAM (World Action Model) is heavier still. Instead of naming an action, it generates video of the predicted future and derives the action from it, through iterative denoising, meaning one action could take multiple full forward passes through a model.

这些是极其消耗算力的任务：VLA（视觉-语言-动作模型）是一个拥有数十亿参数的模型，比之前的专用策略模型大数个数量级，而其视觉和语言阶段占据了大部分计算量，在任何高性能 GPU 上都是计算密集型的。WAM（世界动作模型）则更为沉重。它不直接命名动作，而是通过迭代去噪生成预测未来的视频，并从中推导出动作，这意味着一个动作可能需要多次完整的模型前向传播。

Edge silicon cannot keep up. Nvidia's latest Jetson Thor, the best-in-class robot brain, has only about a tenth of a single B200's compute, and roughly a fourteenth of a B300's. DreamZero (a 14B WAM) needed two GB200 GPUs just to reach ~7Hz, which is on the order of twenty times the compute a single Thor can muster. Run the same model on a Thor and it would fall to a fraction of 1Hz, far too slow for real-time control.

边缘芯片无法跟上需求。Nvidia 最新的 Jetson Thor，作为同类最佳的机器人"大脑"，其算力仅约为单个 B200 的十分之一，大约是 B300 的十四分之一。DreamZero（一个 14B 参数的 WAM）需要两块 GB200 GPU 才能达到约 7Hz 的频率，这大约是单个 Thor 所能提供算力的二十倍。若在 Thor 上运行同一模型，其频率将降至 1Hz 的几分之一，远不足以支持实时控制。

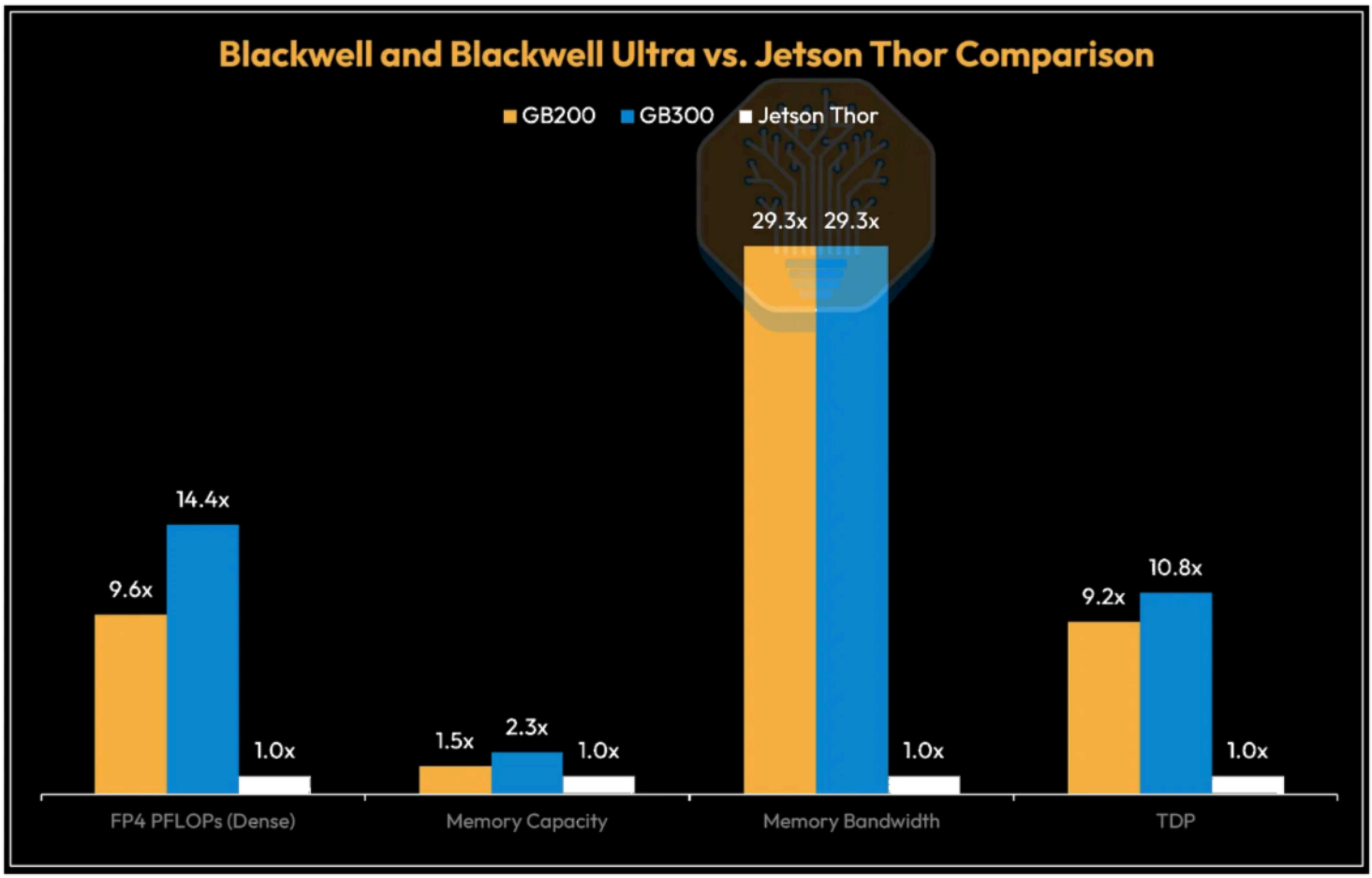
Datacenter vs Edge: NVIDIA Accelerator Specs

	Blackwell		Jetson Series
Accelerator	B200/ GB200	B300/ GB300 (Ultra)	Jetson AGX Thor (T5000)
GPU TDP (W)	1,000/1,200	1,200/1,400	40-130
Foundry Node	4NP		4NP
FP4 PFLOPs - Dense (per Package)	9/10	14/15	
FP8 PFLOPs - Dense (per Package)*	4.5/5	4.5/5	
FP16 PFLOPs - Dense (per Package)*	2.3/2.5	2.3/2.5	
HBM Capacity	192GB HBM3E	288GB HBM3E	
HBM Stacks	8		
Memory Bandwidth	8TB/s		

*NVIDIA publishes only FP4 (2,070 TFLOPS sparse) for Jetson Thor; FP8 and FP16 dense are



Source: SemiAnalysis 来源：SemiAnalysis



Source: SemiAnalysis 来源：SemiAnalysis

Shipping the Robot Model Off Device
将机器人模型发送到设备端之外

If a Jetson Thor can't keep up, and we off-load the heavy, compute-intensive part of the workload to datacenters, it means optimizing local compute for safety loops and high-frequency control loops, and doubling-down on network interfacing.

如果 Jetson Thor 无法跟上，而我们将工作负载中计算密集的重型部分卸载到数据中心，这意味着要优化本地计算以用于安全回路和高频控制回路，并加倍投入网络接口。

The primary consideration is the glass-to-glass latency budget we need to hit, and hit reliably. The network will produce issues with bandwidth and stability, and we will feel this the most when transmitting image data.

首要考虑的是我们需要达到并可靠满足的端到端延迟预算。网络会带来带宽和稳定性问题，而在传输图像数据时，我们会最深刻地感受到这一点。

Transmitting image data quickly and reliably is more than just a dedicated compression ASIC on-device and calling it a day; the entire upstream image processing pipeline must be optimized. Noise is the enemy of compression, the less spatiotemporal structure an image has, the harder it is to compress. So a good local compute system optimized for networking will prioritize the components that keep noise out of the image in the first place: optics, low-light sensor performance, sensor SnR, thermal management, and the image signal processor (ISP).

快速可靠地传输图像数据，不仅仅是在设备端配备一个专用的压缩 ASIC 就万事大吉；整个上游图像处理流水线都必须进行优化。噪声是压缩的敌人，图像的空间时间结构越少，压缩就越困难。因此，一个为网络优化而设计的良好本地计算系统，会优先考虑那些从一开始就防止噪声进入图像的组件：光学元件、低光传感器性能、传感器信噪比、热管理以及图像信号处理器（ISP）。

ISPs turn messy analog signals into digital ones, but can ruin compression efficiency when not optimized for compression directly. And when we consider the imaging pipeline as a whole, we are optimizing for the model, not for human viewing. This is similar to how WAMs do not require the extra denoising steps that typical video diffusion models use; they solve a different task. The same principle applies here: we care about good model outputs and a low bitrate, not necessarily beautiful images (see below).

ISP 将混乱的模拟信号转换为数字信号，但若未针对压缩直接优化，可能会损害压缩效率。而当我们从整体上考虑成像流水线时，我们是在为模型优化，而非为人类观看优化。这与 WAMs 不需要典型视频扩散模型所用的额外去噪步骤类似；它们解决的是不同任务。同样的原则在此适用：我们关心的是良好的模型输出和低比特率，而不一定是美观的图像（见下文）。



Left image is more pleasing to the eye, but the right image has a higher SnR. Source: NASA

左图更赏心悦目，但右图具有更高的信噪比。来源：NASA

Network conditions introduce a lot of variability, so local compute must also incorporate dynamic, real-time adaptation of both the ISP and the compression bitrate, similar to WebRTC, to gracefully handle constantly fluctuating network conditions and do so without adversely affecting model performance.

网络条件带来诸多变数，因此本地计算还必须结合 ISP 和压缩比特率的动态实时自适应调整，类似于 WebRTC，以优雅地应对持续波动的网络状况，且不影响模型性能。

A pipeline that produces a clean, adaptable stream also needs hardware explicitly tuned for offloading – getting the bits off-device and on-schedule. Historically, mobile chipsets prioritized downlink speeds, but robotics requires massive uplink capacity to stream out high-fidelity telemetry. The download can be relatively small. This means multiple transmit antennas with proper placement, since the environment and robot itself can block signals. It also means support for a wider array of RF bands to allow multiple channels for higher throughput and less congestion – especially with factory conditions in mind. Optimizing hardware and embedded firmware for fast handovers between access points and between WiFi and 5G is critical since the typical handover

time will temporarily kill the latency budget. Jitter is the biggest enemy.

一个能产生干净、可适应数据流的管道，还需要专门针对卸载进行优化的硬件——即将数据位从设备上移出并按计划传输。历史上，移动芯片组优先考虑下行链路速度，但机器人技术需要巨大的上行链路容量来流式传输高保真遥测数据。下载量可以相对较小。这意味着需要合理布置多根发射天线，因为环境和机器人本身都可能阻挡信号。这也意味着要支持更广泛的射频频段，以便利用多个信道实现更高吞吐量和更少拥塞——尤其是在考虑工厂环境的情况下。优化硬件和嵌入式固件，以实现接入点之间以及 WiFi 与 5G 之间的快速切换至关重要，因为典型的切换时间会暂时破坏延迟预算。抖动是最大的敌人。

The end goal is a highly coordinated hybrid architecture that maintains a healthy, but not overly indulgent, amount of FLOPs locally for safety and high-frequency control and can reliably off-load the “foundation” part of a foundation model. It’s not a perfect system: simple control loop fallbacks are still required the moment the network connection diminishes or is completely lost. We require latency watchdogs, heartbeat timeouts, and staleness bounds on remote actions.

最终目标是构建一种高度协调的混合架构，在本地维持合理但不过度冗余的计算量（FLOPs），以保障安全性和高频控制，同时能够可靠地将基础模型中的“基础”部分卸载到远端。这并非一个完美的系统：一旦网络连接减弱或完全中断，仍需依赖简单的控制回路作为后备方案。我们还需要延迟监控、心跳超时机制，以及对远程操作的时效性约束。

Power 电源

Datacenter GPUs have to run off the robot, because the battery can’t feed them. A single Blackwell GPU draws ~1.2-1.4 kW and has to be liquid-cooled, built to sit in a rack. A humanoid, by contrast, stores only about 2 kWh in its battery and draws a few hundred watts in normal motion, a kilowatt or two at peak, everything included, which is why its onboard brain is a 40 to 130 W Jetson Thor. Put a datacenter GPU on the robot and the chip alone would out-draw the whole machine, demand cooling it can’t carry, and flatten the battery in minutes. Offloading sidesteps the power constraint. The robot carries only a 20 to 30 W perception compute + wireless board to reach the

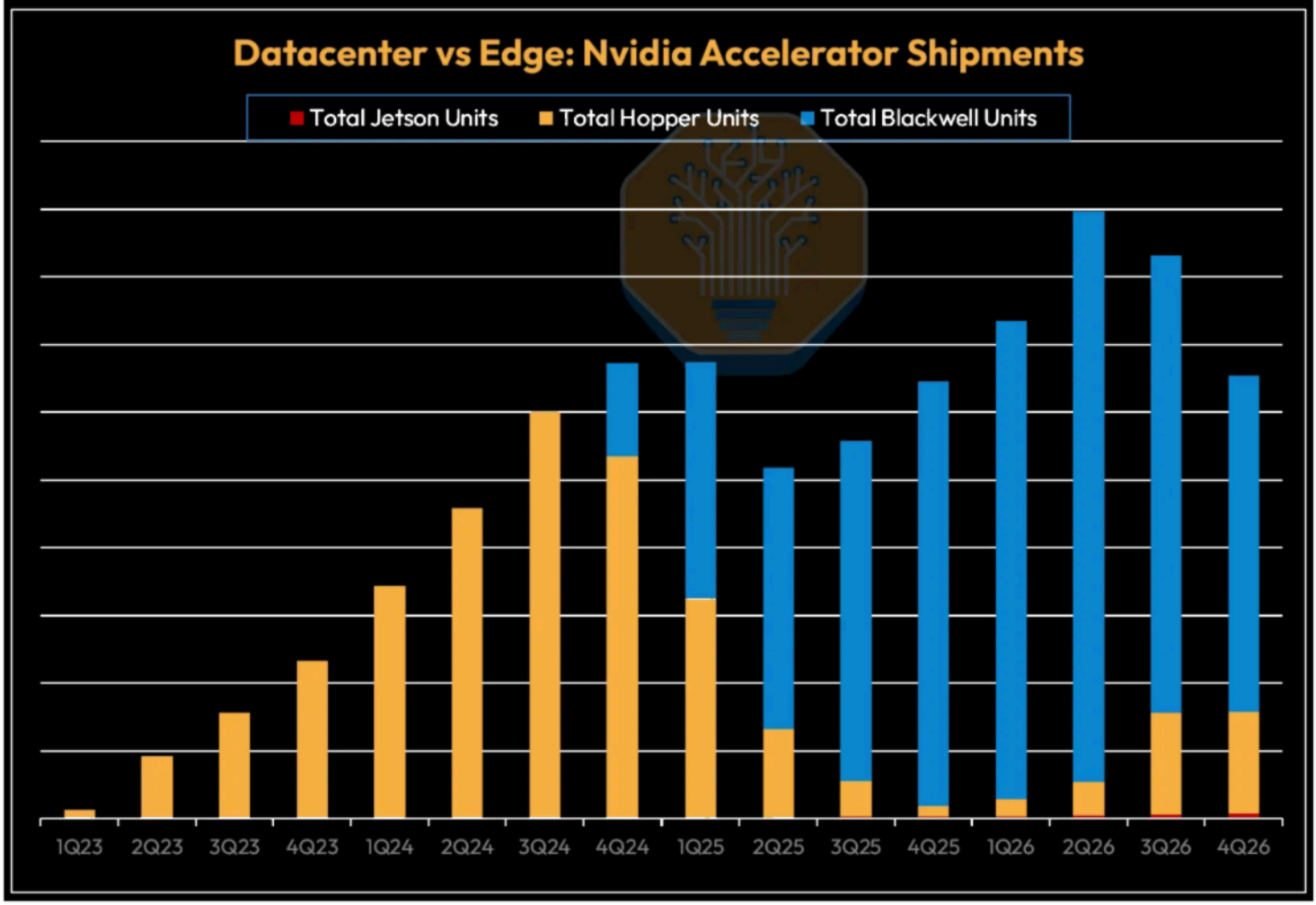
datacenter GPU, comfortably within budget.

数据中心 GPU 无法安装在机器人上，因为电池供不起。一块 Blackwell GPU 功耗高达~1.2-1.4 千瓦，且需液冷，设计上就是放在机架里的。相比之下，人形机器人电池仅储存约 2 千瓦时电量，正常运动时功耗几百瓦，峰值也就一两千瓦（含所有部件），因此其机载大脑是 40 至 130 瓦的 Jetson Thor。若把数据中心 GPU 装到机器人上，仅芯片一项就会超过整机功耗，需要它无法携带的冷却系统，几分钟内就能耗尽电池。卸载计算则绕开了功耗限制。机器人只需携带 20 至 30 瓦的感知计算加无线通信模块来连接数据中心 GPU，完全在预算之内。

The Supply-Chain Wall 供应链之墙

Nvidia's accelerator output is almost entirely datacenter silicon: a long Hopper ramp through 2023 and 2024 handing off to Blackwell, which scales sharply across 2025 and 2026 to dominate the mix. Jetson, the line used in robots, is the thin red sliver along the bottom, barely visible even in 2026. This is because the robotics market is nascent, real volume is years out, and there is next to nothing to build for.

英伟达的加速器产出几乎全是数据中心芯片：2023 至 2024 年 Hopper 长期爬坡，随后交接给 Blackwell，后者在 2025 至 2026 年急剧放量并主导产品结构。用于机器人的 Jetson 系列，是底部那条细红线，即便到 2026 年也几乎看不见。这是因为机器人市场尚处于萌芽期，真正放量还要等好几年，目前几乎没有可为之量产的产品。



Source: SemiAnalysis Accelerator Model, we withheld Rubin shipments on this chart, buy the data

来源：SemiAnalysis 加速器模型，此图表中我们未计入 Rubin 出货量，请自行购买数据

Margins tell a similar story. Demand isn't there yet, but when it arrives, pricing will go up and Jetson margins will inflect. But for now, a Blackwell datacenter GPU earns far higher gross margins than a Jetson module, on the order of mid-to-high 70s percent versus mid-60s, so NVIDIA has every incentive to point its scarce leading-edge wafer capacity at the datacenter and little reason to ramp edge silicon while the robotics market stays small. Robotics chips like Jetson Thor also require a lot more memory per unit of compute which is a major roadblock.

利润率也反映了类似的情况。目前需求尚未到位，但一旦需求到来，价格将上涨，Jetson 的利润率也将随之发生转折。但就目前而言，Blackwell 数据中心 GPU 的毛利率远高于 Jetson 模块，前者约为 75%至 80%区间，后者约为 65%左右，因此英伟达有充分理由将其稀缺的领先制程晶圆产能优先分配给数据中心，而在机器人市场仍处于小规模阶段时，几乎没有动力扩大边缘芯片的产能。像 Jetson Thor 这样的机器人芯片在单位算力上还需要更多的内存，这也是一个重大障碍。

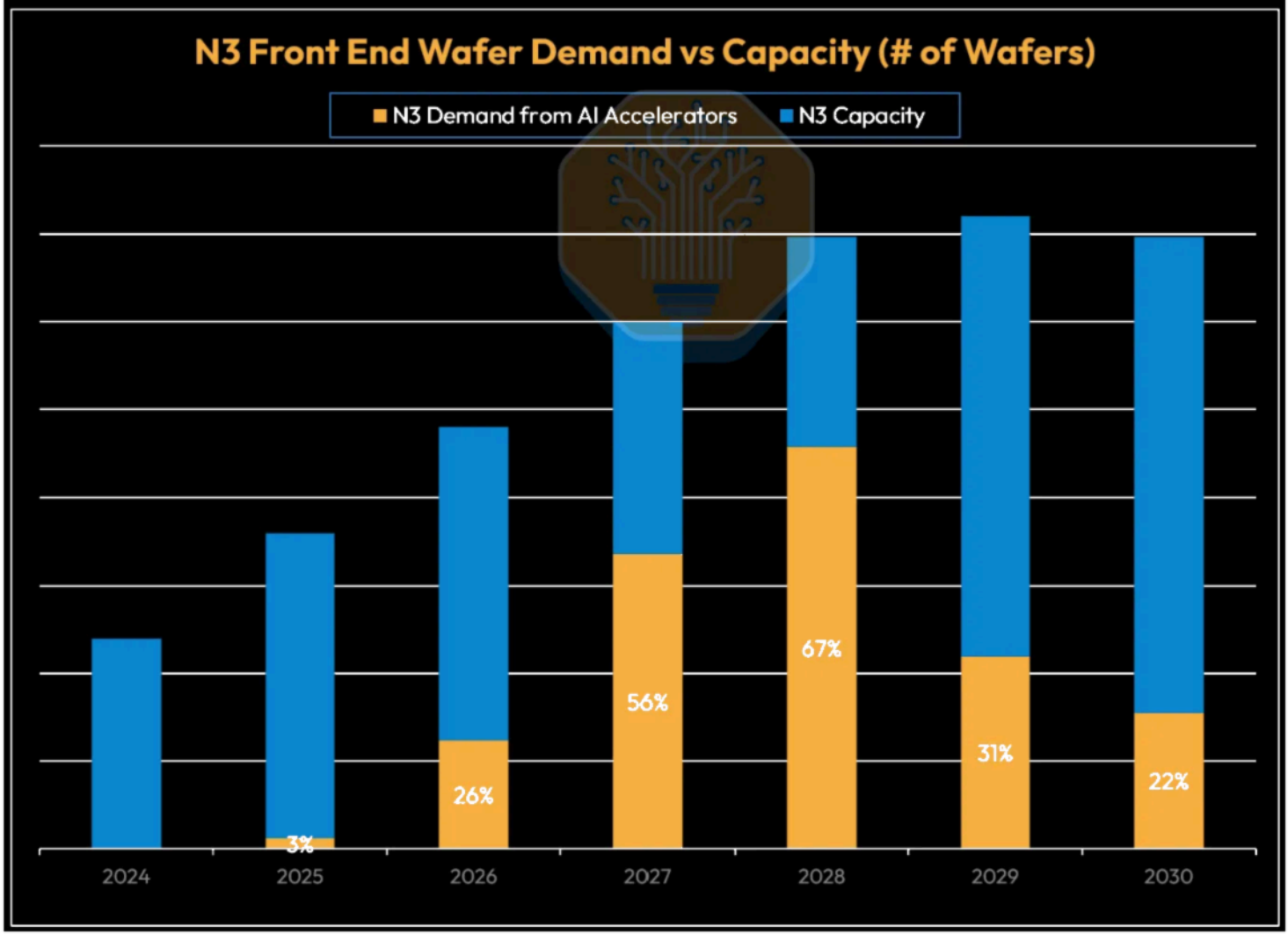
Jetson Thor	
BOM & margin analysis	
Compute die cost	[redacted]
Substrate	
Assembly + test + packaging	
PCB	
VRM	
LPDDR5X	
Total BOM	**1,103**
Margin	63%
ASP	**3,000**

Source: SemiAnalysis Accelerator Model

来源：SemiAnalysis 加速器模型

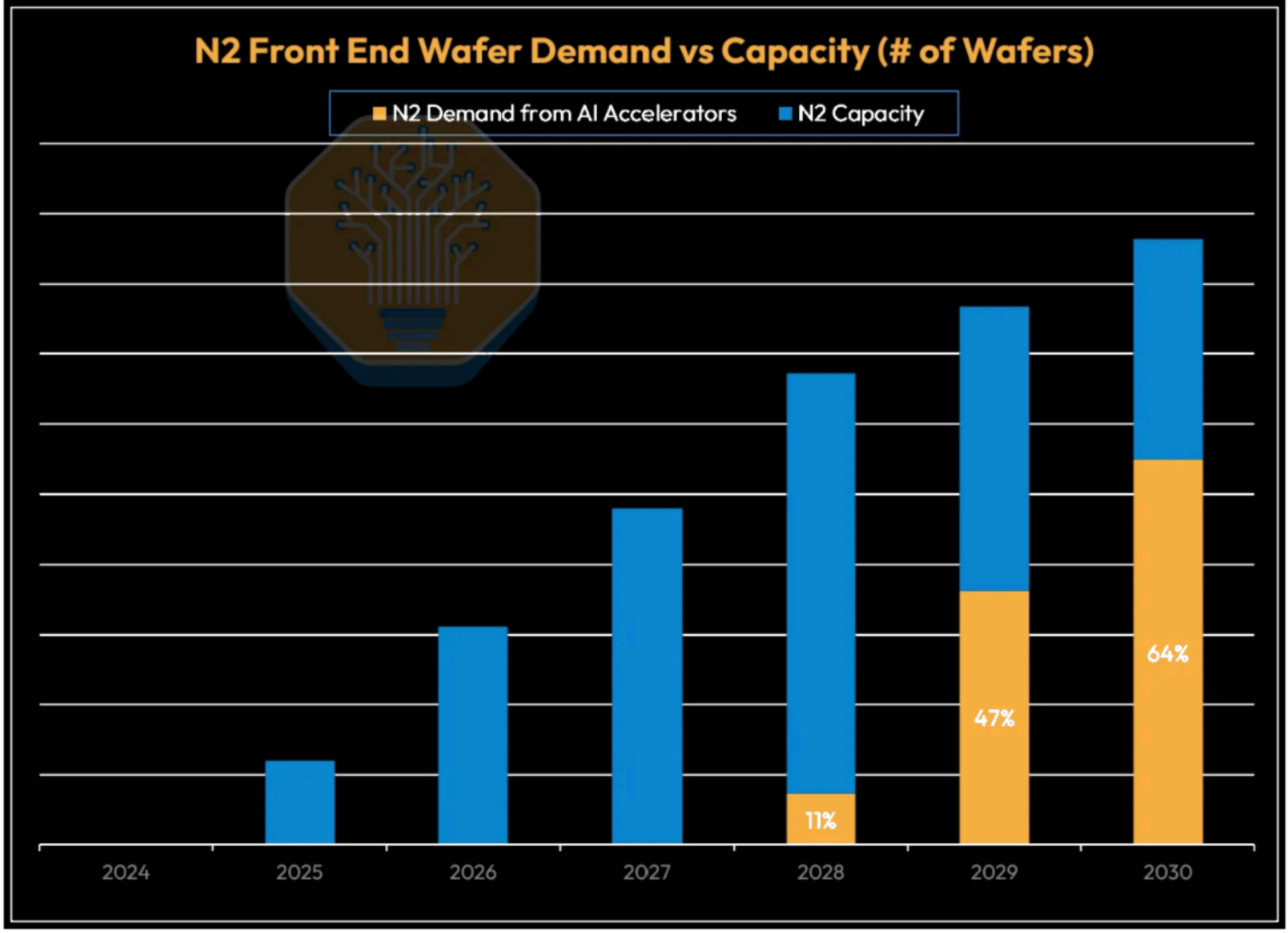
Worse, robot silicon is converging onto the exact nodes the datacenter is fighting over. Nvidia's Jetson line used to be insulated: the previous generation, Orin, ran on Samsung's SF8 process. The current generation, Thor, has closed that gap onto TSMC N4, which Blackwell uses, and as robots run on more capable models their silicon has to stay on the leading edge: the generation after Thor likely moves to N3 alongside Rubin, then N2 alongside Feynman, exactly the nodes the entire AI-accelerator roadmap is fighting over. So a low-volume, lower-margin Jetson will always be competing for advanced-node wafers from the back of the queue.

更糟糕的是，机器人芯片正逐渐收敛到与数据中心争夺的同一批先进制程节点上。英伟达的 Jetson 系列过去曾受到保护：上一代产品 Orin 采用的是三星的 SF8 工艺。而当前一代 Thor 则已缩小差距，改用台积电 N4 工艺——这正是 Blackwell 所使用的节点。随着机器人运行更强大的模型，其芯片必须保持在最前沿：Thor 的下一代很可能与 Rubin 一同迁移至 N3，再下一代则与 Feynman 一同迁移至 N2，而这些恰恰是整个 AI 加速器路线图都在争夺的节点。因此，一个低产量、低利润率的 Jetson 将始终在先进节点晶圆的排队中处于劣势。



Source: SemiAnalysis Accelerator Model

来源：SemiAnalysis 加速器模型



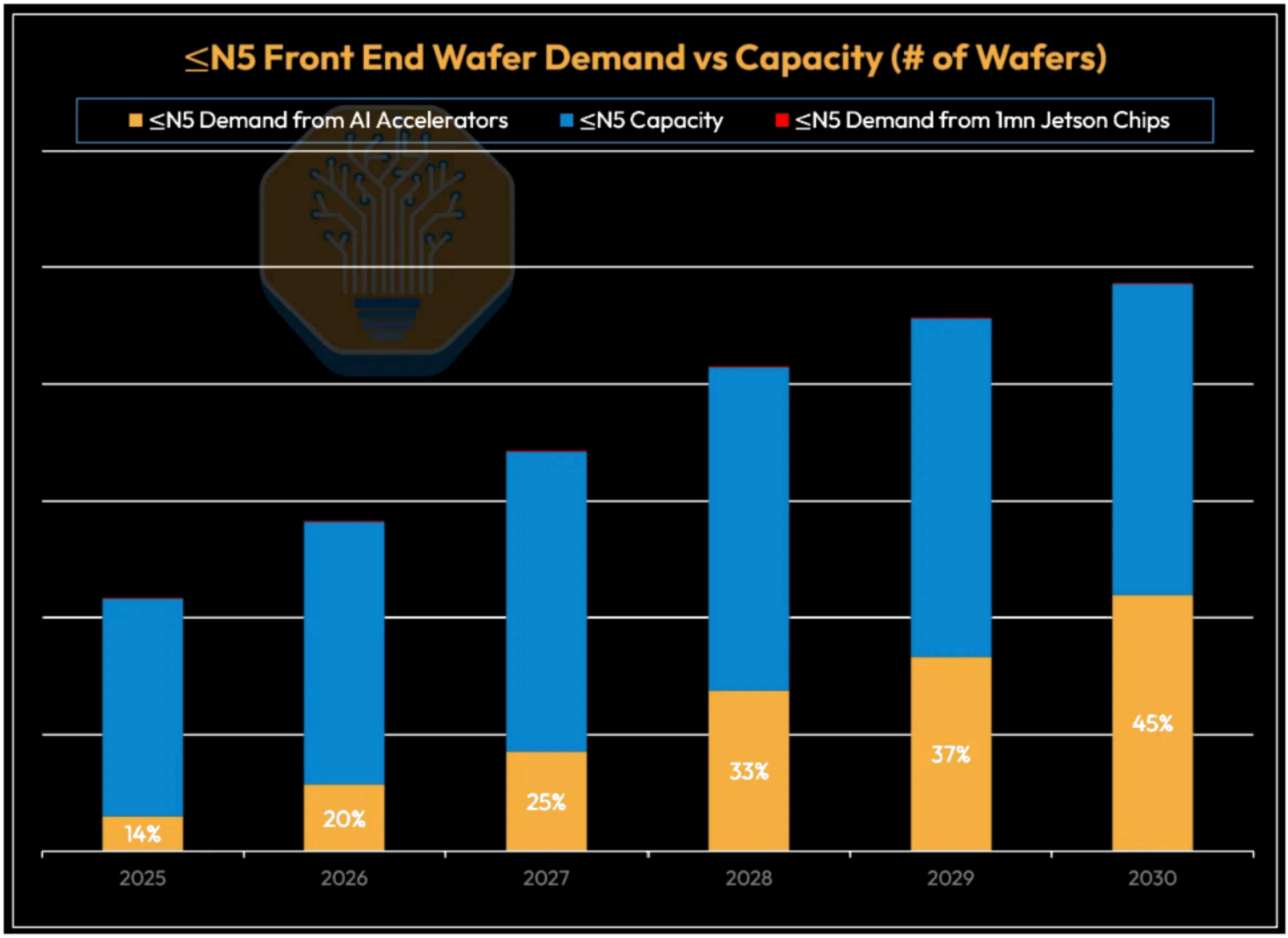
Source: SemiAnalysis Accelerator Model

来源：SemiAnalysis 加速器模型

But the wafer demand from robots is not where the strain is. A million Jetson-class chips in 2030, each around 400mm^2, is only about ten thousand wafers for the year, a rounding error next to the datacenter GPU ramp, because Thor's die is barely half a reticle size chip, and there simply aren't many robots yet. Supply only tightens once robots need tens of millions of chips a year, well beyond 2030. So the real question is not whether robots can get wafers. It is silicon efficiency: for a given fleet of robots, which approach consumes less leading-edge silicon? Putting a chip in every robot, or

serving their cognition from shared datacenter GPUs.

但机器人对晶圆的需求并非瓶颈所在。到 2030 年，一百万个 Jetson 级芯片，每个约 400 平方毫米，全年仅需约一万片晶圆，与数据中心 GPU 的扩张相比不过是四舍五入的误差，因为 Thor 的芯片尺寸勉强只有半块掩模版大小，而且目前机器人的数量本就不多。只有当机器人每年需要数千万颗芯片时，供应才会真正紧张，而那远在 2030 年之后。因此，真正的问题不在于机器人能否获得晶圆，而在于硅效率：对于给定的机器人队伍，哪种方案消耗更少的先进制程硅？是在每台机器人中嵌入芯片，还是从共享的数据中心 GPU 中提供其认知能力。



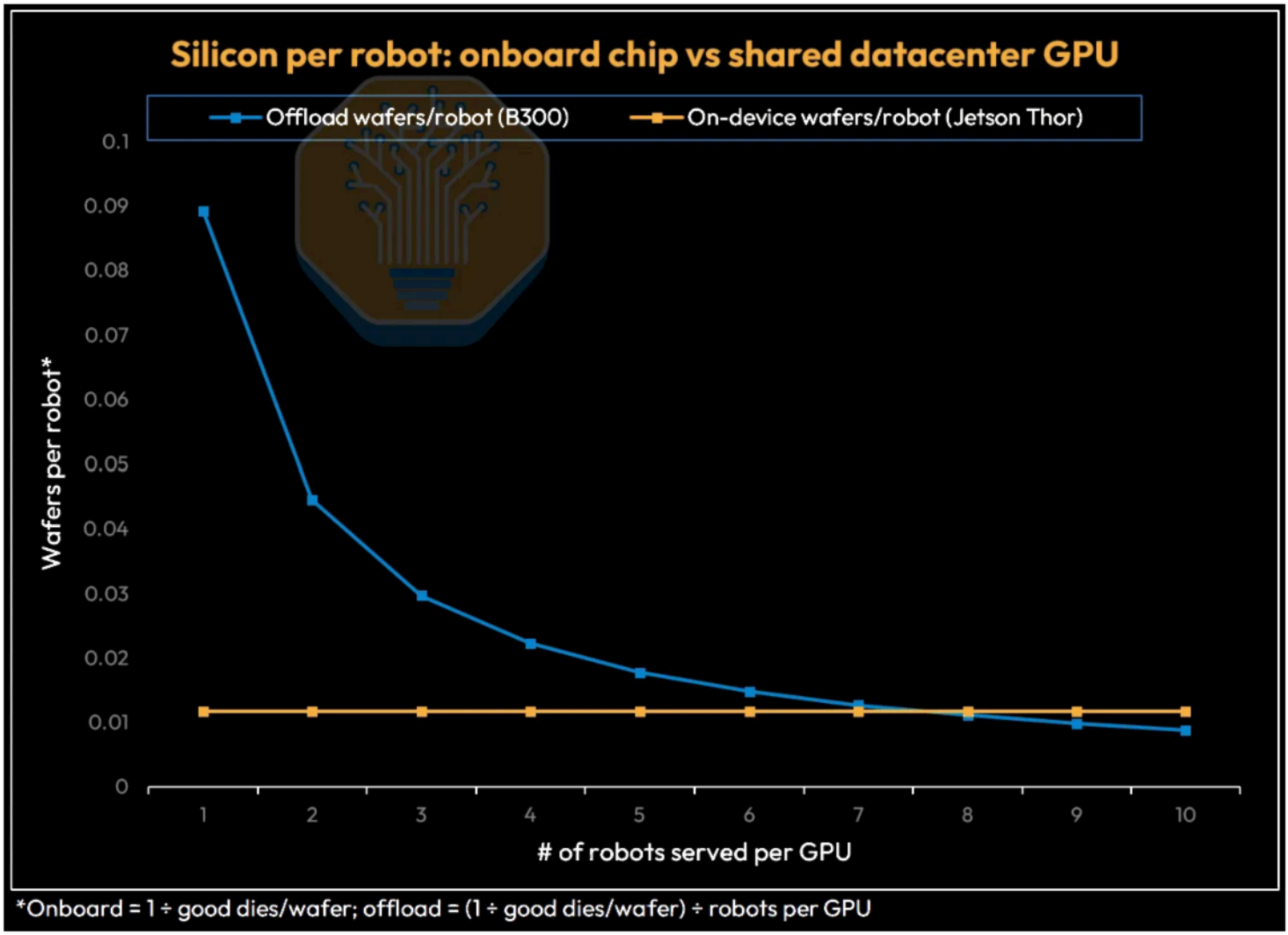
Source: SemiAnalysis Accelerator Model

来源：SemiAnalysis 加速器模型

If you look at silicon efficiency as wafers per robot, the onboard chip and the shared datacenter GPU cross over at around 7 robots per GPU. Beyond that, serving cognition from a shared GPU takes less silicon per robot than putting a chip in every machine. And that's what really matters here, because when fab supply is scarce, the approach

that uses less silicon per robot is the one that scales.

若从每台机器人消耗的晶圆数来看硅效率，机载芯片与共享数据中心 GPU 的交叉点大约在每块 GPU 对应 7 台机器人。超过这一比例后，从共享 GPU 提供认知服务所消耗的硅量将少于每台机器内置芯片。而这正是关键所在，因为当晶圆供应稀缺时，每台机器人消耗硅量更少的方案才能实现规模化扩展。



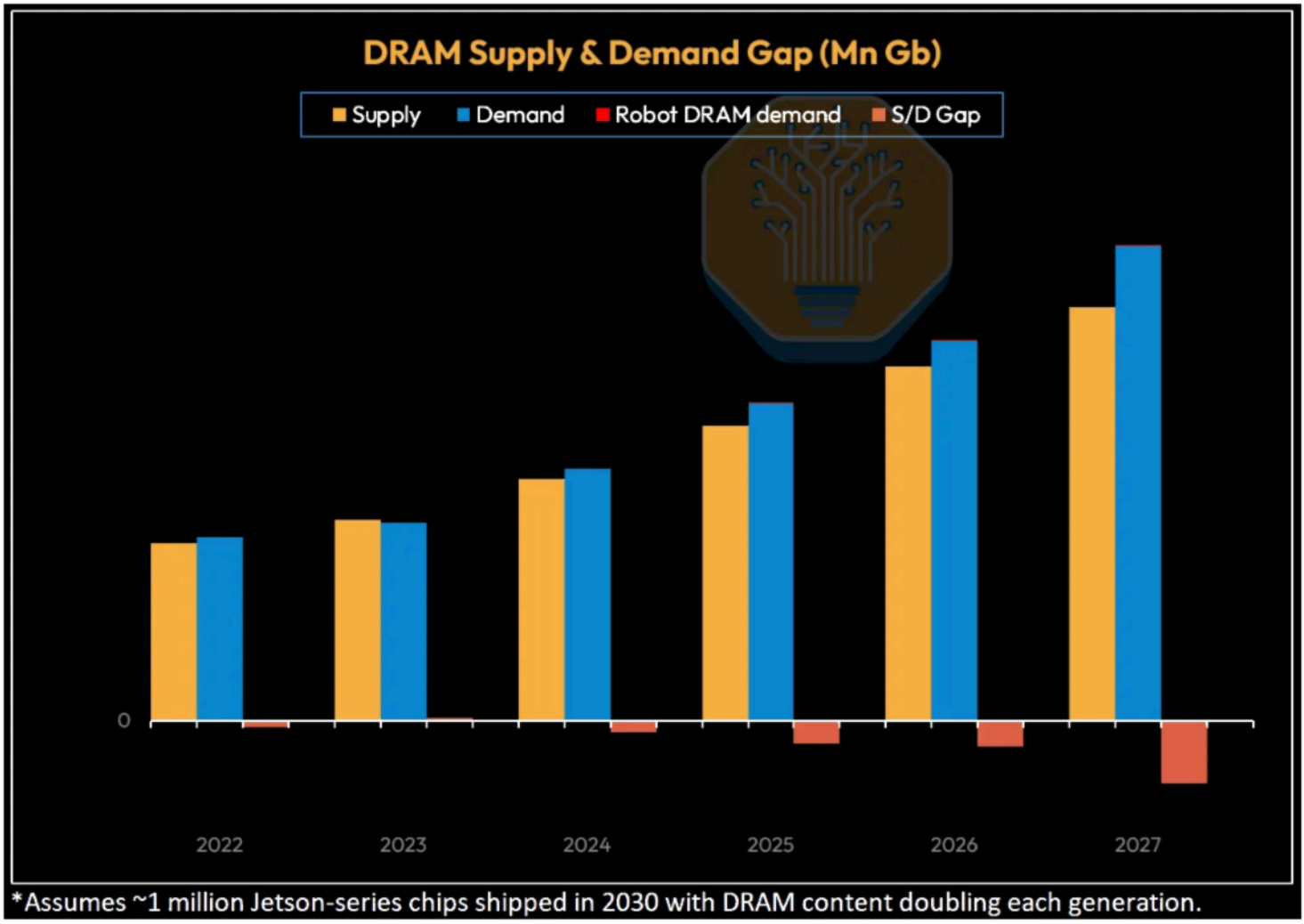
Source: SemiAnalysis 来源：SemiAnalysis

Another major bottleneck is memory. Each Jetson generation has carried more DRAM than the last: Jetson Xavier used 32GB, AGX Orin 64GB, and Jetson Thor now uses 128GB of LPDDR5X, with future generations set to keep climbing. Although total DRAM wafer capacity continues to grow, most of the incremental capacity is being absorbed by HBM for AI accelerators.

另一个主要瓶颈是内存。每一代 Jetson 所搭载的 DRAM 容量都比上一代更大：Jetson Xavier 使用了 32GB，AGX Orin 为 64GB，而 Jetson Thor 现在则采用了 128GB 的 LPDDR5X，未来几代还将继续攀升。尽管 DRAM 晶圆总产能持续增长，但新增产能的大部分正被用于 AI 加速器的 HBM 所吸收。

That leaves the commodity and LPDDR supply that robot brains depend on competing for a shrinking pool of non-HBM wafers. Similarly, when DRAM is the scarce resource, the approach that uses less DRAM per robot is the one that scales. The LPDDR is better off being allocated to Vera GPUs or wafers to HBM.

这使得机器人大脑所依赖的普通 DRAM 和 LPDDR 供应，不得不争夺日益缩减的非 HBM 晶圆份额。同样，当 DRAM 成为稀缺资源时，每台机器人使用 DRAM 更少的方法才是可扩展的方案。LPDDR 更适合分配给 Vera GPU，或者将晶圆用于 HBM。

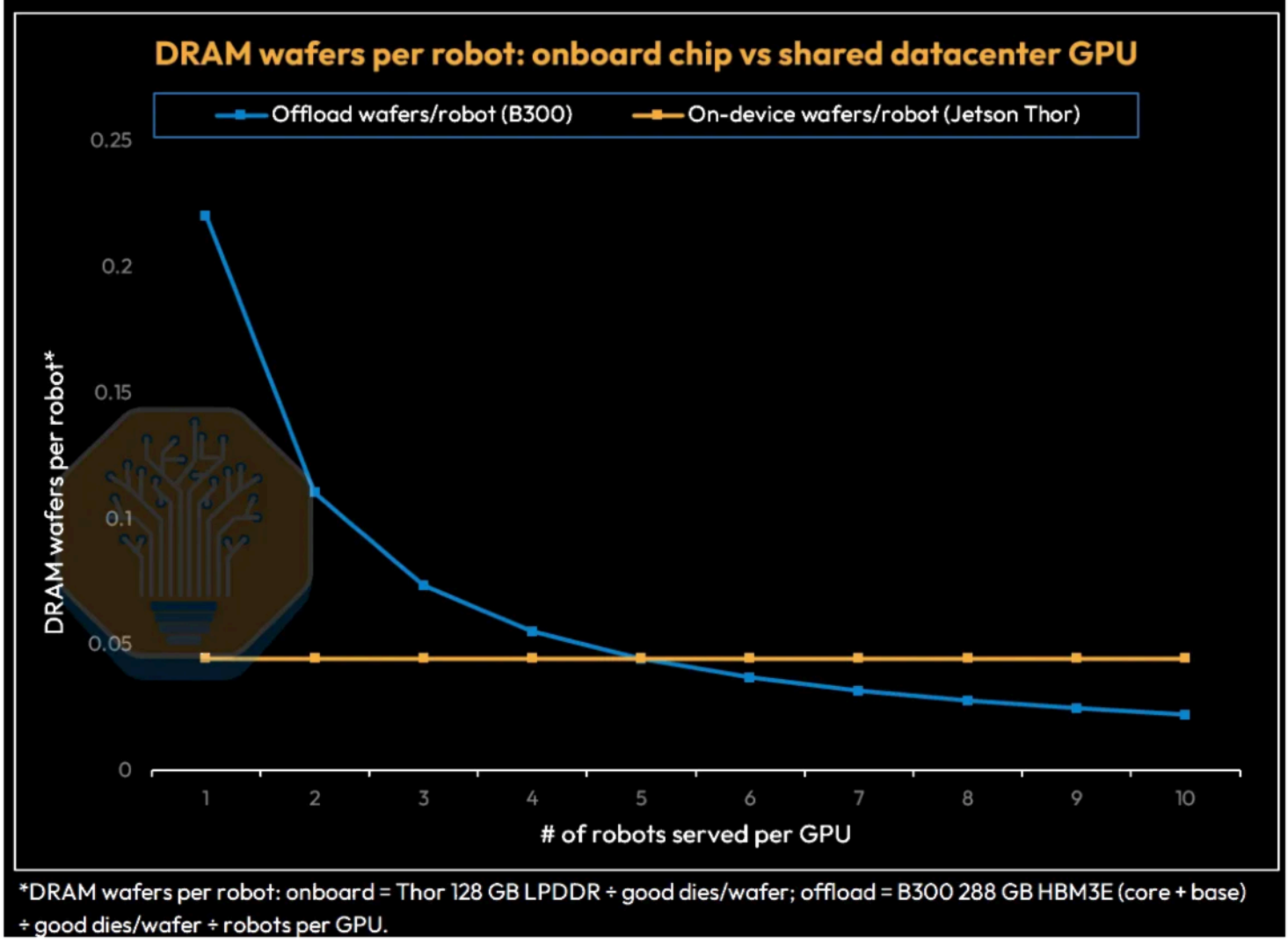


Source: SemiAnalysis Memory Model

来源：SemiAnalysis 内存模型

Memory tells a similar story, the onboard chip and the shared datacenter GPU cross over at around 5 robots per GPU. Beyond that, serving cognition from a shared GPU takes less DRAM per robot than putting a chip in every machine.

内存方面的情况也类似，板载芯片与共享数据中心 GPU 的交汇点大约在每 5 台机器人对应 1 个 GPU。超过这一比例后，从共享 GPU 提供认知服务所需的每台机器人 DRAM 量，要少于在每台机器中内置芯片的方案。



Source: SemiAnalysis 来源：SemiAnalysis

The TCO Verdict: One B300 server vs. 56 Thors

TCO 结论：一台 B300 服务器对比 56 台 Thor

We ran DreamZero, current leader on RoboArena, on a real B300, using the inference optimizations from NVIDIA's WAM paper (CUDA graphs, DiT caching, NVFP4 quantization, moving the scheduler onto the GPU). We also ran a single denoising step to match the DreamZero-Flash setup, though without Flash's retraining recipe, so task quality is unvalidated.

我们在真实的 B300 上运行了 RoboArena 当前领先的 DreamZero，采用了 NVIDIA WAM 论文中的推理优化（CUDA 图、DiT 缓存、NVFP4 量化、将调度器移至 GPU 上）。我们还执行了单步去噪以匹配 DreamZero-Flash 的设置，但未采用 Flash 的重新训练方案，因此任务质量未经验证。

DreamZero's serving profile is prefill-shaped: each inference pushes a full chunk of video latents (thousands of tokens) through the 14B DiT in parallel, so a B300 is

compute-bound at batch size 1. Batching therefore buys nothing: FLOPs scale one-for-one with batch size, and every robot just waits longer. Instead, we time-multiplex, with the B300 serving each robot sequentially. This is specific to video-generating WAMs. A VLA like π0 or GR00T flips the roofline: a few hundred tokens through a small backbone leaves the GPU memory-bandwidth-bound, so batching amortizes the same weight reads across robots and throughput scales nearly linearly with batch size. Serve a VLA fleet with continuous batching; serve a video WAM one robot at a time.

DreamZero 的服务模式以预填充为主：每次推理都会将一整块视频潜在表示（数千个 token）并行推入 14B DiT，因此 B300 在批大小为 1 时受计算限制。因此，批处理毫无收益：FLOPs 随批大小一对一增长，每个机器人只会等待更久。相反，我们采用时分复用，让 B300 依次服务每个机器人。这特定于视频生成的 WAM。像 π0 或 GR00T 这样的 VLA 则翻转了性能上限：几百个 token 通过小型骨干网络会让 GPU 受内存带宽限制，因此批处理能将相同的权重读取分摊到多个机器人上，吞吐量几乎随批大小线性增长。用连续批处理服务 VLA 车队；用逐台顺序方式服务视频 WAM。

A key assumption we made is that Network RTT is 10ms, meaning the GPU sits on-prem near the robots. Each inference buys the robot its next 1.6 seconds of motion, which makes 1.6 seconds our latency budget: the total time a request gets to spend across network, queueing, and compute before the robot runs out of instructions and stalls mid-motion.

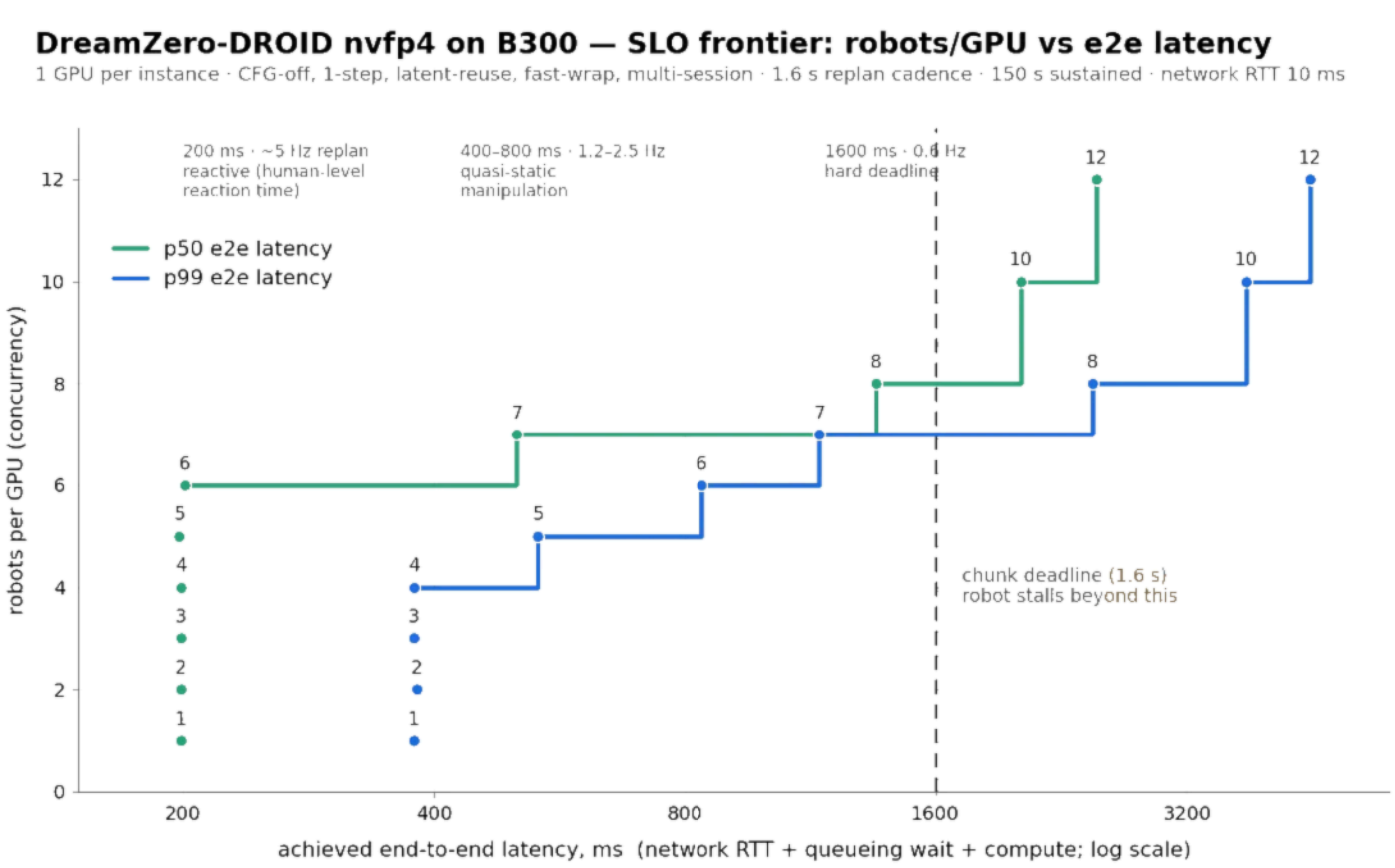
我们做的一个关键假设是网络往返时间为 10 毫秒，这意味着 GPU 就近部署在机器人附近。每次推理为机器人赢得接下来 1.6 秒的运动时间，因此 1.6 秒就是我们的延迟预算：一个请求在网络、排队和计算上总共可花费的时间，超过这个时间机器人就会耗尽指令，在运动中途停顿。

Optimization	H100	GB200
Baseline	1×	1.1×
System-level		
+ CFG Parallelism	1.9×	1.8×
+ DiT Caching	5.5×	5.4×
Implementation-level		
+ Torch Compile + CUDA Graphs	8.9×	10.9×
+ Kernel & Scheduler Opts.	9.6×	14.8×
+ Quantization (NVFP4)	—	16.6×

Source: Nvidia 来源：英伟达

We read the chart below on p99 latency (blue line), the 99th percentile response time. 99 out of 100 requests come back faster than this, so it's effectively the worst case. It matters more than the median (green line) here because a robot stalls the moment any single chunk shows up late. We see that the maximum number of robots we can process on 1 B300 GPU is 7 robots. At 7 robots, p99 is 1.16 seconds, still under the wall, with zero missed chunks over the run.

我们查看下图中的 p99 延迟（蓝线），即第 99 百分位响应时间。100 个请求中有 99 个的响应速度快于这个值，因此这实际上是最坏情况。它比中位数（绿线）更重要，因为只要任何一个数据块延迟到达，机器人就会立刻停顿。我们看到，1 块 B300 GPU 最多能处理 7 台机器人。在 7 台机器人时，p99 为 1.16 秒，仍在时限之内，整个运行过程中没有错过任何数据块。



Source: SemiAnalysis 来源：SemiAnalysis

Now that we know we can time-multiplex 7 robots per GPU, the fundamental TCO question stands:

既然我们已经知道每块 GPU 可以时分复用 7 台机器人，那么根本性的 TCO 问题便浮现出来：

Is it cheaper to offload compute to a B300 server, or to run inference on-device in a Jetson Thor?

将计算任务卸载到 B300 服务器上，还是在 Jetson Thor 设备端进行推理，哪种方式成本更低?

We build up two scenarios: the offload scenario assumes one Blackwell-class server which can serve 7 robots (i.e. one B300 NVL8 server serving 56 robots), and an on-robot scenario of the same 56 robots each carrying its own Jetson Thor. In both scenarios, common parts across both cases (i.e. mechanical components, robot shells)

are not included in the BOM cost calculation.

我们构建了两种场景：卸载场景假设一台 Blackwell 级服务器可服务 7 台机器人（即一台 B300 NVL8 服务器服务 56 台机器人），而设备端场景则是同样的 56 台机器人各自携带一台 Jetson Thor。在两种场景中，两种情况下共有的部分（如机械部件、机器人外壳）不计入 BOM 成本计算。

We also discuss the biggest challenge with this deployment mechanism.

我们还将讨论这种部署机制面临的最大挑战。

The offload scenario is one B300 NVL8 plus 56 wireless boards (one per robot) to connect the robot to the GPU cluster in the datacenter. That brings all-in capex to ~$542K, which, including operating costs like colocation and power, leads us to a TCO of $18.32/hr in the offload scenario.

卸载场景是：一台 B300 NVL8 加上 56 块无线板卡（每台机器人一块），用于将机器人连接到数据中心内的 GPU 集群。这使得总资本支出达到~$542K，若计入托管和电力等运营成本，卸载场景下的总体拥有成本（TCO）为 18.32 美元/小时。

Cost of Offloading Compute to GPU Server

		B300 1200W
Server Cost	USD	
Server Service	USD	
Networking Cost	USD	
Software Licenses and Other Costs	USD	
Other Installation	USD	
Total Cluster Capital Cost	USD	
Wireless TX/RX Card	USD	
Total Capital Cost	**USD**	**$542,434**
Useful Life	Years	6
WACC	%	10.3%
Total Capital Costs per Month	USD	$10,131
Total Operating Costs per Month	USD	$3,242
Capital Cost, per Hour	USD/hr	$13.88
Operating Cost, per Hour	USD/hr	$4.44
Total Cost of Ownership per Hour	**USD/hr**	**$18.32**

Source: SemiAnalysis AI TCO Model

来源：SemiAnalysis AI TCO 模型

The on-robot scenario is 56 Thor modules at $3,500 per unit, each with its own baseboard, storage, cooling, incremental battery, and onboard radio connectivity, landing at ~$230K. Accounting for the above, along with power costs for the Thor modules, leads us to a TCO of $6.58/hr for the 56-chip deployment.

机器人端场景则是 56 个 Thor 模块，单价 3500 美元，每个模块配备独立的底板、存储、散热、增量电池和机载无线电连接，总成本为~$230K。计入上述成本以及 Thor 模块的电力费用后，56 芯片部署方案的总体拥有成本（TCO）为 6.58 美元/小时。

Cost of On-Device Compute

		Jetson Thor
Jetson Thor	USD	
Jetson Thor Baseboard	USD	
Local Storage (NVMe SSD 512GB)	USD	
Thermal Solution	USD	
Battery (Incremental capacity addition)	USD	
Onboard connectivity	USD	
Spares / RMA provisioning (~7%)	USD	
Total Capital Cost	**USD**	**$230,392**
Useful Life	Years	6
WACC	%	10.3%
Total Capital Costs per Month	USD	$4,303.1
Total Operating Costs per Month	USD	$499.4
Capital Cost, per Hour	USD/hr	$5.89
Operating Cost, per Hour	USD/hr	$0.68
Total Cost of Ownership per Hour	**USD/hr**	**$6.58**

Source: SemiAnalysis AI TCO Model

来源：SemiAnalysis AI TCO 模型

Before accounting for utilization, the two cases are close on TCO per FP4 dense FLOPs: $0.15/hr/PFLOP in the offload scenario, and $0.11/hr/PFLOP in the on-robot

scenario.

在计入利用率之前，两种方案在每 FP4 密集 FLOP 的总体拥有成本（TCO）上相差无几：卸载场景为 0.15 美元/小时/PFLOP，而机器人端场景为 0.11 美元/小时/PFLOP。

TCO per Performance Metrics			
		B300 1200W	Jetson Thor
Total FP4 PFLOPs - Dense	PFLOPs	120	58
Total FP8 PFLOPs - Dense	PFLOPs	40	29
Total FP16 PFLOPs - Dense	PFLOPs	20	15
Total Memory Bandwidth	TB/s	64	15
TCO per FP4 Dense FLOPs, before Utilization	USD/hr/PFLOPs	$0.15	$0.11
TCO per FP8 Dense FLOPs, before Utilization	USD/hr/PFLOPs	$0.46	$0.23
TCO per FP16 Dense FLOPs, before Utilization	USD/hr/PFLOPs	$0.92	$0.45
TCO per Memory Bandwidth, before Utilization	USD/hr/TB/s	$0.29	$0.43

Source: SemiAnalysis AI TCO Model

来源：SemiAnalysis AI 总体拥有成本模型

However, utilization is the swing factor in TCO. A robot and the GPU behind it are almost entirely fixed capital, so the bill is the same whether the asset runs or sits idle, and cost per productive hour scales inversely with how much it works.

然而，利用率是总体拥有成本中的关键变量。机器人和其背后的 GPU 几乎完全是固定资本，因此无论资产运行还是闲置，账单都相同，而每生产小时的成本与资产的工作量成反比。

On-robot chip utilization today is highly deployment-dependent. In the home, one company that has deployed robots in volume told us its robots run just 1-2 hours a day, only 4-8% of the clock, because chores are bursty and current models still cap what can run unsupervised. That figure is trending up, toward 4-5 hours (17-21%), as capability improves and new features drive engagement. But the home has a hard ceiling: eventually the chores run out. Industrial demand is continuous and should clear a higher bar. Figure's BMW deployment is an anchor we have: roughly 1,250 operational hours over about 11 months of weekday shifts, working around 10 hours a day at 40% utilization. As model capability improves, industrial utilization should climb well

beyond that.

如今，机器人芯片的利用率高度依赖于部署场景。在家庭环境中，一家已大规模部署机器人的公司告诉我们，他们的机器人每天仅运行 1-2 小时，只占全天时间的 4-8%，这是因为家务任务具有突发性，且当前模型仍限制了可无人监督运行的范围。随着能力提升和新功能带动使用需求，这一数字呈上升趋势，预计将达到每天 4-5 小时（17-21%）。但家庭场景存在一个硬性上限：家务终究会做完。工业需求则是持续性的，应能达到更高的标准。Figure 公司与宝马的合作为我们提供了一个参考基准：在约 11 个月的工作日轮班中，累计运行约 1,250 小时，每天工作约 10 小时，利用率达 40%。随着模型能力的提升，工业利用率应会远超这一水平。

Utilization under Various Deployment Scenarios

Deployment scenario	Hours/day	Utilization
Home deployment, low utilization	2	8.3%
Home deployment, medium utilization	5	20.8%
Home deployment, high utilization	10	41.7%
Industrial deployment, low utilization	10	41.7%
Industrial deployment, medium utilization	15	62.5%
Industrial deployment, high utilization	20	83.3%

Source: SemiAnalysis AI TCO Model

来源：SemiAnalysis AI TCO 模型

We conservatively model ~40% utilization as our base case for on-robot chip utilization, which is factored into the table below. Net of utilization, the offload scenario reaches a TCO per FP4 dense FLOPs of $0.17/hr/PFLOP, while the on-robot scenario reaches a TCO per FP4 dense FLOPs of $0.28/hr/PFLOP. Here, the offload scenario reaches a mere ~60% of the on-robot scenario.

我们保守地将~40%的利用率作为机器人端芯片利用率的基准情景，该数据已纳入下表。扣除利用率因素后，卸载场景的每 FP4 稠密 FLOPs TCO 达到$0.17/小时/PFLOP，而机器人端场景的每 FP4 稠密 FLOPs TCO 为$0.28/小时/PFLOP。在此，卸载场景仅为机器人端场景的~60%。

TCO per Performance Metrics			
		B300 1200W	Jetson Thor
Total FP4 PFLOPs - Dense	PFLOPs	120	58
Total FP8 PFLOPs - Dense	PFLOPs	40	29
Total FP16 PFLOPs - Dense	PFLOPs	20	15
Total Memory Bandwidth	TB/s	64	15
TCO per FP4 Dense FLOPs, before Utilization	USD/hr/PFLOPs	$0.15	$0.11
TCO per FP8 Dense FLOPs, before Utilization	USD/hr/PFLOPs	$0.46	$0.23
TCO per FP16 Dense FLOPs, before Utilization	USD/hr/PFLOPs	$0.92	$0.45
TCO per Memory Bandwidth, before Utilization	USD/hr/TB/s	$0.29	$0.43
Utilization Rate[1, 2]	%	90.0%	40.0%
TCO per FP4 Dense FLOPs, after Utilization	USD/hr/PFLOPs	$0.17	$0.28
TCO per FP8 Dense FLOPs, after Utilization	USD/hr/PFLOPs	$0.51	$0.56
TCO per FP16 Dense FLOPs, after Utilization	USD/hr/PFLOPs	$1.02	$1.13
TCO per Memory Bandwidth, after Utilization	USD/hr/TB/s	$0.32	$1.08

1. We define GPU utilization rate is measured as number of robots served per GPU over maximum batch size.

2. On-robot utilization rate is highly dependent on robot deployment scenarios.

Source: SemiAnalysis AI TCO Model

来源：SemiAnalysis AI TCO 模型

The shared GPU server often runs the other way, as its compute resources are pooled across the fleet and busy around the clock. Given that, in the scenario above, one B300 GPU can serve 7 robots within the latency budget, we model ~90% utilization of the GPU server.

共享 GPU 服务器通常以另一种方式运行，因为其计算资源在车队中池化并全天候忙碌。鉴于上述场景中，一台 B300 GPU 可在延迟预算内服务 7 台机器人，我们将 GPU 服务器的利用率建模为~90%。

Towards the top right corner of the sensitivity analysis below, high GPU server utilization and low chip on-device utilization swings the TCO per PFLOP massively in favor of the offloaded scenario. The offload scenario for current industrial deployments reaches ~60% the TCO per PFLOP of the on-robot scenario, reaching an even lower ~15% for home deployments.

在下方敏感性分析的右上角区域，高 GPU 服务器利用率与低芯片端利用率使得每 PFLOP 的总拥有成本大幅偏向于卸载场景。当前工业部署中的卸载场景，其每 PFLOP 的 TCO 仅为机器人端场景的~60%，而在家庭部署中更是低至~15%。

It is only in several cases towards the bottom right corner wherein low GPU server utilization and high on-device utilization swings the TCO per PFLOP in favor of on-device inference.

只有在右下角区域的少数情况下，低 GPU 服务器利用率与高设备端利用率才会使每 PFLOP 的 TCO 偏向于设备端推理。

TCO per FP4 PFLOPs, with Varying Utilization Assumptions

Utilization (Chip On-device) \ Utilization (GPU Server)	80%	90%	100%
10%	0.17x	0.15x	0.14x
20%	0.34x	0.30x	0.27x
30%	0.51x	0.45x	0.41x
40%	0.68x	0.60x	0.54x
50%	0.84x	0.75x	0.68x
60%	1.01x	0.90x	0.81x
70%	1.18x	1.05x	0.95x
80%	1.35x	1.20x	1.08x

Green indicates scenarios where offloading is cheaper than on-device edge inference.
Red indicates scenarios where offloading is more expensive than on-device edge inference.

Source: SemiAnalysis AI TCO Model

来源：SemiAnalysis AI TCO 模型

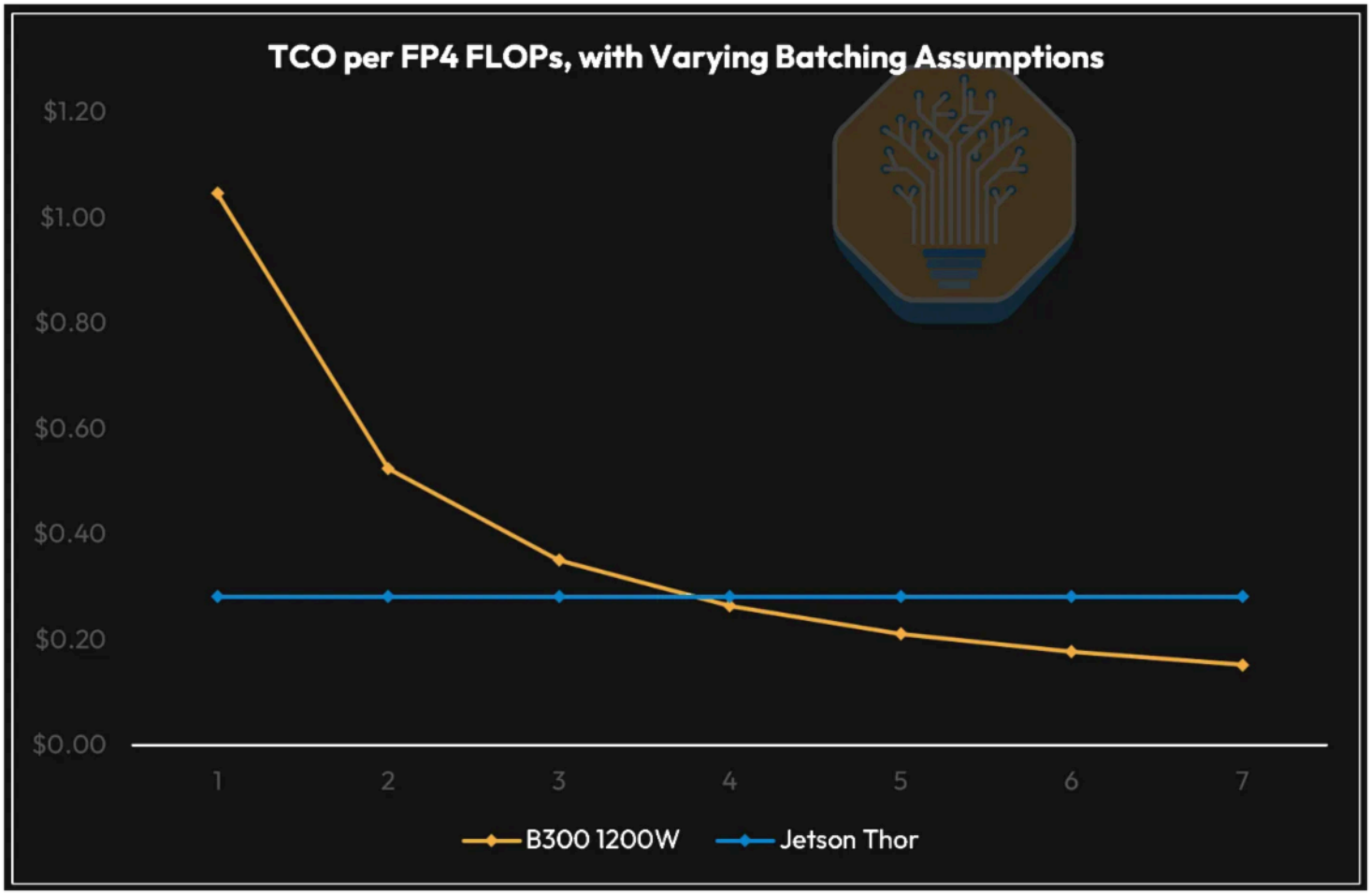
The question arises naturally: How many robots have to be batched in order for such TCO per PFLOP savings to come through in favor of the offload scenario? Varying our batching assumptions below, for a standard industrial deployment, we see that anywhere from 4 robots per GPU and upwards, the TCO per PFLOP shifts in favor of the offloaded scenario.

自然产生一个问题：需要将多少机器人进行批处理，才能让上述每 PFLOP 的 TCO 节省优势体现出来，从而偏向卸载场景？通过调整下方的批处理假设，对于标准工业部署，我们发现每 GPU 从 4 个机器人起，每 PFLOP 的 TCO 就会转向有利于卸载场景。

On the whole, we find that offloading inference to a GPU server is a compelling case, except for cases with very small robot deployments with one on-site server sitting mostly idle - though even then, one could make the argument for renting compute as

needed rather than owning an entire server to serve a few robots.

总体来看，我们发现将推理任务卸载到 GPU 服务器上是一个颇具吸引力的方案，除非是机器人部署规模极小、现场仅有一台服务器且大部分时间闲置的情况——不过即便如此，也可以主张按需租用计算资源，而非为几台机器人专门购置整台服务器。



Source: SemiAnalysis AI TCO Model

来源：SemiAnalysis AI TCO 模型

Where it works and where it doesn't - the network layer

适用与不适用的场景——网络层

There are obvious issues with running the heavy part of your model off-device (see image below). We aren't arguing that it will be easy; we are just arguing that it will be more practical than local compute for some applications.

将模型的重计算部分放在设备外运行存在明显的问题（见下图）。我们并非认为这会轻而易举；只是主张在某些应用中，这比本地计算更为实际。



Ryan Julian @ryancjulian · 5d

my man has obviously never tried to connect to wifi/5G on a factory floor

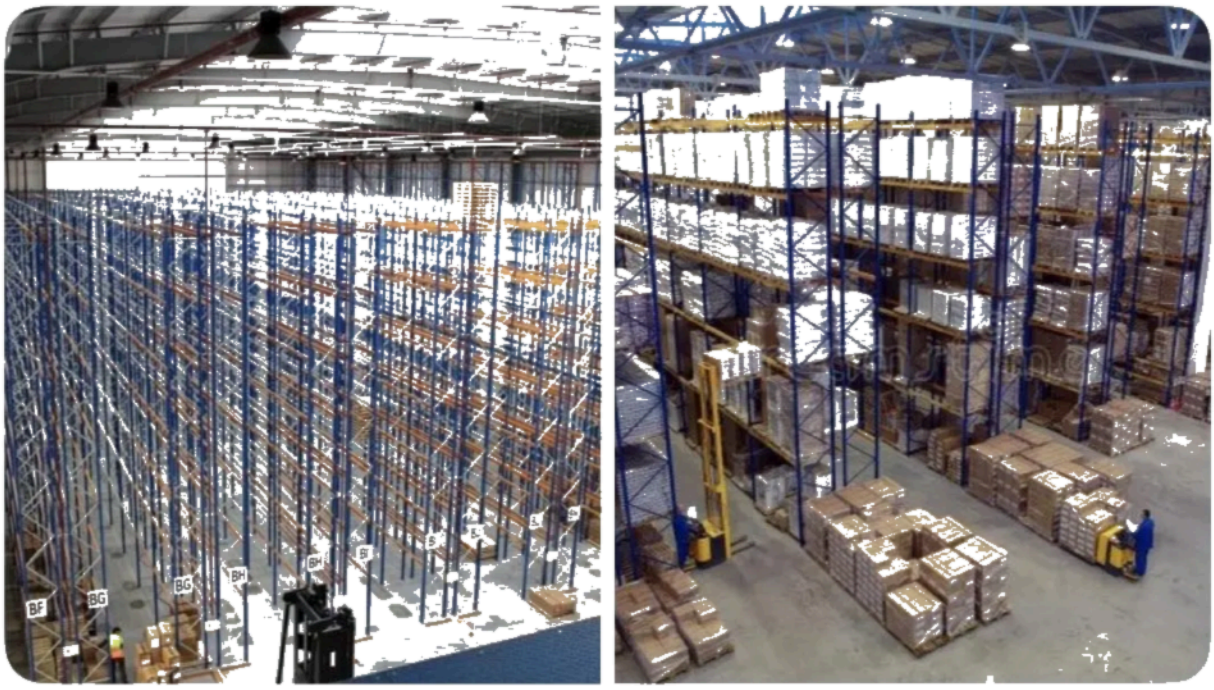
> 
> **Gianluca** @gianlucabencomo · 5d
> Without a doubt:
>
> A world running foundation models on a billion robots won't be doing it locally....

6 2 70 12K



Jack @JacklouisP · 4d

Searching for bars in this



6 176

Source: X 来源：X

Certain settings will be impossible to run off-device compute; caves, ruins, and disaster sites for example. Settings that are unpredictably occluded, dynamic, and with none of the wireless conditions correctly configured will rely on local compute options

because the network cannot exist reliably, if at all.

某些场景将无法依赖设备端计算运行，例如洞穴、废墟和灾难现场。这些环境存在不可预测的遮挡、动态变化，且无线条件无法正确配置，因此将依赖本地计算选项，因为网络即使存在，也无法可靠建立。

Open agriculture and remote fields will be conditional on the latency budget of the model, since services like Starlink cover these areas but have higher latency and jitter. Properly managing this is difficult because we can only control some parts of the setup. We imagine that these settings would utilize a mix of on-device and off-device options, and the model's latency budget will dictate the choice.

开放农业和偏远田地场景将取决于模型的延迟预算，因为星链（Starlink）等服务虽覆盖这些区域，但延迟较高且存在抖动。妥善管理这一点颇具挑战，因为我们只能控制设置中的部分环节。我们设想这些场景将采用设备端与设备端外选项的混合方案，而模型的延迟预算将决定具体选择。

The home is easier, but not ideal. Walls and other obstacles create weak and dead zones, and the typical home router is not optimized for handoffs. Wireless conditions are often good, but congestion is a factor: homes typically run on shared fiber, so neighboring homes and apartments can degrade conditions. Making this work in the home under a tight latency budget would require specialized routers, as well as characterizing performance during peak WiFi utilization hours when the shared fiber link is loaded.

家庭环境相对容易，但并非理想。墙壁及其他障碍物会形成信号弱区和盲区，而普通家用路由器并未针对无缝切换进行优化。无线条件通常良好，但拥塞是一个因素：家庭通常使用共享光纤，因此邻近住宅和公寓可能会降低网络状况。要在严格的延迟预算下让这一方案在家庭中奏效，就需要专用路由器，并需在共享光纤链路负载较高的 WiFi 使用高峰时段评估性能表现。

Factories and warehouses are extremely difficult because they are massive labyrinths of moving metal and heavy machinery that create a worst-case scenario for RF propagation. They suffer severe multipath interference, electromagnetic noise, and constant line-of-sight occlusion. But factories and warehouses are also where we have the most control. URLLC protocols and private 5G bands mean factories can achieve sub-10ms wireless latency when optimized. We can eliminate the internet transit layer entirely by placing the GPUs a few hundred feet away in the same building. Routers

can use beamforming to maximize signal to robots on the floor. Factories and warehouses require the most work to be functional, but they can become the most optimized deployments as a result. The investment required depends on the latency budget, and whether the juice is worth the squeeze depends on whether this setup is cheaper than hundreds of robots with local GPUs.

工厂和仓库环境极为复杂，因为它们是移动金属和重型机械构成的巨大迷宫，这对射频传播构成了最恶劣的工况。它们遭受严重的多径干扰、电磁噪声以及持续的视线遮挡。然而，工厂和仓库也是我们控制力最强的场所。URLLC 协议和私有 5G 频段意味着，在优化得当的情况下，工厂可以实现低于 10 毫秒的无线延迟。我们可以将 GPU 放置在数百英尺外的同一建筑内，从而完全消除互联网传输层。路由器可以利用波束成形技术，最大化向车间内机器人传输的信号强度。工厂和仓库需要投入最多的工作才能实现功能，但正因如此，它们也能成为最优化部署的场所。所需投资取决于延迟预算，而是否值得投入则取决于这种方案是否比数百台配备本地 GPU 的机器人更为经济。

These deployments already exist, and off-device compute in factories and warehouses will be able to lean on existing infrastructure in some cases. And when the latency budget is generous, the in-building buildout may not be needed at all: one robot foundation model developer we know already serves production inference for an automaker and a industrial manufacturer from a cloud availability zone in the local metro, with no GPUs on site, because the control loop is able to tolerate a good amount of latency. These robots already talk to the cloud for data upload and tele-op, so inference rides on infrastructure that exists anyway. The only new cost is compute.

这些部署已经存在，工厂和仓库中的非设备端计算在某些情况下将能够依托现有的基础设施。而当延迟预算较为宽裕时，楼内建设可能根本不需要：我们认识的一家机器人基础模型开发商，已经通过本地都市区的云可用区，为一家汽车制造商和一家工业制造商提供生产推理服务，现场无需 GPU，因为控制回路能够容忍相当程度的延迟。这些机器人本就会与云端通信以进行数据上传和远程操作，因此推理服务搭载在原本就存在的基础设施上。唯一新增的成本是计算资源。

The country and regulatory environment in which the robot operates also has a significant impact. Certain nations employ national firewalls that perform deep packet inspection, which can severely degrade latency and jitter. Ultimately, once data is transmitted through the network, the host country exerts a degree of control that can directly compromise wireless conditions. This means that how countries react to

robotics in the future will shape the on-device versus off-device tradeoff on a per-country basis. Speculatively, countries like the US will offer excellent wireless conditions for off-device robotics, while China and other countries that perform deeper packet inspection may end up pushing harder toward on-device compute.

机器人运行所在的国家和监管环境也会产生重大影响。某些国家采用执行深度数据包检测的国家防火墙，这可能会严重恶化延迟和抖动。最终，一旦数据通过网络传输，东道国就会施加一定程度的控制，从而直接损害无线条件。这意味着各国未来对机器人的反应方式将决定在设备端与设备端外计算之间的权衡，且这种权衡因国家而异。推测而言，像美国这样的国家将为设备端外机器人提供出色的无线条件，而中国及其他进行更深度数据包检测的国家可能最终会更倾向于推动设备端计算。

Key to all of this is specifically architecting for safety, low-level control, and fallbacks for when the network inevitably fails. The robot needs to be able to run an extremely low-level local policy that returns it to a safe zone on failure. Think Figure's policy that allows the robot to return for maintenance when an actuator fails, automatically accommodating a modified embodiment. In this case, it's a modified brain; the fallback is a simple control loop that is 100% safety focused. Network uptime needs to hit 99.99%, so we need reliability first and this fallback second. Without both, there will be no off-device deployments. This failure mode is already the norm in today's warehouses: AGVs trigger a safety stop and freeze in place when wireless drops.

这一切的关键在于，要专门针对安全性、底层控制以及网络不可避免地出现故障时的后备方案进行架构设计。机器人需要能够运行一个极低层级的本地策略，以便在发生故障时将其带回安全区域。可以想想 Figure 公司的策略：当执行器发生故障时，机器人能够自动适应改变后的机体形态，并返回进行维护。在这种情况下，改变的相当于“大脑”；而后备方案则是一个完全以安全为核心的简单控制回路。网络正常运行时间需要达到 99.99%，因此我们首先要确保可靠性，其次才是这个后备方案。如果两者缺一，就无法实现设备外的部署。这种故障模式在当今的仓库中已是常态：当无线信号中断时，AGV（自动导引车）会触发安全停止并原地冻结。

Conclusion 结论

For a large and growing share of robot deployments, the brain belongs in the datacenter. The barrier to mass adoption is price: an expensive humanoid has no mass market, and prices have to come down for adoption to grow. That makes cost the battleground, and pulling compute out of every robot to share GPUs in a datacenter is

a strong lever. It also fixes the worst part of being a robotics company: chips bought months before robots sell are dead capital, and ten thousand staged robots is $30M depreciating in a warehouse, whereas offloaded compute is a shared GPU that is already earning its keep.

对于机器人部署中庞大且不断增长的部分而言，大脑应归属于数据中心。大规模普及的障碍在于价格：昂贵的仿人机器人没有大众市场，而价格必须下降才能推动普及增长。这使得成本成为战场，将计算从每台机器人中剥离出来，在数据中心共享 GPU，是一个强有力的杠杆。这也解决了作为机器人公司最糟糕的部分：在机器人销售前数月购得的芯片是死资本，一万台待售机器人意味着三百万美元在仓库中贬值，而卸载计算则意味着共享 GPU 已经在创造收益。

Supply points the same way: leading-edge wafers and DRAM are largely allocated for, and NVIDIA has little reason to divert them while LLMs eat the supply, so offloading rides that buildout instead of fighting it.

供应端也指向同一方向：领先的晶圆和 DRAM 大多已被分配，而 LLM 消耗着供应，NVIDIA 几乎没有理由将其分流，因此卸载计算顺应了这一建设浪潮，而非与之对抗。

None of this makes offloading universal. It depends on a network that holds up, and the environments vary widely: factories can be engineered for low latency, homes are workable with the right hardware, open fields are conditional on the latency budget, and some settings, like caves and disaster sites, rule it out entirely. A safety-focused local fallback is non-negotiable everywhere. But where the network can be made reliable, the economics win out, and the gap only grows as fleets scale and robots work more hours. What remains is execution, and the winners will be the few teams that can make the datacenter feel local.

这些都不意味着卸载计算具有普适性。它依赖于稳定可靠的网络，而环境差异巨大：工厂可以通过工程手段实现低延迟，家庭环境在配备合适硬件的情况下可行，开阔场地则取决于延迟预算，而某些场景，如洞穴和灾害现场，则完全排除这种可能。在任何地方，以安全为核心的本地回退方案都是不可妥协的底线。但在网络能够做到可靠的地方，经济性优势便会显现，而且随着机器人车队规模扩大、工作时长增加，这一差距只会不断拉大。剩下的就是执行问题，而赢家将是少数几个能让数据中心体验如同本地计算一般的团队。

Recommend SemiAnalysis to your readers

向您的读者推荐 SemiAnalysis

Bridging the gap between the world's most important industry, semiconductors, and business.

弥合全球最重要产业——半导体——与商业之间的鸿沟。



49 Likes 49 个赞 · 1 Restack 1 次 Restack

Previous 好的，请发送需要翻译的内容

Discussion about this post

关于此帖的讨论

Comments 评论 **Restacks** **转帖**



Write a comment...

© 2026 Dylan Patel · Privacy · Terms · Collection notice
Substack is the home for great culture