arXiv:2601.10463v1 [cs.AR] 15 Jan 2026

XR 工作负载的架构分类：跨层原型及其影响

XINYU SHI、SIMEI YANG 和 Francky Catthoor

增强现实和虚拟现实的边缘和移动平台（统称为扩展现实（XR））必须在严格的功耗和面积约束下，实现确定性的超低延迟性能。然而，XR 工作负载的多样性正在迅速增加，其特征是算子类型异构、数据流结构复杂。这一趋势给以卷积神经网络（CNN）为中心的传统加速器架构带来了重大挑战，导致传统的以计算为中心的优化策略收益递减。尽管这一问题十分重要，但目前仍缺乏对完整 XR 流水线的系统性架构层面的理解。本文采用跨层方法，对 XR 工作负载进行架构分类，将基于模型的高层次设计空间探索（DSE）与商用图形处理器和中央处理器硬件上的实测分析相结合。通过分析涵盖 12 种不同 XR 内核的代表性工作负载，我们将其复杂的架构特征提炼为一小组跨层工作负载原型（例如，容量受限型和开销敏感型）。基于这些原型，我们进一步提炼出关键的架构洞见，并为下一代 XR片上系统提供可行的设计指南。我们的研究表明，为了在未来的 XR系统中实现更高的能效和性能，XR架构设计必须从通用资源扩展转向阶段感知调度和弹性资源分配。

其他关键词 与短语：扩展现实（XR）、设计空间探索（DSE）、工作负载特征分析、存储层次结构、片蠢量扩展。

ACM 引用格式：
XINYU SHI、SIMEI YANG 和 Francky Catthoor。2026。XR 工作负载的架构分类：跨层架构类型及其影响。1，1（2026年1月），22页。https://doi.org/10.1145/nnnnnnn.nnnnnnn

1 引言

近年来，增强现实和虚拟现实（AR/VR）应用的多样性和复杂性不断提升，推动了对能够高效支持这些工作负载的设备平台的需求增长 [1, 2]。尤其是边缘和移动扩展现实（XR）平台，必须在严格的功耗、面积和热预算限制下运行，同时保持确定性、超低端到端延迟和高帧率 [3]。与基于云的人工智能服务不同，边缘和移动XR系统必须持续处理高分辨率传感器数据，并实时在设备端生成沉浸式图形。随着XR软件栈从孤立的视觉内核演变为日益异构的设备端工作负载（融合多个感知和渲染阶段），XR计算图也变得大得多、更多样。这一演变揭示了一项根本性挑战：当面对多种算子类型和数据流时，固定功能、以卷积神经网络为中心的加速器效率会下降。

现有加速器大多针对具有规则张量形状的密集卷积和通用矩阵乘法进行优化，而以卷积神经网络为中心的设计则会专门调整数据流和缓冲，以利用结构化复用（例如 Eyeriss） [4]。先前的研究表明，僵化且高度专用的建图方式（例如脉动阵列式执行）可能会遭受显著的

作者联系方式：Xinyu Shi；Simei Yang；Francky Catthoor。

允许免费制作本作品全部或部分内容的数字或纸质副本，供个人或课堂使用，但不得为营利或商业利益而制作或分发副本，且副本必须包含本声明及首页上的完整引文。对于本作品中由作者以外的其他人拥有版权的部分，必须尊重其版权。注明出处后可摘录。若要以其他方式复制、重新出版、发布至服务器或重新分发至列表，则须事先获得特定许可和/或支付费用。请通过 permissions@acm.org 申请许可。© 2026 版权所有归所有者/作者所有。出版权已授权给 ACM。ACM XXXX-XXXX/2026/1-ART https://doi.org/10.1145/nnnnnnn.nnnnnnn

效率损失，这是由于不利的数据映射和利用率降低，在不规则或稀疏计算模式下尤为明显 [5, 6]。相比之下，扩展现实工作负载通常包含更广泛的运算原语组合，不仅限于卷积/通用矩阵乘法，还包括变换、重采样和特征匹配；这些运算通常会引入数据相关索引、不规则依赖，以及必须显式存储并在各阶段间重复使用的中间张量 [1]。总的来说，这些特性会增加非顺序内存流量并扩大工作集，因此主导瓶颈可能会随阶段而变化——例如，从计算资源利用不足转为带宽压力，继而转为片上容量限制或同步/调度开销。因此，仅提高峰值计算吞吐量或带宽，往往不足以持续提升扩展现实工作负载的性能和能效。

尽管这种软硬件不匹配十分突出，但目前对 XR 工作负载的系统性架构理解仍然有限。以往研究往往聚焦于孤立的内核（例如代价体、姿态估计）、特定任务加速器，或较高层次的算法抽象，而这些研究难以展现微架构行为及映射引发的效应。目前仍缺少全面且以证据为基础的特征分析，将 XR 算子的多样性与具体的架构敏感性联系起来，例如片上存储驻留、内存带宽需求以及控制或同步开销，从而使架构师能够摆脱以 CNN 为中心的假设，深入理解 XR系统。

在本研究中，我们通过提出一种 XR 工作负载的架构分类来弥补这一不足。我们采用跨层方法，将商用硬件上的实测分析与基于模型的解析式设计空间探索（DSE）相结合。在性能分析方面，我们结合 GPU 内核级性能分析与 CPU 指令集架构级插桩，以捕捉算子级利用率和端到端应用行为；仅凭 GPU 内核无法推断出这些行为，例如控制流强度和内存访问规律性。在建模方面，我们开发了一种轻量级解析式 DSE 框架，用于量化能耗–延迟权衡如何随片上内存容量以及一系列具有代表性的映射选择而变化，其中包括分块、缓冲和内核融合。为支持对异构算子进行一致分析，我们采用一种可重构的近存单指令多数据风格计算模板作为解析式抽象，而非具体的微架构方案。通过将性能分析指标与解析式扫描相互印证，我们发现，在系统性改变片上容量和映射方式时，瓶颈会反复发生转变——从计算受限转为带宽受限，再从带宽受限转为容量受限或开销受限。这些转变定义了一小组跨层工作负载原型，揭示了 XR 工作负载中常见的性能限制机制。

本工作的主要贡献如下。

- **全面的架构特征分析**。我们深入分析了 12 种具有代表性的 XR 工作负载，并通过实证剖析和解析式设计空间探索，量化了它们对计算吞吐量、存储层次结构行为及执行开销的敏感性。
- **跨层工作负载原型的识别**。我们将 XR 工作负载共有的重复行为模式形式化，从而提供一种精简的抽象方式，用于分析以 CNN 为中心的模型之外的架构瓶颈。
- **可操作的架构洞见**。我们将这些原型转化为面向 XR 的加速器和片上系统的具体设计启示，指出性能受限于片上容量、带宽效率、计算吞吐量的情形，以及——对于不规则工作负载而言尤为关键的——受同步、索引和阶段级开销限制的情形。

本文其余部分安排如下。第 2 节回顾扩展现实加速器、扩展现实片上系统平台和系统级设计空间探索的相关工作。第 3 节介绍工作负载套件及其算法特性。第 4 节介绍我们的解析建模与映射方法。第 5 节报告设计空间探索结果及跨工作负载能耗–延迟趋势。第 6 节详细报告中央处理器和图形处理器平台的性能分析。最后，第 7 节总结归纳出的工作负载原型，并提炼其架构启示。

2 相关工作

扩展现实加速领域的先前工作大致可分为两个方向：（1）**硬件设计**，涵盖特定任务加速器和可扩展片上系统平台；（2）**系统级方法**，包括设计空间探索、基准测试和以内存为中心的架构。本节回顾这两个方向上的代表性工作，并指出其方法论上的不足，尤其是在工作负载多样性、映射方法和多粒度证据方面的不足，然后介绍我们以工作负载为驱动、感知映射的特征分析。

2.1 XR 加速器与片上系统平台

当前最先进的面向 XR 应用的硬件研究大多仍然是任务或模型专用的，针对特定的内核族进行优化，例如光流、深度/几何估计、同步定位与建图、视频处理或神经三维重建。在这些领域中，带宽和工作集驻留往往主导能耗–延迟权衡，推动设计采用结构化数据流、积极的片上缓冲、算子融合和量化，以摊销片外流量。代表性实例包括面向光流和深度估计、基于现场可编程门阵列/专用集成电路平台的流水线代价体加速器 [7, 8], ，以及面向 NeRF 的设计；后者协同优化采样/编码、网络推理和体渲染，以减少中间张量和片外往返传输 [9, 10]。除固定功能流水线之外，算子级协同设计研究还表明，重构计算顺序和数据流能够为代价体、注意力和光线步进等反复出现的结构模式释放显著的效率增益 [11–13]。总体而言，这些研究在其目标内核上展现出很高的效率，但其映射见解和资源配置策略往往难以迁移到异构 XR 工作负载，因为后者结合了多种算子类型和执行模式。

为支持单一算子族以外的更多场景，近期的平台级 XR 片上系统和加速器原型在共享存储层次结构下集成了异构计算引擎 [14–17]。这些系统结合了 ISP/DSP 前端、神经网络处理器以及单指令多数据/向量通路，以兼顾领域专用效率与平台可扩展性；其中一些系统还探索了混合精度或基于Posit的执行，以缓解带宽压力，同时确保多个感知前端任务的准确性 [17]。然而，这些平台通常仅在有限数量的代表性工作负载上进行演示，对于由神经渲染和三维重建/几何处理流水线主导的新兴 XR 工作负载，其覆盖范围和建图讨论都相对有限。因此，我们需要一种工作负载驱动、兼顾建图的视角，以便推广至多样化的 XR 算子族。

2.2 系统级设计空间探索、基准测试与以内存为中心的方向

除了加速器微架构之外，越来越多的研究从系统级视角探讨 XR 优化，借助设计空间探索（DSE）和硬件–算法协同设计，在严格的片上存储预算下分析能耗与延迟之间的权衡。代表性工作包括：面向 SoC 的存内计算基准测试 [18] ，用于量化面向 XR 工作负载的端到端权衡；结合神经架构搜索的异构神经网络处理器–存内计算设计 [19] ，用于在片上资源受限的条件下划分模型；以及近传感器推理，将早期特征提取任务卸载到传感器节点，以降低系统能耗 [20]。尽管这些研究为划分点、缓存容量配置和模型适配提供了宝贵见解，但它们通常只关注少量典型模型，和/或假设特定的架构基础平台与划分模板，因此难以将结论推广至更广泛且快速演进的扩展现实工作负载谱。

作为这些方法驱动型研究的补充，近期的 XR 架构研究日益关注以内存为中心的解决方案。最值得注意的是，三维（3D）堆叠式存储器/逻辑集成 [21]，旨在直接解决带宽和数据驻留瓶颈。通过显式建模垂直带宽和内存邻近性，这些研究表明，将计算移至更靠近内存的位置，可以显著

减少片外流量，与传统二维架构相比，带来显著的加速效果和能耗节省。然而，这些优势通常仅在有限的一组工作负载上得到验证，而且主要依赖建模层面的证据，对多粒度硬件剖析的采用较少，也鲜有讨论异构扩展现实工作负载的映射方法。

与此同时，系统级基准测试和运行时间研究 [22, 23] 在真实的延迟约束下提供端到端特征分析，揭示了单独的内核分析可能掩盖的瓶颈。然而，这些工作主要由场景或应用驱动，通常难以提供面向架构的指导，帮助理解算子族行为、存储层次敏感性或映射策略效应。这一差距凸显了开展面向架构的特征分析的必要性，以便将系统级约束与算子级和映射级行为联系起来。

总而言之，先前的工作通过硬件设计（从特定任务加速器到可扩展的 XR片上系统）和系统级方法（设计空间探索/协同设计、基准测试以及以内存为中心的方向）推动了扩展现实优化。然而，许多研究要么只关注狭窄的工作负载子集，要么缺乏一种统一的映射感知方法，无法推广至异构扩展现实工作负载。这一差距促使我们开展工作负载驱动的架构分类，通过一致的剖析和分析性探索，将工作负载多样性、多粒度行为与以内存为中心的权衡联系起来。

3 应用领域特征

现代扩展现实工作负载涵盖多种视觉任务，包括深度估计、光流/立体匹配、同步定位与建图以及神经渲染。许多扩展现实模型包含大量稠密张量计算，但也高度依赖不规则原语和迭代几何运算，这些运算与传统的以卷积神经网络为中心的推理有着根本区别。典型示例包括代价体构建、变形/重采样、非局部注意力与匹配、体积光线步进以及点云配准。这些算子带来了各异的局部性和工作集行为，对存储层次结构造成了不同于标准滑动窗口复用的压力。

我们评估了一套包含 12 种具有代表性的扩展现实工作负载，这些工作负载的选取旨在：(1) 覆盖实际扩展现实软件栈中常见的关键阶段；(2) 涵盖多种结构模式（卷积神经网络编码器–解码器、基于 Transformer/图神经网络的匹配器、基于多层感知机的神经渲染以及混合流水线）；(3) 体现其在实际边缘部署中的相关性。表 1 总结了这套工作负载，并报告了在统一批次=1 设置下各工作负载的规模指标。此处，权重表示模型参数（或适用情况下的持久状态），激活值 表示我们执行设置下每帧的激活占用，而*GFLOPs* 表示每帧浮点运算次数。对于 Cupoch ICP 等非深度神经网络组件，这些以张量为中心的指标并不直接适用，因此不予列出。从架构角度来看，我们并非仅按算法功能对工作负载分组，而是将其归纳为具有不同数据流模式和工作集特征的结构类别。

编码器–解码器卷积神经网络。MonoDepth2 [24] 和 HR-Depth [25] 等模型采用 U-Net 风格架构，并使用多尺度特征金字塔和跳跃连接。与分类骨干网络（如 ResNet）相比，解码器路径会保留高分辨率的中间特征图，因此激活工作集要大得多。因此，性能和能耗往往对片上容量以及将关键中间结果保留在片上以减少片外流量的能力较为敏感。

代价体与基于变形的模型。 PWC-Net [26],RAFT/RAFT-Stereo [27], 等光流和立体视觉工作负载，以及 TartanVO [28] 等混合流水线，会显式构建相关性体积/代价体积，并通过变形或重采样实现特征对齐。代价体构建需要在搜索窗口内进行大量聚集访问，而变形则执行基于坐标的采样。这些步骤会破坏标准卷积所具有的规则空间局部性，并可能限制传统映射策略下的数据重用实现。

表 1. 工作负载套件及结构特征概览

网络	网络描述	权重（MB）	激活（MB）	GFLOPs
深度估计				
Monodepth2	ResNet 编码器–解码器（U-Net 风格）	53.87	69.42	11.73
HR-Depth	编码器–解码器卷积神经网络（多尺度）	53.64	409.81	22.43
光流与立体匹配				
PWC-Net	金字塔卷积神经网络 + 代价体	35.76	1833.46	169.48
RAFT-Stereo	循环卷积神经网络 + 3D 相关性体	42.27	618.05	84.44
SLAM 前端：特征匹配与跟踪				
LightGlue	卷积神经网络骨干网络 + Transformer 匹配器	12.21	954.02	90.44
SuperGlue	图神经网络匹配器	11.82	996.33	104.33
LoFTR	无检测器 Transformer 匹配器	87.98	5105.88	427.96
SLAM 后端与建图				
TartanVO	PWC-Net 匹配器 + ResNet50 位姿头	19.96	1177.81	30.94
Cupoch ICP	基于 CUDA 的点云 ICP 流水线	–	–	–
神经渲染（*NeRF* 风格的多层感知机）				
NeRF（以及 TinyNeRF）	全连接多层感知机 + 光线步进	1239.42	11318	813.54
视觉骨干网络				
ViT	视觉 Transformer 编码器（分类器）	25.63	1486.87	18.20

特征匹配网络（基于 Transformer/图神经网络）。 LightGlue [29], LoFTR [30], 和 SuperGlue 式匹配器 [31] 将 CNN 特征与基于 Transformer 或图的匹配模块相结合，而 ViT [32]则代表纯粹的基于 token 的骨干网络。分块、自注意力/交叉注意力和图消息传递等算子需要维护多个中间张量（例如查询/键/值或节点/边状态），并且往往会导致非局部访问模式。因此，它们的瓶颈通常取决于工作集驻留和内存传输，而非类似卷积的数据重用。

神经体渲染（NeRF 风格）。NeRF [33] 和 TinyNeRF [34] 代表由光线步进循环与小型多层感知机计算构成的神经图形工作负载。虽然多层感知机本身通常较为紧凑，但整体执行主要受海量逐光线样本计算和中间累积所支配，这可能产生以光线为主导、计算和内存行为随阶段变化的激活流。这种结构可能使计算强度与权重带宽相互脱钩，并使性能对采样策略和逐光线状态的片上缓冲十分敏感。

迭代几何求解器。Cupoch ICP [35] 等工作负载依赖迭代优化（例如迭代最近点），每次迭代都会执行最近邻搜索、刚体变换估计和误差评估。从架构角度看，这些流水线具有复杂的控制流、稀疏点云运算和小规模线性代数运算，而且通常受延迟限制，而非受吞吐量限制。因此，除了原始计算能力之外，它们的性能还受到非规则内存访问、同步以及分支/控制开销的影响。

4 架构建模与高级参数化设计空间探索框架

在架构级评估扩展现实工作负载，需要分析每个应用对计算引擎和存储层次结构造成的压力，尤其是片外带宽需求、片上容量压力和计算吞吐量需求。扩展现实工作负载在结构上具有异构性：其算子和图拓扑高度多样，主导瓶颈也常随阶段变化（例如，从计算受限转为带宽受限或容量受限）。这种多样性使得通过临时性的、针对特定算子的分析来比较应用或提取可推广的趋势变得困难。为实现系统化且可比较的跨工作负载评估，我们开发了一种轻量级、基于 C++ 的设计空间探索框架——一种快速的分析型设计空间探索模型，

该模型无需寄存器传输级或周期精确模拟，即可捕捉紧张的片上存储预算下的关键能耗–延迟权衡。该框架旨在涵盖扩展现实工作负载中常见的多种可向量化算子模式，从而对异构图进行一致的成本核算。我们的目标并非声称能够准确预测周期级表现；相反，我们提供一种可解释、层次感知的成本模型，支持在庞大的设计空间中快速扫描，并揭示各工作负载对片上容量、内存带宽和映射选择的结构敏感性。这一能力对于识别瓶颈临界点、解释收益递减，以及在不同扩展现实网络之间生成可比较指标至关重要。本节其余部分将详细介绍所提出的设计空间探索模型的设计原理和关键组件。

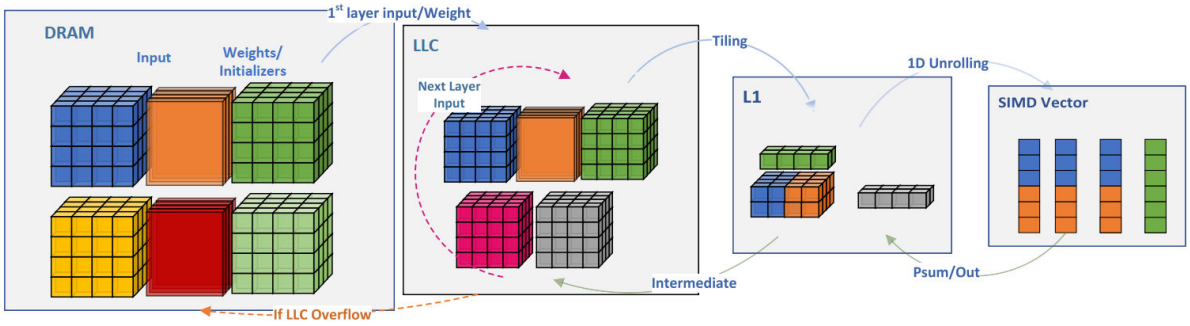


图 1. 近存 SIMD 建模模板数据流概览

4.1 架构模板概览

为了在一致的执行基底上进行设计空间探索建模，我们采用单核 SIMD 向量处理器作为基线架构模板（灵感源自低功耗向量架构[36][37][38]），并搭配可配置的多级片上存储层次结构。我们选择 SIMD 风格模板，主要是为了在各类 XR 算子类别之间实现建模的通用性和映射的一致性。XR 流水线并非仅由密集型 MAC 核构成；其中还包含许多难以用僵化的脉动阵列式数据流表达的可向量化基元。在实践中，要在纯 MAC TPU 上支持此类基元，往往需要额外的 DSP/SIMD 辅助单元，并针对各阶段进行算子专用数据流定制，这会使映射策略更加复杂，也会模糊本文的目标：通过统一评估模型，在受控的片上存储预算下揭示架构敏感性。重要的是，该模板是用于 Fast-DSE 的分析工具，并非声称这是唯一最优的 XR 加速器设计。

SIMD 向量处理器模块：计算核心是一种宽 SIMD 向量处理器，具有可配置数量的通道（默认值：16）。向量寄存器文件（VRF）的宽度为 512 位，并支持多种数据类型（例如 FP32/FP16/INT8）。该核心假定提供常见的、与扩展现实相关的向量基本操作，包括融合乘加、逐元素算术运算、比较/最小–最大运算、归约运算以及数据移动操作（例如置换和收集/散射式访问）。这种设计既适用于计算密集型的卷积/通用矩阵乘法阶段，也适用于扩展现实网络中常见的变换/重采样/匹配密集型阶段，同时在各类算子间保持统一的执行模型。

L1 片上存储器：L1 片上存储器充当软件管理的暂存存储器，由映射器显式控制，并具备可配置的容量。L1 主要用作内层分块工作缓冲区，支持细粒度的数据移动、分区以及面向大型算子的灵活张量分块。它能够改善热点张量切片和短生命周期中间结果的局部性，同时也是一个关键设计参数，决定算子能否利用内层循环复用而无需向更低层级溢出。

末级缓存复用缓冲区（软件管理）：片上末级存储提供 16–64 MB 的容量，连接 L1 暂存存储器与动态随机存取存储器。在本文中，末级缓存被建模为软件可见的数据重用缓冲区，而不是由硬件管理的缓存。其主要作用是容纳超出 L1 容量的大型中间结果或权重，从而基于容量驻留为数据重用提供一个分析上限。将末级缓存视为容量受限的缓冲区后，我们的设计空间探索便可抽象掉具体的硬件替换策略（例如最近最少使用），专注于研究容量分配与管理如何影响 DRAM 流量和整体能效。

DRAM主存：DRAM 是存储层级中最低层的存储，其访问延迟和能耗远高于片上存储器，但对于存储大型特征图和模型参数而言不可或缺。我们假设所有预训练权重和网络输入最初都存储在 DRAM 中。DRAM 参数（能耗、带宽和延迟）取自 Ramulator[39] 和 DDR4 标准 DRAMPower[40] 数据表，以确保与主流技术假设一致。

4.2 数据流与建图策略

为确保跨工作负载可比性，我们的框架采用统一且保守的映射策略。我们不采用针对特定网络的调度或布局优化，而是将映射选择限制在一组精简且通用的规则内（驻留方式选择、一级缓存分块和有限的分块内融合）。这样一来，后续的（$L1$，LLC）扫描主要反映工作负载本身对片上容量和片外流量的敏感性，而非针对各模型的手动调优。建模流程从每个网络的中间表示（IR）开始。对于每个算子，我们解析其算子类型、张量形状和常量初始化器（例如权重和查找表），并构建生产者–消费者依赖有向无环图。随后，我们通过拓扑遍历确定一个确定性的执行顺序，作为张量存活性分析和层级流量核算的通用调度。

我们的基线层次结构为 DRAM $\rightarrow$ LLC $\rightarrow$ L1 $\rightarrow$ SIMD，其结构类似于先前建图框架（例如 [41]）中用于推理数据移动和局部性的显式存储层次结构抽象。与采用隐式替换机制的硬件缓存不同，我们将 LLC 视为由软件管理、容量受限的复用缓冲区，以便进行可控的驻留分析。具体而言，我们沿固定执行顺序，根据张量存活性，在容量约束下近似理想的保留策略：张量在生成时变为存活状态，并在其最后一个使用者完成使用后释放；当存活数据占用量超过 LLC 容量时，张量便会溢出到动态随机存取存储器。这一抽象消除了冲突与替换动态，从而为分析容量瓶颈提供稳定、由容量驱动的视角。

为使模型适用于大规模参数扫描并保持轻量，我们采用线性化能耗核算方案。存储能耗根据传输字节数计算，读写系数分别取自 CACTI 类归一化静态随机存取存储器/动态随机存取存储器功耗、性能和面积表。计算能耗则根据归一化单位操作能耗估算，相关数据取自同一工艺与功耗、性能和面积库（例如等效于乘加运算或单指令多数据操作的能耗）；系数则根据计算模板和操作类别选取。

图 2 总结了我们的统一建图优化流程，该流程由三个部分组成：L1 分块、驻留方式选择和有限的卷积核融合。我们对所有工作负载采用相同的分块与驻留方式选择流程，以保持跨网络可比性。如图 2 所示，L1–单指令多数据交互决定候选分块能否容纳在有效的 L1 容量预算内；具体的分块搜索过程将在第 4.3 节中介绍。下面介绍两个与分块配套、并在所有工作负载中一致应用的轻量级映射规则。

基于占用量的驻留方式选择 对于卷积/类 GEMM 算子，我们使用简单的权重占用量启发式，在输出驻留（OS）和权重驻留（WS）之间进行选择。核心理由是，驻留方式的选择不应影响后续 L1 分块的可行性和质量：L1 必须同时容纳至少一个输入分块、一个输出/部分和分块，以及（若采用 WS）一个权重分块。因此，我们

使用层的权重占用量 B_w 作为保守指标，判断权重能否驻留在 L1 中而不挤占激活分块的空间。具体而言，当 $B_w < \rho \cdot L1_{\text{eff}}$（所有实验均采用固定的 $\rho = 0.5$）时，我们选择 WS，从而为输入/输出分块及相关管理预留剩余的 L1 容量。在此条件下，让权重保持驻留可最大限度地提高内层计算循环中的权重数据重用，并减少权重在 L1↔与 LLC 之间的重复传输。反之，当 Bw 相对于 L1 较大时，我们选择 OS，使 L1 容量优先用于输出/部分和累加及激活分块，避免出现不可行或需要过度切分数据块的病态分块方案。在我们的框架中，此驻留方式决策仅影响数据重用假设和层级间流量核算，而具体的数据块可行性与形状则由第 4.3 节中的统一 L1 分块流程决定。

保持合法性的局部融合。 为减少常见扩展现实算子链中可避免的一级缓存和末级缓存往返，我们采用了一组精简的局部融合规则。融合仅限于简单的、局限于分块内的生产者–消费者模式：消费者可以立即在生产者的分块上执行，且无需改变张量形状或引入额外的大型活动缓冲区。具体而言，当两个算子具有相同的分块形状，且融合执行不会增加活动张量的数量，也不需要在一个分块内物化额外的多输入中间结果时，我们会融合 (i) 卷积/通用矩阵乘法后接一元激活函数，以及 (ii) 逐元素加法后接激活函数。我们明确避免跨越重塑、拼接或其他布局/物化边界进行融合，因为此类算子通常会破坏分块对齐，并使驻留情况的核算复杂化。

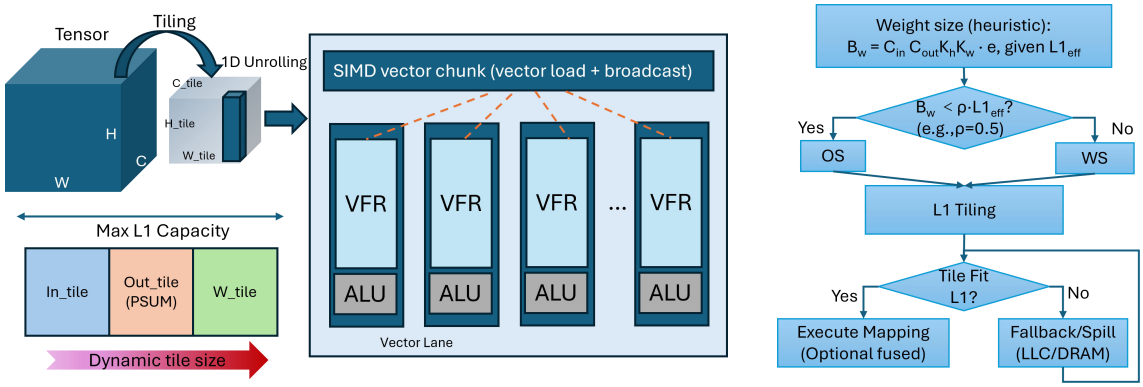


图 2. SIMDengine 上一级缓存驻留分块与驻留式执行流程（卷积层示例）。

4.3 分块自动化

我们进一步详细介绍映射器中使用的一级缓存分块自动化。为保持跨网络可比性，我们采用统一且保守的映射策略；在此策略下，大型张量乘加算子（例如卷积和通用矩阵乘法/矩阵乘法）通常无法直接映射，因为输入、权重和输出的组合工作集可能超过一级缓存的有效容量。因此，我们引入一种自动化的一级缓存分块机制：给定一级缓存容量预算后，该机制 (i) 保证映射合法性，并 (ii) 通过与工作负载无关的流程搜索高质量的分块配置。受篇幅限制，我们以卷积分块作为代表性案例进行说明；同一机制也可扩展至通用矩阵乘法/矩阵乘法，采用类似的分块变量和相同的可行性测试。图 3(a) 总结了分块变量和一级缓存可行性约束，而图 3(b)（算法 1）展示了搜索流程。

要使卷积分块在我们的一级缓存–SIMD 模板中执行，一级缓存必须同时容纳三个占用量：(i) 输入激活分块，(ii) 对应的权重分块，以及 (iii) 输出/部分和 (PSUM) 分块。如图 3(a) 所示，我们通过沿通道维度和输出空间维度进行分块，来参数化候选分块方案

(a) L1 卷积分块摘要

（卷积）平铺变量。$\mathbf{t}_{\text{conv}} = (c^t{}_{in}, c^t_{out}\ h^t_{out}, w^t_{out})$.
派生维度和一级缓存约束。

$$H_{out} = \left\lfloor \frac{H_{in} - K_h + 2\,pad}{stride} \right\rfloor + 1,$$

$$W_{out} = \left\lfloor \frac{W_{in} - K_w + 2\,pad}{stride} \right\rfloor + 1,$$

$$H^{in}_{tile} = \text{分钟}(H_{in},\ stride(h^t_{out} - 1)\ + K_h + 2\,pad),$$

$$W^{in}_{tile} = \text{分钟}(W_{in},\ stride(w^t_{out} - 1)\ + K_w + 2\,pad),$$

$$S_{in} = c^t_{in}\, H^{in}_{tile}\, W^{in}_{tile},$$

$$S_{ker} = c^t_{in}\, c^t_{out}\, K_h K_w, \qquad S_{out} = c^t_{out}\, h^t_{out}\, w^t_{out},$$

$$(S_{in} + S_{ker} + S_{out}) \le L1_{\text{eff}}, \qquad e = 4\text{B}。$$

(b) 算法 1：用于 L1卷积平铺的模拟退火

1: **输入：** $(H_{in},\ W_{in},\ K_h,\ K_w,\ stride,\ pad)$，$L1_{\text{eff}}$，$e$
2: **初始化：** $x \leftarrow (c^t_{in},\ c^t{}_{out}\ h^t_{out}, w^t_{out})$；$T \leftarrow T_0$；$f \leftarrow$成本（$x$）；$x^\star \leftarrow x$
3: **当** $T > T_{\min}$ **时执行**
4: **对于** $i = 1$ **到** L **执行**
5: $y \leftarrow x$；选择 $d \in \{c^t_{in},\ c^t_{out},\ h^t_{out},\ w^t_{out}\}$；$y_d \leftarrow$ 钳制$(y_d \pm \delta)$
6: **如果**非法(y) **则** $g \leftarrow +\infty$ **否则** $g \leftarrow$ 开销(y)
7: $\Delta \leftarrow g - f$
8: **如果** $\Delta \le 0$**或**随机() $< e^{-\Delta/T}$ **则**
9: $x \leftarrow y$；$f \leftarrow g$；**如果** $f <$ 开销$(x^\star)$ **则** $x^\star \leftarrow x$
10: **结束循环**；$T \leftarrow \alpha_T T$
11: **结束 while 循环**；**返回** $x^\star$

图 3. 卷积层的 L1 自动分块。(a) 卷积数据块变量以及输入/权重/输出占用量对应的 L1 容量约束。(b) 在该约束下探索可行数据块并返回低成本分块方案的模拟退火搜索。所有模拟退火超参数（T_0、$T_{\min}$、α_T、L 和 δ）在所有工作负载中均为全局固定值。

。重要的是，所需的输入分块占用量并非独立选择：它由卷积感受野（步幅/卷积核/填充）通过输出分块决定，因此也包含了光环/边界扩展。只有当这三个数据块的总占用量不超过有效的一级缓存容量预算时，分块方案才是合法的（图 3(a)）；否则，该建图将需要溢出，无法作为驻留于一级缓存的分块执行。

在合法的分块方案中，我们使用一个轻量级代理目标函数对候选方案进行排序：

$$J(t) = \alpha \cdot \text{Compute}(t) + \beta \cdot \text{Bytes}(t) + \gamma \cdot N_{\text{tiles}}(t). \tag{1}$$

这里，Compute(t) 和 Bytes(t) 采用第 4.2 节所述的层级记账方法，分别作为所选分块下算术运算量和 L1 边界流量的代理指标。该目标函数的构造遵循 Halide 自动调度器 [42, 43], 等调度框架的设计理念：结合计算和内存访问估算来指导调度与映射决策；这种方法已广泛应用于张量程序调度器和加速器映射框架。N_tiles(t) 项根据分块所隐含的数据块数量推导得出，用于对过度碎片化的分块方案进行正则化，因为正如先前的映射研究所述，这类方案往往会增加每个数据块的控制/设置开销，并加剧边界/光环冗余[41]。权重（α，β，γ）在所有工作负载和容量点上均为全局固定值；我们对其进行一次缩放，使三个项对于一个具有代表性的可行分块具有相近的量级，且不会针对每个工作负载单独调优。

我们采用模拟退火（SA）作为轻量级元启发式，在固定评估预算下搜索离散容量约束分块空间，这符合编译器驱动和自动调优驱动的设计空间探索中的常见做法（例如，TVM 风格的调度搜索[44]）。算法 1（图 3(b)）概述了映射器所用的分块搜索过程。对于给定的层和 $L1_{\text{eff}}$，分块状态为 $x = (c^t_{\text{in}},\ c^t_{\text{out}},\ h^t_{\text{out}},\ w^t_{\text{out}})$。我们从一个显然可行的状态开始进行模拟退火：逐步缩小分块因子，直到满足图 3(a) 中的 L1 可行性约束。随后，模拟退火通过每次移动扰动一个分块维度来探索相邻分块方案；违反 L1 约束的候选方案视为非法（代价为无穷大），而可行候选方案则按 $J(\cdot)$ 评分，并根据固定降温计划下的标准模拟退火准则决定是否接受。在搜索预算内找到的最佳可行分块方案 x^* 将作为结果返回。

对于每个卷积/类 GEMM 算子以及扫描中的每个一级缓存容量点，分块器都会在固定 SA 预算下返回找到的最佳可行分块。随后将选定的分块传递给模型，以便对不同工作负载进行一致的流量与能耗核算。

5 设计空间探索结果分析

本节使用第 4 节中的快速解析模型，研究片上存储容量如何影响扩展现实工作负载的能耗和存储器侧延迟表现。我们的片上 SRAM 能耗/面积假设以公开报道的技术数据和缩放趋势为依据 [45–47]。对于每个网络，我们将一级缓存暂存器容量从 16 KB 扫描至 256 KB，并将末级缓存容量从 16MB 扫描至 64 MB，涵盖从小型基准配置到在该技术/面积基准下认为可行的最大容量配置。[1]

本节始终将动态随机存取存储器容量固定为 8 GB。在批次为 1 的推理中，该容量足以容纳所有受评估工作负载的工作集，因此，我们的结果不会受到单纯由动态随机存取存储器容量不足所引发的次级效应影响，例如占用空间溢出。因此，动态随机存取存储器能耗和存储器侧延迟的变化主要反映片外流量和访问行为的差异，而非地址空间容量的影响。这样的设置使我们能够在统一的容量假设下，对不同应用的层级行为进行受控比较。

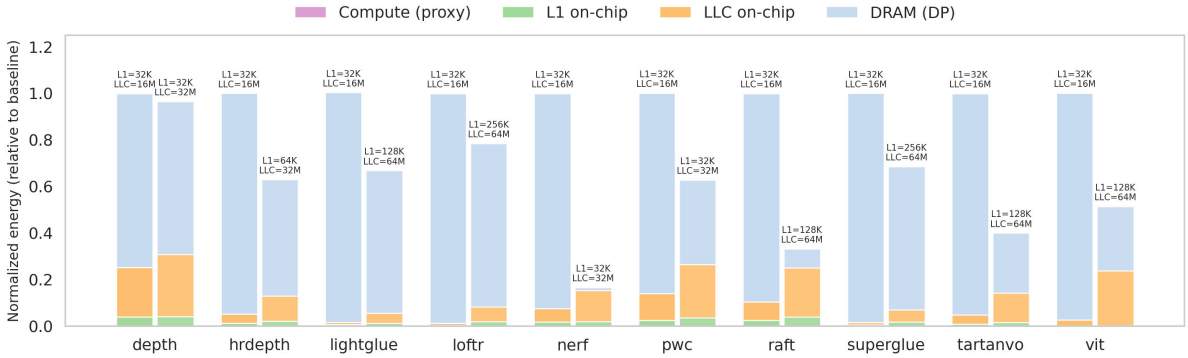
对于总能耗，我们报告一级缓存能耗、末级缓存能耗、动态随机存取存储器能耗和向量核心能耗之和，定义见第 4 节。对于延迟，由于解析模型无法在异构时钟域下实现周期精确建模，我们不会将片上周期数解读为绝对时间。相反，我们采用屋顶线式延迟代理指标，并将存储器侧延迟定义为片上服务时间与 DRAM 服务时间中的较大值，依据相应的流量和服务带宽计算得出。该代理指标仅用作相对指标，用于比较每个工作负载中的设计点。它反映了实际情况：一旦片外流量不可忽略，DRAM 服务时间就可能主导有效存储器侧停顿部分。

5.1 随一级缓存/末级缓存容量变化的存储器侧能耗

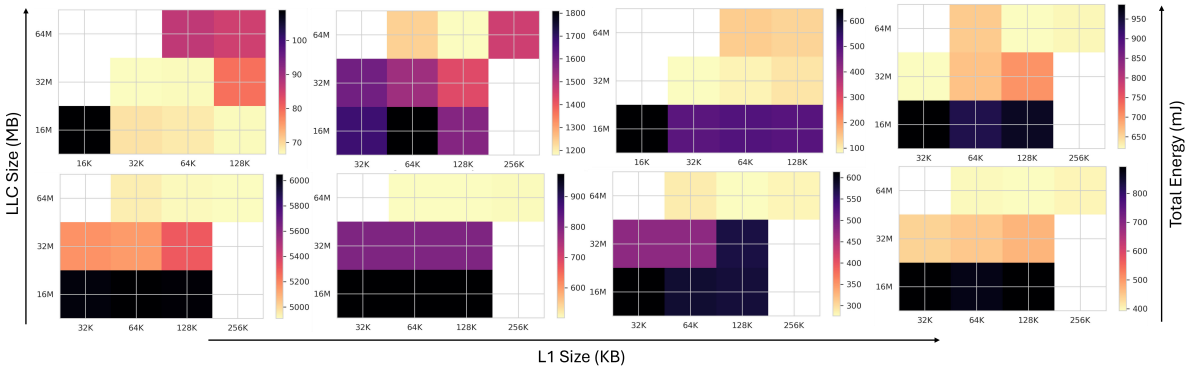
图 4 总结了容量扫描对 XR 套件中代表性工作负载的能耗影响。[2] 图 4(a) 展示了基线配置（32 KB 一级缓存和 16 MB 末级缓存）以及各应用能效最高的配置下的能耗分解。该对比突显了基线配置下占主导的存储器能耗组成部分，以及转向最佳设计点后哪些部分有所降低。图 4(b) 以驻留式热力图呈现了所评估的一级缓存和末级缓存网格上的二维能耗分布，颜色表示总能耗（颜色越深，能耗越高）。每个单元格表示特定配置点下的归一化总能耗。在该视图中，相邻单元格之间的明显变化表示工作集驻留状态发生转变，即部分重复使用的中间数据转为片上驻留；而相对均匀的区域则表明继续扩容带来的边际收益递减。

在我们的能耗模型中，对于大多数工作负载，基线配置下动态随机存取存储器是内存能耗的主要来源，其次是末级缓存，然后是一级缓存。将基线能耗分解与最佳情况配置进行比较后，我们发现，增大末级缓存容量通常会使相当一部分内存能耗从动态随机存取存储器转移到末级缓存，因为更多中间张量能够驻留在片上。对于有效复用占用空间在数十兆字节以内的网络，这种转移在最优配置下可将动态随机存取存储器能耗降至可忽略的比例。相比之下，对于数据量较大的工作负载，即使末级缓存容量达到我们测试的最大值，动态随机存取存储器仍是重要的能耗组成部分，这表明在我们的映射策略下，它们的有效复用占用空间超出了可用的片上存储容量。

[1]在面向部署的场景中，不同网络通常会配置不同的片上存储器容量。例如，数据量大或中间数据占用大的工作负载可能需要更大的存储器配置，而占用较小的工作负载则可以用较小的配置高效运行。在此，我们有意采用统一的容量网格，以便开展受控的敏感性研究并进行跨工作负载比较，而非针对每个网络单独调优。[2]我们的设计空间探索聚焦于能够在基于分块的 SIMD/张量抽象下保持一致表示的卷积核，从而实现受控的跨工作负载比较。点云 ICP 流水线不纳入设计空间探索，因为其稀疏且控制流密集的内核（例如最近邻搜索）无法由该抽象表示；我们仍会在第 6 节通过硬件剖析对其进行分析。省略 TinyNeRF 是因为它采用与 NeRF 相同的 NeRF 风格内核结构，并未呈现独特的容量敏感性状态。



（a）能耗分解（全局）



（b）一级缓存与末级缓存的二维能耗热力图（从左上到右下依次为：Monodepth2、LightGlue、NeRF、PWCNet、LoFTR、视觉 Transformer、RAFT-Stereo、TartanVO。）

图 4. 能耗与片上容量：各工作负载的能耗分解及一级缓存—末级缓存容量。

接下来，我们考察图 4(b) 中沿一级缓存维度的敏感性。总体而言，一级缓存容量的影响在很大程度上取决于末级缓存的大小。当末级缓存较小（16 MB）时，对于大多数工作负载，一级缓存容量对总能耗的影响有限，因为动态随机存取存储器流量主导了能耗预算。Monodepth2 是一个显著的例外：其中间数据占用相对较小，因此即使末级缓存较小，也能捕获更大比例的数据重用，使片上缓冲的剩余敏感性更明显。当末级缓存增至 32 MB 及以上时，一级缓存的表现因工作负载而异。一些工作负载（如 LoFTR 和视觉 Transformer）对一级缓存容量基本不敏感，因为其主要数据重用由末级缓存驻留实现，而一级缓存主要缓冲生命周期较短的数据块。其他工作负载则呈现轻微的非单调变化，其能耗最优的 L1 容量取决于末级缓存容量。当末级缓存较小时，增大一级缓存可能会暂时增加总能耗，因为较大的数据块可能会使部分和及中间片段驻留更长时间，从而增加片上访问开销，同时动态随机存取存储器流量仍然较高。此外，在能耗模型中，较大的一级缓存容量会增加访问开销和漏电开销。一旦末级缓存足够大，能够容纳关键的中间张量，同样的一级缓存扩容就能减少冗余重复读取并降低总能耗。这种扩容至适配的效应源于固定映射策略下离散的分块选择。对于大多数工作负载，沿末级缓存维度的热力图显示出比沿一级缓存维度更系统的敏感性：在一级缓存固定时，增大末级缓存通常会使配置从高能耗区域转向低能耗区域，而这种

这一趋势在不同一级缓存配置下总体一致。一个常见特征是，在 16 MB 与 32 MB 末级缓存配置之间出现明显的过渡区间，这表明一部分重要的重复使用的中间数据开始驻留在芯片上，从而避免反复从动态随机存取存储器中重新读取。超过这一区间后，表现更依赖于具体工作负载。对于某些工作负载（例如 NeRF 和 PWC-Net），可转化的数据重用大多已在 32 MB 配置左右实现，因此进一步增大末级缓存的收益递减。另一些工作负载（例如 LoFTR、视觉 Transformer、RAFT-Stereo 和 TartanVO）在 32 MB 配置左右仍会频繁访问动态随机存取存储器，只有接近 64 MB 配置时才进入能耗较低的区域；这与其同时存活的中间结果总体占用空间更大相符。基于这些观察，我们将工作负载归纳为三种反复出现的容量响应类型：(1)较早趋于饱和／收益递减型工作负载（例如 Monodepth2、HRDepth 和基于多层感知机的NeRF）会迅速进入热力图中相对均匀的区域：适度配置末级缓存即可实现大部分可转化的数据重用，进一步扩展一级缓存／末级缓存只能有限地降低能耗。(2)容量门控、由末级缓存驱动动态随机存取存储器访问减少型工作负载（例如 RAFT-Stereo、PWC-Net、视觉 Transformer 和 TartanVO）沿末级缓存轴呈现明显的过渡区间：一旦末级缓存跨过工作集边界，大量动态随机存取存储器能耗便会被末级缓存能耗取代，而一级缓存的影响较弱，或只有在末级缓存足够大时才有益。(3) 动态随机存取存储器访问持续活跃／数据重用实现差距型工作负载（例如 LightGlue、SuperGlue 和 LoFTR）在所探索的配置范围内始终以动态随机存取存储器访问为主导：尽管总能耗会因片上访问开销和离散分块效应而随一级缓存／末级缓存配置变化，但仅靠被动扩展缓存，并不能像第 (2) 组那样显著降低动态随机存取存储器能耗，这表明在观察到的执行结构下，数据重用难以实现。

5.2 能耗—延迟交互与帕累托权衡

第 5.1 节表明，增加片上容量（一级缓存/末级缓存）可以减少 DRAM 流量，从而提高能效。现在我们将问题扩展到性能方面：通过增加容量来减少 DRAM 流量时，存储器侧延迟是否也会相应改善？哪些工作负载呈现出能耗–延迟权衡，不同工作负载是否表现出不同的轨迹形态？图 5 绘出了每种缓存配置下的总能耗与存储器侧延迟（定义为屋顶线式的片上服务时间和 DRAM 服务时间的最大值）。两个坐标轴均按工作负载进行归一化，因此我们关注的不是不同工作负载之间的绝对值，而是每种工作负载内部的轨迹和帕累托形态。[3]

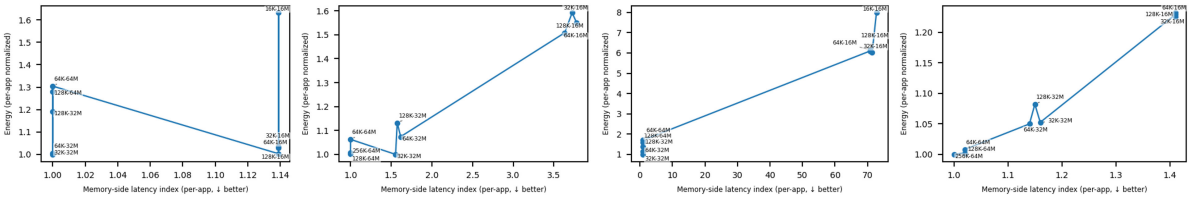


图 5. 帕累托前沿 能耗与存储器侧延迟的帕累托前沿（工作负载从左到右：LoFTR、NeRF、PWC-Net 和 MonoDepth 2).

大多数工作负载都呈现近似对角线的协同改善趋势：能够减少 DRAM 流量的配置，往往也能同时降低能耗和存储器侧延迟。NeRF 是一个明显的例子——片上缓存较小时，存储器侧延迟较高；但当末级缓存足够大时，能耗和延迟都会大幅下降，这表明缓存驻留能够有效抑制重复的片外数据传输。LoFTR 呈现出类似但更渐进的趋势：随着末级缓存增大，能耗和延迟代理指标稳步改善，直到关键中间特征能够存放在片上后趋于平稳。在这些情况下，在所探索的设计空间中，能效最高的配置接近延迟最低的配置。

[3]套件中剩余的工 作负载也呈现出类似的定性趋势；因此，我们仅展示四个代表性网络，以使 he
讨论保持简洁 同时保留关键行为。

然而，情况并非总是如此。图 5 还展示了一些工作负载，其权衡关系更为复杂，需要明确进行多目标平衡。例如，PWC-Net 呈现出两种运行状态：增加末级缓存会显著降低延迟，但能耗最低的配置并不总是与延迟最低的配置相同，这主要取决于片上能耗与分块之间的相互作用。这两个点都位于帕累托前沿上，因此，最佳运行点取决于系统优先考虑能耗还是延迟。图 5 所示的深度模型（例如 Monodepth2）则体现了另一种情况：改变片上容量会显著影响能耗，但对存储器侧延迟的影响很小。这表明，对于此类工作负载，增加片上资源主要是重塑能耗特征，而非降低限制性能的存储器侧延迟。

6 工作负载剖析与表征

为了解 XR 工作负载的计算与存储器行为，同时避免将算法瓶颈与实现或映射造成的局限混为一谈，我们采用双层性能分析方法。我们使用 NVIDIA Nsight[48] 在内核级对图形处理器执行进行剖析，量化架构利用率与效率（例如算术流水线活动、存储器吞吐量与局部性，以及调度效率）。同时，我们使用 Intel Pin[49] 对中央处理器执行指令集架构级插桩，提取整个应用的特征，例如指令组成、存储器访问模式和指令级并行度，以捕获整个应用的指令流特征（包括运行时／控制开销）。这种双重视角的方法有助于区分面向工作负载的属性与内核分解、编译器变换或映射选择所带来的影响。

6.1 图形处理器性能分析

我们使用 Nsight Compute（NCU）对所有应用进行性能分析，将硬件事件归因于各个内核启动，并按执行时间加权汇总指标。实验在 NVIDIA Quadro T2000 上进行，输入分辨率和批次=1 设置与第 3 节相同，以反映实时 XR 推理的约束。我们重点关注归一化的利用率／局部性指标（峰值百分比、命中率），以便跨内核和跨库进行比较。此外，我们按算子族对内核进行汇总（例如卷积/通用矩阵乘法、变换/重采样、逐元素运算、归约、内存复制／数据移动、搜索／索引），并分析一组精简的关键绩效指标，以表征计算饱和度、带宽／局部性和调度效率（将在后文图 6 的雷达图中使用）。

在 NCU 提供的众多硬件计数器中，我们选取了六项简洁且具有代表性的指标，作为表 2 所示架构分析的关键绩效指标。这些关键绩效指标概括了三个方面：计算活动、存储器流量／局部性和指令发射效率，可为跨应用比较提供精简且广泛采用的视角。我们将其用作一阶指标，而非用于全面诊断停顿。

图 6 展示了全部十二个应用在这六项关键绩效指标上的雷达图。总体而言，所有工作负载都呈现中等程度的计算饱和度：所有工作负载的平均 SM_Util% 和 FMA_PIPE% 均处于 30% 至 50% 的范围内，这表明对大多数工作负载而言，持续达到峰值算术吞吐量并非主要瓶颈。相比之下，若干算子类别的 DRAM_BW% 峰值显著，且通常伴随 L_1_Hit% 降低、L_2_Hit% 升高以及 Issue% 偏低。这些模式凸显了存储器流量和有限的指令吞吐量更加突出的一些执行阶段，与整体工作负载所呈现的中等计算饱和度形成对比。

按算子族对所有网络中的内核分组后，可以发现若干反复出现的趋势。卷积/通用矩阵乘法（包括多层感知机/全连接层）通常构成主要的偏计算型主干：与其他算子族相比，这些内核往往具有更高的 SM_利用率和 FMA_流水线占比，且经常占据运行时间的很大一部分（曲线较粗）。其局部性通常有利于二级缓存（L_2_命中率较高），而一级缓存命中率可能有限——在 NeRF 和视觉 Transformer 等多层感知机占比较高的场景中尤为明显。相比之下，带宽压力在变换/重采样及相关算子族（例如扭曲/网格采样、相关性/代价体）中最为显著，这些算子族往往会导致较高的动态随机存取存储器_带宽占比

表 2. 本文使用的主要 Nsight Compute (NCU) 性能分析指标及其解读。

指标	解读（在本文的性能分析语境中）
FMA 流水线%	FMA 流水线利用率（SMSP/SM，按峰值归一化）。数值较高表示较强的 FMA 活动水平和计算强度；较低的值表明执行并非以 FMA 为主，或属于停顿受限执行
SM 利用率%	SM 活动水平（活跃执行周期所占比例）。结合指令发射率，前者高、后者低表示执行处于活跃但停顿状态；两者均高则表示持续发射。
一级缓存命中率%	全局内存访问的 L1/TEX 缓存命中率。数值较低表示近核数据重用有限，或采用流式/步幅访问模式。
L2 命中率	全局内存访问的 L2 缓存命中率。L2 命中率较高而 L1 较低，说明数据重用被捕获在末级缓存中；两者都较低表示缓存友好性差，或工作集超出片上容量。
DRAM 带宽百分比	持续外部动态随机存取存储器带宽（相对于峰值归一化）。数值较高表明片外流量较大，且可能受带宽或动态随机存取存储器服务能力限制，尤其是在缓存命中率较低时。
指令发射率	Warp 发射效率。数值较低表明停顿导致持续发射受限（例如，内存依赖、分歧、同步或延迟隐藏不足）。

，同时一级缓存的数据重用较弱；二级缓存命中率仍可能较高，这表明即使大量未命中仍会访问动态随机存取存储器，二级缓存也能捕获部分局部性。最后，逐元素／归约／数据移动内核构成背景结构，其计算流水线活动较低，带宽需求则各不相同。归约内核往往伴随较低的指令发射率（这与依赖／同步效应相符）；数据移动内核则会产生大量流量，但数据重用有限（若源自基于索引的移动，这一特征可能尤为突出）。在 ICP-Cupoch 等占用空间较小的工作负载中，较高的缓存命中率可能与较低的整体利用率并存，这与由短内核主导的碎片化执行相符。

基于算子族视角，我们结合时间加权平均 KPI 与算子类别中最极端的表现来概括各应用，并归纳出扩展现实工作负载与 GPU 计算与内存层次结构交互时反复出现的四种执行模式。

模式 A — 利用率解耦、随阶段交替。 这些工作负载的应用级 SM_利用率、FMA_流水线利用率和 DRAM_带宽利用率平均处于中等水平，但随着时间推移，特定算子类别会明显拉动 KPI 指标概况，使其朝不同方向变化。卷积/多层感知机阶段的 FMA_流水线利用率较高，而采样、分块和数据移动阶段则会推高 DRAM_带宽利用率，且通常伴随着较低的指令发射率。L2_命中率通常仍然较高，表明即使 DRAM 流量激增，二级缓存仍捕获到了一定的局部性。因此，工作负载会在偏计算阶段（FMA_流水线利用率较高）与内存/带宽受压阶段（DRAM_带宽利用率较高、指令发射率较低）之间交替。NeRF、TinyNeRF 和视觉 Transformer 属于此类。

模式 B — 均衡、缓存友好型流水线。 模式 B呈现出更均衡的关键绩效指标分布：应用级 SM_Util%、FMA_PIPE%、DRAM_BW% 和 L1/L2_Hit% 集中在中等范围内，且占主导地位的卷积/通用矩阵乘法阶段与应用平均值高度吻合，这表明大部分运行时间都花在规则且偏重计算的阶段，缓存行为也较为稳定。虽然变换与归约类别偶尔会产生局部极值，但由于持续时间较短，不会改变整体特征。因此，这些工作负载在适度扩展时能够获得相对稳定（但不一定呈线性）的性能提升。Monodepth2 和 HRDepth 是这种模式的代表。

模式 C — 以 L2 缓存为中心的双重压力流水线。在模式 C 中，计算活动和 DRAM 流量始终保持不可忽视的水平。应用层面的 SM_Util% 和 FMA_PIPE% 通常处于 40–50% 范围内，而 DRAM_BW% 则维持在较高水平但不会出现尖峰；与此同时，L2_Hit% 在整套测试中处于最高水平之一，表明

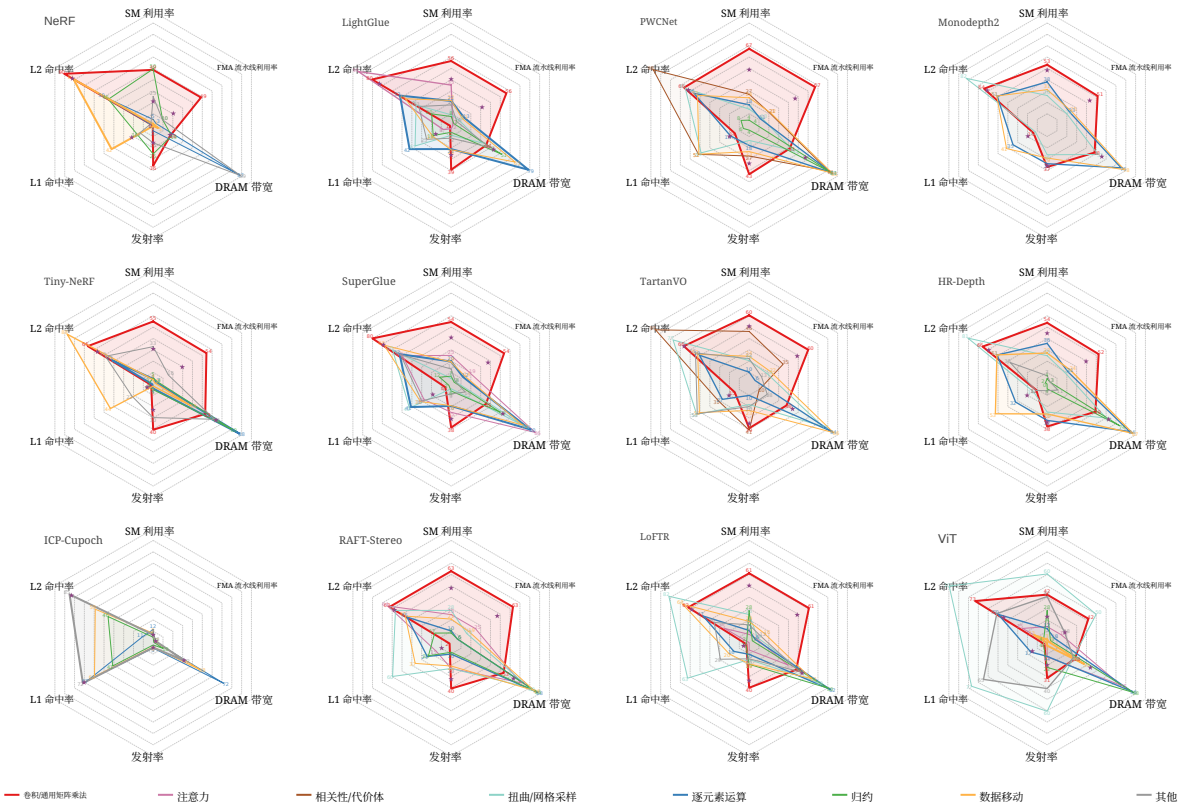


图 6. NCU 剖析雷达图。每个多边形对应一个算子家族，线宽与其占总运行时间的比例成正比（线条越粗，表示该家族运行时间越长）。星号表示应用层面各项关键绩效指标的时间加权平均值。

L2 缓存捕获了相当大比例的访问（以及任何可用的时间局部性）。该模式内部的关键区分指标是 L1_Hit%：（i）C1（一级缓存严重匮乏），其中 L1_Hit% 骤降至个位数，主要卷积/通用矩阵乘法内核的一级缓存命中率接近于零（例如 RAFT 和 LoFTR）；以及（ii）C2（一级缓存有效性中等），其中 L1_Hit% 仍不为零但较为有限，同时执行过程仍从根本上由 L2 缓存主导（例如 LightGlue、SuperGlue、PWC-Net 和 TartanVO）。

模式 D — 缓存驻留、低强度工作负载。模式 D 位于关键绩效指标空间中低利用率、高命中率的区域。应用层面的 SM_利用率、FMA_流水线利用率和指令发射率，均明显低于其他模式；而一级缓存/L2_命中率极高，动态随机存取存储器_带宽占比则保持适中，这与较小的工作集和较低的算术强度相符。占主导地位的内核往往较短且控制密集（通常归为其他），这表明相对于问题规模，计算和带宽资源配置过剩。因此，进一步提升算术逻辑单元的吞吐量或动态随机存取存储器带宽不太可能带来明显收益；相比之下，减少控制开销、改进快速本地存储通常更为重要。Cupoch ICP 属于这一模式，但该模式的定义由指标决定，也可能涵盖其他小规模、缓存驻留的几何或类图工作负载。

值得注意的是，我们很少观察到稳定的“L2 命中率低、L1 命中率高”模式，也很少遇到 L2 命中率持续偏低的工作负载。这可归因于两个因素：首先，现代 NVIDIA 图形处理器会将几乎所有全局内存流量先导入

统一的 L2 缓存，再送往动态随机存取存储器（DRAM）；因此，只要工作负载具有一定的时间或空间局部性，通常就能获得较为理想的 L2 命中率，而 L2 命中率持续偏低则需要工作集非常大，或访问模式高度不规则。其次，主流的 XR 感知内核——cuDNN 风格的卷积、密集型通用矩阵乘法（GEMM）/多层感知机（MLP）以及标准逐元素运算——采用大量分块处理，并倾向于通过合并的 L2 流式访问来实现数据重用。相比之下，每个 SM 的 L1 缓存容量受限，且与共享内存共用，因此难以持续维持较高的 L1 命中率。大型稀疏图神经网络或推荐模型具有非常不规则的访问模式，更容易削弱缓存效果；不过，这类云端后端工作负载在批处理方式和延迟约束方面有所不同，不在本文的 XR 推理研究范围之内。

6.2 CPU 性能分析

为补充 GPU 内核性能分析，我们在仅 CPU 执行环境下对每个应用进行端到端剖析（即，整个流水线均在中央处理器上运行，不涉及任何图形处理器）。这种视角能够捕捉整个应用的指令流特征，包括设备端内核测量中无法观察到的控制平面开销和运行时影响。

图 7(a) 展示了动态指令构成，表明算术指令在各工作负载中均占相当大的比例，而显式内存操作和控制流指令也不可忽视。为进一步概括 CPU 端的数据访问压力，图 7(b) 将每个工作负载映射到两个互补的指令集架构指标上：指令级算术强度 AI = 算术% 加载%+存储% 以及每指令字节数 BPI（每条已执行指令访问的字节数）。此外，数据点还根据 CPU 端缓存命中率代理指标着色，以反映缓存友好性。总体而言，这些特征有助于区分计算密集型工作负载（ AI较高、 BPI适中）与内存压力型工作负载（ AI较低和/或 BPI较高），并突出显示缓存友好性较差与数据移动开销增加相关的情形。

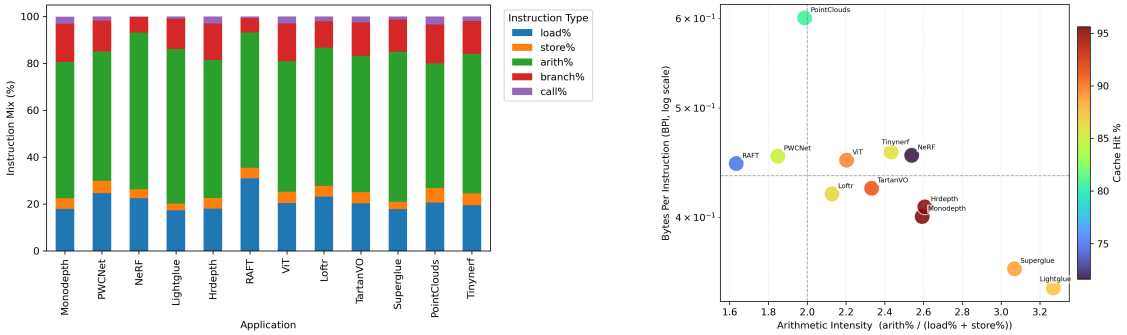


图 7. CPU 指令集架构级性能分析结果。(a) 动态指令分布（加载/存储、算术运算和控制流）。(b) 各工作负载的 AI–BPI 投影，其中 AI 反映算术运算与显式内存指令所占比例，BPI 表示每条指令访问的字节数；数据点按缓存命中率着色。

根据图 7(b) 中的 AI–BPI 位置以及缓存命中率（CPU 侧缓存友好性的代理指标），我们从 CPU 侧视角总结出四种反复出现的运行模式：

算术运算密集且缓存友好（AI 高、BPI 低、命中率高）。此区域中的工作负载呈现出偏向计算的指令流，且缓存友好性强，因此不太可能受加载/存储指令活动主导。在我们的测试套件中，HRDepth 和 MonoDepth 属于这一类别；纳入 LightGlue 和 SuperGlue 时，它们也呈现出类似特征。

混合/均衡的指令流（AI 中等、BPI 中等、命中率中等至较高）。此运行模式反映了混合 d 结构，其中 算术运算和内存指令都占有相当比重，但缓存友好性仍属合理。Vi T,

TartanVO、LoFTR 和 TinyNeRF 聚集在这一区域，表明其指令流特征较为混合，而非明显偏向计算或内存指令分布。

内存指令密集型（低算术强度、非平凡每指令字节数）。 该区域中的点表示相对于算术运算而言更高的加载/存储密度，通常伴随着显著的数据访问需求。PWC-Net，尤其是 RAFT，呈现出这一特征；RAFT 还表现出更低的缓存命中率，使内存指令压力与较弱的缓存友好性相互叠加。这一趋势与这些流水线中变换/重采样密集阶段的特征一致。

尽管算术强度尚可，但缓存友好性/数据移动仍面临挑战。 这一机制表明，仅仅采用偏计算型的指令组合，并不能保证缓存行为良好：即使算术强度相对较高，有限的数据重用或不规则访问仍可能带来缓存友好性和数据移动压力。NeRF 是最具代表性的示例，其算术强度相对较高，但缓存命中率明显较低。点云则呈现出互补特征，其每指令字节数显著更高，表明每条指令对应的数据访问需求大得多。

7 工作负载原型与跨层架构启示

第 5 节和第 6 节从三个互补角度共同刻画了 XR 工作负载套件的架构行为。本节将这些观察结果整合为一个紧凑的跨层行为空间，并将反复出现的行为归纳为少数几种工作负载原型。最后，我们将这些原型转化为面向 XR 的加速器设计架构指南。

7.1 跨层行为空间

为避免针对单一平台或实现细节过度拟合，我们将每种工作负载置于一个包含三个互补维度的紧凑跨层空间中，每个维度都对应一个不同的设计杠杆。**（D1）片上容量敏感性（设计空间探索视角）**。我们的设计空间探索模型在一致的映射策略下扫描一级缓存/末级缓存容量，量化额外的片上存储如何将潜在的数据重用转化为整个存储层次结构中实际的流量减少。该视角揭示工作负载是否存在明确的工作集阈值，能大幅降低片外压力，还是仅随着容量增加而逐步改善、收益递减。**（D2）GPU 执行模式与复用实现（GPU 视角）**。图形处理器性能分析刻画主要的运行时压力，以及实际实现数据复用的位置——是在计算单元附近（例如 L1/TEX），还是主要在更大的片上层级（例如二级缓存）。在实际应用中，观测到的执行模式受主要算子类别、其内核分解方式，以及性能/能耗压力体现为计算吞吐量限制、片上数据移动还是片外带宽需求等因素影响。**（D3）整应用指令压力与控制/内存行为（中央处理器视角）**。CPU 指令集架构性能分析提供了整程序参考，可补充以设备内核为中心的 GPU 性能分析。由于中央处理器视角自然涵盖控制平面和运行时影响（例如调度、同步和框架开销），它能够揭示端到端指令压力和缓存友好性；仅分析 GPU 内核时，这些特征在很大程度上不可见。这有助于区分面向工作负载的访问/控制特征与 GPU 内核分解和映射所特有的影响。综合来看，（D1）在统一映射模型下刻画容量杠杆效应，（D2）概述*GPU*侧执行模式及实际复用实现情况，（D3）则为指令/控制压力和缓存行为提供端到端参考。我们以这三个维度构成紧凑的跨层空间，用于概述 XR 工作负载行为。

7.2 跨层工作负载类型

我们围绕图形处理器侧的执行模式组织结构原型，并根据设计空间探索容量趋势和 CPU ISA叠加层细化边界。由此得到四种结构原型；阶段交替则被视为一种时间叠加，可与任何结构原型并存。

类型 I：容量受限的变换流水线，近缓存局部性较弱。（代表性应用： RAFT、PWC-Net、视觉 Transformer、TartanVO。）这一组工作负载呈现出类似模式 C 的图形处理器特征（一级缓存活动度高，动态随机存取存储器需求较高），但在我们探索的一级缓存/末级缓存网格范围内，其相当一部分片外流量似乎可通过片上扩容转化为片上重用。设计空间探索扫描显示，部分工作负载（例如 RAFT）存在明显的转变区间：随着末级缓存跨过工作集边界，动态随机存取存储器能耗急剧下降，压力也从片外流量转向片上重用。其他工作负载（例如 TartanVO、视觉 Transformer，以及部分 PWC-Net）也能从中显著受益，但沿末级缓存轴的转变更为渐进。这表明，基线动态随机存取存储器需求主要由容量失效和数据重用捕获不充分所致，而非单纯由强制流式访问造成。视觉 Transformer 主要因其容量受限的注意力阶段而归入这一模式，其端到端流水线也可能呈现阶段交替。

原型II：复用难以实现、响应平坦的变换／匹配流水线。（代表性应用： LightGlue、SuperGlue、LoFTR。）这些工作负载呈现出类似模式 C 的图形处理器特征，但其设计空间探索曲线明显平坦：在所探索的一级缓存／末级缓存预算范围内，动态随机存取存储器始终是主要能耗来源。中央处理器端指令集架构特征相对偏向计算，且缓存命中率较高，这表明仅凭算法级内存访问强度，无法充分解释图形处理器上持续的动态随机存取存储器需求。相反，证据表明，图形处理器上的局部性潜力难以转化为实际收益，原因包括间接索引（聚集/散布）、各阶段间的中间结果实体化，以及内核执行碎片化带来的开销。因此，单靠扩大缓存规模收益有限；要有效提供支持，需要显式分阶段处理，并采用开销更低的数据移动原语。

类型 III：均衡且缓存友好的流水线，但扩大缓存规模带来的收益递减。（代表性应用：Monodepth2、HRDepth；以及 NeRF 风格网络中的多层感知机计算阶段。）这些流水线适合传统的密集型加速（以卷积/通用矩阵乘法/多层感知机为主），当适度的片上容量足以容纳主要工作集后，进一步扩大缓存规模的收益便会递减。这一运行模式大体已由传统密集型加速设计所覆盖：进一步进行通用扩展仍可提升绝对性能，但在面积／功耗预算受限的扩展现实片上系统场景中，其边际价值较低，而其他类型暴露出的瓶颈则更为突出。对于 NeRF 风格工作负载，我们强调，此类型仅描述独立的多层感知机计算阶段；由于渲染／采样循环的存在，端到端流水线仍可能呈现阶段交替（时间叠加）。

类型 IV：对开销和延迟敏感的不规则几何内核。（代表性应用：ICP-Cupoch 类型的点云工作负载。）这类工作负载往往无法维持较高的图形处理器利用率，原因并非峰值算术吞吐量不足，而是执行过程受不规则遍历和较短的依赖链主导，例如反复进行最近邻查询、采用聚集/散布式索引，以及在迭代细化过程中进行细粒度同步。在中央处理器上，这些内核通常表现为数据移动密集、算术密度较低的指令流；缓存命中率可能随数据结构和工作集大小而变化，但其行为与不规则空间遍历和细粒度同步相符（例如空间数据结构中的指针追逐式访问）。因此，单纯配置更大的缓存或更高的峰值带宽，并不能直接解决主要瓶颈；有效的支持还需要低开销的不规则访问原语、延迟容忍调度，以及能够减少同步和中间结果实体化的显式暂存机制。

时间叠加：相位交替渲染/采样流水线。（代表性应用：NeRF 风格网络、TinyNeRF、视觉 Transformer。）除结构原型外，还有若干流水线呈现阶段交替：执行会在偏计算阶段（如多层感知机推理/着色）与密集访问/采样阶段（如光线步进、重采样/变形、分块、密集收集索引）之间反复切换。即使增加缓存容量可以减少特定流水线的片外流量，端到端效率仍可能受各阶段最佳计算–带宽–容量平衡不匹配的制约。因此，单一静态

配置点必然会使至少一个阶段处于次优状态，因此需要采用阶段感知调度和弹性资源分配。

7.3 架构启示

第 5–6 节表明，XR 工作负载很少由单一资源决定；相反，结果取决于工作集适配、复用实现和执行结构随时间的相互作用。因此，片上容量并非一种普适的“增加缓存”旋钮：当它跨越工作集阈值时，效果显著；但当难以在计算单元附近实现数据重用，或开销与阶段转换占主导时，收益便会递减。在我们的测试套件中，反复出现三种机制：容量阈值、复用实现差距以及对时间变化和开销的敏感性。下面我们围绕这些机制总结可付诸实践的架构启示。

启示 1：配置足够的末级片上容量，以跨越工作集阈值（若存在）。对于容量受限的情形，充足的末级片上容量是避免反复溢出/填充和持续承受动态随机存取存储器惩罚的首要条件。低于该阈值时，单纯扩展计算能力不太可能改变根本的片外成本特征。

启示 2：当近缓存复用在结构上难以实现时，应提供显式的局部性/暂存机制。对于平响应工作负载（原型 II），仅靠被动缓存无法可靠地将片外流量转化为片上重用。架构应提供便笺式暂存（例如可编程缓冲和双缓冲），以及轻量级、适用于聚集/散布的数据移动支持，使运行时能够减少中间结果实体化、合并不规则访问，并在不同阶段之间分摊同步开销。

启示 3：提供弹性，并在不同阶段和边际收益递减的场景间重新分配预算。对于阶段交替流水线，静态配置必然会导致至少一个阶段的资源利用不足；对于均衡的缓存友好型场景，一旦主要工作集已驻留片上，收益也会逐渐递减。因此，实用的扩展现实片上系统策略应避免在已得到充分支持的阶段过度投入通用扩展，而应将面积/功耗分配给弹性执行支持（阶段感知调度和可适配片上数据流），以及那些额外资源能够显著改变成本特征的场景（容量阈值和不规则性容忍度）。

启示 4：对于不规则且开销敏感型内核，应优先支持低开销执行，而非一味追求原始带宽。在延迟或不规则性占主导的场景中，关键需求是降低开销并适应不规则性，而不只是提高带宽。轻量级执行模式、不规则性感知调度、高效索引原语，以及靠近片上数据的低延迟同步，有助于缓解面向吞吐量的执行中的内核碎片化和依赖链停顿。

8 结论

本文从三个互补视角，对具有代表性的边缘扩展现实工作负载进行了系统的架构级表征。我们结合了：（1）轻量、快速的分析型设计空间探索模型，用于系统地遍历片上存储器容量和映射选择；（2）跨平台性能分析——GPU内核级和CPU指令集架构级——以量化利用率、局部性、调度效率以及端到端指令流特征。这些视角共同构成了扩展现实行为的“架构制图”，使我们能够对 12 个具有代表性的扩展现实内核的工作集压力、带宽需求、计算利用率和运行时开销进行可比分析。基于由此得到的跨层行为空间，我们提炼出五种可复现的工作负载原型，并将其转化为面向下一代扩展现实片上系统的可行架构启示。

我们的结果表明，扩展现实的性能和能效通常受阈值效应与相变效应支配，而非来自通用资源扩展所带来的线性增益。多个内核表现出明显的片上容量临界点，以及复用实现差距：尽管存在算法复用，却未能被被动缓存有效捕获，

并且多阶段图还会引入时间变化性。因此，扩展现实片上系统的设计应强调由阈值和阶段驱动的组织方式：在能够解锁容量受限阶段的地方配置片上容量；为抗缓存阶段增加显式分阶段支持和非规则数据通路；针对阶段交替流水线采用阶段感知调度和弹性资源分配；而对于开销主导型内核，应优先降低同步与调度开销，而不是继续扩展峰值计算能力或带宽。

不过，仍有一些假设需要说明。首先，我们有意侧重于存储层次敏感性和跨工作负载可比性，而非周期精确或经实现校准的预测。计算能耗和延迟采用简化的代理指标建模（例如，与浮点运算次数成正比的计算能耗，以及基于屋顶线的延迟处理方式），而单指令多数据架构模板仅作为统一的分析基准，并非旨在提出最优的扩展现实加速器方案。因此，我们的设计空间探索结果应被视为用于分析趋势和临界点的敏感性证据，而非针对特定硅实现的绝对预测。其次，我们的分析针对批量=1 近实时推理约束下的单应用行为，并未建模完整系统的运行时影响，例如多应用并发、干扰或动态互连／运行时策略。

参考文献

[1] Muhammad Huzaifa、Rishi Desai、Samuel Grayson、Xutao Jiang、Ying Jing、Jae Lee、Fang Lu、Yihan Pang、Joseph Ravichandran、Finn Sinclair等。2021年。ILLIXR：支持端到端扩展现实研究。载于*2021 年 IEEE 国际工作负载特征分析研讨会（IISWC）*。IEEE，24–38。[2] Minna Vasarainen、Sami Paavola 和 Liubov Vetoshkina。2021年。《扩展现实系统性文献综述：工作生活中的虚拟现实、增强现实与混合现实》。《style id='5'>《国际虚拟现实期刊》21，2（2021年10月），1–28。https://doi.org/10.20870/IJVR.2021.21. 2.4620[3] Jyothsna Priya Kattakinda、Sachin Dhananjaya、Samuel Mojzis、Amine Nadif、Mattheus Hanna、Radu Apsan 和 Ivano Malavolta。2024年。探究虚拟现实应用在游戏、教育和娱乐中的能耗。载于*ACM/IEEE 第八届游戏与软件工程国际研讨会论文集*。50–57。[4] Yu-Hsin Chen、Tien-Ju Yang、Joel Emer 和 Vivienne Sze。2019年。Eyeriss v2：面向移动设备中新兴深度神经网络的灵活加速器。《*IEEE 电路与系统新兴及精选主题期刊*》9，2（2019年），292–308。[5] Eric Qin、Ananda Samajdar、Hyoukjun Kwon、Vineet Nadella、Sudarshan Srinivasan、Dipankar Das、Bharat Kaul 和 Tushar Krishna。2020年。SIGMA：一种用于深度神经网络训练、具备灵活互连的稀疏非规则 GEMM 加速器。载于*2020 年 IEEE 高性能计算机体系结构国际研讨会（HPCA）*。

58 70 10 1109 47549 2020 00015 –。https://doi.org/。/ [6] 2018 HPCA。。Hyoukjun Kwon、A. Samajdar 和 Tushar Krishna。。MAERI：通过可重构互连支持在 DNN 加速器上灵活映射数据流。第二十三届编程语言与操作系统体系结构支持国际会议论文集(2018 3990189)。

[7] https://api.semanticscholar.org/CorpusID: Ye Liu、Shuang Hao、Xiuyuan Qi、Kun Huang、Minghui Yang、Zili Huang、Yiting Li、Liang Chang、Liang Zhou、Yu Long等。2025年。一种面向自动驾驶、基于现场可编程门阵列的超高性能可扩展光流硬件加速器。*IEEE 电路与系统汇刊 I*：常规论文（2025年）。[8] Miao Sun、Yingjie Cao、Jian Qian、Jie Li、Sifan Zhou、Ziyu Zhao、Yifan Wu、Tao Xia、Yajie Qin、Lei Qiu等。2023年。一款40纳米、2tops/w的深度补全神经网络加速器片上系统，配备高效深度引擎，适用于实时激光雷达系统。《*IEEE 电路与系统汇刊 II*：简报》70，5（2023年），1704–1708。[9] Yuanfang Wang、Yu Li、Haoyang Zhang、Jun Yu 和 Kun Wang。2023年。Moth：一种用于在现场可编程门阵列上进行神经辐射场推理的硬件加速器。载于*2023 年 IEEE 第 31 届现场可编程定制计算机年度国际研讨会（FCCM）*。IEEE，227–227。[10] Donghyeon Han、Junha Ryu、Sangyeob Kim、Sangjin Kim、Jongjun Park 和 Hoi-Jun Yoo。2023年。MetaVRain：一种移动神经网络三维渲染处理器，结合基于帧束熟悉度的 NeRF 加速与混合 DNN 计算。《*IEEE 固态电路期刊*》59，1（2023年），65–78。[11] Yang Wang、Huizheng Wang、Ruiqi Guo、Fengbin Tu、Jianxun Yang、Shaojun Wei、Yang Hu、Shouyi Yin等。2024年。CV-CIM：一种混合域异或派生的相似性感知存内计算技术，支持代价体构建。《*IEEE 固态电路期刊*》60，2（2024年），719–733。[12] Haoran You、Zhanyi Sun、Huihong Shi、Zhongzhi Yu、Yang Zhao、Yongan Zhang、Chaojian Li、Baopu Li 和 Yingyan Lin。2023年。Vitcod：通过专用算法与加速器协同设计实现视觉 Transformer 加速。载于*2023 年 IEEE 高性能计算机体系结构国际研讨会（HPCA）*。IEEE，273–286。

[13] Yonggan Fu、Zhifan Ye、Jiayi Yuan、Shunyao Zhang、Sixu Li、Haoran You 和 Yingyan Lin。2023年。Gen-nerf：通过算法-硬件协同设计实现高效且可泛化的神经辐射场。载于第五十届国际计算机体系结构研讨会论文集。1–12。[14] H Ekin Sumbul、Tony F Wu、Yuecheng Li、Syed Shakib Sarwar、William Koven、Eli Murphy-Trotzky、Xingxing Cai、Elnaz Ansari、Daniel H Morris、Huichu Liu等。2022年。采用7纳米工艺的定制低功耗DNN加速器芯片，用于编解码化身的AR/VR硬件原型系统级设计与集成。载于2022年 IEEE 定制集成电路会议（CICC）。IEEE，01–08。[15] Tejas Chaudhari、Tanushree Dewangan、Mukul Lokhande、Santosh Kumar Vishvakarma等。2025年。XR-NPE：面向扩展现实感知工作负载的高吞吐量混合精度SIMD神经处理引擎。arXiv预印本 arXiv:2508.13049（2025年）。[16] Arpan Suravi Prasad、Luca Benini 和 Francesco Conti。2023年。专用化兼顾灵活性：一种用于高效、高灵活性 AR/VR处理的异构架构。载于2023年 ACM/IEEE 第60届设计自动化会议（DAC）。IEEE，1–6。[17] Kathleen Feng、Kartik Prabhu、Kai Bartolone、Jeffrey Yu 和 Priyanka Raina。2025年。Aspen：一种用于扩展现实感知工作负载、每秒630帧的实时Posit统一加速器。载于2025年 IEEE 定制集成电路会议（CICC）。IEEE，1–3。[18] H Ekin Sumbul、Jae-sun Seo、Daniel H Morris 和 Edith Beigne。2023年。一种全数字、按行流水化的存内计算神经网络加速器，具有面向增强/虚拟现实应用的片上系统级基准测试。IEEE Micro 44，2（2023年），61–70。[19] Yiwei Zhao、Ziyun Li、Win-San Khwa、Xiaoyu Sun、Sai Qian Zhang、Syed Shakib Sarwar、Kleber Hugo Stangherlin、Yi-Lun Lu、Jorge Tomas Gomez、Jae-Sun Seo等。2024年。面向NPU-CIM异构AR/VR设备的混合模型神经架构搜索。arXiv预印本 arXiv:2410.08326（2024年）。[20] Reid Pinkham、Andrew Berkovich 和 Zhengya Zhang。2021年。面向增强现实与虚拟现实的近传感器分布式DNN处理。《IEEE 电路与系统新兴及精选主题期刊》11，4（2021年），663–676。[21] Lita Yang、Robert M Radway、Yu-Hsin Chen、Tony F Wu、Huichu Liu、Elnaz Ansari、Vikas Chandra、Subhasish Mitra 和 Edith Beigné。2022年。面向 AR/VR应用的三维堆叠神经网络加速器架构。IEEE Micro 42，6（2022年），116–124。[22] Hyoukjun Kwon、Krishnakumar Nair、Jamin Seo、Jason Yik、Debabrata Mohapatra、Dongyuan Zhan、Jinook Song、Peter Capak、Peizhao Zhang、Peter Vajda等。2023年。Xrbench：面向元宇宙的扩展现实（xr）机器学习基准套件。《机器学习与系统论文集》5（2023年），1–20。[23] Steven Gao、Jeffrey Liu、Qinjun Jiang、Finn Sinclair、William Sentosa、Brighten Godfrey 和 Sarita Adve。2025年。XRgo：面向低功耗扩展现实设备的渲染卸载设计与评估。载于第十六届 ACM 多媒体系统会议论文集。124–135。[24] Clément Godard、Oisin Mac Aodha、Michael Firman 和 Gabriel J Brostow。2019年。深入探究自监督单目深度估计。载于 IEEE/CVF 国际计算机视觉会议论文集。3828–3838。[25] Xiaoyang Lyu、Liang Liu、Mengmeng Wang、Xin Kong、Lina Liu、Yong Liu、Xinxin Chen 和 Yi Yuan。2021年。HR-Depth：高分辨率自监督单目深度估计。载于 AAAI 人工智能会议论文集，第35卷。2294–2301。[26] Deqing Sun、Xiaodong Yang、Ming-Yu Liu 和 Jan Kautz。2018年。PWC-Net：使用金字塔、变形和代价体进行光流估计的卷积神经网络。载于 IEEE 计算机视觉与模式识别会议论文集。8934–8943。[27] Lahav Lipson、Zachary Teed 和 Jia Deng。2021年。RAFT-Stereo：用于立体匹配的多级循环场变换。载于2021年国际三维视觉会议（3DV）。IEEE，218–227。[28] Wenshan Wang、Yaoyu Hu 和 Sebastian Scherer。2021年。TartanVO：一种可泛化的学习式视觉里程计。载于机器人学习会议。PMLR，1761–1772。[29] Philipp Lindenberger、Paul-Edouard Sarlin 和 Marc Pollefeys。2023年。LightGlue：以闪电般的速度进行局部特征匹配。载于 IEEE/CVF 国际计算机视觉会议论文集。17627–17638。[30] Jiaming S un、Zehong Shen、Yuang Wang、Hujun Bao 和 Xiaowei Zhou。2021年。LoFTR：一种无需检测器、基于 Transformer 的局部特征匹配方法。载于 IEEE/CVF 计算机视觉与模式识别会议论文集。8922–8931。[31] Paul-Edouard Sarlin、Daniel DeTone、Tomasz Malisiewicz 和 Andrew Rabinovich。2020年。SuperGlue：利用图神经网络学习特征匹配。载于 IEEE/CVF 计算机视觉与模式识别会议论文集。4938–4947。[32] Alexey Dosovitskiy。2020年。一张图像胜过16×16个词：用于大规模图像识别的 Transformer。arXiv预印本 arXiv:2010.11929（2020年）。[33] Ben Mildenhall、Pratul P Srinivasan、Matthew Tancik、Jonathan T Barron、Ravi Ramamoorthi 和 Ren Ng。2021年。NeRF：将场景表示为用于视图合成的神经辐射场。《ACM 通讯》65，1（2021年），99–106。[34] Tianli Zhao、Jiayuan Chen、Cong Leng 和 Jian Cheng。2023年。TinyNeRF：朝着体素辐射场压缩100倍迈进。载于 AAAI 人工智能会议论文集，第37卷。3588–3596。[35] Jaesik Park等。2021年。Cupoch：用于三维数据处理的 CUDA加速库。PyPI软件包。https://pypi.org/project/cupoch/0. 2.7.4/ 访问日期：2025-10-20。[36] Matheus Cavalcante、Fabian Schuiki、Florian Zaruba、Michael Schaffner 和 Luca Benini。2019年。Ara：一种1-GHz+ 可扩展且高能效的RISC-V向量处理器，采用22纳米FD-SOI工艺并支持多精度浮点。IEEE 超大规模集成电路系统汇刊28，2（2019年），530–543。

[37] Pengbo Yu、Flavio Ponzina、Alexandre Levisse、Mohit Gupta、Dwaipayan Biswas、Giovanni Ansaloni、David Atienza 和 Francky Catthoor。2024年。一种节能的软 SIMD 微架构及其在量化卷积神经网络中的应用。*IEEE 超大规模集成电路系统汇刊*32, 6 (2024年), 1018–1031。[38] Nitish Satya Murthy、Nathan Laubeuf、Debjyoti Bhattacharjee、Francky Catthoor 和 Marian Verhelst。2024年。用于边缘端深度神经网络训练的向量处理器自适应分块缩放通用矩阵乘法。载于*2024*年 *IFIP/IEEE* 第*32*届超大规模集成电路国际会议（*VLSI-SoC*）。IEEE，1–6。[39] Yoongu Kim、Weikun Yang 和 Onur Mutlu。2015年。Ramulator：一款快速且可扩展的 DRAM 模拟器。*IEEE* 计算机体系结构快报15, 1 (2015年), 45–49。[40] Karthik Chandrasekar、Christian Weis、Yonghui Li、Sven Goossens、Matthias Jung、Omar Naji、Benny Akesson、Norbert Wehn 和 Kees Goossens。2014年。DRAMPower：开源 DRAM 功耗与能耗估算工具。开源软件。http://www.drampower. info 访问日期：2025年5月18日。[41] Angshuman Parashar、Priyanka Raina、Yakun Sophia Shao、Yu-Hsin Chen、Victor A Ying、Anurag Mukkara、Rangharajan Venkatesan、Brucek Khailany、Stephen W Keckler 和 Joel Emer。2019年。Timeloop：一种系统化的深度神经网络加速器评估方法。载于*2019*年 *IEEE* 系统与软件性能分析国际研讨会（*ISPASS*）。IEEE，304–315。[42] Jonathan Ragan-Kelley。2018年。Halide：用于优化图像处理流水线的语言与编译器。CS348V：计算机图形系统，斯坦福大学，讲义幻灯片。https://graphics.stanford.edu/courses/cs348v-18-winter/lectures/04_halide.pdf 访问日期：2025年12月15日。[43] Andrew Adams、Karima Ma、Luke Anderson、Riyadh Baghdadi、Tzu-Mao Li、Michaël Gharbi、Benoit Steiner、Steven Johnson、Kayvon Fatahalian、Frédo Durand 等。2019年。通过树搜索和随机程序学习优化 Halide。*ACM* 图形学汇刊（*TOG*）38, 4 (2019年), 1–12。[44] Tianqi Chen 、 Lianmin Zheng 、 Eddie Yan 、 Ziheng Jiang 、 Thierry Moreau 、 Luis Ceze 、 Carlos Guestrin 和 Arvind Krishnamurthy。2018年。学习优化张量程序。神经信息处理系统进展31 (2018年)。[45] Zhenlin Pei、Hsiao-Hsuan Liu、Mahta Mayahinia、Mehdi Tahoori、Francky Catthoor、Zsolt Tokei、Prashant Dubey 和 Chenyun Pan。2025年。利用差分传输线进行互连/存储器协同设计与协同优化。*IEEE* 超大规模集成电路系统汇刊

33, 11 2025 3118 3130 10 1109 2025 3595818 ( ), – 。https://doi.org/ . /TVLSI. 。

[46] Hsiao-Hsuan Liu、Carlo Gilardi、Shairfe M. Salahuddin、Zhenlin Pei、Pieter Schuddinck、Yang Xiang、Pieter Weckx、Geert Hellings、Marie Garcia Bardon、Julien Ryckaert、Chenyun Pan、Subhasish Mitra 和 Francky Catthoor。2024年。SRAM 数据阵列的未来设计方向：采用有源互连的分层子阵列。*IEEE* 电路与系统汇刊 *I*：常规论文71, 12 (2024年), 6495–6506。https://doi.org/10.1109/TCSI.2024.3410518[47] Junren Chen、Kanishkan Vadivel、Dawit Abdi、Priya Venugopal、Refik Bilgic、Giacomo Indiveri、Fernando Garcia Redondo 和 Dwaipayan Biswas。2024年。事件驱动架构中面向边缘推理的非易失性存储器扩展。载于*2024*年 *IEEE* 亚太电路与系统会议（*APCCAS*）。

654 658 10 1109 62602 2024 10808989 – 。https://doi.org/ . /APCCAS . 。英伟达公司。 。

[48] 2023 NVIDIA Nsight Compute。GPU 内核性能分析工具。https://developer.nvidia.com/nsight-compute 访问日期：2025年5月15日。[49] Chi-Keung Luk、Robert Cohn、Robert Muth、Harish Patil、Artur Klauser、Geoff Lowney、Steven Wallace、Vijay Janapa Reddi 和 Kim Hazelwood。2005年。Pin：利用动态插桩构建定制程序分析工具。*ACM SIGPLAN* 通讯40, 6 (2005年), 190–200。