# NVIDIA TURING GPU 架构

显卡技术的颠覆性创新

# 目录

# 插图目录

# 表格目录

# NVIDIA TURING 架构简介

伴随游戏市场的持续发展及其对更优质 3D 图形的要求与日俱增，NVIDIA® 已将 GPU 发展为世界领先的并行处理引擎，可满足众多计算密集型应用程序的需求。NVIDIA GPU 不仅能渲染高度逼真且引人入胜的 3D 游戏，还可加速内容创建工作流程、高性能计算 (HPC) 和数据中心应用程序，以及众多人工智能系统与应用程序。

作为十多年来架构发展的重大飞跃，Turing 架构推出全新核心 GPU 架构，可大力提升 PC 游戏、专业图形应用程序和深度学习推理的效率与性能。

通过采用基于硬件的全新加速器和混合渲染方法，Turing 架构将光栅化、实时光线追踪、AI 和模拟技术融于一身，可在 PC 游戏中实现令人难以置信的真实感、由神经网络驱动的全新惊艳效果、电影级交互体验，并可在用户创建和浏览复杂 3D 模型时为其提供流畅的交互性。

Turing 核心架构采用全新 GPU 处理器（流式多元处理器，SM）架构，可有效提升着色器执行效率，同时还配备支持最新 GDDR6 显存技术的全新显存系统架构。得益于这两大关键配置，Turing 的图形性能得以显著提升。

ImageNet Challenge 等图像处理应用程序已率先在深度学习领域初尝硕果，因此可以预见，AI 具备解决众多重要图形问题的潜力。Turing Tensor 核心可助力一套基于深度学习的神经服务，不仅能为基于云的系统提供快速 AI 推理，还可在游戏和专业图形领域实现出色的图形效果。

一直以来，实时光线追踪都是我们在计算机图形渲染领域梦寐以求的目标。现在梦想照进现实，一切尽在采用 NVIDIA Turing GPU 架构的单 GPU 系统。Turing GPU 采用全新 RT 核心，这些加速器单元能够以非凡效率专门执行光线追踪操作，从而能够彻底摒弃以往基于软件仿真且代价高昂的光线追踪方式。通过与 NVIDIA RTX™ 软件技术和精密的过滤算法相结合，这些新单元可助力 Turing 提供实时光线追踪渲染，包括凭借对阴影、反射和折射物理属性的准确把握来获得逼真的物体和环境。

在开发 Turing 的同时，Microsoft 还于 2018 年初宣布推出面向 AI 的 DirectML 和 DirectX Raytracing (DXR) API。通过将 Turing GPU 架构和 Microsoft 的新型 AI 及光线追踪 API 相结合，游戏开发者可在其游戏中快速部署实时 AI 和光线追踪。

Turing 不仅具有开创性的 AI 和光线追踪功能，而且配备众多新推出的高级着色功能，可提高性能、改善图像质量，并能提供更高水平的几何图形复杂度。

此外，Turing GPU 还继承了 Volta 架构为 NVIDIA CUDA™ 平台引入的所有增强功能，从而能够提升计算应用程序的能力、灵活度、效率和可移植性。Turing GPU 架构拥有诸多特性，包括独立线程调度、具有多应用程序地址空间隔离的硬件加速多进程服务 (MPS) 以及协作组等。

新推出的几款 NVIDIA GeForce® 和 NVIDIA Quadro™ GPU 产品将会搭载 Turing GPU。本文中，我们主要关注 NVIDIA 旗舰款 Turing GPU 的架构和多项功能。Turing GPU 代号为 TU102，而即将发售的 GeForce RTX 2080 Ti 和 Quadro RTX 6000 便搭载此款 GPU。技术细节（包括 TU104 和 TU106 Turing GPU 的产品规格）均包含在附录中。

图 1 展示了 Turing 如何利用全新架构重塑图形，这一架构包括增强版 Tensor 核心、全新 RT 核心以及新推出的众多高级着色功能。Turing 将可编程着色、实时光线追踪和 AI 算法融于一身，能够为游戏和专业应用程序打造极为逼真且又具备准确物理属性的图形。

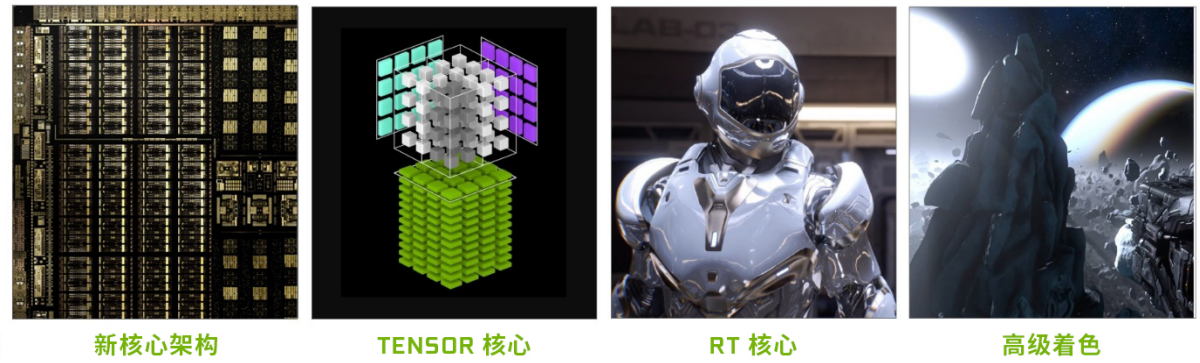



图 1. Turing 重塑图形

## NVIDIA Turing 的主要特性

NVIDIA Turing 是世界上先进的 GPU 架构。高端 TU102 GPU 拥有 186 亿个晶体管，这些晶体管均采用 TSMC 12 nm FFN (FinFET NVIDIA) 高性能制造工艺打造而成。

GeForce RTX 2080 Ti Founders Edition GPU 可提供非凡的计算性能，具体如下：

- 14.2 TFLOPS[1] 的峰值单精度 (FP32) 性能
- 28.5 TFLOPS[1] 的峰值半精度 (FP16) 性能
- 14.2 TIPS[1] 的 FP 并行计算性能（通过与独立的整数执行单元并行执行）
- 113.8 Tensor TFLOPS[1,2]
- 100 亿条光线/秒
- 78 Tera RTX-OPS[3]

Quadro RTX 6000 可提供专为专业工作流程设计的卓越计算性能：

- 16.3 TFLOPS[1] 的峰值单精度 (FP32) 性能
- 32.6 TFLOPS[1] 的峰值半精度 (FP16) 性能
- 16.3 TIPS[1] 的 FP 并行计算性能（通过与独立的整数执行单元并行执行）
- 130.5 Tensor TFLOPS[1,2]

---

[1] 基于 GPU 加速频率。

[2] FP16 矩阵数学和 FP16 累加运算。

[3] 如需了解 RTX-OPS 详情，请参阅附录 C：RTX-OPS 说明。

- 100 亿条光线/秒
- 84 Tera RTX-OPS[3]

下节将会简要介绍 Turing 的重大创新。本白皮书将对各个方面作更详细的描述。

## 新型流式多元处理器 (SM)

Turing 采用一种新型处理器架构 Turing SM，能够大幅提升着色效率，且相较 Pascal 系列，能使每个 CUDA 核心提供的性能提升 50%。这些改进得益于两项关键的架构变更。其一，Turing SM 添加了新的独立整型数据通道，可与浮点数据通道同时执行指令。在前几代中，执行这些指令将会阻止浮点指令的发出。其二，SM 存储路径已经过重新设计，能够将共享内存、纹理缓存和内存负载缓存整合到一个单元中。对于普通工作负载而言，这可为 L1 缓存提供 2 倍以上的带宽和可用容量。

## Turing Tensor 核心

Tensor 核心是专为执行张量或矩阵运算而设计的专用执行单元，而这些运算正是深度学习所采用的核心计算函数。与 Volta Tensor 核心类似，Turing Tensor 核心也能够大幅加速处于深度学习神经网络训练和推理运算核心的矩阵计算。Turing GPU 拥有已针对推理作出改进的新型 Tensor 核心设计。Turing Tensor 核心为推理工作负载加入新的 INT8 和 INT4 精度模式，可容许量化且无需 FP16 精度。Turing Tensor 核心首次为 GeForce 游戏 PC 和基于 Quadro 的工作站引入了基于深度学习的全新 AI 功能。Tensor 核心可为一项新技术“深度学习超级采样 (DLSS)”提供动力支持。DLSS 利用深度神经网络来提取渲染场景的多维特征，并能以智能方式结合多帧细节，从而构建高质量的最终图像。与传统技术（如 TAA）相比，DLSS 使用更少的输入样本，同时还能避免此类技术在透明度和其他复杂场景元素方面遭遇的算法难题。

# 实时光线追踪加速

Turing 采用实时光线追踪，能使单个 GPU 凭借对阴影、反射和折射物理属性的准确把握，渲染十分逼真的 3D 游戏和复杂的专业模型。Turing 的新型 RT 核心能够加速光线追踪并可用于多个系统和接口，如 NVIDIA RTX 光线追踪技术和 Microsoft DXR、NVIDIA OptiX™ 以及 Vulkan 光线追踪等 API，以提供实时光线追踪体验。

# 着色技术的全新进展

## 网格着色

通过为图形流水线的顶点、曲面细分和几何着色阶段提供新着色器模型，网格着色能够改进 NVIDIA 的几何处理架构，进而支持更加灵活高效的几何计算方法。例如，这种更灵活的模型通过采用高度并行的 GPU 网格着色程序，以消除 CPU 在处理物体列表时的关键性能瓶颈，从而能为每个场景实现高出一个数量级的物体着色支持。此外，网格着色还能为高级几何合成和物体细节级别 (LOD) 管理提供新算法。

## 可变速率着色 (VRS)

VRS 允许开发者动态控制着色速率，使其既可每 16 像素只进行一次着色，也可对每个像素进行多达 8 次着色。该款应用程序使用着色率表面和每个基元（三角形）值的组合来指定着色率。VRS 是一款非常强大的工具，能够帮助开发者更有效地进行着色处理，在全分辨率着色不会为图像质量带来任何明显提升的屏幕区域减少着色工作量，进而提高帧速率。我们已找到几种基于 VRS 的算法，从而能够根据内容的细节级别（内容自适应着色）、内容移动速率（移动自适应着色）并针对 VR 应用程序、镜头分辨率和眼睛位置（注视点渲染），采用不同的算法。

## 纹理空间着色

纹理空间着色是在独立坐标空间（纹理空间）中对物体进行着色，并会将其存储至内存中，且像素着色器会从该空间采样，而非直接评估结果。拥有此能力后，开发者可将着色

结果缓存至内存并重复使用结果或对其重新采样，由此消除重复的着色工作或使用不同的采样方法来提升质量。

### 多视图渲染 (MVR)

MVR 可有力扩展 Pascal 单遍立体 (SPS) 技术。SPS 仅能渲染两个除 X 偏移量之外相同的视图，而 MVR 可支持单遍渲染多个视图，即使视图基于完全不同的原点位置或视图方向时也是如此。您可通过一个简单的编程模型使用此项技术，在该模型中，编译器会自动提取与视图无关的代码，同时会识别视图相依属性以获得理想的执行效果。

## 用于图形的深度学习功能

NVIDIA NGX™ 是 NVIDIA RTX 技术中一种基于深度学习的新型神经图形框架。NVIDIA NGX 利用深度神经网络 (DNN) 和一套“神经服务”执行基于 AI 的功能，以此加速并增强图形、渲染和其他客户端应用。NGX 采用 Turing Tensor 核心执行基于深度学习的运算，并能加速向最终用户直接交付 NVIDIA 深度学习研究。NGX 包括以下功能：超高品质 NGX DLSS（深度学习超级采样）、AI InPainting 内容感知图像替换、AI Slow-Mo 超高质量且顺畅无阻的慢动作，以及 AI Super Rez 智能分辨率调整。

## 用于推理的深度学习功能

Turing GPU 可提供卓越的推理性能。凭借 Turing Tensor 核心以及 TensorRT（NVIDIA 运行时推理框架）、CUDA 和 CuDNN 库的持续改进，Turing GPU 可为推理应用程序提供出众性能。Turing Tensor 核心还支持快速 INT8 矩阵运算，能够在维持超低精度损失的同时大幅加速推理吞吐效率。此外，Turing Tensor 核心现还可实现全新低精度 INT4 矩阵运算，为研发 8 位子神经网络提供支持。

## GDDR6 高性能显存子系统

Turing 是首款支持 GDDR6 显存的 GPU 架构。GDDR6 是高带宽 GDDR DRAM 内存设计的又一次重大飞跃。Turing GPU 中的 GDDR6 存储器接口电路经过全面重新设计，在速度、

能效和降噪方面均实现了提升；与 Pascal GPU 中使用的 GDDR5X 显存相比，GDDR6 可达到 14 Gbps 的传输速率，并将能效提高 20%。

## 第二代 NVIDIA NVLink

Turing TU102 和 TU104 GPU 采用 NVIDIA NVLink™ 高速互联技术，可在数对 Turing GPU 之间提供可靠的高带宽和低延迟连通性。NVLink 具有高达 100GB/秒的双向带宽，能够使定制工作负载在两个 GPU 之间进行高效分割并共享内存容量。对于游戏工作负载，NVLink 所增加的带宽及其专用 GPU 间通道可为 SLI 提供多种新功能，例如增加新模式或更高分辨率显示配置。对于大内存工作负载（包括专业光线追踪应用程序），NVLink 可在两个 GPU 的帧缓存间分割场景数据，提供高达 96 GB 的共享帧缓存（两块 48 GB Quadro RTX 8000 GPU），且硬件会基于内存分配位置将内存请求自动传送至正确的 GPU。

## USB-C 和 VirtualLink

Turing GPU 为 USB Type-C™ 和 VirtualLink™ 增添了硬件支持 [4]。VirtualLink 是全新的开放式行业标准，旨在通过单个 USB-C 接口满足新一代 VR 头盔的电源、显示和带宽要求。VirtualLink 不仅能简化当前设置 VR 头盔的繁琐流程，还将为更多设备引入 VR。

[4] 为兼容最新的 VirtualLink 标准，Turing GPU 已根据 "VirtualLink Advance Overview" 实施了硬件支持。如需详细了解 VirtualLink，请参阅 http://www.virtuallink.org.

# 深入了解 TURING GPU 架构

Turing TU102 GPU 在 Turing GPU 系列中性能最为出众，我们将在本节重点介绍。TU104 和 TU106 GPU 的基本架构与 TU102 相同，但已针对不同的使用模型和细分市场缩小至相应等级。有关 TU104 和 TU106 芯片架构和目标用途（市场）的详细信息，我们已列于附录 A：Turing TU104 GPU 和附录 B：Turing TU106 GPU。

## TURING TU102 GPU

TU102 GPU 包含 6 个图像处理集群 (GPC)、36 个纹理处理集群 (TPC) 和 72 个流式多元处理器 (SM)。（请参见图 2，了解含 72 个 SM 单元的完整 TU102 GPU 的内部构造。）每个 GPC 均包含一个专用的光栅化引擎和 6 个 TPC，且每个 TPC 均包含两个 SM。每个 SM 包含 64 个 CUDA 核心、8 个 Tensor 核心、1 个 256 KB 寄存器堆、4 个纹理单元以及 96 KB 的 L1 或共享内存，且我们可根据计算或图形工作负载将这些内存设置为不同容量。

每个 SM 中的全新 RT 核心处理引擎负责执行光线追踪加速（Turing 光线追踪技术一节对 RT 核心和光线追踪功能作出深入探讨）。

完整的 TU102 GPU 架构包括以下构件：

- 4608 个 CUDA 核心
- 72 个 RT 核心

- 576 个 Tensor 核心
- 288 个纹理单元
- 12 个 32 位 GDDR6 显存控制器（共 384 位）。

每个显存控制器均附有 8 个光栅化处理单元 (ROP) 单元和 512 KB 的 L2 缓存。完整的 TU102 GPU 由 96 个 ROP 单元和 6144 KB 的 L2 缓存组成。请参见图 3，了解 Turing TU102 GPU 的内部构造。表 1 对 Pascal GP102 和 Turing TU102 的 GPU 特性作出了对比。


**注意：**TU102 GPU 还包含 144 个 FP64 单元（每个 SM 分有两个），但并未在图中呈现出来。FP64 TFLOP 速率是 FP32 运算 TFLOP 速率的 1/32。TU102 GPU 包含少量 FP64 硬件单元，旨在确保任何带有 FP64 代码的程序都能正确运行。

## 图 2. 含 72 个 SM 单元的完整 Turing TU102 GPU 内部构造

表 1. NVIDIA Pascal GP102 与 Turing TU102 对比表

| GPU 特性 | GTX 1080Ti | RTX 2080 Ti | Quadro P6000 | Quadro RTX 6000 |
|---|---|---|---|---|
| 架构 | Pascal | Turing | Pascal | Turing |
| GPC 数量 | 6 | 6 | 6 | 6 |
| TPC 数量 | 28 | 34 | 30 | 36 |
| SM 数量 | 28 | 68 | 30 | 72 |
| CUDA 核心数/SM | 128 | 64 | 128 | 64 |
| CUDA 核心数/GPU | 3584 | 4352 | 3840 | 4608 |
| Tensor 核心数/SM | 不适用 | 8 | 不适用 | 8 |
| Tensor 核心数/GPU | 不适用 | 544 | 不适用 | 576 |
| RT 核心数 | 不适用 | 68 | 不适用 | 72 |
| GPU 基础频率 (MHz)（参考版本：Founders Edition） | 1480/1480 | 1350/1350 | 1506 | 1455 |
| GPU 加速频率 (MHz)（参考版本：Founders Edition） | 1582/1582 | 1545/1635 | 1645 | 1770 |
| RTX-OPS (Tera-OPS)（参考版本：Founders Edition） | 11.3/11.3 | 76/78 | 不适用 | 84 |
| 光线投射数量（10 亿条光线/秒）（参考版本：Founders Edition） | 1.1/1.1 | 10/10 | 不适用 | 10 |
| FP32 TFLOPS 峰值*（参考版本：Founders Edition） | 11.3/11.3 | 13.4/14.2 | 12.6 | 16.3 |
| INT32 TIPS 峰值*（参考版本：Founders Edition） | 不适用 | 13.4/14.2 | 不适用 | 16.3 |

| GPU 特性 | GTX 1080Ti | RTX 2080 Ti | Quadro P6000 | Quadro RTX 6000 |
|---|---|---|---|---|
| FP16 TFLOPS 峰值*（参考版本：Founders Edition） | 不适用 | 26.9/28.5 | 不适用 | 32.6 |
| 使用 FP16 累加的 FP16 Tensor TFLOPS 峰值*（参考版本：Founders Edition） | 不适用 | 107.6/113.8 | 不适用 | 130.5 |
| 使用 FP32 累加的 FP16 Tensor TFLOPS 峰值*（参考版本：Founders Edition） | 不适用 | 53.8/56.9 | 不适用 | 130.5 |
| INT8 Tensor TOPS 峰值*（参考版本：Founders Edition） | 不适用 | 215.2/227.7 | 不适用 | 261.0 |
| INT4 Tensor TOPS 峰值*（参考版本：Founders Edition） | 不适用 | 430.3/455.4 | 不适用 | 522.0 |
| 帧缓存大小和类型 | 11264 MB GDDR5X | 11264 MB GDDR6 | 24576 MB GDDR5X | 24576 MB GDDR6 |
| 显存位宽 | 352 位 | 352 位 | 384 位 | 384 位 |
| 显存频率（数据速率） | 11 Gbps | 14 Gbps | 9 Gbps | 14 Gbps |
| 显存带宽（GB/秒） | 484 | 616 | 432 | 672 |
| ROP 数量 | 88 | 88 | 96 | 96 |
| 纹理单元数量 | 224 | 272 | 240 | 288 |
| 纹素填充率（10 亿纹素/秒） | 354.4/354.4 | 420.2/444.7 | 395 | 510 |
| L2 缓存大小 | 2816 KB | 5632 KB | 3072 KB | 6144 KB |
| 寄存器堆大小/SM | 256 KB | 256 KB | 256 KB | 256 KB |
| 寄存器堆大小/GPU | 7168 KB | 17408 KB | 7680 KB | 18432 KB |

| GPU 特性 | GTX 1080Ti | RTX 2080 Ti | Quadro P6000 | Quadro RTX 6000 |
|---|---|---|---|---|
| 热设计功耗 (TDP)*（参考版本：Founders Edition） | 250/250 W | 250/260 W | 250 W | 260 W |
| 晶体管数 | 120 亿 | 186 亿 | 120 亿 | 186 亿 |
| 芯片大小 | 471 | 754 | 471 | 754 |
| 制造工艺 | 16 nm | 12 nm FFN | 16 nm | 12 nm FFN |

**注意**：*TFLOPS、TIPS 和 TOPS 峰值速率基于 GPU 加速频率。

**功率图只表示显卡的 TDP。注意，使用 VirtualLink™ 或 USB Type-C™ 接口另需多达 35 瓦功率，此功率图中并未予以显示。

随着 GPU 加速计算变得越来越流行，搭载多块 GPU 的系统正越来越多地部署在服务器、工作站和超级计算机中。TU102 和 TU104 GPU 采用第二代 NVIDIA NVLink™ 高速互联技术，该技术最初设计用于 Volta GV100 GPU，可为 SLI 和其他多 GPU 用例提供高速多 GPU 连接性能。NVLink 允许每块 GPU 直接访问与之连接的其他 GPU 的显存，可加速 GPU 间的通信；NVLink 还能合并多个 GPU 的显存，从而为更大数据集和更快速的内存计算提供支持。

TU102 包含两个 NVLink x8 链路，每个链路在每个方向上均可提供高达 25 GB/秒的传输带宽，总计双向带宽可达 100 GB/秒（请参见图 3）。

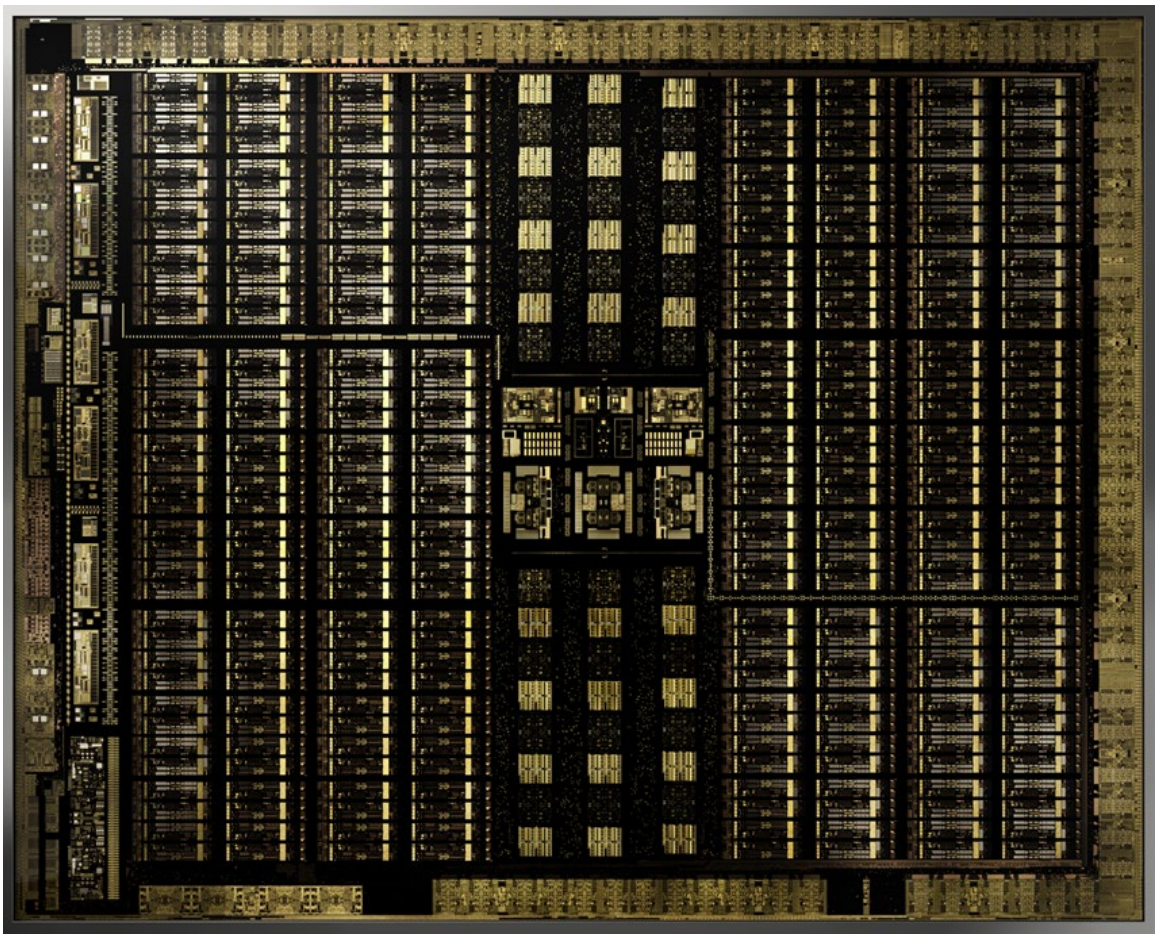


图 3. NVIDIA Turing TU102 GPU

# TURING 流式多元处理器 (SM) 架构

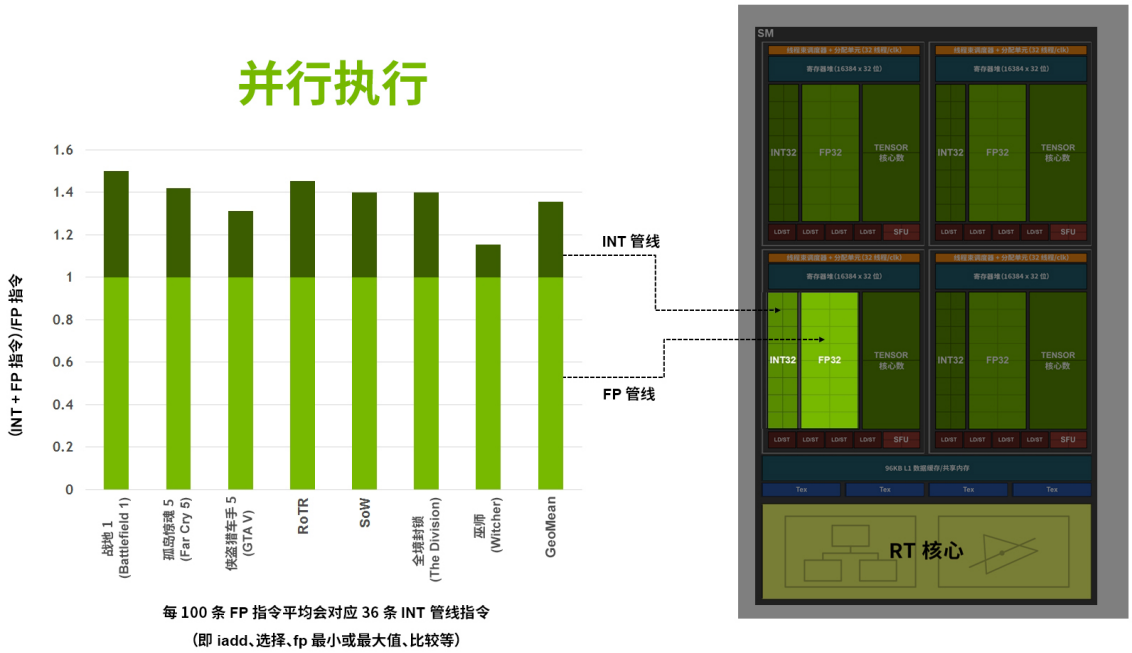
Turing 架构采用全新 SM 设计，其中包含我们在 Volta GV100 SM 架构中引入的众多特性。每个 TPC 均包含两个 SM，每个 SM 共有 64 个 FP32 核心和 64 个 INT32 核心。相比之下，Pascal GP10x GPU 的每个 TPC 仅有一个 SM，且每个 SM 只含 128 个 FP32 核心。Turing SM 支持并行执行 FP32 与 INT32 运算（详情如下），并可执行类似于 Volta GV100 GPU 的独立线程调度。每个 Turing SM 还拥有 8 个混合精度 Turing Tensor 核心（Turing Tensor 核心一节将对此作出详细说明）和 1 个 RT 核心（Turing 光线追踪技术一节对 RT 核心的具体功能作出详细介绍）。请参见图 4，了解 Turing TU102、TU104 和 TU106 SM 的内部构造。

我们将 Turing SM 划分为四个处理块，每个处理块均包含 16 个 FP32 核心、16 个 INT32 核心、2 个 Tensor 核心、1 个线程束调度器和 1 个分配单元。每个处理块还具有一个新型 L0 指令缓存和一个 64 KB 寄存器堆。这四个处理块共享一个组合式 96 KB L1 数据缓存或共享内存。传统的图形工作负载将 96 KB 的 L1 缓存或共享内存划分为 64 KB 专用图形着色器 RAM 和 32 KB 纹理缓存和寄存器堆溢出区。计算工作负载可将 96 KB 划分为 32 KB 共享内存和 64 KB L1 缓存，或分为 64 KB 共享内存和 32 KB L1 缓存。

Turing 对核心执行数据通道作出了重大改进。现代着色器工作负载通常会混合包含 FP 算术指令（如 FADD 或 FMAD）和更简单的指令（如用于寻址和获取数据的整数加法、浮点比较或处理结果的最小及最大值等）。在以往的着色器架构中，每当运行其中一个非 FP 数学指令时，浮点数据通道就会停止运作。Turing 通过在每个 CUDA 核心旁添加第二个并行执行单元，使之能与浮点数据通道并行执行这些指令。

图 5 显示整数流水线与浮点指令的混合执行结果有所不同，但在几款现代应用程序中，每 100 个浮点指令通常约可与 36 个额外的整数流水线指令并行执行。通过将这些指令移至单独的流水线，浮点将能额外获得 36% 的有效吞吐量。


图 4. Turing TU102/TU104/TU106 流式多元处理器 (SM)




通过分析众多工作负载，我们可以发现每 100 次浮点运算平均可与 36 次整数运算并行执行。

## 图 5. 浮点指令和整数指令在 Turing SM 中的并行执行结果

Turing SM 还为共享内存、L1 和纹理缓存引入了新型统一架构。这种统一设计允许 L1 缓存充分利用资源，进而将每 TPC 的命中带宽提升至 Pascal 的 2 倍；当共享内存分配未使用全部共享内存容量时，该架构还允许对 L1 缓存进行重新配置以增大其容量。Turing L1 缓存大小可增至 64 KB，使每个 SM 可分配到 32 KB 的共享内存；其也可减至 32 KB，从而为共享内存预留 64 KB 的分配量。此外，我们也已为 Turing 增加 L2 缓存容量。

图 6 显示 L1 数据缓存和 Turing SM 共享内存子系统的全新组合如何大幅提升性能，同时简化编程并减少实现应用程序性能峰值或接近峰值所需的调整。与以往 Pascal GPU 中使用的 L1 缓存实现相比，将 L1 数据缓存与共享内存相结合可以减少延迟并提供更高的带宽。

总而言之，SM 的变更可使 Turing 每个 CUDA 核心提供的性能提升 50%。图 7 显示当前游戏应用程序中一组着色器工作负载的处理结果。

2 倍 L1 带宽

L1 命中延迟降低

高达 2.7 倍的 L1 容量

2 倍 L2 容量

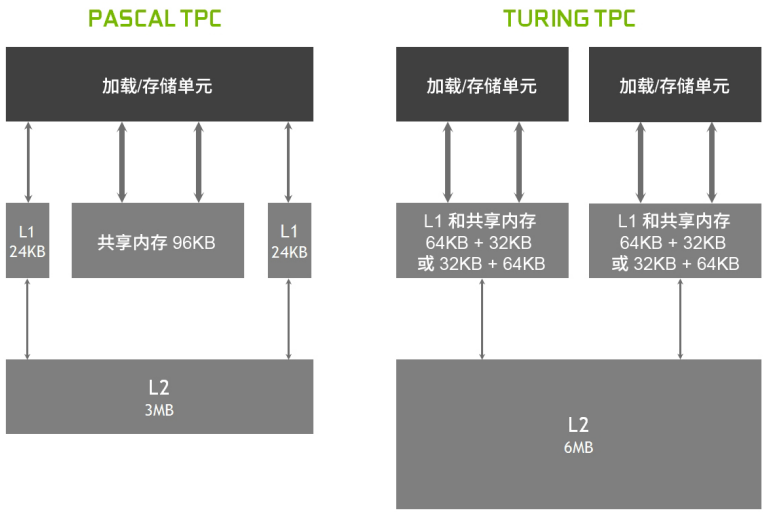



图 6. 新的共享内存架构

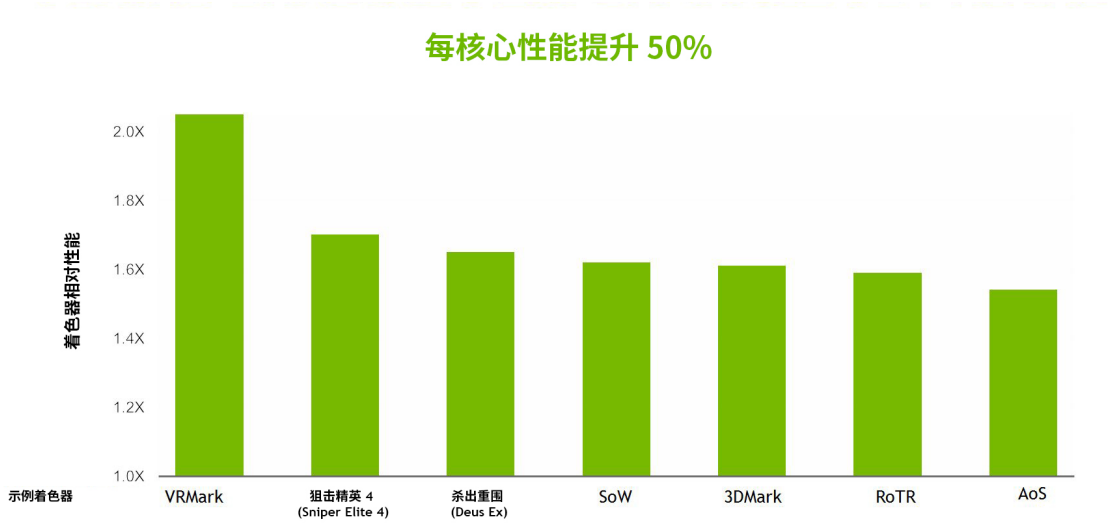



图 7. 在众多不同工作负载上，Turing 相较 Pascal 的着色性能提升情况。

# Turing Tensor 核心

Turing GPU 包括 Volta GV100 GPU 中首次采用的增强版 Tensor 核心。Turing Tensor 核心设计添加了 INT8 和 INT4 精度模式，专用于推理可容许量化的工作负载。Turing Tensor 核心还能完全支持 FP16，以便处理具有更高精度要求的工作负载。

通过将 Tensor 核心引入基于 Turing 的 GeForce 游戏 GPU，我们将能为游戏应用程序首次部署实时深度学习。Turing Tensor 核心能够加速 NVIDIA NGX 神经服务中基于 AI 的特性，以此增强图形、渲染和其他类型的客户端应用。NGX AI 特性示例包括深度学习超级采样 (DLSS)、AI InPainting、AI Super Rez 和 AI Slow-Mo。详情请参见 NVIDIA NGX 技术小节（39 页）。

Turing Tensor 核心能够为处于神经网络训练和推理函数中心的矩阵乘法运算实现加速。Turing Tensor 核心尤其擅长推理计算，在这类计算中，经过训练的深度神经网络 (DNN) 可基于给定输入，推理并传递相关的实用信息。推理示例包括识别 Facebook 照片中的好友图像，识别及分类不同类型的汽车、行人和自动驾驶汽车所面临的道路危险，实时翻译人类语言，以及在网上零售和社交媒体系统中创建个性化的用户推荐。

一块 TU102 GPU 共包含 576 个 Tensor 核心，其中每个 SM 分有 8 个核心，SM 内的每个处理块分有 2 个核心。每个 Tensor 核心可使用 FP16 输入，在每个时钟周期内执行多达 64 次浮点混合乘加 (FMA) 运算。每个 SM 中的 8 个 Tensor 核心可在每个时钟周期内总共执行 512 次 FP16 乘积累加运算，或在每个时钟周期内共计执行 1024 次 FP 运算。新添加的 INT8 精度模式在运作时要比该速度快一倍，即能在每个时钟周期内执行 2048 次整数运算。

Turing Tensor 核心能够显著加速矩阵运算，不仅适用于新的神经图形函数，还可用于深度学习训练和推理运算。如需详细了解 Tensor 核心的基本运算详情，请参见 NVIDIA Tesla V100 GPU 架构白皮书。

图 8 展示了能为 AI 推理提供多精度模式的全新 Turing Tensor 核心。

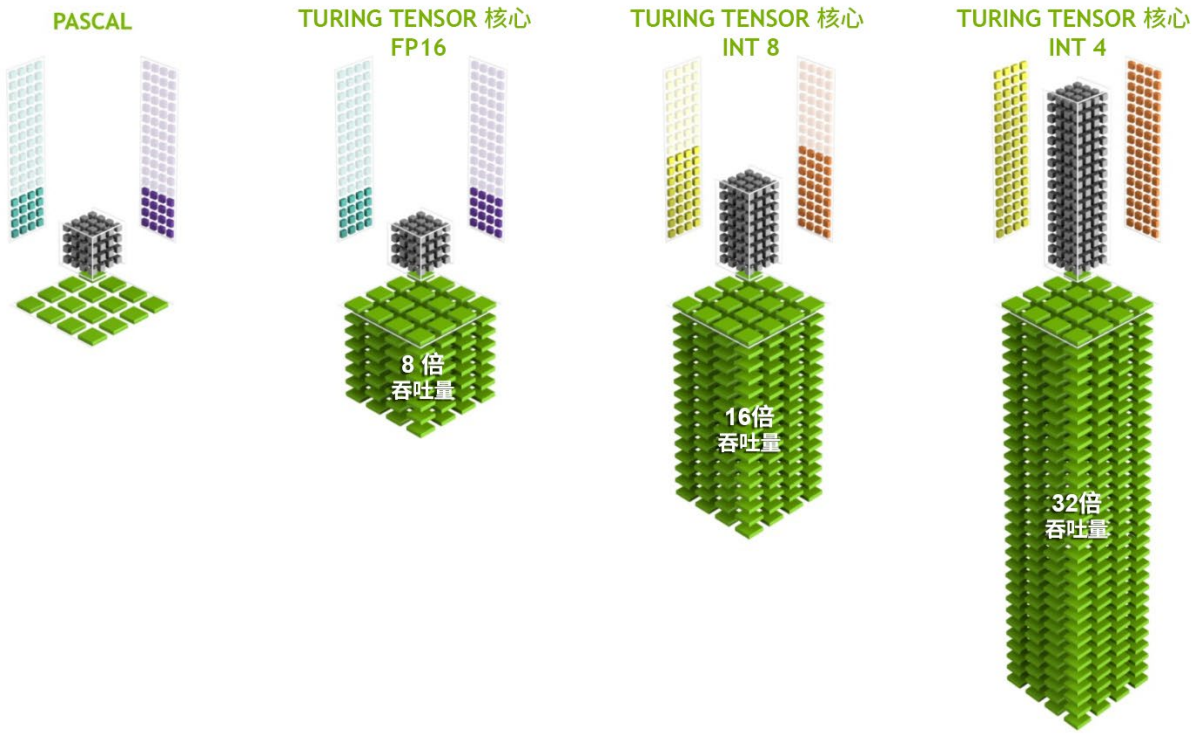



图 8. 全新 Turing Tensor 核心可为 AI 推理提供多精度模式

# TURING 已针对数据中心应用程序实现优化

NVIDIA GPU 已成为深度学习训练的标准行业解决方案，而基于 GPU 的推理正越来越受关注，其采用率也在迅速攀升。目前，全球众多领先企业均采用 NVIDIA GPU 在其数据中心和终端设备上运行推理应用程序。许多传统上在 CPU 运行推理应用程序的企业现都转为采用 NVIDIA GPU，并在大幅减少投入的同时获得了惊人的性能提升。例如，与超大型数据中心基于 CPU 的服务器相比，Pascal 架构上基于 NVIDIA Tesla® P4 GPU 的推理可提供 10 倍的行业领先推理性能，以及 25 倍的能效 [5]。作为首个基于 Turing 的 GPU，NVIDIA Tesla T4 GPU 进一步扩大了这一领先优势，并可提供突破性性能和灵活的多精度功能，从 FP32、FP16 到 INT8 以及 INT4 均可涵盖。

[5] 与英特尔至强金牌 6140 相比，对后者使用的是英特尔深度学习部署工具和 Resnet-50。

NVIDIA Tesla T4 是针对超大规模数据中心的新兴精尖推理解决方案，可为图像分类与标记、视频分析、自然语言处理、自动语音识别以及智能搜索等各类应用提供通用推理加速。Tesla T4 的广泛推理能力使其能够应用于企业解决方案和终端设备。

NVIDIA Tesla T4 GPU 具有 2560 个 CUDA 核心和 320 个 Tensor 核心，可提供高达 130 TOPS（万亿次运算/秒）的 INT8 运算和多达 260 TOPS 的 INT4 推理性能（请参见附录 A：Turing TU104 GPU，了解更多 Tesla T4 规格信息）。与基于 CPU 的推理相比，由全新 Turing Tensor 核心驱动的 Tesla T4 可提供高达 近 40 倍的推理性能[6]（请参见图 9）。

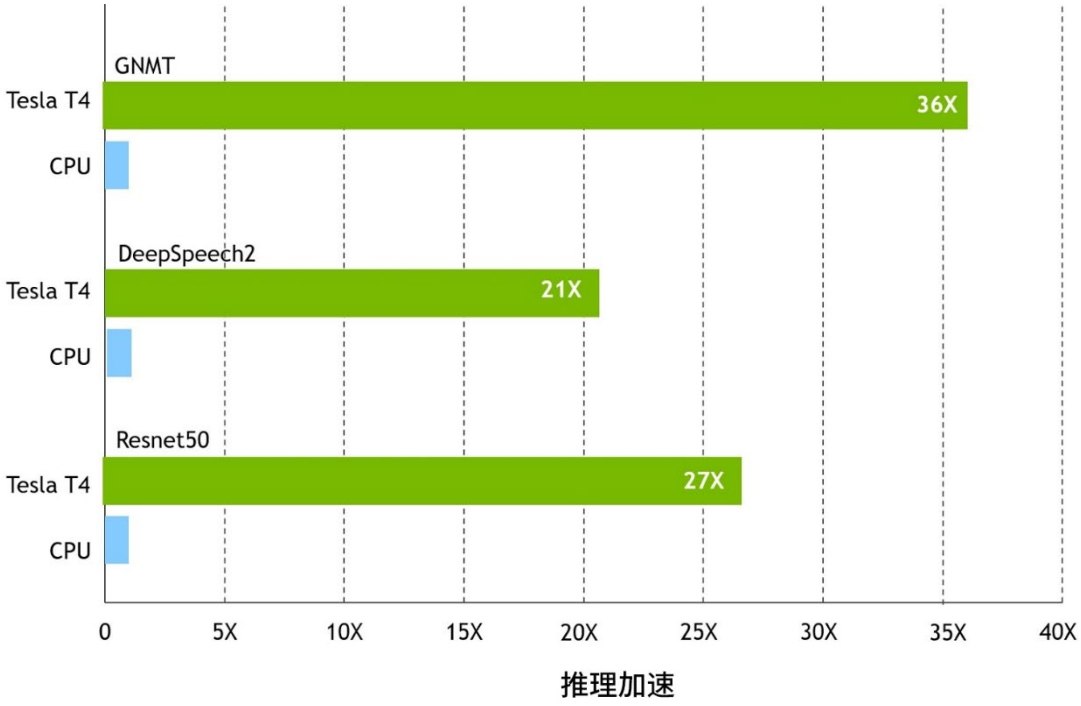
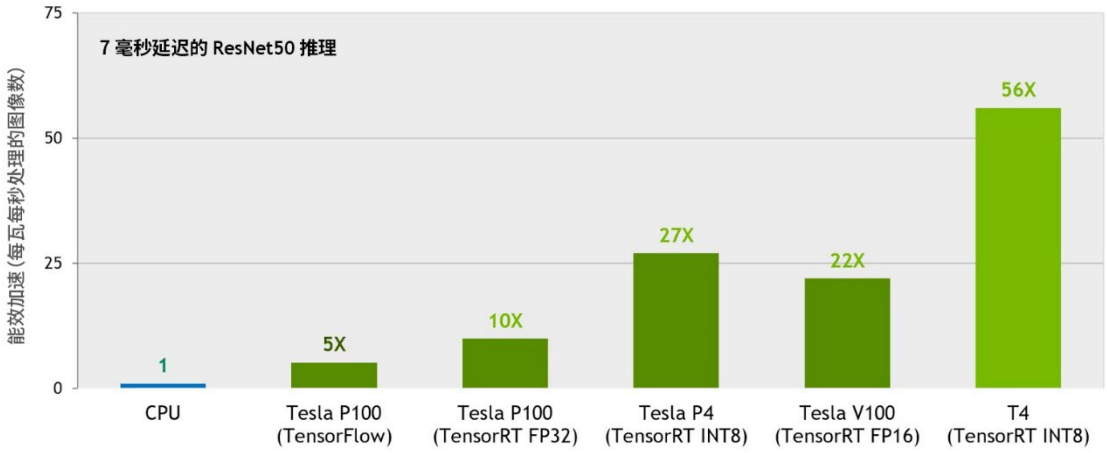



图 9. Tesla T4 可提供约 40 倍的推理性能

[6] 最大批量下，Resnet-50 推理吞吐量的延迟低于 7 毫秒。基于使用英特尔 OpenVino 的英特尔 Skylake 6140 衡量出的 CPU 性能。基于使用 TensorRT5.0 的 Tesla T4 衡量出的 GPU 性能。

能效对于数据中心至关重要，Tesla T4 的能效高达 CPU 推理的 50 多倍，更比 NVIDIA 上一代 Tesla P4 GPU 高出一倍 [7]（请参见图 10）。




## 图 10. Tesla T4 的能效高达 CPU 推理的 50 多倍

Turing GPU 架构不仅配备 Turing Tensor 核心，还具备有助提高数据中心应用性能的其他特性。其中一些主要特性包括：

- 增强版视频引擎

  与之前的 Pascal 和 Volta GPU 架构相比，Turing 能够支持更多的视频解码格式，如 HEVC 4:4:4（8/10/12 位）和 VP9（10/12 位）（请参见视频和显示引擎一节，我们已自 25 页起提供详细介绍）。相比基于 Pascal 的同等 Tesla GPU，Turing 的增强版视频引擎能够大幅提升并发视频流的解码数量（请参见表 2）。

- Turing 多进程服务

  Turing GPU 架构继承了 Volta 架构中首次采用的增强版多进程服务 (MPS) 特性。相比基于 Pascal 的 Tesla GPU，Tesla T4 所采用的 MPS 能够针对小批量改进推理性能，减少启动延迟，提高服务质量，并能为更多并发客户端请求提供支持服务。

[7] SKL CPU 至强金牌 6140 通过英特尔深度学习部署工具衡量性能，其并未实现 7 毫秒延迟。如上所述，GPU 性能由 TensorFlow 或 TensorRT 进行衡量。NVIDIA Tesla T4 的性能为初步预测结果，如有变更，恕不另行通知。

- 更高的显存带宽和更大的显存容量

  凭借 16 GB 的 GPU 显存和 320 GB/秒的显存带宽，Tesla T4 能够提供几乎两倍于其前代 Tesla P4 GPU 的显存带宽和显存容量。凭借 Tesla T4，超大规模数据中心可以将虚拟桌面基础架构 (VDI) 应用程序的用户密度提高一倍。

表 2. 增强版视频引擎，Tesla P4 与 Tesla T4 对比表

| | Tesla P4 (70 W TDP) | Tesla T4 (70 W TDP) |
|---|---|---|
| H264 解码 (1080p30) | 16 路视频流 | 32 路视频流 |
| HEVC 解码 (1080p30) | 16 路视频流 | 44 路视频流 |
| VP9 解码 (1080p30) | 16 路视频流 | 32 路视频流 |

Turing 不仅能为高端游戏和专业图形注入革命性的全新特性，还可提供多精度计算等新功能，并能显著提高数据中心的性能与能效。随着 NVIDIA 深度学习平台其他功能的不断改进（如新发布的 TensorRT 5.0 和 CUDA 10），基于 NVIDIA GPU 的推理解决方案将能大幅减小数据中心的成本、规模和功耗。

# TURING 内存架构和显示特性

本节将会更深入地探讨最重要的新内存层次结构以及 Turing 架构的显示子系统特性。

内存子系统性能对于应用程序加速至关重要。Turing 已改进主内存、缓存内存和压缩架构，能够增加内存带宽并减少访问延迟。GPU 计算特性经改进和增强后，有助加速游戏及众多计算密集型应用程序和算法。全新的显示功能和视频编码及解码功能可支持更高分辨率和 HDR 显示器、更先进的 VR 显示器、日益提升的数据中心视频流要求、8K 视频制作以及其他视频相关应用程序。我们将详细讨论以下特性：

- GDDR6 显存子系统
- L2 缓存和 ROP
- Turing 显存压缩

- 视频和显示引擎
- USB-C 和 VirtualLink

## GDDR6 显存子系统

随着显示器分辨率不断提高，着色器功能和渲染技术变得愈加复杂，显存带宽和容量将对 GPU 性能发挥更大的作用。为尽可能保持最高帧速率和计算速度，GPU 不仅需要更多显存带宽，还需要巨大的显存容量以提供持续性能。

NVIDIA 曾与 DRAM 行业紧密合作，共同开发出全球首款使用 HBM2 和 GDDR5X 显存的 GPU。现如今，Turing 已成为首个采用 GDDR6 显存的 GPU 架构。

GDDR6 是高带宽 GDDR DRAM 内存设计的又一次重大飞跃。凭借众多高速 SerDes 和 RF 技术带来的改进，Turing GPU 中的 GDDR6 存储器接口电路已实现全面重新设计，在速度、能效和降噪方面均得到了提升。这一新型接口设计采用多个新电路并能提升信号训练效果，从而大幅降低由工艺、温度和电源电压引起的噪声和波动。该显存系统可在利用率较低的时段，大量采用时钟门控以显著降低功耗，从而大幅提升整体能效。与 Pascal GPU 中所用的 GDDR5X 显存相比，Turing 的 GDDR6 显存子系统可提供 14 Gbps 的信号传输速率，并将能效提升 20%。

实现这一加速需要进行端到端优化。通过使用广泛的信号和电源完整性模拟，NVIDIA 已为 Turing 精心打造出独有的封装和电路板设计，从而满足更高的速度要求。举例来说，该设计已将信号串扰率降低 40%，而信号串扰正是对大型内存系统造成的最严重损害之一。

为达到 14 Gbps 的传输速度，我们已对内存子系统的各个方面进行精心设计，以满足实现此种高频运作所需的严苛标准。设计期间，我们仔细优化了每个信号，旨在提供尽可能清晰的内存接口信号（请参见图 11.）。

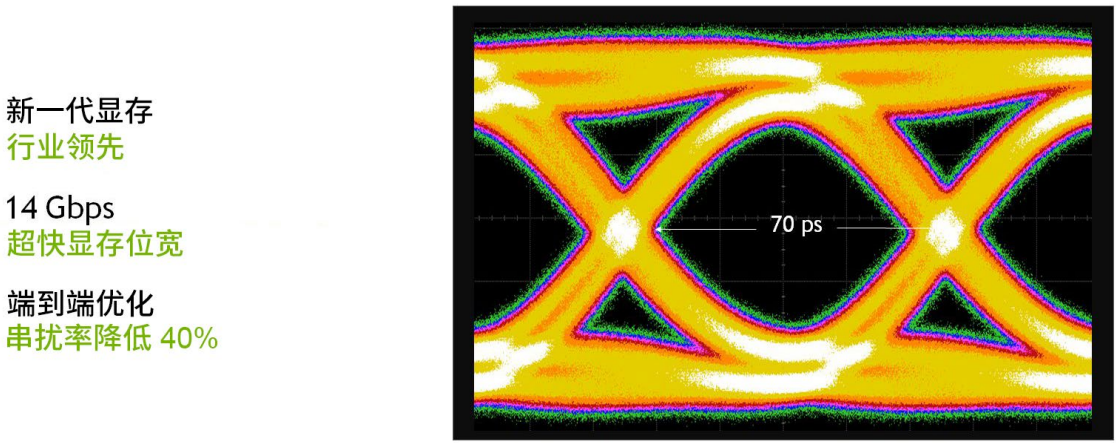



图 11. Turing GDDR6

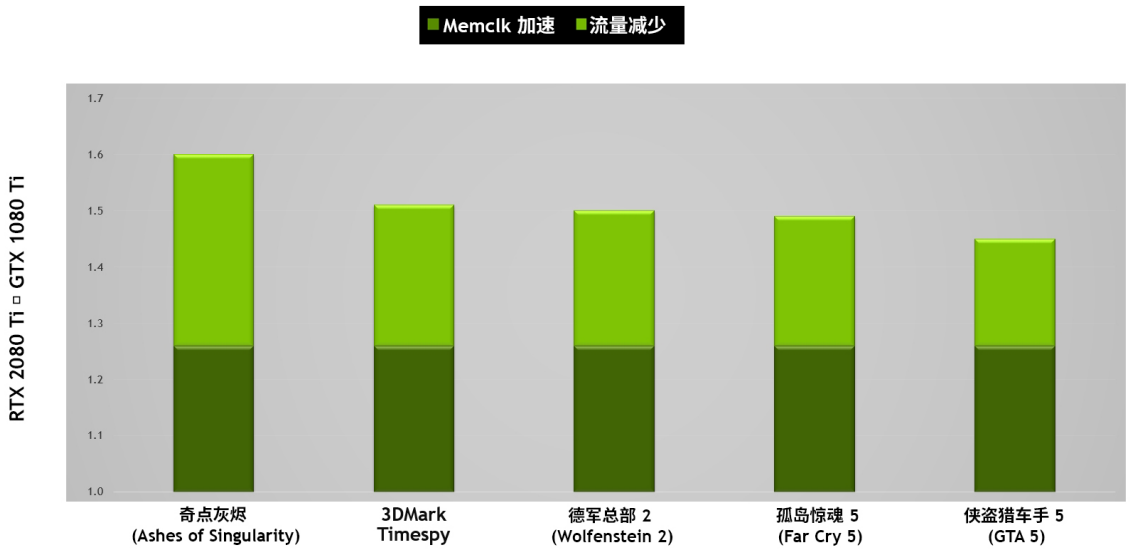
## L2 缓存和 ROP

除配备新的 GDDR6 显存子系统以外，Turing GPU 还已添加更大容量且更快速的 L2 缓存。TU102 GPU 附带 6 MB L2 缓存，相比 TITAN Xp 中使用的上一代 GP102 GPU 所提供的 3 MB L2 缓存，其已高出一倍。TU102 还可提供远高于 GP102 的 L2 缓存带宽。

与上一代 NVIDIA GPU 类似，Turing 中的每个 ROP 分区均包含 8 个 ROP 单元，且每个单元能够处理一个单色样本。一个完整的 TU102 芯片包含 12 个 ROP 分区，共计 96 个 ROP 单元。

## Turing 显存压缩

NVIDIA GPU 使用几种无损显存压缩技术，旨在将数据写入帧缓存时降低显存带宽需求。GPU 的压缩引擎采用各类不同算法，能够根据数据特点确定最有效的压缩方式。这有助减少写入显存及从显存传输至 L2 缓存的数据量，并能降低客户端（如纹理单元）和帧缓存之间传输的数据量。Turing 已对 Pascal 的精尖显存压缩算法作出深入改进，不仅能增加 GDDR6 的原始数据传输速率，还可进一步提高有效带宽。如图 12 所示，原始带宽增加且流量减少会导致 Turing 的有效带宽比 Pascal 高出 50%，这对于保持架构平衡以及支持新型 Turing SM 架构提供的性能至关重要。




相较基于 Pascal GP102 的 1080 Ti，基于 Turing TU102 的 RTX 2080 Ti 在显存子系统和压缩（流量减少）方面的改进约会将有效带宽提高 50%。

图 12. 有效带宽提高 50%

## 视频和显示引擎

消费者对高分辨率显示器的需求逐年增加。例如，8K 分辨率 (7680 x 4320) 所需像素相当于 4K 分辨率 (3820 x 2160) 的 4 倍。游戏玩家和硬件迷还希望显示器能够在分辨率和刷新率上实现双提升，从而获得尽可能流畅的图像。

Turing GPU 具有专为新一波显示器设计的全新显示引擎，可支持更高分辨率、更快刷新率以及 HDR。Turing 支持 DisplayPort 1.4a，可在 60 Hz 刷新率下实现 8K 分辨率，此外还已加入 VESA 的显示串流压缩 (DSC) 1.2 技术，能够提供视觉无损的更高压缩。表 3 显示 Turing GPU 中的 DisplayPort 支持。

表 3. Turing GPU 中的 DisplayPort 支持

| | 带宽/通道 | 支持的最大分辨率 |
|---|---|---|
| DisplayPort 1.2 | 5.4 Gbps | 4K @ 60 Hz |
| DisplayPort 1.3 | 8.1 Gbps | 5K @ 60 Hz |
| DisplayPort 1.4a | 8.1 Gbps | 8K @ 60 Hz |

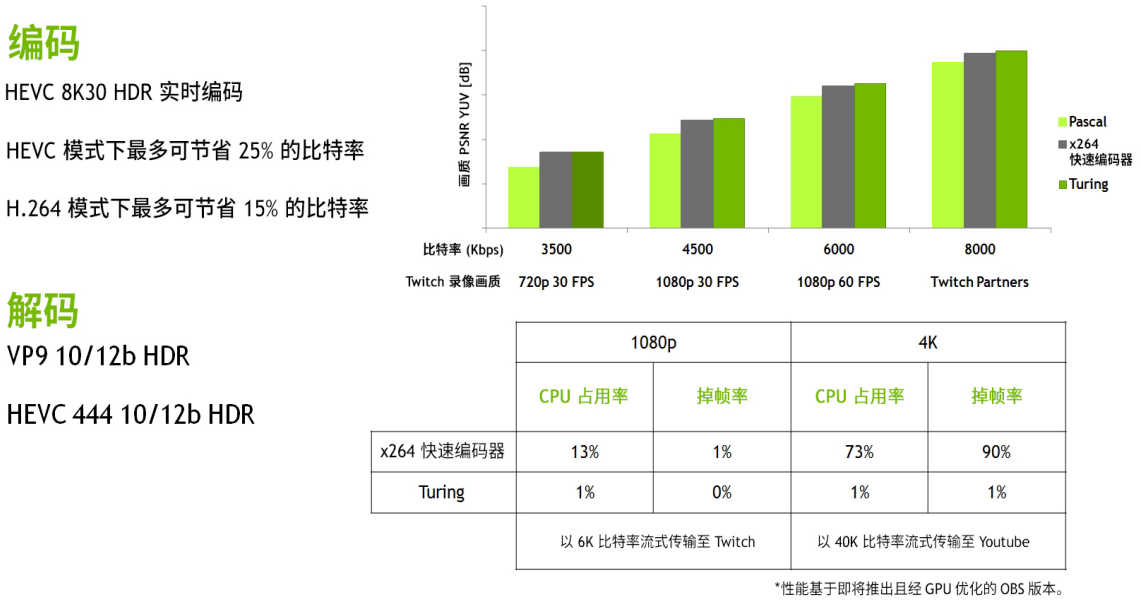
Turing GPU 可在 60 Hz 刷新率下驱动两个 8K 显示器，每个显示器通过一根线缆连接。该 GPU 也可通过 USB-C 上发送 8K 分辨率（请参见 USB-C 和 VirtualLink 一节，我们已自 27 页起提供详细介绍）。

Turing 的全新显示引擎支持在显示流水线中实施原生 HDR 处理。此外，HDR 流水线中还加入了色调映射。色调映射是一项在标准动态范围显示器上近似显示高动态范围图像的技术。Turing 支持 ITU-R 建议书 BT.2100 标准定义的色调映射公式，能够避免不同的 HDR 显示器产生色彩偏移。

Turing GPU 还附带增强版 NVENC 编码器单元，能够以 30 FPS 的帧速率支持 H.265 (HEVC) 8K 编码。新型 NVENC 编码器分别能在 HEVC 和 H.264 模式下节约高达 25% 和 15% 的比特率。

Turing 的新型 NVDEC 解码器已进行升级，现可支持解码 HEVC 4:4:4 8/10/12 位视频流；此外，与 Pascal GP102/107/108 和 Volta GV100 GPU 类似，其还可支持 VP9 10/12 位 HDR。

相较上一代 Pascal GPU 和软件编码器，Turing 已提升编码质量。图 13 显示在常见的 Twitch 和 YouTube 流设置下，Turing 的视频编码器质量优于采用快速设置的 x264 软件编码器，能够大幅降低 CPU 利用率。在典型的 CPU 设置上进行编码时，4K 流式传输会对其产生过于沉重的工作负载，但 Turing 的编码器却能突破这一障碍。




| | 1080p | | 4K | |
|---|---|---|---|---|
| | CPU 占用率 | 掉帧率 | CPU 占用率 | 掉帧率 |
| x264 快速编码器 | 13% | 1% | 73% | 90% |
| Turing | 1% | 0% | 1% | 1% |
| | 以 6K 比特率流式传输至 Twitch | | 以 40K 比特率流式传输至 Youtube | |

*性能基于即将推出且经 GPU 优化的 OBS 版本。

Turing 与 Pascal 以及快速 x264 软件编码器的新视频特性和视频质量比较

图 13. 视频特性提升情况

# USB-C 和 VIRTUALLINK

目前，在 PC 上接入 VR 头盔时需在头盔和系统之间连接多条线缆：一条显示器线缆，用于将 GPU 图像数据发送至头盔中的两个显示器；一条电源线缆，用于为头盔供电；一条 USB 连接线，用于传输摄像头流及回读头盔中的头部姿势信息（以更新 GPU 渲染的帧画面）。多条线缆会降低最终用户的舒适度，并会导致其在使用头盔时无法自如移动。为适应这些线缆，头盔制造商需要加入复杂设计，并增大头盔体积。

为解决该问题，我们在设计 Turing GPU 时已针对 USB Type-C™ 和 VirtualLink™ 添加硬件支持。VirtualLink 是一种新的开放式行业标准，涵盖领先的硅、软件和头盔制造商，并由 NVIDIA、Oculus、Valve、Microsoft 和 AMD 主导。

VirtualLink 专为满足当前和新一代 VR 头盔的连接需求而开发。VirtualLink 采用一种全新的 USB-C 替代模式，旨在通过单个 USB-C 接口提供驱动 VR 头盔所需的供电、显示和数据传输条件。

VirtualLink 可同时支持四通道高比特率 3 (HBR3) 显示接口，并能与头盔实现超高速 USB 3 连接，以便追踪运动。相比之下，USB-C 仅支持四通道 HBR3 显示接口或两通道 HBR3 显示接口与两通道超高速 USB 3 的组合。

VirtualLink 不仅能简化当前设置 VR 头盔的繁琐流程，还将为更多设备引入 VR。单一接口解决方案可将 VR 引入仅能容纳单个小体积 USB-C 接口的小型设备（如轻薄笔记本电脑）而非当今的 VR 基础设施，因为后者需要的是能够配备多个接口的 PC。

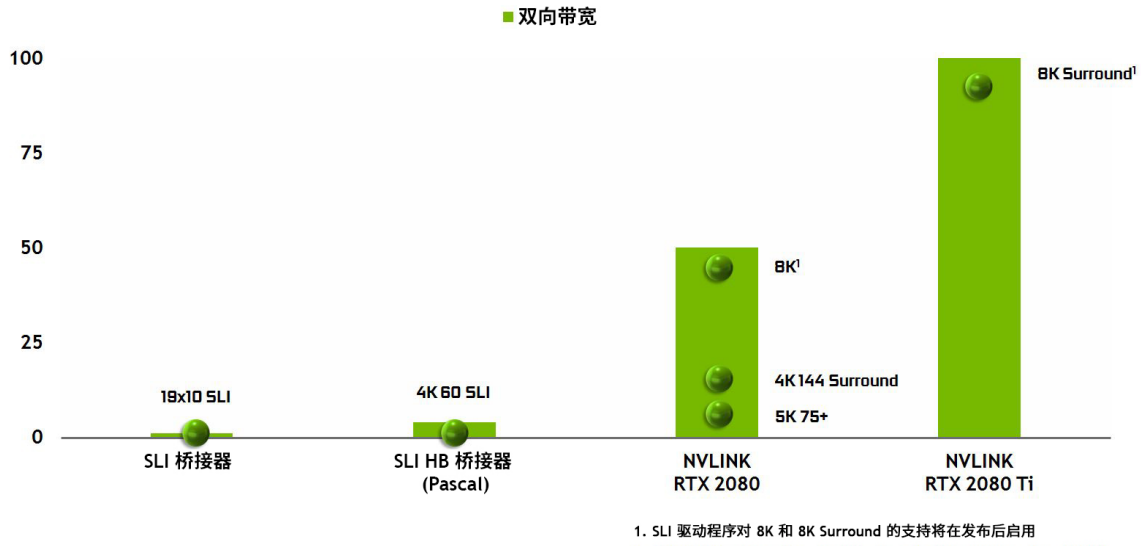
## NVLINK 改进 SLI

在 Pascal GPU 架构之前，NVIDIA GPU 将单个多输入输出 (MIO) 接口用作 SLI 桥接器，以允许第二个（第三或第四个）GPU 将其最终渲染帧输出传输到以物理方式连接至显示器的主 GPU。Pascal 通过使用更快速的双 MIO 接口增强 SLI 桥接器，进而增加 GPU 间的带宽，实现更高分辨率输出，并为 NVIDIA Surround 提供多个高分辨率显示器。

注意：在某些情况下，某些 GPU 之间的对等 SLI 数据也会在 Pascal 和前几代 GPU 中通过 PCIe 总线进行传输。

Turing TU102 和 TU104 GPU 采用 NVLink 而非 MIO 和 PCIe 接口来实现 SLI GPU 间数据传输。Turing TU102 GPU 具有两个 x8 第二代 NVLink 链路，而 Turing TU104 具有一个 x8 第二代 NVLink 链路。每个链路可在两个 GPU 之间的每个方向上提供 25 GB/秒的峰值带宽（50 GB/秒的双向带宽）。TU102 中的两个链路可在每个方向上提供 50 GB/秒的单向带宽，或 100 GB/秒的双向带宽。配备 NVLink 的 Turing GPU 可支持双路 SLI，但不支持 3 路和 4 路 SLI 配置。

相比前几代 SLI 桥接器，新型 NVLink 桥接器已增加带宽，能够将先前无法实现的高级显示器拓扑变为现实（请参见图 14）。




**注意：**SLI 驱动程序对 8K 和 8K Surround 的支持将在发布后启用。

图 14. NVLink 实现新型 SLI 显示器拓扑

# TURING 光线追踪技术

光线追踪是一种计算密集型渲染技术，能够真实模拟场景照明效果和场景物体。基于 Turing GPU 的光线追踪技术可以实时渲染具备准确物理属性的反射、折射、阴影和间接照明。请参阅附录 D：光线追踪概述（位于第 79 页），了解有关光线追踪工作原理的基本概述。

过去，GPU 架构无法使用单个 GPU 为游戏或图形应用程序执行实时光线追踪。多年来，虽然 NVIDIA 的 GPU 加速 NVIDIA Iray® 插件和 OptiX 光线追踪引擎一直在为设计师、艺术家和技术总监提供逼真的光线追踪渲染，但实时执行高质量光线追踪仍未能实现。与之类似，现有的 NVIDIA Volta GPU 能够渲染逼真的电影级光线追踪场景，但却无法在单个 GPU 上开展实时渲染。由于自带密集处理特性，游戏中尚未采用光线追踪以执行任何重大渲染任务。相反，那些每秒需要 30 到 90 帧以上动画的游戏多年来一直依赖经 GPU 加速的快速光栅化渲染技术，但却以牺牲完全逼真的场景画面为代价。

在 GPU 上实现实时光线追踪是一项巨大的技术挑战，需要 NVIDIA 的研究、GPU 硬件设计和软件工程团队携手开展近 10 年的协作。在游戏和其他应用中，我们通过在 Turing TU102、TU104 和 TU106 GPU 中集成多个基于硬件的新型光线追踪加速引擎（称为 RT 核心），并结合使用 NVIDIA RTX 软件技术来实现实时光线追踪。

图 15 显示了 SOL MAN 图像，来自于在 Turing TU102 GPU 上使用 NVIDIA RTX 技术实时运行的 NVIDIA SOL 光线追踪演示（观看演示）。

如前所述，光栅化技术多年来一直作为实时渲染标准（对于电脑游戏尤为如此）；虽然很多光栅化场景看起来十分出色，但光栅化渲染仍存在很大局限。例如，若仅使用光栅化技术渲染反射和阴影，我们需要简化可能导致多种不同伪影的设想。同样，静态光照贴图可能看起来正确无误，但物体一经移动就会失真，光栅化阴影经常会出现锯齿和漏光问题，而屏幕空间反射只能反射屏幕上显示的物体。这些伪影有损游戏体验的真实感，而开发者和艺术家若要试图通过附加效果实施修正，也要付出高昂的代价。

图 15.　NVIDIA SOL 光线追踪演示中的 SOL MAN（观看演示）

尽管光线追踪能够生成逼真度远超光栅化的图像，但它也需要大量计算。我们发现混合渲染是一种理想的解决方案，即混合使用光线追踪和光栅化。通过该方法，我们可以在光栅化最能发挥效用时使用光栅化，并在光线追踪能比光栅化提供最大视觉优势时采用光线追踪，例如渲染反射、折射和阴影。图 16 显示混合渲染流水线。

混合渲染通过在渲染流水线中实现光线追踪和光栅化技术的结合，旨在利用两者最擅长的方面渲染场景。SEED 使用混合渲染模型开展 PICA PICA 实时光线追踪实验，并在依程序组成的环境中提供自我学习代理。PICA PICA 实验基于 SEED 的研发引擎 Halcyon 构建而成，并使用 Microsoft DXR 和 NVIDIA GPU 实现实时光线追踪。


图片由 EA 的 SEED 部门提供（SEED//PICA PICA 硬件光线追踪与 Turing）

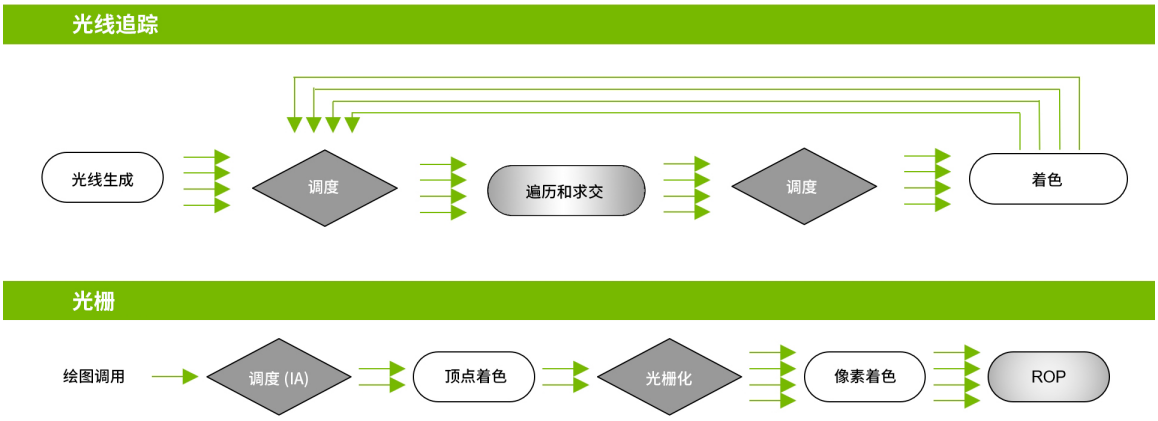
## 图 16. 混合渲染流水线

光栅化和 Z - 缓冲在确定物体可见性时速度远超光线追踪，因而可以替代光线追踪过程中的主光线投射阶段。之后利用光线追踪投射次级光线，以生成具有准确物理属性的高质量反射、折射和阴影。

开发者还可使用材质属性阈值来确定要在场景中执行光线追踪的区域。我们需要采用一种技术来指定只有当表面具有一定反射率（如 70%）时，才会触发是否应在该表面使用光线追踪来生成次级光线。

我们期望许多开发者都能使用混合光栅化或混合光线追踪技术获得高帧率以及卓越的图像质量。对于首要看重图像保真度的专业应用程序，我们也希望人们使用光线追踪来处理整个渲染工作负载、投射主光线和次级光线，从而打造令人惊叹的逼真渲染。

Turing GPU 不仅具有专用的光线追踪加速硬件，还采用高级加速结构，我们将在下一节具体描述。从本质上讲，全新的渲染流水线可使用单个 Turing GPU 在游戏和其他图形应用中进行实时光线追踪（参见图 17）。




光线追踪和光栅化在 Turing GPU 采用的混合渲染模型中同时协同运行。

## 图 17. 光线追踪和光栅化流水线各阶段详细信息

即使 Turing GPU 支持实时光线追踪，但每个像素或表面位置投射的一次或二次光线数目取决于许多因素，如场景复杂性、分辨率、场景中渲染的其他图形效果，当然还有 GPU 性能。不要期望每个像素实时投射数百条光线。事实上，当将 Turing RT 核心加速与先进的去噪滤波技术结合使用时，每个像素所需的光线数量大大降低。NVIDIA 实时光线追踪去噪模块可以显著降低每个像素所需的光线量，而且仍可产生不错的效果。

所选物体的实时光线追踪可令游戏和应用中的许多场景看起来像高端电影特效一样逼真，或者和使用基于软件的专业非实时渲染应用创建的光线追踪图像一样出色。图 18 展示 Epic Games 与 ILMxLAB 和 NVIDIA 合作创建的反射演示示例。光线追踪反射、光线追踪区域光阴影和光线追踪环境光遮蔽可在单个 Quadro RTX 6000 或 GeForce RTX 2080 Ti GPU 上运行，其渲染质量几乎与电影无异。

图 18. 反射演示示例

Turing 光线追踪硬件与 NVIDIA 的 RTX 光线追踪技术、NVIDIA 实时光线追踪库、NVIDIA OptiX、Microsoft DXR API 以及即将推出的 Vulkan 光线追踪 API 协同运行。用户将在游戏中以可播放的帧速率实时体验电影级的光线追踪物体和角色，或在专业图形应用中获得逼真的视觉效果，而凭借以前的 GPU 架构则无法在实时条件下做到这一切。

Turing GPU 可加速用于以下众多渲染和非渲染操作的光线追踪技术：

- 反射和折射
- 阴影和环境光遮蔽
- 全局照明
- 即时离线光线贴图烘培
- 特写镜头和高质量预览
- 用于注视点 VR 渲染的主光线
- 遮挡剔除
- 物理学、碰撞检测、粒子模拟
- 音频仿真（例如，基于 OptiX API 构建的 NVIDIA VRWorks 音频）
- AI 可见性查询

- 引擎内路径追踪（非实时），以为调整实时渲染技术和降噪器、材质组合和场景照明生成参考截图。

以下章节更将更详细地介绍渲染光线追踪阴影、环境光遮蔽以及使用 Turing 光线追踪加速的反射。NVIDIA 开发者网站提供有关可通过 Turing 光线追踪加速的渲染操作的更多信息。

# TURING RT 核心

Turing 架构基于硬件的光线追踪加速的核心是包含在每个 SM 中的新 RT 核心。RT 核心可加速包围盒层次结构 (BVH) 遍历、光线和三角形交集测试（光线投射）功能。（请参见附录 D：光线追踪概述（位于第 79 页），详细了解 BVH 加速结构如何运行）。RT 核心代表在 SM 中运行的线程执行可见性测试。

RT 核心与先进的去噪滤波（NVIDIA Research 开发的高效 BVH 加速结构）和 RTX 兼容的 API 协同运行，以在单个 Turing GPU 上提供实时光线追踪。RT 核心自动遍历 BVH，并通过加速遍历、光线和三角形交集测试分担 SM 的工作负荷，从可让 SM 处理其他顶点及像素，并执行着色运算工作。BVH 构建和重构等功能由驱动程序处理，而光线生成和着色则由应用程序通过新型着色器进行管理。

为了更好地理解 RT 核心的功能，及其加速的具体内容，我们首先应该说明如何在没有专用硬件光线追踪引擎的 GPU 或 CPU 上执行光线追踪。从本质上讲，BVH 遍历的过程需要通过着色器操作来执行，并通过每次光线投射的成千上万个指令槽来测试 BVH 中的包围盒交集，直至最后命中一个三角形，并且交集点的色彩会形成最终的像素颜色（如果没有命中三角形，背景颜色可作为像素颜色）。

在没有硬件加速的情况下，光线追踪需要每条光线有成千上万个软件指令槽来测试 BVH 结构中不断缩小的包围盒，直至命中一个三角形。这是个计算密集型过程，如果没有基于硬件的光线追踪加速，则不可能在 GPU 上实时执行（参见图 19）。

Turing 中的 RT 核心可以处理所有 BVH 遍历和光线-三角形交集测试，使 SM 无需对每条光线花费数千个指令槽，否则这对于整个场景来说可能是非常大的指令量。RT 核心包括两个专门单元。第一个单元执行包围盒测试，第二个单元执行光线-三角形交集测试。SM 只需启动光线探测器，RT 核心便执行 BVH 遍历和光线-三角形测试，并向 SM 返回一个命中或无命中。SM 得到很大程度的解放，从而可以执行其他图形或计算工作。参见图 20 或采用 RT 核心的 Turing 光线追踪说明。

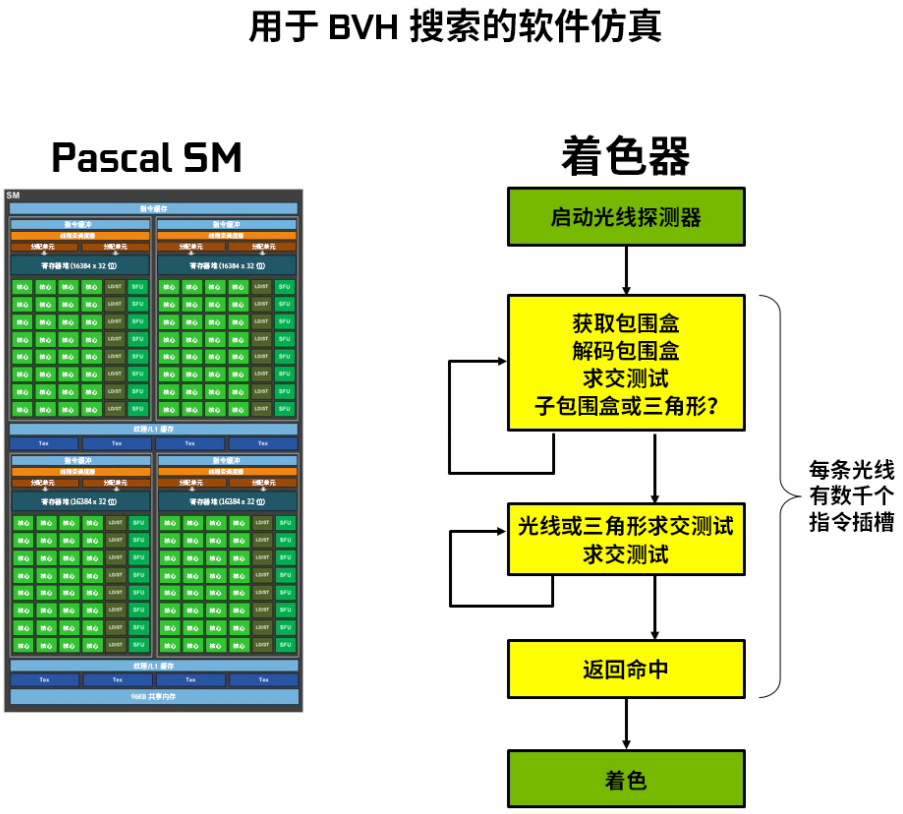



图 19. 使用 Turing 前的光线追踪

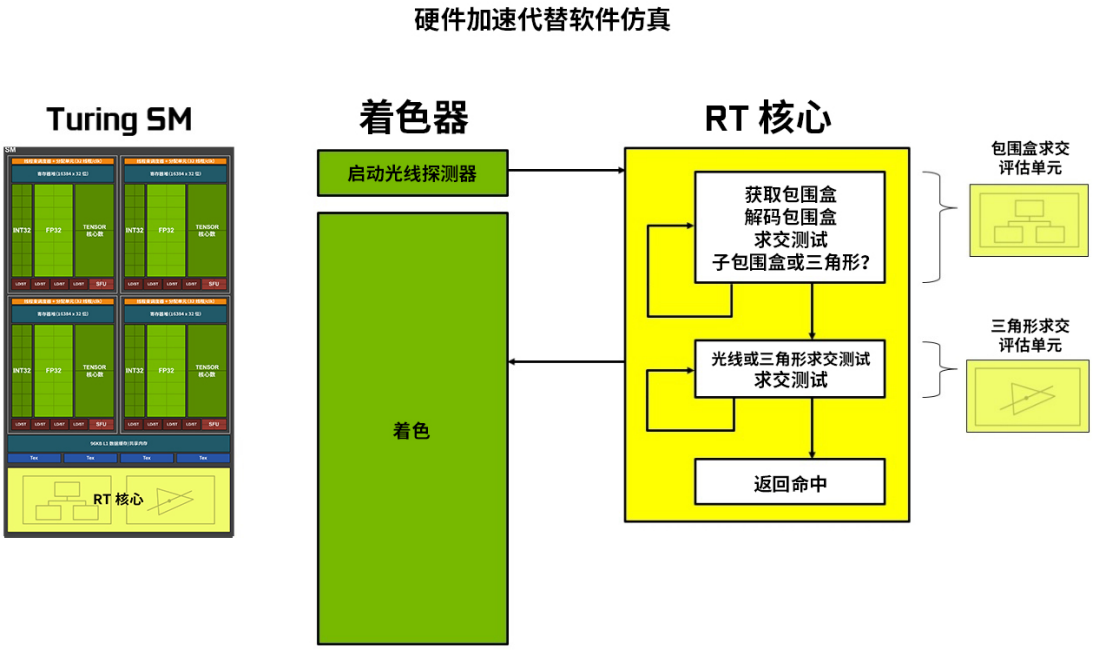



图 20. 采用 RT 核心的 Turing 光线追踪

采用 RT 核心的 Turing 光线追踪性能明显高于 Pascal GPU 的光线追踪性能。在各种工作负载下，Turing 的千兆光线/秒数值远远高于 Pascal，如 图 21 所示。Pascal 在软件中大约用 1.1 千兆光线/秒或 10 TFLOPS / 千兆光线来执行光线追踪，而采用 RT 核心的 Turing 用 10+ 千兆光线/秒来执行，可以 10 倍的速度执行光线追踪。

注意：本文不提供在游戏或应用程序中使用 RTX、DXR 或其他 API 执行光线追踪的开发者详细信息，但许多其他资源提供此信息。良好的初始信息源包括 NVIDIA RTX 和 DirectX 光线追踪简介博文、NVIDIA RTX 技术开发者网站，以及 NVIDIA 在 RTX 上提供的可公开访问的 GDC 2018 课程（名为在游戏中使用 NVIDIA RTX 执行光线追踪）。也可以参考 Microsoft 在 DXR 上的博客。

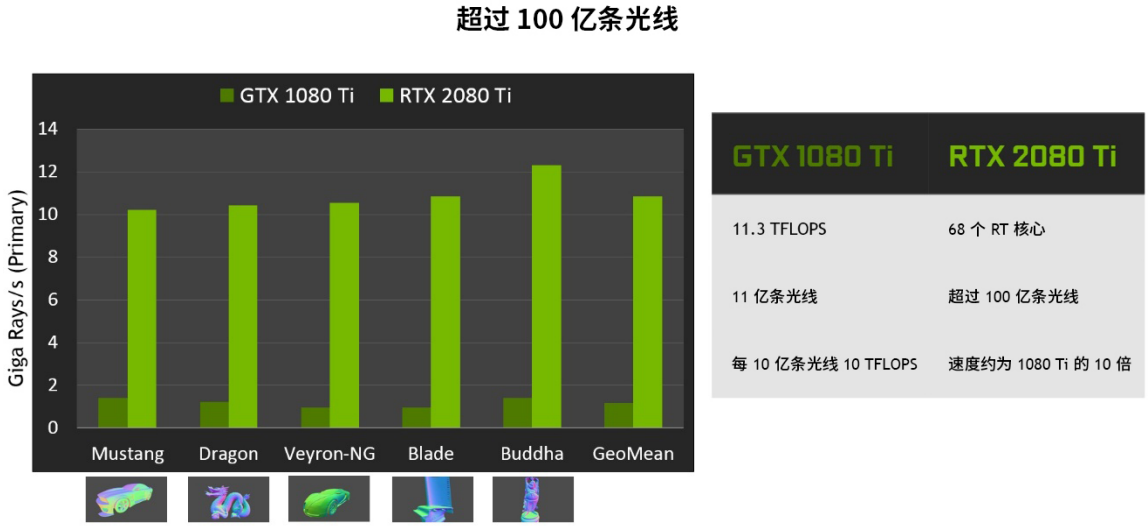



图 21. Turing 光线追踪性能

# NVIDIA NGX 技术

NVIDIA NGX™（神经图形加速）是基于深度学习的新技术堆栈，属于 NVIDIA RTX 技术的一部分。NVIDIA NGX 利用深度神经网络 (DNN) 和一套神经服务执行基于 AI 的功能，以此加速并增强图形、渲染和其他客户端应用。NGX 采用 Turing Tensor 核心执行基于深度学习的运算，并能加速向最终用户直接交付 NVIDIA 深度学习研究。请注意，NGX 不在 Turing 之前的 GPU 架构上运行。

注意：NGX 不在 Turing 之前的 GPU 架构上运行。

## NGX 软件架构

NGX 的特性与 NVIDIA 驱动程序和硬件紧密结合。NGX API 为游戏和应用程序提供几个 AI 功能的访问权限。这些功能已经过 NVIDIA 预先训练，并随时可供集成。API 采用瘦式设计，便于应用程序集成多个 AI 功能。NGX 服务在 GPU 上运行，使其能够支持多个功能和应用程序。

NVIDIA NGX 特性由 NVIDIA GeForce Experience™ (GFE) 应用程序或技术预览版 NVIDIA Quadro Experience™ (QXP) 应用程序管理。GFE 或 QXP 在安装或更新后，将查找 Turing GPU 是否存在。一旦检测到 Turing GPU，系统便会下载并安装 NGX 核心包。GFE/QXP 与

NGX 核心进行通信，以确定游戏和应用程序 ID 是否存在及其与 NGX 的关联性。然后下载可用于各种已安装游戏和应用程序的不同 DNN 模型，以供后续使用。

NGX DNN 模型可与 CUDA 10、DirectX 和 Vulkan 驱动程序交互，也可以利用 NVIDIA TensorRT™ 这个高性能深度学习推理优化器，为深度学习推理应用提供低延迟和高吞吐量。Turing 的增强型 Tensor 核心已为 NGX 模型和服务加速。

# 深度学习超级采样 (DLSS)

在现代游戏中，渲染的帧并非直接显示，而是先对其执行后期处理图像增强步骤。在此步骤中，将来自多个渲染帧的输入组合在一起，以在保留细节的同时，消除诸如锯齿等视觉失真现象。例如，随机采样抗锯齿 (TAA) 是目前最常用的图像增强算法之一，这是一种基于着色器的算法，使用运动矢量组合两帧，以确定在何处对前一帧进行采样。然而，这种图像增强过程从根本上来说很难正确实行。

NVIDIA 的研究人员认识到，这类没有清晰算法解决方案的图像分析和优化问题可通过应用 AI 来完美解决。正如前文所述，图像处理示例（例如 ImageNet）是深度学习的最成功应用之一。深度学习现在已获得超人类的能力，可通过观察图像中的原始像素识别狗、猫、鸟等动物。在这种情况下，目标将是通过查看原始像素将渲染的图像组合在一起，以产生高质量的结果 - 不同的目标，但使用相似的功能。

为解决这一难题而开发的深度神经网络 (DNN) 称为深度学习超级采样 (DLSS)。DLSS 针对一组给定输入样本所产生的输出质量要比 TAA 高得多，我们利用这种能力来改进总体性能。虽然 TAA 以最终目标分辨率渲染，然后组合帧，而 DLSS 通过消除细节，能够以更低的输入样本数更快地渲染，这意味着以目标分辨率得到的结果与 TAA 结果的质量不相上下，但是只需执行大约一半的着色工作。

图 22 所示为 UE4 渗透器上的结果采样演示。DLSS 提供与 TAA 类似的图像质量，并且性能得到很大提升。RTX 2080 Ti 具有更快的原始渲染性能，再搭载性能更高的 DLSS 和 Tensor 核心，致使 RTX 2080 Ti 的性能可达到 GTX 1080 Ti 的两倍。


图 22. 搭载 4K DLSS 的 Turing 性能是搭载 4K TAA 的 Pascal 性能的两倍

产生这一结果的关键是 DLSS 的训练过程，在此过程中，DLSS 将有机会学习如何根据大量超高质量的示例生成所需的输出。为了训练网络，我们收集成千上万的“真值”参考图像，这些图像均采用黄金标准方法渲染，具有完美的图像质量，即 64 倍超级采样 (64xSS)。64 倍超级采样是指我们在像素内以 64 个不同的偏移量进行着色，然后将输出组合在一起，生成具有理想细节并抗锯齿的优质图像，而不是对每个像素进行一次着色处理。我们还会捕捉与之相匹配的正常渲染的原始输入图像。接下来，我们开始训练 DLSS 网络来匹配 64xSS 输出帧，通过遍历每个输入，要求 DLSS 产生一个输出，测量其输出与 64xSS 目标之间的差值，并根据差值调节网络中的权重，这个过程称为反向传播。经过多次迭代后，DLSS 可以自行学习生成接近 64xSS 质量的图像，同时还避免出现影响 TAA 等传统方法的模糊、不清晰和透明问题。

除了上面描述的 DLSS 功能（这是标准的 DLSS 模式）之外，我们还提供第二种模式，称为 DLSS 2X。在这种情况下，以最终目标分辨率渲染 DLSS 输入，然后将其与更大的 DLSS 网络结合，以产生接近 64 倍超级样本渲染水平的输出图像，而这种结果不可能通过任何传统方式实时实现。图 23 显示正在运行的 DLSS 2X 模式，该模式产生的图像质量非常接近 64 倍超级采样参照图像。


图 23. 几乎无法区分 DLSS 2X 和 64xSS 图像

最后，图 24 显示多帧图像增强面临的一种挑战。在这种情况下，半透明屏幕浮动于以不同方式移动的背景前面。TAA 往往会盲目地跟随移动物体的运动矢量，从而造成屏幕上的细节模糊不清。而 DLSS 却能识别出场景中更为复杂的变化，并以一种更智能的方式整合输入，从而避免模糊问题。

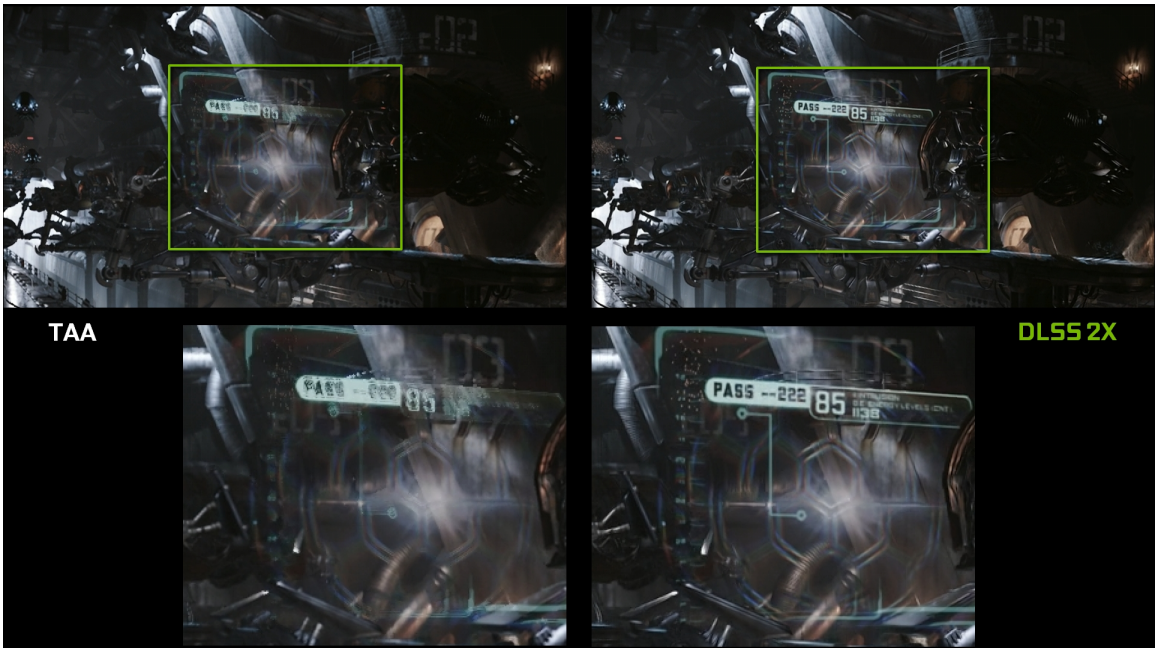



图 24. 与 TAA 相比，DLSS 2X 可提供大幅改善的时间稳定性和图像清晰度

# 图像修复 (INPAINTING)

图像修复 允许应用程序提供从图像中删除现有内容的功能，并使用 NGX AI 算法将删除的内容替换为计算机生成的逼真的替代内容。例如，可使用 图像修复功能 自动从景观图像中删除电源线，并无缝替换为现有的天空背景。图像修复的理念并不新鲜，但是现有的解决方案依靠从图像中的某个地方复制数据来填充空洞。如果算法没有得到很好的优化，这可能导致视觉上出现明显的平铺图案。相反，NGX 的图像修复算法依靠用大量真实图像进行的训练来合成新的内容，以填补空白。因此，可产生更有视觉意义的图片（请参见图 25）。

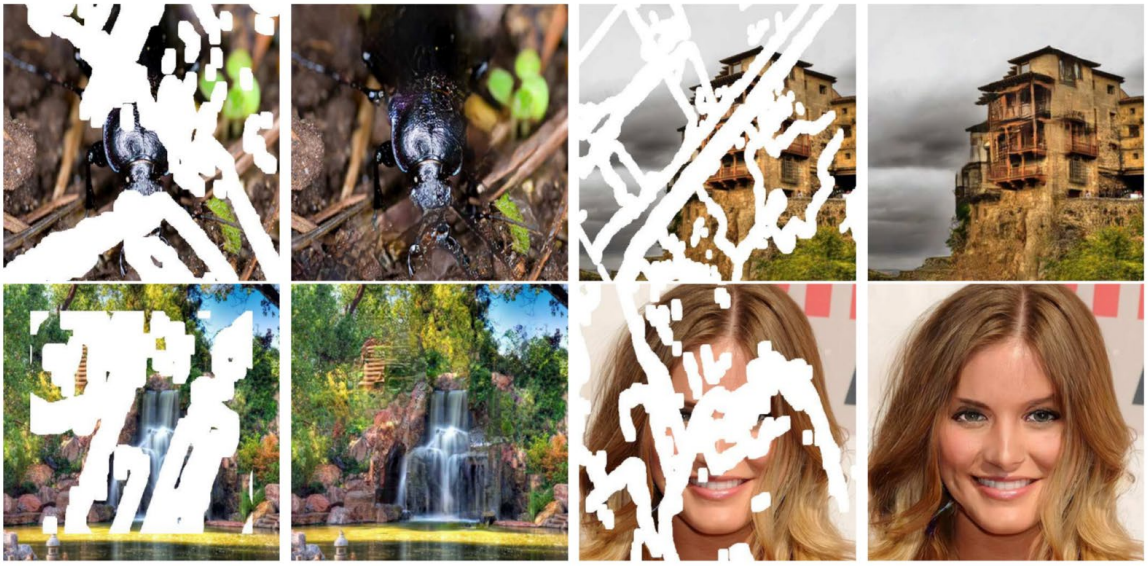


图 25. NGX 图像修复功能示例，缺失的图像数据被智能地替换为有意义的图像信息

## AI SLOW-MO

AI Slow-Mo 将插补帧插入视频流，以提供顺畅的慢动作视频。NGX 分析帧的特性和物体，识别物体和镜头移动，并在现有的视频帧之间创建新的视频帧。如此便可产生插补伪影较少且顺畅的慢动作视频。观看此 NVIDIA Research 视频，了解正在运行的 AI Slow-Mo。

## AI SUPER REZ

AI Super Rez 可将图像或视频的分辨率增加 2 倍、4 倍或 8 倍。与将现有像素拉伸并在像素之间进行过滤的传统方法不同，AI Super Rez 通过解释图像并以智能的方式放置数据来创建新的像素（参见图 26）。此举能够进一步强化放大效果，合理保留景深和其他艺术内容。视频超高分辨率网络经过高度优化，能够以 1080p 到 4K（升采样）实时运行 (~30 fps)，且 PSNR 比双三次插值高 1-2 dB。


图 26. AI Super Rez 相比其他过滤方法提供更高的图像清晰度

# TURING 先进的着色技术

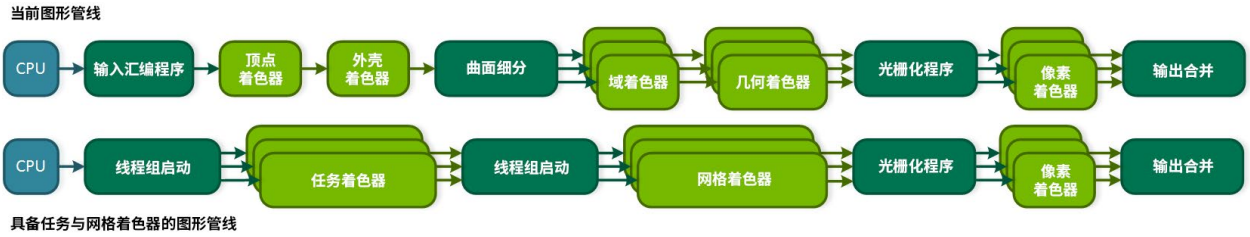
## 网格着色

现实世界视觉元素丰富、几何形状复杂。特别是户外场景可由几十万个元素（岩石、树木等）组成。CAD 模型也面临同样的挑战。如今的图形流水线，包括顶点、曲面细分和几何着色器，在渲染单个物体的各个细节方面做的非常好，但仍然具有局限性。每个物体都需要从 CPU 中调用独特绘图，着色器模型是一个逐线程模型，限制了可使用的算法类型。网格着色采用一种更灵活的新模型，使开发者能够消除 CPU 的绘图调用瓶颈，从而使用更有效的算法来生成三角形。

视觉元素丰富的图像（如图 27 中显示的图像）有太多独特的复杂物体，用现今的图形流水线无法实时渲染。

## 图 27. 网格着色，视觉元素丰富的图像

图 28 显示基于网格着色与现今全几何处理流水线的比较。现在，开发者可以使用顶点着色器直接为光栅化程序生成三角形，也可以使用曲面细分着色器处理经过曲面细分的分块，以为光栅化生成最终三角形。




## 图 28. 当前的图形流水线对比具有任务和网格着色器的图形流水线

网格着色采用两个新的着色器阶段，即任务着色器和网格着色器阶段，这两者均支持相同的功能，但具有更高的灵活性。网格着色器阶段可为光栅化程序生成三角形，但在内部，则使用与计算着色器类似的协作线程模型，而不使用单线程程序模型。在流水线中网格着色器前面的是任务着色器。任务着色器的操作类似于曲面细分的外壳着色器阶段，因其能够动态生成结果。然而，像网格着色器一样，任务着色器使用协作线程模型，其输入和输出均为用户定义，而不是将分块作为输入，将曲面细分决定作为输出。

图 29 显示网格着色的功能示例。这个场景呈现的是一个颇具挑战性的环境，在这个环境中，观察者的视角处于一个有着成千上万个独立物体的广阔视野中。开发者现在可向 GPU 发送包含许多物体的列表，而不是通过 CPU 的唯一绘图调用将每个物体发送至 GPU。然后任务着色器并行处理此物体列表，并启动网格着色器来为对应的三角形着色，然后将其提交至光栅化程序。这种方法不仅可以消除 CPU 处理物体的瓶颈，而且能够将以实时帧速率显示的物体数量增加一个数量级以上。

图 30 显示网格着色支持的另一个优化。在图 29 中，我们希望在近距离观察每颗小行星时都获得真实的细节，但许多小行星离观测者太远，看不见任何细节。优化这种情况的一种方法是获得每个物体的多个版本（在不同的细节级别），并根据当前帧中屏幕空间中物体的大小动态选择适当的版本。任务着色器支持这种优化。当其扫描物体列表时，也可以查看每个物体的大小，并选择适当的 LOD 版本，然后将其发送至网格着色器进行处理，或者在曲面细分的情况下，可以让网格着色器进一步曲面细分来自 LOD 版本的三角形。

展示如何使用网格着色来实时渲染成千上万个物体

图 29. 小行星场景演示的屏幕截图

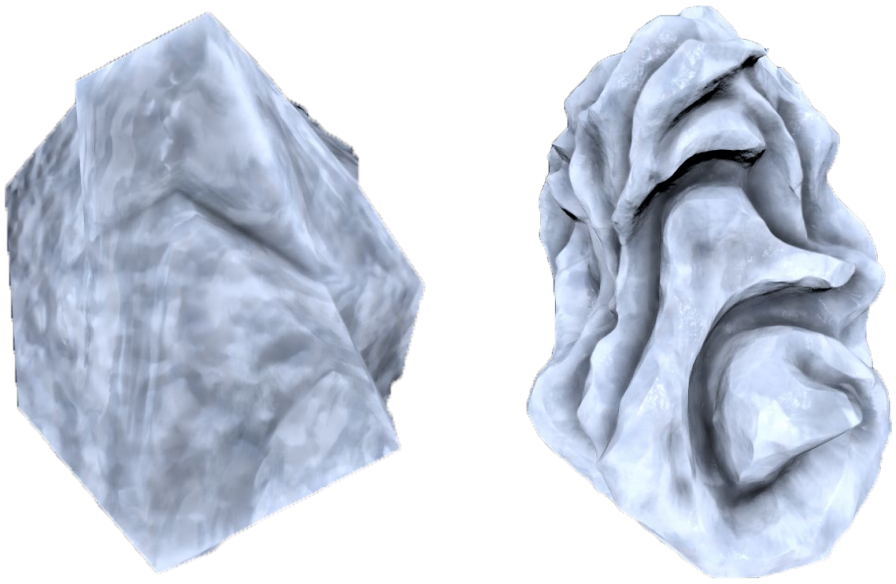


图 30. 低细节级别和高细节级别 (LOD) 的小行星

网格着色也使开发者更容易以不同的方式操作几何图形。图 31 显示网格着色用于为 CAD 应用实行动态剖面功能。已消除由球面边界定义的球形剖面图区域内的任何几何形状，从而揭示出在该区域汽车元件的详细结构。网格着色器和任务着色器可基于几何图形相对于球体的位置裁剪和修改此几何图形，进而执行此操作。

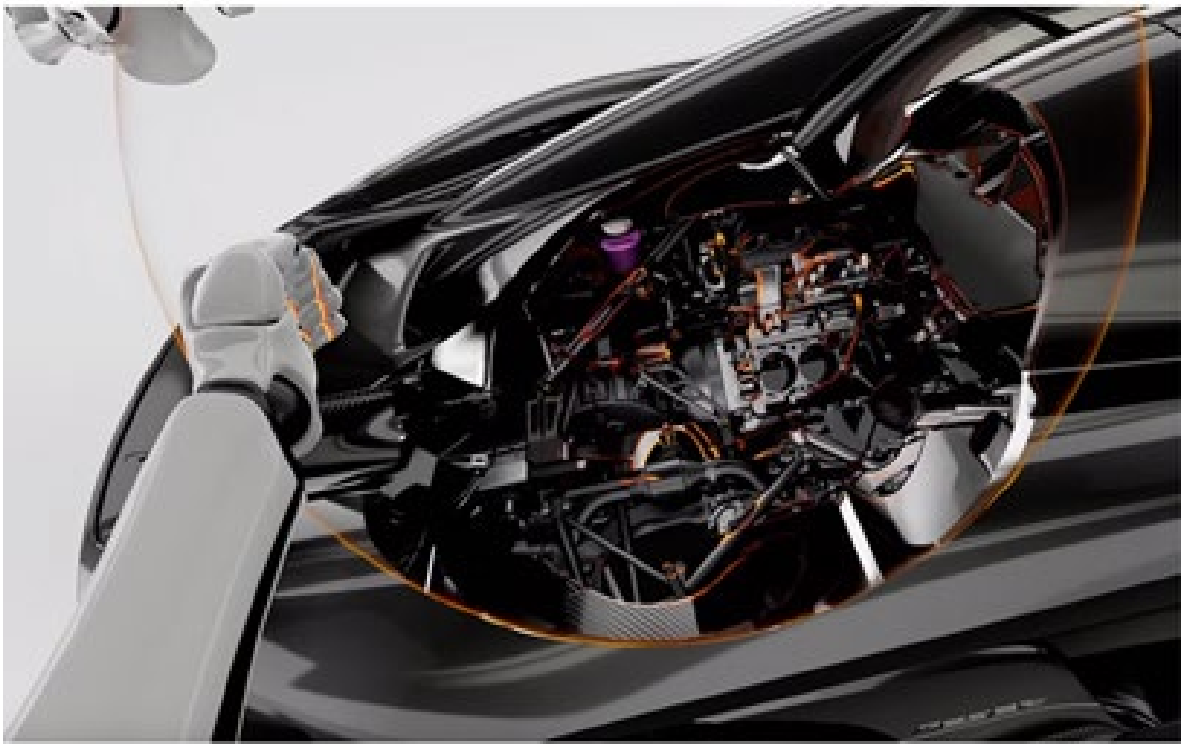


图 31. 在 NVIDIA Holodeck™ 中观看的经动态计算的 Koenigsegg 模型球形剖面图

# 可变速率着色

随着新一代游戏对计算能力的整体需求不断增加，我们一直在寻找能够让开发者省去着色工作的方法，因为这些着色工作并不能提高最终渲染帧的图像质量。整个 GPU 能力应只用于改善用户体验，要么是获得更丰富的图形，要么是更高的帧率。

在前几代产品中，我们通过采用多分辨率着色 (MRS) 和 透镜匹配着色 (LMS) 技术优化与 VR 相关的着色工作负载。VR 系统的一个重要特性是透镜系统中的光学器件对观察表面采用不同的分辨率和采样率。MRS 和 LMS 使开发者能够将渲染表面分割成 16 个子区域，并将每个区域的采样率与镜头匹配，而不是过渡渲染所有区域以满足最大的局部采样需求。

然而，这只是常见问题的一个例子。Turing 采用一种称为可变速率着色 (VRS) 的新功能控制着色速率，此功能的灵活度大幅提升。通过 VRS，我们现在可以非常精细的级别动态调节着色速率 - 屏幕的每个 16 像素 x 16 像素区域现在可有不同的着色速率（参见图 32）。

这种精细级别的控制使开发者能够部署新的算法，而在以前，则无法使用这些算法来优化着色速率及提高性能。本节讨论 VRS 的底层硬件机制，及其支持的一些强大的新算法。

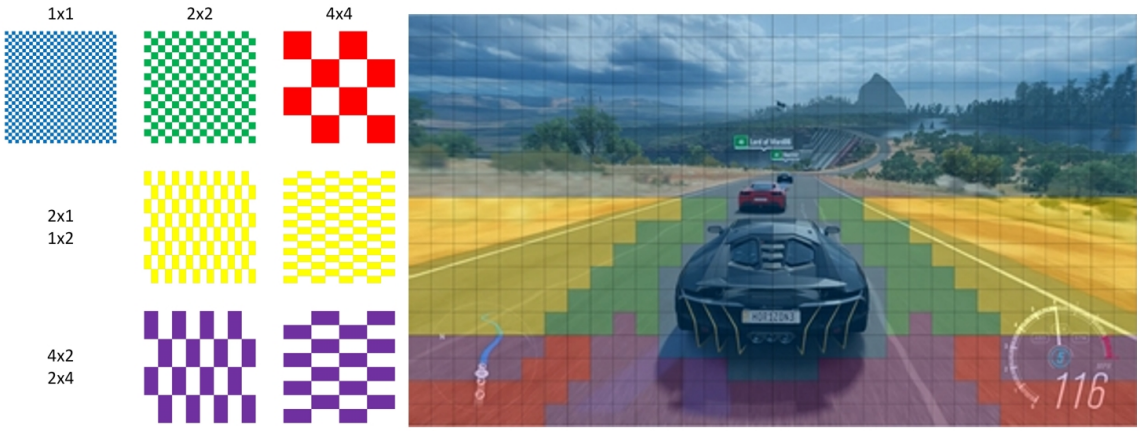



图 32. Turing VRS 支持的着色速率和对游戏帧的应用示例

如果没有 VRS，图 32 场景中的每个像素将被单独着色（1×1 蓝色网格）。通过使用 VRS，三角形的像素着色率可以不同。开发者在每个 16x16 的像素区域最多可拥有七个着色选择，包括让 4 个像素 (2 x 2)，或 16 个像素 (4 x 4) 或非方形像素（1 x 2 或 2 x 4）呈现同一着色效果。图 32 右侧的色彩覆盖展现了帧的可能应用，也许可对赛车进行全速率着色（蓝色区域），而对赛车周围的区域，按照每 4 个像素着色一次（绿色），对左右两侧的道路可按照每 8 个像素着色一次（黄色）的频率来着色。

总体而言，借助 Turing 的 VRS 技术，我们可以混合利用每个可见性样本着色一次（超级采样）和每 16 个可见性样本着色一次的不同速率对场景着色。开发者可在空间中指定着色率（使用纹理），并能指定使用每个基元的着色率属性。由此便可使用多种速率对单个三角形着色，从而为开发者提供精细控制。

注意：VRS 让开发者无需更改可见率便能控制着色率。相较于 MRS 和 LMS 等降低指定区域内总体渲染分辨率的技术，VRS 能够解耦着色率和可见率，适用范围更广。同时，由于 VRS 与 MRS 及 LMS 是由不同硬件路径支持的独立技术，因此可组合使用。

以下三种强大算法均采用 VRS 技术：

- 内容自适应着色：

  对渐变色区域降低着色率

- 移动自适应着色：

  动态降低移动物体的着色率

- 注视点渲染：

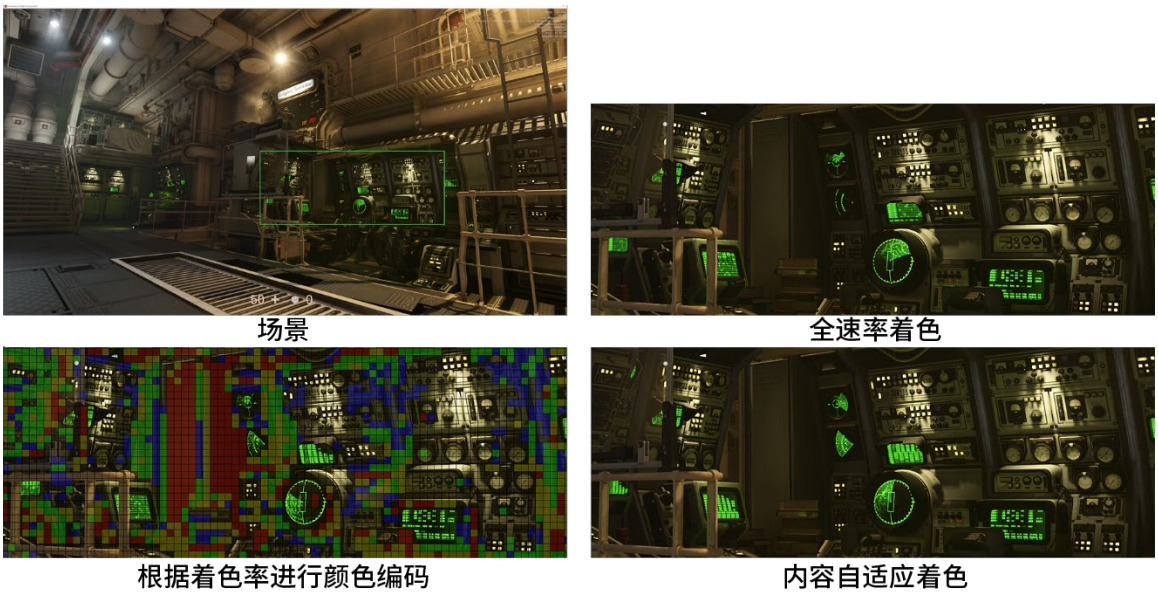
  对远离视野中心的区域降低着色率

## 内容自适应着色

内容自适应着色算法是通过（跨帧）空间和时间颜色连贯性等因素降低着色率。对于下一帧渲染画面的不同部分，其所需着色率是由当前帧结束时的后处理步骤计算得出。如果某特定区域的细节量相对较低（天空或平面墙等），我们便可在下一帧中局部调低该区域的着色率。后处理分析会输出一种纹理，该纹理既能指定每个 16 x 16 图块的着色率，也可用于确定下一帧的着色率。开发者仅需对着色器稍作调整便可基于内容降低着色率，而无需修改现有流水线。

图 33 是内容自适应着色应用示例。实时观看内容自适应着色操作自然是欣赏其实效的最佳途径，但为方便说明此操作过程，我们使用了一些屏幕截图。左上角全屏图像中的绿色方框表示需放大的裁剪区域。左下角表示放大后的区域，着色率覆盖层如图 32 所示。请注意，垂直的平面墙以最低速率（红色 = 4×4）进行着色，仪表和刻度表以全速率（无颜色覆盖 = 1×1）进行着色，而场景其他区域则按不同的中间速率进行着色。

右侧上下两张图像分别表示对此裁剪区域禁用（上）和启用（下）内容自适应着色后的截图，两张图在画质方面无视觉差异（由于采样时间不同，图像的仪表盘动画略有不同）。




通过对原始场景数据应用不同着色率（左下角图像），我们可以创建右下角图像中的内容自适应着色效果。请注意全着色率场景与内容自适应着色场景之间的相似之处

图 33.　内容自适应着色示例

## 移动自适应着色

可变速率着色的第二种应用是利用物体移动进行着色。人眼向来是对移动的物体进行线性追踪，因此即使物体处于运动状态，我们也能看到其细节。但在 LCD 屏幕上，物体不会平滑或连续移动，而是会以每次 60 Hz 的刷新率从一处跳至下一处。人眼试图以平滑方式追踪物体，由于物体位置会在人眼追踪轨迹的前后方移动，所以看起来就像是在视网膜上来回摆动。这样产生的最终结果是，我们无法看清物体的全部细节，而只能看到分辨率较低或模糊的画面。图 34 便可阐明此场景。

图 34. 物体移动、视网膜以及显示器持续作用下产生的感知模糊

在 图 34 中，左图为屏幕显示的图像。如果此图像从左向右移动，人眼观看时会感觉其在来回跳动。结合这种运动后，人眼最终会看到右下角所示图像，也即分辨率较低的原图画面。

此现象的主要寓意是，当物体在场景内快速移动时，对其进行全分辨率着色会造成浪费。在降低采样率后实施着色将能提高效率，同时还能保证足够高的速率，以实现视觉等效。而优化着色后所节省的资源可用于提供更高的帧率，以便更轻松地渲染场景。

VRS 能为我们提供实施该优化的工具。最简单的方法就是通过时间性抗锯齿 (AA) 技术中的运动向量来理解运动。运动的方向和量级可用于为每张图块直接选择合适的着色率。

一种相关方法是在应用程序中通过 VRS 以充分利用模糊效果，有时还会对动态模糊和场景深度 (DOF) 进行显式渲染。应用程序可直接计算出单个物体的模糊度和方向，并使用模糊度来设置每个三角形的着色率。

请注意，这两种示例（内容自适应着色与移动自适应着色）方法也可组合使用，每个区域或三角形的最终着色率可通过应用程序指定的双速率函数计算得出。

## 注视点渲染技术

第三个示例应用是注视点渲染技术。人眼感知的分辨率取决于视角，而注视点渲染技术正是以此类观察为基础。我们对视野中心物体的视觉分辨率最高，而对外围物体的视觉分辨率很低。因此，若已知观察者眼睛的所在位置（通过 VR 或非 VR 系统中的人眼追踪

确定），我们便可使用此原理相应调整着色率。我们可以在视野外围实施较低速率着色，而在视野中心实施较高速率着色。

这三种应用都是不错的示例，但我们也期待开发者去探索除本文所述以外的其他创新应用。VRS 可为开发者提供对着色率的高级控制，而现在他们已能使用任何可利用此功能的算法。

最后，VRS 也允许开发者提升着色率。当使用多重采样抗锯齿 (MSAA) 时，开发者可通过 VRS 将着色率从每像素 1 次（基线）提升至每像素 2 次、4 次或 8 次。提升的着色率不能超过 MSAA 采样数。

# 纹理空间着色

Turing GPU 引入一种名为纹理空间着色 (TSS) 的全新着色功能，其概念是动态计算着色值，并将该值作为纹理空间的纹素存储至纹理内。随后，系统将对像素进行纹理映射，此过程会将屏幕空间中的像素映射至纹理空间，并使用标准纹理查找操作对相应纹素进行采样和筛选。借助此项技术，我们能够以完全独立的速率和单独的（解耦）坐标系统对可见性和外观进行采样。通过 TSS，开发者可（重复）使用解耦着色空间中完成的着色计算，从而同时提高质量和性能。

开发者可借助 TSS 来利用空间和时间渲染冗余。通过将着色与屏幕空间像素网格进行解耦，TSS 可以实现高级别帧到帧稳定性，因为着色位置不会在相邻两帧间移动。对于 VR 等要求大幅提升图像质量且不能出现锯齿伪影和时间域闪烁的应用而言，这种时间稳定性十分重要。

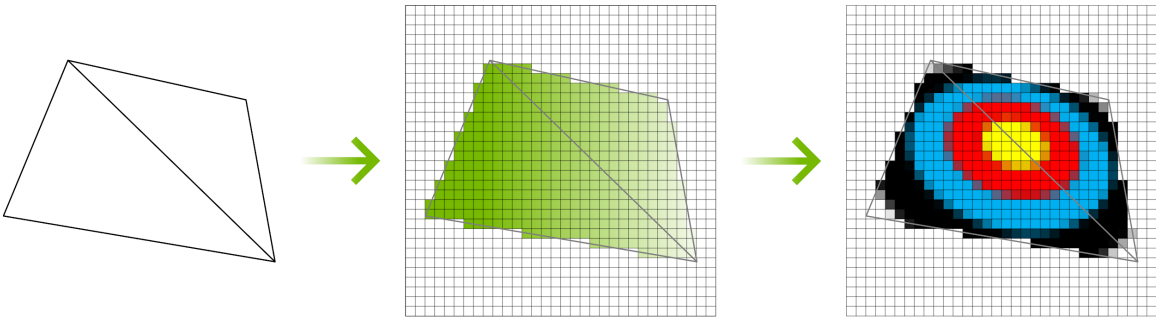
TSS 具有内在的多分辨率灵活性，此特性继承自纹理贴图的 MIP-map 层次结构或金字塔式图像结构。对像素着色时，开发者可调整贴图至纹理空间，选择 MIP 级别（细节级别），从而对着色率实施精细控制。由于低细节级别的纹素较大，因此会覆盖物体的较大部分，而且可能覆盖多个像素。

TSS 会记住已着色的纹素，并只对新请求的纹素实施着色。着色和记录的纹素可重复用于同一帧、相邻场景或后续帧中的其他着色请求。通过控制着色率并重复使用先前着色的

纹素，开发者可管理帧渲染时间，并将其维持在 VR 和 AR 等应用的固定时间预算以内。开发者也可使用相同机制对已知的低频现象（如雾）降低着色率。记忆着色结果的益处还可惠及顶点着色器、计算着色器以及通用计算。TSS 基础架构可用于记忆和重复使用任何复杂计算的结果。

## TSS 的运作机制

图 35 展示了传统的光栅化和着色过程。首先对 3D 场景实施光栅化处理，并将其转换为屏幕空间中的像素。然后，对像素进行可见性测试、外观着色以及深度测试。同一像素的相同屏幕空间像素网格上均会执行这些操作。




对 3D 场景实施光栅化处理，将其转换为屏幕空间中的像素，同时确定可见像素并进行着色。

### 图 35. 传统的光栅化和着色过程

借助 TSS，系统能对可见性采样（光栅化和 Z 测试）和外观采样（着色）这两种主要操作进行解耦，并在不同的采样网格甚至在不同时间线上以不同速率执行这些操作。着色过程已不再与屏幕空间像素直接关联，而是发生在纹理空间内。在图 36 中，系统依旧会对几何体实施光栅化以产生屏幕空间像素，而可见性测试则仍在屏幕空间中进行。但我们能发现，纹素需要覆盖输出像素，而非在屏幕空间中进行着色。换言之，屏幕空间像素的覆盖区域将映射至单独的纹理空间，并对纹理空间中的相关纹素进行着色。映射至纹理空间是一种标准的纹理映射操作，能够对 LOD 和各向异性过滤等技术实施相同控制。为生成最终的屏幕空间像素，我们将从着色纹理中进行采样。我们根据样本请求按需创建纹理，同时仅为引用的纹素生成值。

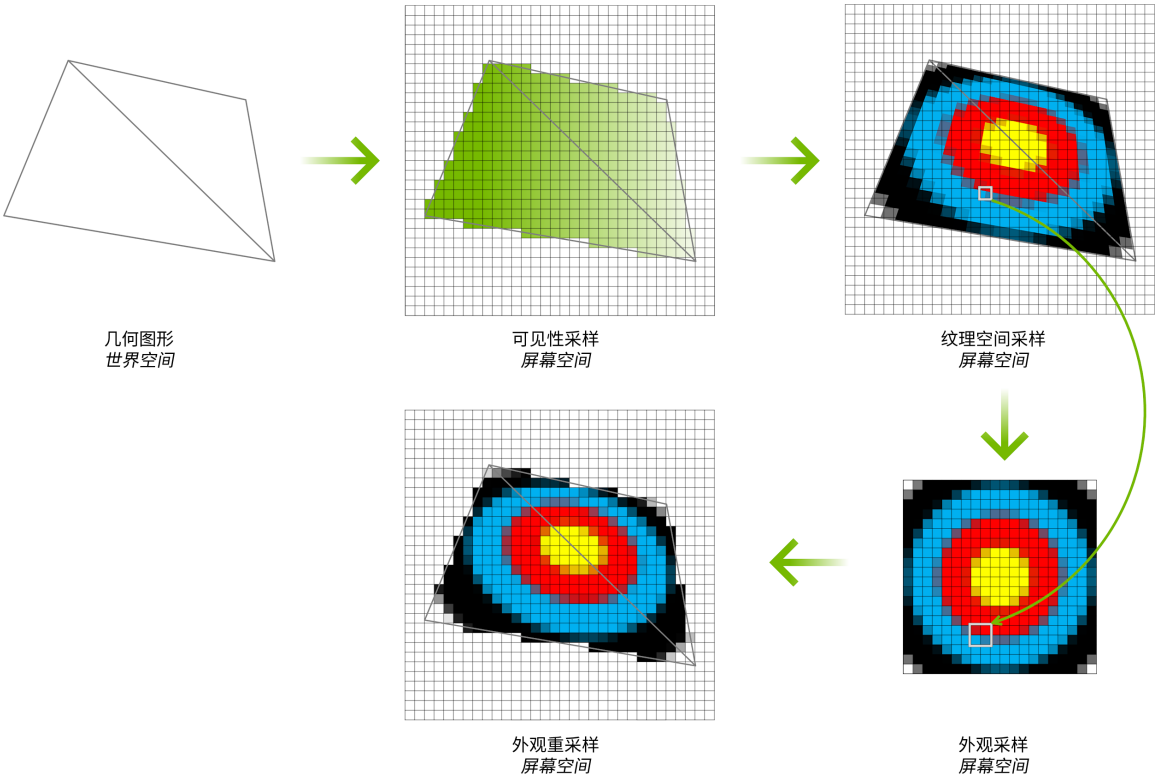



图 36. 纹理空间着色过程

TSS 的一个示例是提高 VR 渲染的效率。图 37 展示了 TSS 在 VR 渲染中的用例。在 VR 中，我们渲染了一对立体图像，且左眼可见的所有元素几乎都将显示在右眼视图中。借助 TSS，我们可对整个左眼视图着色，然后再对着色后的左眼视图进行采样，继而渲染右眼视图。我们仅需在未找到有效样本的情况下（例如，左眼不可见但右眼可见的背景物体），对右眼视图中的新纹素进行着色。

如前文所述，借助 TSS，您可以通过调整纹理 LOD 对每像素着色率实施连续动态控制。通过改变 LOD，我们可以按需选择不同的纹理 MIP 级别，从而减少要着色的纹素数量。请注意，这意味着 TSS 采样法也可用于实现众多相同的着色率降低技术，而这些技术均由 VRS 特性（请参见可变速率着色部分，50 页）提供支持。开发者在挑选最适合的方法时，应以自身目标为准。VRS 对渲染流水线所作的权重更改较少，而 TSS 则具有更高的灵活性，并能支持其他用例。

图 37. 对立体图像进行的纹理空间着色

# 多视图渲染

凭借多视图渲染 (MVR)，开发者只需一次渲染过程，即可从多个视点高效绘制场景，甚至能够绘制同一角色不同姿势的多个实例。Turing 硬件在每次渲染过程中最多可支持四个视图，API 层最多可支持 32 个视图。在渲染多版本视图时，只需执行一次几何体提取和着色操作，Turing 便能对三角形及与之关联的顶点属性作出卓越处理。在通过 D3D12 视图实例化 API 访问时，开发者只需使用变量 SV_ViewID 来索引不同的转换矩阵、参考不同的混合权重或控制任何期望的着色器行为，具体视其正在处理的视图而定。

借助多个活跃视图，每个三角形都能混合包含与视图有关及与之无关的属性（所有视图均会共享的值）。在与视图有关的属性中，反射方向是一个简单示例，因为它取决于人眼的位置、顶点位置和法向量。为提高效率，NVIDIA 编译器会分析输入着色器，并生成执行一次视图不相依代码的编译输出，其执行结果会在所有输入视图间共享，同时还需对每个输出视图的视图相依属性执行一次必要计算。

Turing 的 MVR 进一步扩展了 Pascal 架构中所引入的同步多重投射 (SMP) 功能。SMP 专为加速立体和环绕渲染用例而打造。开发者可借助 SMP 技术指定两个视图，而其中的视图相依属性仅限于顶点 X 坐标和用于光栅化的视口。之后，每个视图可被多路投射至一组多达 16 个的预配置投影机（或视口），从而为透镜匹配着色等用例提供支持。

Turing 能够消除对可容许的视图相依属性的限制，并增加支持的视图数，同时还能继续为每个视图提供多达 16 个投影机支持。如需深入了解 SMP 的功能和用例，请参阅 GeForce GTX 1080 白皮书。

## 多视图渲染用例

MVR 最常见的一个用途是作为 Pascal 单遍立体 (SPS) 功能的扩展，专用于加速虚拟现实 (VR) 渲染。SPS 最初仅允许人眼以同一投影方向彼此进行水平偏移。这种头盔显示器 (HMD) 配置既常见也合乎常理，因为人脸非常对称，而 HMD 几乎都使用单个投影平面。许多 HMD 都对双眼使用单个物理显示器。然而，更优质的 HMD 和具有超大视野 (FOV) 的较新设备需要更高的视图灵活性，以便获取 VR 工作负载中仍然可用的冗余几何处理。

图 38 展示了 200 度 FOV HMD 配置，其中使用两个搭载 MVR 的倾斜面板，以便提供更出色的表现力。MVR 的灵活性还可助力标准立体 VR 显示器实现更精确的校准，以便与个人用户的面部对齐。在立体渲染中，人眼仅在 X 轴上彼此进行偏移的简单假设并非绝对准确，实际操作中会存在一些其他非对称情形，而这些情形需要独立投影方能实现最高保真度对准。

图 39 展示了使用 MVR 单遍执行四个阴影深度缓冲渲染的其他示例（左上角）。右上角展示了从同一网格渲染两个角色的示例，其中仅提取一次网格，视图 ID 用于控制单遍生成两个实例。图 39 底部展示了单遍执行层叠式阴影贴图渲染示例。

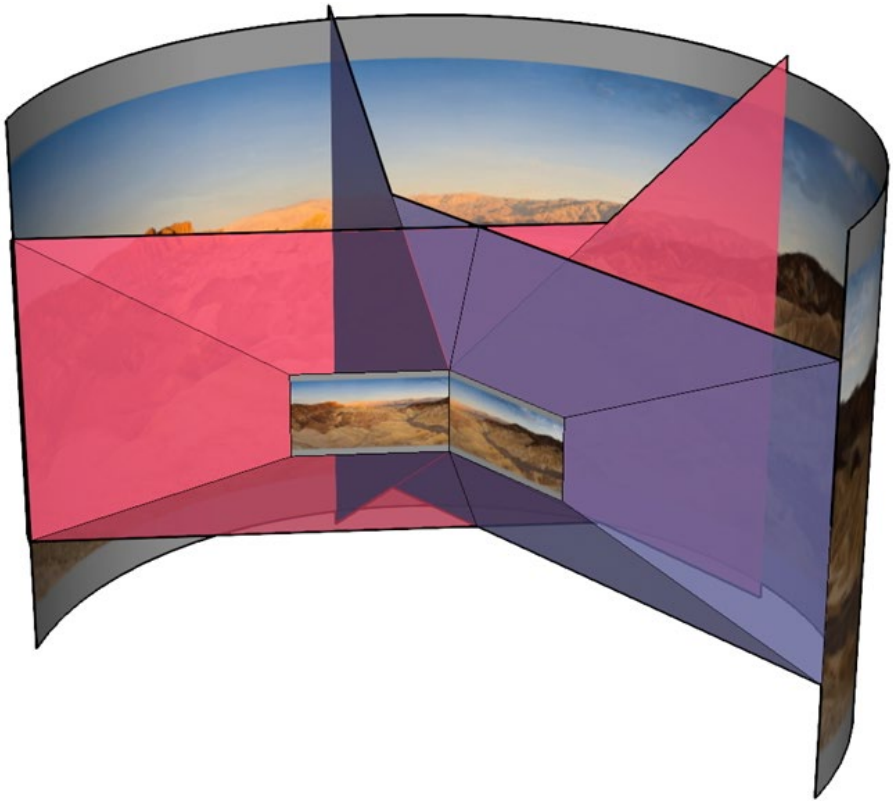


图 38.　200 度 FOV HMD，其中使用两个搭载 MVR 的倾斜面板

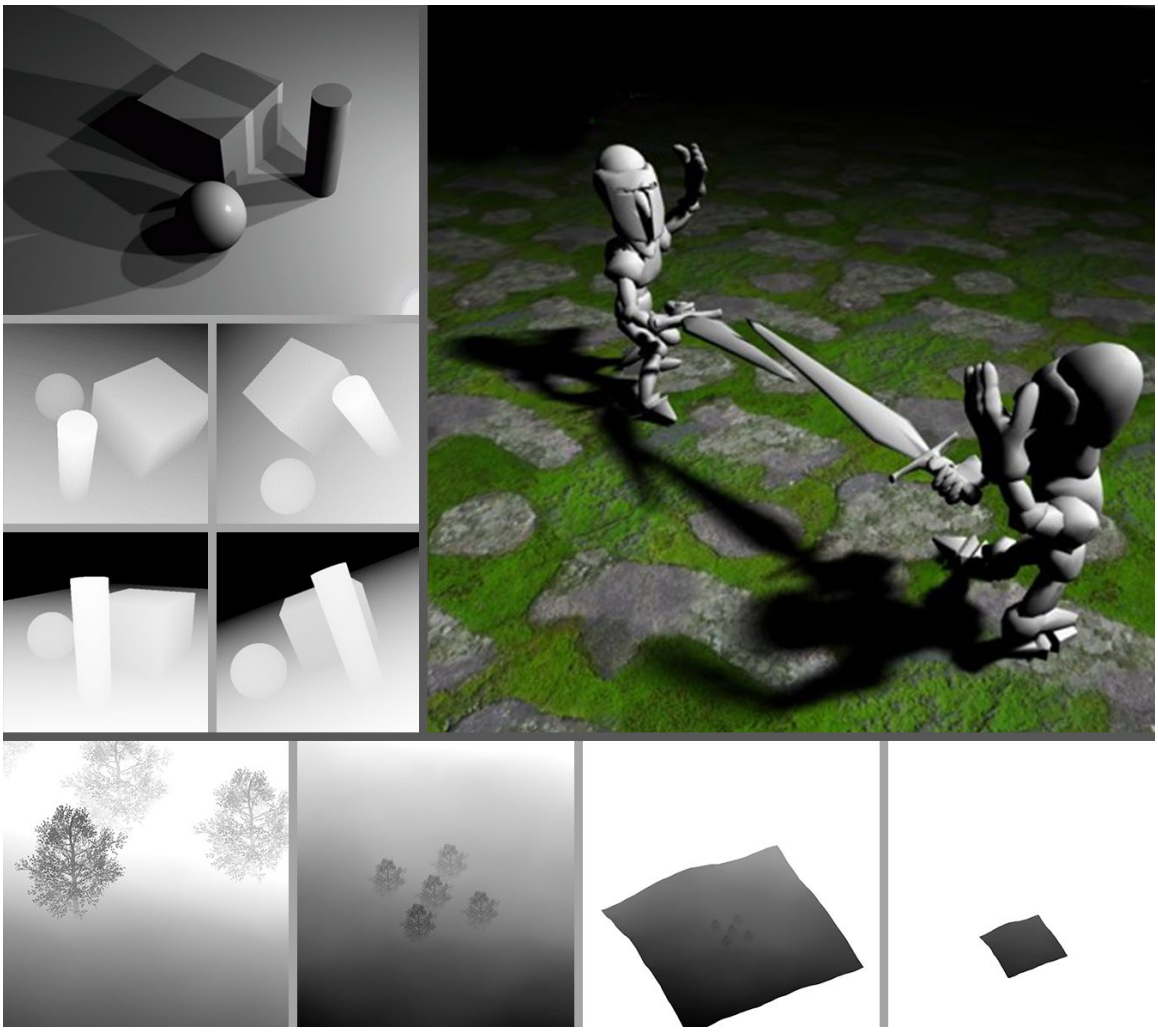


图 39.　MVR 单遍层叠式阴影贴图渲染

# 资源管理和绑定模型

DX12 引入了一种新功能，允许着色器程序直接访问资源视图，而无需执行明确的资源绑定步骤。Turing 已扩展资源支持，并已添加无绑定常量缓冲区视图 (Constant Buffer View) 和无序访问视图 (Unordered Access View)，具体已在 DX12 资源绑定规范第 3 层中进行定义。

Turing 拥有更灵活用的内存模型，允许多种不同的资源类型（如纹理和顶点缓冲区）共存于同一个堆中，从而能够简化应用程序内存管理的各个方面。Turing 支持资源堆的第 2 层。

# TURING 特性增强虚拟现实体验

Turing GPU 架构在虚拟现实 (VR) 技术和头盔显示器 (HMD) 方面均取得了大量显著进步。得益于全新的 Turing 光线追踪、着色和推理技术，VR 性能、沉浸式体验和舒适度均已得到提升。本文其他部分已经详细阐述 Turing 在 VR 方面的进步，此处所列内容仅便于您参考。

Turing GPU 设计可为 USB Type-C 和 VirtualLink 提供硬件支持，这种全新的开放式行业标准可通过单个 USB-C 接口提供驱动 VR 头盔所需的供电、显示和数据传输条件。Turing GPU 能够大幅降低 VR 连接的复杂性，其提供的单接口解决方案还可为传统上不支持多接口的小型设备实现 VR 功能。

另一种有助提升 VR 体验的 Turing GPU 特性是多视图渲染 (MVR)。MVR 是对 Pascal 架构中引入的 SMP 功能的扩展，该技术可对跨越两个不同投影中心的单个几何数据流作出处理，从而更高效地实现 VR 立体显示渲染。Turing MVR 将视口投影数量从两个扩展至四个，使头盔制造商能将额外的视口投影用于倾斜的环绕式侧视图，从而提升沉浸质量。

图 40 展示了面向 VR 的 Turing 特性。

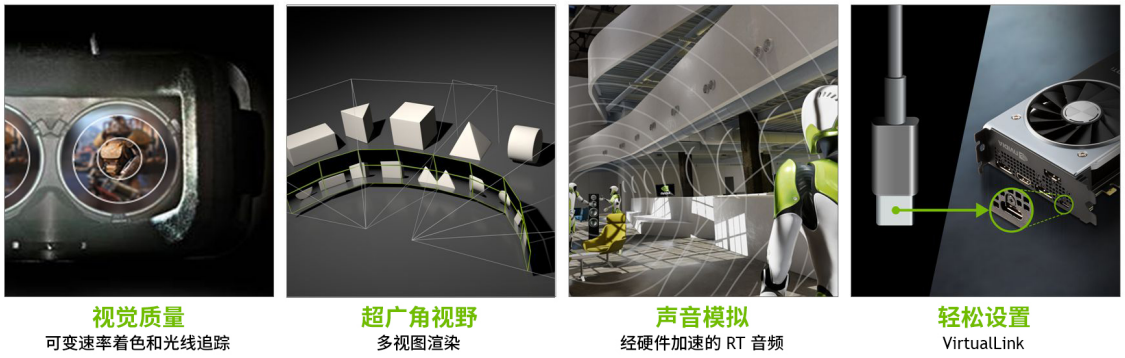



图 40. 面向 VR 的 Turing 特性

注视点渲染技术是全新 Turing VRS 特性在 VR 中的一个用例。VRS 允许开发者控制着色率，以便利用中央凹注视和物体移动等现象减少或增加着色，具体取决于是要提升着色效率还是增加细节。VR 中的注视点渲染使用可变速率着色来减少当前视线范围外的着色，甚至还能增加人眼注视区域内的着色。VRS 可为开发者提供新方法，以便提升 VR 上的沉浸式细节和裁剪效率。

VR 沉浸体验并非仅依赖于图像画面。三维音效对于 VR 沉浸体验也起着至关重要的作用。目前，所有游戏均通过简单的直达声定位来提供 3D 音效。NVIDIA VRWorks™ Audio（与 Pascal GPU 架构一同引入）使用 NVIDIA OptiX™ 软件光线追踪引擎提供双耳声音元素，以实现更出色的间接音效模拟。声音可在表面发生反弹，比直达声音路径更晚到达监听器，由此提供可代表不同类型虚拟环境的混响。通过部署 RT 核心，Turing 可对 Pascal 的 NVIDIA VRWorks Audio 进行扩展，以此为经光线追踪的 NVIDIA VRWorks Audio 提供高达 6 倍的加速。

# 结束语

我们已实现显卡技术的颠覆性创新。全新 NVIDIA Turing GPU 架构是迄今为止极为先进和高效的 GPU 架构。Turing 实现了集实时光线追踪、光栅化、AI 和模拟于一身的全新混合渲染模型。Turing 与新一代图形 API 强强联合，能够为 PC 游戏和专业应用程序实现大幅性能提升与栩栩如生的图形特效。

# 附录 A：
# TURING TU104 GPU

Turing TU104 也与 Turing TU102 GPU 一道发布。TU104 GPU 集成了 TU102 中新引入的所有 Turing 特性，包括 RT 核心、Turing Tensor 核心以及对 Turing SM 所作的架构更改。

完整的 TU104 芯片包含 6 个 GPC、48 个 SM 以及 8 个 32 位内存控制器（共 256 位）。在 TU104 中，每个 GPC 均包含一个光栅单元和四个 TPC。每个 TPC 均包含一个多形体引擎和两个 SM。

每个 SM 均包含全新 RT 核心。与 TU102 一样，TU104 的每个 SM 也包含 64 个 CUDA 核心、256 KB 寄存器堆、96 KB L1 数据缓存或共享内存缓存以及四个纹理单元。完整的 TU104 芯片包含 136 亿个晶体管，同时包含 3072 个 CUDA 核心、368 个 Tensor 核心以及 48 个 RT 核心。此外，TU104 还支持第二代 NVLink 技术。其中包括一个 x8 NVLink 链路，可为每个方向提供 25 GB/秒的带宽（50 GB/秒的总带宽）。图 41 展示了 Turing TU104 的完整芯片图。

TU104 GPU 将会用于不同级别的 GeForce、Tesla 和 Quadro 产品，例如 GeForce RTX 2080、Tesla T4 和 Quadro RTX 5000。

表 4 对 GeForce RTX 2080 和 Quadro RTX 5000 的规格进行了对比。

表 5 列出了 Tesla T4 的具体规格。

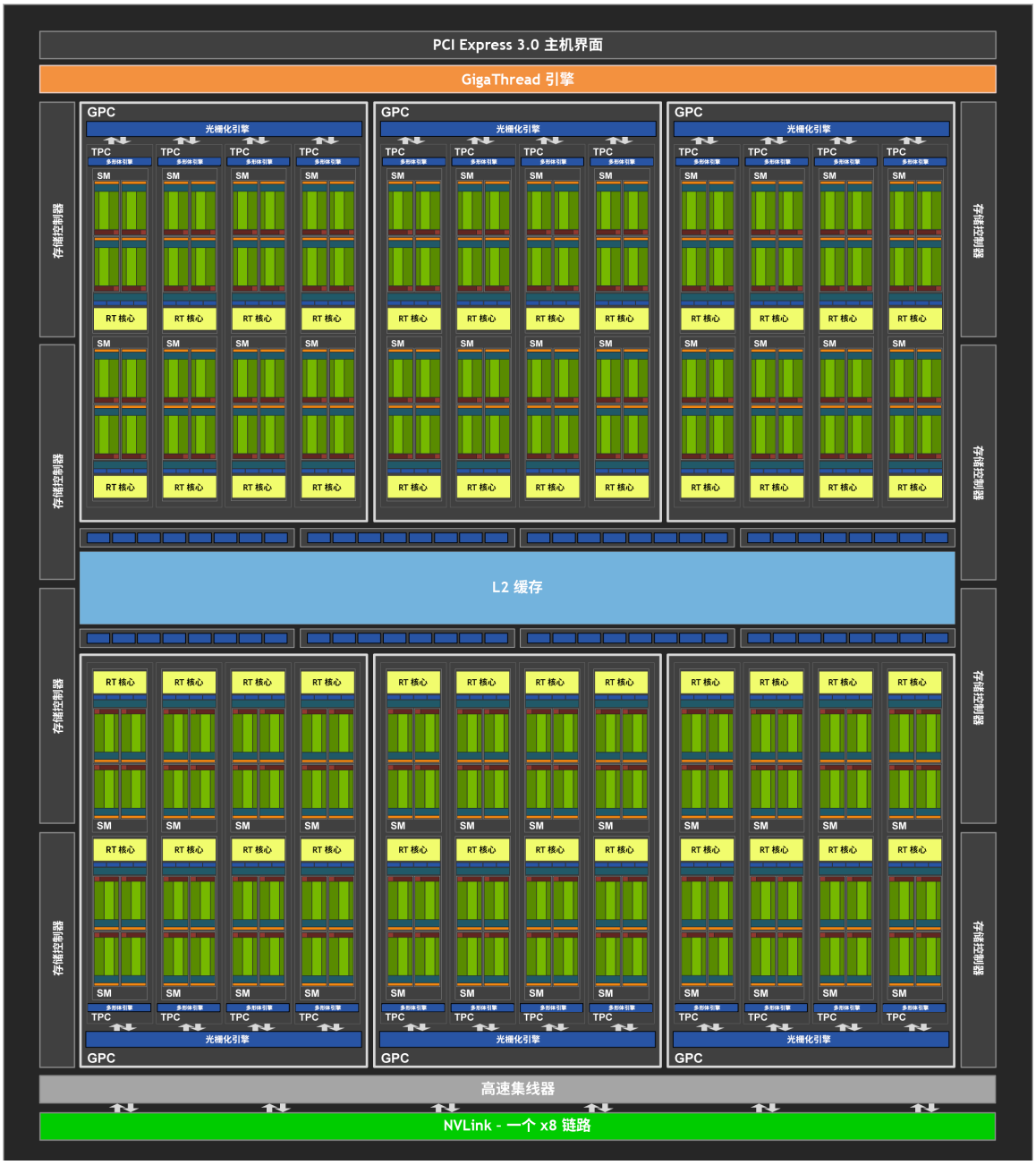



图 41.　Turing TU104 完整芯片图

表 4. NVIDIA Pascal GP104 与 Turing TU104 GPU 规格对比表

| GPU 特性 | GeForce GTX 1080 | GeForce RTX 2080 | Quadro P5000 | Quadro RTX 5000 |
|---|---|---|---|---|
| 架构 | Pascal | Turing | Pascal | Turing |
| GPC 数量 | 4 | 6 | 4 | 6 |
| TPC 数量 | 20 | 23 | 20 | 24 |
| SM 数量 | 20 | 46 | 20 | 48 |
| CUDA 核心数/SM | 128 | 64 | 128 | 64 |
| CUDA 核心数/GPU | 2560 | 2944 | 2560 | 3072 |
| Tensor 核心数/SM | 不适用 | 8 | 不适用 | 8 |
| Tensor 核心数/GPU | 不适用 | 368 | 不适用 | 384 |
| RT 核心数 | 不适用 | 46 | 不适用 | 48 |
| GPU 基础频率 (MHz)（参考版本：Founders Edition） | 1607/1607 | 1515/1515 | 1607 | 1620 |
| GPU 加速频率 (MHz)（参考版本：Founders Edition） | 1733/1733 | 1710/1800 | 1733 | 1815 |
| RTX-OPS (Tera-OPS)（参考版本：Founders Edition） | 8.9/8.9 | 57/60 | 不适用 | 62 |
| 光线投射数量（10 亿条光线/秒）（参考版本：Founders Edition） | 0.89 | 8/8 | 不适用 | 8 |
| FP32 TFLOPS 峰值*（参考版本：Founders Edition） | 8.9 | 10/10.6 | 8.9 | 11.2 |
| INT32 TIPS 峰值*（参考版本：Founders Edition） | 不适用 | 10/10.6 | 不适用 | 11.2 |

| GPU 特性 | GeForce GTX 1080 | GeForce RTX 2080 | Quadro P5000 | Quadro RTX 5000 |
|---|---|---|---|---|
| FP16 TFLOPS 峰值*（参考版本：Founders Edition） | 不适用 | 20.1/21.2 | 不适用 | 22.3 |
| 使用 FP16 累加的 FP16 Tensor TFLOPS 峰值*（参考版本：Founders Edition） | 不适用 | 80.5/84.8 | 不适用 | 89.2 |
| 使用 FP32 累加的 FP16 Tensor TFLOPS 峰值*（参考版本：Founders Edition） | 不适用 | 40.3/42.4 | 不适用 | 89.2 |
| INT8 Tensor TOPS 峰值*（参考版本：Founders Edition） | 不适用 | 161.1/169.6 | 不适用 | 178.4 |
| INT4 Tensor TOPS 峰值*（参考版本：Founders Edition） | 不适用 | 322.2/339.1 | 不适用 | 356.8 |
| 帧缓存大小和类型 | 8192 MB GDDR5X | 8192 MB GDDR6 | 16384 GDDR5X | 16384 GDDR6 |
| 显存位宽 | 256 位 | 256 位 | 256 位 | 256 位 |
| 显存频率（数据速率） | 10 Gbps | 14 Gbps | 9 Gbps | 14 Gbps |
| 显存带宽（GB/秒） | 320 | 448 | 288 | 448 |
| ROP 数量 | 64 | 64 | 64 | 64 |
| 纹理单元数量 | 160 | 184 | 160 | 192 |
| 纹素填充率（10 亿纹素/秒） | 277.3/277.3 | 314.6/331.2 | 277 | 348 |
| L2 缓存大小 | 2048 KB | 4096 KB | 2048 KB | 4096 KB |
| 寄存器堆大小/SM | 256 KB | 256 KB | 256 KB | 256 KB |
| 寄存器堆大小/GPU | 5120 KB | 11776 KB | 5120 KB | 12288 KB |

| GPU 特性 | GeForce GTX 1080 | GeForce RTX 2080 | Quadro P5000 | Quadro RTX 5000 |
|---|---|---|---|---|
| TDP（热设计功耗）* （参考版本：Founders Edition） | 180/180 W | 215/225 W | 180 W | 230 W |
| 晶体管数 | 72 亿 | 136 亿 | 72 亿 | 136 亿 |
| 芯片大小 | 314 $mm^2$ | 545 $mm^2$ | 314 $mm^2$ | 545 $mm^2$ |
| 制造工艺 | 16 nm | 12 nm FFN | 16 nm | 12 nm FFN |
| **注意：** *TFLOPS 和 TOPS 峰值速率基于 GPU 加速频率。 *功率图只表示显卡的 TDP。使用 VirtualLink™ 或 USB Type-C™ 接口另需多达 35 瓦功率，此功率图中并未予以显示。 | | | | |

NVIDIA Tesla T4 是首款基于 Turing 的 GPU，专为数据中心、企业和终端设备中的推理应用程序而设计。Tesla T4 采用的 TU104 芯片包含 5 个 GPC、20 个 TPC、40 个 SM 以及 320 个 Turing Tensor 核心，CUDA 核心总数达 2560 个。此外，Tesla T4 TU104 芯片还包含 256 位显存位宽，显存数据速率为 10 Gbps，总带宽达 320 GB/秒（请参见表 5，了解 Pascal Tesla P4 和 Turing Tesla T4 的规格对比）。

## 表 5. Pascal Tesla P4 和 Turing Tesla T4 对比表

| GPU | Tesla P4 (Pascal) | Tesla T4 (Turing) |
|---|---|---|
| GPC 数量 | 4 | 5 |
| TPC 数量 | 20 | 20 |
| SM 数量 | 20 | 40 |
| CUDA 核心数/SM | 128 | 64 |
| CUDA 核心数/GPU | 2560 | 2560 |
| Tensor 核心数/SM | 不适用 | 8 |
| Tensor 核心数/GPU | 不适用 | 320 |
| RT 核心数 | 不适用 | 40 |
| GPU 基础频率 (MHz) | 810 | 585* |
| GPU 加速频率 (MHz) | 1063 | 1590 |

| GPU | Tesla P4 (Pascal) | Tesla T4 (Turing) |
|---|---|---|
| FP32 TFLOPS 峰值 | 5.5 | 8.1 |
| INT32 TIPS 峰值 | 不适用 | 8.1 |
| FP16 TFLOPS 峰值 | 不适用 | 16.2 |
| 使用 FP16 累加的 FP16 Tensor TFLOPS 峰值* | 不适用 | 65 |
| 使用 FP32 累加的 FP16 Tensor TFLOPS 峰值* | 不适用 | 65 |
| INT8 Tensor TOPS 峰值* | 22 | 130 |
| INT4 Tensor TOPS 峰值* | 不适用 | 260 |
| 帧缓存大小和类型 | 8192 MB GDDR5X | 16384 MB GDDR6 |
| 显存位宽 | 256 位 | 256 位 |
| 显存频率（数据速率） | 6 Gbps | 10 Gbps |
| 显存带宽（GB/秒） | 192 | 320 |
| ROP 数量 | 64 | 64 |
| TDP（热设计功耗） | 75 瓦 | 70 瓦 |
| 晶体管数 | 72 亿 | 136 亿 |
| 芯片大小 | 314 | 545 |
| 制造工艺 | 16 nm | 12 nm FFN |

**注意：** *TFLOPS 和 TOPS 峰值速率基于 GPU 加速频率。

*Tesla T4 基础频率专为 70W TDP 而设计，可在数据中心的服务器机架中高效运转。尽管 T4 能够以更高运转频率处理许多工作负载（如其高速加速频率所示），但基础频率却表明其运转频率最低，该频率通常会出现在异常紧张的推理工作负载中。

# 附录 B：
# TURING TU106 GPU

GeForce RTX 2070 中使用的 Turing TU106 GPU 会于 2018 年 10 月推出。GeForce RTX 2070 旨在提供出类拔萃的卓越性能与能效。TU106 也支持 Turing 架构中新引入的多数关键特性，包括 RT 核心、Turing Tensor 核心以及对 Turing SM 作出的所有架构更改。相较于 TU102 和 TU104，TU106 并不支持 NVLink 或 SLI。

GeForce RTX 2070 全面搭载 TU106 GPU，该 GPU 包含 3 个 GPC、36 个 SM 以及 8 个 32 位显存控制器（共 256 位）。在 TU106 中，每个 GPC 均包含一个光栅单元和六个 TPC。每个 TPC 均包含一个多形体引擎和两个 SM。图 42 展示了 Turing TU106 的完整芯片图。

与 TU102 和 TU104 一样，TU106 中的每个 SM 均包含用于光线追踪的全新 RT 核心。每个 SM 同样包含 64 个 CUDA 核心、256 KB 寄存器堆、96 KB L1 数据缓存或共享内存缓存以及四个纹理单元。完整的 TU106 GPU 包含 108 亿个晶体管，同时包含 2304 个 CUDA 核心、288 个 Tensor 核心以及 36 个 RT 核心。表 6 对 NVIDIA Pascal GP104 和 Turing TU106 作出了对比。


图 42. Turing TU106 完整芯片图

表 6. NVIDIA Pascal GP104 与 Turing TU106 GPU 对比表

| GPU 特性 | GeForce GTX 1070 (GP104) | GeForce RTX 2070 (TU106) |
|---|---|---|
| 架构 | Pascal | Turing |
| GPC 数量 | 3 | 3 |
| TPC 数量 | 15 | 18 |
| SM 数量 | 15 | 36 |
| CUDA 核心数/SM | 128 | 64 |
| CUDA 核心数/GPU | 1920 | 2304 |
| Tensor 核心数/SM | 不适用 | 8 |
| Tensor 核心数/GPU | 不适用 | 288 |
| RT 核心数 | 不适用 | 36 |
| GPU 基础频率 (MHz)<br>（参考版本：Founders Edition） | 1506/1506 | 1410/1410 |
| GPU 加速频率 (MHz)<br>（参考版本：Founders Edition） | 1683/1683 | 1620/1710 |
| RTX-OPS (Tera-OPS)<br>（参考版本：Founders Edition） | 6.5/6.5 | 42/45 |

| GPU 特性 | GeForce GTX 1070 (GP104) | GeForce RTX 2070 (TU106) |
|---|---|---|
| 光线投射数量（10 亿条光线/秒）<br>（参考版本：Founders Edition） | .065/.065 | 6/6 |
| FP32 TFLOPS 峰值*<br>（参考版本：Founders Edition） | 6.5/6.5 | 7.5/7.9 |
| INT32 TIPS 峰值*<br>（参考版本：Founders Edition） | 不适用 | 7.5/7.9 |
| FP16 TFLOPS 峰值*<br>（参考版本：Founders Edition） | 不适用 | 14.9/15.8 |
| 使用 FP16 累加的 FP16 Tensor TFLOPS 峰值*<br>（参考版本：Founders Edition） | 不适用 | 59.7/63 |
| 使用 FP32 累加的 FP16 Tensor TFLOPS 峰值*<br>（参考版本：Founders Edition） | 不适用 | 29.9/31.5 |
| INT8 Tensor TOPS 峰值*<br>（参考版本：Founders Edition） | 不适用 | 119.4/126 |
| INT4 Tensor TOPS 峰值*<br>（参考版本：Founders Edition） | 不适用 | 238.9/252.1 |
| 帧缓存大小和类型 | 8192 MB GDDR5 | 8192 MB GDDR6 |
| 显存位宽 | 256 位 | 256 位 |
| 显存频率（数据速率） | 8 Gbps | 14 Gbps |
| 显存带宽（GB/秒） | 256 | 448 |
| ROP 数量 | 64 | 64 |
| 纹理单元数量 | 120 | 144 |
| 纹素填充率（10 亿纹素/秒） | 202/202 | 233.3/246.2 |
| L2 缓存大小 | 2048 KB | 4096 KB |
| 寄存器堆大小/SM | 256 KB | 256 KB |
| 寄存器堆大小/GPU | 3840 KB | 9216 KB |
| TDP（热设计功耗）<br>（参考版本：Founders Edition） | 150/150 瓦 | 175/185 瓦 |
| 晶体管数 | 72 亿 | 108 亿 |

| GPU 特性 | GeForce GTX 1070 (GP104) | GeForce RTX 2070 (TU106) |
|---|---|---|
| 芯片大小 | 314 $mm^2$ | 445 $mm^2$ |
| 制造工艺 | 16 nm | 12 nm FFN |
| *注意：TFLOPS 和 TOPS 峰值速率基于 GPU 加速频率。 | | |

# 附录 C：

# RTX-OPS 说明

## 混合渲染模型

过去，实时图形依靠光栅化三角形来渲染图像。如今，通过引入 RT 核心与 Tensor 核心，Turing 硬件已能对光照执行实时光线追踪、使用 AI 增强图像，还可实现更多其他应用。随着 2018 年 10 月版 Windows 10 更新中对 DirectX Raytracing 和 Windows ML 技术的引入，图形 API 也沿此方向不断成型。以上更改共同打造出一种全新的渲染模型：混合渲染。在此类模型中，图形应用程序会综合运用传统渲染、光线追踪渲染和 AI 技术，进而实时生成令人惊叹的图像。

如要了解混合渲染的可用操作，我们首先要理解相关工作负载。目前存在多种关键吞吐量。高操作吞吐量对光线追踪和 AI 至关重要，但二者在整个帧时间跨度内都不会用到，所以单纯将这些操作与着色操作进行结合并不会产生有用的指标。首先，我们必须了解处理每个工作负载所消耗的时间（请参见图 43）。


图 43. 一个 Turing 帧时间跨度的工作负载分布

图 43 展示了一个帧时间跨度的工作负载分布示例，该示例基于 Turing 上所运行应用程序中的测量数据，具体而言：

- 会将 DLSS 用作典型的 DNN 工作负载（紫色），我们观察到此工作负载约占帧时间的 20%。在剩余的 80% 的时间内，Turing 均在执行渲染操作（黄色）。
- 在余下的渲染时间中，一段时间将用于光线追踪（绿色），而另一段时间会用于传统的光栅化或 G 缓存评估操作。时间长短因内容而异。根据目前为止所评估的游戏和演示应用程序，我们发现这两部分时长一般会对半分割。因此，在图 43 中，光线追踪约占 FP32 总着色时间的一半。在 Pascal 中，我们会在 CUDA 核心上通过软件模拟光线追踪，每 10 亿条光线约需 10 TFLOPS 的运算性能；而在 Turing 中，我们会在专用的 RT 核心上执行此操作，总吞吐量约为 100 亿条光线，也即对光线追踪的计算力为 100 万亿次运算/秒。
- 第三个要考虑的因素是我们为 Turing 引入了可与 FP32 CUDA 核心并行执行的整数执行单元。通过分析当前游戏中的各类着色器，我们发现每 100 个 FP32 流水线指令约可与 35 个额外的整数流水线指令并行执行。在单流水线架构中，这些指令必须在 CUDA 核心上串行运行并要进行循环，但 Turing 架构现已支持这些指令并行运行。上方时间线假定整数流水线的活跃执行时间约占总着色时间的 35%。

  若给定此工作负载模型，我们就有可能了解 Turing 中的可用操作，并将其与仅有一种操作（而非四种）的前代 GPU 进行对比。这便是 RTX-OPS 的目标，即为混合渲染工作负载提供基于工作负载的实用指标。

# RTX-OPS 基于工作负载的指标阐述

为计算 RTX-OPS，我们将根据每类操作的使用频率减少相应的峰值运算。具体如下：

- Tensor 运算占总时间的 20%
- CUDA 核心使用时间占总时间的 80%
- RT 核心使用时间占总时间的 40%（80% 的一半）
- INT32 流水线使用时间占总时间的 28%（80% 的 35%）

计算方法为：RTX-OPS = TENSOR * 20% + FP32 * 80% + RTOPS * 40% + INT32 * 28%

图 44 展示了 RTX 2080 Ti 每类操作的峰值运算。套入这些峰值运算数后，求得的 RTX-OPS 总数为 78。

计算方法为：14 * 80% + 14 * 28% + 100 * 40% + 114 * 20%。

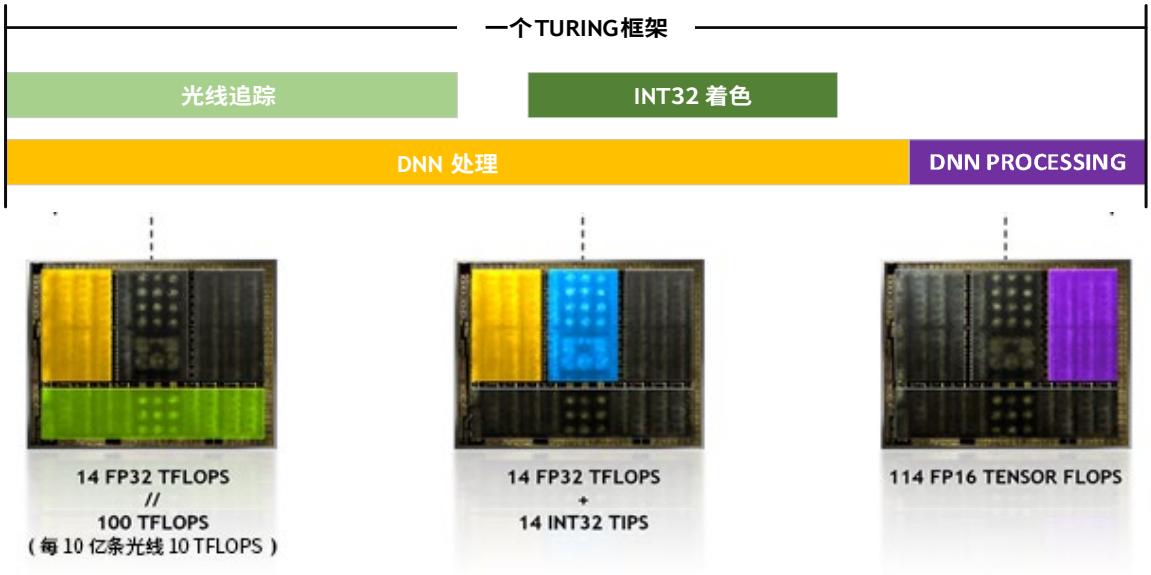



图 44. RTX 2080 Ti 每类操作的峰值运算

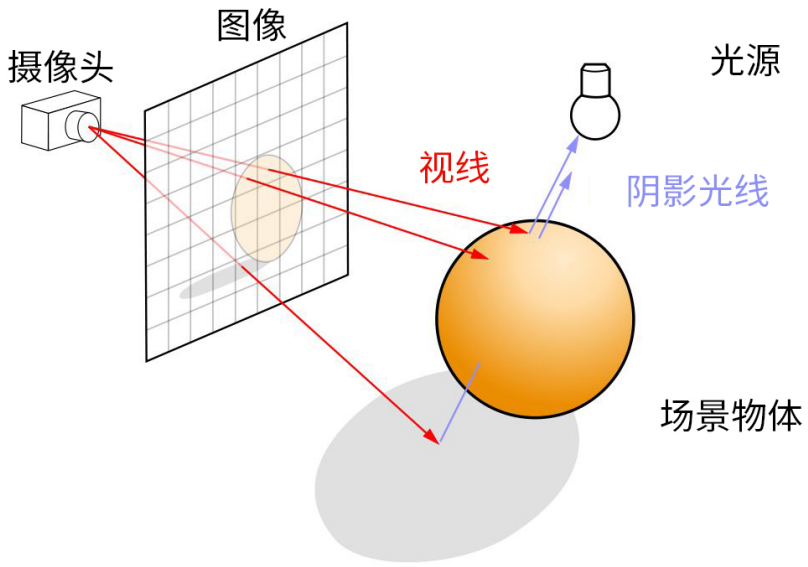
# 附录 D：

# 光线追踪概述

光线追踪是一种渲染技术，可通过渲染具有准确物理属性的反射、折射、阴影和间接照明真实模拟场景照明效果和场景物体。许多光线追踪算法的工作原理可能会与您的设想相反。事实上，我们不是追踪 3D 场景中的光源射入人眼的光线，而是通过 2D 观景式平面（像素平面）将光线从观景式相机（能够确定您看到的场景）向后投射或发射至 3D 场景并返回至光源。这种反向追踪过程远比追踪光源向多个方向发射的所有光线更为高效，因为只有这种通过观景式平面传递至人眼的光线才是渲染场景所需的光线。一些光线直接从光源到达人眼，而其他一些光线可能会由于场景物体的阻挡而形成阴影，还有一些光线会在到达人眼之前在其他物体上形成反射或折射。

当射入场景的光线与物体相交时，我们可根据物体表面相交点的颜色和光照信息计算出各类像素颜色和照明度。我们还须考虑到，不同物体所拥有的表面特性各异，由此或会以不同方式反射、折射或吸收光线。在到达光源之前，光线会在物体上形成反射并击中其他物体，抑或穿过透明物体表面，我们可根据所有相交物体的颜色和光照信息计算出最终的像素颜色。

图 45 展示了光线追踪的基本过程。




图片来源：https://en.wikipedia.org/wiki/Ray_tracing_(graphics)#/media/File:Ray_trace_diagram.svg

图 45.　光线追踪的基本过程

光线追踪在生成逼真场景期间会消耗大量算力，这与射入场景内的光线数和经反射与折射生成的额外光线数极为相关。许多因素均会影响射入场景的光线数，包括但不限于光线追踪目标物体的数量和类型、每帧可使用的 GPU 处理能力、屏幕分辨率以及通过每个像素射入场景的目标光线数。

光线追踪能够生成与相机所拍照片别无二致的图像，多年来一直广泛应用于电影特效领域。事实上，真人动作电影会使用光线追踪将计算机技术生成的效果与相机所拍图像进行无缝融合，而动画电影也可借助光线追踪技术打造极为逼真的画面特效。

# 光线追踪的基本运作机制

如要深入了解光线追踪的运作过程，我们需要认识一些基本原理。下面将从光线投射讲起，该技术用于逼真光线追踪渲染器的核心内环，是一种可见性判断技术。

光线投射实际是光线追踪算法的一个过程，是指从相机（人眼位置）射出一条或多条光线，使之穿过图像平面的每个像素，然后测试这些光线是否与场景中的任何基元（三角形）相交。如果光线穿过像素并到达 3D 场景后击中基元，我们便能确定该光线从起始位置（相机或视点）到基元的距离，还可根据从基元中获取的颜色数据计算出最终的

像素颜色。光线可能会发生反射并击中其他物体，同时从这些物体中获取颜色和光照信息。（有一种相关技术叫作“路径追踪”，这是一种更密集的光线追踪形态，能够追踪成百上千条穿过每个像素的光线，并可在到达光源前追踪光线多次从物体表面反射回来或穿过物体，从而收集颜色和光照信息）。

不同类型的技术和优化可用于加速光线或基元（光线或三角形）求交测试，并减少必须投射的光线数，以此提高性能。否则，若针对场景中的每个基元测试每条光线，不仅效率极低，计算成本也十分高昂。

## 层次包围盒

主流的光线追踪加速技术采用树型加速结构，该结构包含多个分层排列的包围盒，且包围盒中会包含或涵盖数量各异的场景几何体。大型外部包围盒不仅能包含许多基元，也可涵盖大量逐渐缩小的包围盒，且每个包围盒中都会包含更少的几何体。我们将分层排列的包围盒贴切地称为层次包围盒或 BVH。

BVH 通常可排列成一个多层树型结构，每层均有一个或多个节点，其中以顶层的单个根节点为起点，以下各层分别会生成多个后继节点。图 46 展示了 BVH 的树型结构，其中既包含与树中较高节点相关联的较大包围盒，也包含向下遍历时逐渐缩小的包围盒。每个节点均包含在一个包围盒内，且此包围盒还会包含该节点的所有后继节点及与之对应的包围盒。我们使用深度优先树遍历过程针对 BVH 测试每条光线。该过程首先会针对根节点包围盒测试光线（请参见斯坦福兔子模型，顶层节点的大型包围盒已将兔子头部完全包含在内），然后会向下遍历树型结构的后继节点，以便测试该光线将与哪些依次减小的包围框相交。

使用 BVH 方法测试光线或基元可显著减少所需的测试次数。我们不是一味针对场景中的每个基元测试光线，而是仅对树型结构每层的少量包围盒进行测试，直至光线最终击中包含基元的叶节点。

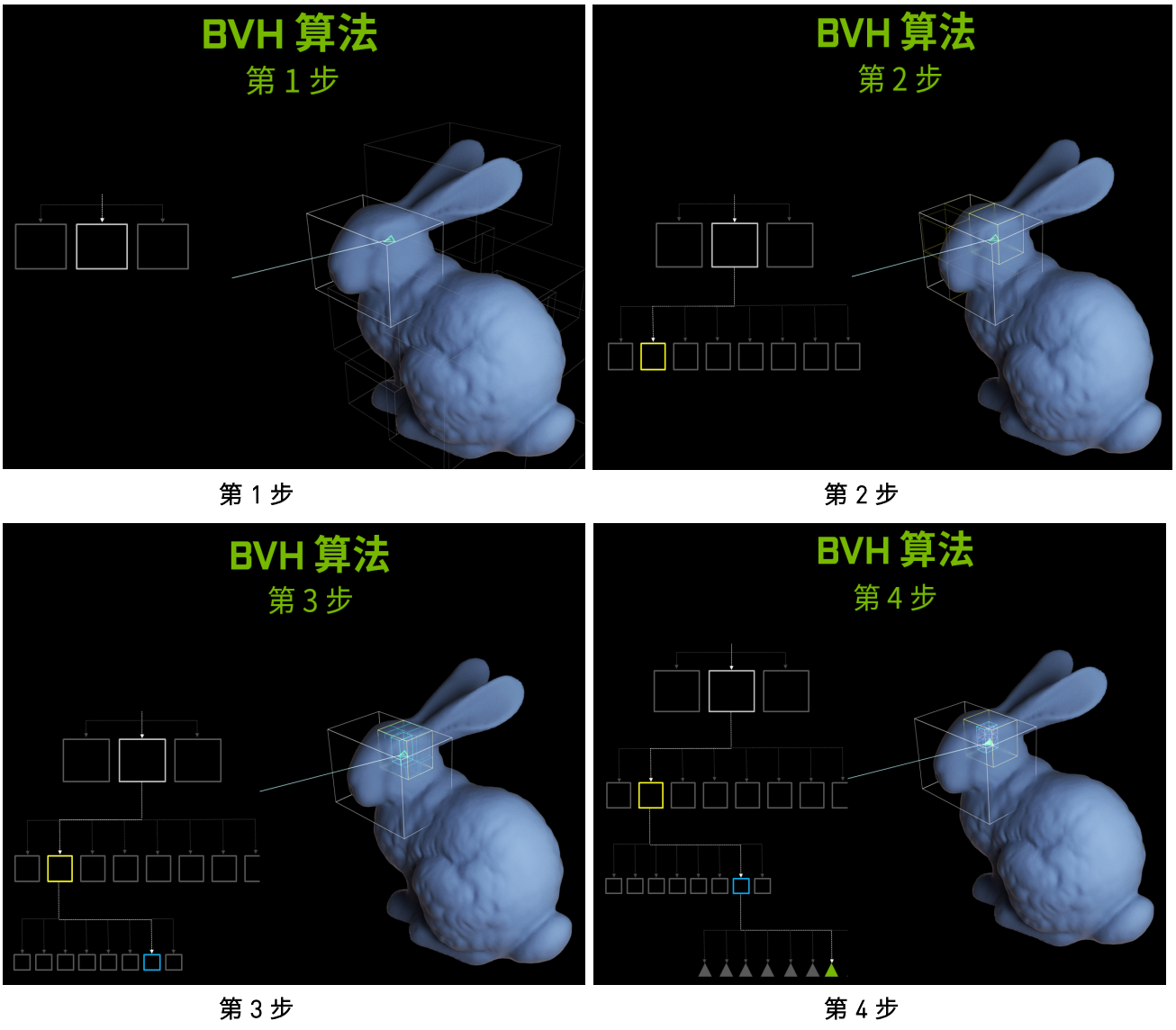



图 46. 树型遍历以及光线与包围盒各层相交的抽象图

在首次渲染场景之前，我们必须根据源几何体创建一个 BVH 结构（称为 BVH 构建）。如果下一帧与前一帧大相径庭，则需构建新的 BVH 结构，以便表示场景中的所有变更。然而，在大多数情况下，我们只需根据某些场景变化便可修改现有的 BVH 结构（称为 BVH 改装），而无需构建全新版 BVH。此改装过程计算成本较低，因此常应用于真实渲染情形中。

# 降噪滤波

不仅加速结构有助提升光线追踪性能，各类高级滤波技术也可提升性能和图像质量，且无需投射额外光线。其中一类滤波技术称为降噪。降噪可显著提升噪声图像的视觉质量，这类噪声图像可能由稀疏数据构成，具有随机伪影、可见量化噪声或其他类噪声。事实上，图像噪声的种类和成因不胜枚举，降噪方法也是多种多样。降噪滤波在减少光线追踪图像渲染时间方面效果尤为显著，并能基于外观无噪声的光线追踪图像生成高保真图像。

目前，NVIDIA 正在利用基于 AI 和非 AI 的算法执行降噪，由此为具体应用挑选出理想算法。我们期待未来能够持续改进基于 AI 的降噪算法并以之取代非 AI 算法，毋庸置疑，这种趋势亦适用于众多与图像相关的其他 AI 应用。

# 光线追踪阴影、环境光遮蔽和反射

阴影是一个至关重要的视觉线索。阴影既能优化地面物体，也可作为照明的一部分，营造场景氛围。目前，大多数游戏均使用阴影贴图，不过我们也已采用一些其他技术来解决部分阴影贴图缺陷。凭借光线追踪和降噪算法的强强联合，Turing GPU 将能攻克阴影贴图中的挑战，例如分辨率不匹配（导致难以生成硬阴影边缘）和接触硬化。

我们可使用百分比靠近软阴影 (PCSS) 和距离场阴影等技术来近似实现接触硬化。尽管 PCSS 计算成本十分高昂，但却无法为任意区域的光线生成完全精确的阴影。距离场阴影仅可用于当前阴影贴图实现中的静态几何体。

借助基于 Turing RTX 的光线追踪加速和快速降噪算法，光线追踪阴影能够取代阴影贴图，并可提供一种实用技术，用于对各类区域光线形成的阴影模拟具备准确物理属性的接触硬化。

如图 47 和图 48 所示，阴影贴图后的实现均匀地对阴影边缘进行了轻微的模糊处理，但未作正确的接触硬化处理。来自同一方向的光线会创建具有可变锥角的光线追踪阴影。如果有需要，光线追踪阴影可提供完全清晰的边缘（左下角为 0 度锥角），或者为不同锥角作正确的接触硬化处理（如右侧所示的 1.5 度和 10 度锥角）。

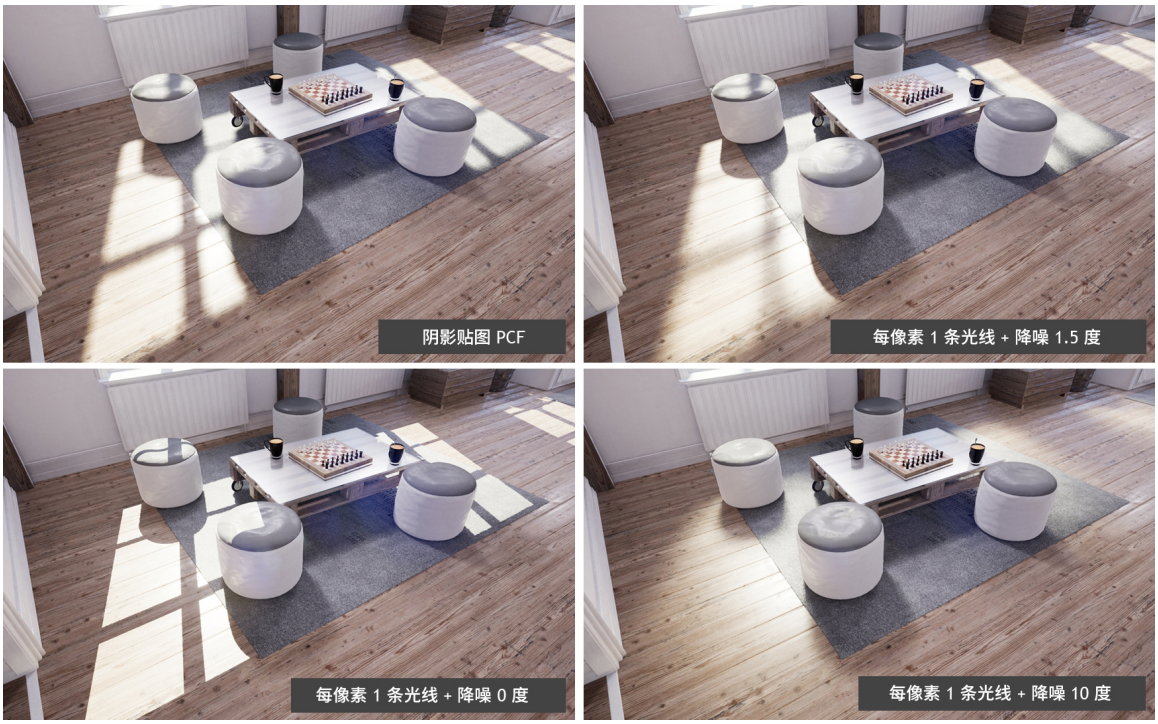



图 47. 比较阴影贴图百分比渐近滤波 (PCF) 与应用降噪算法的光线追踪

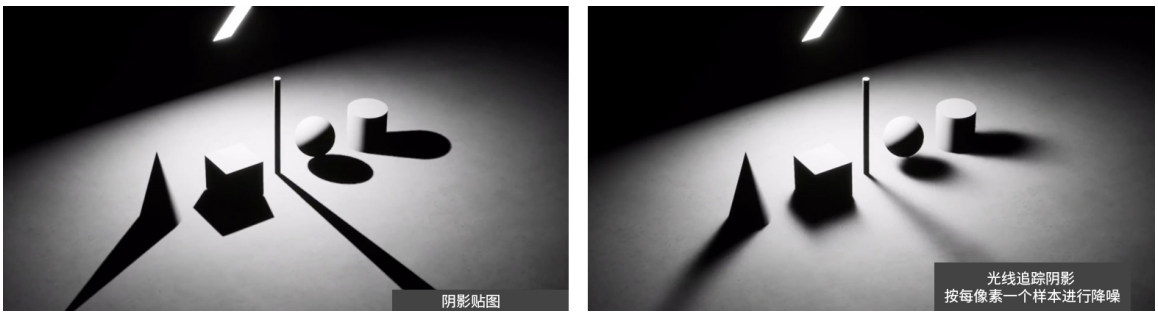



图 48. 比较阴影贴图与每像素抽取 1 个样本应用降噪算法的光线追踪阴影

阴影等环境光遮蔽 (AO) 技术也可用于优化环境中的地面物体。尽管仍有缺陷，但 AO 可通过聚焦物体的折痕及其几何复杂性解决动态全局照明不足的问题，否则这些物体看起来会很扁平。光线追踪环境光遮蔽会对阴影产生微妙的影响。

图 49 比较实时图形领域多年来一直使用的热门屏幕空间环境光遮蔽 (SSAO) 技术和光线追踪环境光遮蔽 (RTAO) 技术。


比较屏幕空间环境光遮蔽与每像素抽取两个样本应用降噪算法的光线追踪环境光遮蔽。请留意沙发、枕头、中间的桌子、人物以及右侧桌子更真实的阴影和清晰度。

## 图 49. 比较屏幕空间环境光遮蔽与光线追踪环境光遮蔽

光线追踪反射可通过光线追踪显著提升视觉质量，这一作用在使用镜面和光滑材质的场景中尤为突出。现如今，即使是最常用的技术（例如融合立方体贴图 (cubemap) 探针的屏幕空间反射技术）也具有一定局限性。屏幕空间反射技术虽然十分经济高效，但却经常会在渲染图像中产生小孔或混乱的伪影。在大多数情况下，立方体贴图 (cubemap) 探针都呈现静态，且分辨率较低，因此我们仅在多采用静态照明的场景中为光滑材质选用此备选方案。借助基于光栅化的技术所能产生的平面反射数量会对其自身有所限制。

而结合降噪技术的光线追踪反射可避免所有这些问题，并产生无伪影的反射，其中包括具有准确物理属性的光泽反射。此外，由于现有的高端 GPU 能以实时帧速率生成一些光线追踪反射，在可承受的范围内，Turing 可支持更广泛地使用光线追踪反射。

如图 50 所示，RTX 光线追踪可渲染具有准确物理属性的反射，带来极强的视觉效果，尤其是拥有许多镜面（平面）和光泽材质的场景。

图 50.　RTX 光线追踪

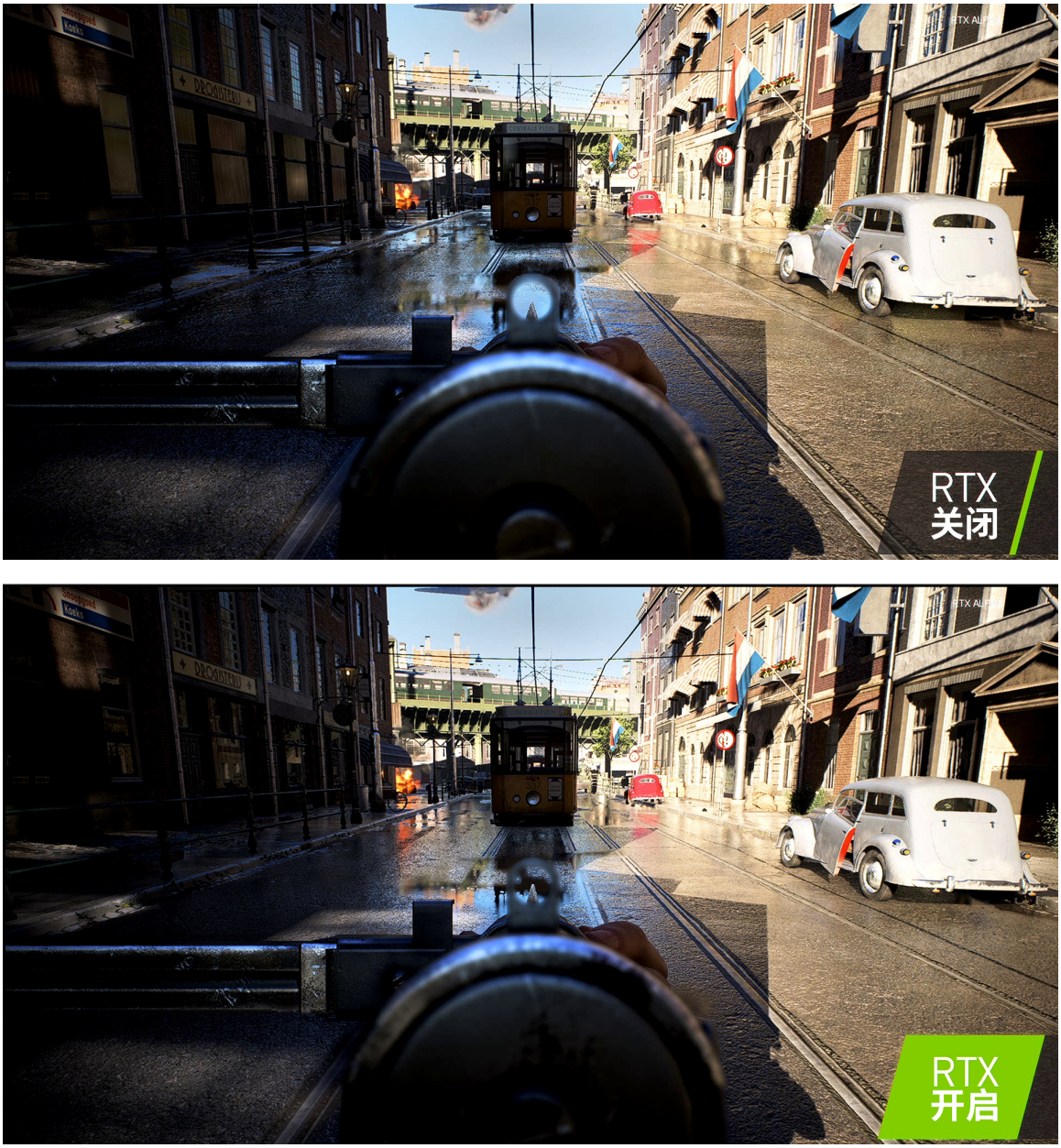
以下图像源自两款即将推出的游戏：“战地 5 (Battlefield V)”和“古墓丽影：暗影 (Shadow of the Tomb Raider)”。这些图像使用 NVIDIA Turing 光线追踪技术提供视觉效果。

- 图 51：“战地 5 (Battlefield V)”中开启和关闭 RTX 的场景
  这一场景暴露了另一个使用非光线追踪反射算法的问题。在此情况下，如果关闭 RTX，场景中会出现局部反射，而部分需通过瞄准镜才能看见的场景就会丢失。而在开启 RTX 的场景下，一切看起来很正常。

- 图 52：“战地 5 (Battlefield V)”中的开启和关闭 RTX 的场景 2
- 图 53：“古墓丽影：暗影 (Shadow of the Tomb Raider)”中的开启 RTX 的场景


“战地 5 (Battlefield V)”（来自发行商 Electronic Arts 和开发商 EA）初始版游戏中的场景。为实现多种游戏效果，DICE 采用了 NVIDIA RTX 技术和 Turing 实时光线追踪技术。在开启 RTX 的场景中，您可以从车身上看见屏幕外爆炸的逼真反射效果。正如关闭 RTX 场景所示，在未使用光线追踪技术的屏幕空间反射中，您绝对看不到这般反射效果。

## 图 51. “战地 5 (Battlefield V)”中开启和关闭 RTX 的场景




这一场景暴露了另一个使用非光线追踪反射算法的问题。在此情况下，如果关闭 RTX，场景中会出现局部反射，而部分需通过瞄准镜才能看见的场景就会丢失。而在开启 RTX 的场景下，一切看起来很正常。

图 52.　“战地 5 (Battlefield V)”中开启和关闭 RTX 的场景 2


“古墓丽影：暗影 (Shadow of the Tomb Raider)”预发行版本中开启和关闭 RTX 的场景。在关闭 RTX 的场景中，孩子们手中的仙女棒并没有投影，使得这些孩子看起来像是悬在空中。而在开启 RTX 的场景下，投影就显得很合理。

图 53. “古墓丽影：暗影 (Shadow of the Tomb Raider)”中的开启 RTX 的场景

简言之，正是凭借众多技术的加持，Turing 才得以实时光线追踪：

- **混合渲染**

  如果渲染步骤仍然有效果，为了减少场景中所需的光线追踪数量，可继续使用光栅化，如若光栅化对渲染步骤不起作用，不若使用光线追踪。

- **降噪算法**

  减少对每像素需投射的光线数量，以此生成精确结果。

- **BVH 算法**

  适用于光线三角形求交运算，即通过减少需经过测试才能找到命中目标的实际三角形数量，使光线追踪操作可更高效地进行。

- **RT 核心**

  上述所有优化均有助于提高光线追踪的效率，但并不足以使其达到实时追踪效果。然而，一旦 BVH 算法标准化，我们便有机会精心设计一款加速器，以期大幅提升光线追踪的操作效率。RT 核心便是这种加速器，它不仅能使 GPU 在光线追踪上的速度提升 10 倍，而且也首次将光线追踪技术引入实时图形领域。

## 通知

本规范中提供的信息在其发布日期之时是准确可靠的。但是，NVIDIA Corporation（以下简称“NVIDIA”）对此类信息的准确性或完整性不作任何明示或暗示的陈述或保证。对使用此类信息的后果或因使用此类信息而可能造成侵犯第三方专利权或其他权利的后果，NVIDIA 概不负责。本出版物将取代之前可能已提供的所有其他产品规范。

NVIDIA 保留随时对这一规范进行纠正、修订、增强、改进以及其他改动及终止提供任何产品或服务的权利，恕不另行通知。客户在下订单之前应获取最新的相关规范并验证这些信息是否为当前信息以及是否完整。

除非 NVIDIA 授权代表与客户另行签署销售协议，否则 NVIDIA 产品的销售受订单确认时所提供的 NVIDIA 标准销售条款与条件的制约。就购买这一规范中提到的 NVIDIA 产品而言，NVIDIA 在此明确拒绝应用客户的任何一般条款与条件。

NVIDIA 产品并非针对医学、军事、航空、航天或生命保障设备而设计，并未授权用于也不保证适合用于上述设备，亦不得用于 NVIDIA 产品之出错或故障合理预计会造成人身伤亡或财产或环境破坏的应用场合。客户如果在此类设备或应用场合中融入和/或使用 NVIDIA 产品，NVIDIA 不承担任何相关责任，风险由客户自行承担。

在未经进一步测试或改动的情况下，NVIDIA 并不表示也不担保基于这些规范的产品适合任何具体用途。对每款产品所有参数的测试不一定由 NVIDIA 进行。确保产品适合客户所计划的应用场合并针对该应用场合进行必要的测试以避免应用场合出现问题或产品失灵，是客户单方面的责任。客户产品设计中的缺点可能会影响 NVIDIA 产品的质量和可靠性，并且可能会导致超出本规范以外的额外或不同的条件和/或要求。NVIDIA 不承担因下列情况造成失灵、损坏、成本或问题相关的任何责任：(i) 以违反本规范的方式使用 NVIDIA 产品；或 (ii) 出于客户产品设计之目的。

本规范对 NVIDIA 专利权、版权或其他 NVIDIA 知识产权并未作出任何明示或暗示的许可。NVIDIA 所发布的有关第三方产品或服务的信息并不构成 NVIDIA 对于使用该产品或服务的许可，亦不构成担保或支持。使用此类信息可能需要获得第三方的专利权或其他知识产权的许可，或者需要获得 NVIDIA 的专利权或其他知识产权的许可。只有在获得 NVIDIA 书面批准的情况下才可以复制本规范中的信息，而且必须毫无改动地复制并附带所有相应的条件、限制条款和通知。

所有 NVIDIA 设计规范、参考板、文件、图纸、诊断信息、列表和其他文档（统称与单称均为“资料”）均“如实”提供。NVIDIA 并未作出与资料相关的明示、暗示、法定或其他形式的保证，并明确否认与非侵权、适销性和特定用途适用性相关的所有暗示保证。尽管客户可能会因任何原因造成损失，但是 NVIDIA 针对本文所述产品向客户承担的全部责任应仅限于该产品的 NVIDIA 销售条款与条件。

### VESA DisplayPort

DisplayPort 和 DisplayPort Compliance Logo、DisplayPort Compliance Logo for Dual-mode Sources 以及 DisplayPort Compliance Logo for Active Cables 是 Video Electronics Standards Association 在美国和其他国家/地区的商标。

### HDMI

HDMI、HDMI 徽标和 High-Definition Multimedia Interface（高清多媒体接口）是 HDMI Licensing LLC 的商标或注册商标。

### ARM

ARM、AMBA 和 ARM Powered 是 ARM Limited 的注册商标。Cortex、MPCore 和 Mali 是 ARM Limited 的商标。其他所有品牌或产品名称均为其各自所有者的资产。“ARM”用于表示 ARM Holdings plc、其运营公司 ARM Limited 和地区子公司 ARM Inc.、ARM KK、ARM Korea Limited.、ARM Taiwan Limited、ARM France SAS、ARM Consulting (Shanghai) Co.Ltd.;ARM Germany GmbH、ARM Embedded Technologies Pvt.Ltd.;ARM Norway、AS 和 ARM Sweden AB。

### OpenCL

OpenCL 是在 Khronos Group Inc. 许可下使用的 Apple Inc. 商标。

## 商标



## 版权所有