arXiv:2301.05262v1 [cs.GR] 12 Jan 2023

神经阴影映射

Sayantan Datta
sayantan.datta@mail.mcgill.ca
麦吉尔大学 加拿大魁北克省蒙特利尔 Meta Reality Labs 美国华盛顿州雷德蒙德

Derek Nowrouzezahrai
derek@cim.mcgill.ca 麦吉尔大学 加拿大魁北克省蒙特利尔

Christoph Schied
Zhao Dong
cschied@nvidia.com
zhaodong@fb.com
Meta Reality Labs 美国华盛顿州雷德蒙德

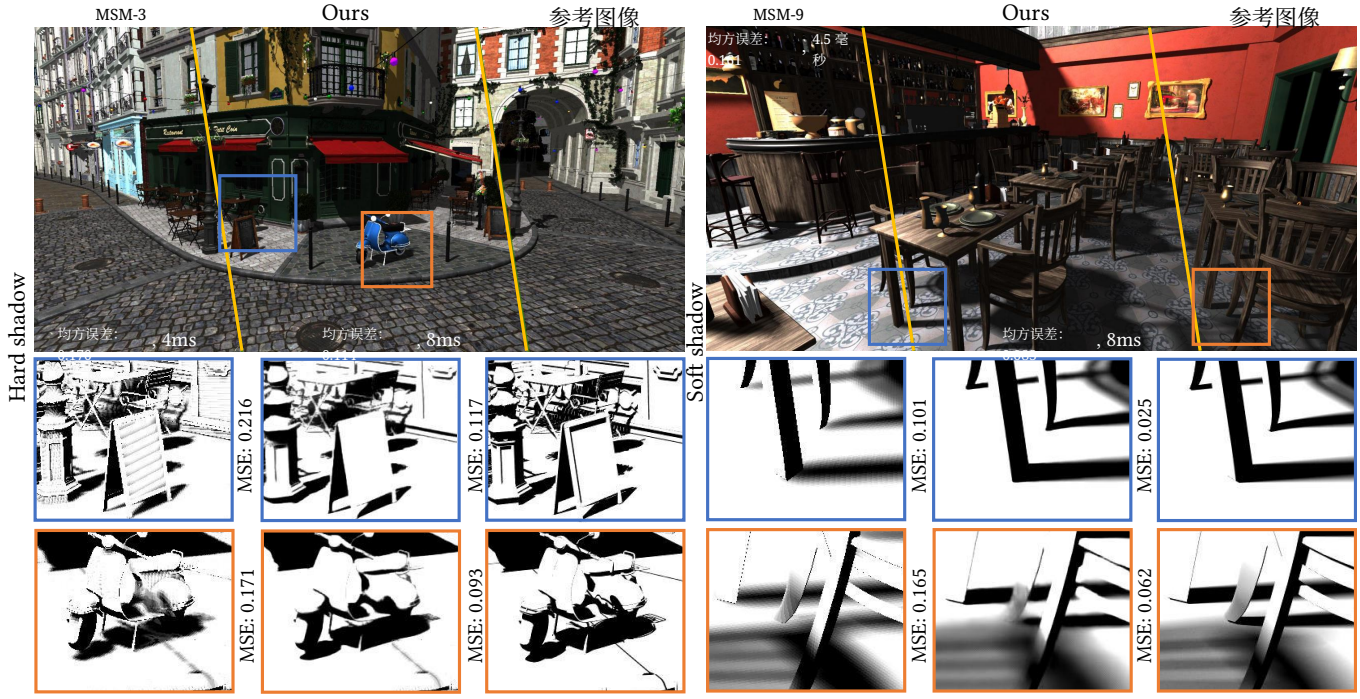

图 1：我们的方法可实现硬阴影和软阴影，其效果接近离线光线追踪，同时在性能-精度谱上取得了理想的平衡。在高性能端，我们生成的结果比 $n \times n$ 矩量阴影贴图（MSM-n）质量更高。我们只需使用标准阴影贴图输入，即可生成视觉效果（及时间效果）接近光线追踪参考图的结果，超越成本更高的去噪交互式光线追踪方法。

摘要

我们提出了一种基础阴影贴图的神经扩展，可快速生成高质量的硬阴影和软阴影。与快速预滤波阴影贴图相比，我们的效果更佳，同时生成的视觉结果可媲美光线追踪硬阴影和软阴影。我们展示了，结合内存带宽感知架构专用化与精心设计的时间窗口训练，可以实现一种快速、紧凑且易于训练的神经阴影方法。我们的技术在内存注重带宽，无需后处理时间抗锯齿或降噪，并支持包含动态视角、光源和几何的场景，同时对未见过的物体保持稳健。

允许出于个人或课堂用途免费制作本作品全部或部分内容的数字副本或纸质副本，前提是这些副本不得用于或分发以获取利润或商业利益，并且每份副本的首页均须附上本声明和完整引文。对于本作品中归其他作者所有的组件，必须尊重其版权。注明出处后，允许摘录。若要以其他方式复制、再版、发布到服务器或分发至列表，则必须事先获得具体许可和/或支付费用。请向 permissions@acm.org 申请许可。
SIGGRAPH '22 会议论文集，*2022*年*8*月7日至*11*日，加拿大不列颠哥伦比亚省温哥华 © 2022年版权归所有者/作者所有。出版权已授权给 ACM。ACM ISBN 978-1-4503-9337-9/22/08...$15.00 https://doi.org/10.1145/3528233.3530700

CCS 概念

• **计算方法** → **可见性**；机器学习；光栅化。

关键词

阴影映射、神经网络

ACM 参考格式：

Sayantan Datta、Derek Nowrouzezahrai、Christoph Schied 和 Zhao Dong。2022年。《神经阴影映射》。载于计算机图形学与交互技术特别兴趣小组会议论文集（*SIGGRAPH '22* 会议论文集），*2022*年*8*月7日至*11*日，加拿大不列颠哥伦比亚省温哥华。ACM，美国纽约州纽约，17页。
https://doi.org/10.1145/3528233.3530700

1 引言

阴影提供重要的几何、深度和着色线索。在资源受限系统上实时渲染硬阴影和软阴影仍是一项挑战。基于预滤波的方法 [Annen 等，2007；Donnelly 和 Lauritzen，2006；Peters 和 Klein 2015] 速度快，但只是近似结果。这些方法容易产生漏光伪影、阴影对比度降低以及接触硬化效果有限等问题。交互式光线追踪 [Keller 和 Waechter 2009] 结合后处理降噪 [Chaitanya 等，2017；Schied 等 2017] 和上采样 [Xiao 等 2020] 可以实现高质量的动态阴影，但即使是速度最快的 GPU 光线追踪器也无法满足交互式图形的性能需求。光线追踪硬件普及率低，以及将 GPU 光线追踪器集成到基于光栅化的管线中会增加工程复杂性，也是其局限之一。基于预计算的方法 [Mildenhall 等，2020；Sloan 等 2002] 通常不支持动态物体或近场光传输，而且需要大量内存。

我们提出了一种基于机器学习的方法，可实时为动态物体生成高质量的硬阴影和软阴影。我们的方法无需光线追踪硬件，性能出色（< 6毫秒），内存占用低（< 1.5MB），且易于部署在普通低端GPU硬件上。我们利用“原生”基于光栅化的阴影映射（即不使用级联等）的输出，生成时间上稳定的硬阴影和软阴影。我们根据多种场景中的半影尺寸统计数据，设计了一种紧凑的神经网络架构。该网络训练速度快，并能泛化到未见过的动态物体。与最先进的高性能基于预滤波的方法相比，我们的方法在支持动态场景的同时提升了画质，效果接近参考图像质量。

我们展示了，精心设计的特征工程、应用感知和内存感知的架构设计，再加上一种新颖的时间稳定性损失函数，可构成一个具备诸多优势的系统：除了紧凑且高性能之外，我们的输出还无需后处理时间抗锯齿（TAA），从而进一步降低渲染器的带宽需求。我们以光源尺寸作为网络参数，使单个训练好的网络能够表示硬阴影和软阴影的变化。我们在多个包含动态几何体、相机和光源的场景中展示了该方法的有效性。我们的结果始终优于常用的交互式方法，也可媲美速度慢得多、对系统和硬件要求更高的交互式光线追踪和基于降噪的管线。最后，我们讨论了可进一步缩小网络规模并降低运行时开销的场景相关优化。

2 相关工作

阴影映射 [Williams 1978] 及其变体是适用于动态场景中点光源和方向光源的高效阴影方法。阴影贴图分辨率和投影会导致阴影混叠伪影，而相应的解决方案（例如深度偏移 [Dou et al. 2014; King 2004]）又会带来其他问题与权衡。现代阴影映射依赖精心设计的系统，将多种级联贴图 [Engel 2006; Zhang et al. 2006]结合起来。关于这一主题的全面综述，请参阅 [Eisemann et al. 2011]。

基于滤波的方法会对基于深度的可见性进行预滤波（在光源空间中），以减少混叠。一种简单的方法是为附近的深度样本 [Reeves 等人 1987]赋予权重；这种百分比渐近滤波仍是交互式应用中常用的技术。近期的一种变体会根据遮挡物与接收体的相对位置调节滤波器大小，用于近似软阴影 [Fernando 2005]。近来，一类新的滤波方法开始用统计代理取代二元深度样本，从而支持更复杂的预滤波 [Annen 等人，2007、2008；Donnelly 和 Lauritzen 2006]，并粗略近似软阴影。矩量阴影贴图 [Peters 和 Klein 2015] 是这类方法中最先进的技术，但在深度复杂度较高的场景中可能会出现条带伪影、混叠和漏光。

屏幕空间方法处理G缓冲区（包括屏幕投影阴影贴图数据），利用图像空间局部性和图形处理器并行化，在延迟着色管线中实现高效滤波。在此，要实现准确滤波，必须确定一个可能具有各向异性的滤波核（这是透视畸变所致），因此其效果会随视角 [Zheng and Saito 2011] 和像素深度 [MohammadBagher et al. 2010]而非线性变化。我们的方法同样处理图像空间G缓冲区数据，但我们改为从数据中学习组合滤波器。高保真软阴影也受益于对发射器和着色点两者的遮挡物深度估计；其中，前者可直接从阴影贴图获取，而后者则可通过对投影阴影贴图进行最小值滤波 [2018] 或平均值滤波 [2010] 来近似。同样，我们依靠学习卷积层和池化层的组合来建模这些深度估计的影响。

光线追踪硬件为动态硬阴影和软阴影开辟了一条令人振奋的新途径。然而，这些方法仍然耗能较高，通常需要后处理降噪（传统方法 [Schied 等人 2017]，或基于机器学习的方法 [Chaitanya 等人，2017年；Munkberg 和 Hasselgren 2020]），并结合时间抗锯齿（TAA）[Edelsten 等人，2019年；Xiao 等人 2020]，才能实现有限的交互性。

3 概述

总体而言，我们的方法十分直接；我们先使用 G缓冲区和阴影映射通道生成一组屏幕空间缓冲区，再将其作为输入传递给网络。在训练过程中，将网络输出与光线追踪阴影这一目标进行比较。尽管方法直接，但如果仅使用 UNet [Nalbach 等，2016；Ronneberger 等， 2015] 而不采用我们提出的训练

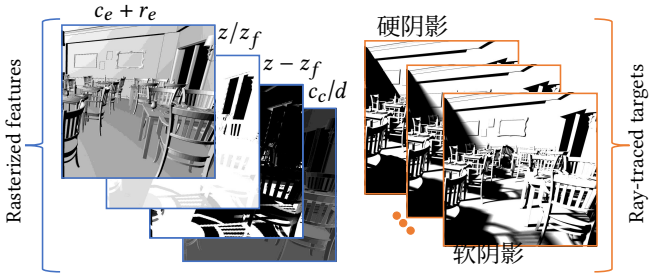

图2：监督学习对的可视化。网络输入是根据发射器尺寸（r_e）调制的光栅化缓冲区。目标则是根据相应的发射器尺寸通过光线追踪生成的。

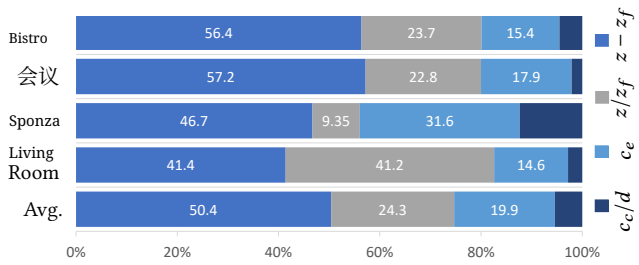

图3：不同场景下所选特征的相对敏感度。

和优化方法，将得到一个时间稳定性不足、受带宽限制、占用空间大（>25MB）且速度过慢（>100ms）的网络，无法用于实时应用。因此，我们的方法重点是在尽量减少对质量影响的同时，审慎作出选择，以保留内存带宽。

我们使用屏幕空间缓冲区作为特征，并以相应的光线追踪阴影作为目标来训练网络。这种方法易于集成到渲染管线中，同时也为将其集成到超采样 [Xiao 等人 2020] 和神经着色 [Nalbach 等人 2016]中留有余地（这可能是未来的工作）。我们的方法也适用于分块渲染，这种方式在移动设备中颇受欢迎。我们提出了一种方法，用于选择一组精简的特征，在不增加内存带宽的前提下保留必要信息。随后，我们展示了一种简单技术，可在单个网络中对不同光源尺寸的阴影进行编码。我们设计了一种损失函数，无需使用历史缓冲区即可增强网络的时间稳定性，从而进一步降低带宽需求。我们还提供了多种网络架构优化方案，以降低内存和计算需求。尽管我们的通用架构支持灵活的光源尺寸，但我们也展示了一种针对固定光源尺寸进一步优化网络的方法，从而能够进一步精简网络层。

3.1 监督训练样本对

我们使用监督学习来训练神经网络。训练示例分别通过光栅化和光线追踪生成特征和目标。光栅化流程包括 G 缓冲区通道，随后是阴影映射通道。二者共同生成以下屏幕空间缓冲区：

- 视空间深度 d 和法线 $\mathbf{n}$，
- 光源至遮挡物深度 z 和光源空间法线 $\mathbf{n_e}$，
- 像素至光源距离 z_f、球形光源的光源半径（大小） r_e，以及
- 法线 $\mathbf{n}$ 与光源方向的点积以及法线 $\mathbf{n}$ 与观察方向的点积 c_e， c_c。

光线追踪通道使用蛮力蒙特卡罗采样和轻度高斯滤波器，生成收敛的硬阴影和软阴影图像。光线追踪目标还采用了 8 倍多重采样抗锯齿（MSAA）。不过，我们在光栅化流程中不使用 MSAA，因为我们希望网络能够从目标图像中隐式学习抗锯齿。

3.1.1 柔和度控制. 我们训练一个网络来预测柔和度各异的一系列阴影。请注意，光栅化产生的相同输入既用于生成硬阴影，也用于生成软阴影。柔和度通过一个表示发射器尺寸的标量参数（连续取值）进行控制。训练时，发射器尺寸编码为 0 到 4 之间的整数纹理，其中 0 表示点光源，4 表示最大尺寸的发射器（直径为 50 cm）。相比于将标量值作为额外的恒定屏幕空间缓冲区传递给网络，更节省带宽的做法是将标量作为直流偏移添加到已有缓冲区中。我们选择将发射器尺寸的直流偏移（r_e）添加到余弦纹理（c_e）中。网络目标也会根据所选发射器尺寸进行相应调整。参见图2。推理期间，网络接受一个介于 0 和 4 之间的标量，并在训练时使用的离散发射器值之间进行内插。

3.2 特征选择

通过网络转储光栅化过程的内容虽然可行，但带宽利用效率低，而且会使网络评估耗时增加 2.5 毫秒。这种低效技术需要在主网络之前级联一个特征提取网络，并对两个网络进行端到端训练。特征提取网络由若干层 1×1 卷积组成，可将光栅化器输出的全部 15 个通道压缩至 4 个。1×1 卷积层相当于跨通道的全连接层，不会在像素之间执行卷积。我们测试了一个深度为 2 层的 1×1 卷积网络，在包含硬阴影和软阴影的一组场景中，记录到的总体误差为 6.64×10^{-3}。

我们的方法通过系统地评估和选择光栅化输出缓冲区，省去了特征提取网络。首先，我们引入敏感度的概念，这是一种用于量化特征重要性的指标。敏感度 衡量输入中的微小扰动导致网络输出发生的变化。直观来说，如果某个通道对解释输出变化的贡献较小，其敏感度就较低。输入通道 f_i 的绝对敏感度 S_i（针对 i^{th}）定义为

$$S_i = \mathbb{E}\left[\frac{(\phi(f_i+\epsilon_i)-\phi(f_i))}{0.1\sigma_i}\right],\quad \epsilon_i \sim \mathcal{N}(0,0.1\sigma_i) \tag{1}$$

其中， ϕ 是网络，随机扰动纹理 ϵ_i 通过对正态分布进行采样获得。 i^{th} 通道对应的标准差σ_i 通过汇总数据集中该通道的所有像素进行经验估计。重复应用该公式数次，以减少采样噪声。为了比较不同训练实例之间的敏感度，我们将**相对敏感度** 计算为 $s_i = S_i/\sum_i S_i$。

以相对敏感度为衡量标准，我们要解决的问题便是从特征集合 $\{d,\ z\ \ z_f\ c_e\ c_c\} + \{z-z_f, z\ z_f\ c_c\ d\ \ \ \}$ $\mathbf{n}$，$\mathbf{n_e}$,/,/,$\mathbf{n}\cdot\mathbf{n_e}$中选出一个特征子集。第一组缓冲区直接由光栅化获得，而第二组则由第一组组合而成。我们采用分层方法来选出最佳特征。第一轮中，我们使用集合 U 中的所有缓冲区训练网络，并剔除相对敏感度较低的缓冲区。我们重复此过程，直至所有缓冲区的敏感度均高于 1.5%。更多信息请参阅补充材料[1] 第 1.0.2 节

[1]附于文末

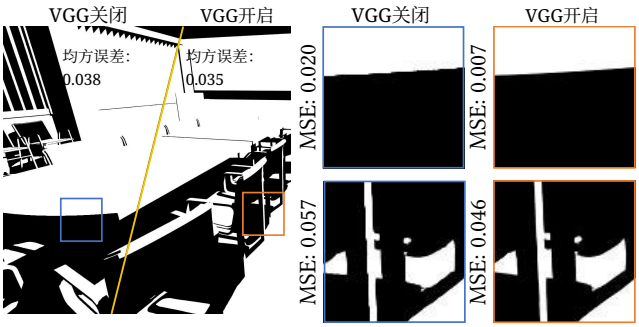

图4：VGG损失对最终输出的影响。VGG损失能够让几何体边缘和阴影轮廓更加锐利。

。最终的缓冲区集合如图3所示。我们获得的误差（6.67×10^{-3}）几乎与进行特征提取时相同，同时节省了额外的 2.5 毫秒。

3.3 损失函数与时间稳定性

损失函数在定义网络特性方面发挥着两个主要作用。首先，它使网络更好地拟合阴影轮廓和几何体的硬边缘，本质上实现了后处理抗锯齿。其次，我们的损失函数无需在训练和推理时使用任何历史缓冲区，就能提高时间稳定性。这种方法不仅节省了内存带宽，也更易于集成到分块渲染器中。

我们通过逐像素差异与 VGG-19 [2015] 感知损失的加权组合，实现了第一个目标。图4展示了 VGG 损失对最终输出的影响，清楚地显示了 VGG-19 对硬边缘产生的抗锯齿效果。

现有方法通常在推理时使用历史缓冲区来更好地辅助网络 [Edelsten 等人，2019；Xiao 等人 2020]，并在训练时重塑损失函数 [Holden 等人，2019；Kaplanyan 等人 2019]。在我们的方案中，我们不使用历史缓冲区，而是在训练时对输入缓冲区施加随机扰动，以重塑损失景观。时间不稳定性源于阴影贴图混叠，即阴影贴图纹素与屏幕像素并非一一对应。因此，相机或发射器的微小移动可能导致深度比较发生较大变化，尤其是在阴影轮廓附近。受噪声到噪声训练的启发 [Lehtinen 等人 2018]，除了传统的监督学习样本对，我们还训练网络从含噪输入对中学习。如图5所示，我们的网络能够内在地学会抑制由微小扰动引起的剧烈变化，同时对整体质量影响极小。在每次训练迭代中，我们按与场景距离和发射器大小成比例的微小数值扰动相机和发射器的位置。对于每次扰动，我们都会收集输入缓冲区用于训练。目标值仅针对其中一次扰动进行计算。我们对每次扰动后的输入运行网络，并将扰动后输出之间的差异最小化，作为额外的损失函数。所有处理扰动输入的网络实例共享相同的权重，而反向传播仅通过其中一个实例进行，

$$\mathcal{L} = L(x_0, \tilde{x}) + \sum_{i=1}^{p} L(x_0, x_i), \tag{2}$$

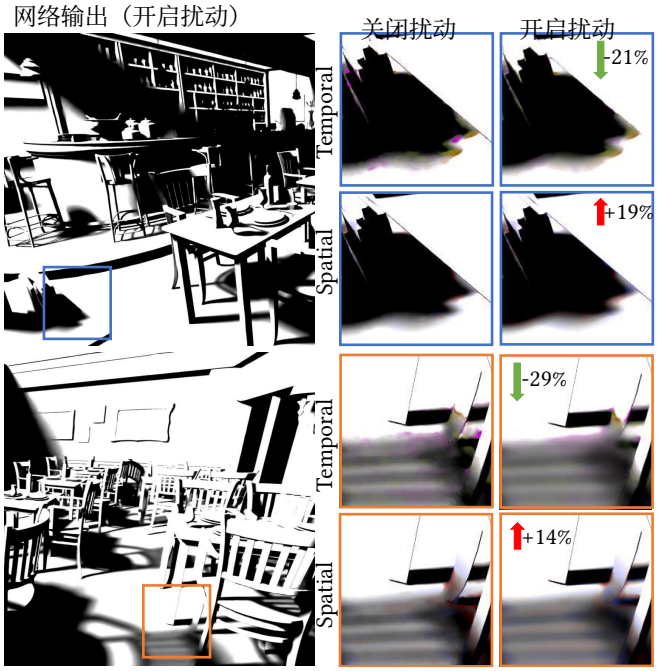

图5：比较扰动损失对时间和空间的影响。如局部裁剪图所示，应用扰动损失可降低时间不稳定性，但会增加空间模糊。我们通过比较连续帧之间的网络输出，测量时间不稳定性；通过比较网络输出与参考图像，测量空间误差。时间不稳定性和空间误差分别用紫色/金色和红色/蓝色伪彩色表示。

其中 $L(y,\ \tilde{y}) = \alpha \cdot |y - \tilde{y}| + (1-\alpha) \cdot VGG19(y,\ \tilde{y})$，$x_i$ 和 $\tilde{x}$ 分别是网络输出和目标。只有一个网络输出——x_0 启用了反向传播。我们设置了 $\alpha = 0.9$ 和扰动数量 $p = 3$。

3.4 时间稳定性测量

有多种技术可用于衡量相对于单个参考帧 [Andersson 等人，2020 年；Wang 等人 2003] 或参考视频 [Mantiuk 等人 2021] 的感知相似度。这些技术衡量时空差异，以反映屏幕空间和时间维度上的整体重建质量。由于我们牺牲空间质量来换取时间稳定性，使用这些指标可能无法体现时间不稳定性的降低，即使扰动损失函数带来了明显的时间稳定性视觉改善也是如此。因此，我们制定了自己的指标，只衡量时间变化，而不考虑与参考图像的空间相似度。为了衡量闪烁，我们计算运动矢量调整的逐像素时间差异 [Yang 等人 2020]。由于闪烁可以量化为帧间像素强度的突变，我们将时间像素差异通过指数函数处理，以加重对较大差异的惩罚。我们对所有

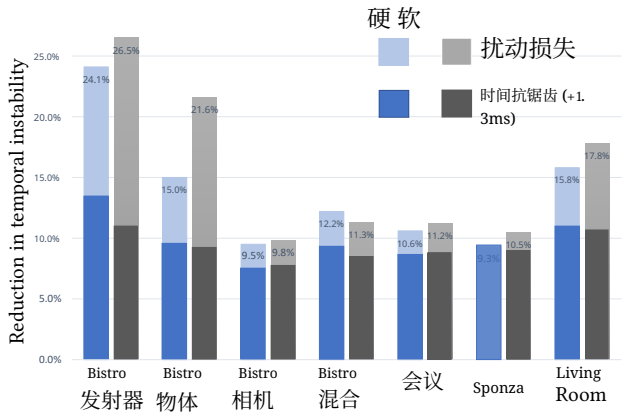

图6：比较扰动损失与时间抗锯齿在降低时间不稳定性方面的效果。参考图像是在未使用扰动损失或时间抗锯齿的情况下训练得到的。时间抗锯齿作为网络（未使用扰动损失训练）评估后的额外处理阶段实现，需额外耗时 1.3 毫秒。

像素和帧上的结果进行汇总，得到

$$E = \frac{1}{P}\sum_{p,t}\{\exp(\alpha D_t(p)) - 1\}\ , \tag{3}$$

其中，$D_t(p) = |I_t(p) - I_{t-1}(m(p))|$ 表示时间 t 和 m两帧之间的逐像素差异，而$p(\ p)$则抽象表示在前一帧像素 $\alpha = 3$处经运动向量调整后的查找。我们设定 P，它控制对随时间发生的大幅强度变化的惩罚；归一化因子为像素总数。我们会排除深度和法线与其重投影结果不匹配的像素。

图6对比了我们的损失函数与 TAA（1 个上一帧）对时间稳定性的影响：我们的扰动损失在动态光源场景中带来的时间稳定性提升最为显著；在动态视角下，提升虽较小，但仍不可忽视。

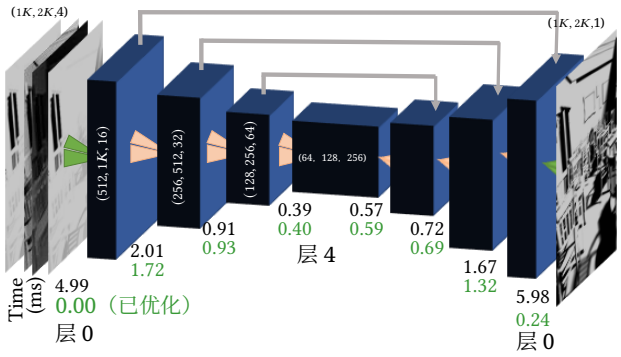

图 7：针对1024×2048 输入的逐层性能优化。

3.5 网络架构与优化

原始 UNet [Ronneberger 等人 2015] 提出的架构速度太慢（>100 毫秒），无法适用于实时图形管线。因此，我们首先着手精简网络。我们的通用网络有 5 层，不过每层由一个 3x3 卷积和一个 1x1 卷积层组成，而非标准的两个 3x3 卷积。与原始 UNet 的一大不同之处在于，我们使用双线性插值代替计算开销较大的转置卷积进行上采样。我们还使用代数和代替拼接层，在解码器侧合并跳跃连接。使用求和层的一个积极副作用是减少了解码器侧的隐藏单元数量。经过这些修改，网络大小从 25MB 降至仅 2.5MB，同时运行时间缩短至 28 毫秒。将网络量化为半精度后，大小进一步缩减至 1.5MB，运行时间降至 17 毫秒。

在不影响性能的情况下，其他用于改善时间稳定性的修改包括使用平均池化代替最大池化，以及移除第一层中的跳跃连接。用平均池化替代最大池化可以减少处理过程中的极端值，并使输出更加平滑。由于原始阴影贴图深度值容易产生混叠噪声，移除第一个跳跃连接可确保含噪输入不会直接影响输出。

在此阶段，我们分析网络在进一步优化前的性能和误差。在集成测试场景上，我们的验证误差为 6.67×10^{-3}。从图 7 可以看出，第一层（组合编码器-解码器）所需时间比其余各层的总和还要长。从内部层（#4）到外部层（#0），分辨率扩大了四倍，而通道数量减半。因此，从内部层向外部层移动时，有效缓存内存带宽翻倍；然而，随着分辨率提高，内存操作也更容易出现缓存未命中。实际上，向外部层移动时，运行时间增加了一倍以上。请参阅补充材料第 3 节。

我们进一步优化了耗时明显更长的第一层。一个简单的方法是在 UNet 的编码器侧用下采样器替换第一层，并在解码器侧用上采样器替换第一层。然而，简单的下采样和上采样会丢失输入中包含的信息，也会产生不够清晰的输出。我们改为将 2x2 的像素方块展平为 4 个独立通道，并将重构缓冲区用作输入

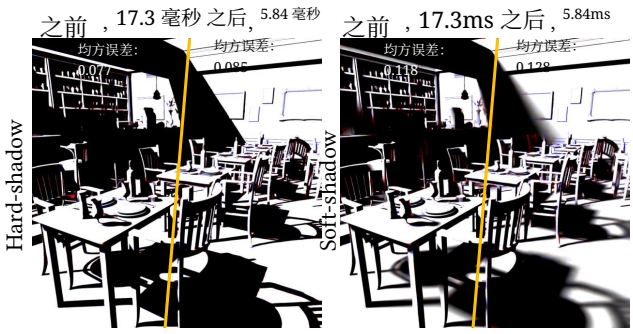

图8：展示第 3.5 节中所有优化效果的图。

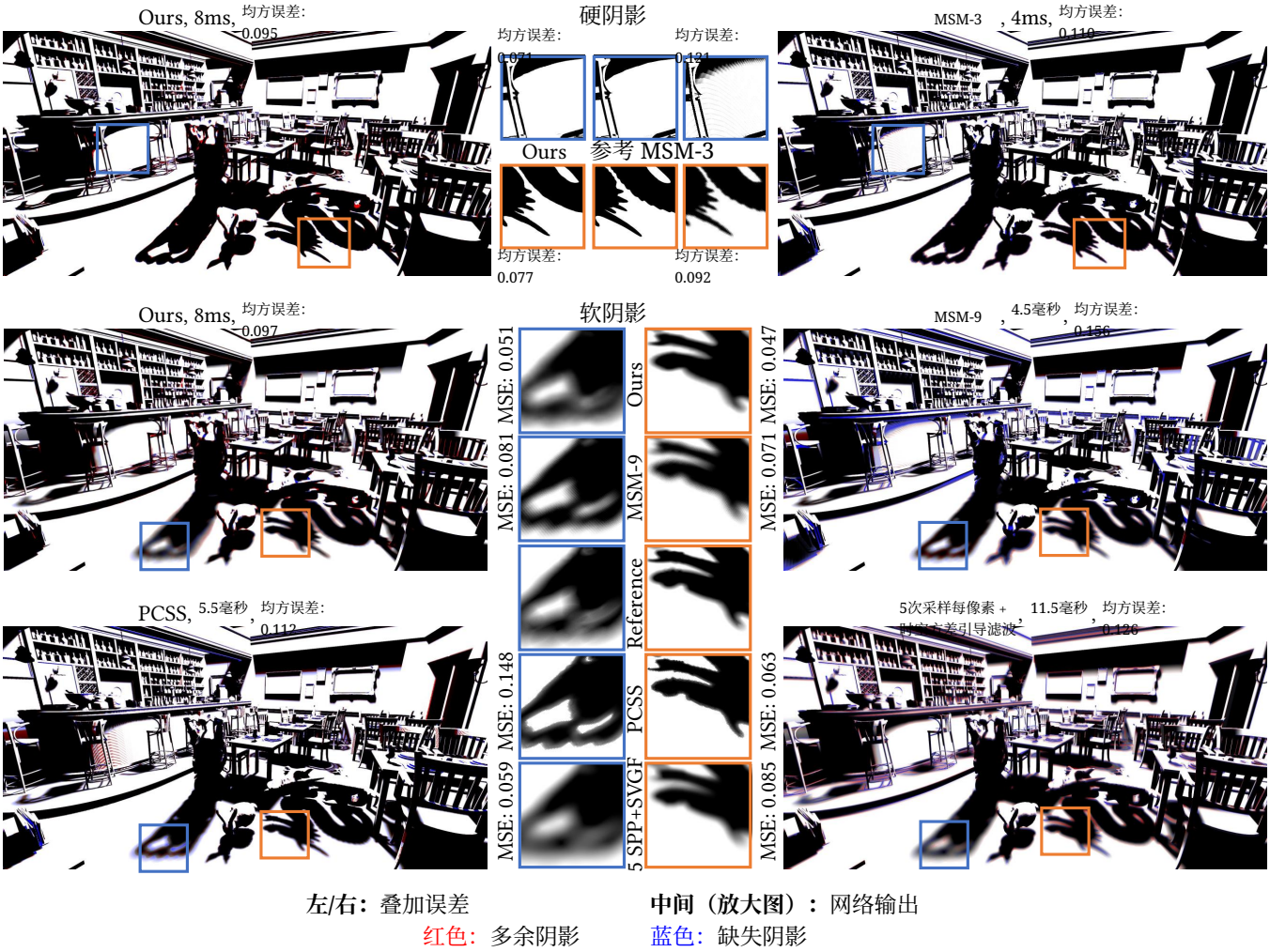

图9：比较我们的网络对未见过物体（佛像、兔子、龙）的泛化能力，以及其他竞争技术（矩量阴影贴图、百分比渐进软阴影、光线追踪和降噪）在硬阴影和软阴影方面的表现。MSM-3 和 MSM-9 是使用 3×3 和 9×9 预滤波核的矩量阴影贴图变体。

，供第二层使用。因此，我们重新排列输入，并将缓冲区维度从 $(h \times w \times ch)$ 更改为 $(h/2 \times w/2 \times 4ch)$。具体来说，我们不再向第一层输入完整分辨率（1024×2048）的 4 通道数据，而是直接向第二层输入四分之一分辨率（512 × 1024）的 16 通道数据。在解码器侧，我们执行相反的操作；将 4 个输出通道重新排列成一个 2x2 的像素方块。请注意，重新排列缓冲区不会为第二层增加任何额外的临时存储，同时还能完全移除第一层（二维卷积操作）。在解码器侧，为了改善训练收敛，我们使用双线性插值将第一个输出通道放大到完整分辨率。然后将其余三个通道加到插值后的输出中，以补足其余细节。优化后的网络性能为 5.8 毫秒。

3.6 网络深度优化

迄今为止，我们的优化具有通用性，适用于各种场景和发射器配置。接下来，我们将探讨针对特定场景的 优化，并调整网络架构以使其更加紧凑。较浅的网络有许多实际优势：参数数量呈指数级减少（减少到原来的二次幂倒数），训练速度更快，运行时推理也更快。我们不会依赖临时的架构调优，而是根据架构捕捉目标阴影效应的能力来选择架构。具体而言，我们将建立一个简单模型来估算场景配置下的最大半影尺寸，然后将该尺寸与适合重现这些效果的网络深度精确关联起来。我们通过实证验证该模型。

为了计算世界空间中的半影尺寸，我们的简化模型假设遮挡物为球形（或者保守地用包围球包住遮挡几何体）。生成训练样本时，

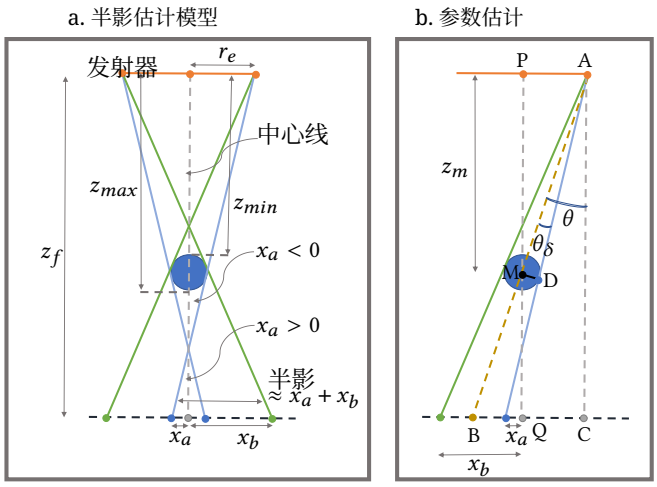

图10：用于估计给定像素处半影尺寸的简化模型。我们假设遮挡物呈球形，形成一个包围球，将遮挡几何体包围起来，如左侧图 (a) 所示。图 (b) 展示了一个用于估计参数的简化示意图θ，θ_δ。

我们还会测量遮挡物距离的最小值和最大值 z_{max}, z_{min}（图2，a）。随后，我们将像素处的半影宽度估算为内半影（x_a）和外半影（x_b）之和（图2，b），即 $\{x_a,\ x_b\} = z_f \tan(\theta \pm \theta_\delta) - r_e$。参数 θ，θ_δ 的推导见补充材料第 4 节。

计算完所有训练帧中每个像素在屏幕空间内的半影尺寸直方图后，我们选取最高的第95百分位数半影尺寸，作为神经网络架构感受野尺寸要求的保守上界。

我们可以调整每层卷积层的参数（卷积核尺寸、步幅）和池化操作参数，以满足目标感受野要求。如果将每个卷积层的空间分辨率减半，那么对于一个 l层网络，其有效感受野将以 $\times 2^l$ 增长。仅使用 3×3 卷积核时，我们可以求解 $l = \log_2(p_w/3)$，其中p_w 是屏幕空间半影宽度的保守估计。

表1为我们的技术提供了实证验证。我们使用相同的数据集训练了三个分别包含3、5和7层的网络。该数据集由包含不同半影尺寸的场景组成。半影尺寸估计值由我们的模型计算得出。在推理过程中，感受野小于预测值的网络

表1：不同半影尺寸下，3/5/7 层网络的均方误差。

预测半影宽度（像素）	网络层 → 感受野尺寸（像素）		
	3 → 24	5 → 96	7 → 384
21	0.005	0.005	0.012
90	0.083	0.009	0.018
180	0.161	0.042	0.019

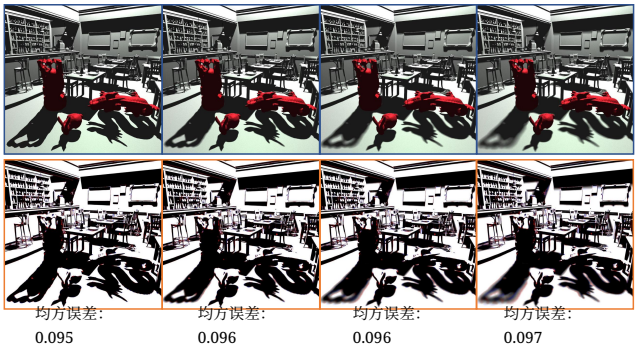

图 11：在经过训练的场景中对未训练物体（红色，第一行）的泛化能力。第二行以伪彩色显示相对于参考图像的误差。从左到右，光源尺寸从 0（硬阴影）到直径 50 厘米线性增大。

半影尺寸的表现较差，如红色所示。关于实证验证的更详细分析请参见补充材料第4.1节。

4 结果与比较

我们在多种场景中展示了本方法的效果。我们在运行时对训练集中的静态环境（例如房间）进行扩展，加入未训练物体，以展示其在游戏等交互式场景中的重要应用（图6）。我们在 Bistro 室内场景上训练单个网络，使其涵盖不同的光源尺寸、光源位置和相机轨迹，并引入（未训练物体）佛像、兔子和龙进行验证。

比较。 在软阴影场景中，我们的比较聚焦于两类基线方法：第一类是高性能的基于光栅化的近似方法，例如矩量阴影贴图（MSM） [Peters 和 Klein 2015] 以及百分比渐进软阴影（PCSS） [Fernando 2005]，这些方法与我们的工程目标（即完全基于光栅化，不使用光线追踪）和性能目标一致，因此作为主要基线；第二类是采用后处理降噪的交互式图形处理器光线追踪，作为更精确的“交互式”基线，即使用时空方差引导滤波（SVGF）的每像素采样数为 5 的光线追踪 [Schied 等人 2017]。请注意，与我们的方法不同，MSM 和 PCSS 均不支持通过光源尺寸显式控制半影样式；因此，我们调整 MSM 和 PCSS 的预滤波卷积核大小，使半影尺寸尽可能接近参考渲染图。MSM 和 PCSS 使用的卷积核大小分别为 3×3 和 9×9。我们的 PCSS 基线还包含深度感知后滤波。

我们的方法在保持竞争力的性能的同时，始终能提升阴影质量（图9）。请观看我们的视频，了解结果的时间稳定性。对于硬阴影，我们将其与 3×3矩量阴影贴图进行比较，获得了无混叠且没有漏光的阴影。更多结果和比较请参阅补充材料第 7 节。

运行时比较是在分辨率为 2k×1k 的条件下，使用 AMD 5600X 中央处理器和 Nvidia 2080Ti 图形处理器测得的。所有图中的运行时间（包括正文和补充材料）均不包含 G缓冲区生成；所有技术的该步骤都需要额外 2–3 毫秒（视场景而定）。每个场景均在 ≤ 400

分辨率为 2k×1k 的图像上，在集群中训练约 16 小时（75 个训练轮次）。

5 局限性

我们的技术与其他屏幕空间方法具有类似的局限性。无论是在相机空间还是在 [Ritschel 等人 2009] 和发射器空间 [Jesus Gumbau2018] 中，都无法获得分层深度信息，导致该问题存在适定性问题，进而产生近似误差。在相机空间中，缺少剥离深度数据使得确定像素之间的相互可见性变得复杂。同样，计算用于柔化阴影轮廓的模糊核大小，需要依赖着色点与遮挡物之间的距离，而在我们的设置中同样无法获得该信息。我们的网络和训练方法实际上旨在补偿这一适定性问题，通过利用数据中固有的复杂模式，弥合各种物体/场景布局之间的视觉差距。图12展示了一个典型的失败案例，我们的补充材料还包含更多讨论（第8节）。

6 结论

我们提出了一种紧凑、快速的神经方法及其训练损失函数，仅使用基础的阴影贴图光栅化输入，适用于合成时间连贯的硬阴影和软阴影。我们展示了，只要采用精心设计、针对具体问题的架构设计和一种新颖简洁的时间损失函数，单个小型网络便能学会根据不同光源尺寸及多种场景生成硬阴影和软阴影。该方法能够稳健地应对未见物体的加入，只需适度的训练预算，并且无需任何后处理降噪和／或时间抗锯齿。我们的方法能够生成稳定的硬阴影和软阴影，性能与常用的交互式近似方法相当，质量则优于（成本更高的）图形处理器光线追踪、降噪（和时间抗锯齿）方案。

基于光栅化的方法在生成软阴影和硬阴影时，依赖启发式方法和脆弱的手动调优，才能获得一致且视觉效果理想的结果。我们采用数据驱动方法，无需此类调优，只需付出适度的代价，就能提升阴影质量；无需光线追踪，即可生成可信且时间连贯的软阴影。我们的方法是一种紧凑的基于神经着色的框架 [Nalbach et al. 2016]，适用于低功耗分块渲染系统，并在复杂的设计空间中实现了颇有吸引力的权衡。我们展示了该方法的优势，这些优势在很大程度上抵消了额外的训练和集成复杂度。

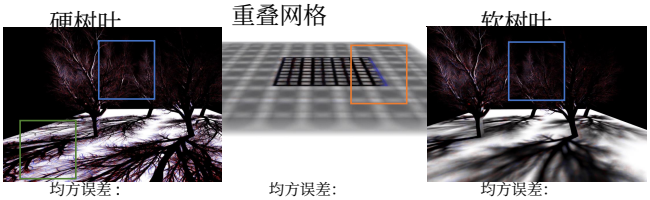

图12：我们的方法在高频率/高深度复杂度条件下的局限性。橙色、蓝色和绿色方框标出了由发射器和摄像机深度信息有限，以及训练场景欠采样（即细节丢失）导致的视觉伪影。

未来，若进一步积极优化神经网络架构，包括量化和程序化架构搜索，很可能还能提升推理性能。若结合稀疏化，例如采用基于彩票中奖券的方法 [，我们推测还可获得显著的额外性能提升，而且不会牺牲质量。

致谢

我们感谢审稿人提出的建设性意见，感谢 ORCA 提供 Amazon Lumberyard Bistro 模型 [Lumberyard 2017]，，感谢斯坦福计算机图形学实验室提供兔子、佛像和龙模型，感谢 Marko Dabrovic 提供 Sponza 模型，以及感谢 Morgan McGuire 提供 Bistro、Conference 和客厅模型 [McGuire 2017]。Sayantan 在 Meta Reality Labs Research 实习期间完成了这项工作。在麦吉尔大学就读期间，他还获得了来自魁北克自然与技术研究基金的博士奖学金。

参考文献

Pontus Andersson、Jim Nilsson、Tomas Akenine-Möller、Magnus Oskarsson、Kalle Åström 和 Mark D. Fairchild。2020。FLIP：交替图像差异评估器。美国计算机协会计算机图形学与交互技术论文集3, 2 2020 15 1 15 23 ( ), : – : . Thomas Annen 、 Tom Mertens 、 Philippe Bekaert 、 Hans-Peter Seidel 和 Jan Kautz。2007。卷积阴影贴图。载于 第*18*届欧洲计算机图形学渲染技术会议论文集（法国格勒诺布尔）（*EGSR'07*）。欧洲计算机图形学协会，德国戈斯拉尔，51–60。Thomas Annen、Tom Mertens、Hans P. Seidel、Eddy Flerackers 和 Jan Kautz。2008。指数阴影贴图。载于*2008*年图形界面会议论文集（加拿大安大略省温莎）（*GI '08*）。加拿大信息处理协会，加拿大，155–161。Chakravarty R. Alla Chaitanya、Anton S. Kaplanyan、Christoph Schied、Marco Salvi、Aaron Lefohn、Derek Nowrouzezahrai 和 Timo Aila。2017。使用循环去噪自编码器交互式重建蒙特卡洛图像序列。*ACM*图形学汇刊36, 4, 文章 98（2017年7月），12页。Blender 在线社区。2018。*Blender*——三维建模与渲染软件包。Blender 基金会，Blender 基金会，阿姆斯特丹。William Donnelly 和 Andrew Lauritzen。2006。方差阴影贴图。载于*2006*年交互式三维图形与游戏研讨会论文集（美国加利福尼亚州雷德伍德城）（*I3D '06*）。计算机协会，美国纽约州纽约，161–165。Hang Dou、Yajie Yan、Ethan Kerzner、Zeng Dai 和 Chris Wyman。2014。阴影贴图的自适应深度偏移。载于第*18*届*ACM SIGGRAPH*交互式三维图形与游戏研讨会论文集（美国加利福尼亚州旧金山）（*I3D '14*）。计算机协会，美国纽约州纽约，97–102。Andrew Edelsten、Paula Jukarainen 和 Anjul Patney。2019。真正的次世代：将深度学习引入游戏与图形学。NVIDIA赞助专场（游戏开发者大会）。Elmar Eisemann、Michael Schwarz、Ulf Assarsson 和 Michael Wimmer。2011。实时阴影（第1版）。A. K. Peters, Ltd.，美国。Wolfgang F Engel。2006。*ShaderX5*：高级渲染技术。C.R. Media。Randima Fernando。2005。百分比渐近软阴影。载于*ACM SIGGRAPH 2005*短篇论文（美国加利福尼亚州洛杉矶）（*SIGGRAPH '05*）。计算机协会，美国纽约州纽约，35–es。Jonathan Frankle 和 Michael Carbin。2018。彩票票据假说：训练剪枝神经网络。*CoRR* abs/1803.03635（2018）。arXiv:1803.03635 Daniel Holden、Bang C. Duong、Sayantan Datta 和 Derek Nowrouzezahrai。2019。子空间神经物理：快速的数据驱动交互式模拟。载于第*18*届*ACM SIGGRAPH/Eurographics*计算机动画年度研讨会论文集（美国加利福尼亚州洛杉矶）（*SCA '19*）。计算机协会，美国纽约州纽约，文章 6，共12页。Mateu Sbert Jesus Gumbau、Miguel Chover。2018。屏幕空间软阴影，*GPU Pro 360* 渲染指南（第1版）。A. K. Peters, Ltd.，美国。Simon Kallweit、Petrik Clarberg、Craig Kolb、Kai-Hwa Yao、Theresa Foley、Lifan Wu、Lucy Chen、Tomas Akenine-Moller、Chris Wyman、Cyril Crassin 和 Nir Benty。2021。Falcor 渲染框架。https://github.com/NVIDIAGameWorks/Falcor https://github.com/NVIDIAGameWorks/Falcor。Anton S. Kaplanyan、Anton Sochenov、Thomas Leimkühler、Mikhail Okunev、Todd Goodall 和 Gizem Rufo。2019。DeepFovea：利用自然视频的学习统计特征进行注视点渲染和视频压缩的神经重建。*ACM*图形学汇刊38, 6, 文章 212（2019年11月），13页。

Alexander Keller 和 Carsten Waechter。2009。实时精确光线追踪。美国专利US20070024615A1。Gary King。2004。阴影映射算法。Nvidia。354–355页。Jaakko Lehtinen、Jacob Munkberg、Jon Hasselgren、Samuli Laine、Tero Karras、Miika Aittala 和 Timo Aila。2018。Noise2Noise：无需干净数据的图像复原学习。*CoRR* abs/1803.04189（2018）。arXiv:1803.04189 Amazon Lumberyard。2017。Amazon Lumberyard Bistro，开放研究内容档案库（ORCA）。http://developer.nvidia.com/orca/amazon-lumberyard-bistro http://developer.nvidia.com/orca/amazon-lumberyard-bistro。Rafał K. Mantiuk、Gyorgy Denes、Alexandre Chapiro、Anton Kaplanyan、Gizem Rufo、Romain Bachy、Trisha Lian 和 Anjul Patney。2021。FovVideoVDP：宽视场视频的可见差异预测器。*ACM*图形学汇刊40, 4, 文章 49（2021年7月），19页。Viktor Heisenberg Marton Tamas。2018。实用屏幕空间软阴影，*GPU Pro 360*阴影指南（第1版）。A. K. Peters, Ltd.，美国。Morgan McGuire。2017。计算机图形学档案库。https://casual-effects.com/data https://casual-effects.com/data。Ben Mildenhall、Pratul P. Srinivasan、Matthew Tancik、Jonathan T. Barron、Ravi Ra-mamoorthi 和 Ren Ng。2020。神经辐射场：用于视图合成的场景神经辐射场表示。*CoRR* abs/2003.08934（2020）。arXiv:2003.08934 Mahdi MohammadBagher、Jan Kautz、Nicolas Holzschuch 和 Cyril Soler。2010。屏幕空间百分比渐近软阴影。载于*ACM SIGGRAPH 2010*海报（美国加利福尼亚州洛杉矶）（*SIGGRAPH '10*）。计算机协会，美国纽约州纽约，文章 133，共1页。https://doi.org/10.1145/1836845.1836987 Jacob Munkberg 和 Jon Hasselgren。2020。利用层嵌入进行神经去噪。《计算机图形学论坛》（2020）。Oliver Nalbach、Elena Arabadzhiyska、Dushyant Mehta、Hans-Peter Seidel 和 Tobias Ritschel。2016。深度着色：用于屏幕空间着色的卷积神经网络。*CoRR* abs/1603.06078（2016）。arXiv:1603.06078 Adam Paszke、Sam Gross、Soumith Chintala、Gregory Chanan、Edward Yang、Zachary DeVito、Zeming Lin、Alban Desmaison、Luca Antiga 和 Adam Lerer。2017。PyTorch中的自动微分。（2017）。Christoph Peters 和 Reinhard Klein。2015。矩阴影映射。载于第*19*届*I3D*研讨会论文集（美国加利福尼亚州旧金山）（*i3D '15*）。计算机协会，美国纽约州纽约，7–14。William T. Reeves、David H. Salesin 和 Robert L. Cook。1987。使用深度贴图渲染抗锯齿阴影。*SIGGRAPH*计算机图形学21, 4（1987年8月），283–291。Tobias Ritschel、Thorsten Grosch 和 Hans-Peter Seidel。2009。在图像空间中近似动态全局光照。载于*2009*年交互式三维图形与游戏研讨会论文集（美国马萨诸塞州波士顿）（*I3D '09*）。计算机协会，美国纽约州纽约，75–82。https://doi.org/10.1145/ 1507149.1507161 Olaf Ronneberger、Philipp Fischer 和 Thomas Brox。2015。U-Net：用于生物医学图像分割的卷积网络。*CoRR* abs/1505.04597（2015）。arXiv:1505.04597 Christoph Schied、Anton Kaplanyan、Chris Wyman、Anjul Patney、Chakravarty R. Alla Chaitanya、John Burgess、Shiqiu Liu、Carsten Dachsbacher、Aaron Lefohn 和 Marco Salvi。2017。时空方差引导滤波：路径追踪全局光照的实时重建。载于高性能图形学会议论文集（美国加利福尼亚州洛杉矶）（*HPG '17*）。计算机协会，美国纽约州纽约，文章 2，共12页。Karen Simonyan 和 Andrew Zisserman。2015。用于大规模图像识别的超深卷积网络。载于国际学习表征会议。Peter-Pike Sloan、Jan Kautz 和 John Snyder。2002。动态低频光照环境下用于实时渲染的预计算辐射传递。*ACM*图形学汇刊21, 3（2002年7月），527–536。Z. Wang、Eero Simoncelli 和 Alan Bovik。2003。用于图像质量评估的多尺度结构相似性。阿西洛马信号、系统与计算机会议记录2, 1398–1402第2卷。Lance Williams。1978。在曲面上投射曲线阴影。载于第*5*届计算机图形学与交互技术年会论文集（*SIGGRAPH '78*）。计算机协会，美国纽约州纽约，270–274。Lei Xiao、Salah Nouri、Matt Chapman、Alexander Fix、Douglas Lanman 和 Anton Kaplanyan。2020。用于实时渲染的神经超采样。*ACM*图形学汇刊39, 4, 文章 142（2020年7月），12页。Lei Yang、Shiqiu Liu 和 Marco Salvi。2020。时间抗锯齿技术综述。《计算机图形学论坛》39, 2（2020），607–621。arXiv:https://onlinelibrary.wiley.com/olddoi/pdf/10.1111/cgf.14018 Fan Zhang、Hanqiu Sun、Leilei Xu 和 Lee Kit Lun。2006。面向大规模虚拟环境的并行分割阴影贴图。载于*2006*年*ACM*虚拟现实连续体及其应用国际会议论文集（中国香港）（*VRCIA '06*）。计算机协会，美国纽约州纽约，311–318。Zhongxiang Zheng 和 Suguru Saito。2011。屏幕空间各向异性模糊软阴影。载于 *ACM SIGGRAPH 2011*海报（加拿大不列颠哥伦比亚省温哥华）（*SIGGRAPH '11*）。计算机协会，美国纽约州纽约，文章 75，共1页。https://doi.org/10.1145/2037715.2037799

神经阴影映射——补充材料

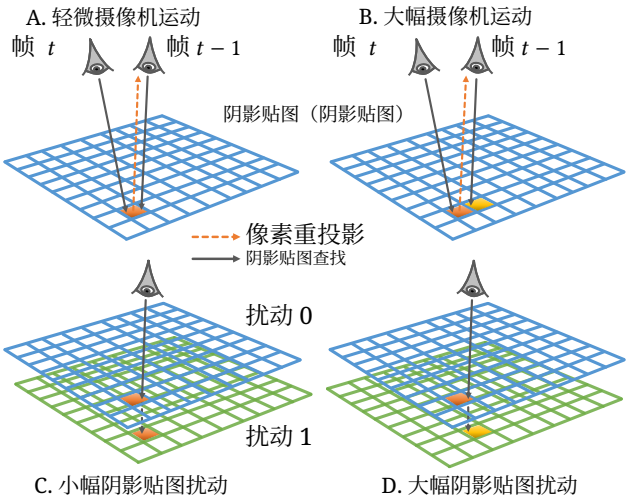

图1：比较运动向量和扰动损失对阴影贴图纹理查找的影响。A 和 C 表明，微小的摄像机运动在时间上等同于阴影贴图在空间中的微小扰动。同样，B 和 D 表明，大幅摄像机运动等同于阴影贴图在空间中的大幅扰动。

1 特征工程

1.0.1 过拟合与欠拟合。 神经网络以能够拟合数据集而闻名，即使特征与目标之间不存在内在关系——这种现象称为过拟合。一个极端的例子是将光线追踪得到的目标映射到作为特征的白噪声纹理。在这种情况下，神经网络就像一个哈希映射，将输入映射到输出。这或许适用于非常小的数据集，但显然无法适用于更大的数据集。实际上，有些特征提供关键信息，有些则没有。因此，为了将噪声与信息区分开来，我们需要一个多样化的数据集。就我们而言，可以通过收集不同发射器位置、相机位置和场景下的数据来增加多样性。事实上，随着训练数据不断增加，我们会进入欠拟合状态，此时网络缺乏足够的容量来学习数据集中的细节。

1.0.2 特征选择。如正文所述，我们使用敏感度作为指标，从集合 $U = \{d, \mathbf{n}\ z,\ \mathbf{ne}\ z_f,\ c_e,\ c_c\} + \{z - z_f, z/^z f,\ c_c/d,\ \mathbf{n}\cdot\mathbf{ne}\}$中选择并剪枝特征。然而，仅依据敏感度进行剪枝并不能得出唯一的缓冲区组合。有几种组合的敏感度相近。在这种情况下，我们使用验证误差作为平局判定规则。例如，如表1所示，缓冲区组合 $\{z/^z f,\ z - z_f,\ c_e,\ c_c/d\}$ 和 $\{z/^z f,\ z - z_f,\ \mathbf{n}\cdot\mathbf{n}\ d\}$ 的敏感度相近，但其中一种组合的验证误差更低。

表1：各种缓冲区组合的验证误差。每一行中，我们添加2个新特征，并移除敏感度低于1.5%的特征。 +/− 表示集合的加法和减法。

剪枝前			剪枝后		
Fea. 计数	特征描述	Err 10^{-3}	Fea. 计数	特征描述	Err 10^{-3}
2	$\{z/^z f,\ c_e\}$	6.86	无变化		
4	$+\{z - z_f, c_c/d\}$	6.65	无变化		
6	$+\{\mathbf{n}\cdot\mathbf{ne}_d\}$	6.61	4	$-\{\mathbf{n}\cdot\mathbf{ne}\ c_c/d\}$	6.69

2 时间损失

本节将比较我们的扰动损失与基于运动向量的损失函数 [Xiao 等人 2020] 提出的方法之间的相似之处和差异。运动向量通过将像素的世界空间位置投影到前一帧，找到该像素在前一帧中的位置。因此，基于运动向量的损失会计算网络输出中经运动向量补偿的时间像素差异，并将其作为误差反向传播通过网络以进行学习。请注意，基于运动向量的损失在训练期间**仅**使用历史数据。

我们认为，我们的扰动损失函数与基于运动向量的损失函数具有相似的最终效果，同时能更灵活地控制时间稳定性，并简化数据收集。对于运动向量损失，由于像素具有离散性，重投影到前一帧的像素可能对应于世界空间中略有不同的位置。假设发射器固定，像素的世界空间值随时间发生微小变化，就可能导致阴影贴图纹理查找结果不同，如图 1(b) 所示。这相当于在保持相机位置固定的同时扰动发射器（我们采用的方法），如图 (d) 所示。然而，对于运动向量损失，如果帧率过高，像素在世界空间中的差异可能很小。该像素及其重投影位置可能对应同一个阴影贴图纹素，如图 (a) 所示。阴影贴图纹素较大时，情况也一样。在这两种情况下，时间误差都为零，网络无法学习。相反，如果帧率过低，我们可能无法找到有效的重投影，此时必须强制将误差

表2：处理 1024x2048 分辨率输入的 5 层网络的逐层计算、可学习参数以及临时存储读写访问。

#	运行时（毫秒）/ 计算 分辨率	（十亿次浮点运算）	# 参数 ($\times 10^3$)	临时存储 IO （百万像素）
0	11.0 毫秒 （1Kx2K）	7.15	3.4	278.9
1	3.67毫秒 （512x1K）	8.05	15.4	117.4
2	1.63毫秒 （256x512）	8.05	61.4	58.7
3	0.79毫秒 (128x256)	8.05	245.8	29.3
4	0.17毫秒 (64x128)	2.68	327.68	6.29

设为零。因此，运动向量损失需要仔细平衡帧率。然而，在我们的方法中，可以直接调整发射器扰动，使差异足够大，从而产生非零的反向传播，如图 (d) 所示。具体做法是将扰动幅度设为与发射器到场景的距离及发射器尺寸成正比。此外，运动向量损失依赖于由关键帧图像构成的长序列，这会增加数据采集流程的复杂性。我们的方法不依赖时间信息，因此可以按照任意所需帧率，使用任意相机、发射器和物体轨迹对场景进行采样。

3 性能优化

我们将讨论 UNet 架构的一个有趣之处，这一点在正文中没有提及。

从表2可以看出，我们网络的运行时性能与临时存储量成正比，而不是与计算操作的数量成正比。请注意，前 4 层的计算浮点运算量几乎保持不变，但随着层级递增，运行时却在下降。这表明性能受内存带宽限制。所需的内存带宽与临时缓冲区的大小以及缓冲区的缓存效果成正比。临时缓冲区较小时，缓存效果最显著。事实上，由于第 0 层的缓存未命中率较高，与第 1 层相比，其运行时增长超过线性。我们使用性能分析器验证了这一点。表2给出了网络运行时按层的详细分解（未优化）。

通过对网络权重进行剪枝 [Frankle 和 Carbin 2018]，或许可以进一步提升性能。我们的网络性能在很大程度上取决于临时缓冲区的大小，而不是缓冲区的连接方式。因此，单纯剪枝权重可能无法提升性能，除非我们同时减小临时存储的大小。另一个值得探索的方向是将临时存储的精度降低至 8 位，因为网络输出始终介于 0 和 1 之间。我们将这些优化留待未来工作中进一步探索。

表 3：处理 1024x2048 分辨率输入的 5 层网络的实测逐层计算时间。

层/ 分辨率	编码器 操作/时间（毫秒）	解码器 操作/时间（毫秒）	总计 (ms)
0 (1Kx2K)	二维卷积 + 下采样 (4.27 + 0.72)	上采样 + 跳跃连接 +二维卷积 (0.24 + 0.37 + 5.37)	11.0
1 (512x1K)	二维卷积 + 下采样 (1.38 + 0.63)	上采样 + 跳跃连接 +二维卷积 (0.12 + 0.19 + 1.36)	3.67
2 (256x512)	二维卷积 + 下采样 (0.58 + 0.33)	上采样 + 跳跃连接 +二维卷积 (0.07 + 0.10 + 0.55)	1.63
3 (128x256)	二维卷积 + 下采样 (0.29 + 0.10)	上采样 + 跳跃连接 +二维卷积 (0.04 + 0.05 + 0.31)	0.79
4 (64x128)	二维卷积 (0.17)		0.17

4 网络深度优化

在本节中，我们首先使用正文中所述的简化模型，推导半影宽度的数学公式。我们按如下方式估计参数 θ和 θ_δ：

$$z_m = PM = \frac{z_{max} + z_{min}}{2} \tag{1}$$

$$r_s = MD = \frac{z_{max} - z_{min}}{2} \tag{2}$$

来自 $\triangle AMP$,

$$AM = \sqrt{z_m^2 + r_e^2} \tag{3}$$

来自 $\triangle AMD$,

$$\theta_\delta = sin^{-1}\frac{MD}{AM} = sin^{-1}\frac{r_s}{\sqrt{z_m^2 + r_e^2}} \tag{4}$$

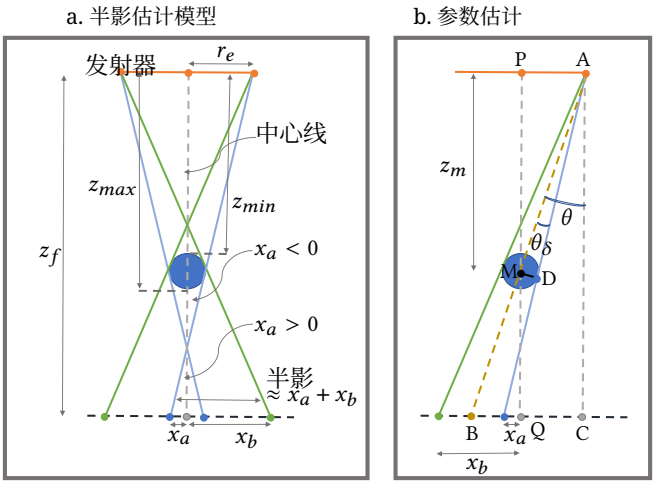

图2：用于估计给定像素处半影尺寸的简化模型。我们假设遮挡物为球形，并在遮挡几何体周围形成一个凸包围球，如左侧图 (a) 所示。图 (b) 展示了用于估计参数 θ，θ_δ 的简化示意图。

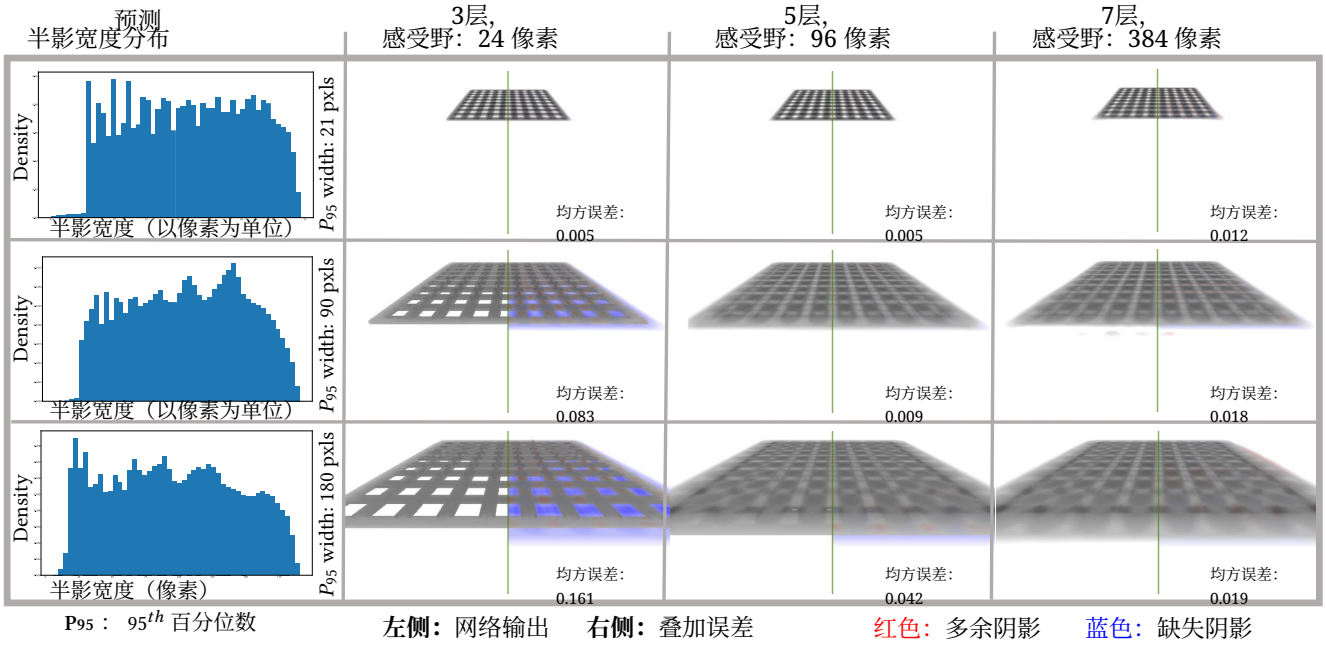

图3：对我们的半影宽度预测模型进行实证验证。我们预测纵轴上的半影宽度，并改变横轴上的层数。对角线表示最佳层数，而对角线下方的元素显示出较大的误差，因为感受野不足以容纳半影宽度。

来自 $\triangle AMP$ 和 $\triangle BMQ$,

$$BQ = \frac{AP \cdot MQ}{MP} = \frac{r_e(z_f - z_m)}{z_m} \tag{5}$$

根据 $\triangle ABC$,

$$\theta = tan^{-1}\frac{BQ + QC}{AC} = tan^{-1}\frac{BQ + r_e}{z_f} \tag{6}$$

因此,

$$x_a = z_f\, tan(\theta - \theta_\delta) - r_e \tag{7}$$

$$x_b = z_f\, tan(\theta + \theta_\delta) - r_e. \tag{8}$$

4.1 实证验证

表4：图3中 3、5 和 7 层网络的计算成本及可学习参数数量。

网络	每像素浮点运算次数	# 可学习参数
3 层	8528	39.25K
5 层	16208	653.7K
7 层	21968	9.501M

在图3中，我们将一个网格物体置于地平面和发射器之间，从而产生不同尺寸的半影。对于图中的每一行，我们预测半影尺寸的分布，并从中找出最大宽度（第95百分位数）。沿各列方向，我们通过调整网络的层数来改变感受野。汇总所有组合的误差后，我们发现对角线下方的元素误差较高，因为网络的感受野不足以容纳半影尺寸。请注意，图3中的网络每像素计算 FLOPs 并不相同，如表4所示。

仅增加网络的层数也会增加计算成本。因此，误差的任何降低都可能归因于计算需求的增加。我们验证了误差降低确实是由于网络感受野增大，而不一定是由于计算量增加。因此，我们对 3 层、5 层和 7 层网络的浮点运算量进行约束，如表 5 所示。从图4可以看出，尽管使用的网络计算量几乎相同，误差仍沿对角线降低。

表 5：图4所示的 3、5 和 7 层网络的计算成本与可学习参数数量。

网络	每像素浮点运算次数	# 可学习参数
3 层	14928	110.9K
5 层	16208	653.7K
7 层	15328	1.626 M

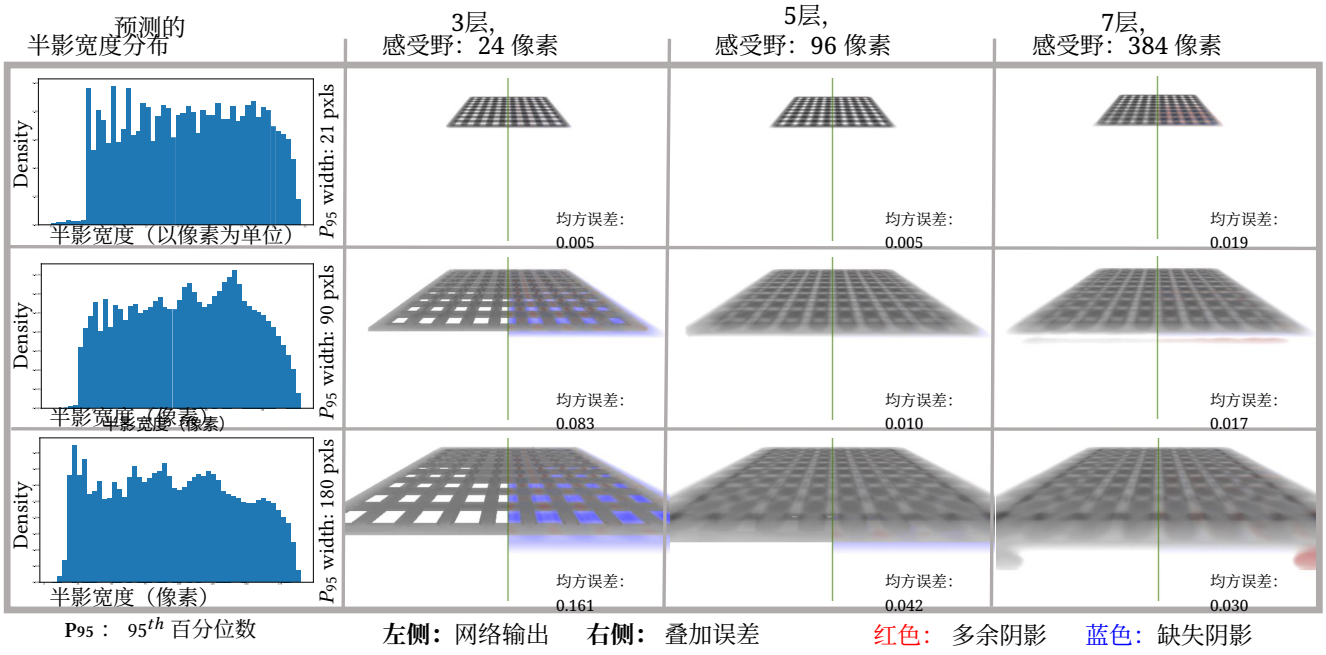

图4：我们半影宽度预测模型的实证验证。我们在纵轴上预测半影宽度，并在保持各网络计算浮点运算量不变的同时，改变横轴上的层数。受浮点运算量约束的网络表现出与图3中未受约束的网络相似的行为。

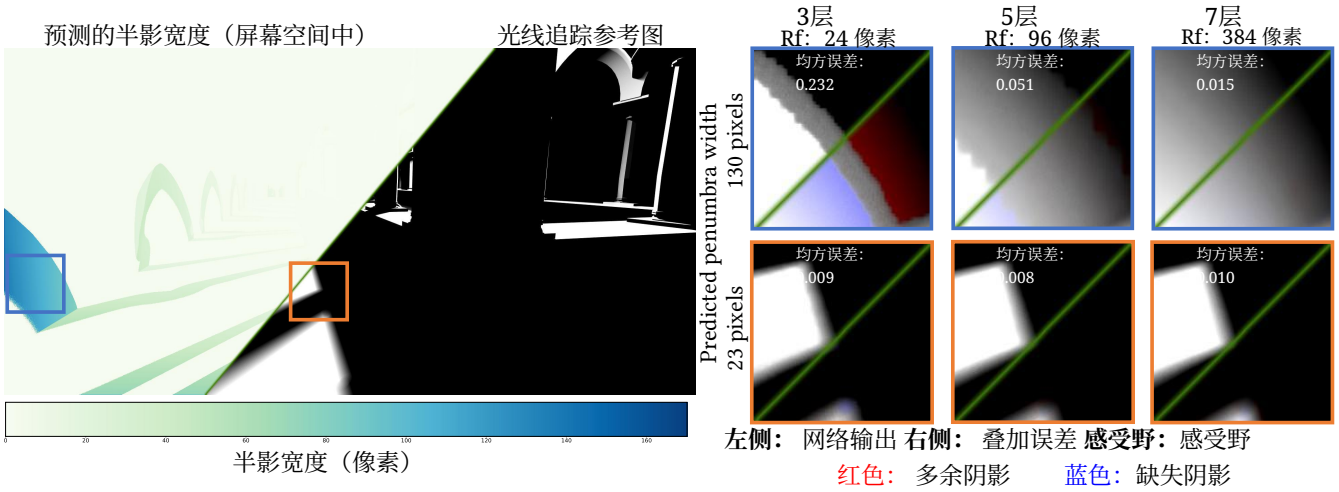

图5：展示了我们在 Sponza 场景中的半影宽度预测模型准确度。模型预测的高半影宽度区域（蓝色剪影）只有通过 7 层网络才能准确重现。

在图5中，我们从 Sponza 场景中选取单帧，分析不同区域内半影宽度预测的有效性。对于半影较大的区域（黄色框选区域），随着网络层数增加，网络输出的准确性也随之提高，这有力地验证了我们的模型。

5 数据生成、训练与推理

协调场景创作、数据收集、训练、推理并生成最终结果，是本项目面临的关键挑战之一。我们使用 Blender [Community 2018]，创作场景，确保所有关键帧相机轨迹都不会超出

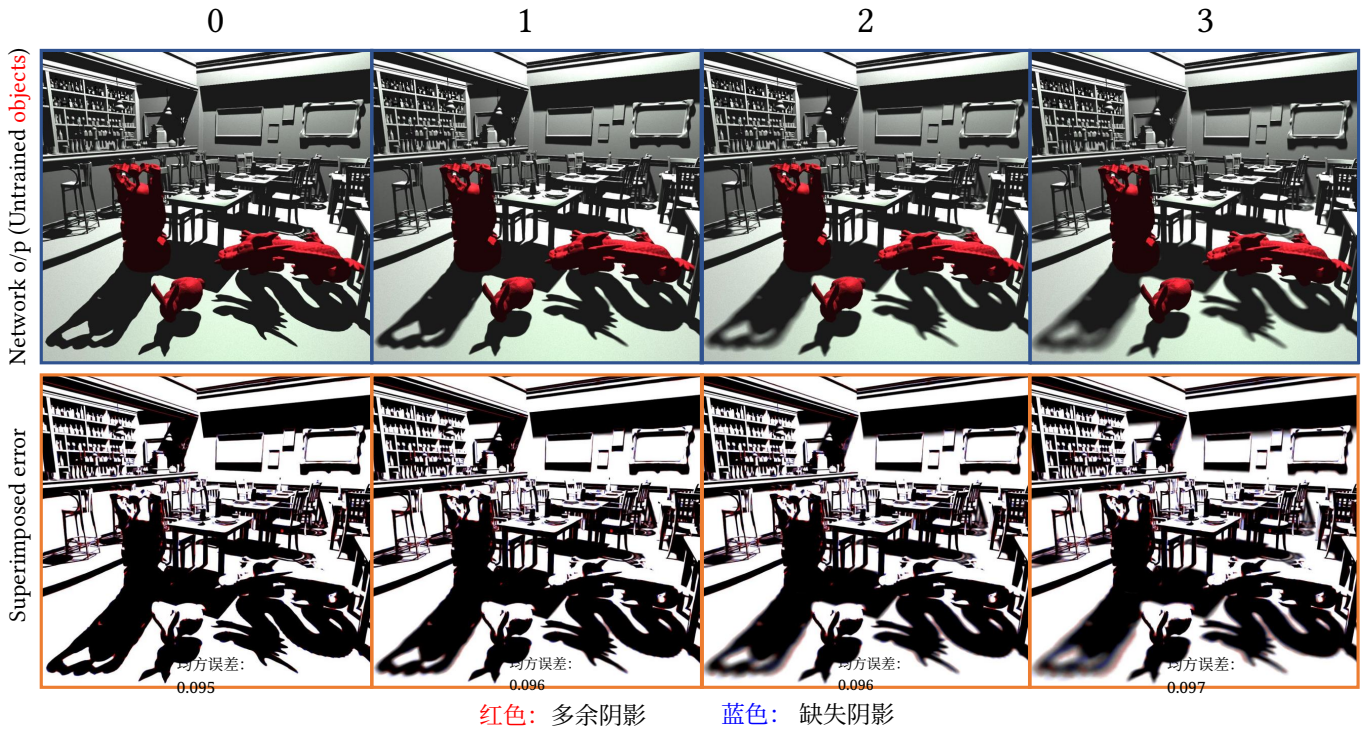

图6：展示我们的方法对第一行中标记为 红色 的未训练物体的有效性。第二行展示了相对于参考图像的比较结果，误差叠加在原始网络输出之上。从左到右，硬阴影的索引为 0，而 3 表示直径为 50 厘米的发射器。第一行中采用了后处理着色。

光源视锥体。我们为相机和发射器设置多个关键帧轨迹，以涵盖不同场景，并将其烘焙到场景中。随后，我们将场景上传到 Linux集群以生成训练数据。我们使用基于 Vulkan（最初基于 DX12）的 Falcor [Kallweit et al. 2021] 修改版本，并配合 Python 脚本收集、处理和整理训练数据。训练时，我们使用 Pytorch [Paszke et al. 2017] 脚本，每个训练轮次都会以随机顺序遍历数据。训练过程中，我们根据测试误差保存 $N(=3)$ 个最佳模型。测试由少量（10-15 张）人工挑选的图像组成。每次训练迭代时，我们以 0.01 的概率运行测试，因此大约每 100 次训练迭代运行一次测试。训练完成后，我们通过比较完整训练数据集上的误差，从 N 个模型中选出最佳模型。训练期间，我们还会收集多项统计数据以供分析。推理时，我们让模型处理新的轨迹；这些轨迹可能与训练数据集中的轨迹有所重叠，但并不相同。我们使用 Falcor 和基于 Cudnn 的解决方案，在本地机器（AMD 5600X、Nvidia 2080Ti）上测试网络性能。

训练数据通过并行执行阴影映射和光线追踪渲染通道生成。阴影映射通道通过抖动发射器和相机位置，输出缓冲区的三种扰动。光线追踪通道每个像素使用 1500 条光线，并将其分布在 8 个或更多累积通道中，从而实现 8 倍或更高的多重采样抗锯齿。光线追踪通道还会同时输出柔和度各不相同的 4 种阴影。总而言之，每帧由 3 种输入缓冲区扰动，以及 4 幅柔和度逐渐增加的抗锯齿光线追踪图像组成。

对生成数据进行视觉检查至关重要。我们将未经处理的光栅化阴影与光线追踪阴影进行比较。我们希望光栅化输出与光线追踪输出相匹配，并且二者应具有最小且一致的偏差（如果存在偏差）。例如，由于用于防止自相交的偏移，光栅化与光线追踪得到的阴影位置可能存在差异。我们发现，将场景缩放到相同尺寸有助于最小化光栅化与光线追踪之间的偏差，并生成一致的数据。偏差还取决于我们生成光线的方式——从发射器到场景，或反之。具体差异可能取决于背面剔除的配置方式。我们更倾向于采用从发射器到场景的光线追踪，因为这种设置更接近阴影映射。

6 网络架构消融研究

我们讨论了所实现的其他各种网络架构的局限性。第一种变体受到指数阴影贴图 [Annen 等人 2008], 的启发，我们在深度测试近似中学习参数 α $e^{\alpha(z-z_f)}$。为了去除网络最终层中的非线性，我们在对数空间中执行 Adam-SGD 优化，并在推理期间对输出取指数。然而，将未经处理的输入 z 直接注入输出会导致严重的阴影走样，而我们的损失函数（VGG-19）无法有效地

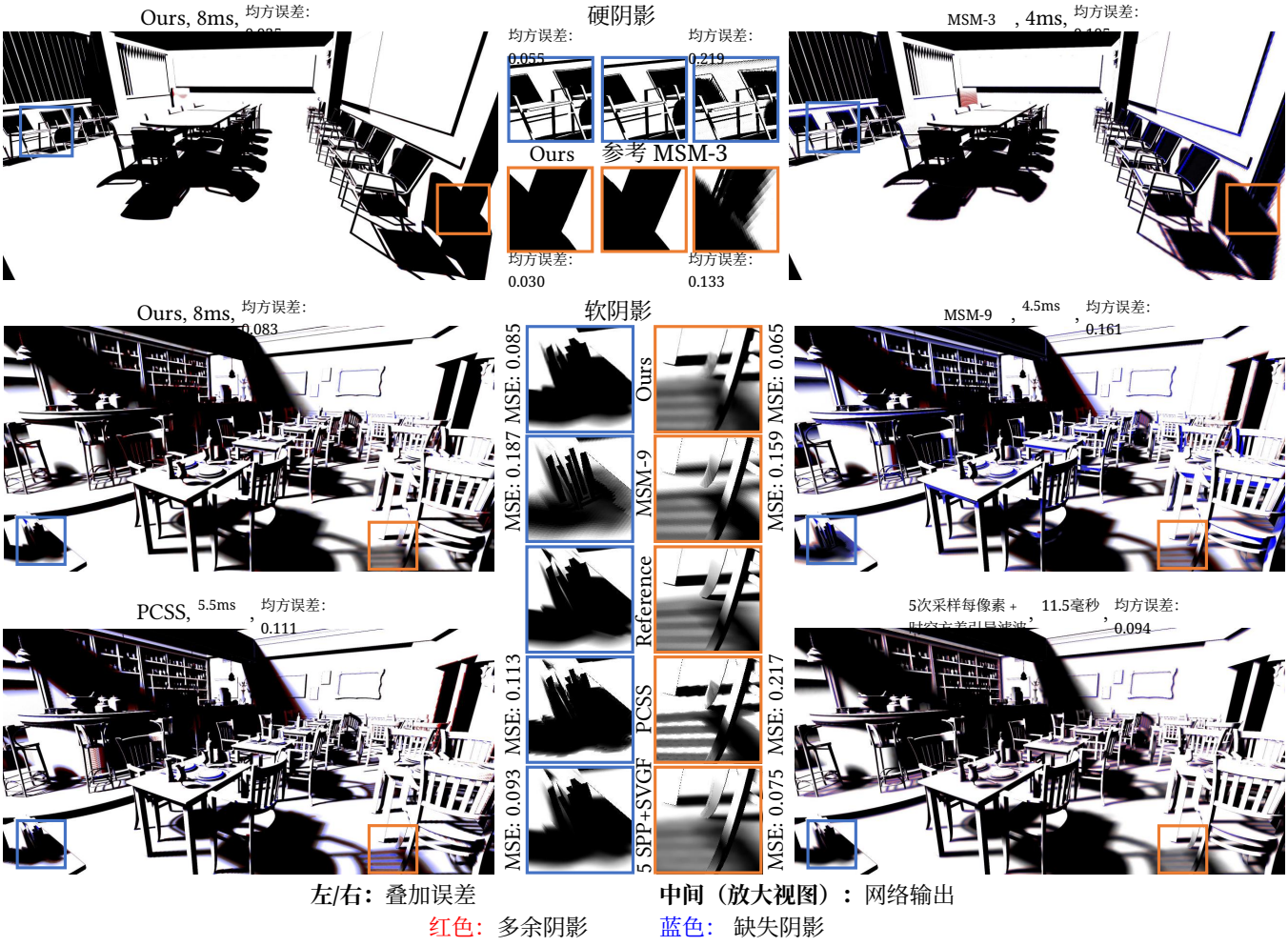

图 7：与其他竞争技术（MSM、百分比渐进软阴影、光线追踪+降噪）的输出相比，我们的网络能够泛化到未见过的轨迹（在同一场景内），适用于硬阴影和软阴影。MSM-3 和 MSM-9 是矩量阴影贴图的变体，分别使用 3×3 和 9×9 预滤波核。会议室模型 ©Anat Grynberg 和 Greg Ward。

纠正对数空间中的混叠。在另一种变体中，我们尝试使用单独的网络对光栅化（来自发射器）深度进行降噪，并以光线追踪深度作为目标。我们将网络分成两半：前半部分用于对深度进行降噪，后半部分则将前半部分的输出处理为最终阴影。我们对这两个网络进行了端到端训练。然而，与基础网络相比，该架构未能生成高质量的最终输出。我们还测试了同一思路的其他变体，但效果同样不佳。

7 结果与比较

图 7 比较了我们的方法与其他竞争方法在训练中未出现的轨迹上的表现。图8展示了我们的方法在未经训练的相机和发射器轨迹上的应用。对于每个场景，我们都使用不同的发射器和相机配置，并在一定的柔和度范围内训练一个独立的网络。因此，单个网络就能生成硬阴影和软阴影，并通过标量输入控制阴影的柔和度。图 6 表明，我们的网络能够泛化到训练集中未出现的各种形状。

8 局限性与未来工作

我们这项技术目前的局限之一是无法自然地扩展到多个光源。解决这一问题是未来工作中一个令人振奋的方向。另一项局限是，当场景混合中的发射器深度分布差异很大时，技术的泛化能力较弱。这主要是因为我们的网络规模（有意为之）较小，拟合此类多模态数据的能力有限。为避免这一问题，可以针对不同的深度变化分别训练多个网络，并在运行时根据发射器距离在这些网络之间切换。半影尺寸超出神经网络的感受野和训练图块尺寸时，也会产生伪影；不过，我们对半影尺寸的保守估计，使我们能够相应地调整架构的感受野。最后，我们的扰动

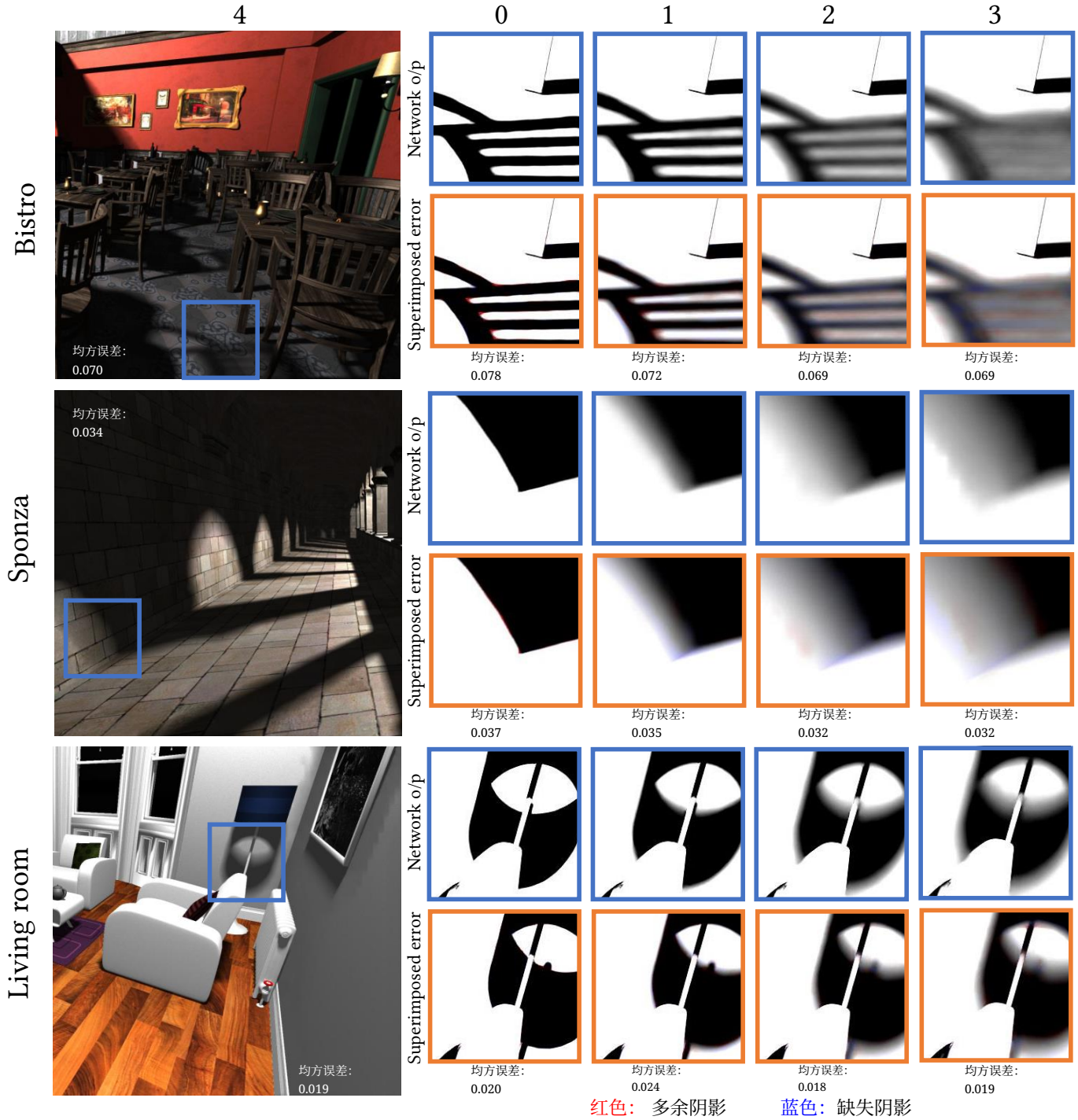

图8：该图展示了我们的网络输出（经阴影后处理），以及局部裁剪图（未经阴影处理）中不同半影尺寸的效果。每个场景均独立训练，并在验证集上测试。每个场景的第二行展示了相对于参考图像的误差（叠加在网络输出上）。硬阴影的索引为 0，而 4 表示直径为 50 厘米的发射器。

损失函数有时会使细小的几何细节过度模糊，例如电线等细小特征；不过，时间稳定性的提升十分显著，适当的微调有助于减轻这种过度模糊。

未来工作中一个令人振奋的方向，是将我们的技术扩展到有限数量的光源。我们可以从图着色方法中汲取灵感 [Marton Tamas 2018]，其中

不重叠的发射器视锥被归为若干层，并对每层分别进行滤波。神经方法可以采用一个网络，以来自 n 光源的缓冲区作为输入，并输出结果。在训练过程中，我们可以禁用（即将输入设为零）一部分光源p ($< n$)，使网络能够更有效地学习每个光源的响应，也就是说，不受*between* 光源间相互影响的干扰。

最后，网络的运行时优化也是一个值得探索的方向。与计算操作相比，UNet 的性能主要取决于内存带宽。通过减少临时缓冲区的数量来提高计算密度的定制剪枝技术，是一个有趣的研究方向。此外，网络输出为非高动态范围，因此可以利用这一点来减小缓冲区的大小。