了解如何剪枝 用于多视图神经图像合成的像素

Marta M ilovanovi´c⋆ 恩佐·塔塔利奥内⋆ 马尔科·卡尼亚佐⋆‡ *F´*埃利克斯·亨*p y*§

⋆ LTCI，T´el´ecom Paris，巴黎综合理工学院，法国
‡ DEI，帕多瓦大学，意大利§
Orange Labs，法国

arXiv:2305.03572v1 [cs.MM] 5 May 2023

摘要

基于图像的渲染技术是为用户带来沉浸式体验的核心，它能够根据多张输入图像生成新视图。由于这类技术在客观和主观质量方面均表现良好，研究界投入了大量精力加以改进。然而，在接收端进行渲染所需的大量数据，阻碍了这类技术在带宽受限环境中的应用，也限制了其在实时应用中的使用。我们提出了 LeHoPP，一种输入像素剪枝方法，通过考察每个输入像素对于渲染视图的重要性，避免使用无关像素。即使不重新训练基于图像的渲染网络，我们的方法也能在合成质量与像素率之间取得良好平衡。在通用神经渲染框架中进行测试时，与其他剪枝基线方法相比，LeHoPP 的平均表现提升了 0.9 dB 至 3.6 dB。

索引词— 沉浸式视频、视频处理、学习式基于图像的渲染、像素剪枝。

1. 引言

沉浸式视频技术使观看者能够感知场景，仿佛置身于视频之中。然而，这些技术仍给行业带来了挑战：为了向观看者提供新视点，需要传输和处理大量数据。这带来了诸多限制，包括功耗、传输带宽和硬件能力。

最近，神经基于图像的渲染 [1, 2]领域取得了巨大进展。这些方法旨在根据一组稀疏的输入视图渲染复杂场景的新视图，并且可以采用不同的 3D 场景表示格式。神经辐射场（NeRF）技术 [3]改变了神经基于图像的渲染领域。通过使用多层感知机（MLP）和位置编码，它能够以出色的精度表示连续的 5D 辐射场。模型能够在渲染新视图时处理非朗伯效应，得益于其视图依赖特性。然而，基于 NeRF 的模型会将场景编码到自身的权重中，并且重—需要经过漫长的优化过程才能渲染陌生场景，因此并不实用。

近年来开发的一种方法称为 IBR-Net [4], ，是一种先进且稳健的神经辐射场方法。得益于其优先优化合成质量的端到端设计，该方法能够很好地泛化到新场景。此外，与 [2], 等最新的单次合成方法相比，它取得了更好的性能，同时相较于单场景推理方法，其合成结果仍具有良好的质量。此外，当前的技术进步使我们能够在客户端部署神经（基于图像的）渲染器，从而生成高质量的合成结果，但这需要给定场景中有许多可用的源视图。这对其利用现有标准进行部署构成了重大障碍（例如 MPEG沉浸式视频 - MIV [5], ，一种用于体积视频传输的标准）。

本文针对沉浸式视频处理和潜在数据传输的应用场景，利用 IBRNet 的泛化能力，同时考虑降低像素率的需求。像素率定义为符合规范的解码器每秒应能够处理的亮度像素数量。像素率的限制在 MIV通用测试条件 (CTC) [6] 中定义。因此，我们提出了一种方法，用于区分重要像素和可剪除的像素。对于每个输入视图，我们的方法都会生成相应的剪枝掩码，并移除非必要像素。剪枝掩码的计算由损失函数引导，该损失函数是针对使用 IBRNet 渲染的目标视图、相对于输入视图计算得出的。最后，在最终阶段，使用剪枝后的源视图渲染目标视图。

像素率是多视点视频传输系统可行性的一项重要约束，因此，我们的工作是完整沉浸式编码流程中的一个重要起步阶段。据我们所知，这是首篇研究像素剪枝对多视点基于纹理的神经渲染器影响的论文，更具体地说，是研究其对 IBRNet 的影响。因此，我们还研究了采用随机剪枝的基线方法，并对它们进行了比较。我们观察到，与随机剪枝像素相比，我们的方法性能更好；即使不针对每个视频序列重新训练或进行额外微调，也能在渲染视图质量和像素率之间实现总体上良好的折衷。

以针对 IBRNet 的每个视频序列进行微调。[1] 本文其余部分安排如下：第2节概述 IBRNet 架构、体积视频标准及相关像素剪枝技术。第3节介绍所提出的方法。第4节介绍实验条件、结果和分析，第5节总结全文。

2. 背景

2.1. IBRNet

IBRNet [4] 是一种新技术，它结合了传统的基于图像的渲染方法和 NeRF 方法 [3], ，能够生成连续的场景辐射场。与以往的基于图像的渲染方法相比，由于其采用端到端的设计方式，以优化合成质量，因此在渲染具有挑战性的场景时表现更佳。此外，与 NeRF 不同，它能够泛化到任意新场景，这对于传输场景中的沉浸式视频渲染器而言，是一种理想的特性。IBRNet 首先确定目标视图的一组相邻源视图，并提取这些视图的图像特征。然后，对于目标视图中的每条相机射线，它会沿投射的射线计算一组采样点的颜色和密度。最后，对该射线上的颜色和密度进行累积，以渲染最终颜色。在训练过程中，通过最小化真实像素颜色与渲染像素颜色之间的均方误差来优化模型。

IBRNet架构如下。首先，将从所有源视图中提取的图像特征输入多层感知机架构，以汇总局部和全局信息，并获得具备多视图感知能力的特征。随后，使用射线变换器模块汇总沿投射射线计算的所有样本的信息，从而改善几何推理和密度预测。

2.2. 体积视频标准

消费者对沉浸式技术的兴趣日益增长，推动了MPEG标准化社区的相关工作 [7]。近期，MPEG发布了一项新规范，旨在促进数字沉浸式媒体的存储和传输，使其能够以六自由度播放，该规范名为MPEG沉浸式视频（MIV） [5, 8]。MIV主配置支持多视图加深度输入格式，其中每个（纹理）视点都对应一张深度图，用于表示场景中物体与摄像机之间的距离。MIV参考软件（TMIV） [9] 由规范性解码过程（基于MIV元数据）和非规范性渲染过程组成，后者使用基于深度图像的渲染（DIBR）方法生成目标视图 [10]。

MIV 旨在利用已经开发的端到端视频编码系统；然而，由于深度数据的特性，对深度图进行编码并使用传统视频编解码器并非易事。由于深度会增加比特率和像素率方面的负担，人们已开发出许多方法来减轻这一开销，其中之一便是解码器端深度估计。这促成了 MIV 的一种新配置文件，称为几何缺失（GA）配置文件 [11], ，该配置文件仅传输码流中的纹理信息。不过，在使用 DIBR 方法渲染目标视图之前，该方法仍会使用深度估计工具计算深度图。

2.3. 像素剪枝

输入像素剪枝已率先应用于计算机视觉中可扩展的自监督学习：在输入图像中随机遮盖图像块，随后重建缺失的像素 [12]。此外，近期有一种方法提出在线选择上下文点，以便在二维视频和其他数据模态上高效地进行元学习 [13]。然而，这些方法假设上下文（输入图像）与目标相同，而场景渲染中的输入则与目标不同。

关于多视角视频的像素剪枝，TMIV 提供了一种基于深度的解决方案。TMIV [9] 中的剪枝是由基于硬件限制而设定的像素率 [14], 硬性上限所驱动的，其方法是从 TMIV 的输入视图中移除冗余样本。剪枝之前，输入视图被标记为基本视图（需要完整发送）或附加视图（需要进行剪枝，仅传输部分内容）。随后，剪枝过程会确定附加视图中的哪些像素已经存在于基本视图或其他附加视图中，因此无需传输和渲染。相反，无法从其他视图中恢复的像素则会予以保留。接着，如果合成深度值/纹理亮度值与源深度值/纹理亮度值之间的差异不超过给定阈值，则相应像素将被剪除。第二轮剪枝 [15] 则基于全局颜色匹配进行，用于恢复部分最初因深度不完善或不同源纹理之间的光照变化而被剪除的像素。

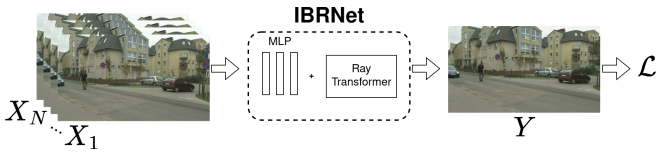
3. 提出的方法

本节介绍我们提出的方法 LeHoPP，该方法是在基于图像的渲染设置中，针对输入图像的像素级别执行的一种剪枝方法。图 1 展示了该方法的示意图。本节将首先介绍如何计算剪枝掩码，然后概述 LeHoPP 的算法流程。

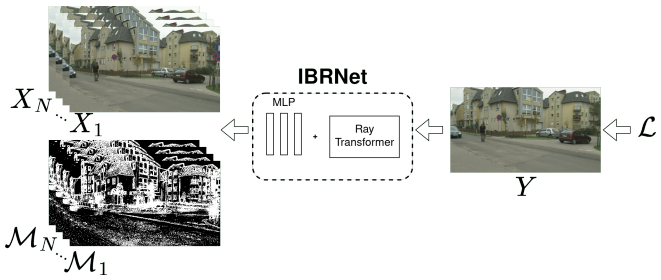
3.1. 剪枝掩码计算

让我们考虑多视图输入图像 $X_i \in \{0, 1, ..., 255\}^{W_i \times H_i \times 3}$，其中 $i = \{1, ..., N\}$， N 是源视图的数量， W_i 和 H_i 分别是宽度和

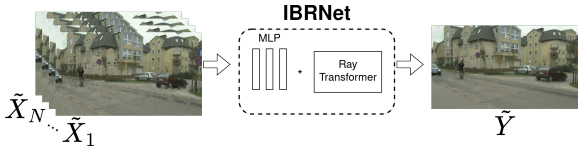
[1] 我们使用了以下网址提供的预训练模型：https://github.com/googleinterns/IBRNet



(a) 前向传播中的渲染损失计算。



(b) 反向传播中的重要性和掩码计算。



(c) 由剪枝和图像修复后的图像生成的渲染结果。

图 1：LeHoPP 方法应用概览。

第 i 个视图的高度。在将 X_i 输入 IBRNet 之前，会对图像进行预处理，包括标准化和缩放，将 X_i 转换为 $\hat{X}_i \in \mathbb{R}^{W_i \times H_i \times 3}$，具体遵循 IBRNet [4] 的标准实现。我们将为每个 X_i 和 $\hat{X}_i$ 关联一个剪枝掩码 $\mathcal{M}_i \in \{0,1\}^{W_i \times H_i}$。该掩码用于指示某些像素应予以保留（1）还是丢弃（0）。

为获得每个图像掩码的值，我们首先需要计算重要性分数 $\mathcal{I}_i$，该分数与源视图中的每个像素相关联。重要性函数表示渲染图像对输入图像中某个像素变化的敏感程度。我们知道，给定用于优化掩码的损失函数 $\mathcal{L}$（目标视图上的均方误差，与训练 IBRNet 时采用的损失保持一致），我们可以通过泰勒级数展开估计损失函数值的一些变化，如下所示

$$\Delta\mathcal{L} \approx \frac{\partial \mathcal{L}}{\partial \hat{X}_i(u,v)} \Delta \hat{X}_i(u,v). \tag{1}$$

据此，我们可以将重要性分数定义为

$$\mathcal{I}_i(u,v) = \left| \frac{\partial \mathcal{L}}{\partial \hat{X}_i(u,v)} \right| \cdot \left| \hat{X}_i(u,v) - \hat{X}_i^{\text{inp}}(u,v) \right|, \tag{2}$$

其中，u 和 v 表示图像中的像素坐标，而 $\hat{X}_i^{\text{inp}}$ 表示 X_i 的修复近似值。为最大限度地提升剪枝掩码 $\mathcal{M}_i$ 的性能，源重要性分数 $\mathcal{I}_i$ 会在图像的 (R,G,B) 通道上累积，并在所有目标视图上取平均值。由于不同的子集附近的源视图用于渲染目标视图，因此，对于某个特定源视图，我们会对从不同目标损失计算中获得的梯度值取平均值。

与文献中的类似方法不同，[16, 17], 鉴于掩码值会被丢弃为像素，我们可以运行图像修复算法，以部分恢复丢失的信息。因此，我们的 $\Delta\hat{X}_i(u,v)$ 并非简单地取像素的值（这是任何剪枝算法中的传统做法），而是取真实值与其图像修复后估计值之间的距离。为了提高计算效率，我们通过对相邻像素的值取平均来进行估计：

$$\hat{X}_i^{\text{inp}}(u,v) = \frac{1}{8}\left[-\hat{X}_i(u,v) + \sum_{i=u-1}^{u+1} \sum_{j=v-1}^{v+1} \hat{X}_i(i,j) \right]. \tag{3}$$

这里，图像修复并非用于在新视角渲染后填补图像中的遮挡区域，而是在像素剪枝后进行，因为 IBRNet 是基于完整图像训练的。我们没有将黑色/灰色的剪枝像素输入 IBRNet 渲染器，而是对被剪枝的区域进行图像修复，并将其影响纳入我们的重要性评分中。

我们对帧内周期中的所有帧使用相同的掩码，以降低掩码更新的频率以及潜在传输开销。因此，重要性分数是在一个帧内周期（16 帧）的所有帧上累积得到的。最后，鉴于我们的目标是从输入视图中移除 $\gamma \in [0;1]$ 比例的像素，我们可以使用分位数函数 $\mathcal{Q}_{\mathcal{I}}(\gamma)$ 确定应用于重要性值的阈值，从而最终计算出剪枝掩码：

$$\mathcal{M}_i(u,v) = \begin{cases} 1 & \text{if } \mathcal{I}_i(u,v,c) \geq \mathcal{Q}_{\mathcal{I}}(\gamma), \\ 0 & \text{otherwise.} \end{cases} \tag{4}$$

3.2. LeHoPP 流程概述

LeHoPP 的方案如图 1 所示，由三个步骤组成。首先，将多视图图像 X_i 输入 IBRNet 并执行前向传播步骤，然后根据合成视图 Y 计算损失 $\mathcal{L}$（图 1a）。接着，通过反向传播，我们可以计算 $\frac{\partial \mathcal{L}}{\partial \hat{X}_i}$，据此按照公式 (2) 计算重要性分数 $\mathcal{I}_i$，并按照公式 (4) 计算掩码（图 1b）。最后，将掩码应用于输入图像以剪除像素，这些像素随后经过图像修复，所得图像 $\tilde{X}_i$ 被输入 IBRNet 渲染器，以合成目标视图 $\tilde{Y}$（图 1c）。

4. 结果与讨论

4.1. 测试条件

我们的数据集由来自 MIV CTC [6], 的八个透视序列组成，其中包含由稀疏相机配置捕获的高分辨率自然和计算机生成的多视图内容。

我们将 LeHoPP 与两种基线方法进行比较：基准*1* (32×32基于块的随机像素剪枝)和 *base2* (4×4基于块的随机像素剪枝)。由于我们没有找到任何能够解决我们这一特定场景下像素剪枝问题的现有方法，因此决定将所提出的方法与一种简单的“盲目”方法（随机剪枝）进行比较。我们强调，TMIV剪枝器并不适合作为我们实验设置中的对比方法，因为它针对的是不同的目标。该方法基于基于深度图像的渲染，而 IBRNet 处理的是仅纹理数据，并且其输出是一小组完整视图以及称为图像块的小块图像区域，这些内容无法应用于 IBRNet 的渲染场景。此外，由于我们不依赖深度图，因此使用我们的方法剪枝后得到的纹理视频也无法用于 TMIV 渲染。

在我们的实验中，修剪百分比 γ 是唯一变化的参数，用于展示质量与像素率之间的权衡。此外，剪枝像素在输入渲染之前会先进行图像修复，所采用的是 OpenCV 库中的 Telea [18] 方法。对于所有序列，渲染所使用的输入源视图数量均设为 9。我们使用 PSNR、SSIM [19], 和 LPIPS [20] 指标评估合成视图的质量，具体采用 IBRNet 提出的方案。

4.2. 实验结果

不同修剪百分比 γ（5%、10% 和 20%）下的结果如表 1 所示。它们表示目标视频帧的渲染视图质量，该质量是对一个序列中的所有视图取平均得到的。我们可以观察到，平均而言，对于所有计算得到的指标以及给定的三个 γ值，LeHoPP 的表现均优于 基准*1*和 *base2* 。我们进一步注意到，尽管 γ的 20%取值显著减少了渲染过程中的源像素数量，但其性能与锚点相差不远；在锚点设置下，渲染直接基于未经任何剪枝的源输入完成。我们的方法在 Carpark、Fan、Hall、Street 和 Mirror 序列上表现尤为出色。然而，在 Painter、青蛙和萨满序列上，我们观察到其有效性略低。导致这一现象的一些次优因素（出于计算效率的考虑）包括：用于重要性计算的图像修复近似（见式（3）），以及剪枝过程本身的一次性特性（采用多个更小的步骤可以使式（1）中的近似更加精确），更重要的是，

未对 IBRNet 进行微调，以增强其对剪枝像素的鲁棒性。

4.3. 分析与讨论

本文的主要贡献——像素重要性计算——已被证明是像素剪枝的一项可靠指标。下面以一个示例场景为例，仔细考察 LeHoPP 各关键要素的贡献，以及它相较于所考察的基线方法为何能始终表现更优：我们将使用 Street序列进行深入分析。

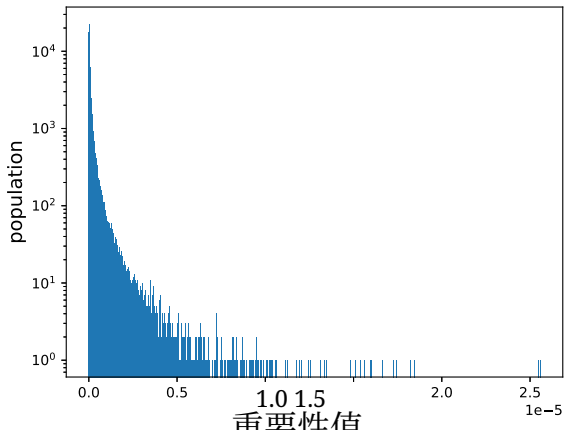


图2：Street 视频序列在第 0 帧、视角 1 下的重要性值直方图。



(a) 剪枝输入 ($\gamma = 20\%$)

(b) 未经输入修复的输出 (21.35 dB)

(c) 采用图像修复的剪枝输入 ($\gamma = 20\%$)

(d) 输入修复后的输出 (40.73 dB)

图 3：给定输入视图下目标图像的定性比较：为便于可视化，(a) 中的剪枝像素以绿色显示。

图2展示了重要性值的分布 $\mathcal{I}_i$：大多数像素（总体）的重要性值都非常小，这表明原则上可以剪枝大量像素。

在渲染前用图像修复后的对应像素替换剪枝像素，有助于网络以更高质量渲染视图。图 3 对比了 Street 序列合成视图的质量：一种是在渲染前未进行图像修复（图 3b），另一种是在渲染前进行了图像修复（图 3d）。

此外，为了从数据率方面衡量 LeHoPP 相较于 *base2* 的性能，我们进行了比较实验。

表 1：不同像素剪枝百分比下的结果：5%、10% 和 20%。

指标	γ	方法	序列 画家	Frog	Carpark	Fan	萨满	Hall	Street	镜子	平均值
峰值信噪比 (↑)	0%	锚点	31.14	28.18	38.96	37.67	47.06	39.85	43.36	33.38	37.45
	5%	基准*1*	29.89	27.25	35.79	33.62	41.41	38.11	37.78	30.43	34.29
		base2	30.72	27.75	37.40	35.17	43.42	39.23	40.49	31.64	35.73
		LeHoPP	**31.04**	**27.98**	**38.76**	**37.40**	**44.76**	**39.78**	**42.73**	**33.13**	**36.95**
	10%	基准*1*	29.03	26.43	33.62	31.38	38.58	36.78	35.01	29.08	32.49
		base2	30.31	**27.30**	36.05	33.57	41.29	38.65	38.58	30.57	34.54
		LeHoPP	**30.44**	27.27	**38.45**	**36.76**	**41.74**	**39.69**	**42.10**	**32.37**	**36.10**
	20%	基准*1*	27.74	24.96	30.31	28.57	35.36	34.80	31.57	27.22	30.07
		base2	**29.48**	**26.40**	33.78	31.21	**38.46**	37.49	35.78	29.05	32.71
		LeHoPP	28.04	24.90	**37.39**	**34.26**	34.55	39.42	**40.37**	**29.75**	**33.58**
结构相似性 (↑)	0%	锚点	0.9561	0.9092	0.9901	0.9926	0.9982	0.9965	0.9953	0.9684	0.9758
	5%	基准*1*	0.9450	0.8941	0.9788	0.9784	0.9911	0.9934	0.9853	0.9538	0.9651
		base2	0.9484	0.8962	0.9819	0.9819	0.9930	0.9948	0.9881	0.9575	0.9677
		LeHoPP	**0.9511**	**0.9038**	**0.9878**	**0.9913**	**0.9952**	**0.9963**	**0.9930**	**0.9675**	**0.9733**
	10%	基准*1*	0.9337	0.8771	0.9652	0.9626	0.9827	0.9914	0.9739	0.9388	0.9532
		base2	0.9397	0.8820	0.9718	0.9695	0.9869	0.9929	0.9797	0.9455	0.9585
		LeHoPP	**0.9412**	**0.8923**	**0.9844**	**0.9872**	**0.9871**	**0.9961**	**0.9894**	**0.9640**	**0.9677**
	20%	基准*1*	0.9103	0.8390	0.9333	0.9272	0.9641	0.9855	0.9473	0.9069	0.9267
		base2	**0.9197**	0.8505	0.9468	0.9403	**0.9722**	0.9883	0.9594	0.9180	0.9369
		LeHoPP	0.9061	**0.8509**	**0.9745**	**0.9657**	0.9473	**0.9956**	**0.9791**	**0.9436**	**0.9453**
LPIPS（↓）	0%	锚点	0.1260	0.1637	0.0694	0.0313	0.0162	0.1167	0.0559	0.0796	0.0823
	5%	基准*1*	0.1548	0.1816	0.0883	0.0591	0.0458	0.1335	0.0805	0.1102	0.1067
		base2	**0.1487**	0.1756	0.0831	0.0561	**0.0364**	0.1352	0.0719	0.1046	0.1015
		LeHoPP	0.1496	**0.1743**	**0.0786**	**0.0423**	0.0536	**0.1221**	**0.0667**	**0.0927**	**0.0975**
	10%	基准*1*	0.1834	0.2018	0.1108	0.0897	0.0779	0.1519	0.1073	0.1404	0.1329
		base2	**0.1749**	**0.1909**	0.1008	0.0845	**0.0606**	0.1534	0.0916	0.1323	0.1236
		LeHoPP	0.1853	0.1946	**0.0909**	**0.0628**	0.1077	**0.1296**	**0.0813**	**0.1136**	**0.1207**
	20%	基准*1*	0.2397	0.2471	0.1627	0.1547	0.1434	0.1908	0.1648	0.2007	0.1878
		base2	**0.2319**	**0.2286**	0.1446	0.1448	**0.1167**	0.1910	0.1388	0.1920	**0.1736**
		LeHoPP	0.2781	0.2544	**0.1242**	**0.1329**	0.2517	**0.1483**	**0.1189**	**0.1756**	0.1855

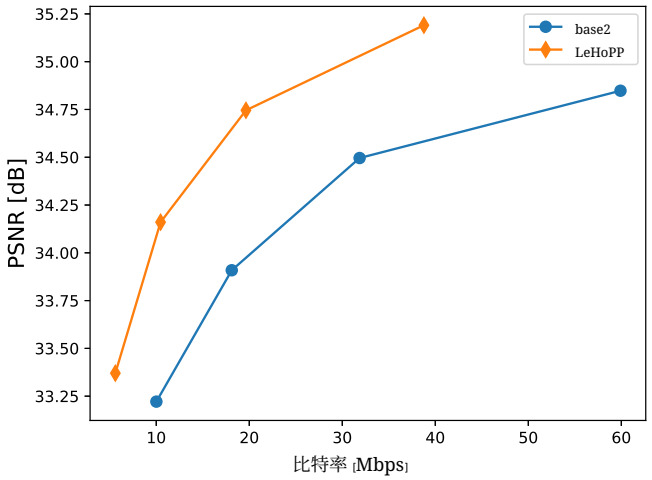


图 4：与*base2*相比，LeHoPP 方法在 Street 视频序列上的率失真性能。

压缩实验。本研究采用 HEVC 并行编码（simulcast）配置，QP 为 $\in \{22, 27, 32, 37\}$。首先，对 Street 序列的所有视图进行剪枝，剪枝比例为 $\gamma = 10\%$；随后对其进行压缩，最后将解码后的剪枝视图输入 IBRNet 渲染器，且不进行图像修复。与*base2*相比，LeHoPP 实现了 54.24% 的 BD-比特率 [21]，并且在给定 QP 的情况下，其质量和比特率均始终表现更优（图 4）。

MIV 标准 [14] 中的像素率限制为每秒 30 帧、3200 万像素。高分辨率且源视图数量较多的序列几乎无法满足这一要求，即使仅考虑传输源纹理视图而不包含对应的深度图。因此，本研究测试的 5% 至 20% 修剪百分比很适合展示实际可能的使用场景，此时保留的像素数量仍处于规定的像素率范围内。

我们用于计算重要性分数的图像修复近似是图像修复的一种代理方法，并且在剪枝区域较小时是一种很好的简化。

因此，表 1 中的结果表明，与基准*1* 和*base2* 相比，LeHoPP 在较小的 γ 值下取得了更高的增益。此外，在进行重要性计算时，由于内存限制以及反向传播计算图规模较大，在渲染 Shaman、Painter、Frog、Mirror 和 Fan 时，仅使用了较少的视图作为源输入。这也可能导致这些序列的重要性值不够精确。此外，对于重要性值的计算，该方法针对特定目标视图，每帧需要耗费数分钟的计算时间。

TMIV 修剪技术依赖深度图和纹理，并且是为 DIBR 优化而设计的。此外，MIV 标准无法传输捕获场景的所有源视图。因此，尽管在数据层面兼容（MIV 的 GA 配置文件可以传输仅含纹理的多视图视频），它仍与 IBRNet 这类神经渲染器不兼容。因此，若要用于潜在的传输场景，就有必要为 IBRNet 渲染器修剪纹理。所提出的系统并非独立的沉浸式视频编码方案，而是此类系统中首个必要的预处理阶段。我们的研究展示了这种方法：它能够学习哪些像素并非必要，在保持较高合成质量的同时降低像素率。

5. 结论

本文提出了 LeHoPP，这是一种学习如何修剪像素的新方法，可在多视图设置下与基于神经网络的图像合成器配合使用，且无需重新训练模型。LeHoPP 利用场景不同视图之间现有的冗余，并使用 IBRNet 计算渲染视图损失及其对每个输入像素的影响，从而根据所需的修剪百分比，为每个视图计算剪枝掩码。我们的方法优于随机块级修剪基线方法，并在不对场景进行微调的情况下，在 MPEG 沉浸式视频数据集上展现出良好的效率。我们相信，这项工作将推动神经渲染器的进一步发展，并促进 MIV 等标准得到更广泛的应用。未来的工作可以改进剪枝掩码的计算方式，并将剪枝模块集成到训练流程中，以获得更好的性能。

6. 致谢

本研究部分得到了欧盟在 NextGenerationEU 框架下意大利国家复苏与韧性计划（NRRP）的支持，具体来自“未来电信”合作项目（PE0000001——“RESTART”计划）。

7. 参考文献

[1] E. Penner 和 L. Zhang，“用于视图合成的软三维重建”，*ACM* 图形学汇刊 *(TOG)*，第 36 卷，第 1–11 页，2017 年。

[2] B. Mildenhall、P. P. Srinivasan、R. Ortiz-Cayon、N. K. Kalantari、R. Ra- mamoorthi、R. Ng 和 A. Kar，《局部光场融合：基于指导性采样准则的实用视图合成》，*ACM* 图形学汇刊（*TOG*），2019 年。[3] B. Mildenhall、P. P. Srinivasan、M. Tancik、J. T. Barron、R. Ramamoor- thi 和 R. Ng，《NeRF：将场景表示为用于视图合成的神经辐射场》，载于欧洲计算机视觉会议。Springer，2020 年 8 月，第 405–421 页。[4] Q. Wang、Z. Wang、K. Genova、P. Srinivasan、H. Zhou、J. T. Barron、R. Martin-Brualla、N. Snavely 和 T. Funkhouser，《IBRNet：学习多视图基于图像的渲染》，载于 *2021 IEEE/CVF* 计算机视觉与模式识别会议（*CVPR*），2021 年，第 4688–4697 页。[5]《ISO/IEC FDIS 23090-12 MPEG 沉浸式视频文本》，ISO/IEC JTC1/SC29/WG4 MPEG2021/N00111，2021 年 7 月。[6] J. Jung 和 B. Kroon，《MPEG 沉浸式视频通用测试条件》，ISO/IEC JTC 1/SC 29/WG 04 0051，2020 年 10 月。[7] M. Wien、J. M. Boyce、T. Stockhammer 和 W.-H. Peng，《沉浸式视频编码的标准化状态》，*IEEE* 新兴电路与系统专题期刊，第 9 卷，第 1 期，第 5–17 页，2019 年 3 月。[8] V. K. M. Vadakital、A. Dziembowski、G. Lafruit、F. Thudor、G. Lee 和 P. R. Alface，《MPEG 沉浸式视频标准——当前状态与未来展望》，*IEEE* 多媒体，第 29 卷，2022 年 3 月，第 101 页。[9] B. Salahieh、B. Kroon、J. Jung、B. Kroon 和 A. Dziembowski，《MPEG 沉浸式视频测试模型 7》，ISO/IEC JTC 1/SC 29/WG 4 N0005，2020 年 10 月。[10] S. Fachada、D. Bonatto、A. Schenkel 和 G. Lafruit，《用于虚拟现实的多参考视图基于深度图像的视图合成》，载于 *2018* 年 *3DTV* 会议：*3D* 视频的真实视觉——采集、传输与显示（*3DTV-CON*），2018 年，第 1–4 页。[11] D. Mieloch、P. Garus、M. Milovanovi´c、J. Jung、J. Y. Jeong、S. L. Ravi 和 B. Salahieh，《MPEG 沉浸式视频中解码器端深度估计的概述与效率》，*IEEE* 电路与系统视频技术汇刊，第 32 卷，第 9 期，第 6360–6374 页，2022 年。[12] K. He、X. Chen、S. Xie、Y. Li、P. Doll´ar 和 R. Girshick，《掩码自编码器是可扩展的视觉学习器》，载于*IEEE/CVF* 计算机视觉与模式识别会议论文集，2022 年，第 16 000–16 009 页。[13] J. Tack、S. Kim、S. Yu、J. Lee、J. Shin 和 J. R. Schwarz，《通过基于误差的上下文修剪实现隐式神经表示的高效元学习》，*arXiv* 预印本 *arXiv:2302.00617*，2023 年。[14] B. Kroon、V. K. Malamal Vadakital 和 J. Jung，《沉浸式视频通用测试条件推荐像素率限制》，ISO/IEC JTC 1/SC 29/WG 11 MPEG/m49826，2019 年 7 月。[15] H.-C. Shin、J.-Y. Jeong、G. Lee、M. U. Kakli、J. Yun 和 J. Seo，《提升 MPEG 沉浸式视频视觉质量的改进修剪算法》，*ETRI* 期刊，第 44 卷，第 1 期，第 73–84 页，2022 年。[16] E. Tartaglione、A. Bragagnolo、A. Fiandrotti 和 M. Grangetto，《基于损失的敏感度正则化：迈向深度稀疏神经网络》，神经网络，第 146 卷，第 230–237 页，2022 年。[17] C. L. Deng 和 E. Tartaglione，《压缩显式体素网格表示：快速 NeRF 也能变得小巧》，载于*IEEE/CVF* 冬季计算机视觉应用会议论文集（*WACV*），2023 年 1 月，第 1236–1245 页。[18] A. Telea，《一种基于快速行进法的图像修复技术》，图形工具期刊，第 9 卷，2004 年。[19] Z. Wang、A. Bovik、H. Sheikh 和 E. Simoncelli，《图像质量评估：从误差可见性到结构相似性》，*IEEE* 图像处理汇刊，第 13 卷，第 4 期，第 600–612 页，2004 年。[20] R. Zhang、P. Isola、A. A. Efros、E. Shechtman 和 O. Wang，《深度特征作为感知度量的惊人效用》，载于*2018 IEEE/CVF* 计算机视觉与模式识别会议，2018 年，第 586–595 页。[21] G. Bjontegaard，《率失真曲线之间平均 PSNR 差异的计算》，ITU-T Q.6/16，文件 VCEG-M33，2001 年 4 月。