西北工业大学

数字图像处理–论文翻译

原论文标题：Fully Sparse 3D Occupancy Prediction

高宏达

软件学院

软件工程

2024 年 11 月

学号：2022302798

Northwestern Polytechnical University

摘要

乘员预测在自动驾驶中起着举足轻重的作用。以往的方法通常构建密集的三维体积，忽略了场景固有的稀疏性，计算成本高昂。为了弥补这一缺陷，我们引入了一种新颖的完全稀疏占位网络，称为 SparseOcc。SparseOcc 最初从仅摄像头输入重建稀疏三维表示，随后通过稀疏查询从三维稀疏表示预测语义/实例占用率。我们设计了一种遮罩引导的稀疏采样，使稀疏查询能够以完全稀疏的方式与二维特征进行交互，从而避免昂贵的密集特征或全局关注。此外，我们还设计了一种周到的基于射线的评估指标，即 RayIoU，以解决传统体素级 mIoU 标准中沿深度轴的不一致性惩罚问题。SparseOcc 证明了它的有效性，在输入 7 个历史帧的情况下，RayIoU 达到 34.0，同时保持了 17.3 FPS 的实时推理速度。通过加入更多的历史帧至 15 个，SparseOcc 不断提高其性能，在不增加任何附加功能的情况下达到 35.1 RayIoU。

1 介绍

以视觉为中心的三维空间占用预测 [1] 侧重于从视觉图像中将三维场景划分为结构化网格。每个网格都会被贴上一个标签，表明其是否被占用。这项任务比三维物体检测提供了更多的几何细节，并为基于激光雷达的感知提供了另一种表示方法 [63, 23, 60, 61, 62, 31, 32]。

现有方法 [27, 16, 57, 45, 28] 通常会构建密集的三维特征，但计算开销较大（例如，在 Tesla A100 GPU 上为 2 3 FPS）。然而，密集表示对于占位预测并非必要。我们对几何稀疏性进行了统计，发现 90% 以上的体素是空的。这体现了利用稀疏性加速占用预测的巨大空间。一些研究 [26, 19] 探索了三维场景的稀疏性，但它们仍然依赖于稀疏到密集模块来进行密集预测。这启发我们寻求一种不需要任何密集设计的纯稀疏占用网络。

在本文中，我们提出了首个完全稀疏占位网络 SparseOcc。如图 1 (a) 所示，SparseOcc 包括两个步骤。首先，它利用稀疏体素解码器，以从粗到细的方式重建场景的稀疏几何图形。这样只对非自由区域建模，大大节省了计算成本。其次，我们设计了一个具有稀疏语义/实例查询功能的掩码转换器，以便从稀疏空间预测片段的掩码和标签。掩码转换器不仅提高了语义占用的性能，还为全视角占用铺平了道路。设计了掩码引导的稀疏采样，以在掩码转换器中实现稀疏交叉关注。

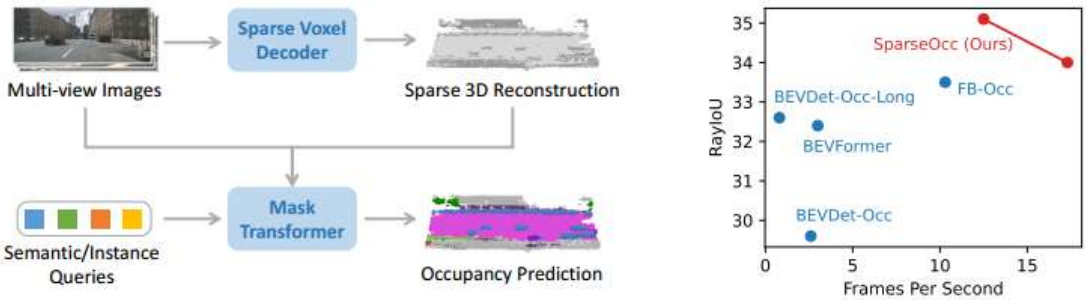


图 1: (a) SparseOcc 通过稀疏体素解码器从仅相机输入重建稀疏 3D 表示，然后通过一组稀疏查询估计每个片段的掩码和标签。(b) Occ3D-nuScenes 验证分割的性能比较。FPS 是在带有 PyTorch fp32 后端的 Tesla A100 上测量的。

因此，我们的 SparseOcc 充分利用了稀疏属性，摆脱了任何密集设计，如密集三维特征、稀疏到密集模块和全局注意力。

此外，我们还注意到流行的体素级平均交叉联合（mIoU）占用率评估指标存在缺陷，并进一步设计了射线级评估指标 RayIoU 作为解决方案。鉴于未扫描体素的标记模糊不清，mIoU 标准是一个难以确定的公式。以前的方法 [50] 通过只评估观察区域来缓解这一问题，但会引起深度不一致惩罚的额外问题。而 RayIoU 可同时解决上述两个问题。它通过检索指定射线的深度和类别预测来评估预测的三维占位体积。具体来说，RayIoU 将查询光线投射到预测的三维体积中，并将其首次触及的所占体素网格的距离和类别正确的光线判定为真阳性预测。这就制定了一个更加公平合理的标准。

得益于稀疏性设计，SparseOcc 在 Occ3D-nuScenes [50] 上实现了 34.0 RayIoU，同时保持了 17.3 FPS 的实时推理速度（Tesla A100，PyTorch fp32 后端），并输入了 7 个历史帧。通过加入更多的历史帧到 15 帧，SparseOcc 的性能不断提高，达到了 35.1 RayIoU，实现了最先进的性能，而且没有任何附加功能。SparseOcc 与以往方法在性能和效率方面的比较见下表。

我们将我们的贡献总结如下：

1. 我们提出了 SparseOcc，它是首个无需耗时的密集设计的完全稀疏占位网络。它在 Occ3D-nuScenes 基准上实现了 34.0 RayIoU，实时推理速度为 17.3 FPS。

2. 我们提出了用于占用率评估的射线标准 RayIoU。通过查询三维体积的射线，它解决了未扫描自由体素的模糊惩罚问题和 mIoU 指标中的不一致深度惩罚问题。

2 相关工作

基于摄像头的三维占位预测。占位网络最初是由 Mescheder 等人提出的 [37, 42]，重点关注三维空间中的连续物体表示。最近的变体 [1, 4, 45, 50, 54, 56, 11, 58] 大多从鸟瞰（BEV）感知 [25, 24, 27, 16, 15, 18, 17, 55, 34, 35, 30, 33, 53, 59] 中汲取灵感，从图像输入中预测体素级语义信息。例如，MonoScene[4] 通过视线投影模块连接的二维和三维 UNet[43] 估算占用率。SurroundOcc [57] 提出了一种从粗到

细的架构。然而，大量体素查询的计算量很大。TPVFormer [19] 提出了三视角视图表示法来补充垂直结构信息，但这不可避免地会导致信息丢失。VoxFormer [26] 基于单目深度预测初始化稀疏查询。不过，VoxFormer 并非完全稀疏，因为它仍需要一个稀疏到密集的 MAE [13] 模块来完成场景。在 CVPR 2023 占有率挑战赛中出现了一些方法 [28, 40, 9]，但它们都没有利用完全稀疏的设计。在本文中，我们首先探索了全稀疏架构，用于仅摄像头输入的三维占位预测。

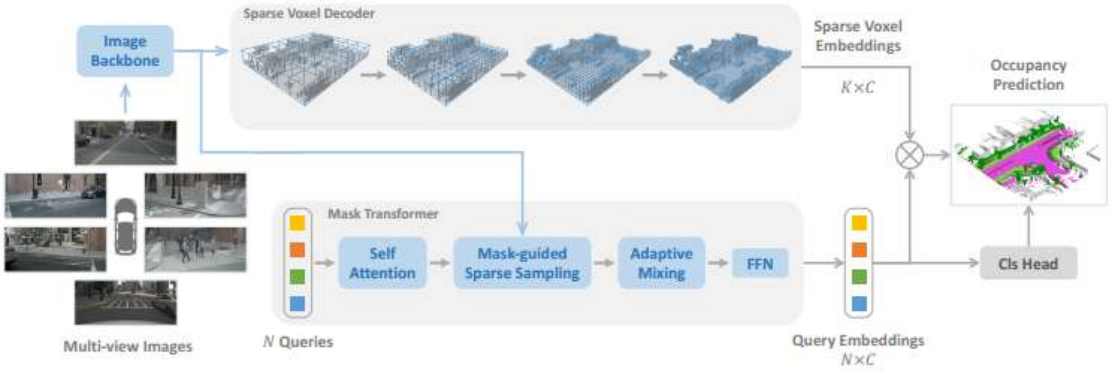


图 2: SparseOcc 是一个完全稀疏的架构，因为它既不依赖于密集的 3D 特征，也没有稀疏到密集和全局关注操作。稀疏体素解码器重建场景的稀疏几何，由 K 个体素（K W × H × D）组成。掩模转换器然后使用 N 个稀疏查询来预测每个片段的掩模和标签。通过用实例查询代替语义查询，SparseOcc 可以很容易地扩展到全局占用。

用于 3D 视觉的稀疏架构。稀疏架构利用点云固有的稀疏性，在基于激光雷达的重建 [48] 和感知 [7, 63, 60, 61] 中得到广泛应用。然而，当涉及到从视觉到 3D 的任务时，由于缺乏点云输入，直接适应是不可行的。之前的一项工作 SparseBEV [33] 为基于摄像头的 3D 物体检测提出了一种完全稀疏的架构。然而，直接调整这种方法并非易事，因为三维物体检测侧重于稀疏的物体集，而三维占位需要对每个体素进行密集预测。因此，为三维占位预测设计一个完全稀疏的架构仍然是一项具有挑战性的任务。

从摆拍图像进行端到端三维重建。与三维占位预测相关的一项任务是，三维重建可从多个摆好姿势的图像中恢复三维几何图形。最近的方法侧重于更紧凑、更高效的端到端三维重建管道 [39, 47, 2, 46, 10]。Atlas [39] 从多视角输入图像中提取特征，并将其映射到三维空间，从而构建截断符号距离函数 [8]。NeuralRecon [47] 直接将局部表面重建为稀疏的 TSDF 体积，并使用基于 GRU 的 TSDF 融合模块来融合先前片段的特征。VoRTX [46] 利用变换器来解决多视角图像中的遮挡问题。

掩码变换器。最近，人们广泛研究了同时处理语义分割和实例分割的统一分割模型。Cheng 等人首先从模型架构、损失函数和训练策略等方面提出了用于统一分割的 MaskFormer [6]。随后，Mask2Former [5] 引入了掩码注意，对实例掩码的感受野进行限制，以获得更好的性能。随后，Mask3D [44] 成功地将掩膜变换器扩展用于点云分割，并取得了最先进的性能。OpenMask3D [49] 进一步实现了开放词汇的三维实例分割任务，并提出了零镜头三维分割模型。

3 SparseOcc

SparseOcc 是一种以视觉为中心的占用模型，只需要摄像头输入。如图 2 所示 SparseOcc 包含三个模块：一个由图像主干和 FPN [29] 组成的图像编码器，用于从多视角图像中提取二维特征；一个稀疏体素解码器（第 3.1 节），用于从图像特征中预测具有相关嵌入的稀疏类无关三维占位；一个掩码变换器解码器（第 3.2 节），用于区分稀疏三维空间中的语义和实例

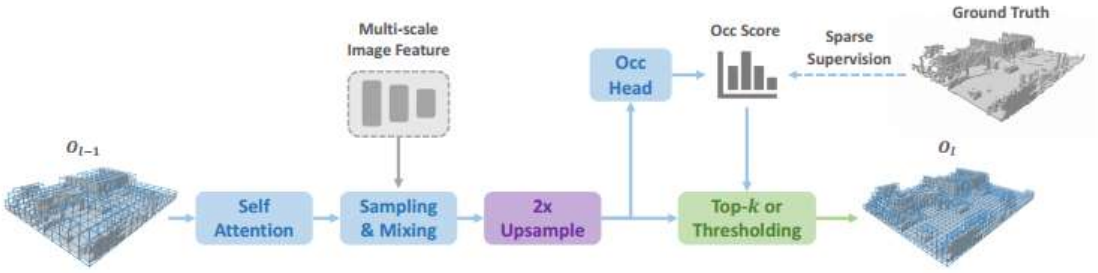


图 3: 稀疏体素解码器采用由粗到精的三层管道。在每一层中，我们利用类似于变压器的架构进行 3D-2D 交互。在每一层的最后，将体素分辨率上采样 2 倍，并估计体素占用的概率。

3.1 稀疏体素解码器

由于三维占位地面实况 [50, 45, 57, 54] 是一个尺寸为 W × H × D（如 200×200×16）的稠密体，现有方法通常会构建一个形状为 W × H × D × C 的稠密三维特征，但会产生计算开销。在本文中，我们认为占用率预测并不需要这种密集表示。在我们的统计中，我们发现场景中超过 90% 的体素是自由的。这促使我们探索一种稀疏的三维表示法，只对场景中的非自由区域进行建模，从而节省计算资源。

整体架构。我们设计的稀疏体素解码器如图 3 所示。一般来说，它遵循从粗到细

的结构，但只对非自由区域进行建模。解码器从在三维空间（如 25×25）中平均分配的一组粗体素查询开始。在每一层中，我们首先对每个体素进行 2 倍的升采样，例如，一个大小为 d 的体素将被升采样为 8 个大小为 d 2 的体素。接下来，我们为每个体素估算占用率分数，并进行修剪以去除无用的体素网格。在这里，我们有两种剪枝方法：一种是基于阈值（例如，只保留分数大于 0.5 的体素）；另一种是通过 top-k 选择。k 是一个与数据集相关的参数，是通过计算每个样本在不同分辨率下非自由体素的最大数量得到的。剪枝后的体素标记将作为下一层的输入

详细设计。在每一层中，我们使用类似变压器的架构 [52] 来处理体素查询。具体架构的灵感来自 SparseBEV [33]，这是一种使用稀疏方案的检测方法。具体来说，在第 l 层中，有 Kl-1 个由三维位置和 C-dim 内容向量描述的体素查询，我们首先使用自我关注来聚合这些查询体素的局部和全局特征。然后，使用线性层对来自相关内容向量的每个体素查询生成三维采样偏移 {(xi, yi, zi)}。这些采样偏移量被用来转换体素查询以获得全局坐标中的参考点。最后，我们将这些采样的参考点投影到多视图图像空间中，通过自适应混合来整合图像特征 [12,51,20]。总之，我们的方法与 SparseBEV 的不同之处在于将查询公式从柱子转移到 3D 体素。其他组件，如自关注，自适应采样和混音是直接借用的。

时序建模。先前的密集占用方法 [27,16] 通常将历史 BEV/3D 特征扭曲为当前时间戳，并使用可变形的注意力 [64] 或 3D 卷积来融合时间信息。然而，由于我们的 3D 特征的稀疏性，这种方法并不直接适用于我们的情况。为了解决这个问题，我们利用前面提到的全局采样参考点的灵活性，将它们扭曲为以前的时间戳来采样历史多视图图像特征。通过自适应混合对采样的多帧特征进行叠加和聚合，从而实现时间建模。

监督。我们计算每层稀疏体素的损失。我们使用二元交叉熵（BCE）损失作为监督，假设我们正在重建一个类别不可知论的稀疏占用空间。只对保留的稀疏体素进行监督，忽略前期剪枝过程中丢弃的区域。而且，由于严重的类别不平衡，模型很容易被占比较大的类别所主导，比如地面，从而忽略了场景中其他重要的元素，比如汽车、人等。因此，不同类别的体素被赋予不同的损失权值。例如，c 类体素的损失权重为：

$$w_c = \frac{\sum_{i=1}^{C} M_i}{M_c}, \tag{1}$$

其中 Mi 为 ground truth 中属于第 i 类的体素数。

3.2 掩膜转换器

我们的掩码转换器的灵感来自 Mask2Former[5]，它使用 N 个稀疏语义/实例查询，通过二进制掩码查询 Qm(属于 [0，1]）和内容向量 Qc（属于 R）解耦。掩模变压器包括三个步骤：多头自关注（MHSA)、掩模引导稀疏采样和自适应混合。MHSA 通常用于不同查询之间的交互。掩模引导稀疏采样和自适应混合负责查询和二维图像特征之间的交互。

掩模引导稀疏采样。mask transformer 的一个简单基线是使用 Mask2Former 中的 masked crossattention 模块。然而，它考虑了键的所有位置，具有难以忍受的计算。这里，我们设计了一个简单的替代方案。我们首先在由前面的第 (L1) 个 Transformer 解码器层预测的遮罩内随机选择一组 3D 点。然后，我们将这些 3D 点投影到多视图图像上，并通过双线性插值来提取它们的特征。此外，我们的稀疏采样机制通过简单地扭曲采样点 (如在稀疏体素解码器中所做的) 使得时间建模更容易。

预测。对于类别预测，我们基于查询嵌入 Qc 应用具有 sigmoid 激活的线性分类器。对于掩码预测，查询嵌入通过 MLP 转换为掩码嵌入。屏蔽嵌入 M R Q×C 具有与查询嵌入 Qc 相同的形状，并且与稀疏体素嵌入 V R K×C 点积以产生屏蔽预测。因此，我们的掩模变换器的预测空间被约束到来自稀疏体素解码器的稀疏 3D 空间，而不是完整的 3D 场景。屏蔽预测将作为下一个变换器层的屏蔽查询 Qm。

监督。来自稀疏体素解码器的重建结果可能不可靠，因为它可能忽略或不准确地检测某些元素。因此，监督掩模变换器存在一定的挑战，因为它的预测被限制在这个不可靠的空间内。在遗漏检测的情况下，在预测的稀疏占用率中缺少一些地面实况片段，我们选择丢弃这些片段以防止混淆。至于检测不准确的元素，我们简单地将它们归类为附加的“无对象”类别。

损失函数。根据马斯克公式 [6]，我们使用匈牙利匹配法将地面真实值与预测值进行匹配。焦点损失 Lfocal 用于分类，而 DICE 损失 [38] Ldice 和 BCE 掩模损失 Lmask 的组合用于掩模预测。因此，SparseOcc 的总损耗由四部分组成:

$$L = L_{focal} + L_{mask} + L_{dice} + L_{occ}, \tag{2}$$

其中 Locc 是稀疏体素解码器的损失。

4 射线级 mIoU

4.1 重新审视体素水平的 mIoU

Occ3D 数据集 [50] 及其提议的评估指标被广泛认为是该领域的基准。地面真实占用率从激光雷达点云重建，并采用体素水平的平均交集 (mIoU) 来评估性能。由于距离、遮挡等因素，积累的点云并不完美。一些未被激光雷达扫描的区域被标记为空闲区域，从而导致实例支离破碎。这就产生了标签不一致的问题。为了解决这个问题，Occ3D 使用二进制可见遮罩来指示在当前相机视图中是否观察到体素。只有观察到的体素有助于评估

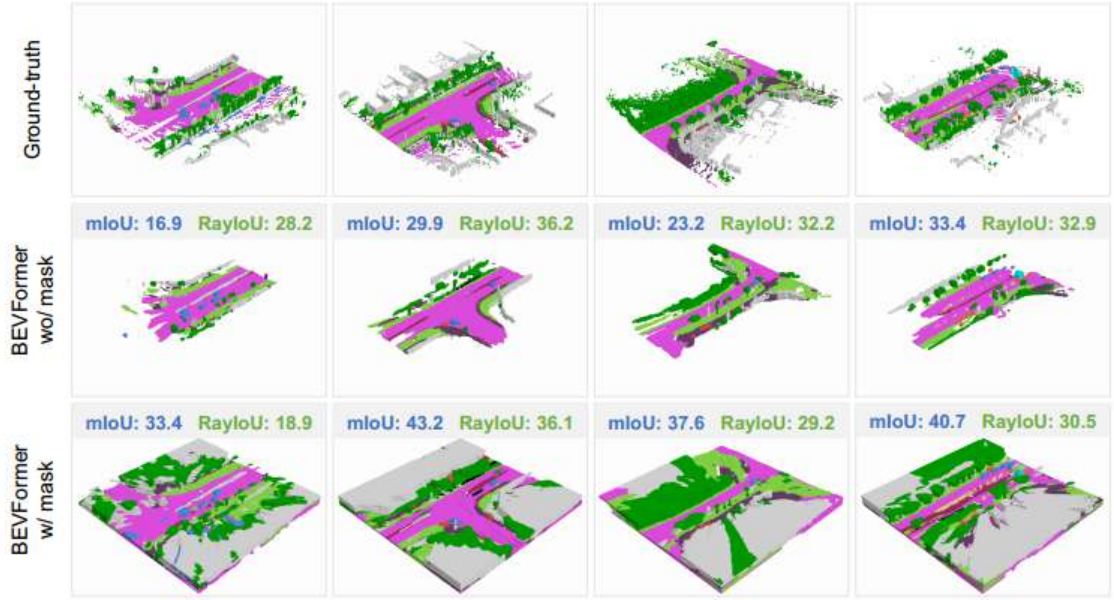


图 4: 定性和定量结果之间差异的可视化。我们观察到，使用可见遮罩训练现有的密集占用方法（例如 BEVFormer）会产生较厚的表面，从而导致当前 mIoU 指标的不合理膨胀改进。相比之下，我们的新 RayIoU 指标提供了更准确的模型性能反映。

然而，我们发现，仅在观察到的体素位置上计算 mIoU 仍然是脆弱的，并且可以通过预测更厚的表面来破解。密集方法 (例如, BEVFormer [27]) 可以通过使用可视掩码进行训练来轻松实现这一点。在训练过程中，表面后面的区域缺乏监督，导致模型用重复的预测填充它，从而导致更厚的表面。作为一个例子，考虑 BEVFormer，它在使用可见遮罩进行训练时会生成一个厚而嘈杂的表面 (见图 4)。尽管如此，在当前的评估协议下，其性能表现出不合理的夸大的改进 (+51500 万)。

定性和定量结果之间的不一致是由沿深度方向的不一致的惩罚引起的。图 5 中的玩具示例揭示了当前度量的几个问题:

1. 如果模型填充了表面后面的所有区域，它会不一致地影响深度预测。该模型可以通过填充表面后面的所有区域并预测更近的深度来获得更高的 IoU。这种厚表面问题在使用可见遮罩或 2D 监督训练的密集模型中非常常见。

2. 如果预测的占用率代表一个薄的表面，惩罚变得过于严格。即使只有一个体素的偏差也会导致 IoU 为零。

3. 可见遮罩仅考虑当前时刻的可见区域，将占用预测简化为深度估计任务，忽略了场景完成能力

4.2 光线投射平均 IoU

为了解决上述问题，我们提出了一种新的评价指标: 光线级 mIoU(简称 RayIoU)。在 RayIoU 中，集合元素是查询射线而不是体素。我们通过将查询射线投影到预测的 3D 占用体积中来模拟激光雷达行为。对于每个查询射线，我们计算它在与任何表面相交之前行进的距离，并检索相应的类标签。然后，我们将相同的过程应用于地面实况占用，以获得地面实况深度和类别标签。在射线不与地面真实中存在的任何体素相交的情况下，它将被排除在评估过程之外。

如图 6 (a) 所示，真实数据集中的原始激光雷达射线从近到远趋于不平衡。因此，我们对射线进行重新采样，以实现不同距离上的平衡分布 (图 6 (b))。在近场中，我们修改射线通道以在投影到地平面上时实现等距间隔。在远场，我们提高射线通道的角度分辨率，以确保在不同范围内的数据密度更加一致。此外，我们的查询射线可以源自自我路径的当前、过去或未来时刻的激光雷达位置。时间投射 (图 6 (c)) 允许我们在维持适定任务的同时评估场景完成性能。

如果类别标签一致并且地面真实深度和预测深度之间的 L1 误差小于某个阈值 (例

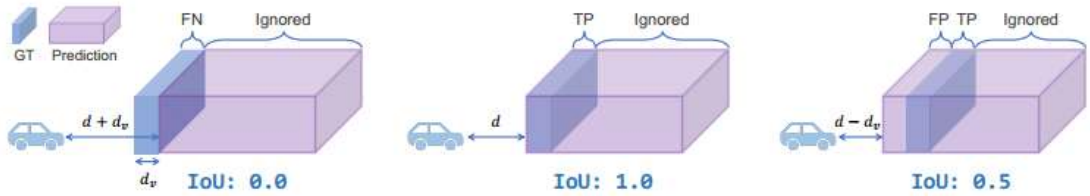


图 5: 由当前参数引起的不一致深度惩罚的说明。考虑这样一个场景，我们面前有一堵墙，它的真实距离是 d，厚度是 dv。当预测的厚度 dp 比 dv 高时，我们会遇到沿深度不一致的惩罚。具体来说，如果预测墙比实际距离（总距离 d + dv）远 dv，则其 IoU 将为零。相反，如果预测壁比实际壁（总距离 d - dv）近 dv，则 IoU 保持为 0.5。这是因为表面后面的所有体素都充满了重复的预测。同样，当预测深度为 d - 2dv 时，得到的 IoU 为 13，以此类推。

(a) Simulating LiDAR (b) Equal-distant resampling (c) Temporal casting

图 6: RayIoU 的占地面积。(a) 激光雷达原始射线样品在不同距离处不平衡。我们重新采样光线以平衡距离上的权重。(c) 为了研究场景补全的性能，我们建议通过在被访问的航路点上投射光线来评估在大时间跨度内可见区域的占用情况。

如，2m)，则查询射线被分类为真阳性 (TP)。设 C 为类别数，则 RayIoU 计算如下:

$$\text{RayIoU} = \frac{1}{C}\sum_{c=1}^{C}\frac{\text{TP}_c}{\text{TP}_c + \text{FP}_c + \text{FN}_c}, \tag{3}$$

其中 TPc、FPc 和 FNc 对应于 ci 类的真阳性、假阳性和假阴性预测的数量。

RayIoU 解决了上述所有三个问题:

1. 由于查询光线计算到它接触的第一个体素的距离，因此该模型不能通过预测较厚的表面来获得较高的 IoU。

2.RayIoU 基于距离阈值来确定真阳性，这减轻了体素级 mIoU 的过于严格的性质。

3. 查询射线可以来自场景中的任何位置。这种灵活性允许 RayIoU 考虑模型的场景完成能力，防止占用估计降低到仅仅深度预测。

Table 1: 3D occupancy prediction performance on Occ3D-nuScenes [50]. We use RayIoU to compare our SparseOcc with other methods. "8f" and "16f" mean fusing temporal information from 8 or 16 frames. SparseOcc outperforms all existing methods under a weaker setting.

Method	Backbone	Input Size	Epoch	RayIoU	$\text{RayIoU}_{1m, 2m, 4m}$			mIoU	FPS
BEVFormer (4f) [27]	R101	1600×900	24	32.4	26.1	32.9	38.0	39.2	3.0
RenderOcc [40]	Swin-B	1408×512	12	19.5	13.4	19.6	25.5	24.4	-
SimpleOcc [11]	R101	672×336	12	22.5	17.0	22.7	27.9	31.8	9.7
BEVDet-Occ (2f) [15]	R50	704×256	90	29.6	23.6	30.0	35.1	36.1	2.6
BEVDet-Occ-Long (8f)	R50	704×384	90	32.6	26.6	33.1	38.2	**39.3**	0.8
FB-Occ (16f) [28]	R50	704×256	90	33.5	26.7	34.1	39.7	39.1	10.3
SparseOcc (8f)	R50	704×256	24	34.0	28.0	34.7	39.4	30.1	**17.3**
SparseOcc (16f)	R50	704×256	24	35.1	29.1	35.8	40.3	30.6	12.5
SparseOcc (16f)	R50	704×256	48	**36.1**	**30.2**	**36.8**	**41.2**	30.9	12.5

5 实验

我们在 Occ3D-nuScenes[50] 数据集上评估了我们的模型。Occ3D-nuScenes 基于 nuScenes[3] 数据集，该数据集由 6 台环视相机、1 台激光雷达和 5 台雷达收集的大规模多模态数据组成。数据集总共有 1000 个视频，分为 700/150/150 个视频用于训练/验证/测试。每个视频大约有 20 秒的持续时间，每 0.5s 对关键样本进行注释。

我们使用提出的 RayIoU 来评估语义分割性能。查询光线来自自我路径的 8 个 LiDAR 位置。我们在三个距离阈值下计算 RayIoU: 1 米、2 米和 4 米。最终的排名指标是这些距离阈值的平均值。

5.1 实现细节

我们使用 PyTorch [41] 实现我们的模型。按照以前的方法，我们采用 ResNet-50 [14] 作为图像主干。遮罩变换由 3 层组成，不同层之间的权重是共享的。在我们的主要实验中，我们使用语义查询，其中每个查询对应于一个语义类，而不是一个实例。RayIoU 中的光线投射模块基于 [21] 的代码库实现。

在训练过程中，我们使用 AdamW [36] 优化器，全局批处理大小为 8。初始学习速率设置为 2×104，并按照余弦退火策略衰减。对于所有实验，我们训练我们的模型 24 个时期。FPS 是使用 PyTorch fp32 后端在 Tesla A100 GPU 上测量的。

5.2 表中的主要结果。

1 和图 1 (b), 我们在 Occ3D-nuScenes 的验证分割上比较了 SparseOcc 和先前的最先进方法。尽管在较弱的设置下 (ResNet-50 [14]，8 个历史帧，输入图像分辨率为 704 × 256)，SparseOcc 明显优于之前的方法，包括 CVPR 2023 年占位挑战赛的获胜者 FB-Occ，以及许多复杂的设计，包括前后视图转换，深度网，联合深度和语义预训练等。SparseOcc 实现了更好的结果 (+1.6 RayIoU)，同时比 FB-Occ 更快更简单，这证明了我们的解决方案的优越性。

我们在图 7 中进一步提供了定性结果。BEVDet-Occ 和 FB-Occ 都是密集方法，并且在表面后面做出许多冗余预测。相比之下，SparseOcc 丢弃了超过 90% 的体素，同时仍然有效地建模场景的几何形状并捕捉细粒度的细节。

5.3 消融

在本节中，我们对 Occ3D-nuScenes 的验证分割进行了消融，以确认每个模块的有效性。默认情况下，我们使用 SparseOcc 的单帧版本作为基线。我们对模型的选择是大胆的。

稀疏体素解码器与密集体素解码器。在表 2 中，我们将稀疏体素解码器与密集体素解码器进行了比较。在这里，我们实现了两条基线，它们都输出一个形状为

图 7: 语义占用预测的可视化比较。尽管丢弃了超过 90% 的体素，我们的 SparseOcc 有效地模拟了场景的几何形状，并捕获了细粒度的细节（例如，底部一行的黄色标记的交通锥）。

Table 2: Sparse voxel decoder vs. dense voxel decoder. Our sparse voxel decoder achieves nearly 4× faster inference speed than the dense counterparts.

Voxel Decoder	RayIoU	RayIoU_{1m}	RayIoU_{2m}	RayIoU_{4m}	FPS
Dense coarse-to-fine	**29.9**	**24.0**	30.4	**35.4**	6.3
Dense patch-based	25.8	20.4	26.0	30.9	7.8
Sparse coarse-to-fine	**29.9**	23.9	**30.5**	35.2	**24.0**

Table 3: Ablation of mask transformer (MT) and the cross attention module in MT. Mask-guided sparse sampling is stronger and faster than the dense cross attention.

MT	Cross Attention	RayIoU	RayIoU_{1m}	RayIoU_{2m}	RayIoU_{4m}	FPS
-	-	27.0	20.3	27.5	33.1	**29.0**
√	Dense cross attention	28.7	22.9	29.3	33.8	16.2
√	Sparse sampling	25.8	20.5	26.2	30.8	24.0
√	**+ Mask-guided**	**29.2**	**23.4**	**29.8**	**34.5**	24.0

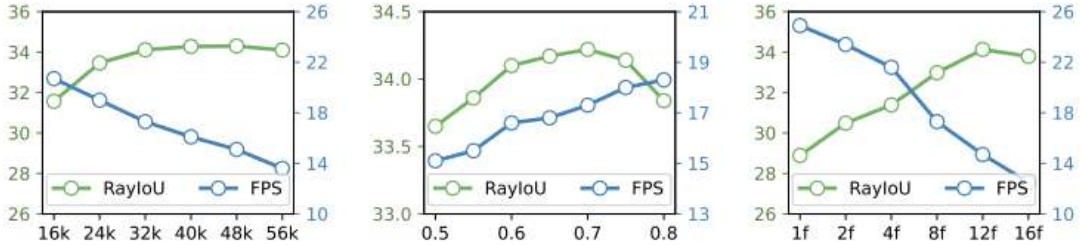
200×200×16×C 的密集特征图。第一个基线是不修剪空体素的从粗到精的架构。在这个基线中，我们还用 3D 卷积代替自关注，并使用 3D 反卷积来提升样本预测。另一个基线是基于 patch 的架构，通过将 3D 空间划分为少量的 patch 作为 PETRv2[35] 进行 BEV 分割。我们使用 25×25×2 = 1250 个查询，每个查询对应一个特定的形状块 8×8×8。反卷积层堆栈用于将粗查询提升到全分辨率 3D 体积。

从表中可以看出，密集的粗到细基线达到了 29.9 RayIoU 的良好性能，但推理速度较慢，为 6.3 FPS。基于补丁的版本稍微快一点，有 7.8 FPS 的推理速度，但性能严重下降了 4.1 rayiu。相反，我们的稀疏体素解码器产生 K × C 形状的稀疏 3D 特征（其中 K = 32000 200×200×16），在不影响性能的情况下，实现了比同类产品快近 4 倍的推理速度。这证明了稀疏设计的必要性和有效性。

面具变压器。在表 3 中，我们消去了掩模变压器的有效性。第一行是一个简单的逐体素基线，它使用 mlp 堆栈直接预测来自稀疏体素解码器的语义。引入带交叉注意的掩模转换器（这在 MaskFormer 和 Mask3D 中是常见的做法）可以提高 1.7 RayIoU 的性能，但不可避免地会减慢推理速度，因为它会关注图像中的所有位置。因此，为了加快密集的交叉注意管道，我们采用了稀疏采样机制，使推理时间减少了 50%。通过进一步引入预测遮罩来指导采样点的生成，我们最终实现了 29.2 RayIoU 和 24 FPS。

有限的体素是否足以覆盖场景？在本研究中，我们深入研究了体素稀疏度对最终性能的影响。为了研究这一点，我们系统地减少了图 8 (a) 中的 k 值。从适度的 16k 值开始，我们观察到当 k 设置为 32k 48k 时，性能达到最佳，这仅占密集体素总数的 5% 7.5% （200×200×16 = 640000）。令人惊讶的是，进一步增加 k 并没有产生任何性能改进；相反，它引入了噪音。因此，我们的研究结果表明，约 5% 的稀疏度水平就足够了。不断增加密度会降低精度和速度。

通过 top-k 进行修剪是简单有效的，但它与特定的数据集有关。在现实世界中，我们可以用阈值法代替 top-k。得分低于给定阈值（例如，0.7）的体素将被修剪。阈值分割的性能与 top-k 相似（见图 8 (b)），并且具有推广到不同场景的能力。



(a) Pruning by top-k (b) Pruning by thresholding (c) Number of frames

图 8: 体素稀疏和时间建模的消融。(a) 当 k 设置为 32000（5% 稀疏度）时，性能最优。(b) Top-k 也可以用阈值法代替，例如，得分低于某一阈值的体素将被修剪。(c) 随着帧数的增加，性能继续提高，但在 12 帧后开始饱和。

时序建模。在图 8 (c) 中，我们验证了时间融合的有效性。我们可以看到，SparseOcc 的时间建模是非常有效的，性能随着帧数的增加而稳步提高。性能在 12 帧时达到峰值，然后达到饱和。然而，由于采样点需要与每一帧交互，推理速度下降很快。

5.4 更多的研究

戴着可见面具训练的效果。有趣的是，我们观察到一个奇怪的现象。在传统的体素级 mIoU 度量下，密集方法可以在训练过程中忽略不可见体素，从而显著受益。这些不可见体素由 Occ3D-nuScenes 数据集提供的二进制可见掩码表示。然而，我们发现这种策略实际上损害了我们的新 RayIoU 指标下的性能。例如，我们训练 BEVFormer 的两个变体：一个在训练期间使用可见遮罩，另一个不使用。如表 4 所示，前者在基于体素的 mIoU 上得分比后者高 15 分，但在 RayIoU 上得分低 1 分。在 FB-Occ 上也观察到这种现象。

为了进一步探讨这一点，我们在表 4 中展示了每个类的 RayIoU。该表显示，使用可视遮罩进行训练可以提高大多数前景类（如公共汽车、自行车和卡车）的性能。然而，它会对可驾驶的路面和地形等背景类产生负面影响。

这一观察结果提出了一个进一步的问题：为什么背景类别的性能会下降？为了解决这个问题，我们提供了图 9 中 FB-Occ 预测可驾驶表面的深度误差和高度图的视觉比较，在训练期间使用和不使用可见遮罩。该图表明，使用可见遮罩的训练导致更厚和更高的地面预测，导致遥远区域的深度误差很大。相反，没有可见掩模的模型预测深度的准确性更高。

Table 4: To verify the effect of the visible mask, wo provide per-class RayIoU of BEVFormer and FB-Occ. † uses the visible mask during training. We find that training with the visible mask hurts the performance of background classes such as drivable surface, terrian and sidewalk.

Method	mIoU	RayIoU	Per-class RayIoU																
			others	barrier	bicycle	bus	car	cons. veh.	motor.	pedes.	tfc. cone	trailer	truck	drv. surf.	other flat	sidewalk	terrain	manmade	vegetation
BEVFormer	23.7	**33.7**	5.0	42.2	18.2	**55.2**	**57.1**	**22.7**	21.3	31.0	**27.1**	**30.7**	49.4	**58.4**	**30.4**	**29.4**	**31.7**	36.3	26.5
BEVFormer †	**39.2**	32.4	**6.4**	**44.8**	**24.0**	**55.2**	56.7	21.0	**29.8**	**33.5**	26.8	27.9	**49.5**	45.8	18.7	22.4	18.5	**39.1**	**29.8**
FB-Occ	27.9	**35.6**	**10.5**	44.8	25.6	55.6	51.7	22.6	27.2	**34.3**	**30.3**	23.7	44.1	**65.5**	**33.3**	**31.4**	**32.5**	**39.6**	**33.3**
FB-Occ †	**39.1**	33.5	5.0	**44.9**	**26.2**	**59.7**	**55.1**	**27.9**	**29.1**	**34.3**	29.6	**29.1**	**50.5**	44.4	22.4	21.5	19.5	39.3	31.1

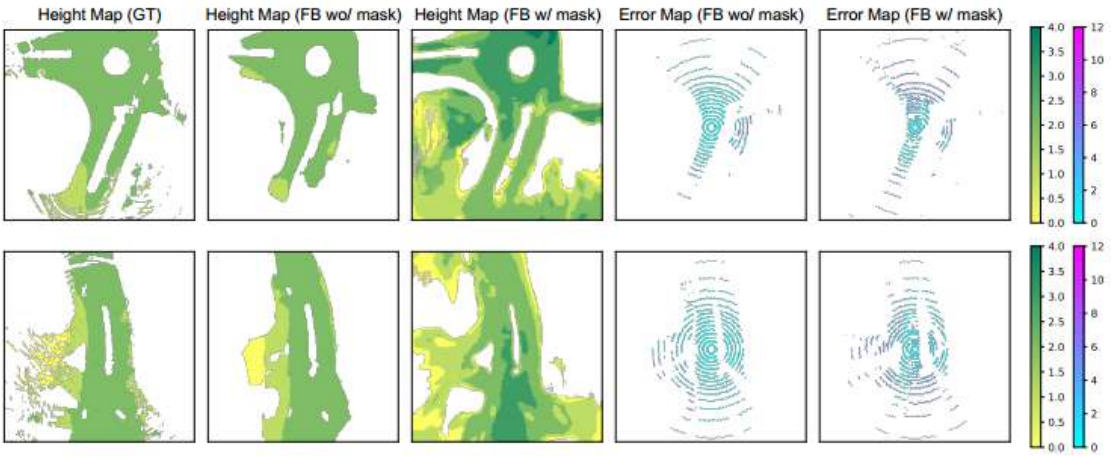


图 9: 为什么在训练期间使用可见遮罩时，背景类（例如可驱动表面）的性能会下降？我们提供了 FB-Occ 预测的可驾驶表面的可视化。这里，“FB w/ mask”和“FB wo/ mask”分别表示有和没有可见掩码的训练。我们观察到，“FB w/ mask”倾向于预测更高和更厚的路面，导致沿射线的深度误差显著。相比之下，“FB wo/ mask”预测的路面既准确又一致。

Table 5: Panoptic occupancy prediction performance on Occ3D-nuScenes.

Method	Backbone	Input Size	Epoch	RayPQ	RayPQ_{1m}	RayPQ_{2m}	RayPQ_{4m}
SparseOcc	R50	704×256	24	**14.1**	10.2	14.5	17.6

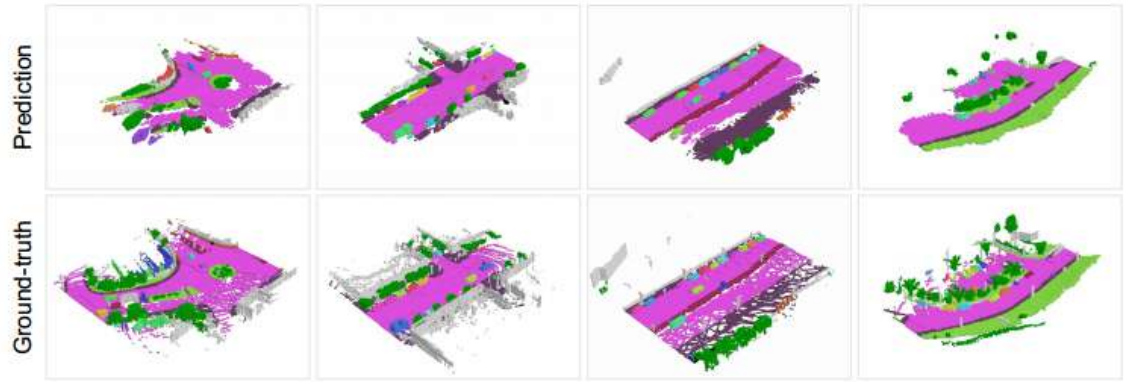


图 10: 全景占位预测。不同的实例用颜色来区分。我们的模型可以同时捕获细粒度的物体和道路结构。

从这些观察中，我们得出了一些有价值的见解：在训练过程中忽略不可见的体素，通过解决未扫描体素的模糊标记问题，有利于前景类。然而，这也降低了深度估计的准确性，因为模型倾向于预测更厚更近的表面。我们希望我们的发现对未来的研究有益。

展示全景的入住率。然后，我们证明 SparseOcc 可以很容易地扩展到全景占用预测，这是一项来自全景分割的任务，它不仅可以将图像分割为语义上有意义的区域，还可以检测和区分单个实例。与全视分割相比，全视占位预测需要模型具有几何感知才能构建用于分割的三维场景。通过额外地向掩模转换器引入实例查询，我们使用仅摄像头输入无缝地实现了第一个完全稀疏的全景占用预测框架。

首先，我们利用三维目标检测任务的地面真值边界框生成全景占用地面真值；具体来说，我们定义了 8 个实例类别（包括汽车、卡车、工程车、公共汽车、拖车、摩托车、自行车、行人）和 10 个人员类别（包括地形、人造、植被等）。每个实例段是通过基于现有的语义占用基准（如 Occ3D-nuScenes）对边界框内的体素进行分组来识别的。

然后，我们基于众所周知的 panoptic quality (PQ)[22] 度量来设计 RayPQ，它被定

义为分割质量（SQ）和识别质量（RQ）的乘法：其中真正（TP）的定义与 RayIoU

$$\mathrm{PQ} = \underbrace{\frac{\sum_{(p,g)\in TP} \mathrm{IoU}(p,g)}{|TP|}}_{\text{segmentation quality (SQ)}} \times \underbrace{\frac{|TP|}{|TP| + \frac{1}{2}|FP| + \frac{1}{2}|FN|}}_{\text{recognition quality (RQ)}}, \tag{4}$$

中的定义相同。预测值 p 与真实值 g 之间的 IoU 阈值设为 0.5。

在表 5 中，我们报告了 SparseOcc 在全景占用基准上的性能。与 RayIoU 类似，我们在三个距离阈值下计算 RayPQ: 1、2 和 4 米。SparseOcc 实现了 14.1 的平均 RayPQ。可视化结果如图 10 所示。

5.5 局限性

累计错误。为了实现一个完全稀疏的架构，我们在早期阶段丢弃了大量的空体素。但是，错误丢弃的空体素在后续阶段无法恢复。此外，掩模变压器的预测被限制在稀疏体素解码器预测的空间内。在这个不可靠的空间中，一些地面真实实例没有出现，导致掩模变压器的训练不足。

6 总结

在本文中，我们提出了一种完全稀疏占用网络，命名为 SparseOcc，它不依赖于密集的 3D 特征，也不具有稀疏到密集和全局关注操作。我们还创建了 RayIoU，这是一种用于占用评估的光线级别度量，消除了之前度量的不一致性缺陷。实验表明，SparseOcc 在 Occ3D-nuScenes 数据集上的速度和精度都达到了最先进的性能。我们希望这一令人兴奋的结果能够吸引更多的人关注全稀疏的 3D 占用模式。

7 致谢

我们感谢匿名审稿人的建议，使这项工作更好。国家重点研发计划项目（No. 2022ZD0160900）、国家自然科学基金项目（No. 62076119, No. 61921006）、中央高校基本科研业务费

专项资金（No. 020214380119）、软件新技术与产业化协同创新中心资助。

8 参考文献

[1] Tesla AI Day. https://www.youtube.com/watch?v=j0z4FweCy4M (2021)
[2] Bozic, A., Palafox, P., Thies, J., Dai, A., Nießner, M.: Transformerfusion: Monocular rgb scene reconstruction using transformers. In: NeurIPS (2021)
[3] Caesar, H., Bankiti, V., Lang, A.H., Vora, S., Liong, V.E., Xu, Q., Krishnan, A., Pan, Y., Baldan, G., Beijbom, O.: nuscenes: A multimodal dataset for autonomous driving. In: CVPR (2020)
[4] Cao, A.Q., de Charette, R.: Monoscene: Monocular 3d semantic scene completion. In: CVPR (2022)
[5] Cheng, B., Misra, I., Schwing, A.G., Kirillov, A., Girdhar, R.: Masked-attention mask transformer for universal image segmentation. In: CVPR (2022)
[6] Cheng, B., Schwing, A., Kirillov, A.: Per-pixel classification is not all you need for semantic segmentation. In: NeurIPS (2021)
[7] Choy, C., Gwak, J., Savarese, S.: 4d spatio-temporal convnets: Minkowski convolutional neural networks. In: Proceedings of the IEEE/CVF conference on computer vision and pattern recognition. pp. 3075–3084 (2019)
[8] Curless, B., Levoy, M.: A volumetric method for building complex models from range images. In: SIGGRAPH (1996)
[9] Ding, Y., Huang, L., Zhong, J.: Multi-scale occ: 4th place solution for Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition 2023 3d occupancy prediction challenge. arXiv preprint arXiv:2306.11414 (2023)
[10] Feng, Z., Yang, L., Guo, P., Li, B.: Cvrecon: Rethinking 3d geometric feature learning for neural reconstruction. In: ICCV (2023)
[11] Gan, W., Mo, N., Xu, H., Yokoya, N.: A comprehensive framework for 3d occupancy estimation in autonomous driving. IEEE Transactions on Intelligent Vehicles pp. 1–19 (2024)
[12] Gao, Z., Wang, L., Han, B., Guo, S.: Adamixer: A fast-converging query-based object detector. In: CVPR (2022)
[13] He, K., Chen, X., Xie, S., Li, Y., Dollár, P., Girshick, R.: Masked autoencoders are scalable vision learners. In: CVPR (2022)

[14] He, K., Zhang, X., Ren, S., Sun, J.: Deep residual learning for image recognition. In: CVPR (2016)
[15] Huang, J., Huang, G.: Bevdet4d: Exploit temporal cues in multi-camera 3d object detection. arXiv preprint arXiv:2203.17054 (2022)
[16] Huang, J., Huang, G., Zhu, Z., Du, D.: Bevdet: High-performance multi-camera 3d object detection in bird-eye-view. arXiv preprint arXiv:2112.11790 (2021)
[17] Huang, L., Li, Z., Sima, C., Wang, W., Wang, J., Qiao, Y., Li, H.: Leveraging vision-centric multi-modal expertise for 3d object detection. In: NeurIPS (2024)
[18] Huang, L., Wang, H., Zeng, J., Zhang, S., Cao, L., Ji, R., Yan, J., Li, H.: Geometric-aware pretraining for vision-centric 3d object detection. arXiv preprint arXiv:2304.03105 (2023)
[19] Huang, Y., Zheng, W., Zhang, Y., Zhou, J., Lu, J.: Tri-perspective view for vision-based 3d semantic occupancy prediction. In: CVPR (2023)
[20] Jia, X., De Brabandere, B., Tuytelaars, T., Gool, L.V.: Dynamic filter networks. In: NeurIPS (2016) 13 [21] Khurana, T., Hu, P., Held, D., Ramanan, D.: Point cloud forecasting as a proxy for 4d occupancy forecasting. In: CVPR (2023)
[22] Kirillov, A., He, K., Girshick, R., Rother, C., Dollár, P.: Panoptic segmentation. In: CVPR (2019)
[23] Lang, A.H., Vora, S., Caesar, H., Zhou, L., Yang, J., Beijbom, O.: Pointpillars: Fast encoders for object detection from point clouds. In: CVPR (2019)
[24] Li, H., Li, Y., Wang, H., Zeng, J., Cai, P., Xu, H., Lin, D., Yan, J., Xu, F., Xiong, L., Wang, J., Zhu, F., Yan, K., Xu, C., Wang, T., Mu, B., Ren, S., Peng, Z., Qiao, Y.: Open-sourced data ecosystem in autonomous driving: the present and future. arXiv preprint arXiv:2312.03408 (2023)
[25] Li, H., Sima, C., Dai, J., Wang, W., Lu, L., Wang, H., Zeng, J., Li, Z., Yang, J., Deng, H., et al.: Delving into the devils of bird' s-eye-view perception: A review, evaluation and recipe. IEEE TPAMI (2023)
[26] Li, Y., Yu, Z., Choy, C., Xiao, C., Alvarez, J.M., Fidler, S., Feng, C., Anandkumar, A.: Voxformer: Sparse voxel transformer for camera-based 3d semantic scene completion. In: CVPR (2023)
[27] Li, Z., Wang, W., Li, H., Xie, E., Sima, C., Lu, T., Qiao, Y., Dai, J.: Bevformer: Learning bird' s-eye-view representation from multi-camera images via spatiotemporal transformers. In: ECCV (2022)
[28] Li, Z., Yu, Z., Austin, D., Fang, M., Lan, S., Kautz, J., Alvarez, J.M.: Fb-occ: 3d occupancy prediction based on forward-backward view transformation. arXiv preprint arXiv:2307.01492 (2023)

[29] Lin, T.Y., Dollár, P., Girshick, R., He, K., Hariharan, B., Belongie, S.: Feature pyramid networks for object detection. In: CVPR (2017)
[30] Lin, X., Lin, T., Pei, Z., Huang, L., Su, Z.: Sparse4d: Multi-view 3d object detection with sparse spatial-temporal fusion. arXiv preprint arXiv:2211.10581 (2022)
[31] Liu, H., Lu, T., Xu, Y., Liu, J., Li, W., Chen, L.: Camliflow: Bidirectional camera-lidar fusion for joint optical flow and scene flow estimation. In: CVPR (2022)
[32] Liu, H., Lu, T., Xu, Y., Liu, J., Wang, L.: Learning optical flow and scene flow with bidirectional camera-lidar fusion. arXiv preprint arXiv:2303.12017 (2023)
[33] Liu, H., Teng, Y., Lu, T., Wang, H., Wang, L.: Sparsebev: High-performance sparse 3d object detection from multi-camera videos. In: ICCV (2023)
[34] Liu, Y., Wang, T., Zhang, X., Sun, J.: PETR: position embedding transformation for multi-view 3d object detection. In: ECCV (2022)
[35] Liu, Y., Yan, J., Jia, F., Li, S., Gao, Q., Wang, T., Zhang, X., Sun, J.: Petrv2: A unified framework for 3d perception from multi-camera images. arXiv preprint arXiv:2206.01256 (2022)
[36] Loshchilov, I., Hutter, F.: Decoupled weight decay regularization. In: ICLR (2019)
[37] Mescheder, L., Oechsle, M., Niemeyer, M., Nowozin, S., Geiger, A.: Occupancy networks: Learning 3d reconstruction in function space. In: CVPR (2019)
[38] Milletari, F., Navab, N., Ahmadi, S.A.: V-net: Fully convolutional neural networks for volumetric medical image segmentation. In: 3DV (2016)
[39] Murez, Z., Van As, T., Bartolozzi, J., Sinha, A., Badrinarayanan, V., Rabinovich, A.: Atlas: End-to-end 3d scene reconstruction from posed images. In: ECCV (2020)
[40] Pan, M., Liu, J., Zhang, R., Huang, P., Li, X., Liu, L., Zhang, S.: Renderocc: Vision-centric 3d occupancy prediction with 2d rendering supervision. arXiv preprint arXiv:2309.09502 (2023)
[41] Paszke, A., Gross, S., Massa, F., Lerer, A., Bradbury, J., Chanan, G., Killeen, T., Lin, Z., Gimelshein, N., Antiga, L., et al.: Pytorch: An imperative style, high-performance deep learning library. In: NeurIPS (2019)
[42] Peng, S., Niemeyer, M., Mescheder, L., Pollefeys, M., Geiger, A.: Convolutional occupancy networks. In: ECCV (2020) 14 [43] Ronneberger, O., Fischer, P., Brox, T.: U-net: Convolutional networks for biomedical image segmentation. In: MICCAI (2015)
[44] Schult, J., Engelmann, F., Hermans, A., Litany, O., Tang, S., Leibe, B.: Mask3d:

Mask transformer for 3d semantic instance segmentation. In: ICRA (2023)
[45] Sima, C., Tong, W., Wang, T., Chen, L., Wu, S., Deng, H., Gu, Y., Lu, L., Luo, P., Lin, D., Li, H.: Scene as occupancy. In: ICCV (2023)
[46] Stier, N., Rich, A., Sen, P., Höllerer, T.: Vortx: Volumetric 3d reconstruction with transformers for voxelwise view selection and fusion. In: 3DV (2021)
[47] Sun, J., Xie, Y., Chen, L., Zhou, X., Bao, H.: Neuralrecon: Real-time coherent 3d reconstruction from monocular video. In: CVPR (2021)
[48] Takikawa, T., Litalien, J., Yin, K., Kreis, K., Loop, C., Nowrouzezahrai, D., Jacobson, A., McGuire, M., Fidler, S.: Neural geometric level of detail: Real-time rendering with implicit 3d shapes. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 11358–11367 (2021)
[49] Takmaz, A., Fedele, E., Sumner, R.W., Pollefeys, M., Tombari, F., Engelmann, F.: Openmask3d: Open-vocabulary 3d instance segmentation. arXiv preprint arXiv:2306.13631 (2023)
[50] Tian, X., Jiang, T., Yun, L., Wang, Y., Wang, Y., Zhao, H.: Occ3d: A large-scale 3d occupancy prediction benchmark for autonomous driving. In: NeurIPS Datasets and Benchmarks (2023)
[51] Tolstikhin, I.O., Houlsby, N., Kolesnikov, A., Beyer, L., Zhai, X., Unterthiner, T., Yung, J., Steiner, A., Keysers, D., Uszkoreit, J., et al.: Mlp-mixer: An all-mlp architecture for vision. In: NeurIPS (2021)
[52] Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A.N., Kaiser, Ł., Polosukhin, I.: Attention is all you need. In: NeurIPS (2017)
[53] Wang, S., Liu, Y., Wang, T., Li, Y., Zhang, X.: Exploring object-centric temporal modeling for efficient multi-view 3d object detection. arXiv preprint arXiv:2303.11926 (2023)
[54] Wang, X., Zhu, Z., Xu, W., Zhang, Y., Wei, Y., Chi, X., Ye, Y., Du, D., Lu, J., Wang, X.: Openoccupancy: A large scale benchmark for surrounding semantic occupancy perception. arXiv preprint arXiv:2303.03991 (2023)
[55] Wang, Y., Guizilini, V.C., Zhang, T., Wang, Y., Zhao, H., Solomon, J.: Detr3d: 3d object detection from multi-view images via 3d-to-2d queries. In: CoRL (2022)
[56] Wang, Y., Chen, Y., Liao, X., Fan, L., Zhang, Z.: Panoocc: Unified occupancy representation for camera-based 3d panoptic segmentation. arXiv preprint arXiv:2306.10013 (2023)
[57] Wei, Y., Zhao, L., Zheng, W., Zhu, Z., Zhou, J., Lu, J.: Surroundocc: Multi-camera 3d occupancy prediction for autonomous driving. In: ICCV (2023)
[58] Yang, Z., Chen, L., Sun, Y., Li, H.: Visual point cloud forecasting enables

scalable autonomous driving. In: CVPR (2024)
[59] Yang, Z., Jiang, L., Sun, Y., Schiele, B., Jia, J.: A unified query-based paradigm for point cloud understanding. In: ICCV (2022)
[60] Yang, Z., Sun, Y., Liu, S., Jia, J.: 3dssd: Point-based 3d single stage object detector. In: CVPR (2020)
[61] Yang, Z., Sun, Y., Liu, S., Shen, X., Jia, J.: Std: Sparse-to-dense 3d object detector for point cloud. In: ICCV (2019)
[62] Yang, Z., Zhou, Y., Chen, Z., Ngiam, J.: 3d-man: 3d multi-frame attention network for object detection. In: ICCV (2021)
[63] Yin, T., Zhou, X., Krahenbuhl, P.: Center-based 3d object detection and tracking. In: CVPR (2021)
[64] Zhu, X., Su, W., Lu, L., Li, B., Wang, X., Dai, J.: Deformable detr: Deformable transformers for end-to-end object detection. arXiv preprint arXiv:2010.04159 (2020)

Fully Sparse 3D Occupancy Prediction

Haisong Liu[1,2*], **Yang Chen**[1*], **Haiguang Wang**[1], **Zetong Yang**[2], **Tianyu Li**[2], **Jia Zeng**[2], **Li Chen**[2], **Hongyang Li**[2], **Limin Wang**[1,2,✉]

[1]Nanjing University [2]Shanghai AI Lab

https://github.com/MCG-NJU/SparseOcc

Abstract

Occupancy prediction plays a pivotal role in autonomous driving. Previous methods typically construct dense 3D volumes, neglecting the inherent sparsity of the scene and suffering from high computational costs. To bridge the gap, we introduce a novel fully sparse occupancy network, termed SparseOcc. SparseOcc initially reconstructs a sparse 3D representation from camera-only inputs and subsequently predicts semantic/instance occupancy from the 3D sparse representation by sparse queries. A mask-guided sparse sampling is designed to enable sparse queries to interact with 2D features in a fully sparse manner, thereby circumventing costly dense features or global attention. Additionally, we design a thoughtful ray-based evaluation metric, namely RayIoU, to solve the inconsistency penalty along the depth axis raised in traditional voxel-level mIoU criteria. SparseOcc demonstrates its effectiveness by achieving a RayIoU of 34.0, while maintaining a real-time inference speed of 17.3 FPS, with 7 history frames inputs. By incorporating more preceding frames to 15, SparseOcc continuously improves its performance to 35.1 RayIoU without bells and whistles.

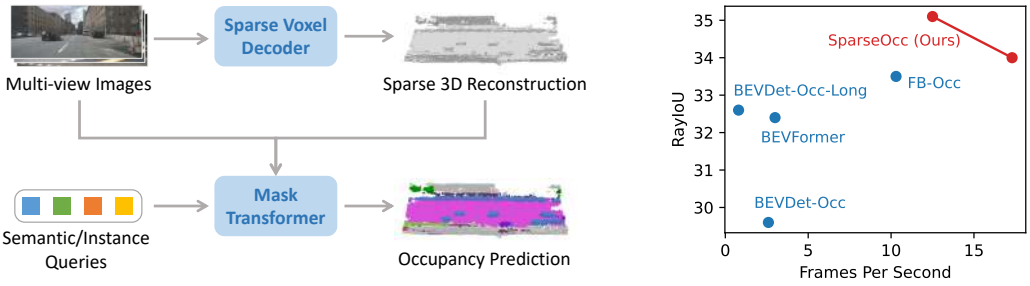
1 Introduction

Vision-centric 3D occupancy prediction [1] focuses on partitioning 3D scenes into structured grids from visual images. Each grid is assigned a label indicating if it is occupied or not. This task offers more geometric details than 3D object detection and produces an alternative representation to LiDAR-based perception [63, 23, 60, 61, 62, 31, 32].

Existing methods [27, 16, 57, 45, 28] typically construct dense 3D features yet suffer from computational overhead (*e.g.*, $2 \sim 3$ FPS on the Tesla A100 GPU). However, dense representations are not necessary for occupancy prediction. We statistic the geometry sparsity and find that more than 90% of the voxels are empty. This manifests a large room in occupancy prediction acceleration by exploiting the sparsity. Some works [26, 19] explore the sparsity of 3D scenes, but they still rely on sparse-to-dense modules for dense predictions. This inspires us to seek a pure sparse occupancy network without any dense design.

In this paper, we propose SparseOcc, the first fully sparse occupancy network. As depicted in Fig. 1 (a), SparseOcc includes two steps. First, it leverages a *sparse voxel decoder* to reconstruct the sparse geometry of a scene in a coarse-to-fine manner. This only models non-free regions, saving computational costs significantly. Second, we design a *mask transformer* with sparse semantic/instance queries to predict masks and labels of segments from the sparse space. The mask transformer not only improves performance on semantic occupancy but also paves the way for panoptic occupancy. A *mask-guided sparse sampling* is designed to achieve sparse cross-attention in the mask transformer.

*: Equal contribution.
✉: Corresponding author.



(a) Overview of SparseOcc (b) Performance Comparison

Figure 1: (a) SparseOcc reconstructs a sparse 3D representation from camera-only inputs by a sparse voxel decoder, and then estimates the mask and label of each segment via a set of sparse queries. (b) Performance comparison on the validation split of Occ3D-nuScenes. FPS is measured on a Tesla A100 with the PyTorch `fp32` backend.

As such, our SparseOcc fully exploits the sparse property and gets rid of any dense design like dense 3D features, sparse-to-dense modules, and global attention.

Besides, we notice flaws in popular voxel-level mean Intersection-over-Union (mIoU) metrics for occupancy evaluation and further design a ray-level evaluation, RayIoU, as the solution. The mIoU criterion is an ill-posed formulation given the ambiguous labeling of unscanned voxels. Previous methods[50] relieve this issue by only evaluating observed areas but raise extra issues in inconsistency penalty along depths. Instead, RayIoU addresses the two aforementioned issues simultaneously. It evaluates predicted 3D occupancy volume by retrieving depth and category predictions of designated rays. To be specific, RayIoU casts query rays into predicted 3D volumes and decides true positive predictions as the ray with the correct distance and class of its first touched occupied voxel grid. This formulates a more fair and reasonable criterion.

Thanks to the sparsity design, SparseOcc achieves 34.0 RayIoU on Occ3D-nuScenes [50], while maintaining a real-time inference speed of 17.3 FPS (Tesla A100, PyTorch `fp32` backend), with 7 history frames inputs. By incorporating more preceding frames to 15, SparseOcc continuously improves its performance to 35.1 RayIoU, achieving state-of-the-art performance without bells and whistles. The comparison between SparseOcc with previous methods in terms of performance and efficiency is shown in Fig. 1 (b).

We summarize our contributions as follows:

1. We propose SparseOcc, the first fully sparse occupancy network without any time-consuming dense designs. It achieves 34.0 RayIoU on Occ3D-nuScenes benchmark with an real-time inference speed of 17.3 FPS.
2. We present RayIoU, a ray-wise criterion for occupancy evaluation. By querying rays to 3D volume, it solves the ambiguous penalty issue for unscanned free voxels and the inconsistent depth penalty issue in the mIoU metric.

2 Related Work

Camera-based 3D Occupancy Prediction. The occupancy network is originally proposed by Mescheder *et al*. [37, 42], focusing on continuous object representations in 3D space. Recent variations [1, 4, 45, 50, 54, 56, 11, 58] mostly draw inspiration from Bird's Eye View (BEV) perception [25, 24, 27, 16, 15, 18, 17, 55, 34, 35, 30, 33, 53, 59] and predicts voxel-level semantic information from image inputs. For instance, MonoScene [4] estimates occupancy through a 2D and a 3D UNet [43] connected by a sight projection module. SurroundOcc [57] proposes a coarse-to-fine architecture. However, the large number of voxel queries is computationally heavy. TPVFormer [19] proposes tri-perspective view representations to supplement vertical structural information, but this inevitably leads to information loss. VoxFormer [26] initializes sparse queries based on monocular depth prediction. Nevertheless, VoxFormer is not fully sparse as it still requires a sparse-to-dense

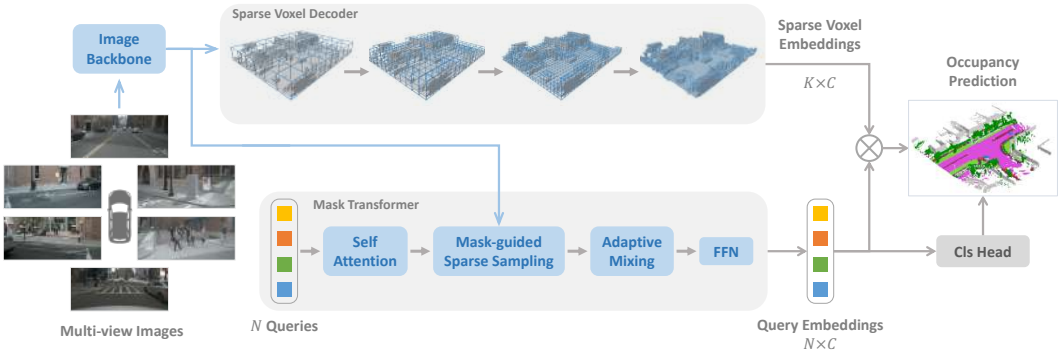


Figure 2: SparseOcc is a fully sparse architecture since it neither relies on dense 3D feature, nor has sparse-to-dense and global attention operations. The sparse voxel decoder reconstructs the sparse geometry of the scene, consisting of K voxels ($K \ll W \times H \times D$). The mask transformer then uses N sparse queries to predict the mask and label of each segment. SparseOcc can be easily extended to panoptic occupancy by replacing the semantic queries with instance queries.

MAE [13] module to complete the scene. Some methods emerged in the CVPR 2023 occupancy challenge [28, 40, 9], but none of them exploits a fully sparse design. In this paper, we make the first step to explore the fully sparse architecture for 3D occupancy prediction from camera-only inputs.

Sparse Architectures for 3D Vision. Sparse architectures find widespread adoption in LiDAR-based reconstruction [48] and perception [7, 63, 60, 61], leveraging the inherent sparsity of point clouds. However, when it comes to vision-to-3D tasks, a direct adaptation is not feasible due to the absence of point cloud inputs. A prior work, SparseBEV [33], proposes a fully sparse architecture for camera-based 3D object detection. Nevertheless, directly adapting this approach is non-trivial because 3D object detection focuses on a sparse set of objects, whereas 3D occupancy requires dense predictions for each voxel. Consequently, designing a fully sparse architecture for 3D occupancy prediction remains a challenging task.

End-to-end 3D Reconstruction from Posed Images. As a related task to 3D occupancy prediction, 3D reconstruction recovers the 3D geometry from multiple posed images. Recent methods focus on more compact and efficient end-to-end 3D reconstruction pipelines [39, 47, 2, 46, 10]. Atlas [39] extracts features from multi-view input images and maps them to 3D space to construct the truncated signed distance function [8]. NeuralRecon [47] directly reconstructs local surfaces as sparse TSDF volumes and uses a GRU-based TSDF fusion module to fuse features from previous fragments. VoRTX [46] utilizes transformers to address occlusion issues in multi-view images.

Mask Transformer. Recently, unified segmentation models have been widely studied to handle semantic and instance segmentation concurrently. Cheng *et al*. first propose MaskFormer [6] for unified segmentation in terms of model architecture, loss functions, and training strategies. Mask2Former [5] then introduces masked attention, with restricted receptive fields on instance masks, for better performance. Later on, Mask3D [44] successfully extends the mask transformer for point cloud segmentation with state-of-the-art performance. OpenMask3D [49] further achieves the open-vocabulary 3D instance segmentation task and proposes a model for zero-shot 3D segmentation.

3 SparseOcc

SparseOcc is a vision-centric occupancy model that only requires camera inputs. As shown in Fig. 2, SparseOcc has three modules: an image encoder consisting of an image backbone and FPN [29] to extract 2D features from multi-view images; a sparse voxel decoder (Sec. 3.1) to predict sparse class-agnostic 3D occupancy with correlated embeddings from the image features; a mask transformer decoder (Sec 3.2) to distinguish semantics and instances in the sparse 3D space.

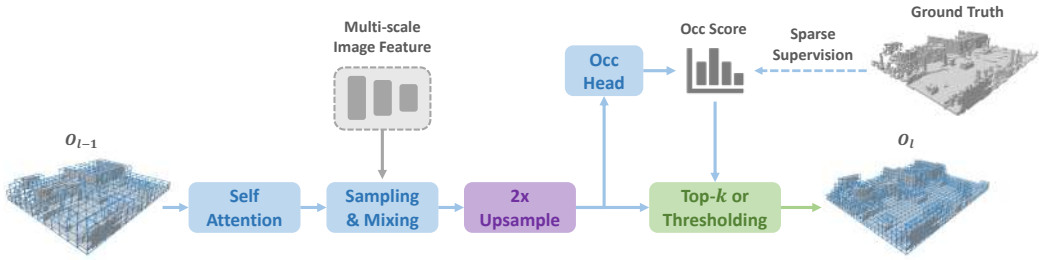


Figure 3: The sparse voxel decoder employs a coarse-to-fine pipeline with three layers. Within each layer, we utilize a transformer-like architecture for 3D-2D interaction. At the end of every layer, the voxel resolution is upsampled by a factor of 2×, and probabilities of voxel occupancy are estimated.

3.1 Sparse Voxel Decoder

Since 3D occupancy ground truth [50, 45, 57, 54] is a dense volume with dimensions $W \times H \times D$ (*e.g.*, 200×200×16), existing methods typically build a dense 3D feature of shape $W \times H \times D \times C$, but suffer from computational overhead. In this paper, we argue that such dense representation is not necessary for occupancy prediction. As in our statistics, we find that over 90% of the voxels in the scene are free. This motivates us to explore a sparse 3D representation that only models the non-free areas of the scene, thereby saving computational resources.

Overall architecture. Our designed sparse voxel decoder is shown in Fig. 3. In general, it follows a coarse-to-fine structure but only models the non-free regions. The decoder starts from a set of coarse voxel queries equally distributed in the 3D space (*e.g.*, 25×25). In each layer, we first upsample each voxel by 2×, *e.g.*, a voxel with size d will be upsampled into 8 voxels with size $\frac{d}{2}$. Next, we estimate an occupancy score for each voxel and conduct pruning to remove useless voxel grids. Here we have two approaches for pruning: one is based on a threshold (*e.g.*, only keeps score > 0.5); the other is by top-k selection. In our implementation, we simply keep voxels with top-k occupancy scores for training efficiency. k is a dataset-related parameter, obtained by counting the maximum number of non-free voxels in each sample at different resolutions. The voxel tokens after pruning will serve as the input for the next layer.

Detailed design. Within each layer, we use a transformer-like [52] architecture to handle voxel queries. The concrete architecture is inspired by SparseBEV [33], a detection method using a sparse scheme. To be specific, in layer l with K_{l-1} voxel queries described by 3D locations and a C-dim content vector, we first use self-attention to aggregate local and global features for those query voxels. Then, a linear layer is used to generate 3D sampling offsets $\{(\Delta x_i, \Delta y_i, \Delta z_i)\}$ for each voxel query from the associated content vector. These sampling offsets are utilized to transform voxel queries to obtain reference points in global coordinates. We finally project those sampled reference points to multi-view image space for integrating image features by adaptive mixing [12, 51, 20]. In summary, our approach differs from SparseBEV by shifting the query formulation from pillars to 3D voxels. Other components such as self attention, adaptive sampling and mixing are directly borrowed.

Temporal modeling. Previous dense occupancy methods [27, 16] typically warp the history BEV/3D feature to the current timestamp, and use deformable attention [64] or 3D convolutions to fuse temporal information. However, this approach is not directly applicable in our case due to the sparse nature of our 3D features. To handle this, we leverage the flexibility of the aforementioned global sampled reference points by warping them to previous timestamps to sample history multi-view image features. The sampled multi-frame features are stacked and aggregated by adaptive mixing so as for temporal modeling.

Supervision. We compute loss for the sparsified voxels from each layer. We use *binary cross entropy* (BCE) loss as the supervision, given that we are reconstructing a class-agnostic sparse occupancy space. Only the kept sparse voxels are supervised, while the discarded regions during pruning in earlier stages are ignored.

Moreover, due to the severe class imbalance, the model can be easily dominated by categories with a large proportion, such as the ground, thereby ignoring other important elements in the scene, such as cars, people, etc. Therefore, voxels belonging to different classes are assigned with different loss weights. For example, voxels belonging to class c are assigned with a loss weight of:

$$w_c = \frac{\sum_{i=1}^{C} M_i}{M_c}, \tag{1}$$

where M_i is the number of voxels belonging to the i-th class in ground truth.

3.2 Mask Transformer

Our mask transformer is inspired by Mask2Former [5], which uses N sparse semantic/instance queries decoupled by binary mask queries $\mathbf{Q}_m \in [0,1]^{N \times K}$ and content vectors $\mathbf{Q}_c \in \mathbb{R}^{N \times C}$. The mask transformer consists of three steps: multi-head self attention (MHSA), mask-guided sparse sampling, and adaptive mixing. MHSA is used for the interaction between different queries as the common practice. Mask-guided sparse sampling and adaptive mixing are responsible for the interaction between queries and 2D image features.

Mask-guided sparse sampling. A simple baseline of mask transformer is to use the masked cross-attention module in Mask2Former. However, it attends to all positions of the key, with unbearable computations. Here, we design a simple alternative. We first randomly select a set of 3D points within the mask predicted by the previous $(l-1)$-th Transformer decoder layer. Then, we project those 3D points to multi-view images and extract their features by bilinear interpolation. Besides, our sparse sampling mechanism makes the temporal modeling easier by simply warping the sampling points (as done in the sparse voxel decoder).

Prediction. For class prediction, we apply a linear classifier with a sigmoid activation based on the query embeddings $\mathbf{Q}_c$. For mask prediction, the query embeddings are converted to mask embeddings by an MLP. The mask embeddings $\mathbf{M} \in \mathbb{R}^{Q \times C}$ have the same shape as query embeddings $\mathbf{Q}_c$ and are dot-producted with the sparse voxel embeddings $\mathbf{V} \in \mathbb{R}^{K \times C}$ to produce mask predictions. Thus, the prediction space of our mask transformer is constrained to the sparsified 3D space from the sparse voxel decoder, rather than the full 3D scene. The mask predictions will serve as the mask queries $\mathbf{Q}_m$ for the next transformer layer.

Supervision. The reconstruction result from the sparse voxel decoder may not be reliable, as it may overlook or inaccurately detect certain elements. Thus, supervising the mask transformer presents certain challenges since its predictions are confined within this unreliable space. In cases of missed detection, where some ground truth segments are absent in the predicted sparse occupancy, we opt to discard these segments to prevent confusion. As for inaccurately detected elements, we simply categorize them as an additional "no object" category.

Loss Functions. Following MaskFormer [6], we match the ground truth with the predictions using Hungarian matching. Focal loss L_{focal} is used for classification, while a combination of DICE loss [38] L_{dice} and BCE mask loss L_{mask} is used for mask prediction. Thus, the total loss of SparseOcc is composed of four parts:

$$L = L_{focal} + L_{mask} + L_{dice} + L_{occ}, \tag{2}$$

where L_{occ} is the loss of sparse voxel decoder.

4 Ray-level mIoU

4.1 Revisiting the Voxel-level mIoU

The Occ3D dataset [50], along with its proposed evaluation metrics, are widely recognized as benchmarks in this field. The ground truth occupancy is reconstructed from LiDAR point clouds, and the mean Intersection over Union (mIoU) at the voxel level is employed to assess performance. Due to factors such as distance and occlusion, the accumulated point clouds are not perfect. Some areas unscanned by LiDAR are marked as free, resulting in fragmented instances. This raises the problem

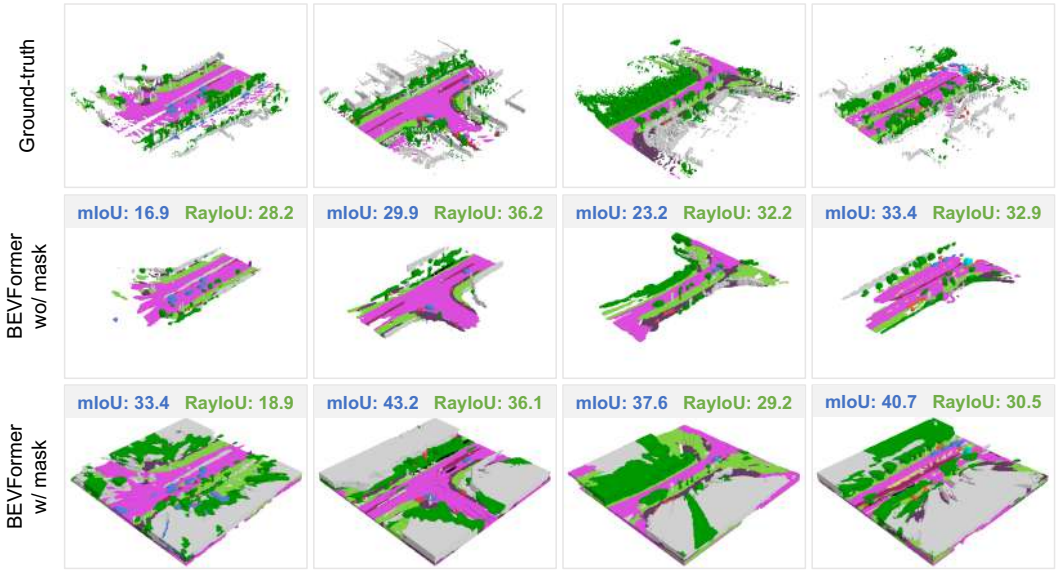


Figure 4: Visualization of the discrepancy between qualitative and quantitative results. We observe that training existing dense occupancy methods (*e.g.* BEVFormer) with a visible mask results in a thick surface, leading to an unreasonably inflated improvement in the current mIoU metrics. In contrast, our new RayIoU metrics provide a more accurate reflection of model performance.

of label inconsistency. To solve this problem, Occ3D uses a binary *visible mask* that indicates whether a voxel is observed in the current camera view. Only the observed voxels contribute to evaluation.

However, we found that solely calculating mIoU on the observed voxel positions remains vulnerable and can be hacked by *predicting a thicker surface*. Dense methods (*e.g.*, BEVFormer [27]) can easily achieve this by training with the visible mask. During training, the area behind the surface lacks supervision, causing the model to fill it with duplicated predictions, resulting in a thicker surface. As an example, consider BEVFormer, which generates a thick and noisy surface when trained with the visible mask (see Fig. 4). Despite this, its performance exhibits an unreasonably inflated improvement (+5∼15 mIoU) under the current evaluation protocol.

The misalignment between qualitative and quantitative results is caused by the inconsistent penalty along the depth direction. A toy example in Fig. 5 reveals several issues with the current metrics:

1. If the model fills all areas behind the surface, it inconsistently penalizes depth predictions. The model can obtain a higher IoU by filling all areas behind the surface and predicting a closer depth. This thick surface issue is very common in dense models trained with visible masks or 2D supervision.
2. If the predicted occupancy represents a thin surface, the penalty becomes overly strict. Even a deviation of just one voxel results in an IoU of zero.
3. The visible mask only considers the visible area at the current moment, reducing occupancy prediction to a depth estimation task and overlooking the scene completion ability.

4.2 Mean IoU by Ray Casting

To address the above issues, we propose a new evaluation metric: Ray-level mIoU (RayIoU for short). In RayIoU, the set elements are query rays rather than voxels. We emulate LiDAR behavior by projecting query rays into the predicted 3D occupancy volume. For each query ray, we compute the distance it travels before intersecting any surface and retrieve the corresponding class label. We then apply the same procedure to the ground-truth occupancy to obtain the ground-truth depth and class label. In case a ray does not intersect with any voxel present in the ground truth, it will be excluded from the evaluation process.

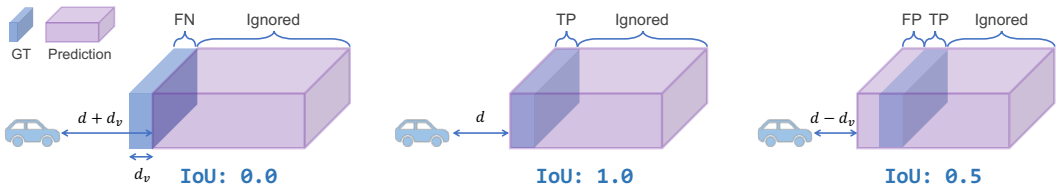


Figure 5: Illustration of inconsistent depth penalties caused by current metrics. Consider a scenario where we have a wall in front of us, with a ground-truth distance of d and a thickness of d_v. When the prediction has a thickness of $d_p \gg d_v$, we encounter an inconsistent penalty along depth. Specifically, if the predicted wall is d_v farther than the ground truth (total distance $d + d_v$), its IoU will be zero. Conversely, if the predicted wall is d_v closer than the ground truth (total distance $d - d_v$), the IoU remains at 0.5. This occurs because all voxels behind the surface are filled with duplicated predictions. Similarly, when the predicted depth is $d - 2d_v$, the resulting IoU is $\frac{1}{3}$, and so forth.

(a) Simulating LiDAR (b) Equal-distant resampling (c) Temporal casting

Figure 6: Covered area of RayIoU. (a) The raw LiDAR ray samples are unbalanced at different distances. (b) We resample the rays to balance the weight on distance. (c) To investigate the performance of scene completion, we propose evaluating occupancy in the visible area on a wide time span, by casting rays on visited waypoints.

As shown in Fig. 6 (a), the raw LiDAR rays in a real dataset tend to be unbalanced from near to far. Thus, we resample the rays to achieve a balanced distribution across different distances (Fig. 6 (b)). In the near field, we modify the ray channels to achieve equal-distant spacing when projected onto the ground plane. In the far field, we increase the angular resolution of the ray channels to ensure a more uniform data density across varying ranges. Moreover, our query ray can originate from the LiDAR position at the current, past, or future moments of the ego path. Temporal casting (Fig. 6 (c)) allows us to evaluate scene completion performance while maintaining a well-posed task.

A query ray is classified as a *true positive* (TP) if the class labels coincide and the L1 error between the ground-truth depth and the predicted depth is less than a certain threshold (*e.g.*, 2m). Let C be the number of classes, then RayIoU is calculated as follows:

$$\text{RayIoU} = \frac{1}{C} \sum_{c=1}^{C} \frac{\text{TP}_c}{\text{TP}_c + \text{FP}_c + \text{FN}_c}, \tag{3}$$

where TP_c, FP_c and FN_c correspond to the number of true positive, false positive, and false negative predictions for class c_i.

RayIoU addresses all three of the aforementioned problems:

1. Since the query ray calculates the distance to the first voxel it touches, the model cannot obtain a higher IoU by predicting a thicker surface.
2. RayIoU determines true positives based on a distance threshold, which mitigates the overly strict nature of voxel-level mIoU.
3. The query ray can originate from any position in the scene. This flexibility allows RayIoU to consider the model's scene completion ability, preventing the reduction of occupancy estimation to mere depth prediction.

Table 1: 3D occupancy prediction performance on Occ3D-nuScenes [50]. We use RayIoU to compare our SparseOcc with other methods. "8f" and "16f" mean fusing temporal information from 8 or 16 frames. SparseOcc outperforms all existing methods under a weaker setting.

Method	Backbone	Input Size	Epoch	RayIoU	$\text{RayIoU}_{1m, 2m, 4m}$			mIoU	FPS
BEVFormer (4f) [27]	R101	1600×900	24	32.4	26.1	32.9	38.0	39.2	3.0
RenderOcc [40]	Swin-B	1408×512	12	19.5	13.4	19.6	25.5	24.4	-
SimpleOcc [11]	R101	672×336	12	22.5	17.0	22.7	27.9	31.8	9.7
BEVDet-Occ (2f) [15]	R50	704×256	90	29.6	23.6	30.0	35.1	36.1	2.6
BEVDet-Occ-Long (8f)	R50	704×384	90	32.6	26.6	33.1	38.2	**39.3**	0.8
FB-Occ (16f) [28]	R50	704×256	90	33.5	26.7	34.1	39.7	39.1	10.3
SparseOcc (8f)	R50	704×256	24	34.0	28.0	34.7	39.4	30.1	**17.3**
SparseOcc (16f)	R50	704×256	24	35.1	29.1	35.8	40.3	30.6	12.5
SparseOcc (16f)	R50	704×256	48	**36.1**	**30.2**	**36.8**	**41.2**	30.9	12.5

5 Experiments

We evaluate our model on the Occ3D-nuScenes [50] dataset. Occ3D-nuScenes is based on the nuScenes [3] dataset, which consists of large-scale multimodal data collected from 6 surround-view cameras, 1 lidar and 5 radars. The dataset has 1000 videos in total and is split into 700/150/150 videos for training/validation/testing. Each video has roughly 20s duration and the key samples are annotated every 0.5s.

We use the proposed RayIoU to evaluate the semantic segmentation performance. The query rays originate from 8 LiDAR positions of the ego path. We calculate RayIoU under three distance thresholds: 1, 2 and 4 meters. The final ranking metric is averaged over these distance thresholds.

5.1 Implementation Details

We implement our model using PyTorch [41]. Following previous methods, we adopt ResNet-50 [14] as the image backbone. The mask transformer consists of 3 layers with shared weights across different layers. In our main experiments, we employ semantic queries where each query corresponds to a semantic class, rather than an instance. The ray casting module in RayIoU is implemented based on the codebase of [21].

During training, we use the AdamW [36] optimizer with a global batch size of 8. The initial learning rate is set to 2×10^{-4} and is decayed with cosine annealing policy. For all experiments, we train our models for 24 epochs. FPS is measured on a Tesla A100 GPU with the PyTorch `fp32` backend.

5.2 Main Results

In Tab. 1 and Fig. 1 (b), we compare SparseOcc with previous state-of-the-art methods on the validation split of Occ3D-nuScenes. Despite under a weaker setting (ResNet-50 [14], 8 history frames, and input image resolution of 704 × 256), SparseOcc significantly outperforms previous methods including FB-Occ, the winner of CVPR 2023 occupancy challenge, with many complicated designs including forward-backward view transformation, depth net, joint depth and semantic pre-training, and so on. SparseOcc achieves better results (+1.6 RayIoU) while being much faster and simpler than FB-Occ, which demonstrates the superiority of our solution.

We further provide qualitative results in Fig. 7. Both BEVDet-Occ and FB-Occ are dense methods and make many redundant predictions behind the surface. In contrast, SparseOcc discards over 90% of voxels while still effectively modeling the geometry of the scene and capturing fine-grained details.

5.3 Ablations

In this section, we conduct ablations on the validation split of Occ3D-nuScenes to confirm the effectiveness of each module. By default, we use the single frame version of SparseOcc as the baseline. The choice for our model is made **bold**.

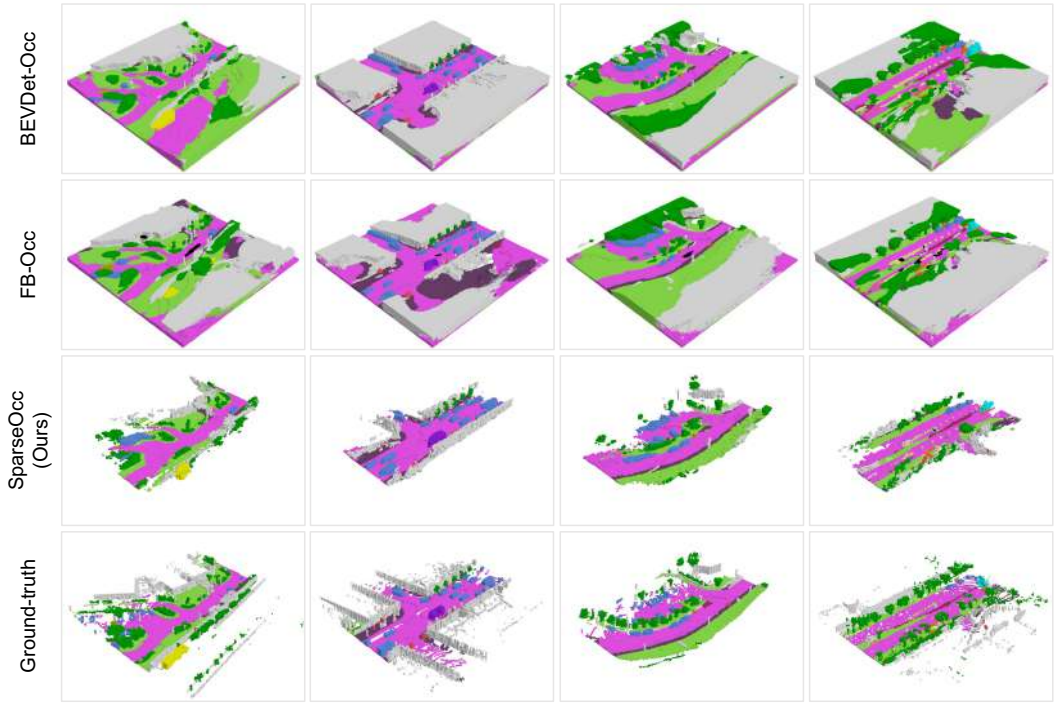


Figure 7: Visualized comparison of semantic occupancy prediction. Despite discarding over 90% of voxels, our SparseOcc effectively models the geometry of the scene and captures fine-grained details (*e.g.*, the yellow-marked traffic cone in the bottom row).

Table 2: Sparse voxel decoder vs. dense voxel decoder. Our sparse voxel decoder achieves nearly 4× faster inference speed than the dense counterparts.

Voxel Decoder	RayIoU	RayIoU_{1m}	RayIoU_{2m}	RayIoU_{4m}	FPS
Dense coarse-to-fine	**29.9**	**24.0**	30.4	**35.4**	6.3
Dense patch-based	25.8	20.4	26.0	30.9	7.8
Sparse coarse-to-fine	**29.9**	23.9	**30.5**	35.2	**24.0**

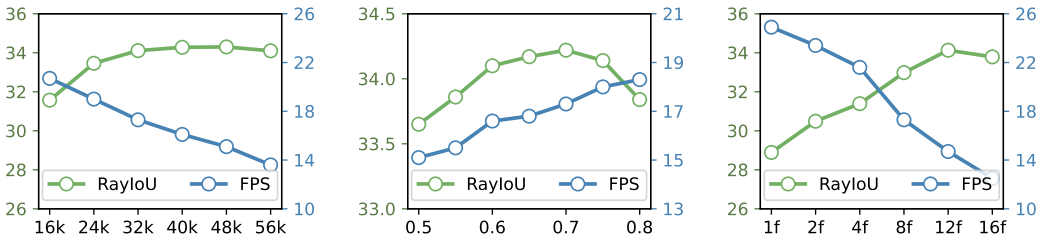
Sparse voxel decoder vs. dense voxel decoder. In Tab. 2, we compare our sparse voxel decoder to the dense counterparts. Here, we implement two baselines, and both of them output a dense feature map with shape as $200\times200\times16\times C$. The first baseline is a coarse-to-fine architecture without pruning empty voxels. In this baseline, we also replace self-attention with 3D convolution and use 3D deconvolution to upsample predictions. The other baseline is a patch-based architecture by dividing the 3D space into a small number of patches as PETRv2 [35] for BEV segmentation. We use $25\times25\times2 = 1250$ queries and each one of them corresponds to a specific patch of shape $8\times8\times8$. A stack of deconvolution layers are used to lift the coarse queries to a full-resolution 3D volume.

As we can see from the table, the dense coarse-to-fine baseline achieves a good performance of 29.9 RayIoU but with a slow inference speed of 6.3 FPS. The patch-based one is slightly faster with 7.8 FPS inference speed but with a severe performance drop by 4.1 RayIoU. Instead, our sparse voxel decoder produces sparse 3D features in the shape of $K \times C$ (where $K = 32000 \ll 200\times200\times16$), achieving an inference speed that is nearly 4× faster than the counterparts without compromising performance. This demonstrates the necessity and effectiveness of our sparse design.

Mask Transformer. In Tab. 3, we ablate the effectiveness of the mask transformer. The first row is a simple per-voxel baseline which directly predicts semantics from the sparse voxel decoder using a stack of MLPs. Introducing mask transformer with vanilla cross attention (as it is the common practice in MaskFormer and Mask3D) gives a performance boost of 1.7 RayIoU, but inevitably slows down the inference speed as it attends to all locations in an image. Therefore, to speed up the dense

Table 3: Ablation of mask transformer (MT) and the cross attention module in MT. Mask-guided sparse sampling is stronger and faster than the dense cross attention.

MT	Cross Attention	RayIoU	RayIoU_{1m}	RayIoU_{2m}	RayIoU_{4m}	FPS
-	-	27.0	20.3	27.5	33.1	**29.0**
√	Dense cross attention	28.7	22.9	29.3	33.8	16.2
√	Sparse sampling	25.8	20.5	26.2	30.8	24.0
√	**+ Mask-guided**	**29.2**	**23.4**	**29.8**	**34.5**	24.0



(a) Pruning by top-k (b) Pruning by thresholding (c) Number of frames

Figure 8: Ablations on voxel sparsity and temporal modeling. (a) The optimal performance occurs when k is set to 32000 (5% sparsity). (b) Top-k can also be substituted with thresholding, *e.g.*, voxels scoring less than a certain threshold will be pruned. (c) The performance continues to increase with the number of frames, but it starts to saturate after 12 frames.

cross-attention pipeline, we adopt a sparse sampling mechanism which brings a 50% reduction in inference time. By further introducing the predicted masks to guide the generation of sampling points, we finally achieve 29.2 RayIoU with 24 FPS.

Is a limited set of voxels sufficient to cover the scene? In this study, we delve deeper into the impact of voxel sparsity on final performance. To investigate this, we systematically ablate the value of k in Fig. 8 (a). Starting from a modest value of 16k, we observe that the optimal performance occurs when k is set to 32k $\sim$ 48k, which is only 5% $\sim$ 7.5% of the total number of dense voxels (200×200×16 = 640000). Surprisingly, further increasing k does not yield any performance improvements; instead, it introduces noise. Thus, our findings suggest that a ∼5% sparsity level is sufficient. Keep increasing the density will reduce both accuracy and speed.

Pruning by top-k is simple and effective, but it is related to specific dataset. In real world, we can substitute top-k with a thresholding method. Voxels scoring less than a given threshold (*e.g.*, 0.7) will be pruned. Thresholding achieves similar performance to top-k (see Fig. 8 (b)), and has the ability to generalize to different scenes.

Temporal modeling. In Fig. 8 (c), we validate the effectiveness of temporal fusion. We can see that the temporal modeling of SparseOcc is very effective, with performance steadily increasing as the number of frames increases. The performance peaks at 12 frames and then saturates. However, the inference speed drops rapidly as the sampling points need to interact with every frame.

5.4 More Studies

The effect of training with visible masks. Interestingly, we observed a peculiar phenomenon. Under the traditional voxel-level mIoU metric, dense methods can significantly benefit from disregarding the non-visible voxels during training. These non-visible voxels are indicated by a binary visible mask provided by the Occ3D-nuScenes dataset. However, we find that this strategy actually impairs performance under our new RayIoU metric. For instance, we train two variants of BEVFormer: one uses the visible mask during training, and the other does not. As shown in Tab. 4, the former scores 15 points higher than the latter on the voxel-based mIoU, but it scores 1 point lower on RayIoU. This phenomenon is also observed on FB-Occ.

Table 4: To verify the effect of the visible mask, wo provide per-class RayIoU of BEVFormer and FB-Occ. † uses the visible mask during training. We find that training with the visible mask hurts the performance of background classes such as drivable surface, terrian and sidewalk.

Method	mIoU	RayIoU	Per-class RayIoU																
			others	barrier	bicycle	bus	car	cons. veh.	motor.	pedes.	tfc. cone	trailer	truck	drv. surf.	other flat	sidewalk	terrain	manmade	vegetation
BEVFormer	23.7	**33.7**	5.0	42.2	18.2	**55.2**	**57.1**	**22.7**	21.3	31.0	**27.1**	**30.7**	49.4	**58.4**	**30.4**	**29.4**	**31.7**	36.3	26.5
BEVFormer †	**39.2**	32.4	**6.4**	**44.8**	**24.0**	**55.2**	56.7	21.0	**29.8**	**33.5**	26.8	27.9	**49.5**	45.8	18.7	22.4	18.5	**39.1**	**29.8**
FB-Occ	27.9	**35.6**	**10.5**	44.8	25.6	55.6	51.7	22.6	27.2	**34.3**	**30.3**	23.7	44.1	**65.5**	**33.3**	**31.4**	**32.5**	**39.6**	**33.3**
FB-Occ †	**39.1**	33.5	5.0	**44.9**	**26.2**	**59.7**	**55.1**	**27.9**	**29.1**	**34.3**	29.6	**29.1**	**50.5**	44.4	22.4	21.5	19.5	39.3	31.1

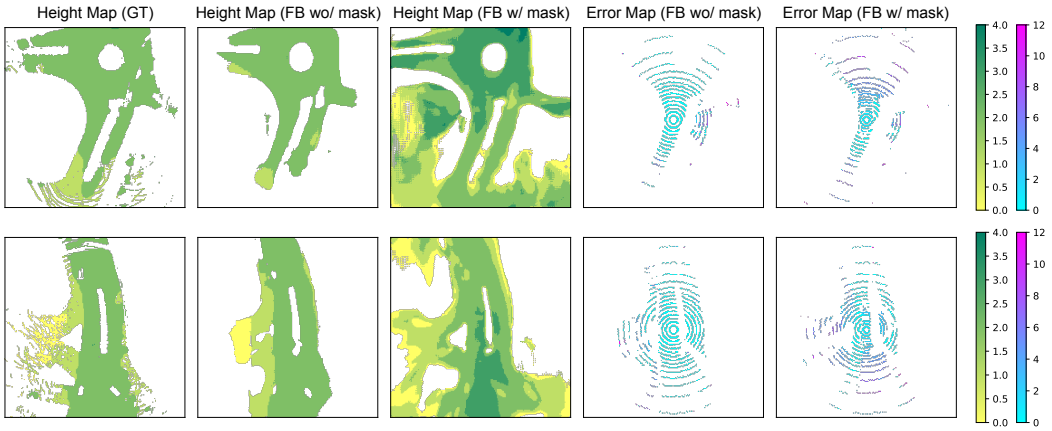


Figure 9: Why does the performance of background classes, such as drivable surfaces, degrade when using the visible mask during training? We provide a visualization of the drivable surface as predicted by FB-Occ. Here, "FB w/ mask" and "FB wo/ mask" denote training with and without the visible mask, respectively. We observe that "FB w/ mask" tends to predict a higher and thicker road surface, resulting in significant depth errors along a ray. In contrast, "FB wo/ mask" predicts a road surface that is both accurate and consistent.

To further explore this, we present the per-class RayIoU in Tab. 4. The table reveals that training with the visible mask enhances performance for most foreground classes such as bus, bicycle, and truck. However, it negatively impacts background classes like drivable surface and terrain.

This observation raises a further question: *Why does the performance of the background category degrade?* To address this, we offer a visual comparison of the depth errors and height maps of the predicted drivable surface from FB-Occ in Fig. 9, both with and without the use of visible mask during training. The figure illustrates that training with visible masks results in a thicker and higher ground prediction, leading to substantial depth errors in distant areas. Conversely, models trained without the visible mask predict depth with greater accuracy.

From these observations, we derive some valuable insights: ignoring non-visible voxels during training benefits foreground classes by resolving the issue of ambiguous labeling of unscanned voxels. However, it also compromises the accuracy of depth estimation, as models tend to predict a thicker and closer surface. We hope that our findings will benefit future research.

Panoptic occupancy. We then show that SparseOcc can be easily extended for panoptic occupancy prediction, a task derived from panoptic segmentation that segments images to not only semantically meaningful regions but also to detect and distinguish individual instances. Compared to panoptic segmentation, panoptic occupancy prediction requires the model to have geometric awareness in order to construct the 3D scene for segmentation. By additionally introducing instance queries to the mask transformer, we seamlessly achieve the first fully sparse panoptic occupancy prediction framework using camera-only inputs.

Table 5: Panoptic occupancy prediction performance on Occ3D-nuScenes.

Method	Backbone	Input Size	Epoch	RayPQ	RayPQ_{1m}	RayPQ_{2m}	RayPQ_{4m}
SparseOcc	R50	704×256	24	**14.1**	10.2	14.5	17.6

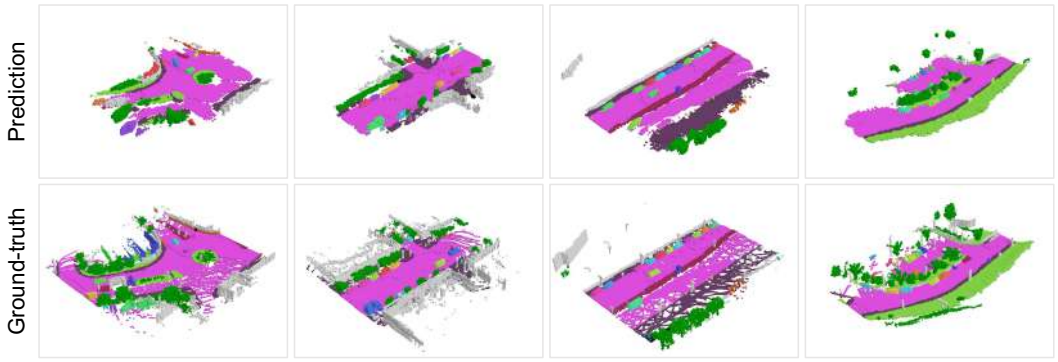


Figure 10: Panoptic occupancy prediction. Different instances are distinguished by colors. Our model can capture fine-grained objects and road structures simultaneously.

Firstly, we utilize the ground-truth bounding boxes from the 3D object detection task to generate the panoptic occupancy ground truth. Specifically, we define eight instance categories (including `car`, `truck`, `construction vehicle`, `bus`, `trailer`, `motorcycle`, `bicycle`, `pedestrian`) and ten staff categories (including `terrain`, `manmade`, `vegetation`, etc). Each instance segment is identified by grouping the voxels inside the bounding box based on an existing semantic occupancy benchmark, such as Occ3D-nuScenes.

We then design RayPQ based on the well-known panoptic quality (PQ) [22] metric, which is defined as the multiplication of *segmentation quality* (SQ) and *recognition quality* (RQ):

$$\text{PQ} = \underbrace{\frac{\sum_{(p,g)\in TP} \text{IoU}(p,g)}{|TP|}}_{\text{segmentation quality (SQ)}} \times \underbrace{\frac{|TP|}{|TP| + \frac{1}{2}|FP| + \frac{1}{2}|FN|}}_{\text{recognition quality (RQ)}}, \tag{4}$$

where the definition of true positive (TP) is the same as that in RayIoU. The threshold of IoU between prediction p and ground-truth g is set to 0.5.

In Tab. 5, we report the performance of SparseOcc on panoptic occupancy benchmark. Similar to RayIoU, we calculate RayPQ under three distance thresholds: 1, 2 and 4 meters. SparseOcc achieves an averaged RayPQ of 14.1. The visualizations are presented in Fig. 10.

5.5 Limitations

Accumulative errors. In order to implement a fully sparse architecture, we discard a large number of empty voxels in the early stages. However, empty voxels that are mistakenly discarded cannot be recovered in subsequent stages. Moreover, the prediction of the mask transformer is constrained within a space predicted by the sparse voxel decoder. Some ground-truth instances do not appear in this unreliable space, leading to inadequate training of the mask transformer.

6 Conclusion

In this paper, we proposed a fully sparse occupancy network, named SparseOcc, which neither relies on dense 3D feature, nor has sparse-to-dense and global attention operations. We also created RayIoU, a ray-level metric for occupancy evaluation, eliminating the inconsistency flaws of previous metric. Experiments show that SparseOcc achieves the state-of-the-art performance on the Occ3D-nuScenes dataset for both speed and accuracy. We hope this exciting result will attract more attention to the fully sparse 3D occupancy paradigm.

Acknowledgements

We thank the anonymous reviewers for their suggestions that make this work better. This work is supported by the National Key R&D Program of China (No. 2022ZD0160900), the National Natural Science Foundation of China (No. 62076119, No. 61921006), the Fundamental Research Funds for the Central Universities (No. 020214380119), and the Collaborative Innovation Center of Novel Software Technology and Industrialization.

References

[1] Tesla AI Day. https://www.youtube.com/watch?v=j0z4FweCy4M (2021)

[2] Bozic, A., Palafox, P., Thies, J., Dai, A., Nießner, M.: Transformerfusion: Monocular rgb scene reconstruction using transformers. In: NeurIPS (2021)

[3] Caesar, H., Bankiti, V., Lang, A.H., Vora, S., Liong, V.E., Xu, Q., Krishnan, A., Pan, Y., Baldan, G., Beijbom, O.: nuscenes: A multimodal dataset for autonomous driving. In: CVPR (2020)

[4] Cao, A.Q., de Charette, R.: Monoscene: Monocular 3d semantic scene completion. In: CVPR (2022)

[5] Cheng, B., Misra, I., Schwing, A.G., Kirillov, A., Girdhar, R.: Masked-attention mask transformer for universal image segmentation. In: CVPR (2022)

[6] Cheng, B., Schwing, A., Kirillov, A.: Per-pixel classification is not all you need for semantic segmentation. In: NeurIPS (2021)

[7] Choy, C., Gwak, J., Savarese, S.: 4d spatio-temporal convnets: Minkowski convolutional neural networks. In: Proceedings of the IEEE/CVF conference on computer vision and pattern recognition. pp. 3075–3084 (2019)

[8] Curless, B., Levoy, M.: A volumetric method for building complex models from range images. In: SIGGRAPH (1996)

[9] Ding, Y., Huang, L., Zhong, J.: Multi-scale occ: 4th place solution for Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition 2023 3d occupancy prediction challenge. arXiv preprint arXiv:2306.11414 (2023)

[10] Feng, Z., Yang, L., Guo, P., Li, B.: Cvrecon: Rethinking 3d geometric feature learning for neural reconstruction. In: ICCV (2023)

[11] Gan, W., Mo, N., Xu, H., Yokoya, N.: A comprehensive framework for 3d occupancy estimation in autonomous driving. IEEE Transactions on Intelligent Vehicles pp. 1–19 (2024)

[12] Gao, Z., Wang, L., Han, B., Guo, S.: Adamixer: A fast-converging query-based object detector. In: CVPR (2022)

[13] He, K., Chen, X., Xie, S., Li, Y., Dollár, P., Girshick, R.: Masked autoencoders are scalable vision learners. In: CVPR (2022)

[14] He, K., Zhang, X., Ren, S., Sun, J.: Deep residual learning for image recognition. In: CVPR (2016)

[15] Huang, J., Huang, G.: Bevdet4d: Exploit temporal cues in multi-camera 3d object detection. arXiv preprint arXiv:2203.17054 (2022)

[16] Huang, J., Huang, G., Zhu, Z., Du, D.: Bevdet: High-performance multi-camera 3d object detection in bird-eye-view. arXiv preprint arXiv:2112.11790 (2021)

[17] Huang, L., Li, Z., Sima, C., Wang, W., Wang, J., Qiao, Y., Li, H.: Leveraging vision-centric multi-modal expertise for 3d object detection. In: NeurIPS (2024)

[18] Huang, L., Wang, H., Zeng, J., Zhang, S., Cao, L., Ji, R., Yan, J., Li, H.: Geometric-aware pretraining for vision-centric 3d object detection. arXiv preprint arXiv:2304.03105 (2023)

[19] Huang, Y., Zheng, W., Zhang, Y., Zhou, J., Lu, J.: Tri-perspective view for vision-based 3d semantic occupancy prediction. In: CVPR (2023)

[20] Jia, X., De Brabandere, B., Tuytelaars, T., Gool, L.V.: Dynamic filter networks. In: NeurIPS (2016)

[21] Khurana, T., Hu, P., Held, D., Ramanan, D.: Point cloud forecasting as a proxy for 4d occupancy forecasting. In: CVPR (2023)

[22] Kirillov, A., He, K., Girshick, R., Rother, C., Dollár, P.: Panoptic segmentation. In: CVPR (2019)

[23] Lang, A.H., Vora, S., Caesar, H., Zhou, L., Yang, J., Beijbom, O.: Pointpillars: Fast encoders for object detection from point clouds. In: CVPR (2019)

[24] Li, H., Li, Y., Wang, H., Zeng, J., Cai, P., Xu, H., Lin, D., Yan, J., Xu, F., Xiong, L., Wang, J., Zhu, F., Yan, K., Xu, C., Wang, T., Mu, B., Ren, S., Peng, Z., Qiao, Y.: Open-sourced data ecosystem in autonomous driving: the present and future. arXiv preprint arXiv:2312.03408 (2023)

[25] Li, H., Sima, C., Dai, J., Wang, W., Lu, L., Wang, H., Zeng, J., Li, Z., Yang, J., Deng, H., et al.: Delving into the devils of bird's-eye-view perception: A review, evaluation and recipe. IEEE TPAMI (2023)

[26] Li, Y., Yu, Z., Choy, C., Xiao, C., Alvarez, J.M., Fidler, S., Feng, C., Anandkumar, A.: Voxformer: Sparse voxel transformer for camera-based 3d semantic scene completion. In: CVPR (2023)

[27] Li, Z., Wang, W., Li, H., Xie, E., Sima, C., Lu, T., Qiao, Y., Dai, J.: Bevformer: Learning bird's-eye-view representation from multi-camera images via spatiotemporal transformers. In: ECCV (2022)

[28] Li, Z., Yu, Z., Austin, D., Fang, M., Lan, S., Kautz, J., Alvarez, J.M.: Fb-occ: 3d occupancy prediction based on forward-backward view transformation. arXiv preprint arXiv:2307.01492 (2023)

[29] Lin, T.Y., Dollár, P., Girshick, R., He, K., Hariharan, B., Belongie, S.: Feature pyramid networks for object detection. In: CVPR (2017)

[30] Lin, X., Lin, T., Pei, Z., Huang, L., Su, Z.: Sparse4d: Multi-view 3d object detection with sparse spatial-temporal fusion. arXiv preprint arXiv:2211.10581 (2022)

[31] Liu, H., Lu, T., Xu, Y., Liu, J., Li, W., Chen, L.: Camliflow: Bidirectional camera-lidar fusion for joint optical flow and scene flow estimation. In: CVPR (2022)

[32] Liu, H., Lu, T., Xu, Y., Liu, J., Wang, L.: Learning optical flow and scene flow with bidirectional camera-lidar fusion. arXiv preprint arXiv:2303.12017 (2023)

[33] Liu, H., Teng, Y., Lu, T., Wang, H., Wang, L.: Sparsebev: High-performance sparse 3d object detection from multi-camera videos. In: ICCV (2023)

[34] Liu, Y., Wang, T., Zhang, X., Sun, J.: PETR: position embedding transformation for multi-view 3d object detection. In: ECCV (2022)

[35] Liu, Y., Yan, J., Jia, F., Li, S., Gao, Q., Wang, T., Zhang, X., Sun, J.: Petrv2: A unified framework for 3d perception from multi-camera images. arXiv preprint arXiv:2206.01256 (2022)

[36] Loshchilov, I., Hutter, F.: Decoupled weight decay regularization. In: ICLR (2019)

[37] Mescheder, L., Oechsle, M., Niemeyer, M., Nowozin, S., Geiger, A.: Occupancy networks: Learning 3d reconstruction in function space. In: CVPR (2019)

[38] Milletari, F., Navab, N., Ahmadi, S.A.: V-net: Fully convolutional neural networks for volumetric medical image segmentation. In: 3DV (2016)

[39] Murez, Z., Van As, T., Bartolozzi, J., Sinha, A., Badrinarayanan, V., Rabinovich, A.: Atlas: End-to-end 3d scene reconstruction from posed images. In: ECCV (2020)

[40] Pan, M., Liu, J., Zhang, R., Huang, P., Li, X., Liu, L., Zhang, S.: Renderocc: Vision-centric 3d occupancy prediction with 2d rendering supervision. arXiv preprint arXiv:2309.09502 (2023)

[41] Paszke, A., Gross, S., Massa, F., Lerer, A., Bradbury, J., Chanan, G., Killeen, T., Lin, Z., Gimelshein, N., Antiga, L., et al.: Pytorch: An imperative style, high-performance deep learning library. In: NeurIPS (2019)

[42] Peng, S., Niemeyer, M., Mescheder, L., Pollefeys, M., Geiger, A.: Convolutional occupancy networks. In: ECCV (2020)

[43] Ronneberger, O., Fischer, P., Brox, T.: U-net: Convolutional networks for biomedical image segmentation. In: MICCAI (2015)

[44] Schult, J., Engelmann, F., Hermans, A., Litany, O., Tang, S., Leibe, B.: Mask3d: Mask transformer for 3d semantic instance segmentation. In: ICRA (2023)

[45] Sima, C., Tong, W., Wang, T., Chen, L., Wu, S., Deng, H., Gu, Y., Lu, L., Luo, P., Lin, D., Li, H.: Scene as occupancy. In: ICCV (2023)

[46] Stier, N., Rich, A., Sen, P., Höllerer, T.: Vortx: Volumetric 3d reconstruction with transformers for voxelwise view selection and fusion. In: 3DV (2021)

[47] Sun, J., Xie, Y., Chen, L., Zhou, X., Bao, H.: Neuralrecon: Real-time coherent 3d reconstruction from monocular video. In: CVPR (2021)

[48] Takikawa, T., Litalien, J., Yin, K., Kreis, K., Loop, C., Nowrouzezahrai, D., Jacobson, A., McGuire, M., Fidler, S.: Neural geometric level of detail: Real-time rendering with implicit 3d shapes. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 11358–11367 (2021)

[49] Takmaz, A., Fedele, E., Sumner, R.W., Pollefeys, M., Tombari, F., Engelmann, F.: Openmask3d: Open-vocabulary 3d instance segmentation. arXiv preprint arXiv:2306.13631 (2023)

[50] Tian, X., Jiang, T., Yun, L., Wang, Y., Wang, Y., Zhao, H.: Occ3d: A large-scale 3d occupancy prediction benchmark for autonomous driving. In: NeurIPS Datasets and Benchmarks (2023)

[51] Tolstikhin, I.O., Houlsby, N., Kolesnikov, A., Beyer, L., Zhai, X., Unterthiner, T., Yung, J., Steiner, A., Keysers, D., Uszkoreit, J., et al.: Mlp-mixer: An all-mlp architecture for vision. In: NeurIPS (2021)

[52] Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A.N., Kaiser, Ł., Polosukhin, I.: Attention is all you need. In: NeurIPS (2017)

[53] Wang, S., Liu, Y., Wang, T., Li, Y., Zhang, X.: Exploring object-centric temporal modeling for efficient multi-view 3d object detection. arXiv preprint arXiv:2303.11926 (2023)

[54] Wang, X., Zhu, Z., Xu, W., Zhang, Y., Wei, Y., Chi, X., Ye, Y., Du, D., Lu, J., Wang, X.: Openoccupancy: A large scale benchmark for surrounding semantic occupancy perception. arXiv preprint arXiv:2303.03991 (2023)

[55] Wang, Y., Guizilini, V.C., Zhang, T., Wang, Y., Zhao, H., Solomon, J.: Detr3d: 3d object detection from multi-view images via 3d-to-2d queries. In: CoRL (2022)

[56] Wang, Y., Chen, Y., Liao, X., Fan, L., Zhang, Z.: Panoocc: Unified occupancy representation for camera-based 3d panoptic segmentation. arXiv preprint arXiv:2306.10013 (2023)

[57] Wei, Y., Zhao, L., Zheng, W., Zhu, Z., Zhou, J., Lu, J.: Surroundocc: Multi-camera 3d occupancy prediction for autonomous driving. In: ICCV (2023)

[58] Yang, Z., Chen, L., Sun, Y., Li, H.: Visual point cloud forecasting enables scalable autonomous driving. In: CVPR (2024)

[59] Yang, Z., Jiang, L., Sun, Y., Schiele, B., Jia, J.: A unified query-based paradigm for point cloud understanding. In: ICCV (2022)

[60] Yang, Z., Sun, Y., Liu, S., Jia, J.: 3dssd: Point-based 3d single stage object detector. In: CVPR (2020)

[61] Yang, Z., Sun, Y., Liu, S., Shen, X., Jia, J.: Std: Sparse-to-dense 3d object detector for point cloud. In: ICCV (2019)

[62] Yang, Z., Zhou, Y., Chen, Z., Ngiam, J.: 3d-man: 3d multi-frame attention network for object detection. In: ICCV (2021)

[63] Yin, T., Zhou, X., Krahenbuhl, P.: Center-based 3d object detection and tracking. In: CVPR (2021)

[64] Zhu, X., Su, W., Lu, L., Li, B., Wang, X., Dai, J.: Deformable detr: Deformable transformers for end-to-end object detection. arXiv preprint arXiv:2010.04159 (2020)