arXiv:2304.01529v1 [cs.CV] 4 Apr 2023

IterativePFN：真正的迭代式点云滤波

Dasith de Silva Edirimuni[1], Xuequan Lu[1*], Zhiwen Shao[2†], Gang Li[1],Antonio Robles-Kelly[1,4],Ying He[3]1信息技术学院，迪肯大学2中国矿业大学计算机科学与技术学院3南洋理工大学计算机科学与工程学院4国防科学技术集团，澳大利亚

{dtdesilva, xuequan.lu, gang.li, antonio.robles-kelly}@deakin.edu.au, zhiwen shao@cumt.edu.cn, yhe@ntu.edu.sg

摘要

点云的质量常受采集过程中引入的噪声影响。因此，一项基础的三维视觉任务便是去除噪声，这也称为点云滤波或去噪。当前最先进的基于学习的方法侧重于训练神经网络，以推断滤波位移，并直接将噪声点移至底层干净表面。在噪声较强的情况下，这些方法会迭代执行滤波过程。然而，这种迭代滤波仅在测试阶段进行，且在确保点快速收敛至干净表面方面效果较差。我们提出***IterativePFN***（迭代式点云滤波网络），该网络由多个迭代模块组成，可在单个网络内部对真正的迭代滤波过程进行建模。我们采用一种新颖的损失函数训练*IterativePFN*网络，该函数在每次迭代中使用自适应真实值目标，以捕捉训练期间各中间滤波结果之间的关系。这能确保滤波结果更快地收敛至干净表面。与当前最先进的方法相比，我们的方法能够取得更好的性能。源代码见：https://github.com/ddsediri/IterativePFN。

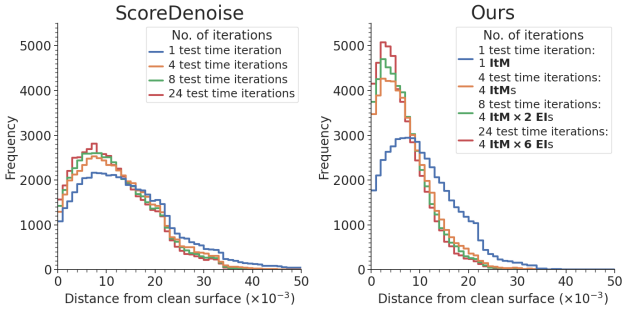

图1. ScoreDenoise [24]在含有5万个点和2.5%高斯噪声的Casting 形状上，经过1、4、8和24次测试时迭代后，滤波后的点到干净表面的距离直方图。我们将其与提出的 IterativePFN 进行比较，其中1个迭代模块（ItM）对应1次内部迭代，4个 ItM 相当于1次外部迭代（EI）。所提出的网络中包含4个 ItM。请注意，1个 ItM 相当于 ScoreDenoise 的1次测试时迭代。我们的滤波结果收敛到更接近曲面的位置。

1. 引言

点云是三维几何信息的一种自然表示形式，在三维计算机视觉领域有着广泛的应用。这些应用涵盖机器人学、自动驾驶和城市规划等领域 [16, 21, 37, 41]。点云由三维传感器采集，由无序的点组成，且缺乏连通性信息。此外，点云数据的采集容易出错，因为传感器质量和环境因素可能会引入噪声伪影。去除噪声的过程是一项基础研究问题，也正是点云滤波（亦称去噪）这一领域的研究动机。滤波有助于法线估计等其他任务，并进一步促进三维渲染和曲面重建。

传统的点云滤波方法，例如基于 MLS 的方法、[2, 12]，双边滤波机制 [9] 和边缘恢复算法 [20, 36] ，依赖点集的局部信息，即点法线，来对点云进行滤波。然而，这类方法受到法线精度的限制。另一类方法包括局部最优投影（LOP）系列方法 [14, 19, 27], , 这类方法会对点云进行下采样和正则化，但会损失重要的几何细节。近年来，研究人员提出了基于深度学习的滤波方法，以缓解传统方法的缺点和局限性 [24–26, 30, 44]。

*通讯作者：X. Lu，获 PJ03906.PG00507.F002 资助。
†Z. Shao 获国家自然科学基金资助（编号：62106268）。

早期的**基于深度学习的**滤波方法（例如 PointProNets） [33], 需要预处理后的二维高度图来对点云进行滤波。然而，随着 PointNet、PointNet++ 和 DGCNN 的出现，直接进行点集卷积成为可能 [28, 29, 39]。近期的方法利用基于这些架构的特征编码器，为点集输入生成更丰富的潜在表示，并更有效地滤除噪声 [22,24,30,44]。这些方法大致可分为：1）**重采样**，2）**概率**，以及 3）**位移**方法。基于重采样的方法（例如 DMRDenoise） [22] 会损失几何细节，因为这类方法依赖于识别下采样后的底层干净表面，并沿其进行上采样。ScoreDenoise 对噪声卷积概率的梯度对数进行建模，以找到给定位置上的点，并通过受朗之万采样启发的梯度上升过程进行迭代 [24]滤波。然而，如图1所示，经过多次测试时迭代的滤波后，滤波点仍需较长时间才能收敛到干净表面。相比之下，对于包含 4 个 IterationModules 的 IterativePFN网络，其中 1 个迭代模块（ItM）表示 1 次内部滤波迭代，并且与 ScoreDenoise 的 1 次测试时迭代相对应，我们可以看到，在相同数量的测试时迭代内，更多滤波点能够收敛到更接近干净表面的位置。

在基于位移的方法中，PointCleanNet（PCN） [30] 对高噪声表现出敏感性，而 Pointfil- ter [44] 采用了受双边滤波启发的加权损失函数，导致彼此距离较近的曲面在滤波过程中塌缩为单一曲面。此外，基于梯度下降和基于位移的方法会在测试阶段对点云进行迭代滤波，但在训练过程中并未考虑真正的迭代滤波。尽管 RePCDNet [7] 提供了一种受循环神经网络启发的替代方案，以便在训练过程中捕获这些信息，但在每次迭代中，RePCDNet 都试图直接将点移动到干净表面上，而没有考虑到在不同迭代次数下，点所具有的残余噪声会按迭代次数递减。此外，它使用单一网络对点进行滤波，增加了网络正确区分不同噪声尺度的负担，并且需要多次测试时迭代，从而导致效率较低。基于这些主要局限性，我们提出：

- 一种新颖的神经网络架构，由堆叠的编码器-解码器模块组成，称为**IterationModule**，用于在内部建模真实的迭代滤波过程（见图2）。每个 IterationModule 表示一次滤波迭代，且 τ阶 IterationModule 的输出将成为 $\tau+1$阶 IterationModule 的输入。因此， $\tau+1$阶 IterationModule 表示第 $t=\tau+1$次滤波迭代。这使网络能够理解各次迭代之间的滤波关系。

- 一种新颖的损失函数：在每次迭代中，将最近邻损失定义为 L_2 范数最小化，即最小化由第 τ 个 IterationModule 推断出的滤波位移与目标点云在 $t=\tau$ 处的最近点之间的距离；该目标点云的噪声尺度 σ_τ 低于目标在 $t=\tau-1$ 处的噪声尺度 $\sigma_{\tau-1}$。这促使滤波过程逐步进行，并推动结果收敛到干净表面。

- 一种广义的补丁拼接方法，在确定重叠局部块内最佳滤波点时设计高斯权重。补丁拼接提高了效率，因为它有助于同时对多个点进行滤波。

我们将本方法与最先进的方法进行了全面比较实验，结果表明，本方法在合成数据和真实世界数据上均具有优势。

2. 相关工作

传统方法。Levin 提出的移动最小二乘（MLS） [17] 经 Alexa 等人的研究扩展，用于点云滤波。 [2] 其目的是最小化滤波后曲面与其真实值之间的近似误差。随后，Adamson 和 Alexa 提出了隐式移动最小二乘（IMLS） [1]，而 Guennebaud 和 Gross 则提出了各向异性点集曲面（APSS） [12], ，这两种方法都试图提升含有尖锐几何特征的点云的滤波效果。然而，这些 MLS 技术依赖局部曲面的参数，例如点法线。为缓解这一问题，Lipman 等人 [19] 提出了局部最优投影（LOP）方法。除了不依赖局部曲面参数外，基于 LOP 的方法还会对输入点云进行降采样和正则化。这项工作催生了一类新的点云滤波方法，包括 Preiner 等人提出的连续局部最优投影（CLOP） [27] 以及 Huang 等人提出的加权局部最优投影（WLOP）。 [14]总体而言，这两类方法都存在相同的缺点：无法保留底层干净曲面的尖锐几何特征，并且对噪声敏感。

此外，Cazals 和 Pouget 提出对点集 [6] 拟合 n 阶多项式曲面，以计算点法线和曲率等感兴趣的量，并已在 [4, 46]中见到基于学习的应用。此外，将噪声点投影到拟合曲面上，即可得到相应的滤波点。Digne 提出了一种基于相似性的滤波方法，该方法为每个点使用局部描述子，随后利用描述子之间的相似性，确定每个点的滤波位移 [8]。最近，Digne 和 de Franchis 提出将网格双边滤波机制 [11]推广到点云，通过滤波点来实现

一种基于点邻域各向异性加权的滤波方法。这种加权同时考虑点位置及其法向量。其他方法，例如 L_1 以及 L_0 Avron 等人和 Sun、Schaefer 与 Wang 提出的最小化方法 [3, 36]以及 Lu 等人的低秩矩阵近似 [20]，会估计法向量，并将其用于滤波算法以对点云进行去噪。Remil 等人的工作还将滤波过程重新表述为一个可通过增广拉格朗日乘子求解的稀疏优化问题 [32]。

基于深度学习的方法。在基于学习的方法中，Roveri 等人提出了 PointProNets [33]，并采用传统的卷积神经网络架构，对重新投影到三维空间中的含噪二维高度图进行滤波。相比之下，Lu 等人的深度特征保持（DFP）网络使用传统卷积神经网络处理二维高度图，以估计点法线；这些法线可与 [20] 的位置更新算法结合使用。随着 Point-Net 和 PointNet ++ [28, 29] 的出现，点集的直接卷积成为可能。此外，Wang 等人 [39]提出了一种图卷积架构，该架构接收由点集生成的最近邻图，从而生成丰富的点特征表示。以下方法采用受 PointNet 或 DGCNN 启发的骨干网络来处理点云输入。Yu 等人开发了 EC-Net，通过具备边缘感知能力的上采样执行去噪，从而将点整合到曲面上 [43]。Luo 和 Hu 在其 DM-RDenoise 机制中采用了类似方法 [22]：该机制首先逐步识别底层的下采样、低噪声流形，随后沿该流形对点进行上采样。Rakotosaona 等人提出的 PCN [30], 对点云块中的中心点进行滤波。PCN 通过使用损失函数，最小化预测位移与真值块中点之间的 L_2 范数，从而推断这些滤波位移。此外，他们加入了排斥项，以确保点的规则分布。Zhang 等人提出了 Pointfilter [44]，进一步发展了基于位移的研究思路。他们在训练期间使用真实值法向量来计算双边损失。Pistilli 等人提出了 GPDNet，这是一种用于推断滤波位移的图卷积架构 [26]。最近，Chen 等人提出了 RePCD-Net，这是一种循环架构，在训练期间迭代地对点云进行滤波，并考虑每次迭代的训练损失。然而，该方法在测试阶段仍必须迭代应用 [7]。

在测试时通过对含噪数据进行朗之万采样，迭代地恢复噪声较少的数据，这一方法最初由二维生成建模领域提出 [35]，后来被应用于点云生成和重建 [5, 23]。该方法依赖于学习数据的未归一化的概率分布对数梯度。Luo 和 Hu 将其扩展到点云滤波。他们的方法 ScoreDenoise 对点云局部块 [35] 所对应的噪声卷积概率分布的梯度对数进行建模分布。Mao 等人提出 PDFlow，通过利用归一化流，在更高维度上揭示无噪数据的潜在表示，进而获得滤波位移 [25]。

3. 研究问题与动机

如第1节所述，先前的点云滤波方法可分为重采样、概率方法和基于位移的方法，其中第二类包括基于分数的方法。我们的动机源于基于位移的方法与基于分数的方法在滤波目标上的相互作用，以及基于分数的方法如何为构建真正迭代的基于位移的方法提供启发，即使其训练目标具有迭代性。PCN [30]提出，滤波目标应通过位移将噪声点回归 $\boldsymbol{x}_i$ 至其下方的干净表面 $\boldsymbol{d}_i$。该位移由回归网络输出，网络以一个以 $\mathcal{X}$为中心的噪声点云补丁 $\boldsymbol{x}_i$作为输入。滤波目标表示为 $\tilde{\boldsymbol{x}}_i = \boldsymbol{x}_i + \boldsymbol{d}_i$ 其中 $\tilde{\boldsymbol{x}}_i$ 是滤波后的点。在测试期间，时间 $t = \tau$ 的输出将作为时间 $t = \tau + 1$下一次迭代的输入。这就形成了测试时的迭代滤波目标：

$$\tilde{\boldsymbol{x}}_i^{(t)} = \tilde{\boldsymbol{x}}_i^{(t-1)} + \boldsymbol{d}_i^{(t)}, t = 1, \cdots, T \tag{1}$$

PCN 的训练目标旨在将噪声点回归至干净补丁 $\mathcal{Y}$，同时确保其位于补丁中心。因此，PCN 的训练目标通过以下损失函数实现：

$$L_i^{PCN} = \alpha \min_{\boldsymbol{x}_j \in \mathcal{Y}} \|\boldsymbol{d}_i - \boldsymbol{\delta x}_j\|_2^2 + (1-\alpha) \max_{\boldsymbol{x}_j \in \mathcal{Y}} \|\boldsymbol{d}_i - \boldsymbol{\delta x}_j\|_2^2, \tag{2}$$

其中 $\boldsymbol{\delta x}_j = \boldsymbol{x}_j - \boldsymbol{x}_i$。基于分数的方法 ScoreDe- noise [24], 具有类似的滤波目标：

$$\tilde{\boldsymbol{x}}_i^{(t)} = \tilde{\boldsymbol{x}}_i^{(t-1)} + \alpha^{(t)} \mathcal{E}_i(\tilde{\boldsymbol{x}}_i^{(t-1)}),\ t = 1, \cdots, T \tag{3}$$

其中 $\mathcal{E}_i(\boldsymbol{x}) = (1/K) \sum_{\boldsymbol{x}_j \in kNN(\boldsymbol{x}_i)} \mathcal{S}_j(\boldsymbol{x})$ 是给定点的 k 个近邻的分数集成平均。这些分数由 ScoreDenoise 网络预测，对应于噪声卷积概率 $\nabla_{\boldsymbol{x}} \log[(p * n)(\boldsymbol{x})]$ 的梯度对数。其训练目标为：

$$L_i^{SD} = \mathbb{E}_{\boldsymbol{x} \sim \mathcal{N}(\boldsymbol{x}_i)} \left[\|s(\boldsymbol{x}) - \mathcal{S}_i(\boldsymbol{x})\|_2^2 \right], \tag{4}$$

其中 $s(\boldsymbol{x}) = NN(\boldsymbol{x}, \mathcal{Y}) - \boldsymbol{x}$，且 $NN(\boldsymbol{x}, \mathcal{Y})$ 是干净补丁中距离 $\boldsymbol{x}$ 最近的点。

基于上述分析，我们注意到，[24] 的训练目标是考虑 $\boldsymbol{x}$ 附近$\boldsymbol{x}_i$ 的集中分布，而 PCN 的训练目标仅尝试推断 $\boldsymbol{x}_i$的位移。尽管这两种

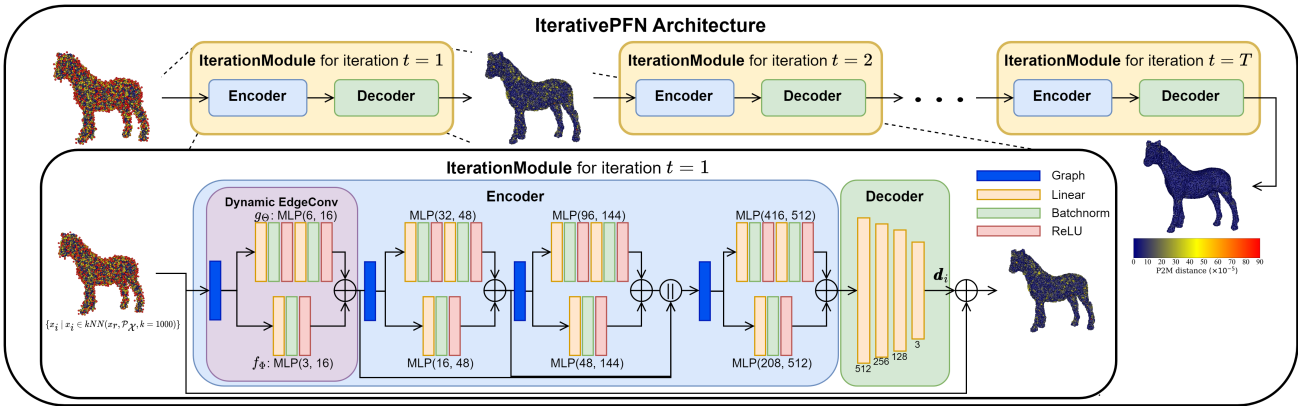

图 2. 我们的 IterativePFN 方法概览。与现有方法在测试阶段需要反复应用不同，我们在网络中使用 T IterationModule 显式建模了 T 次滤波迭代。

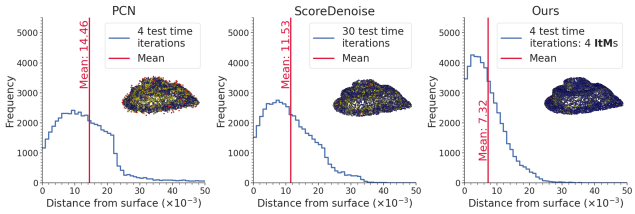

图 3. 每个滤波后点云中各滤波点到干净表面的距离直方图。我们考察初始铸件形状，其点分辨率为 50K，所添加的高斯噪声为 2.5%。

方法都能较好地去除噪声，部分原因在于测试阶段，它们会模拟反向马尔可夫过程的朗之万方程，迭代地去除噪声。然而，我们也观察到一些缺点：**首先**，无论是位移方法还是基于分数的方法，都只在测试阶段进行迭代。在训练过程中，它们接触的是具有底层高斯噪声分布的含噪局部块，并尝试推断滤波位移 [30] 或分数 [24]，以将点直接移至干净补丁上。因此，PCN 的位置更新公式 Eq. (1) 和梯度上升方程 Eq. (3) 都忽略了滤波点中还包含加性噪声这一事实 $\sigma_\tau \xi$ 其中 $\xi \sim \mathcal{N}(0,\ I) \wedge \sigma_\tau \in \mathbb{R}$ 这是由朗之万方程的随机性导致的。

其次，ScoreDenoise [24] 依赖逐渐减小的步长，$\alpha^{(t)}$，以确保公式 (3) 收敛到稳态。然而，这确实会导致收敛速度非常缓慢[5]。这些观察促使我们提出一种基于学习的滤波方法，将迭代滤波有效地融入训练目标，使网络能够在每次迭代中成功识别底层噪声分布，从而施加更稳健的滤波目标，让滤波结果更快地收敛到干净表面。图 3 展示了距干净表面给定距离处滤波点数量的直方图。

我们采用 PCN 和 ScoreDenoise 默认的测试阶段迭代次数，分别为 4 和 30。对于我们的方法，我们使用 4 个 IterationModule，即 *4* 次内部迭代或 *1* 次外部迭代。对于 PCN，点到干净表面的平均距离为 0.014（以包围球半径为单位）；对于 ScoreDenoise，该值为 0.012，二者都远高于我们方法的平均值 0.007。对于使用 PCN 和 ScoreDenoise 滤波的点云，在测试阶段经过多次滤波迭代后，更多滤波点仍远离底层干净表面；而我们的方法经过一次外部迭代后，便能将点滤波至更靠近表面的位置。

4. 真正的迭代式点云滤波

本节将详细介绍我们提出的**IterativePFN** 网络，该网络由一系列编码器-解码器对组成。每一对称为 **IterationModule**，用于建模单次内部滤波迭代，并对应于 ScoreDenoise 或 PCN 等方法在测试阶段的一次迭代。在每次内部迭代中，相应的 IterationModule 接收一组含噪点的局部块，并构建一个有向图。随后，Dynamic EdgeConv 层对该局部块及其关联图进行处理，为顶点生成潜在表示，以回归该局部块中每个点的滤波位移。为建模 T 次迭代的效果，我们使用 T 个迭代模块。一次外部迭代表示使用网络中的所有迭代模块对点云进行滤波的效果。在推理过程中，我们对含噪点云进行最远点采样，以生成输入补丁。然而，这会导致不同补丁之间出现重叠区域。因此，我们提出一种通用的补丁拼接方法，以恢复最佳滤波点。

4.1. 图卷积与网络架构

基于图卷积的神经网络已被证明能够生成三维点的丰富潜在表示

基于图的 [26, 39]。具体而言，Pistilli 等人 [26]的工作重点研究了用于点云滤波的图卷积。除生成丰富的特征表示外，与使用受 PointNet 启发的骨干网络（如 PCN 和 Pointfilter ）的点集卷积方法相比，图卷积在效率和更短的推理时间方面也具有优势。这些方法经过设计，可以输入由点组成的补丁，对单个中心补丁点进行滤波，因此处理速度较慢。在基于图卷积的方法中，图内的所有顶点都会被同时处理，并为图内的每个单独顶点（点）推断滤波位移。对于我们的 IterativePFN 网络，我们在每个编码器中使用 Wang 等人 [39]，提出的动态边卷积层的修改形式，以生成输入图的丰富特征表示。形式化地，顶点特征更新为 $\mathbf{h}_i^{l+1} = f_{\mathbf{\Phi}}(\mathbf{h}_i^l) + \sum_{j:(i,j)\in\mathcal{E}} g_{\mathbf{\Theta}}(\mathbf{h}_i^l \| \mathbf{h}_j^l - \mathbf{h}_i^l)$，其中$i$是图上的一个顶点，$(i, j)$构成一条边，而$(* \| *)$表示拼接。这里，$f_{\mathbf{\Phi}}: \mathbb{R}^{F^l} \to \mathbb{R}^{F^{l+1}}$ 和$g_{\mathbf{\Theta}}: \mathbb{R}^{F^l} \times \mathbb{R}^{F^l} \to \mathbb{R}^{F^{l+1}}$由多层感知机参数化。$F^l$ 是第 l 层的特征维度。随后，由 4 个全连接（FC）层组成的解码器会使用这些潜在特征，推断每次迭代中的位移。

4.2. 图构建

给定干净点云 $\mathcal{P}_\mathcal{Y} = \{\boldsymbol{y}_i \mid \boldsymbol{y}_i \in \mathbb{R}^3,\ i = 1, ..., n\}$，通过添加标准差为 σ_0 的高斯噪声来扰动点，该标准差介于包围球半径的 0.5% 到 2% 之间。因此，含噪点云表示为 $\mathcal{P}_\mathcal{X} = \mathcal{P}_\mathcal{Y} + \sigma_0\xi$, $\xi \sim \mathcal{N}(0, I)$。随后，给定参考点 $\boldsymbol{x}_r \in \mathcal{P}_\mathcal{X}$，我们获得局部块 $\mathcal{X}$，使得 $\mathcal{X} = \{\boldsymbol{x}_i \mid \boldsymbol{x}_i \in \mathcal{P}_\mathcal{X} \wedge \boldsymbol{x}_i \in kNN(\boldsymbol{x}_r, \mathcal{P}_\mathcal{X}, k = 1000)\}$，其中 $kNN(\boldsymbol{x}, \mathcal{P}_\mathcal{X}, k = m)$ 表示点集 $\mathcal{P}_\mathcal{X}$ 中点 $\boldsymbol{x}$ 的 m 个最近邻。具体而言，我们选取$\boldsymbol{x}_r$的 1000 个最近邻。相对于 $\mathcal{X}$的对应干净补丁 $\mathcal{Y}$ 表示为 $\mathcal{Y} = \{\boldsymbol{y}_i \mid \boldsymbol{y}_i \in \mathcal{P}_\mathcal{Y} \wedge \boldsymbol{y}_i \in kNN(\boldsymbol{x}_r, \mathcal{P}_\mathcal{Y}, k = 1200)\}$。$\mathcal{X}$ 的边界点可能源自干净表面上位于 $\mathcal{Y}$ 中 $\boldsymbol{x}_r$ 的 1000 个最近邻之外的点；这些点可能由于噪声而被扰动至 $\mathcal{X}$ 上。为考虑这一点，我们遵循 ScoreDenoise 的策略，将 $\mathcal{Y}$ 中的点数设为略大于 $\mathcal{X}$。之后，将局部块以参考点为中心，即 $\mathcal{X} = \mathcal{X} - \boldsymbol{x}_r$ 和$\mathcal{Y} = \mathcal{Y} - \boldsymbol{x}_r$。随后，使用局部块点的索引构建由顶点和边组成的有向图 $\mathcal{G} = (\mathcal{V}, \mathcal{E})$，其中$\mathcal{V} = \{i \mid \boldsymbol{x}_i \in \mathcal{X}\}$和 $\mathcal{E} = \{(i, j) \mid \boldsymbol{x}_i, \boldsymbol{x}_j \in \mathcal{X} \wedge \boldsymbol{x}_j \in kNN(\boldsymbol{x}_i, \mathcal{X}, k = 32)\}$。对于每个顶点，将其与 32 个最近邻相连形成边。

4.3. 训练期间基于位移的迭代滤波

尽管以往基于位移的方法（如 PCN 和 Pointfilter）试图推断出将噪声点直接移至干净补丁所需的位移——

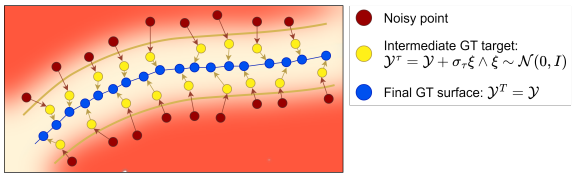

图 4。在中间滤波迭代过程中使用自适应真实值目标，以促进向曲面收敛。

而我们的方法使用多个 IterationModules 迭代地减少噪声。因此，我们提出了一种新的训练目标，旨在逐步减少点云噪声，如图4所示。我们的损失函数为：

$$L_i^{(\tau)}(\mathcal{Y}^{(\tau)}) = \left\| \boldsymbol{d}_i^{(\tau)} - \boldsymbol{\delta x}_i^{(\tau)}(\mathcal{Y}^{(\tau)}) \right\|_2^2, \tag{5}$$

其中 $\boldsymbol{d}_i^{(\tau)}$ （滤波位移）是 IterationModule $t = \tau$，$\boldsymbol{\delta x}_i^{(\tau)}(\mathcal{Y}^{(\tau)}) = NN(\boldsymbol{x}_i^{(\tau-1)}, \mathcal{Y}^{(\tau)}) - \boldsymbol{x}_i^{(\tau-1)}$的输出，而 $NN(\boldsymbol{x}_i^{(\tau-1)}, \mathcal{Y}^{(\tau)})$ 是$\boldsymbol{x}_i^{(\tau-1)}$在 $\mathcal{Y}^{(\tau)}$中的最近邻。此外，$\mathcal{Y}^{(\tau)} = \mathcal{Y} + \sigma_\tau\xi \wedge \xi \sim \mathcal{N}(0, I)$ 是迭代 $t = \tau$时的**自适应真实值**局部块。与 $\mathcal{Y}^{(\tau-1)}$相比，该目标局部块的噪声较少。我们确保 $\sigma_0 > \sigma_1 > \cdots > \sigma_T$，其中我们设定 $\sigma_T = 0$。也就是说，$\mathcal{Y}^{(T)} = \mathcal{Y}$。为简单起见，我们设定 $\sigma_{\tau+1} = \sigma_\tau/\delta$，其中 $\delta = 16/T$ 是控制目标局部块噪声尺度的衰减超参数，$12 \geq T \geq 2$ 和 $T - 2 \geq \tau \geq 0$。更准确地说，我们通过最小化滤波位移与同一底层干净表面的较低噪声版本中的最近邻之间的损失函数，来区别于 PCN 和 Pointfilter 的训练目标。

4.4. 广义补丁拼接



图 5. 沿曲面使用最远点采样构建的最近邻点补丁。

在推理过程中，我们使用最远点采样来获取 R 参考点 $\{\boldsymbol{x}_r\}_{r=1}^R$，并以这些点为中心构建输入补丁。与 PCN 和 Pointfilter 等方法不同，在这些方法中，每个补丁仅用于滤波一个中心点；我们会同时滤波补丁中的所有点。如图5所示，补丁之间可能存在重叠区域，其中包含重复点。因此，我们必须找出能够最佳滤波这些重复点的补丁。

Zhou 等人提出了补丁拼接机制，用于在这些区域中获取最佳的滤波点 [45]。然而，该方法依赖于通过自注意力模块获得的网络推断权重，以推断出能产生最佳滤波点的最佳补丁。但由于依赖自注意力模块，该方法计算开销较大，也难以推广到不使用此类模块的方法。因此，我们希望设计一种更通用的补丁拼接方法，使其能够与任何基于图的神经网络架构结合使用。我们发现，靠近补丁边界的点，其滤波结果不如靠近参考点的点 $\boldsymbol{x}_r$，而参考点位于中心。这是因为靠近补丁边界的点具有不对称的邻域，因此在图卷积过程中会接收到有偏的信号，从而影响其滤波输出。因此，一种直观的策略是根据输入点与参考点的距离为其赋予不同权重 $\boldsymbol{x}_r$ ，依据

符合高斯分布， $w_i = \exp(-\|\boldsymbol{x}_i - \boldsymbol{x}_r\|_2^2 / r_s^2) \sum_i \exp(-\|\boldsymbol{x}_i - \boldsymbol{x}_r\|_2^2 / r_s^2)$，其中 r_s 为支持半径，我们根据经验将其设为$r_s = r/3$ ，其中 r 为补丁半径。

4.5. 迭代滤波与补丁拼接

现在， $t = \tau$ 处的损失函数定义为 $L^{(\tau)} = \sum_i w_i L_i^{(\tau)}$ ，其中每个点的单独损失贡献为 $L_i^{(\tau)}$ ，根据公式（5）。最后，我们对各次迭代的损失贡献求和，得到最终损失函数，

$$\mathcal{L}_a = \sum_{\tau=1}^{T} L^{(\tau)}, \tag{6}$$

该损失函数用于训练我们的网络。它的特点是真正采用了迭代式的训练时滤波方案，在测试阶段可以处理含噪补丁并对点进行滤波，无需依赖多次外部迭代。

5. 实验结果

5.1. 数据集与实现

我们遵循 ScoreDenoise [24] 和 PDFlow [25]，并使用 PUNet [42]数据集训练我们的网络。我们重新训练了所有其他方法，包括 PCN [30], GPDNet [26], DMRDenoise [22]和 Pointfilter [44]。RePCDNet 的实现 [7] 无法获取，因此未纳入比较。我们使用泊松盘采样对 40 个训练网格进行采样，采样分辨率为 1 万、 3 万和 5 万个点，由此为训练集生成 120 个点云。随后，向每个点云添加高斯噪声，其标准差范围为包围球半径的 0.5% 至 2%。测试时，我们采用 20 个测试网格，采样分辨率为 1 万和 5 万个点，与 [24, 25]相似。这 40 个点云受到高斯噪声扰动，噪声标准差分别为包围球半径的 1%、2 % 和 2.5%。

对于真实世界扫描的比较，我们考察以下数据集上的测试结果：Paris-Rue-Madame 数据库包含由移动激光扫描系统 L3D2 [34], 采集的扫描数据，记录了巴黎第六区 Rue Madame 街道的情况。该数据库包含真实世界中的噪声伪影，这些伪影源于扫描技术的局限性，因此为比较真实世界数据上的性能提供了理想基础。由于没有可用的真实值点云和网格，我们仅展示定性结果。接下来，我们考察 [38] 的 Kinect v1 和 Kinect v2 数据集，其中分别包含使用微软 Kinect v1 和 Kinect v2 相机采集的 71 个和 72 个真实世界扫描。我们使用倒角距离（CD）[10]和点到网格距离（P2M） [18] 这两项评估指标比较所有方法的性能。两项指标均使用 PyTorch3D [31]的最新实现进行计算。

实现。我们的 IterativePFN网络使用 PyTorch 1.11.0 和 CUDA 11.3，在 NVIDIA A100 GPU 上训练。我们使用 Adam 优化器，以 1×10^{-4} 的学习率训练网络 100 个 epoch。为确保测试时间的比较公平，所有方法均在 NVIDIA GeForce RTX 3090 GPU 上进行测试。

5.2. 合成数据上的结果

表 1 和图 6给出了各方法在合成数据上的比较结果。基于位移的基线方法 PCN 和 GPDNet 在低分辨率和高分辨率下的表现都不够理想。具体而言，PC N 对高噪声较为敏感，滤波后的点云会出现收缩。基于重采样的 DMRDenoise 和基于得分匹配的 ScoreDenoise 都对噪声较为敏感。基于归一化流的 PDFlow 在低分辨率下表现良好，但在高分辨率和高噪声下表现不佳。我们的方法在所有分辨率和噪声尺度下均具有优势。Pointfilter 同样能够很好地泛化到不同分辨率和噪声尺度，因为它采用了受加权双边滤波启发、利用真实法线信息的损失函数。然而，与我们的方法相比，其表现仍稍逊一筹。此外，对于图 6 所示的铸件形状等复杂形状，Pointfilter 的加权损失会导致相邻很近的曲面塌缩到同一个曲面上。

5.3. 扫描数据上的结果

图 7展示了 RueMadame 数据库中两个场景的滤波结果。由于其中仅包含含噪扫描数据，我们只考虑视觉结果。如场景 1 的结果所示，只有我们的方法能够持续对路牌杆和车辆引擎盖等表面进行滤波。PDFlow 留下了许多噪声伪影，而 ScoreDenoise 的表现仅略好一些。同样，Pointfilter 也留下了噪声伪影，而且需要花费大量时间才能完成场景滤波。这是因为它们采用的滤波策略需要以**每**个

方法	1万个点						5万个点					
	1%噪声		2%噪声		2.5%噪声		1%噪声		2%噪声		2.5%噪声	
	CD	点到网格CD		点到网格CD		点到网格CD		点到网格CD		点到网格CD		P2M
含噪	36.9	16.03	79.39	47.72	105.02	70.03	18.69	12.82	50.48	41.36	72.49	62.03
PCN	36.86	15.99	79.26	47.59	104.86	69.87	11.03	6.46	19.78	13.7	32.03	24.86
GPDNet	23.1	7.14	42.84	18.55	58.37	30.66	10.49	6.35	32.88	25.03	50.85	41.34
DMRDenoise	47.12	21.96	50.85	25.23	52.77	26.69	12.05	7.62	14.43	9.7	16.96	11.9
PDFlow	21.26	6.74	32.46	13.24	36.27	17.02	6.51	4.16	12.7	9.21	18.74	14.26
ScoreDenoise	25.22	7.54	36.83	13.8	42.32	19.04	7.16	4.0	12.89	8.33	14.45	9.58
Pointfilter	24.61	7.3	35.34	11.55	40.99	15.05	7.58	4.32	9.07	5.07	10.99	6.29
Ours	**20.56**	**5.01**	**30.43**	**8.45**	**33.52**	**10.45**	**6.05**	**3.02**	**8.03**	**4.36**	**10.15**	**5.88**

表 1. PUNet 数据集上的滤波结果。CD 和点到网格距离乘以 10^5。

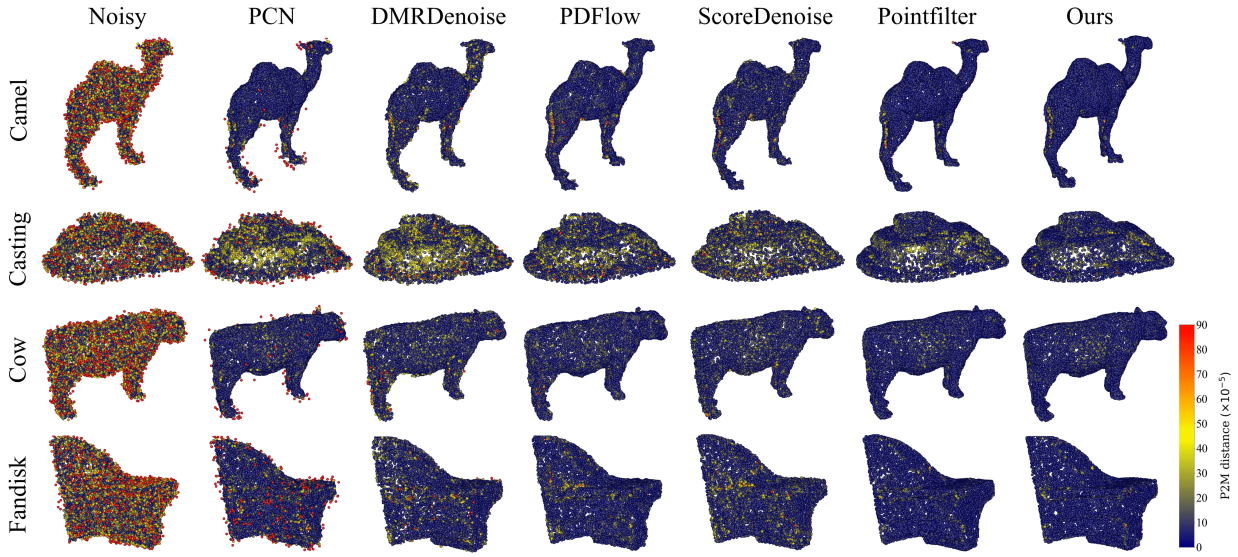

图 6. 对于 50K 分辨率形状，在高斯噪声为包围球半径的 2% 时，逐点 P2M 距离的可视化结果。

个局部块中心点为基础（OPPP）。因此，需要为每个点构建局部块，效率较低。有关各方法运行时间的详情，请参阅补充文档。表 2 展示了 Kinect v1 和 v2 数据集上的结果。我们的方法采用较小的局部块尺寸 100，并使用边界位于 32 k个最近邻之间的设置，在 CD 指标上取得最佳结果，在 P2M 指标上取得第二佳结果。补充材料中提供了合成数据和扫描数据上的更多结果。

5.4. 消融研究

本节考察以下因素对我们方法的重要性：1）IterativePFN 网络中用于建模迭代次数的迭代模块数量；2）真正的迭代滤波所产生的影响；3）基于式（7）给出的固定真实值目标，与基于式（5）和式（6）给出的自适应真实值目标所产生的影响。基于固定真实值目标的损失函数定义如下：

$$\mathcal{L}_b = \sum_{\tau=1}^{T} \left[\sum_i w_i \left(\left\| \boldsymbol{d}_i^{(\tau)} - \boldsymbol{\delta x}_i^{(\tau)}(\mathcal{Y}) \right\|_2^2 \right) \right], \qquad (7)$$

其中 $\boldsymbol{\delta x}_i^{(\tau)}(\mathcal{Y}) = NN(\boldsymbol{x}_i^{(\tau-1)}, \mathcal{Y}) - \boldsymbol{x}_i^{(\tau-1)}$。表 3展示了迭代次数对滤波的影响。尽管 8 次迭代，即 8 个 IterationModules，在 1% 和 2% 噪声下取得了最佳结果，但与 4 次迭代相比，它在未见过的 2.5% 噪声下泛化能力较差。这是因为较多的 IterationModules 使网络专门适应了 0.5% 至 2% 的训练噪声尺度。此外，8 个 IterationModules 会导致更大的网络规模和更高的内存使用量。对于 1、2、4、8 和 12 次迭代，运行时内存消耗分别为 3.9GB、7.6GB、15GB、29.5GB 和 44.1GB。因此，我们选择 4 次迭代作为最优迭代次数。此外，为了确认真正迭代滤波的有效性，我们训练了一个包含 1 个 ItM（DPFN）和 6 个动态边卷积层的深度网络，

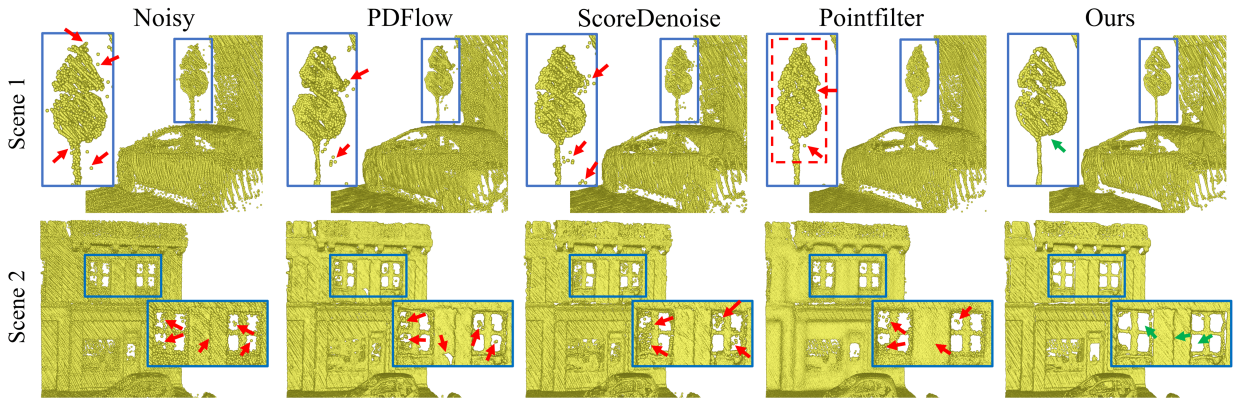

图 7. 真实 RueMadame 数据集中两个场景的结果。绿色和红色箭头分别用于标示滤波准确和不准确的区域。PDFlow 和 ScoreDenoise 留下了大量离群点，而 Pointfilter 则扭曲了原有形状。

方法	Kinect v1		Kinect v2	
	CD	P2M	CD	P2M
含噪	14.49	9.32	22.63	13.39
PCN	13.73	8.75	22.48	13.29
GPDNet	14.83	8.69	23.09	11.78
DMRDenoise	22.78	12.89	-	-
PDFlow	13.34	8.69	19.87	11.69
ScoreDenoise	13.22	8.18	19.66	11.08
Pointfilter	13.77	**7.91**	18.85	**10.29**
我们的方法 ($\lvert\mathcal{X}\rvert = 1000$)	14.07	9.16	19.68	11.69
我们的方法 ($\lvert\mathcal{X}\rvert = 100$)	**13.2**	8.43	**18.69**	10.92

表 2. Kinect v1 和 Kinect v2 数据集上的结果。倒角距离和点到网格距离乘以 10^5。

消融实验	1万个点					
	1%噪声		2%噪声		2.5%噪声	
	CD	点到网格倒角距离		点到网格CD		P2M
$\mathcal{L}_a$ & 1 it.	21.95	5.42	32.38	9.55	36.98	12.71
$\mathcal{L}_a$ & 2 it.	21.13	5.14	30.82	8.67	34.33	11.0
$\mathcal{L}_a$ & 4 it.	20.56	5.01	30.43	8.45	**33.52**	**10.45**
$\mathcal{L}_a$ & 8 it.	**19.78**	**4.9**	**30.12**	**8.3**	33.88	10.78
$\mathcal{L}_a$ & 12 it.	20.49	5.23	30.64	8.87	34.46	11.25
$\mathcal{L}_a$ & DPFN	21.03	5.05	30.96	8.53	35.2	11.4
$\mathcal{L}_b$ & 4 次迭代。	20.64	5.04	30.59	8.54	34.17	10.87

表3。不同迭代次数和不同损失函数的消融实验结果。倒角距离和点到网格距离乘以 10^5。

其参数数量（3.2M）与包含 4 个 ItM 的 IterativePFN 相同。该变体的性能并不理想，尤其是在高噪声下。这表明真正迭代滤波方法的重要性。最后，我们考察了在训练过程中保持真实值目标固定的影响，即 $\mathcal{L}_b$，其中 $\mathcal{Y}$ 对应于干净补丁。我们发现，自适应真实值目标 $\mathcal{L}_a$ 带来了明显差异，尤其是在未见过的 2.5% 噪声尺度下。补充材料还包含关于以下方面的更多消融研究：1）更高噪声下迭代次数的影响；2）补丁拼接对滤波的影响；以及 3）在每个 ItM 中使用 PointNet++作为编码器骨干网络时的滤波结果。

6. 局限性与未来工作

自适应真实目标依赖于在训练期间添加具有给定分布和噪声尺度（即标准差）的噪声。然而，这意味着噪声分布应易于复现，例如采用具有不同尺度且按递减顺序排列的高斯噪声。未来，将这种迭代方法推广到利用模拟真实世界噪声的含噪数据会很有意义，其噪声分布应类似于真实世界扫描仪（例如激光雷达扫描仪）产生的噪声分布。此外，我们希望将这种由堆叠式编码器-解码器对构成的新型网络架构应用于其他任务，例如点云上采样，因为这些任务可能受益于真正的迭代方法。

7. 结论

本文提出了 IterativePFN，它由多个 IterationModule 组成，能够在内部显式建模迭代滤波过程；这不同于最先进的学习方法，后者仅在测试阶段执行迭代滤波。此外，最先进的方法试图学习滤波位移，直接将噪声点移至底层干净表面，却忽略了中间滤波点之间的关系。我们的 IterativePFN网络采用重新设计的损失函数，在每次迭代中使用自适应真实值目标，以捕捉训练期间各中间滤波结果之间的关系。我们的方法能确保滤波结果更快地收敛到干净表面，整体表现也优于最先进的方法。

参考文献

[1] A. Adamson 和 M. Alexa。点采样胞复形。ACM 图形学汇刊，25：671–680，2006年。2[2] M. Alexa、J. Behr、D. Cohen-Or、S. Fleishman、D. Levin 和 Claudio T. Silva。点集曲面的计算与渲染。IEEE 可视化与计算机图形学汇刊，9：3–15，2003年。1, 2[3] H. Avron、Andrei Sharf、C. Greif 和 D. Cohen-Or。尖锐点集曲面的 L1 稀疏重建。ACM 图形学汇刊，29：135:1–135:12，2010年。3[4] Yizhak Ben-Shabat 和 Stephen Gould。DeepFit：通过神经网络加权最小二乘进行三维曲面拟合。载于《计算机视觉——ECCV 2020》，第20–34页。Springer International Publishing，2020年。2[5] Ruojin Cai、Guandao Yang、Hadar Averbuch-Elor、Zekun Hao、Serge Belongie、Noah Snavely 和 Bharath Hariharan。用于形状生成的梯度场学习。载于《计算机视觉——ECCV 2020》，第364–381页。Springer International Publishing，2020年。3, 4[6] Frederic Cazals 和 Marc Pouget。利用密切喷射的多项式拟合估计微分量。《计算机辅助几何设计》，22(2)：121–146，2005年。2, 11[7] Honghua Chen、Zeyong Wei、Xianzhi Li、Yabin Xu、Mingqiang Wei 和 Jun Wang。RePCD-Net：特征感知循环点云去噪网络。《国际计算机视觉杂志》，130(3)：615–629，2022年。2, 3, 6[8] Julie Digne。基于相似性的点云滤波。2012年 IEEE计算机学会计算机视觉与模式识别会议研讨会，第73–79页，2012年。2[9] Julie Digne 和 C. D. Franchis。点云双边滤波器。《图像处理快报》，7：278–287，2017年。1, 11[10] Haoqiang Fan、Hao Su 和 Leonidas J. Guibas。用于从单幅图像进行三维物体重建的点集生成网络。载于 IEEE计算机视觉与模式识别会议（CVPR）论文集，2017年。6[11] S. Fleishman、Iddo Drori 和 D. Cohen-Or。网格双边去噪。ACM SIGGRAPH 2003 论文集，2003年。2[12] Ga¨el Guennebaud 和 M. Gross。代数点集曲面。载于 SIGGRAPH 2007，2007年。1, 2[13] Hugues Hoppe、T. DeRose、T. Duchamp、J. McDonald 和 W. Stuetzle。无序点的曲面重建。第19届年度计算机图形学与交互技术会议论文集，1992年。11[14] Hui Huang、Dan Li、Hongxing Zhang、U. Ascher 和 D. Cohen-Or。用于曲面重建的无序点云整合。ACM SIGGRAPH Asia 2009 论文集，2009年。1, 2, 11[15] Hui Huang、Shihao Wu、Minglun Gong、D. Cohen-Or、U. Ascher 和 Hongxing Zhang。边缘感知点集重采样。ACM图形学汇刊（TOG），32 12，2013 1 11：–。[16] Youngki Kim、Kiyoun Kwon 和 Duhwan Mun。基于网格偏移的方法：生成增量体，以通过三维打印支持局部受损零件的维护。《机械科学与技术杂志》，35(7)：3131–3143，2021年。1[17] D. Levin。《移动最小二乘的逼近能力》。《数学计算》，67:1517–1531，1998年。2[18] Ruihui Li、Xianzhi Li、Pheng-Ann Heng 和 Chi-Wing Fu。通过解耦细化进行点云上采样。发表于 IEEE计算机视觉与模式识别会议（CVPR）论文集，2021年。6[19]Y. Lipman、D. Cohen-Or、D. Levin 和 H. Tal-Ezer。用于几何重建的无参数化投影。ACM SIGGRAPH 2007 论文集，2007年。1，2[20] Xuequan Lu、S. Schaefer、Jun Luo、Lizhuang Ma 和 Y. He。用于三维几何滤波的低秩矩阵近似。IEEE可视化与计算机图形学汇刊，PP，2020年。1，3[21] Chenxu Luo、Xiaodong Yang 和 A. Yuille。用于自动驾驶的自监督柱体运动学习。发表于 CVPR，2021年。1[22] Shitong Luo 和 Wei Hu。用于点云去噪的可微流形重建。发表于第28届ACM国际多媒体会议论文集，第1330–1338页。计算机协会，2020年。2，3，6，16[23] Shitong Luo 和 Wei Hu。用于三维点云生成的扩散概率模型。发表于 IEEE/CVF计算机视觉与模式识别会议（CVPR）论文集，第2837–2845页，2021年。3[24] Shitong Luo 和 Wei Hu。基于分数的点云去噪。发表于 IEEE/CVF国际计算机视觉会议（ICCV）论文集，第4583–4592页，2021年10月。1，2，3，4，6，13，16[25] Aihua Mao、Zihui Du、Yu-Hui Wen、Jun Xuan 和 Yong-Jin Liu。PD-Flow：一种采用归一化流的点云去噪框架。发表于欧洲计算机视觉会议（ECCV），2022年。1，3，6[26] Francesca Pistilli、Giulia Fracastoro、Diego Valsesia 和 Enrico Magli。学习用于点云去噪的图卷积表示。发表于《计算机视觉——ECCV 2020》，第103–118页。Springer International Publishing，2020年。1，3，5，6[27] R. Preiner、O. Mattausch、Murat Arikan、R. Pajarola 和 M. Wimmer。用于快速L1重建的连续投影。《ACM图形学汇刊》（TOG），33:1–13，2014年。1，2[28] C. Qi、Hao Su、Kaichun Mo 和 L. Guibas。PointNet：用于三维分类与分割的点集深度学习。2017 IEEE计算机视觉与模式识别会议（CVPR），第77–85页，2017年。2，3[29] Charles Ruizhongtai Qi、Li Yi、Hao Su 和 Leonidas J. Guibas。PointNet++：度量空间中点集的深度层次特征学习。发表于《神经信息处理系统进展》，第30卷。Curran Associates, Inc.，2017年。2，3[30] Marie-Julie Rakotosaona、Vittorio La Barbera、Paul Guerrero、N. Mitra 和 M. Ovsjanikov。PointCleanNet：学习对稠密点云进行去噪并移除离群点。《计算机图形学论坛》，39，2020年。1，2，3，4，6[31] Nikhila Ravi、Jeremy Reizenstein、David Novotny、Taylor Gordon、Wan-Yen Lo、Justin Johnson 和 Georgia Gkioxari。

使用 PyTorch3D 加速三维深度学习。arXiv，2020年。6[32] Oussama Remil、Qian Xie、Xingyu Xie、Kai Xu 和 J. Wang。三维形状的数据驱动稀疏先验。《计算机图形学论坛》，36，2017年。3[33] Riccardo Roveri、A. Cengiz ¨ Oztireli、Ioana Pandele 和 Markus Gross。PointProNets：利用卷积神经网络整合点云。《计算机图形学论坛》，37(2):87–99，2018年。2，3[34] Andr´es Serna、Beatriz Marcotegui、Franc¸ois Goulette 和 Jean-Emmanuel Deschaud。Paris-Rue-Madame 数据库：用于城市检测、分割和分类方法基准测试的三维移动式激光扫描仪数据集。发表于 ICPRAM，2014年。6[35] Yang Song 和 Stefano Ermon。通过估计数据分布的梯度进行生成建模。发表于第33届神经信息处理系统国际会议论文集。Curran Associates Inc.，2019年。3[36] Yujing Sun、S. Schaefer 和 Wenping Wang。通过 L0 最小化对点集去噪。《计算机辅助几何设计》，35-36:2–15，2015年。1，3[37] Philipp R. W. Urech、M. Dissegna、C. Girot 和 A. Grêt-Regamey。点云建模作为景观设计与规划之间的桥梁。《景观与城市规划》，203:103903，2020年。1[38] Peng-Shuai Wang、Yang Liu 和 Xin Tong。通过级联法线回归进行网格去噪。《ACM 图形学汇刊》，35(6):文章232，2016年。6[39] Yue Wang、Yongbin Sun、Ziwei Liu、Sanjay E. Sarma、Michael M. Bronstein 和 Justin M. Solomon。用于点云学习的动态图库卷积神经网络。《ACM图形学汇刊》（TOG），2019年。2，3，5[40] X Wen、P Xiang、Z Han、Y Cao、P Wan、W Zheng 和 Y Liu。PMP-Net：通过学习多步点移动路径实现点云补全。发表于2021 IEEE/CVF计算机视觉与模式识别会议（CVPR），第7439–7448页，2021年。16[41]Bekiroglu Yasemin、Bj¨ orkman M˚arten、Gandler Gabriela Zarzar、Exner Johannes、Ek Carl Henrik 和 Kragic Danica。真实机器人采集的用于形状建模与补全的视觉与触觉三维点云数据。《数据简报》，30(105335-)，2020年。1[42] Lequan Yu、Xianzhi Li、Chi-Wing Fu、Daniel Cohen-Or 和 Pheng-Ann Heng。PU-Net：点云上采样网络。发表于IEEE计算机视觉与模式识别会议（CVPR）论文集，2018年。6[43] L. Yu、X. Li、C. W. Fu、P. A. Heng 和 D. Cohen-Or。EC-Net：边缘感知点集整合网络。《计算机科学讲义》第11211卷。Springer 出版社，2018年。3[44] Dongbo Zhang、Xuequan Lu、Hong Qin 和 Y. He。Point-filter：通过编码器—解码器建模进行点云滤波。IEEE可视化与计算机图形学汇刊，27:2015–2027，2021年。1，2，3，6[45] Jun Zhou、Wei Jin、Mingjie Wang、Xiuping Liu、Zhiyang Li 和 Zhaobin Liu。通过局部块拼接快速准确地估计点云法线。《计算机辅助设计》，142，2022年。6，15

[46] Runsong Zhu、Yuan Liu、Zhen Dong、Yuan Wang、Tengping Jiang、Wenping Wang 和 Bisheng Yang。《Adafit：重新思考点云上的基于学习的法线估计》。载于*IEEE/CVF*国际计算机视觉会议（*ICCV*）论文集，第6118–6127 页，2021年10月。2

IterativePFN：真正的迭代式点云滤波补充材料

Dasith de SilvaEdirimuni[1], Xuequan Lu[1], Zhiwen Shao[2],GangLi[1],AntonioRobles-Kelly[1,4],YingHe[3†]信息技术学院，迪肯大学[2]中国矿业大学计算机科学与技术学院[3]南洋理工大学计算机科学与工程学院[4]国防科学技术集团，澳大利亚{dtdesilva, xuequan.lu, gang.li, antonio.robles-kelly}@deakin.edu.au, zhiwen shao@cumt.edu.cn, yhe@ntu.edu.sg

在本文的补充文档中，我们提供以下内容：

A. 合成数据和扫描数据上的更多比较

A.1. Rue Madame 和 Kinect 数据集上的更多视觉比较 A.2. 在含高斯噪声的 PUNet 数据集和 Kinect 数据集上对传统方法的比较 A.3. PUNet 数据集上不同噪声模式的定量比较 B. 关于迭代次数的更多消融实验 C. 关于补丁拼接的消融研究 D. PointNet++基编码器的滤波结果 E. 基于学习的方法的运行时间

A. 合成数据和扫描数据上的更多比较

本节中，我们提供了合成数据和扫描数据上的更多视觉结果和定量结果。

A.1. Rue Madame 和 Kinect 数据集上的更多视觉比较

表 2 和图 7（见正文）分别展示了 Kinect 数据集上的定量结果和 RueMadame 数据集上的可视化结果。为进一步展示我们的方法在真实世界噪声下的鲁棒性，我们在图8中提供了 RueMadame 数据集另外两个场景的可视化结果。此外，我们还提供了 Kinect v2 数据集中 4 个扫描结果的可视化对比，见图9。由于正文篇幅限制，这些内容未能纳入正文。

如图所示。8在 Rue Madame 场景中进行噪声滤波时，与其他方法相比，我们的方法表现明显更好。在场景 3 中，我们的方法对车辆的车顶和车身滤波效果最佳，而其他方法的滤波结果仍残留大量噪声伪影。此外，如图所示。9与最先进的方法相比，我们能够更好地滤波复杂的 Kinect v2 扫描数据，例如男孩模型。正文表2 中的结果也证实了这一点：我们在 CD 指标上取得了最佳结果。这表明，与其他最先进的方法相比，我们的方法生成的滤波后点云在点分布和邻近程度上都更接近 Kinect v2 数据集中的干净点云。

A.2. 高斯噪声和 Kinect 数据集下 PUNet 数据集上传统方法的比较

由于篇幅有限，我们无法在正文论文中展示传统方法的滤波性能对比。在表4 和表5中，我们给出了双边滤波器 [9]，Jet 拟合机制 [6] 和 WLOP 正则化机制 [14]的定量结果。双边滤波器依赖于初始点法线，这些法线是使用主成分分析（PCA）计算得到的 [13]，因为这是一种广泛使用的法线估计技术。WLOP 对含噪点云进行正则化和降采样。随后需要对其进行上采样，为此我们采用 Huang 等人的边缘感知重采样（EAR）机制 [15] 。

总体而言，与基于学习的方法相比，传统方法的表现较差。此外，这些方法需要进行参数调优才能获得最佳结果，而这一过程可能既繁琐又耗时。双边滤波器等方法还依赖初始点法线才能有效滤波，而这些法线的精度会影响最终的滤波结果。

*通讯作者：X. Lu，获 PJ03906.PG00507.F002 资助。†Z. Shao 获得国家自然科学基金资助（编号：62106268）。

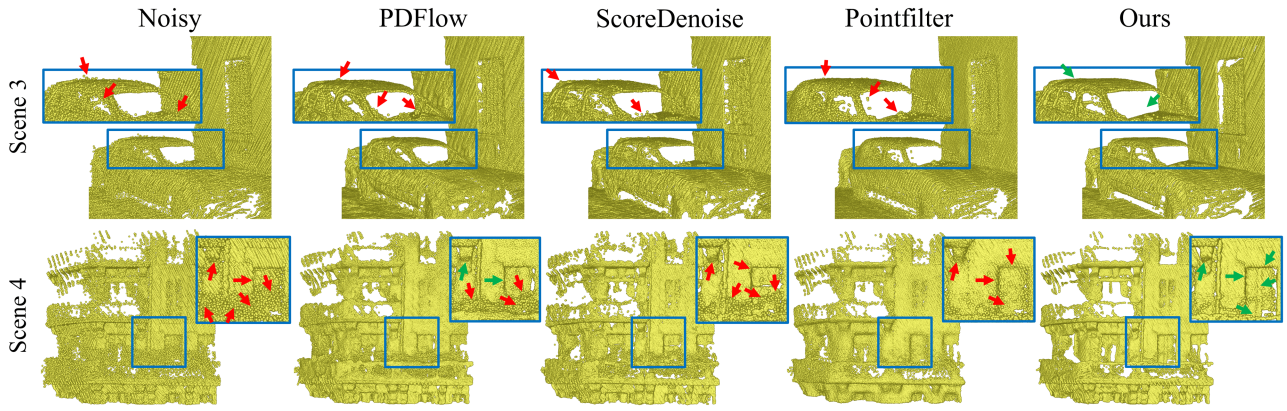

图 8. RueMadame 数据集另外两个场景的可视化结果。绿色和红色箭头分别用于指示过滤准确和不准确的区域。

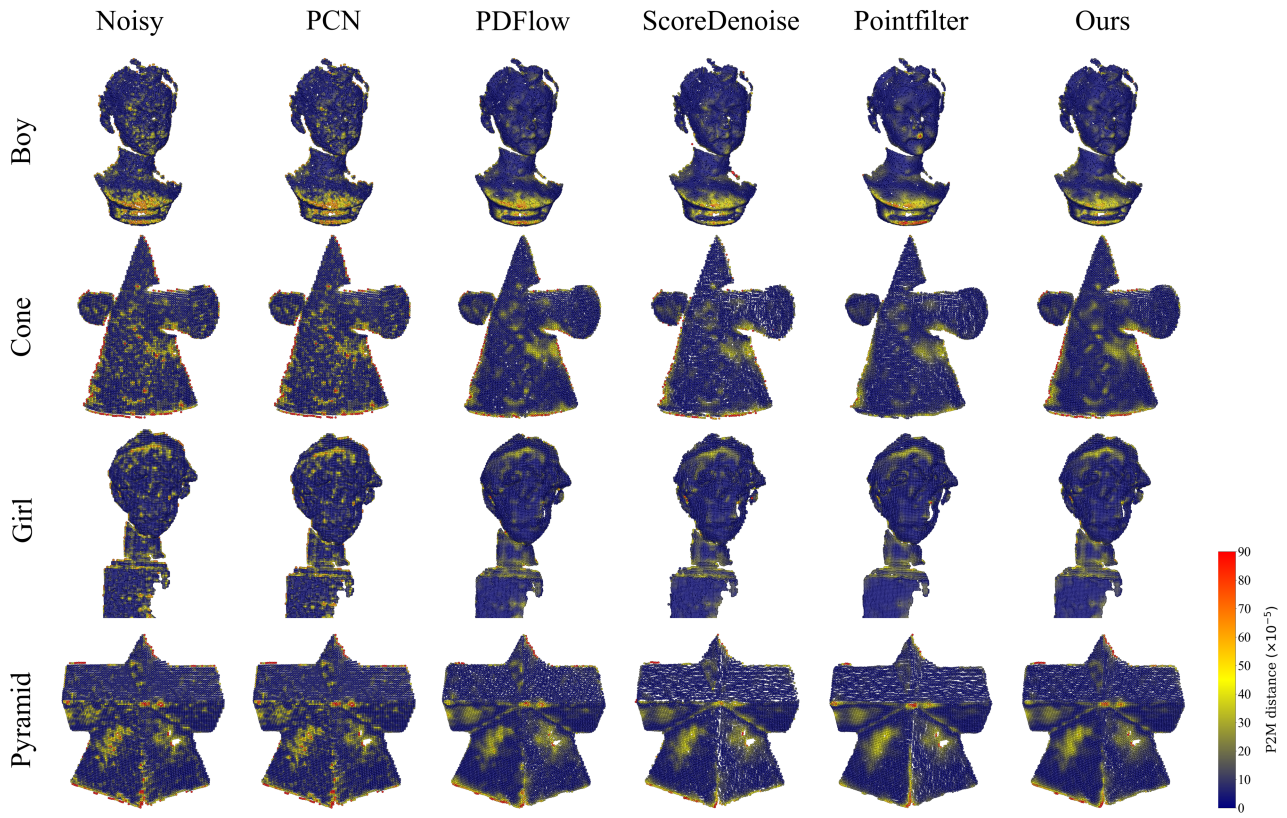

图 9. Kinect v2 数据集 4 次扫描的逐点 P2M 距离可视化结果。

这些法线的精度会影响最终的滤波结果。相比之下，WLOP 在 Kinect v1 数据集上获得了良好的 P2M 结果。这可能是由于重采样过程所致：WLOP 下采样后的输出通过 EAR 机制进行了上采样。尽管这一方法对于拓扑结构相对简单的 Kinect v1 扫描数据效果颇佳，但对于与.....中形状相关的更复杂拓扑结构，

在 PUNet 数据集上，重采样过程的效果较差。如表4所示，由于 WLOP 机制无法准确识别干净表面，且会遗漏几何细节，重采样过程产生的倒角距离和 P2M 值较高。此外，WLOP 在 Kinect v2 数据集上的表现较差，这表明它难以在不同滤波场景中实现良好的泛化。

方法	1万个点						5万个点					
	1%噪声		2%噪声		2.5%噪声		1%噪声		2%噪声		2.5%噪声	
	CD	P2M	CD	P2M	CD	P2M	CD	点到网格CD		P2M	CD	P2M
双边	31.47	14.95	51.94	26.75	73.83	44.31	17.09	13.9	17.98	13.38	25.34	19.55
Jet	31.91	13.42	55.25	31.14	61.51	36.56	7.95	4.39	13.67	8.77	16.68	11.18
WLOP	60.41	30.85	88.59	52.95	109.68	74.29	9.61	6.27	19.94	14.53	28.44	21.66
Ours	**20.56**	**5.01**	**30.43**	**8.45**	**33.52**	**10.45**	**6.05**	**3.02**	**8.03**	**4.36**	**10.15**	**5.88**

表4. 高斯噪声条件下PUNet 数据集上传统方法的滤波结果。倒角距离和点到网格距离均乘以10^5。

方法	Kinect v1		Kinect v2	
	CD	P2M	CD	P2M
双边	13.65	9.27	20.14	12.1
Jet	13.41	8.78	19.82	11.82
WLOP	13.89	**7.65**	33.0	14.79
Ours	**13.2**	8.43	**18.69**	**10.92**

表 5. Kinect v1 和 Kinect v2 数据集上的滤波结果。倒角距离和点到网格距离乘以 10^5。

A.3. PUNet 数据集上不同噪声模式的定量比较

除了高斯噪声外，我们还关注不同噪声模式下的滤波结果比较。本节遵循 ScoreDenoise 的研究，考察各种方法在以下噪声类型下的滤波性能：

非各向同性高斯噪声，其中我们将高斯噪声分布的协方差矩阵设为：

$$\Sigma = s^2 \times \begin{bmatrix} 1 & -1/2 & -1/4 \\ -1/2 & 1 & -1/4 \\ -1/4 & -1/4 & 1 \end{bmatrix} \tag{8}$$

噪声尺度参数 s 设为1%、2% 和2.5% 包围球半径的对应数值。结果见表6和图11。与各向同性高斯噪声的情况类似，正如主论文中的表1所示，我们的方法优于其他方法，并且对这种噪声模式具有良好的泛化能力。

离散噪声服从以下分布：

$$p(\boldsymbol{x}; s) = \begin{cases} 0.1, & \boldsymbol{x} = (\pm s, 0, 0) \text{ or } \boldsymbol{x} = (0, \pm s, 0) \\ & \text{or } \boldsymbol{x} = (0, 0, \pm s), \\ 0.4, & \boldsymbol{x} = (0, 0, 0) \\ 0, & \text{Otherwise} \end{cases} \tag{9}$$

其中 s 设为1%、2% 和2.5% 包围球半径的对应数值。结果见表7 和图12。在不同分辨率和噪声尺度下，我们的方法均优于其他方法。

拉普拉斯噪声，噪声尺度分别设为包围球半径的 1%、2% 和 2.5%。结果如下表 8 和图13。如含噪点云的倒角距离和 P2M 指标结果所示，这种噪声模式的强度相对较高。尽管如此，我们的方法始终优于其他方法，并能从干净的对应点云中恢复出倒角距离和点到网格距离较低的滤波后的点云。

在半径为的三维球体内服从均匀分布的噪声，定义为：

$$p(\boldsymbol{x}; s) = \begin{cases} \frac{3}{4\pi s^3}, & \|\boldsymbol{x}\|_2 \leq s, \\ 0, & \text{Otherwise} \end{cases} \tag{10}$$

其中 s 设为1%、2% 和2.5% 包围球半径的对应数值。结果见表9 和图14。这种噪声模式对应于在半径为 s的球体内均匀采样得到的噪声，并且正如 [24], 所指出的，这是一个非单峰噪声分布的例子，不同于高斯噪声或拉普拉斯噪声。我们的方法再次优于其他最先进的方法。

B. 关于迭代次数的进一步消融实验

在表 10 中，我们研究了在较高噪声设置下迭代次数的影响。所有模型的最大训练噪声为2%，远低于表 10中未见过的高噪声设置。我们发现，对于我们的网络，最佳滤波迭代次数为 4。包含更多 ItM 的变体似乎过度适应了训练噪声尺度，因此在更高的未见噪声尺度下表现欠佳。此外，DPFN 网络仅包含 1 个 ItM，其中有 6 个 Dynamic Edge-Conv 层；而另一种配置包含 4 个 ItM，每个 ItM 各有 4 个 Dynamic Edge-Conv 层。随着噪声增加，前者的表现较差。这一结果表明，多个 ItM 和真正的 迭代滤波至关重要，相较之下，具有相同参数数量的深层单次迭代网络表现不如前者。

C. 关于补丁拼接的消融研究

推理时，给定一个含噪点云，我们使用通过最远点采样确定的参考点集合 $\{\boldsymbol{x}_r\}_{r=1}^{R}$，为每个参考点构建由 1000 个最近邻组成的局部块

方法	1万个点						5万个点					
	1%噪声		2%噪声		2.5%噪声		1%噪声		2%噪声		2.5%噪声	
	CD	P2M	CD	P2M	CD	P2M	CD	P2M	CD	P2M	CD	P2M
含噪	36.17	15.9	78.88	47.75	104.2	69.69	18.55	12.8	50.37	41.38	72.32	62.0
双边	30.92	14.98	52.56	27.77	74.19	45.08	17.08	13.99	18.99	14.52	27.72	22.07
Jet	31.57	13.54	55.39	31.59	60.89	35.89	7.87	4.38	13.84	9.0	17.21	11.74
WLOP	60.02	29.87	92.14	56.69	108.78	70.55	9.67	6.34	20.53	15.13	29.59	22.68
PCN	36.12	15.86	78.74	47.6	104.04	69.53	10.82	6.37	20.49	14.42	34.17	26.98
GPDNet	22.71	7.17	43.34	19.24	59.25	31.68	10.54	6.46	33.09	25.38	51.0	41.67
DMRDenoise	47.8	22.76	50.96	25.26	53.47	27.62	12.05	7.6	14.63	9.9	17.57	12.48
PDFlow	21.03	6.78	32.93	13.76	37.12	17.81	6.53	4.19	13.12	9.6	20.56	15.97
ScoreDenoise	24.79	7.54	37.01	14.09	42.41	19.1	7.11	4.0	13.19	8.62	14.92	9.96
Pointfilter	23.99	7.2	35.29	11.84	41.57	15.71	7.57	4.36	9.64	5.6	12.4	7.51
Ours	**19.94**	**4.94**	**30.3**	**8.6**	**34.39**	**11.23**	**6.02**	**3.05**	**8.3**	**4.6**	**11.52**	**6.99**

表 6. PUNet 数据集在非各向同性高斯噪声下的滤波结果。倒角距离和点到网格距离乘以 10^5。

方法	1万个点						5万个点					
	1%噪声		2%噪声		2.5%噪声		1%噪声		2%噪声		2.5%噪声	
	CD	P2M	CD	P2M	CD	P2M	CD	P2M	CD	P2M	CD	P2M
含噪	11.6	5.87	30.08	12.73	36.99	17.35	6.71	4.44	13.76	10.31	18.36	14.56
双边	19.6	15.33	27.84	15.08	28.78	14.56	15.61	14.38	15.85	14.09	15.74	13.74
Jet	17.82	10.46	38.94	22.82	41.06	23.74	4.57	3.22	8.13	5.67	8.94	6.22
WLOP	59.72	29.85	66.12	35.0	68.2	36.57	8.28	5.31	10.11	6.77	11.32	7.73
PCN	11.64	5.89	30.06	12.71	36.95	17.31	6.21	4.14	8.86	5.95	10.26	7.05
GPDNet	7.88	4.33	19.19	6.34	22.32	7.65	4.6	3.14	7.75	5.29	10.38	7.45
DMRDenoise	48.4	24.25	48.42	23.74	48.43	23.86	11.1	6.9	12.5	7.95	12.68	8.13
PDFlow	8.71	4.64	18.86	6.33	23.37	8.29	4.33	3.04	6.72	4.53	7.38	5.0
ScoreDenoise	12.44	5.32	21.82	7.07	28.81	11.72	4.48	2.91	6.16	3.87	7.32	4.62
Pointfilter	11.14	5.72	20.56	6.92	22.49	7.57	5.43	3.92	6.12	4.16	6.43	4.34
Ours	**6.41**	**3.67**	**16.49**	**4.66**	**18.72**	**5.21**	**3.47**	**2.53**	**4.26**	**2.8**	**4.62**	**2.99**

表7. 在含离散噪声的 PUNet 数据集上的滤波结果。倒角距离和点到网格距离乘以 10^5。

方法	1万个点						5万个点					
	1%噪声		2%噪声		2.5%噪声		1%噪声		2%噪声		2.5%噪声	
	CD	点到网格CD		点到网格CD		P2M	CD	点到网格CD		点到网格CD		P2M
含噪	49.83	25.95	117.73	83.79	162.23	124.83	28.69	22.23	86.64	76.74	127.21	115.91
双边	36.87	17.95	80.15	51.36	112.47	78.99	18.2	14.66	41.25	35.58	71.13	63.96
Jet	35.87	15.64	63.13	38.0	70.8	44.24	9.4	5.43	18.63	13.04	28.33	21.79
WLOP	60.94	31.19	103.49	69.38	126.41	88.96	11.73	7.98	32.5	25.96	52.69	44.16
PCN	49.75	25.87	117.6	83.64	162.1	124.66	13.71	8.89	47.62	40.94	86.11	78.13
GPDNet	28.61	10.13	63.79	36.67	96.58	65.43	15.33	10.53	58.14	49.37	93.35	82.92
DMRDenoise	48.45	23.48	55.12	28.84	59.61	32.72	12.6	8.12	17.62	12.62	23.45	17.98
PDFlow	25.38	8.76	43.19	22.05	51.87	30.12	8.21	5.52	22.76	17.89	45.08	38.08
ScoreDenoise	29.03	9.64	46.02	21.36	51.84	27.3	8.25	4.89	16.77	11.67	18.51	12.79
Pointfilter	27.73	8.61	42.5	16.73	54.66	25.55	8.27	4.85	12.44	7.59	17.54	11.69
Ours	**23.93**	**6.02**	**33.96**	**11.0**	**42.93**	**17.55**	**6.53**	**3.36**	**9.99**	**5.8**	**16.6**	**10.96**

表 8。 PUNet 数据集在拉普拉斯噪声下的滤波结果。倒角距离（CD）和点到网格距离（P2M）均乘以 10^5。

方法	1万个点						5万个点					
	1%噪声		2%噪声		2.5%噪声		1%噪声		2%噪声		2.5%噪声	
	CD	P2M	CD	P2M	CD	P2M	CD	P2M	CD	P2M	CD	P2M
含噪	11.89	6.21	35.01	14.17	44.54	19.62	7.96	4.7	16.93	10.99	22.45	15.55
双边	19.92	15.42	31.4	14.78	33.86	14.46	16.49	14.41	17.25	14.14	17.01	13.6
Jet	18.15	10.55	42.27	22.72	46.06	24.38	5.6	3.28	9.56	5.8	10.42	6.4
WLOP	59.88	29.79	71.06	37.63	67.11	35.71	8.12	5.17	10.63	7.2	12.05	8.22
PCN	11.93	6.22	34.96	14.14	44.48	19.56	7.24	4.25	10.62	6.09	11.81	6.92
GPDNet	7.84	4.31	22.31	6.59	27.25	8.22	5.45	3.24	9.97	5.87	13.66	8.6
DMRDenoise	49.14	24.01	47.69	22.61	48.06	22.67	11.18	6.9	12.23	7.71	12.19	7.72
PDFlow	8.74	4.64	20.26	6.24	24.65	8.38	4.56	3.03	6.82	4.54	7.55	5.09
ScoreDenoise	12.74	5.35	24.67	7.01	31.4	11.68	5.05	2.88	6.91	3.78	8.02	4.74
Pointfilter	11.39	5.7	24.47	6.92	27.76	7.68	6.31	3.96	7.42	4.2	7.63	4.23
Ours	**6.45**	**3.68**	**20.09**	**4.74**	**24.16**	**5.45**	**4.43**	**2.54**	**5.99**	**2.98**	**6.42**	**3.24**

表9. 在 PUNet 数据集上，噪声均匀分布于半径为 s 的三维球体内时的滤波结果。此处，s 对应噪声尺度。倒角距离和点到网格距离乘以 10^5。

消融实验：1万个点	2.75% 噪声		3% 噪声		3.25% 噪声	
	CD	P2M	CD	P2M	CD	P2M
$\mathcal{L}_a$ & 1 次迭代	40.76	15.46	46.33	19.62	53.55	25.36
$\mathcal{L}_a$ & 2 次迭代	37.53	13.2	43.91	17.82	52.8	24.72
$\mathcal{L}_a$ **& 4 次迭代**	**36.31**	**12.3**	**41.87**	**16.44**	**51.2**	**23.6**
$\mathcal{L}_a$ & 8 次迭代	37.44	13.15	45.39	19.1	56.65	27.97
$\mathcal{L}_a$ & 12 次迭代	37.95	13.59	45.14	18.86	55.17	26.79
$\mathcal{L}_a$ & DPFN	39.49	14.48	47.06	20.11	57.35	28.25
$\mathcal{L}_b$ & 4 次迭代	37.62	13.2	44.22	18.13	54.36	25.93

表10. 较高噪声下不同迭代次数和损失函数的消融实验结果。倒角距离和 P2M 距离乘以 10^5。

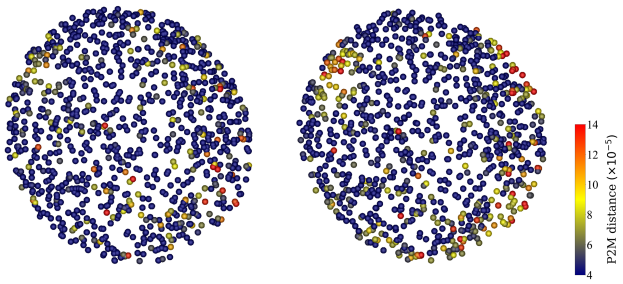

图10. 局部块 (a) 展示了经补丁拼接滤波的含噪局部块，局部块 (b) 则对应同一含噪局部块在未进行补丁拼接时的滤波结果。图中给出了每个点的逐点 P2M 距离。如局部块 (b) 所示，补丁的边界点往往具有较大的 P2M 距离；也就是说，未进行补丁拼接时，远离中心区域的滤波结果准确性较低。然而，靠近中心区域的结果准确性较高。因此，补丁拼接旨在从重叠局部块中恢复最佳滤波点，并减少边界点处的滤波误差。局部块 (a) 展示了采用拼接的情况。可以看到，边界附近的滤波结果更加准确。

消融实验	1万个点					
	1%噪声		2%噪声		2.5%噪声	
	CD	P2M	CD	P2M	CD	P2M
不使用 PS	21.19	5.45	32.38	10.2	38.67	14.98
采用 PS	**20.56**	**5.01**	**30.43**	**8.45**	**33.52**	**10.45**

表 11. 使用和不使用补丁拼接（PS）的消融实验结果。倒角距离和点到网格距离乘以 10^5。

波。通常，对于含有 5万个点的含噪点云，我们会构建 300 个这样的补丁，并对每个补丁进行滤波，同时滤波其中的所有补丁点。由于这 300 个补丁存在重叠区域，同一些点可能会在多个补丁中重复出现，导致某些噪声点对应多个滤波结果。因此，有必要从所有补丁中选出每个点对应最佳滤波结果的最佳滤波点。受 Zhou 等人启发， [45], 我们设计了一种广义补丁拼接机制，在选择最佳滤波点时使用高斯权重。对于噪声点，距离补丁中心参考点越近， $\boldsymbol{x}_r$，其权重就越高。这样设计的依据是，位于补丁边界的点相比靠近中心点的点，滤波效果较差。推理时，对于给定的初始噪声点，我们会在其所属补丁中选取该噪声点权重最高的滤波结果，以此恢复最佳滤波后的点。

图中展示了加入补丁拼接的影响。10在这里，局部块 (b) 未经补丁拼接便进行了滤波。可以看到，许多边界点的逐点 P2M 值较高，表明这些滤波结果离干净表面更远。相比之下，局部块 (a) 使用

方法	1万个点						5万个点					
	1%噪声		2%噪声		2.5%噪声		1%噪声		2%噪声		2.5%噪声	
	CD	点到网格CD		点到网格CD		点到网格CD		点到网格CD		点到网格CD		P2M
含噪	36.9	16.03	79.39	47.72	105.02	70.03	18.69	12.82	50.48	41.36	72.49	62.03
我们的方法 (PN++)	22.9	5.93	33.07	10.28	36.92	13.06	6.18	3.09	9.23	5.22	12.1	7.47
我们的方法 (PN++ w/ GRU)	22.81	5.81	33.15	10.31	37.08	13.17	6.15	3.07	9.19	5.2	12.04	7.44
Ours	**20.56**	**5.01**	**30.43**	**8.45**	**33.52**	**10.45**	**6.05**	**3.02**	**8.03**	**4.36**	**10.15**	**5.88**

表12. 对比采用基于 Pointnet++ 的编码器的 IterativePFNs 与原始基于图卷积的实现，在 PUNet 数据集上的滤波结果。倒角距离和点到网格距离乘以 10^5。

补丁拼接进行滤波后，高 P2M 值的边界点更少。表 11显示，补丁拼接能带来更好的整体滤波效果，尤其是在噪声尺度较高时。

D. 基于 PointNet++ 编码器的滤波结果

我们的动机是，滤波是一个反向马尔可夫过程，会迭代地去除噪声。为了获得滤波后的点 $\boldsymbol{x}_i^\tau$，我们只需要前一次迭代得到的滤波后的点，即 $\boldsymbol{x}_i^{\tau-1}$。因此，只需一个简单的编码器-解码器模块（ItM）即可获得滤波位移。为了验证我们的假设，并展示我们的方法对其他三维点集卷积架构的泛化能力，我们构建了变体 IterativePFNs，其中包含 4 个由基于 PointNet++ 的编码器组成的 ItM:

- **PN++**：一种基础的基于 PointNet++ 的编码器，将我们原始实现中对点云块进行的图卷积替换为使用 PointNet++ 架构对这些局部块直接进行点集卷积。
- **PN++ 采用 GRU**：与之类似，[40], 我们构建了一种基于 PointNet++的变体，在编码器中加入 GRU 层，以保留来自前一次迭代的特征信息作为额外的记忆。

我们最初的网络采用基于DGCNN的编码器，其性能如表12所示更优。在编码器中额外使用GRU层并未带来有意义的性能提升，因为原始PointNet++ 实现的表现与PN++ （使用GRU的网络）相当。这进一步支持了我们的假设：在后续迭代中对一个点进行滤波，$\boldsymbol{x}_i^\tau$是一个反向马尔可夫过程，仅需要滤波后的点 $\boldsymbol{x}_i^{\tau-1}$ （来自前一次迭代）。

E. 基于学习的方法的运行时间

最后，我们比较不同基于学习的方法的运行时间。我们不比较传统方法，因为它们通常需要进行参数调优，因此需要用户

方法	时间（秒）
PCN	167.75
GPDNet	79.87
DMRDenoise	**11.08**
PDFlow	32.74
ScoreDenoise	18.7
Pointfilter	84.32
Ours	22.91

表13. 使用不同的基于学习的方法对含有 2% 高斯噪声的 5万个点点云进行滤波的运行时间。

交互才能获得最佳结果。此外，双边滤波机制以及 WLOP 的上采样步骤，都需要单独计算点法线。这一过程可能相当耗时。我们的方法颇具竞争力，运行时间排名第三，同时在合成数据和扫描数据上都取得了最佳结果。相比之下，DMRDenoise [22], 的运行时间最短，但在滤波任务上的表现通常较差。ScoreDenoise [24], 虽然运行速度相对较快，但滤波结果仍不理想。表 13中提到的 ScoreDenoise 运行时间，对应于默认执行 30 个梯度上升步骤后的滤波结果，即测试时迭代 30 次。然而，滤波结果中仍保留了大量噪声，因此需要额外迭代来进一步降低噪声，这不可避免地会增加运行时间。GPDNet、Pointfilter 和 PCN 的效率最差。对于 Pointfilter 和 PCN，每个中心点对应一个补丁的滤波策略导致运行时间较长，因为每个输入补丁只用于滤波一个中心点。相比之下，我们的方法会同时滤波所有补丁点，并在内部执行迭代滤波；结合我们提出的广义补丁拼接机制，它能在运行时间较短的同时取得最佳滤波结果。

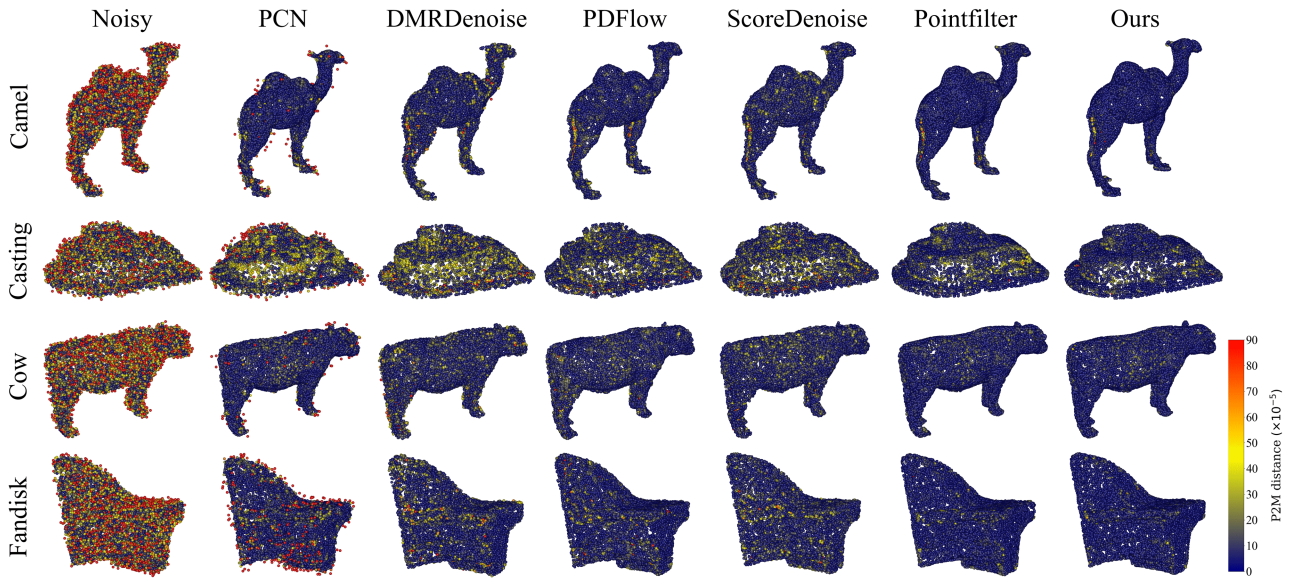

图11. 50K 分辨率形状在非各向同性高斯噪声下，尺度参数为包围球半径的 2% 时的逐点 P2M 距离可视化结果。

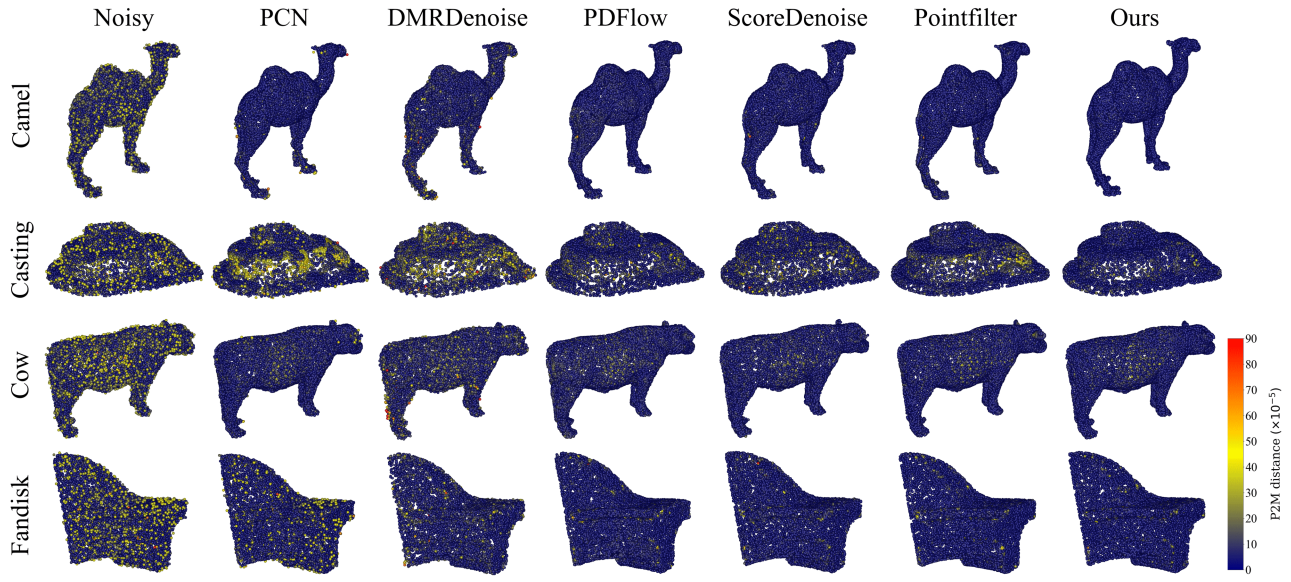

图12. 50K 分辨率形状在离散噪声下，尺度参数为包围球半径的 2% 时的逐点 P2M 距离可视化结果。

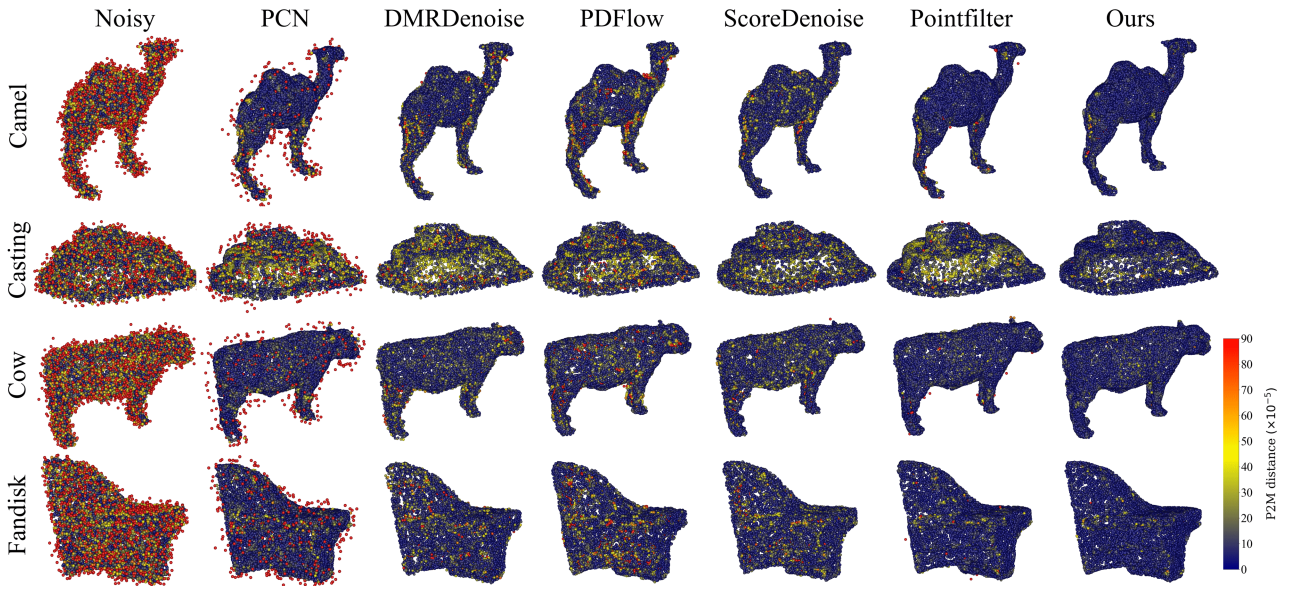

图13. 对包围球半径的 2% 作为噪声尺度的拉普拉斯噪声作用下，50K 分辨率形状的逐点 P2M 距离可视化结果。

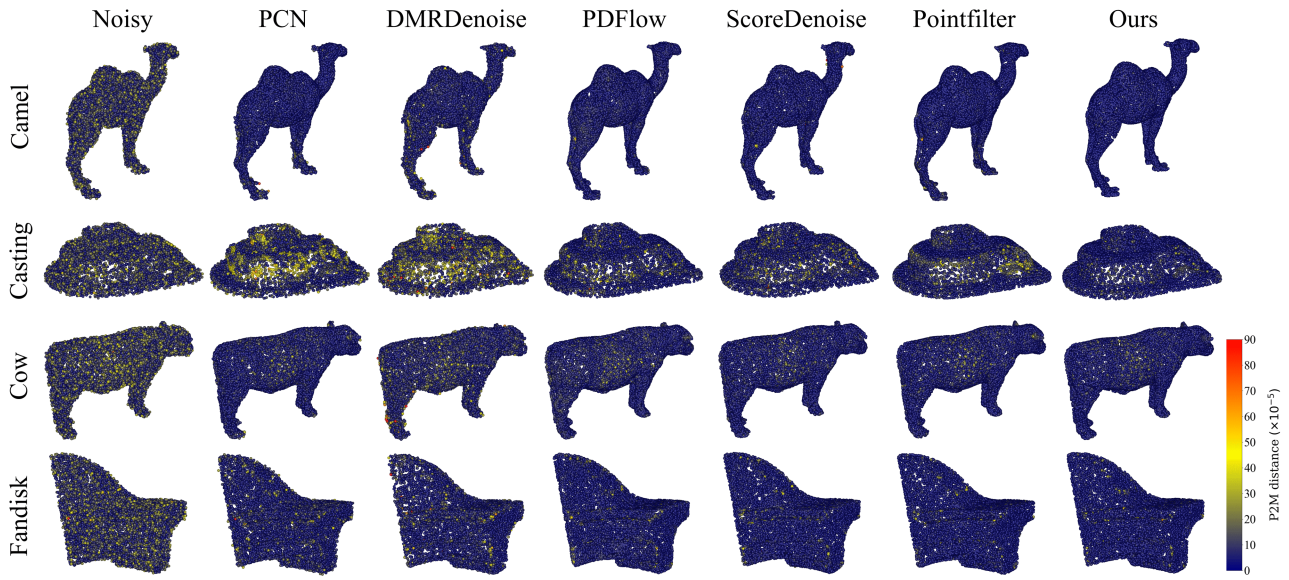

图14. 对半径为 s 的三维球体内均匀分布的噪声作用下，50K 分辨率形状的逐点 P2M 距离可视化结果。其中，s对应噪声尺度，其值为包围球半径的 2%。