小组汇报iMAP: Implicit Mapping and Positioning in Real-Time

论文写作范式思考

第3章方法论部分，System Overview不提，3.2说Implicit Scene Neural Network隐式网络的编码，3.3是深度和颜色的渲染，3.4是误差的定义和优化过程，3.5是关键帧选取策略，3.6是主动采样

为什么MLP可以作为Nerf系统的场景表示？

> NeRF (Neural Radiance Fields) uses Multilayer Perceptrons (MLPs) as the scene representation because MLPs have demonstrated the ability to approximate complex functions, making them suitable for modeling scene geometry and

appearance. Here's why NeRF can effectively use MLPs as the scene representation:

1. **Flexibility in Representations**: NeRF aims to model the volumetric scene representation by learning the 3D radiance field, which describes how light interacts with the scene's geometry and appearance. MLPs are highly flexible and can approximate complex functions, allowing them to capture intricate variations in radiance across space.
2. **Volumetric Function Approximation**: NeRF's goal is to approximate the volumetric scene function that maps 3D spatial coordinates to radiance values. MLPs can be used to learn this function, enabling the generation of novel views of the scene from previously unseen viewpoints.
3. **Continuous Function Modeling**: Scenes often exhibit continuous variations in radiance and appearance. MLPs can model continuous functions, enabling NeRF to capture smooth transitions in radiance values across different spatial coordinates.
4. **End-to-End Learning**: NeRF aims to directly model the radiance field from image data without relying on intermediate representations. Using MLPs allows NeRF to learn the mapping from 3D coordinates to radiance values in an end-to-end manner.
5. **Adaptability to Complex Scenes**: Scenes can have complex geometry and lighting conditions. MLPs can adapt to these complexities by learning from a diverse set of views and images, which makes NeRF capable of capturing fine details and intricate scene structures.

6. **Generalization to Unseen Views**: Once trained, an MLP-based NeRF model can generalize its learned scene representation to generate novel views of the scene from viewpoints that were not present in the training data. This makes NeRF suitable for view synthesis and virtual reality applications.
7. **Implicit Function Modeling**: MLPs are used in NeRF to model implicit functions, where the 3D coordinates are input directly to the network without requiring an explicit surface representation. This approach is advantageous for scenes with non-uniform geometry.
8. **Hierarchical Representation**: NeRF can use a hierarchical approach by stacking multiple MLP layers, allowing it to capture scene representations at different levels of abstraction. This can improve the model's ability to capture both global and local scene features.

It's important to note that while MLPs offer these advantages, NeRF also comes with challenges such as memory and computation requirements for training and rendering. These challenges have led to variations and improvements in the original NeRF architecture, such as NeRF++, PlenOctrees, and more, which aim to address scalability and efficiency issues while maintaining the use of MLPs for scene representation.

Abstract

我们首次展示了**多层感知器（MLP）作为手持RGB-D相机实时SLAM系统中唯一的场景表示**。我们的网络在实时操作中进行训练，无需先验数据，构建了一种**密集的、场景特定的**隐式三维模型，用于**物体占用和颜色的建模，并立即用于跟踪**。通过**对实时**

图像流进行持续训练神经网络以实现实时SLAM需要进行重大创新。我们的iMAP算法采用**关键帧结构**和**多进程计算流程**，具有**动态信息引导**的**像素采样**以提高速度，跟踪频率为10 Hz，全局地图更新频率为2 Hz。相对于标准的密集SLAM技术，隐式MLP的优势包括高效的几何表示、自动细节控制以及对未观测区域（如物体的背面）的平滑、合理的填充。

Introduction

实时同时定位与建图（SLAM）系统对于智能体设备而言，必须逐步构建三维世界的表示，以实现定位和场景理解的功能。理想的表示应能准确编码几何信息，并且具备高效性，根据场景的大小和复杂性自适应地利用可用的内存容量；具备预测能力，能够合理地估计未直接观测到的区域的形状；同时具备灵活性，不需要大量的训练数据或手动调整即可在新场景中运行。

隐式神经表示是最近在离线重建中的一个有希望的进展，它使用多层感知器（MLP）将查询的三维点映射到占据或颜色，并从头开始进行优化以适应特定场景。MLP是一种通用的隐式函数逼近器，能够用少量参数表示可变细节，并且不会产生量化伪影。即使没有先前的训练，网络结构中存在的固有先验使其能够从部分数据中进行完整的几何估计，并对未观测到的区域进行合理的补全。

在本论文中，我们首次展示了多层感知器（MLP）可以作为实时SLAM系统中仅有的场景表示，使用手持式RGB-D相机。我们的网络是随机初始化的，并通过实时操作进行训练，不需要任何先前的训练数据。我们的iMAP系统采用了类似PTAM [11]的关键帧结构和多处理计算流程。在超过10 Hz的跟踪过程中，我们将实时RGB-D观测与MLP场景地图生成的深度和颜色预测进行对齐。同时，映射过程选择并维护一组历史关键帧，这些关键帧的视角覆盖整个场景，并使用它们不断训练和改进MLP，同时联合优化关键帧的位姿。

在跟踪和建图过程中，我们动态采样最具信息量的RGB-D像素，以减少几何不确定性，实现实时速度。我们的系统在Python中运行，所有优化都是通过单个台式机CPU/GPU系统上的标准PyTorch框架 [20] 实现的。

通过将SLAM视为一个连续学习问题，我们实现了一种能够以连续和自适应分辨率高效表示场景的表示方式，并具有出色的插值能力，以实现完整、无缝的重建（图1）。通过约10至20个关键帧和仅有1MB参数的MLP，我们可以精确地对整个房间进行建图。我们的场景表示没有固定的分辨率；关键帧的分布自动实现了高效的多尺度建图。

我们在各种真实世界序列上展示了我们的系统，并对来自室内规模的Replica数据集 [29] 中的8个场景进行了详尽的评估和分析。我们展示了iMAP相较于标准的密集SLAM系统具有更完整的场景重建，并且具有显著较小的内存占用。在TUM RGB-D数据集 [30] 上，我们展示了与最先进的SLAM系统相媲美的跟踪性能。

本文的贡献：

1. **首个使用隐式神经场景表示的密集实时SLAM系统，能够同时优化完整的3D地图和相机位姿。**

> 💬 在传统的SLAM系统中，通常使用特定的地图表示方法，例如占用栅格、特征点或半稠密点云等。这篇论文中，作者首次引入了使用隐式神经场景表示作为地图模型的方法。这种方法使用多层感知器（MLP）将三维点映射到占据或颜色，通过实时操作进行训练，不需要任何先前的训练数据，能够从部分数据中进行完整的几何估计，并对未观测到的区域进行合理的补全。这种地图表示方法具有很好的可伸缩性和表示能力。

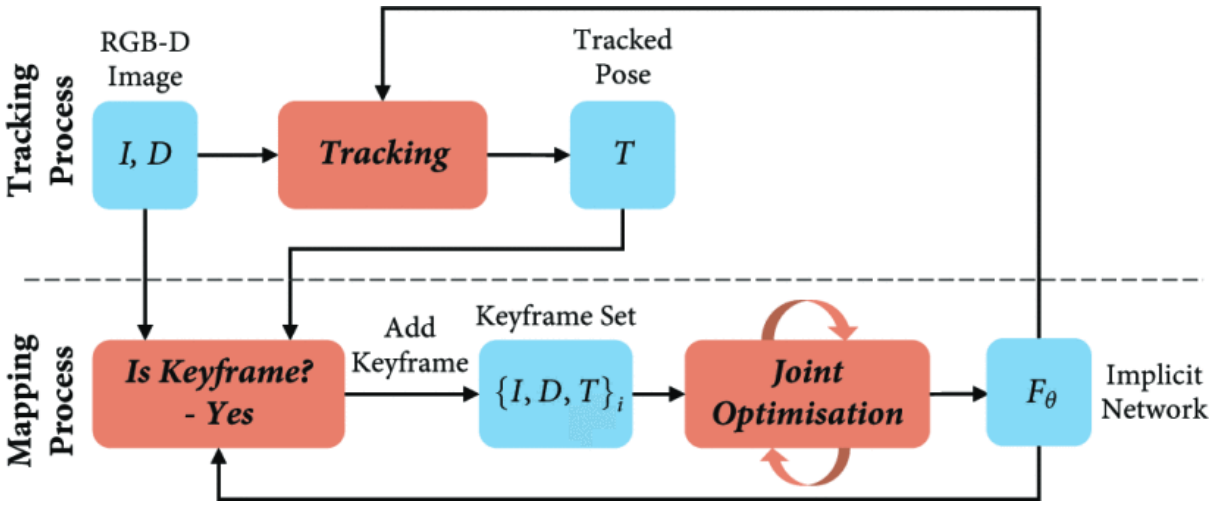
2. **能够在实时环境中逐步训练隐式场景网络，通过自动化的关键帧选择和有损稀疏主动采样来实现。**

> 💬 在他们的系统中，映射过程选择并维护一组历史关键帧，这些关键帧的视角覆盖整个场景，并使用它们不断训练和改进MLP，同时联合优化关键帧的位姿。此外，他们还通过动态采样最具信息量的RGB-D像素，以减少几何不确定性，实现实时速度。

3. **提供了基于 PyTorch 的并行实现，完全适用于手持式RGB-D相机的在线SLAM计算，具备多进程支持。**

> 💬 他们的系统在Python中运行，所有优化都是通过单个台式机CPU/GPU系统上的标准PyTorch框架实现的。他们的系统采用了类似PTAM的关键帧结构和多处理计算流程，这样的架构支持并行计算，使得系统可以在实时环境中运行。

System Construction



3.1 System Overview

上图概述了 iMAP 的工作原理。

- 使用全连接的神经网络 F_θ 表示一个3D体素地图，**将 3D 坐标映射到 颜色 c 和体积密度 d**（第3.2节）。并且给定相机位姿，我们可以通过在反投影的射线中累积网络查询样本，来渲染像素的颜色和深度 （第3.3节）。
- 我们通过对相对稀疏的主动采样测量逐步优化网络权重和相机位姿，从深度和彩色视频中构建场景地图（第3.6节）。两个过程同时运行：
  - 跟踪（第3.4节），它针对当前帧相对于**锁定的**网络 优化位姿；
  - 建图（第3.4节），它联合优化网络和所选关键帧的相机位姿，逐步选择的关键帧基于信息增益（第3.5节）。

3.2. 隐式场景神经网络

我们使用NeRF [15]中的网络架构，

- 使用具有4个隐藏层的MLP，特征大小为256，并且有两个输出头，
- 将3D坐标 p = (x, y, z) 映射到颜色和体积密度值：Fθ(p) = (c, ρ)。
- 与NeRF不同的是，我们不考虑视角方向，因为我们对建模镜面反射不感兴趣。

我们应用了Fourier Feature Networks [32]中提出的高斯位置嵌入方法，将输入的 3D 坐标映射到 n维空间：sin(Bp)，其中 B 是一个从标准差为σ的正态分布中采样得到的 [n×3] 矩阵。这个嵌入作为 MLP 的输入，并且也被连接到网络的第二个激活层。受到 SIREN [27]的启发，我们允许优化嵌入矩阵 B，它被实现为具有正弦激活函数的单个全连接层。

💬 上面提到的高斯位置嵌入方法是一个将输入空间（在这个例子中是3D坐标）映射到更高维特征空间的过程。这种方法的基本步骤如下：

1. 首先，创建一个矩阵B，该矩阵的每个元素都从标准差为σ的正态分布中采样得到。这个矩阵的大小为[n×3]。
2. 然后，将3D坐标p与矩阵B相乘，结果记作Bp。
3. 对Bp的每个元素应用正弦函数，得到嵌入sin(Bp)。

将3D坐标映射到高维空间有一些优点。首先，这可以增加模型的表达能力，使其能够更好地逼近复杂的函数。其次，通过将原始坐标映射到正弦和余弦函数的频率空间，可以创建出具有多尺度特性的表示，这对于处理实际场景中的尺度变化非常有帮助。

3.3 Depth and Colour Rendering

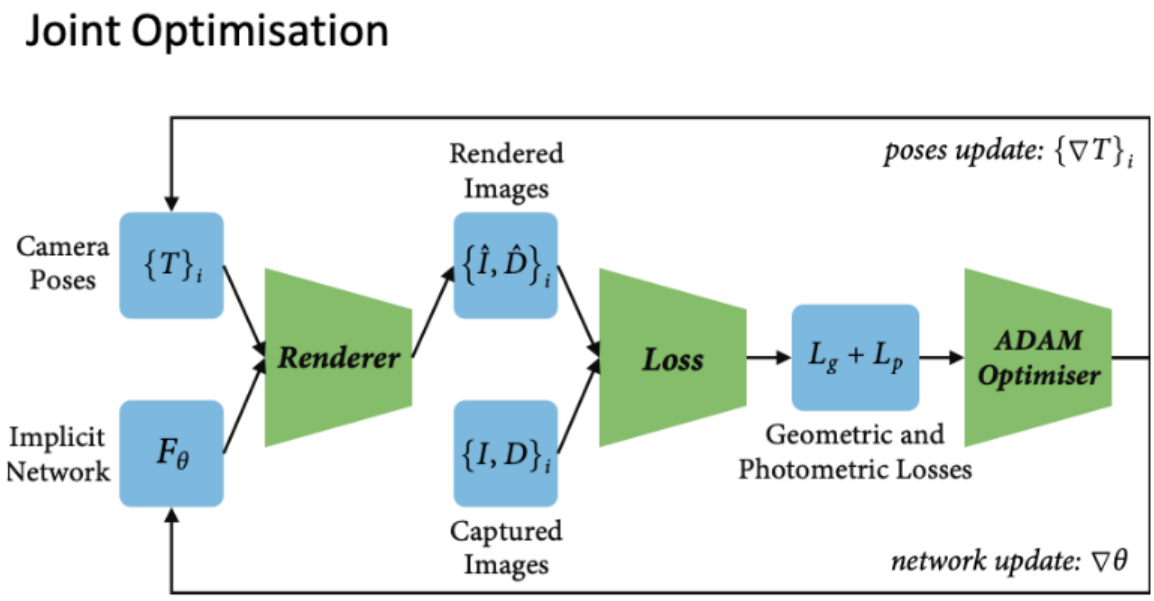
输入：相机位姿，相机坐标，（以及相机内参）

输出：这个像素点的深度和颜色

具体步骤：

1. 根据相机位姿，像素坐标，以及相机内参，获取到视线方向并且反投影到世界坐标系下。这样确定了一条射线
2. 在确定了射线之后，沿着这个射线取 N 个样本点 pi，对应深度值为 {d1, ..., dN}，然后对这些样本点在网络中查询颜色和体积密度 (ci, ρi) = F_θ (pi)。这里使用的分层体积采样策略和 NeRF 当中是一样的
3. 将深度和颜色渲染为期望值，即对所有样本的深度和颜色进行加权平均，权重为射线终止概率

3.4 Joint Optimization



优化的目标：隐式神经网络的参数 θ，不断增长的关键帧的位姿

输入：所有关键帧的 RGB，Depth，Pose （*I, D, T*）

误差函数：深度误差 + 光度误差

优化器：ADAM optimizer

Camera Tracking

这个 tracking 部分的用途：运行一个并行的 tracking 过程，相较于 joint optimization，以更高的帧率，不断优化最新帧的姿态。

重点：

- 相同的 Loss 和 optimizer，但是 θ 固定， 只优化相机位姿
- 更高帧率，应该是逐帧运行
- Tracking 输出的位姿为之后的 mapping 提供了初始值

3.5 Keyframe Selection

某一帧是否为关键帧的判断步骤：

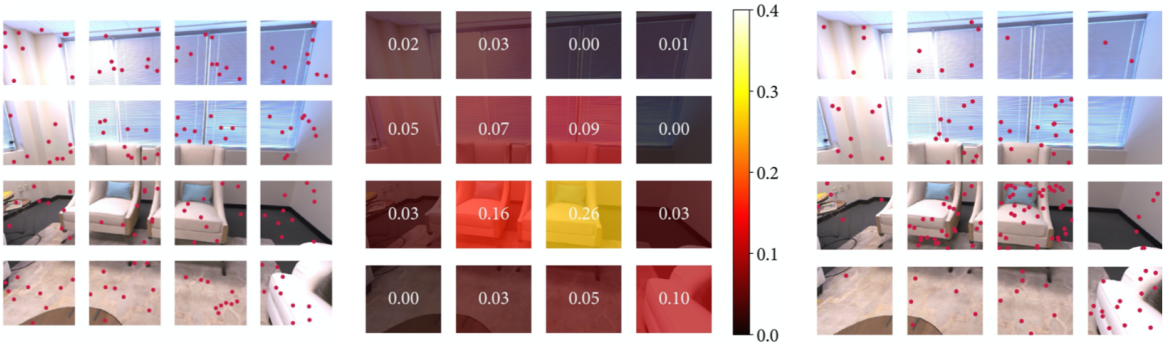
1. 对于当前帧，会渲染出一组均匀的像素样本 s，并根据当前的 MLP 计算这些样本的深度。
2. 比较这些样本的渲染深度与他们的实际深度，计算归一化深度误差

3. 如果该误差小于某个阈值 tD（在这个案例中，tD=0.1），那么就可以认为该像素已经被当前的 MLP 所解释。
4. 对所有样本 s 计算满足上述条件的像素的比例 P，这个比例可以被解释为当前帧已经被我们的地图快照解释的比例。
5. 当这个比例 P 下降到某个阈值 tP 以下（在这个案例中，tP=0.65）时，就认为当前帧看到了显著的新区域，因此将其添加到关键帧集合中。

在建图过程中接收到的每个帧都用于联合优化几次迭代（在10到20次之间），因此我们的关键帧集合始终由所选的关键帧集合以及不断变化的最新帧组成。

3.6 Active Sampling

- 进行优化的时候，如果所有的像素都进行渲染，计算量会非常大
- 利用图像规律性在每次迭代时仅渲染和优化一组非常稀疏的随机像素（每个图像 200 个）
- 基于**图像渲染的 Loss**（深度误差 + 光度误差）进行采样



阶段一

1. 均匀地在每个关键帧的深度和颜色图像上选择一组像素 si 进行采样。这些像素用于更新网络和相机位姿，并计算 Loss。
2. 将每个图像划分为一个8x8的网格，并计算每个网格区域 Rj (j = {1, 2, ..., 64}) 的平均图像渲染的 Loss （公式 7）。
3. 将 Loss 转换为概率分布（公式 8）fi[j]。每个区域的概率代表了在该区域内的平均损失。损失越大，分配到该区域的采样点越多。

阶段二

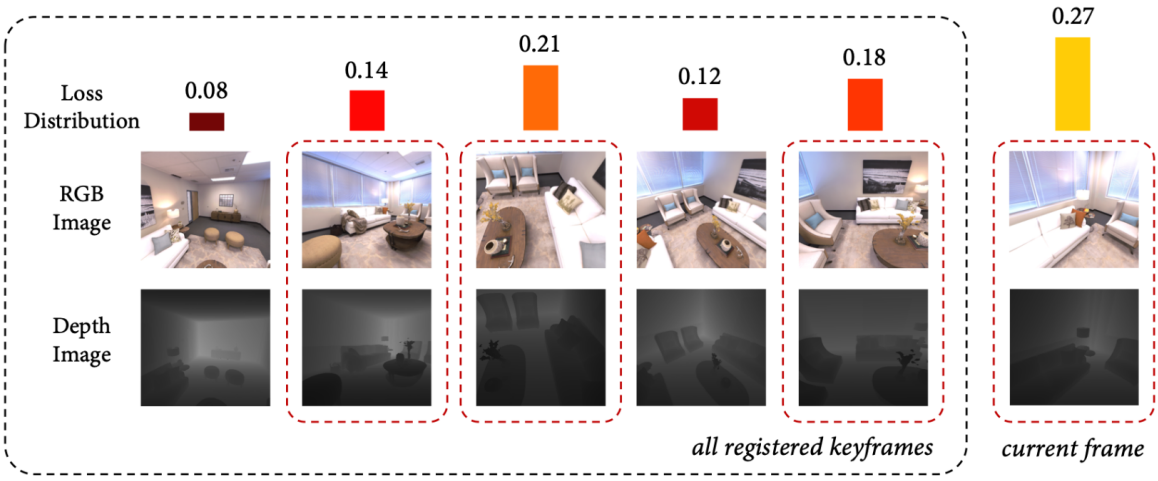
1. 使用上一步计算得到的概率分布，重新采样一组新的像素。

2. 每个区域的采样数量是 ni • fi[j]，其中 ni 是每个关键帧中的总采样数量。

关键帧的主动采样

这些关键帧起到了存储器的作用，帮助避免了网络的遗忘。关键帧主动采样的目标是将更多的样本分配给那些损失较高的关键帧，因为这些帧通常涉及到新探索的区域、细节丰富的区域或者网络开始遗忘的区域。这样做可以确保模型在训练中更多地关注这些区域。

但是在每次训练迭代过程中，**系统只会选取三个关键帧用于训练模型**。这是为了限制训练的复杂性和计算量（实时系统）。这三个关键帧是根据损失分布进行随机采样的，这样可以确保模型不会过度关注某些特定的关键帧，而是在不同的迭代中关注到不同的关键帧。



最后一个关键帧和当前的实时帧始终会被包含在联合优化中，这样可以构成一个不断变化的窗口，其中包含了5帧图像。这种方式既保证了模型在训练过程中能够关注到最新的图像信息，也使模型能够有效地学习场景的不同部分。

结论

论文iMAP提出的方法在多个方面有显著的优势：

1. **实时性能**：iMAP能够实时进行相机追踪和3D场景映射，这使得其在许多实时应用中（如增强现实）变得非常有价值。
2. **高质量的渲染**：iMAP可以生成高质量的渲染图像，提供精确的深度和颜色信息。这是由于其联合优化了相机位姿和神经网络参数，因此在特定的相机视角下，渲染出的图像质量优于单独优化的结果。
3. **鲁棒性**：iMAP在面对视角变化、动态场景等问题时具有鲁棒性，能够有效地适应和处理这些问题。

然而，这种方法也存在一些限制：

1. **计算资源要求**：尽管iMAP能够实时运行，但是其依赖于强大的计算资源，比如高性能GPU，这可能限制了其在低功耗设备上的应用。
2. **动态场景的处理**：虽然iMAP对动态场景具有一定的鲁棒性，但是如果场景中的动态变化太大或太频繁，可能仍然会影响结果的质量。
3. **难以处理的情况**：对于某些特定的情况，如极度低照度、高反射面或者纹理稀疏的区域，iMAP可能会面临挑战。

关于改进的方向，这主要取决于具体的应用需求和场景。一些可能的方向包括：

1. **优化计算效率**：通过算法优化或硬件加速，以便在有限的计算资源上运行iMAP，使其可以在更广泛的设备上应用。
2. **增强对动态场景的处理能力**：通过引入时间信息或者使用更复杂的动态模型，以更好地处理动态变化。
3. **处理困难场景**：通过结合其他传感器数据（如激光雷达、红外传感器等），或者改进神经网络架构和优化策略，以处理低照度、高反射面或纹理稀疏的问题。
4. **结合语义信息**：将物体识别、语义分割等信息结合进3D映射和渲染，可以提供更丰富的场景描述，帮助理解和处理场景中的复杂性。