论文阅读笔记：NICE-SLAM

Contributions

- 提出了NICE-SLAM，一个密集的RGB-D SLAM系统，具有实时性、可扩展性、预测性和对各种具有挑战性的场景的鲁棒性。
- NICE-SLAM的核心是基于网格的分层神经隐式编码。与全局神经场景编码相比，这种表示方式允许局部更新，这是大场景方法的先决条件。
- 在各种数据集上做了广泛的评估，展示了NICE-SLAM在定位和建图方面的优势。

论文提及的观点

abstract&introduction部分

- 现有方法对场景的重建过于平滑，并且难以扩展到更大的场景；这些限制主要是使用了简单的全连接网络架构，并且does not incorporate local information in the observations.

  > 这里的局部信息是什么意思呢，说的是高频的、比较细节的东西吗？

- NICE-SLAM作者认为，SLAM系统需要具备以下条件：
  - 实时

    - 能对没观测到的区域进行合理预测
    - 扩展到大型场景
    - 足够鲁棒，容忍噪声或者遗漏的观测
- NICE-SLAM作者认为：
    - 传统SLAM方法满足实时性、适用于大场景，但是unable to make plausible geometry estimation
    - 基于学习的SLAM方法有一定程度预测能力，需要在特定任务的数据集上训练，更倾向于处理噪声和异常值，只适用于有多个对象的小场景。
    - 近期的工作iMAP在面对更大的场景时，重建精度和跟踪精度显著下降。其中作者认为iMAP的关键限制因素是使用单一的MLP表示场景。
    - 近期的一些工作（Convolutional occupancy networks，Neuralrecon: Real-time coherent 3d reconstruction from monocular video.）发现建立多层次的基于网格的特征可以帮助保留几何细节、重建复杂场景。但是这些方法是离线的，不具备实时性。
- NICE-SLAM结合了分层场景表示和隐式神经表示，利用层次特征网格表示场景的几何形状和外观，结合不同空间分辨率下预训练的神经隐式解码器的归纳偏差，利用occupancy和color decoder输出渲染的深度和彩色图像。通过最小化re-rendering loss来优化特征网络。

related work部分

Dense Visual SLAM

- NICE-SLAM把地图表示分为两大类：

  > 第一次见，很新奇

    - 3D几何图形锚定到关键帧上，说白了就是深度图，比如DTAM、DSO、TANDEM、DROID-SLAM等工作；
    - 3D几何图形锚定到统一的世界坐标中，说白了就是surfels或者voxel grids，里面存储的是occupancies后者TSDF values，比如KinectFusion等作品。
- NICE-SLAM同样采用体素网格表示，但是存储的是几何图形的隐式编码，在mapping的过程中直接对其优进行优化。

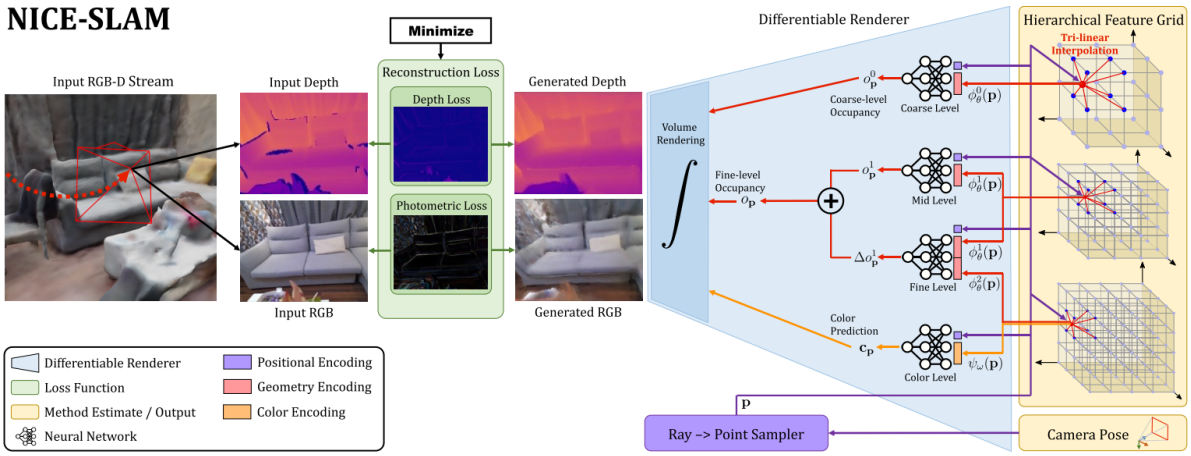
  这种方式允许NICE-SLAM在较低的网格分辨率下实现更精确的几何图形。

Neural Implicit Representations

- 简单罗列了各方面近期的一些工作，并且也说了用RGB-D训练NeRF，但需要位姿，或者解决相机位姿优化问题的工作，但是提到了这样的工作优化时间很长，不能够实时应用；
- 再一次说iMAP只使用一个MLP表示场景，容量有限，无法表示详细的场景几何形状，无法做到精确的相机跟踪。
- NICE-SLAM将可学习的潜在嵌入向量（embeddings）与一个预先训练的连续隐式解码器相结合。这样可以重建复杂的几何结构和预测更大的室内场景的细节纹理，同时保持更少的计算量和更快的收敛速度。

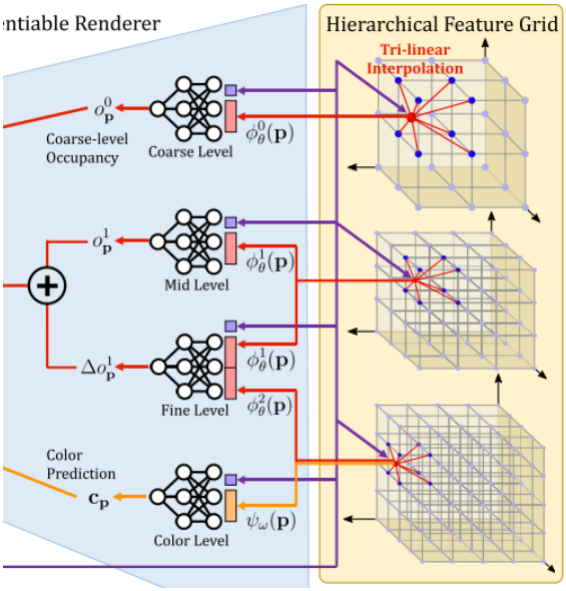
  > 这里究竟是为什么呢？？？不过这里说Local implicit grid representations for 3d scenes和Convolutional occupancy networks也是将传统的网络结构和学习到的特征表示相结合，或许我可以从这里寻找答案?

具体实现



Hierarchical Scene Representation

将多级网格特征和预训练的解码器相结合，用于预测occupancy，对应下面这个区域



需要注意的是，Color Level的解码器是没有预训练的，需要参与优化！这里的公式，**带θ的是几何相关的待优化参数，带ω的是颜色相关的待优化参数**。很明显颜色相关的解码器是g_w，很显然需要进行优化。

Mid-&Fine-level Geometric Representation

文章说观测到的场景用mid-和fine-level的特征网格进行表示。然后重建过程中，先优化mid-level的特征网格，再利用fine-level的特征网格进行细化。

> 所以fine-level也是被优化了？（肯定要优化啊！！！）

具体实现：

- 通常，mid-level的体素网格长度为32cm，fine-level的体素网格长度为16cm

  在TUM RGBD数据集上则为16cm、8cm，看起来大概保持了一个2比1的比例

- 对于mid-level，直接用关联的MLP（从上往下数第二个）将特征解码得到occupancy，这里网络的输入是点p的positional encoding和在mid-level feature grid中通过三线性插值得到的点p处的feature。

- 为了学习到更高频的细节，NICE-SLAM用残差的方式添加了fine-level的特征。如从上往下数第三个MLP所示，该网络输入点p的positional encoding、点p在mid-level通过三线性插值得到的feature、点p在fine-level通过三线性插值得到的feature，输出mid-level的occupancy偏移量（offset）
- 有个非常需要注意的地方，解码器是预训练好的，被固定了。所以**整个优化过程中，只对Feature Grid1&2进行了优化。**NICE-SLAM说这有助于稳定优化过程、保证几何的一致性。

Coarse-level Geometric Representation

coarse-level feature grid的目的是捕捉场景中的high-level几何信息，比如墙壁、地板等，并且coarse-level是独立优化的。它的目标是预测观察到的几何形状之外的近似occupancy。

> 意思是mid-和fine-level是同时一起优化的?

具体实现：

- 边长2m
- 与mid-level一样的处理过程，得到occupancy
- 在Tracking过程中，coarse-level occupancy只用于预测没有观测到的部分场景，这种方式可以保证即使新的一帧很大一部分内容都看不见，但是也能保证Tracking成功。

Pre-training Feature Decoders

- 使用3个预训练的MLP网络进行解码，将网格特征解码为occupancy
- 其中，coarse-level和mid-level的decoder是作为ConvONet的一部分进行训练的（ConvONet由CNN编码器和MLP解码器组成）。

  使用预测值与ground-truth之间的binary cross-entropy loss来训练encoder和decoder，这个过程和ConvONet是一样的。训练好之后只使用decoder的部分，这部分被固定住放到NICE-SLAM框架中，NICE-SLAM通过不断的训练feature grid来适应这个decoder。

  通过上面这样的方式，可以在解码优化的特征时，预训练好的decoder可以利用从训练集学习到的特定分辨率先验。

  > 这里到底是如何使用的？怎么利用的？？？训练好的decoder怎么包含进来先验信息的？？？

- 对于fine-level，使用了和上面同样的预训练过程，但是在输入到decoder之前，会简单的将mid-level和fine-level的特征进行一个拼接。

Color Representation

从这里的描述大概能看出，NICE-SLAM主要还是关注深度，或者说更关注场景的几何形状，主要靠深度进行Tracking和Mapping，这里仅仅提了一句对颜色信息进行编码，可以给Tracking提供额外的信号。

这里的特征网格和解码器都是需要优化的，看他们的符号就知道了。

NICE-SLAM的作者说联合优化颜色特征网格和对应的解码器可以改善跟踪的性能，但是也提到这可能会导致遗忘的问题，因为颜色只具有局部一致性。如果想要可视化整个场景的颜色，可以把这里作为一个后处理的步骤进行全局优化。

> 前半段大概懂他的意思，后半段不太懂怎么进行一个全局优化。

Network Design

对于所有的MLP解码器，使用了宽度32、全连接层数为5的MLP网络。

除了coarse-level，其他地方都采用了可学习的高斯位置编码，这能让NICE-SLAM学习到高频的细节和外观。

> coarse-level确实不需要，毕竟不学习高频信息。不过什么是可学习的高斯位置编码？？？

Depth and Color Rendering

这里用的和NeRF很接近的采样和渲染策略。

> 好熟悉

大概就是，沿着某条射线采样，先类似NeRF的分层采样，之后再在该点的深度附近进行均匀采样。

具体的看原论文就行，这部分内容很少。

Mapping and Tracking

一个线程用于coarse-level的优化

一个线程用于mid-level和fine-level的优化和颜色的优化

一个线程用于Tracking

Mapping

- 从当前帧和选定的关键帧中统一采样一定数量的像素点，以分段方式进行优化来最小化几何和光度损失。
    - 首先利用几何误差优化mid-level的特征网格；
    - 之后利用fine-level的几何误差联合优化mid-level和fine-level的特征网格；
    - 最后进行一个local BA共同优化所选择的K个关键帧的各个级别的特征网格、颜色解码器和位姿；
- 这种多阶段优化方式能够更好的收敛，因为更高分辨率的外观和精细的特征可以依赖来自mid-level已经优化好的几何信息。

Tracking

- 使用了和Mapping一样的光度损失；
- 使用了修改的几何损失，这里和Mapping略有区别。这里的目的是降低几何结构中某些区域的权重，比如物体边缘的权重。

    > 意思就是让Tracking更关心图像中心部分的信息

- NICE-SLAM在这里提到coarse-level的短期预测能力能够让NICE-SLAM在丢帧或者快速相机运动的情况下表现得更加鲁棒，这部分他们也做了实验。

Robustness to Dynamic Objects

NICE-SLAM提到为了在Tracking过程中对动态物体更鲁棒，他们对深度or颜色重渲染损失较大的像素点进行了过滤，当损失超过了中值的10倍时会被剔除。

另外，NICE-SLAM说在动态环境下，联合优化相机位姿和场景表示是非常有意义的工作，是未来的一个研究方向。

Keyframe Selection

类似iMAP的方法维护一个全局的关键帧列表，在这个列表中也根据信息增益添加新的关键帧。但是与iMAP不同的是，iMAP用MLP网络表示场景，所以不得不从所有的关键帧中进行选择和优化；而NICE-SLAM这里是使用网格对场景进行表示的，所以优化场景的几何结构时，是需要包含当前帧和与当前帧有视觉重叠的关键帧。NICE-SLAM这样的方式也避免了iMAP所面对的遗忘问题。这种关键帧选择策略不仅确保了当前视图之外的几何图形保持静态，并且只需要优化必要的参数，非常高效。

具体实现：

- 首先随机采样一定数量的像素点；
- 使用优化后的相机位姿反向投影，拿到这些点的深度；

  > 这里就得到点云了

- 将点云再投影到全局关键帧列表中的每一个关键帧；

  > 有点像重投影误差

- 从有投影点的关键帧中随机选择K-2个关键帧，再选择最近的关键帧和当前帧，所以总共有K个活动帧。

实验与总结

- 实验部分，介绍Geometry Forecast and Hole Filling时，NICE-SLAM提了一句iMAP中没有编码任何场景的先验知识，反过来NICE-SLAM是因为预训练了decoder，所以它的coarse才拥有这样的能力的?
- 总结部分，NICE-SLAM说自己的预测能力仅限于coarse-level，另外不能够进行loop closures，这也是一个未来的研究方向。
- NICE-SLAM与传统方法相比缺乏特性，与学习方法相比存在性能差距。

补充材料

Frustum Feature Selection

基于网格的表示可以只用优化当前视图截锥的几何形状，但是由于三线性插值本身，对所有体素的简单优化会影响到视图截锥之外的特征

> 毕竟来自关键帧的一些射线上需要采样的点并没有老老实实呆在这个视锥之内，而视锥之外采样的点肯定要通过三线性插值从有可能在视锥之外的特征网格处取值，一优化就改变视锥之外的特征了。

解决的办法也很简单，在优化过程中只更新在当前视图截锥内部的全部特征，这样可以保持之前重建的几何形状，又可以大大减少优化过程中的参数数量

> 我的理解就是，视锥之外的特征还是被拿过去做三线性插值去采样了，这里计算了损失，但是更新参数的时候，视锥之外的就不动了，只更新视锥内部的。不过这样会不会在某些情况下导致重建的场景存在明显的锯齿？？？
>
> 但是我看后面的实验，居然是只更新视锥内部的才没有锯齿？！！

Hierarchical Feature Grid Initialization

- **Coarse-level Feature Grid**：随机初始化；
- **Mid-level Feature Grid**：也是随机初始化的，经验表明随机初始化具有更好的收敛性；
- **Fine-level Feature Grid**：需要初始化到fine-level的decoder输出为0，毕竟这里给出来的是残差，初始化的残差肯定要是0。

  这样才能够保证从coarse-to-fine的优化过程中，能量是平稳过渡的。论文也提到，在对fine-level对应的decoder进行预训练的时候，增加了额外的正则化损失，保证如果fine-level的特征为0，那么无论mid-level的特征如何，输出的残差都应该为0。这样能够允许NICE-SLAM在运行的时候对fine-level的特征网格进行零初始化。

Justification for Design Choices

- **Why 3-level Feature Grids?**

  分层网格可以带来更好的收敛性，3-level可以保证质量、实时性能、内存消耗之间的平衡。这里论文有进行实验证明这一点；

- **Why is the Mid-level Output not a Residual to the Coarse-level Output?**

  在设计的过程中，Coarse-level的体素尺寸明显大于mid-level和fine-level，而且Coarse-level更新会影响较大的区域。为了保证小的局部收敛效率，NICE-SLAM断开了coarse与mid、fine之间的连接，只使用coarse进行预测。

Mesh Visualization

这里提到重建的场景采用分层特征网格隐式表示。NICE-SLAM使用marching cube算法创建了一个用于可视化的网格。对于每个观察点，我们使用精细级解码器和颜色解码器预测其占用值。对于预测区域中那些看不见的点(即粗网格中有部分观测的体素)，我们从粗解码器中预测占用率，并将颜色设置为青色进行可视化。

> 这里我感觉coarse的预测会不会单纯的是因为糊了？比如一张峡谷的高清图，很明显能看到峡谷，但是降低分辨率之后因为很模糊，峡谷可能就变成平地了？？？

Decoder Pretraining

NICE-SLAM使用了ConvONet提供的合成室内场景数据集对encoder-decoder预训练。这里NICE-SLAM使用的是Point Cloud Encoder而不是Voxel Encoder。所有的级别都是再ConvONet中使用room_grid64的设置进行训练的，所有特征网格的特征维度都是32。其他超参数采用了与ConvONet一样的设定。

> 所以我的疑惑很大，为什么这里会想到用ConvONet的decoder来做NICE-SLAM的decoder？？？不过ConvONet是他家的工作，说不定知道一些什么吧。。。。

相关工作和基础知识

> 下面是lqx自己整理的，想看但是还没来得及看的QAQ

- DTAM，这是2011年的一个工作，NICE-SLAM作者说最近很多基于学习的SLAM系统都是从这个上面修改的，因为DTAM足够简单。比如Deeptam: Deep tracking and mapping
- CodeSLAM、SceneCode、NodeSLAM等优化了解码到关键帧或对象深度图的表示，这三个工作经常有看到提，可以看一看。
- DI-Fusion（对比实验的常客）

- TSDF-Fusion（对比实验的常客）
- Local implicit grid representations for 3d scenes
- **Convolutional occupancy networks**是NICE-SLAM的解码器，需要好好的看一看！！！
- Neuralrecon: Real-time coherent 3d reconstruction from monocular video
- 三线性插值提取Geometry是什么情况?
- **learnable Gaussian positional encoding**是什么？（NICE-SLAM论文提到iMAP和Fourier features let networks learn high frequency functions in low dimensional domains都用）
- 对NeRF相关的改进工作，特别是有feature grid这种的，非常的疑惑，需要好好了解一下!