

CLNeRF: Continual Learning Meets NeRF

Zhipeng Cai Matthias Muller

摘要

新视角合成旨在在给定一组校准图像的情况下渲染未见视角。在实际应用中，场景的覆盖范围、外观或几何可能随时间变化，不断采集新图像。有效地整合这种持续变化是一个尚未解决的挑战。标准的 NeRF 基准仅涉及场景覆盖范围的扩展。为了研究其他实际场景变化，我们提出了一个新的数据集，称为 World Across Time (WAT)，其中包含随时间而变的外观和几何的场景。我们还提出了一种简单而有效的方法，称为 CLNeRF，将持续学习 (CL) 引入神经辐射场 (NeRFs)。CLNeRF 结合了生成式回放和即时神经图形原语 (NGP) 架构，以有效防止灾难性遗忘并在新数据到达时高效更新模型。我们还在 NGP 中添加了可训练的外观和几何嵌入，使得一个紧凑的模型能够处理复杂的场景变化。无需存储历史图像，CLNeRF 在多次扫描变化的场景上进行顺序训练的性能与一次性训练所有扫描的上限模型相当。与其他 CL 基线相比，CLNeRF 在标准基准和 WAT 上表现更好。源代码、演示和 WAT 数据集可在 <https://github.com/IntelLabs/CLNeRF> 获取。

关键词：持续学习；NeRF

1 引言

本选题选取了“CLNeRF: Continual Learning Meets NeRF”为研究对象，基于对新视角合成技术的最新发展认知，着眼于动态场景的演变，提出了结合连续学习和神经辐射场 (NeRF) 的创新方法。当前大多数视角合成方法侧重于静态场景或场景覆盖范围的扩展，忽视了动态场景的外观和几何上的不断变化。因此，本选题的依据在于解决现有方法在处理动态场景时的不足，并通过引入连续学习的思想，使模型能够在变化的环境中持续学习，避免传统方法可能遇到的灾难性遗忘问题（效果如图 1 所示）。这一研究意义重大，不仅填补了新视角合成领域中的空白，更有望提高合成图像的真实感，推动技术的应用创新，为相关领域的发展做出积极贡献。

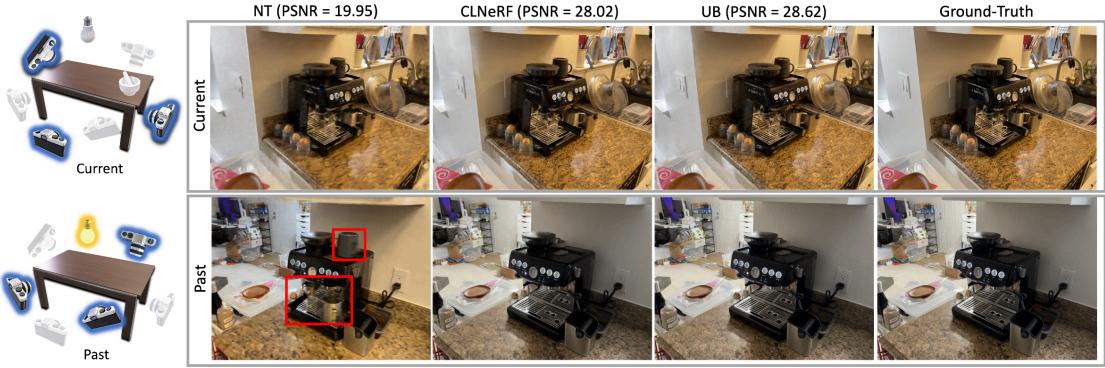


图 1. **示意图.** 这项工作研究了 NeRF 的持续学习。文章提出了一个有效的系统 CLNeRF，可以从数据集中顺序学习，而无需存储历史图像。顶部和底部行分别显示了相同场景的当前和过去数据集的新视角渲染效果。CLNeRF 准确地渲染了当前和过去的扫描，与一次性训练所有扫描的上限模型（UB）性能相当。在对扫描序列进行朴素训练（NT）会导致对当前扫描过拟合，从而在过去扫描中产生错误的外观和几何。

2 相关工作

2.1 基于隐式场表达的三维重建

NeRF(基于隐式场表达的三维重建)被认为是多视角合成中最受欢迎的技术之一。Vanilla NeRF[23] 使用神经网络(多层次感知器 MLP) 隐式地表示场景。这些 MLP 将 3D 位置和视角映射到它们对应的颜色和不透明度。通过将相机射线投影到 3D 空间并执行体积渲染来合成场景的图像。尽管 Vanilla NeRF 在插值新视角方面效果显著，但存在一些限制，例如训练和推断速度较慢。

这一问题通过使用显式场景表示 [31, 24] 或空间分布的小型 MLP[26] 得到解决。CLNeRF 将这些先进的架构应用于确保在持续学习过程中进行高效的模型更新。Vanilla NeRF 仅考虑静态场景；为了处理多样的光照或天气条件，引入了可训练的外观嵌入 [20, 32]。野外照片中的瞬时对象通过引入瞬时 MLP[20] 或使用分割掩模 [32] 来处理。CLNeRF 采用这些技术，使单一模型能够处理复杂的场景变化。与本工作同时进行，Chung 等人 [6] 也研究了在持续学习的背景下 NeRF。然而，他们仅考虑静态场景和 Vanilla NeRF 架构。本文考虑了外观和几何变化的场景，并引入了一个新的数据集来研究这种情况。持续学习和更先进的架构的合理组合也使 CLNeRF 更简单(无需额外的超参数)且更有效地减轻了遗忘问题。

2.2 持续学习

持续学习旨在从具有分布转变的数据序列中学习，而无需存储历史数据 [7]。对于非独立同分布的数据序列进行朴素训练会遭受灾难性遗忘 [12]，在历史数据上表现不佳。一种常见的方法是通过对训练目标进行正则化来防止遗忘 [12, 15]。然而，由于正则化不依赖于历史数据，在实践中效果较差。参数隔离方法通过冻结(部分)先前任务的神经元并使用新的神经元来学习后续任务，从而防止遗忘 [19, 29]。尽管这些方法可以保证记忆 [19]，但在大量任务的情况下，这些方法的容量有限或需要显著扩展网络。基于回放的方法利用历史数据来防止遗忘。这些历史数据要么存储在小型回放缓冲区中 [5, 18]，要么通过生成模型合成 [30]。生成

式回放 [30]，即合成历史数据，在图像分类方面效果较差，因为生成模型引入了额外的参数，并在高分辨率图像上表现不佳。相反，本工作表明先进的 NeRF 模型和生成式回放相辅相成，因为高质量的回放数据可以在不引入新模型参数的情况下渲染。

3 本文方法

3.1 本文方法概述

CLNeRF 是一种针对 Neural Radiance Fields 的持续学习方法，专注于处理场景在外观和几何上持续演变的实际应用场景。为此文章提出了一个新的评估基准，即 World Across Time (WAT)，旨在研究场景图像随着时间推移作为多个数据集序列的情况。为了有效地应对这种挑战，CLNeRF 在无需存储历史图像的情况下，可以顺序学习并适应连续变化的场景。

CLNeRF 采用先进的 NeRF 架构和生成式回放相结合的方式，以防止遗忘问题并提高模型的性能。相较于传统的朴素训练，CLNeRF 引入了可训练的外观嵌入和几何嵌入，使得模型能够更好地适应场景的多样性变化。此外，CLNeRF 采用了空间分布的小型 MLPs 和显式场景表示等先进架构，以确保在持续学习的过程中模型更新的高效性。

实验中展示了其在新视角合成中的卓越表现，尤其是在 WAT 基准上。通过顺序学习多个数据集，CLNeRF 不仅能够准确渲染当前扫描，还能在不引入新模型参数的情况下，以与一次性训练所有扫描的上限模型相当的性能水平渲染过去的扫描。相比之下，传统的朴素训练方法则容易过拟合当前数据集，导致对过去扫描的外观和几何产生错误的渲染。因此 CLNeRF 的方法为持续学习与新视角合成的有机结合提供了有效而创新的解决方案。方法流程图如图 2 所示。

3.2 方法提要

在介绍 CLNeRF 之前，首先回顾一下 NeRF 的基础知识，并明确持续学习的问题。

NeRF 训练了一个由参数 θ 参数化的模型，该模型将 3D 位置 $x \in R^3$ 和视角方向 $d \in S^3$ (从相机中心到 x 的单位向量) 映射到相应的颜色 $c(x, d|\theta) \in ([0, 1])^3$ 和不透明度 $\sigma(x|\theta) \in [0, 1]$ 。给定目标图像视角，我们独立地渲染每个像素的颜色。对于每个像素，我们从相机中心 $o \in R^3$ 沿着光线投射到像素中心，并在光线上采样一组 3D 点 $X = x_i | x_i = o + \tau_i d$ ，其中 τ_i 是从相机中心到采样点的欧几里德距离。然后，我们根据体积渲染 [21] 方程渲染光线的颜色 $\hat{C}(X)$ ：

$$\hat{C}(X) = \sum_i \omega_i c(x_i, d|\theta),$$

直观地说，公式 (1) 计算了所有采样点上颜色的加权和。权重 w_i 是基于不透明度和到相机中心的距离计算的。

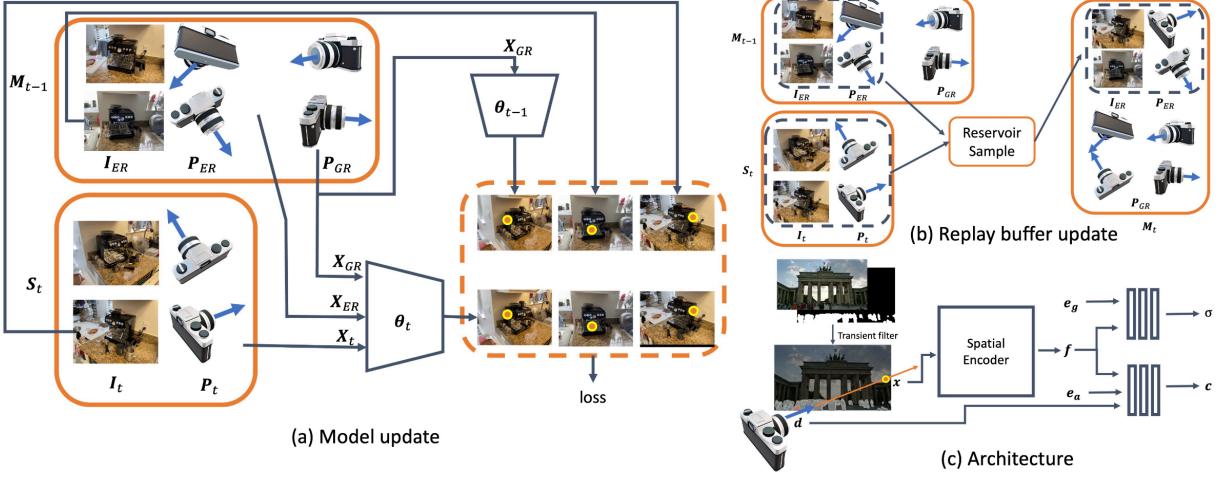


图 2. 系统概述 (a) 在持续 NeRF 的每个时间步 t , 生成一个新的数据集 S_t 。为了更新模型 Θ_t , 我们在每次训练迭代中从存储在经验回放 (P_{ER})、生成回放 (P_{GR}) 和新数据 (P_t) 中的相机参数中随机生成一组相机光线。对于来自新数据 (X_t) 或经验回放 (X_{ER}) 的光线, 使用相应的图像颜色进行监督。对于来自生成回放的光线, 即 X_{GR} , 文章使用最新部署的模型 $t-1$ 生成伪标签进行监督。在训练完 t 后, 我们用 t 替换先前部署的模型 $t-1$, 并更新回放缓冲区 M_t 。(b): 为了在可选的经验回放下更新回放缓冲区 M_t , 文章对 M_{t-1} 和 S_t 中的图像-相机参数对执行蓄水池采样, 并将未被蓄水池采样选择的所有图像的相机参数添加到 P_{GR} 中。(c) 文章使用分割掩模来过滤瞬时对象, 并对基础架构应用外观嵌入 e_a 和几何嵌入 e_g 来处理不同时间步的场景变化。

持续学习在本文中将 NeRF 的问题称为持续 NeRF。在持续 NeRF 的每个时间步 t 中:

1. 生成一组带有它们的相机参数(内参和外参) S_t 的训练图像。
2. 通过以下方式更新当前模型 θ_t 和回放缓冲区 M_t (用于存储历史数据):

$$M_t, \theta_t \leftarrow update(S_t, \theta_{t-1}, M_{t-1}),$$

3. 部署 θ_t 以渲染新的视角直到 $t+1$ 。

这个过程模拟了实际情景, 其中模型 θ_t 持续部署。偶尔会出现一组新图像, 可能包含场景的新视角以及外观或几何上的变化。目标是更新 θ_t ; 理想情况下, 存储(以保持 M_t 中的历史数据)和内存(以部署 t)的需求应该很小。

如图 2 所示, CLNeRF 解决了持续 NeRF 的三个主要问题: (1) 使用最小存储有效地更新 θ_t , (2) 在可选的经验回放期间更新 M_t , 以及 (3) 使用单一紧凑模型处理各种场景变化。我们在下面更详细地介绍了每个组件。

3.3 方法模型更新

CLNeRF 应用基于回放的方法 [5, 30] 来防止灾难性遗忘。为了支持极端的存储限制应用, CLNeRF 将生成回放 [30] 与先进的 NeRF 架构结合起来, 即使不能存储任何历史图像, 也能发挥作用。

图 2(a) 描述了 CLNeRF 在每个时间步 t 的模型更新过程。所有历史图像的相机参数都存储在回放缓冲区 M_{t-1} 中, 用于生成回放。当存储足够进行经验回放 [5] 时, 可以选择保留

少量图像 I_{ER} 。在 θ_t 的每次训练迭代中，CLNeRF 从 $P_t \cup P_{GR} \cup P_{ER}$ 中均匀生成一批相机光线 $X = X_{ER} \cup X_{GR} \cup X_t$ ，其中 P_t 、 P_{GR} 和 P_{ER} 分别是新数据 S_t 、生成回放数据和经验回放数据的相机参数。

其中，LNeRF 是标准 NeRF 训练的损失， $C(\cdot)$ 是来自新数据或回放的监督信号， $\hat{C}(\cdot | \theta_t)$ 是由 θ_t 渲染的颜色。对于从 P_{GR} 采样的光线 $X \in X_{GR}$ ，我们执行生成回放，即将监督信号 $C(X)$ 设置为由 θ_{t-1} 生成的图像颜色 $\hat{C}(X | \theta_{t-1})$ 。对于其他光线， $C(X)$ 是地面实况图像颜色。在模型更新后，我们用 θ_t 替换先前部署的模型 θ_{t-1} ，并更新回放缓冲区 M_t 。只有 θ_t 和 M_t 被保留，直到下一组数据 S_{t+1} 到达。

尽管所有相机参数都存储在 M_{t-1} 中，但它们仅占用很小的存储空间，最多为 $N_{t-1}(d_{pose} + d_{int})$ ，其中 N_{t-1} 是历史图像的数量， d_{pose} 和 d_{int} 分别是相机姿态和内部参数的维度；对于常见的相机模型， $d_{pose} = 6$ ， $d_{int} \leq 5$ [10]。如果多个图像由同一相机拍摄，则 d_{int} 是共享的。具体示例中，存储使用不同相机捕获的 1000 个样本的参数大约需要 45KB 的存储空间，远小于存储单个高分辨率图像的存储需求。这确保了 CLNeRF 的有效性，即使在存储极限的应用中也是如此。

我们还强调随机抽样的重要性。CLNeRF 在迄今为止所有视图之间分配均匀的抽样权重。一些基于图像分类的持续学习方法 [5] 和 NeRF 的并行工作 [6] 提出了偏倚的抽样策略，其中固定且较大的百分比（12 至 23）的射线是从新数据 (P_t) 中抽样的。这种策略不仅引入了新的超参数（例如，旧数据的损失权重或新数据射线比例 [6]），而且表现比均匀随机抽样差。

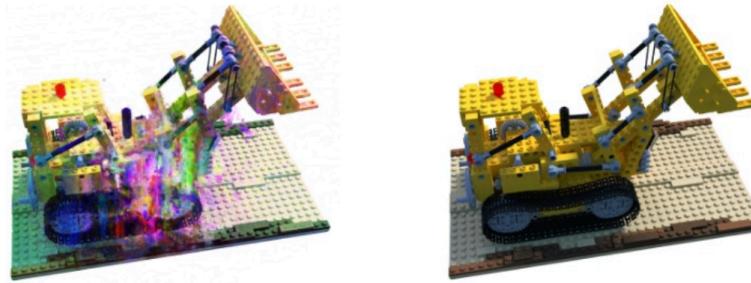
3.4 回放模型更新

在极端情况下，即无法存储任何图像进行经验回放的情况下，仅在 M_t 中存储历史数据的相机参数，以使 CLNeRF 对各种存储大小的实际系统具有灵活性和有效性。当存储足够维护一部分历史图像进行经验回放时使用一个蓄水池缓冲区 [5]。具体而言，只要未达到存储限制就将当前数据添加到 M_t 中。否则，如图 2(b) 所示，鉴于 M_{t-1} 可以存储 K 个图像，首先为 S_t 中的每个图像 I_i 生成一个随机整数 $j_i \in 1, 2, \dots, N_i$ ，其中 N_i 表示 I_i 在连续学习数据序列中的顺序。如果 $j_i > K$ ，则不将 I_i 添加到 M_t 中。否则用 I_i 替换 M_{t-1} 中第 j_i 个图像。请注意， M_t 存储所有相机参数，无论相应的图像是否存储。

文章还尝试使用优先回放缓冲区 [27]，其中 M_t 保留渲染质量最低的图像。具体而言，在更新 θ_t 后，我们迭代 M_{t-1} 和 S_t 中的所有图像，并从 θ_t 中选择具有最低渲染 PSNR [11] 的 K 个图像。尽管在强化学习中广泛使用，但在连续 NeRF 中，优先回放并不比蓄水池抽样表现更好。蓄水池缓冲区的实现更简单且更有效。

3.5 室外场景

CLNeRF 默认使用 Instant Neural Graphics Primitives (NGP) [24] 架构。这不仅在连续 NeRF 期间实现了高效的模型更新，还确保了生成回放的低开销和有效性。NGP 用作 CLNeRF 的骨干结果表明，与基本的 NeRF [23] 相比，NGP 在性能和效率上都有更好的表现。一个紧凑的连续 NeRF 系统应该使用单一模型来整合场景变化，以便模型大小不会随时间显著增加。文章通过将可训练的外观和几何嵌入添加到基本架构来实现这一点。给定一个空间位置 x 和一个视角方向 d ，文章首先将 x 编码为特征向量 f （对于 NGP，使用基于网格的哈希编码器，对于基本的 NeRF，使用 MLP）。然后通过 $c = D_c(f, d, e_a)$ 和 $\sigma = D_\sigma(f, e_g)$ 生成颜色和不透



(a) Test view rendered by NGP with transient MLP.



(b) Test view rendered by NGP with masked transient objects.



(c) Training view rendered by NGP with transient MLP.



(d) GT training view.

图 3. NGP 与瞬态 MLP 的结果

明度，其中 D_c 和 D_σ 是颜色和不透明度解码器（对于 NGP 和基本的 NeRF 均为 MLP）； e_a 是可训练的外观嵌入， e_g 是几何嵌入。鉴于相同场景的一系列数据集，不同数据集之间存在外观和几何变化，文章为每个扫描添加一个外观嵌入和一个几何嵌入，即每个连续 NeRF 时间步 t 。文章将外观和几何嵌入的维度分别设置为 48 和 16，这确保了在连续 NeRF 期间模型大小的最小增加。

CLNeRF 使用分割掩码（来自 [3]）来过滤瞬态对象。正如图 3 所示，文章还尝试使用瞬态 MLP 和鲁棒训练目标 [20]，但发现 NGP 与这种策略不兼容——非瞬态网络过度拟合于瞬态对象，并且无法自动过滤它们。文章只移除单个扫描内的瞬态/动态对象，例如移动的行人。不同数据集之间的场景变化，例如新建筑的增加，则通过几何嵌入来处理。

4 复现细节

4.1 与已有开源代码对比

本次复现的论文代码已经在 github 上放出 <https://github.com/IntelLabs/CLNeRF>，作者提供了端到端的训练过程能够利用符合输入格式的图片数据集来完成新视角的合成，在论文中作者制作了名为 WAT 的数据集，能够满足论文提出的时间序列数据集的要求，并展示出了 CLNeRF 模型相较于其他模型的效果。同时在代码中作者还提供了其他的 NeRF 模型用于进行对比以及读取不同格式的数据集 (SynthNeRF, nerfpp 等)。在网络架构部分采用了表现较好且快速的 Instant-NGP，能够更好地匹配持续学习算法的需求，更快的训练速度保证了持续学习的效率。

本次试验中我对于实验环境的搭建方面做了更改，优化了环境搭建部分的难度，更改了

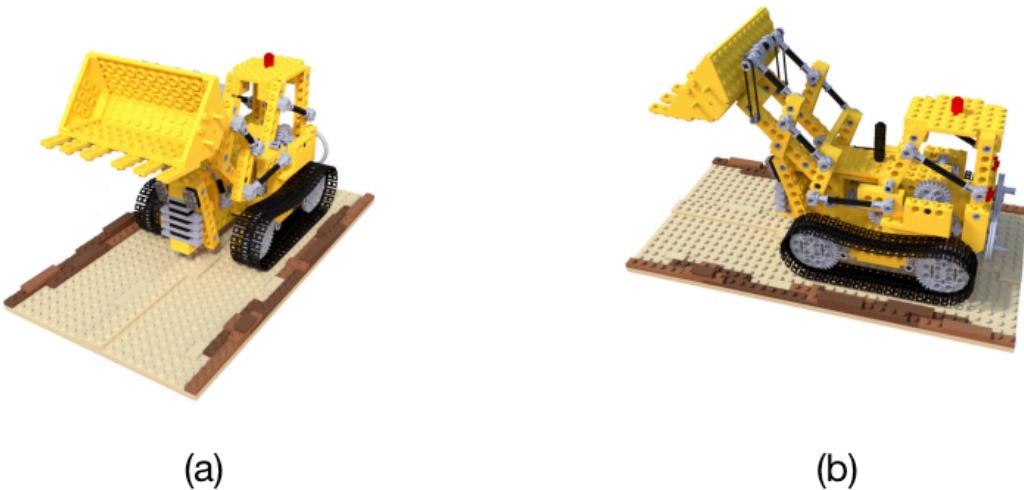


图 4. Lego (a) 与 (b) 为从不同的位姿进行渲染的结果

互相冲突的库版本同时修改了其在代码中的具体使用。同时在 github 已放出的链接数据集的下载链接也存在一些问题，将其更改为了能够正确下载的链接。除此之外我对原本的代码整体架构做了整理，整合在读取数据方面存在过多冗余文件的问题，而代码的结构内容例如 scripts 中嵌套文件夹过多以及存在 sh 文件存放位置混乱，在进行实验时对此类内容做了重新的整理和修改。

因为复现实验代码的过程中需要进行验证自己收集的数据集，因此在实验过程中增加了能够将拍摄的视频按照特定的帧率截取为图片，再利用已开源的软件 Colmap 利用 SfM 算法来得到对应的位姿用于 CLNeRF 模型。代码的训练部分存在着在多 GPU 并行训练的条件下读取数据并更新 replay buffer 时出现文件存取顺序问题，通过修改读取数据代码部分从未避免了冲突问题。



图 5. WAT-car (a) 和 (b) 为对 car 数据集进行渲染的结果，(a) 是在关闭后备箱的情况下而 (b) 则是在开启后备箱的情况下渲染的结果；两组 scan 在同一个 nerf 训练的情况下都表现出了良好的效果



(a)

(b)

图 6. **WAT-car** (a) 和 (b) 为对 car 数据集在车处于开启车门和关闭车门情境下进行渲染的结果，两组渲染的结果都在 c 和 σ 的结果上出现了问题，模型在更新的过程中存在问题。

4.2 实验环境搭建

实验代码需要的环境为 $\text{python}=3.8$, $\text{pytorch}=1.11.0$, $\text{torchvision}=0.12.0$, $\text{pytorch-lightning}=1.9.5$;

另外实验中需要配置多个 python 库来在训练的过程中进行加速处理，其中 torch-sactter 的版本需要同时匹配 torch 与 cuda 版本来进行安装；安装 apex 前需要在环境内有 cmake 和 gcc，同时需要手动的下载 apex 22.04 以下的版本避免在安装过程中出错；pytorch-lightning 的最新版本中更改了 DDPStrategy 的使用方法，需要在代码段中更改调用的方式。

此外实验还需要运行 preparedatasets 下载特定的验证数据集，其中 WAT 数据集的下载链接已经失效，更改为最新的下载链接。

4.3 使用说明

在 CLNeRF 上运行实验，支持 WAT、SynthNeRF 和 NeRF++ 数据集，通过运行不同的 run.sh 文件来完成对各个数据集以及在 MEIL-NeRF、经验重播 (ER)、弹性权重共享 (EWC) 和朴素训练/在序列数据上微调 (NT) 上进行实验。此外可以使用 CLNeRF 模型渲染视频的命令，用户需根据具体情况修改相关参数和路径：

```
rep=10
scale=8.0(scripts/NT/WAT/breville.sh)
ckpt_path=/export/work/zcai/WorkSpace/CLNeRF/CLNeRF/ckpts/NGPGv2_CL/
colmap_ngpa_CLNerf/
bash scripts/CLNeRF/WAT/render_video.sh $task_number $task_curr $scene $ckpt_path
$rep $scale $render_fname
```

5 实验结果分析

首先在 SynthNeRF 数据集上进行实验，如图 4 所示，在没有存在时间序列的数据集条件之下，CLNeRF 模型在视角渲染方面表现良好，基于 Instant-NGP 的网络架构进一步加快了训练和渲染的过程，因为此类的数据集不具备论文中所提出的渲染条件，作者在论文中介绍了 WAT (WorldAcrossTime) 数据集，即该数据集包含用于 NeRF 持续学习的多个 Colmap



rep_size = 5

(a)

rep_size = 10

(b)

图 7. **WAT-car** (a) 和 (b) 为对 car 数据集进行渲染的结果, (a) 是在关闭后备箱的情况下而 (b) 则是在开启后备箱的情况下渲染的结果; 两组 scan 在同一个 nerf 训练的情况下都表现出了良好的效果

重建场景。对于每个场景提供了在不同时间捕获的多个 scan, 其中相同的场景具有不同的外观和几何条件。在实验中对作者提供的 WAT 数据集进行了验证和使用, 选取了数据集中名为 car 的 scan 进行了基于 CLNeRF 模型的训练和渲染, 结果如图 5 所示。

对于同一组 WAT 中 car 数据集的情况, 在进行训练和渲染另一组变化时出现了如图 6 的错误, 在大量的场景变化的数据集训练时会出现渲染错误的情况, CLNeRF 模型对于 C 和 σ 的渲染仍保存的为之前的模型信息, 在此之中模型未能正确的更新。除此之外, 还进行了消融实验, 分别设置了 reply buffer 的 size 为 5 以及 10, 如图 7。size 的大小设置决定了在 reply

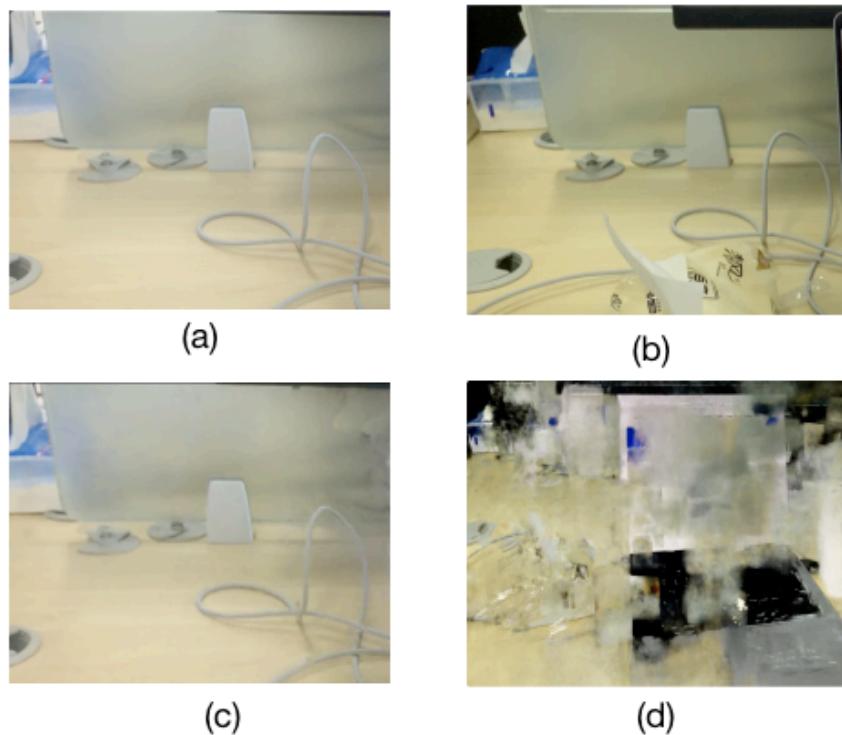


图 8. **自采集数据集** (a) 采集的无遮挡 scan; (b) 采集的有遮挡的数据集; (c) 以 (a) 为条件进行渲染的结果; (d) 以 (b) 为数据集进行渲染的结果。

buffer 模型更新时所参考到的历史图片数量，在选择减少参考 size 之后可以明显地看到图像不仅在 σ 另外还是 C 上都存在着较大的错误。

另外还使用了自主收集的数据集进行验证，如图 8 所示，分别比较在有无物品的区别，按照文章所给的 WAT 数据集格式进行了训练和渲染，但在有纸巾遮挡的 scan 中出现较大的错误。之后考虑到是因为输入图片数量过少以及 replay buffer 中数目过大导致了在部分位置的颜色等信息发生错误。

6 总结与展望

本次复现工作进行了对神经辐射场 (NeRFs) 的持续学习。文章提出了一个新的数据集——WAT (World Across Time)，其中包含随着时间推移而发生外观和几何变化的自然场景。此外还提出了 CLNeRF 模型，这是一个有效的持续学习系统，其性能接近一次性训练整个数据集的上界模型。CLNeRF 利用生成性回放，在没有存储任何历史图像的情况下表现良好。尽管目前的实验仅涵盖了包含数百张图像的场景，但它们对于在实际世界中部署实用的 NeRFs 有实际的意义。对于 CLNeRF 存在许多有趣的未来研究方向。例如，解决 NGP 的 NaN 损失问题，以使在持续学习期间模型继承更为有效。将 CLNeRF 扩展到类似于 Block-NeRF [32] 规模的方向也是未来工作的有趣方向。

参考文献

- [1] Nerfw PyTorch Lightning Implementation. https://github.com/kwea123/nerf_pl/tree/nerfw.
- [2] Zhizhong Li and Derek Hoiem. *Learning without forgetting*. IEEE Transactions on Pattern Analysis and Machine Intelligence, 40(12):2935–2947, 2017.
- [3] Zhiqiu Lin, Jia Shi, Deepak Pathak, and Deva Ramanan. *The clear benchmark: Continual learning on real-world imagery*. In Thirty-fifth Conference on Neural Information Processing Systems Datasets and Benchmarks Track (Round 2), 2021.
- [4] Lingjie Liu, Jiatao Gu, Kyaw Zaw Lin, Tat-Seng Chua, and Christian Theobalt. *Neural sparse voxel fields*. Advances in Neural Information Processing Systems, 33:15651–15663, 2020.
- [5] David Lopez-Paz and Marc’ Aurelio Ranzato. *Gradient episodic memory for continual learning*. Advances in neural information processing systems, 30, 2017.
- [6] Arun Mallya and Svetlana Lazebnik. *Packnet: Adding multiple tasks to a single network by iterative pruning*. In Proceedings of the IEEE conference on Computer Vision and Pattern Recognition, pages 7765–7773, 2018.
- [7] Ricardo Martin-Brualla, Noha Radwan, Mehdi SM Sajjadi, Jonathan T Barron, Alexey Dosovitskiy, and Daniel Duckworth. *Nerf in the wild: Neural radiance fields for unconstrained photo collections*. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, pages 7210–7219, 2021.

- [8] Nelson Max. *Optical models for direct volume rendering*. IEEE Transactions on Visualization and Computer Graphics, 1(2):99–108, 1995.
- [9] Michael McCloskey and Neal J Cohen. *Catastrophic interference in connectionist networks: The sequential learning problem*. In Psychology of learning and motivation, volume 24, pages 109–165. Elsevier, 1989.
- [10] Ben Mildenhall, Pratul P Srinivasan, Matthew Tancik, Jonathan T Barron, Ravi Ramamoorthi, and Ren Ng. *Nerf: Representing scenes as neural radiance fields for view synthesis*. Communications of the ACM, 65(1):99–106, 2021.
- [11] Thomas Müller, Alex Evans, Christoph Schied, and Alexander Keller. *Instant neural graphics primitives with a multiresolution hash encoding*. ACM Transactions on Graphics (ToG), 41(4):1–15, 2022.
- [12] Ameya Prabhu, Philip HS Torr, and Puneet K Dokania. *Gdumb: A simple approach that questions our progress in continual learning*. In Computer Vision–ECCV 2020: 16th European Conference, Glasgow, UK, August 23–28, 2020, Proceedings, Part II 16, pages 524–540. Springer, 2020.
- [13] Christian Reiser, Songyou Peng, Yiyi Liao, and Andreas Geiger. *Kilonerf: Speeding up neural radiance fields with thousands of tiny mlps*. In Proceedings of the IEEE/CVF International Conference on Computer Vision, pages 14335–14345, 2021.
- [14] Tom Schaul, John Quan, Ioannis Antonoglou, and David Silver. *Prioritized experience replay*. arXiv preprint arXiv:1511.05952, 2015.
- [15] Johannes Lutz Schönberger and Jan-Michael Frahm. *Structure-from-motion revisited*. In Conference on Computer Vision and Pattern Recognition (CVPR), 2016.
- [16] Joan Serra, Didac Suris, Marius Miron, and Alexandros Karatzoglou. *Overcoming catastrophic forgetting with hard attention to the task*. In International conference on machine learning, pages 4548–4557. PMLR, 2018.
- [17] Hanul Shin, Jung Kwon Lee, Jaehong Kim, and Jiwon Kim. *Continual learning with deep generative replay*. Advances in neural information processing systems, 30, 2017.
- [18] Cheng Sun, Min Sun, and Hwann-Tzong Chen. *Direct voxel grid optimization: Super-fast convergence for radiance fields reconstruction*. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, pages 5459–5469, 2022.
- [19] Matthew Tancik, Vincent Casser, Xinchen Yan, Sabeek Pradhan, Ben Mildenhall, Pratul P Srinivasan, Jonathan T Barron, and Henrik Kretzschmar. *Block-nerf: Scalable large scene neural view synthesis*. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, pages 8248–8258, 2022.

- [20] Kai Zhang, Gernot Riegler, Noah Snavely, and Vladlen Koltun. *Nerf++: Analyzing and improving neural radiance fields*. arXiv preprint arXiv:2010.07492, 2020.