УДК 624.21

ИСПОЛЬЗОВАНИЕ НЕЙРОННОЙ СЕТИ NEURAL RADIANCE FIELD ДЛЯ СОЗДАНИЯ ЦИФРОВОЙ МОДЕЛИ МЕСТНОСТИ

***Скурко Тимофей Олегович**, **Лоза Максим Геннадьевич**, студенты 3-го курса кафедры «Геодезия и аэрокосмические геотехнологии»*
Белорусский национальный технический университет, г. Минск
(Научный руководитель – Будо А.Ю., старший преподаватель)

Neural radiance field (NeRF) – это нейронная сеть, которая может генерировать новые виды сложных 3D-сцен на основе частичного набора 2D-изображений. Она обучена использовать потери пакетов рендеринга для воспроизведения входных видов сцены. Она работает, используя входные изображения, представляющие сцену, и выполняет интерполяцию, чтобы визуализировать полную сцену. NeRF — это высокоэффективный способ создания изображений в виде синтетических данных.

Сеть NeRF обучена сопоставлять направление просмотра и пространственное местоположение (вход 5D) с непрозрачностью и цветом (вывод 4D), используя объемный рендеринг. NeRF — это алгоритм с интенсивными вычислениями, обработка сложных сцен может занять часы или дни. Тем не менее, доступны новые алгоритмы, которые значительно улучшают производительность.

Принцип работы. NeRF использует разряженный набор входных изображений для оптимизации функции непрерывной объемной сцены. Результатом этой оптимизации является возможность создавать новые виды сложной сцены. Непрерывная сцена — это 5D-векторная функция со следующими характеристиками: Его входными данными является 3D-местоположение x = (x; y; z) и 2D-направление просмотра (θ; Φ) На выходе получается излучаемый цвет c = (r; g; b) и объемная плотность (α).

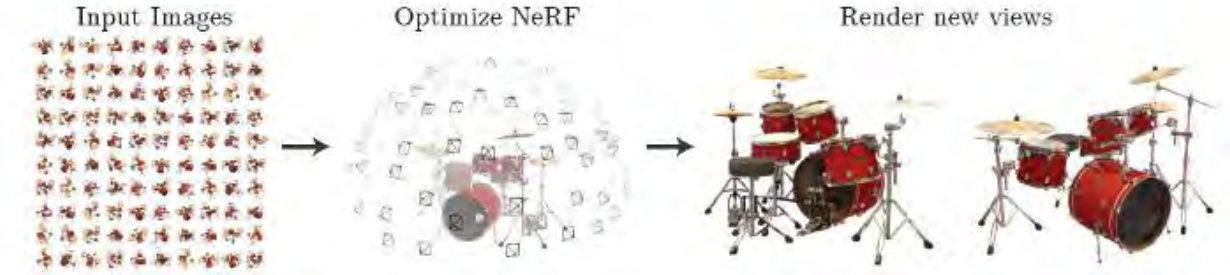

Рисунок 1 – Порядок генерации NeRF

Создание выборочного набора 3D-точек, проведя лучи камеры через сцену. Создание выходного набора плотностей и цветов, введя в нейронную сеть точки выборки с соответствующими направлениями 2D-просмотра. Накапливание плотности и цвета в 2D-изображении, используя классические методы объемного рендеринга.

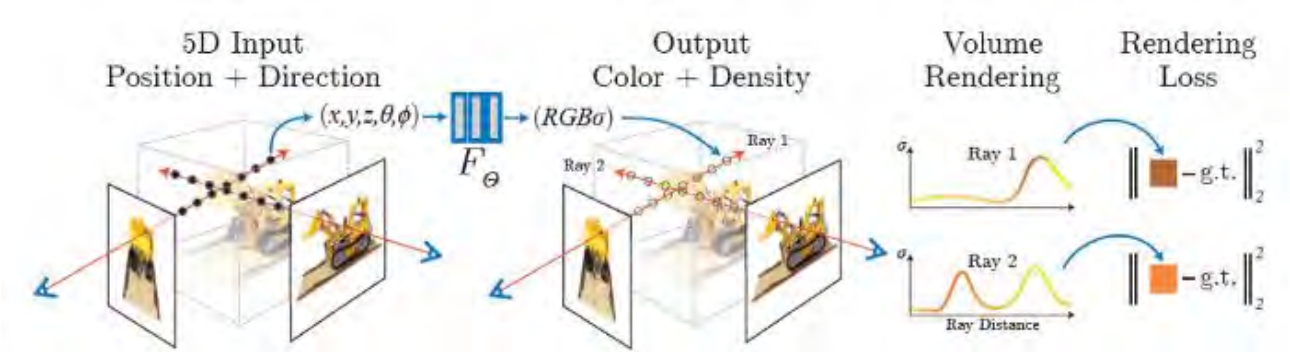

Рисунок 2 – Генерация NeRF

Описанный выше процесс использует градиентный спуск (Численный метод нахождения локального минимума или максимума функции с помощью движения вдоль градиента, один из основных численных методов современной оптимизации.), чтобы свести к минимуму ошибки между каждым наблюдаемым изображением и всеми соответствующими видами, полученными из представления.

Оригинальная модель NeRF имела несколько недостатков — она медленно обучалась и рендерилась, могла обрабатывать только статические сцены. Большим недостатком также было то, что модель NeRF, обученная на одной сцене, не могла быть использована для других сцен.

Вот несколько методов, основанных на NeRF и пытающихся решить некоторые из его проблем:

RegNeRF – расшифровывается как regularizing neural radiance fields, что представляет собой упорядочивание нейронных полей для синтеза изображений из разреженных входов. Это помогает решить проблему производительности, когда количество входных представлений невелико.

Mega-NeRF – это обучаемая структура, которая использует NeRF для создания интерактивных 3D-сред из крупномасштабных визуальных снимков, таких как здания или несколько городских кварталов, собранных в основном дронами.

Традиционный NeRF требует моделирования тысяч изображений с различными условиями освещения, каждое из которых захватывает лишь небольшую часть сцены и затрудняет быстрый рендеринг. Кроме того, большие

модели требуют производительности, которая делает невозможным обучение на одном графическом процессоре.

Mega-NeRF решает эти проблемы, анализируя статистику видимости для крупномасштабных сцен. Для этого требуется разреженная сетевая структура, в которой параметры специализированы для разных областей сцены. Он представляет простой алгоритм, который разделяет обучающие изображения на различные подмодули NeRF, которые можно обучать параллельно.

В данной работе для примера была выбрана реализация метода Mega-NeRF. Исходный код и инструкции к установке представлены на странице проекта github (GitHub - cmusatyalab/mega-nerf at pythonrepo.com).

Требования. Графический процессор NVIDIA. Компилятор с поддержкой C++ 14. Рекомендуются: Windows: Visual Studio 2019 или 2022. Linux: GCC/G++ 8 или выше. Последняя версия CUDA.

Рекомендуются. Windows: CUDA 11.5 или выше. Linux: CUDA 10.2 или выше. CMake v3.21 или выше. (необязательно) Python 3.7 или выше для интерактивных привязок. (опционально) OptiX 7.6 или выше для более быстрого обучения сетке SDF. (необязательно) Vulkan SDK для поддержки DLSS.

Также рекомендуется установить CUDA в переменную среды PATH.

Установка. Копирование репозитория в нужную папку. С помощью CONDA запуск компиляции кода:

```
conda env create -f environment.yml
conda activate mega-nerf
```

Использование. Рекомендуется использовать PixSFM если выравнивание снимков по GPS невозможного .

sudo apt-get install libhdf5-dev

git clone https://github.com/cvg/pixel-perfect-sfm --recursive

cd pixel-perfect-sfm

pip install -r requirements.txt

Если выравнивание возможно, то производится настройка пользовательского набора данных регулярной структуры каталогов.

Извлечение данных

python scripts/create_octree.py --config configs/mega-nerf/${DATASET_NAME}.yaml --dataset_path $DATASET_PATH --container_path $MERGED_OUTPUT --output $OCTREE_PATH