3D Gaussian Splatting
for Real-Time Radiance Field Rendering

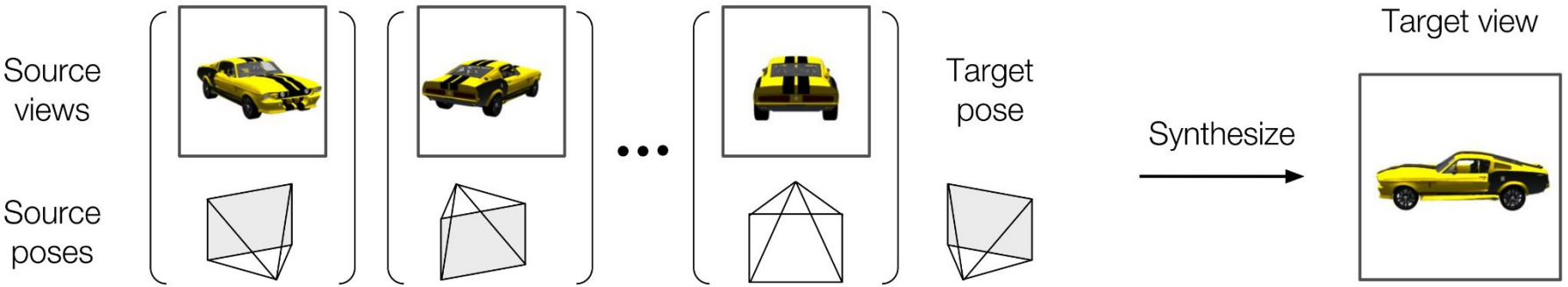
16 (23) октября, 2023

Александр Демин, Артём Бекян

План

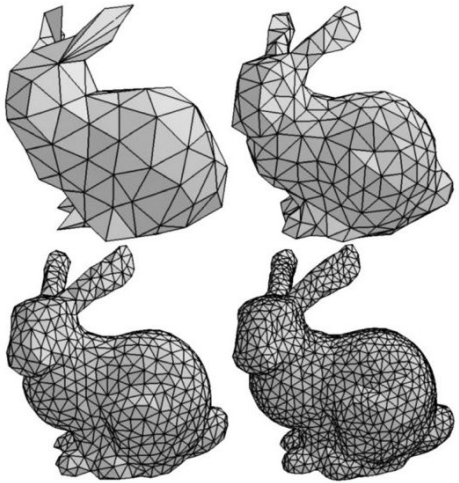
- В двух словах о самой статье (докладчик)
- Несколько экспериментов (хакер)

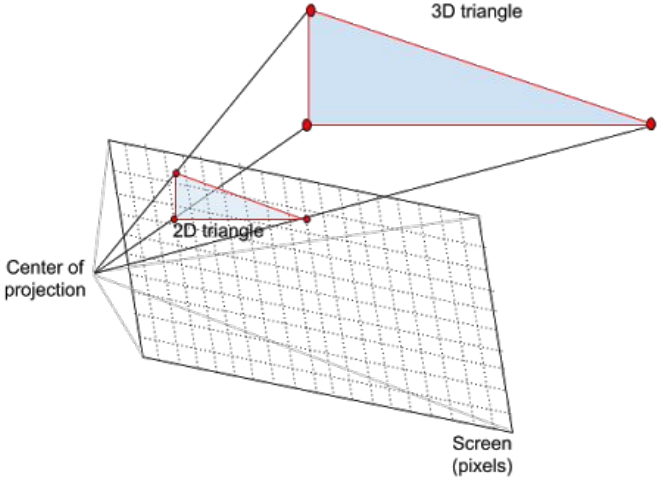
Bird’s eye view

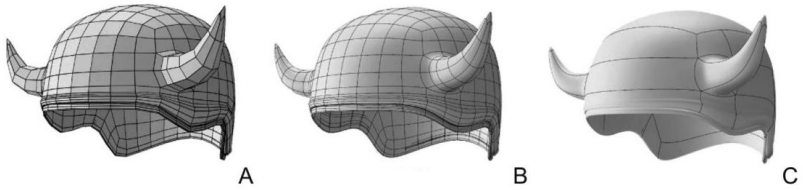


- Имеем съемку объекта с нескольких фиксированных ракурсов
- Хотим научиться строить изображение объекта с любого ракурса

Bird’s eye view







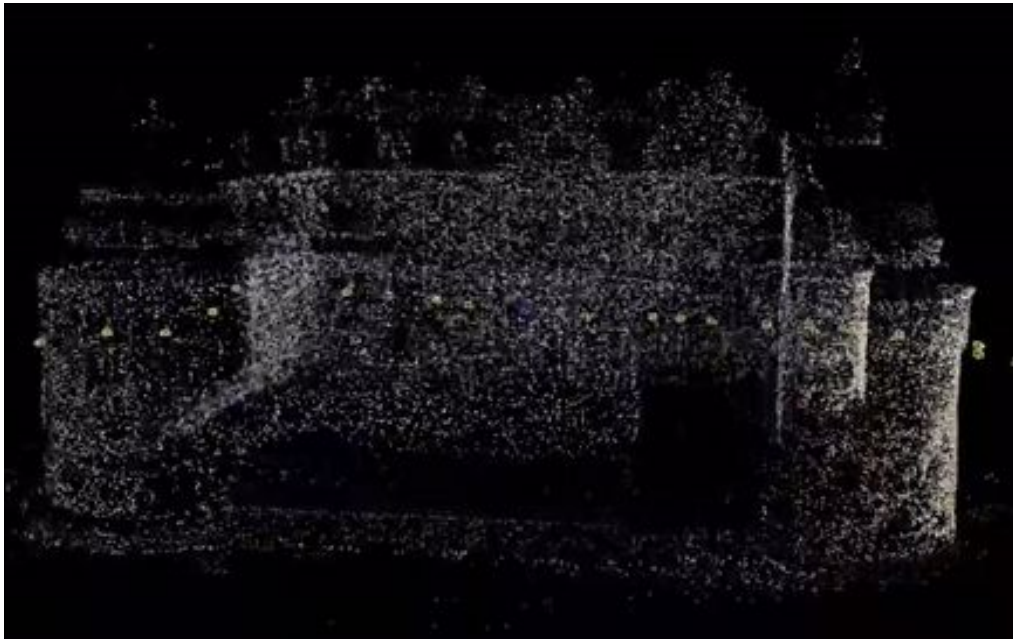
Основные компоненты:

- 3D сцена
- растерайзер (проектор из 3D в 2D)

Bird’s eye view

Идея из статьи: зададим 3D сцену набором маленьких эллипсов – Гауссиан

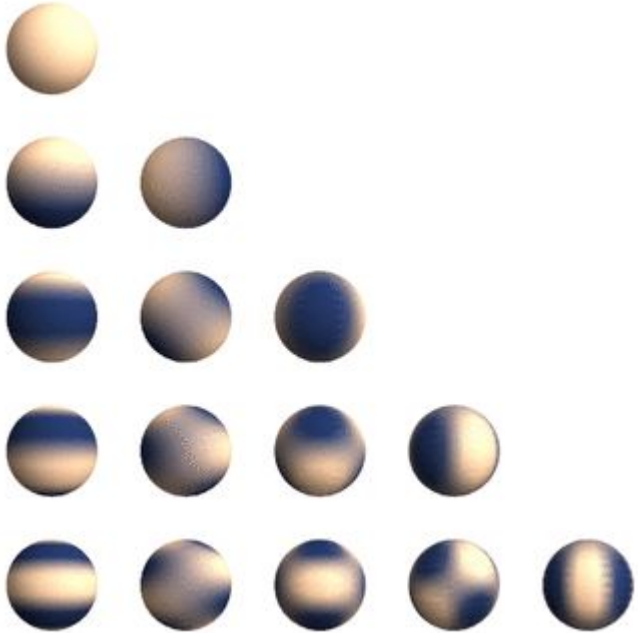
Иначе говоря, “**a bunch of blobs in space**”



Про Гауссианы

Зададим каждую Гауссиану несколькими параметрами:

- **M** - позиция в пространстве (вектор в R^3)
- **S** - растяжение и поворот (матрица в R^3x3)
- **A** - прозрачность (число в R)
- **C** - цвет, или, условно, функция из сферы в пространство цветов

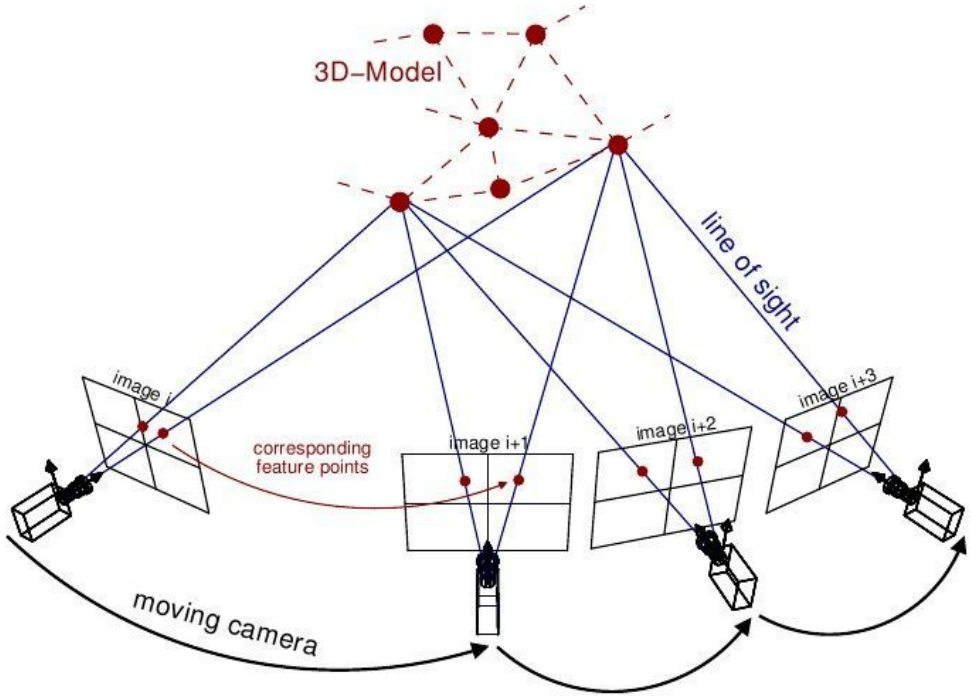


Хотим создать много Гауссиан, и у каждой выучить **M**, **S**, **A**, **C**

Инициализация Гауссиан

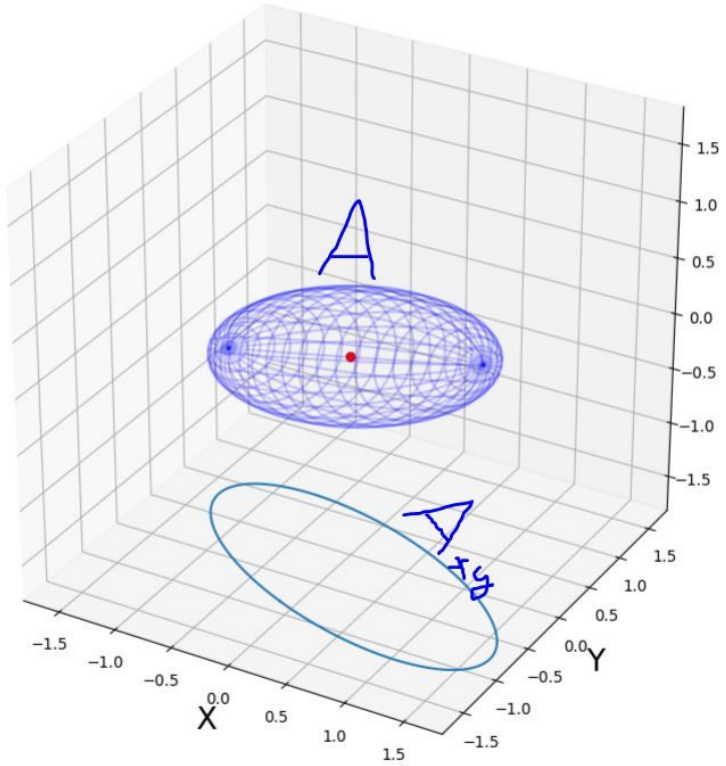
- Можно инициализировать **M**, **S**, **A**, **C** случайно
- Structure from motion:
  Реконструирует позиции Гауссиан **M** из позиций камер

  Пригодится реконструировать ground-truth позиции и углы обзора, с которых делали съемку (COLMAP)



Рендеринг Гауссиан

- Угол обзора камеры в пространстве задается аффинной плоскостью
- Для отрисовки Гауссиан на экране, будем проецировать 3D Гауссианы на 2D плоскость *(используя растерайзер)*



Алгоритм

алгоритм

Algorithm 1 Алгоритм

Input: Датасет с позициями камер и изображениями

Output: Набор Гауссиан, заданных $\mathbf{M}, \mathbf{S}, \mathbf{C}, \mathbf{A}$;

(i) Инициализировать $\mathbf{M}, \mathbf{S}, \mathbf{C}, \mathbf{A}$

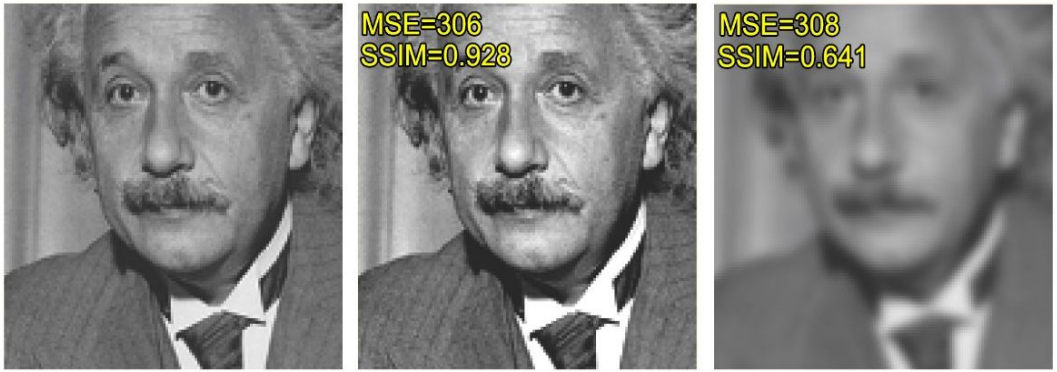
(ii) Пока нет сходимости:

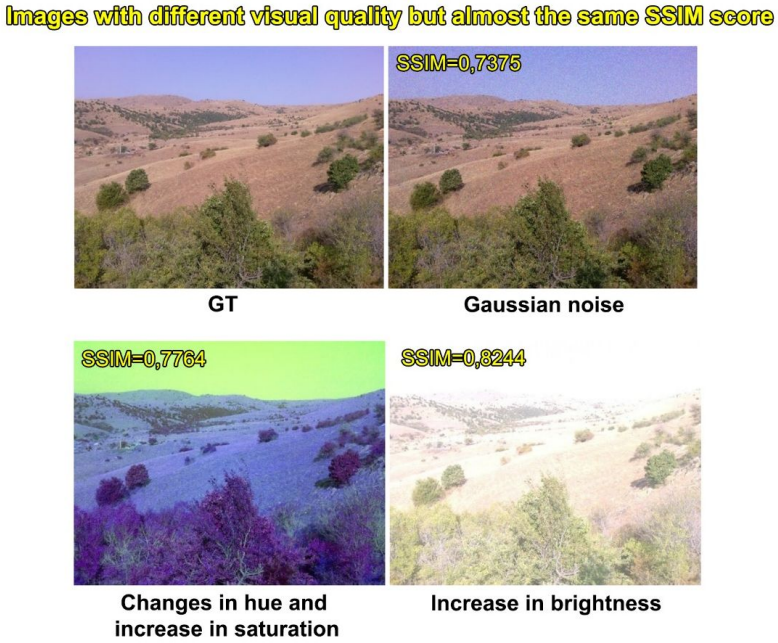
(a) **Сэмплировать** позицию камеры V и ground-truth изображение $\hat{I}$ из датасета

(b) **Спроецировать** Гауссианы на V и получить изображение I (rasterizer)

(c) **Найти лосс** $\mathcal{L}(I, \hat{I})$ и сделать шаг $\mathbf{M}, \mathbf{S}, \mathbf{C}, \mathbf{A}$ по $\nabla\mathcal{L}$

Лосс

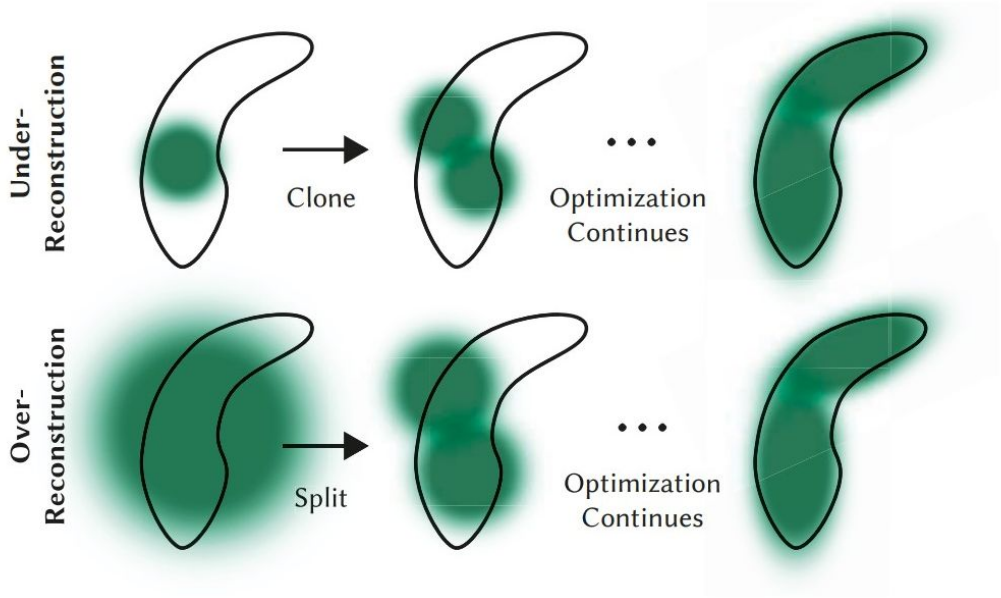




$$\mathcal{L} = (1 - \lambda)\mathcal{L}_1 + \lambda\mathcal{L}_{\text{D-SSIM}}$$

SSIM - структурное сходство

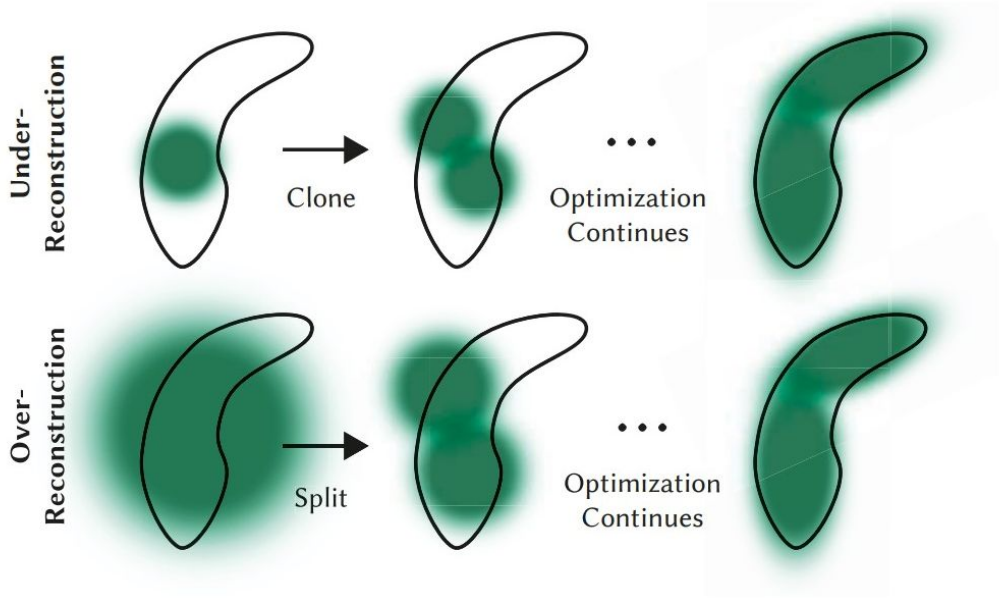
Adaptive Density Control



Если у Гауссианы большой градиент по **M**, то поправим ее:

- Under-Reconstruction: клонируем Гауссиану с небольшим **S**
- Over-Reconstruction: делим пополам Гауссиану с большим **S**

Adaptive Density Control

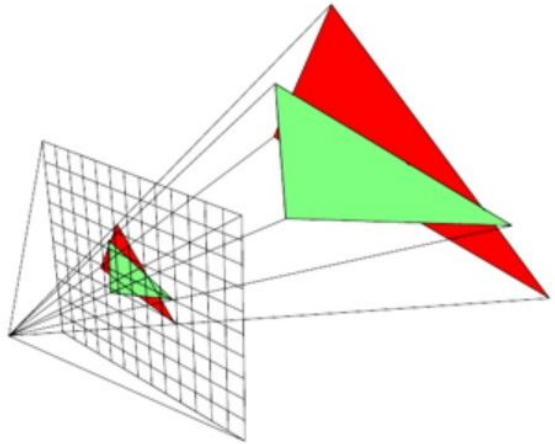


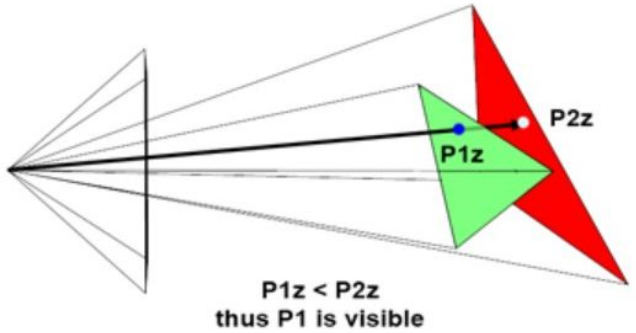
- Раз в 3,000 итераций уменьшаем прозрачность **A** у всех Гауссиан
- Удаляем прозрачные Гауссианы (с небольшой **A**)
- Удаляем слишком плотные Гауссианы, которые заслоняют остальных

Rasterizer

- Изображение разбивается на тайлы 16 x 16
- Гауссианы проецируются на плоскость
- В каждом тайле, Гауссианы сортируются по расстоянию до камеры
- Отсортированные Гауссианы комбинируются, чтобы получить итоговый цвет

- Rasterizer полностью дифференцируемый по **M**, **S**, **C**, **A**





Результаты

- Ни одного нейрона!
- Быстрый rasterizer, быстрое обучение
- Real-time рендер сцены: рендер 30+ кадров в секунду, < 0.03 секунды на кадр
- Качество на уровне со state-of-the-art

Результаты

- Большое потребление GPU памяти при обучении (> 20 GB)
- В труднодоступных местах есть артефакты, особенно, если мало итераций

Бенчмарки

Сравнением с NeRF архитектурами:

- Mip-Nerf360 - state-of-the-art по качеству
- InstantNGP - state-of-the-art по скорости рендера
- Plenoxels - довольно быстрый рендеринг

Метрики:

- SSIM
- PSNR - зашумленность изображений
- LPIPS - perceptual similarity

Dataset	Mip-NeRF360						Tanks&Temples						Deep Blending					
Method|Metric	$SSIM^{\uparrow}$	$PSNR^{\uparrow}$	$LPIPS^{\downarrow}$	Train	FPS	Mem	$SSIM^{\uparrow}$	$PSNR^{\uparrow}$	$LPIPS^{\downarrow}$	Train	FPS	Mem	$SSIM^{\uparrow}$	$PSNR^{\uparrow}$	$LPIPS^{\downarrow}$	Train	FPS	Mem
Plenoxels	0.626	23.08	0.463	25m49s	6.79	2.1GB	0.719	21.08	0.379	25m5s	13.0	2.3GB	0.795	23.06	0.510	27m49s	11.2	2.7GB
INGP-Base	0.671	25.30	0.371	5m37s	11.7	13MB	0.723	21.72	0.330	5m26s	17.1	13MB	0.797	23.62	0.423	6m31s	3.26	13MB
INGP-Big	0.699	25.59	0.331	7m30s	9.43	48MB	0.745	21.92	0.305	6m59s	14.4	48MB	0.817	24.96	0.390	8m	2.79	48MB
M-NeRF360	$0.792^{\dagger}$	$27.69^{\dagger}$	$0.237^{\dagger}$	48h	0.06	8.6MB	0.759	22.22	0.257	48h	0.14	8.6MB	0.901	29.40	0.245	48h	0.09	8.6MB
Ours-7K	0.770	25.60	0.279	6m25s	160	523MB	0.767	21.20	0.280	6m55s	197	270MB	0.875	27.78	0.317	4m35s	172	386MB
Ours-30K	0.815	27.21	0.214	41m33s	134	734MB	0.841	23.14	0.183	26m54s	154	411MB	0.903	29.41	0.243	36m2s	137	676MB

Источники картинок:

- A Short 170 Year History Of Neural Radiance Fields (NeRF), Holograms, And Light Fields | Neural Radiance Fields
- PSNR and SSIM: application areas and criticism (videoprocessing.ai)
- Roundabout at night – gsplat
- Orthogonal projection of an ellipsoïd from N to 2 dimensional space - Mathematics Stack Exchange
- Introduction | 3D Modelling For Programmers (gitbooks.io)