POLITECNICO DI TORINO

Master's Degree in Mathematical Engineering

Master's Degree Thesis

Segmenting dynamic points in 3D scenarios

Supervisors

Prof. Tatiana TOMMASI

Prof. Chiara PLIZZARI

Candidate

Francesco BORGNA

March 2024

Summary

Ma che dici Lorem ipsum dolor sit amet, consectetuer adipiscing elit. Ut purus elit, vestibulum ut, placerat ac, adipiscing vitae, felis. Curabitur dictum gravida mauris. Nam arcu libero, nonummy eget, consectetuer id, vulputate a, magna. Donec vehicula augue eu neque. Pellentesque habitant morbi tristique senectus et netus et malesuada fames ac turpis egestas. Mauris ut leo. Cras viverra metus rhoncus sem. Nulla et lectus vestibulum urna fringilla ultrices. Phasellus eu tellus sit amet tortor gravida placerat. Integer sapien est, iaculis in, pretium quis, viverra ac, nunc. Praesent eget sem vel leo ultrices bibendum. Aenean faucibus. Morbi dolor nulla, malesuada eu, pulvinar at, mollis ac, nulla. Curabitur auctor semper nulla. Donec varius orci eget risus. Duis nibh mi, congue eu, accumsan eleifend, sagittis quis, diam. Duis eget orci sit amet orci dignissim rutrum.

Acknowledgements

ACKNOWLEDGMENTS

"HI"
Goofy, Google by Google

Table of Contents

List of Tables

List of Figures

Acronyms

SfM
: Structure from Motion

pcd
: Pointcloud

MLP
: Multi Layer Perceptron

Part I

Related Works

1. Epic Kitchens
2. Epic Fields
3. Photogrammetry
4. COLMAP
5. NeRF
6. NeuralDiff
7. Monocular Depth Estimation
8. motion estimation
9. (N3F)
10. (Gaussian Splatting)

Chapter 1

Neural Rendering

From CVPR 2020 tutorial [26] Neural Rendering is defined as *"Neural rendering is a new class of deep image and video generation approaches that enable explicit or implicit control of scene properties such as illumination, camera parameters, pose, geometry, appearance, and semantic structure. It combines generative machine learning techniques with physical knowledge from computer graphics to obtain controllable and photo-realistic outputs."*.

1.1 NeRF:Representing Scenes as Neural Radiance Fields for View Synthesis

With NeRF [19] the authors introduced a new state-of-the-art model for synthesizing novel views of complex scenes by just using a set of sparse images and their relative positions.

1.1.1 Intro?

With this work they deal with the novel view synthesis problem of complex optimization by modeling the scene representation with a fully-connected neural network(MLP) without any convolutional layer, this is called *neural radiance field*(NeRF). The network take as input the position of a point (x, y, z) and the direction (θ, ϕ) from which we are looking it and gives as results the color(r, g, b) and density (σ) of that point.

To render a NeRF from a viewpoint one can:1) march camera rays in the scene and sample some points from them 2) use those points as input to the neural network to produce an output set of colors and densities, and 3) use volume rendering techniques to obtain a final 2D image. All the previous steps are differentiable such tat we can use gradient descent to optimize the model by minimizing the error

between each image and the corresponding prediction. Repeating this process from multiple viewpoints encourages the network to grasp the 3D scene representation. In Figure 1.1 is reported an example ,presented in the paper,of the main steps.

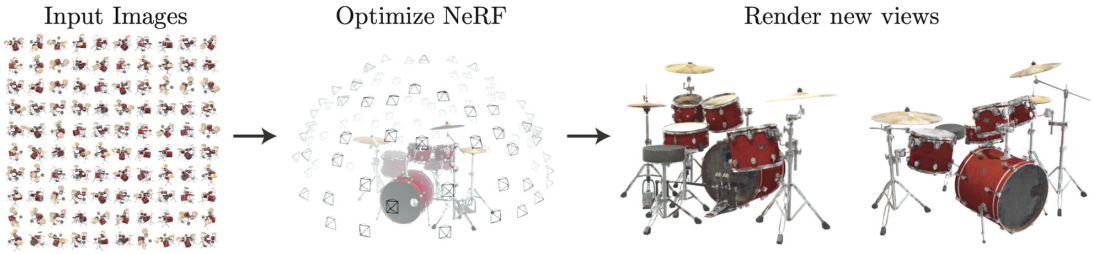

Figure 1.1: NeRF. Optimization of a continous 5D neural radiance field representation(volume density and view-dependent color at any continuous location) of a scene from a seet of input images. The 2D novel views are obtained thanks to classic volume rendering techniques. Here in this example, given 100 images acquired from different viewpoints, they sample two novel views.

Anyway this approch is not enough for a complex scene, in fact the optimization does not converge to a sufficiently high-resolution representation. This problem is solved by trasforming the input coordinates with a positional encoding that enables the network to represent higher frequency functions. Summing up, their contribution consists in:

- An approach for representinc continous scenes as a 5D neural radiance fields.
- A differentiable rendering pipeline based on classical volume rendering techniques,used to optimize the scene representation via input images.
- A positional encoding to map each input 5D coordinate into a higher dimensional space which allows to represent high-frequency scene content.

1.1.2 Related works

At time of writing a promising direction in computer vision consisted in encoding objects and scenes in the weights of an MLP. Hower this method still struggled compared to other methods based on discrete representations like traingle meshes or voxel grids. With their work the writers of the paper managed to obtain state-of-the-art results, denying past disbelief in continous volumetric representation.

Neural 3D shape representations. Implicit continous 3D shapes representation as level sets had been investigated by [27, 28], optimizing signed distance functions or occupancy fields in [29, 30]. These methods were limited by the necessity of a 3D ground truth, which is rarely available. Subsequent works reformulated

the problem using differentiable rendering functions that allowed to use just 2D images as ground truth. Two were the main methods: [31] represents surfaces as 3D occupancy fields; [32]use a representation that outputs a feature vector and RGB color at each continous

coordinate, proposing a differentiable rendering function consisting of a RNN(Recurrent neural network).

Any way these methods have been limited to simple shapes with low complexity, resulting in oversmoothed renderings. The authors strategy of optimizing the networks will reveal themself succesful.

View synthesis and image-based rendering. Photorealistic novel views can be reconstructed if a dense sampling of images have been provided by simple light field sample interpolation techniques [33, 34]. If sparser images are available, some methods have been introduced that rely on extracting traditional geometry and appearance representations from observed images. A popular class of approached uses mesh-based representations of scene with diffuse [35] or view-dependent [36] appearance. Also there are differentiable rasterizers [37, 38, 39] that can directly optimize mesh recontruction to reproduce a set of input images using gradient descent. However due to local minima or poor conditioning of the loss landscape, the optimization is usually difficult.

High-quality photorealistic view synthesis is also performed by volumetric representations, from a set of given RGB images. Volumetric methods can represent complex shapes and materials ,well suited for gradient-based optimization. First works [40] used observed images to directly predict color voxel grids. Later [41, 42, 43] used deep networks to color a sampled volumetric region from some given images. Other works tried using a mix of convolutional networks and sampled voxel grids.

Anyway these methods, being based on discrete sampling of the voxel grid, are restricted to low resolution due to time and computational power. With this paper instead a *continous* representation was proposed, reducing the memory requirements and producing higher quality renderings.

1.1.3 Implementation

A continous 3D scene is represented as a 5D vector-valued function whose input is a 3D coordinate $\mathbf{x} = (x, y, z)$ and 2D viewing direction $\mathbf{d} = (\theta, \phi)$, and whose output is an emited color $\mathbf{c} = (r, g, b)$ and volume density σ. In practice the scene is represented by an MLP network $F_\Theta : (\mathbf{x}, \mathbf{d}) \rightarrow (\mathbf{c}, \sigma)$, where Θ are the weights of the network.

The representation is encouraged to be multiview consistent by restricting the network to predict the volume density σ as a function of just the location $\mathbf{x}$, while the color $\mathbf{c}$ is a function of both the input, location and direction. To do

this the MLP F_Θ first processes the input $\mathbf{x}$ with 8 fully-connected layers (using ReLU activation functions and 256 channels per layer), and outputs σ and a 256-dimensional feature vector. The scheme is summed up in Figure 1.2.

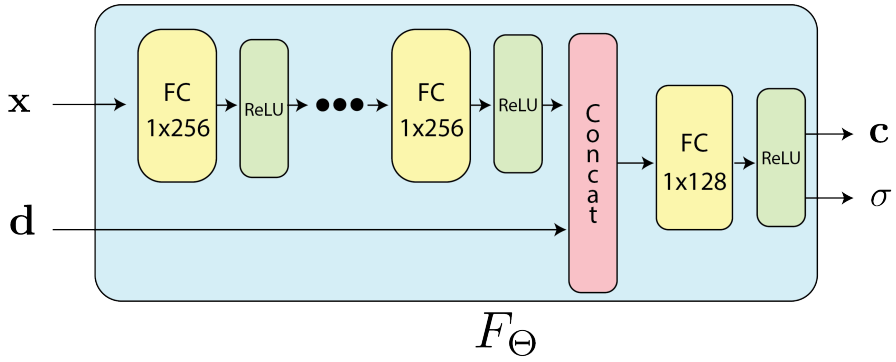

Figure 1.2: F_Θ **Scheme.** The input position $\mathbf{x}$ pass through 8 Fully connected (FC) layers of 256-channels. Each FC layer is followed by a ReLU activation function. This intermediate result is then concatenated with the input direction ($\mathbf{d}$) and fed to one last FC with 128 channels that feeds its output to a ReLU function. The output of the ReLU are the color $\mathbf{c}$ and the volume density (σ).

The effects of the direction in input can be seen in Figure 1.3.

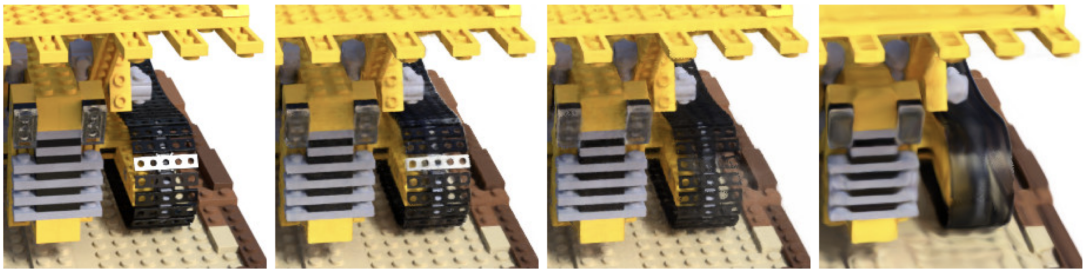

Figure 1.3: Here are reported the results obtained with different strategies, as written underneath each image. In particular removing view dependence prevents the model from recreating the specular reflection on the bulldozer tread. Removing the Positional encoding instead we obtain a blurred image, meaning that high frequencies are not captured nor represented.

Volume Rendering with Radiance Fields

The implicit representation of the scene relies on the volume densities and on the color of every point in that scene. The color of any ray passing through the scene

is rendered using principles from classical volume rendering [44]. **The volume density $\sigma(\mathbf{x})$ can be interpreted as the differential probability of a ray terminating at an infinitesimal particle at location x.** While the expected color C($\mathbf{r}$) of camera ray $\mathbf{r}$(t) = $\mathbf{o}$+ t$\mathbf{d}$ with near and far bounds t_n and t_f is:

$$C(\mathbf{r}) = \int_{t_n}^{t_f} T(t)\sigma(\mathbf{r}(t))\mathbf{c}(\mathbf{r(t),d})\, dt \tag{1.1}$$

where $T(t)$ denotes the accumulated transmittance along the ray from t_n to t,i.e: the probability that the ray travels from t_n to t without hitting any other particle.Namely:

$$T(t) = exp(-\int_{t_n}^{t} \sigma(\mathbf{r}(s))\, ds) \tag{1.2}$$

To obtain a new view in our neural radiance field, we should estimate the $C(\mathbf{r})$ function for each ray passing through each pixel of the focal plane(see Figure 1.4). This

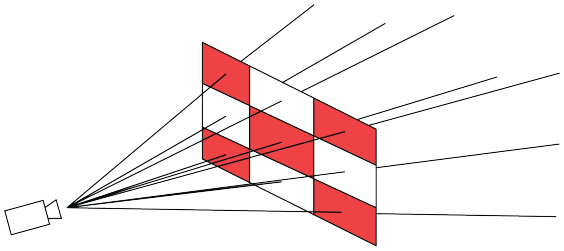

Figure 1.4: Example of rays passing through an image plane of size 3x3 pixels.

integral is approximated using a numerical method known as *quadrature*.Typically deterministic quadrature is used for rendering discretized voxel grids, but in our case it would limit our model's resolution since the network would only be queried at a fixed discrete set of locations. To solve this problem a *stratified* sampling approach has been used, where each interval $[t_n, t_f]$ has been partitioned into N evenly-spaced bins, from which a random sample is then uniformly extracted. Namely:

$$t_i \sim \mathcal{U}[t_n + \frac{i-1}{N}(t_f - t_n), t_n + \frac{i}{N}(t_f - t_n)] \tag{1.3}$$

In this way, even if we are using a discrete set of samples, using stratified sampling allows us to represent a continuous scene representation because the network is evaluated at continuous positions during the training phase. Following the quadrature rule discussed in [45] they used the samples to estimate $C(\mathbf{r})$:

$$\widehat{C}(\mathbf{r}) = \sum_{i=1}^{N} T_i(1 - exp(-\sigma_i\delta_i))\mathbf{c}_i, \quad where \quad T_i = exp(-\sum_{j=1}^{i-1} \sigma_j\delta_j) \tag{1.4}$$

where $\delta_i = t_(i+1)$ is the distance between adjacent samples. It can be noticed that the function that approximate $C(\mathbf{r})$ is differentiable and can be reduced to traditional *alpha compositing*[1] with alpha values being $\alpha_i = 1 - exp(-\sigma_i\delta_i)$.

Optimizing a Neural Radiance Field

In the previous section the core components of a Neural Radiance Field were presented, anyway these partss alone are not able to achieve state-of-the-art results. To improve the quality of the representation two improvements were found succesful:

- **Positional Encoding** of the input coordinates, to encourage the representation of high-frequencies.
- **Hierarchical Sampling** procedure that allows to efficiently sample high-frequency representation.

Positional Encoding. After having found that the model F_Θ performed poorly operating solely on $xyz\theta\phi$ input, in accordance to the work by Rahaman *et al.* [46] that states that deep networks are biased towards learning low-frequency functions; they encode the inputs to a higher dimensional space using high frequency functions.

The new model F_Θ can thus be represented as a combinatino of functions $F_\Theta = F'_\Theta \circ \gamma$, where γ is a function from $\mathbb{R}$ to a higher dimensional space $\mathbb{R}^{2L}$, and F'_Θ is still the basic block introduced in the previous sections. Formally the function used for the encoding part is:

$$\gamma(p) = (sin(2^0\pi p), cos(2^0\pi p), ..., sin(2^L - 1\pi p), cos(2^L - 1\pi p)) \tag{1.5}$$

In particular $\gamma(\dot{)}$ is applied separately to each component of $\mathbf{x}$ and $\mathbf{d}$, after these values are normalized to lie in [−1,1]. Reporting the paper results, they found out that good values of L were: 10 for $\gamma(\mathbf{x})$ and 4 for $\gamma(\mathbf{d})$.

Hierarchical Sampling. Stratified sampling allows to cover continuous regions but anyway is still inefficient: free space and occluded regions that do not contribute to the rendered image are still sampled repeatedly. To solve this problem the authors proposed a *hierarchical* representation which increased rendering efficiency by distributing samples proportionally to their expected effect on the final rendering.

This solution consists in simultaneously optimize two networks: one "coarse" and one "fine". The first step expect to sample a set of N_c points using stratified sampling and feed those to the coarse model. Once this first partial result has been obtained a more informed sampling of points is prooduced. The coarse sampling

[1] *Alpha compositing* is a digital imaging technique used to combine multiple layers of images or graphics with transparency, known as alpha channels.

infact allows to get a rough idea of which parts of the volume are the most relevant. To do this they rewrite the alpha composited color from the coarse model $\hat{C}_c(\mathbf{r})$ in Eq. 1.4 as a weighted sum of all sampled colors c_i along the ray:

$$\hat{C}_c(\mathbf{r}) = \sum_{i=1}^{N_c} w_i c_i, \qquad w_i = T_i(1 - exp(-\sigma_i \delta_i)) \tag{1.6}$$

By normalizing the weights as $\hat{w}_i = \frac{w_i}{\sum_{j=1}^{N_c} w_j}$ we can produce a piecewise-constant probability density function along the ray as seen in Figure 1.5. The next N_f

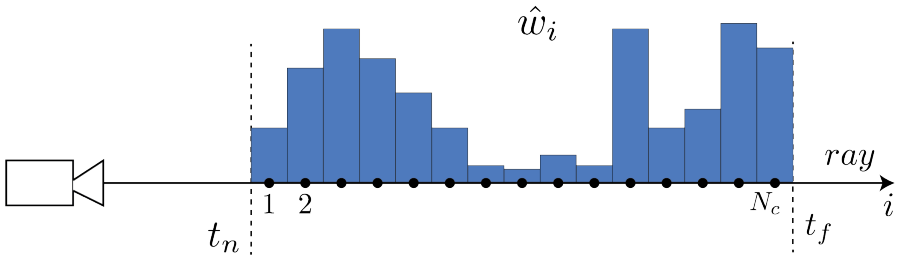

Figure 1.5: PDF of normalized coarse weights $\hat{w}_i$ along a ray with N_c samples.

samples are extraced from this distribution and then, combined to the previous N_c samples, fed to the "fine" network. The final rendered color $\hat{C}_f(\mathbf{r})$ is obtained using Eq. 1.4 with $N_c + N_f$. This procedure revealed succesful in obtaining more samples from regions we expect to contain visible content.

Actual Implementation

To optimize a scene RGB frames are required with the corresponding camera poses and intrinsic parameters(for synthetic data these are easily retrievable from the scene model; while for real images COLMAP was used to extract them). At each iteration a batch of rays from the set of all pixels of all images of the dataset is extracted and following the sampling procedure previosuly described the actual color is predicted for both the "coarse" and the "fine" model. The loss is computed as the total squared error betwwen the rendered and true pixel colors for the two models:

$$\mathcal{L} = \sum_{r \in \mathcal{R}} [\left\| \hat{C}_c(\mathbf{r}) - C(\mathbf{r}) \right\|_2^2 + \left\| \hat{C}_f(\mathbf{r}) - C(\mathbf{r}) \right\|_2^2] \tag{1.7}$$

where $\mathcal{R}$ is the set of rays of each batch, $C(\mathbf{r})$ is the RGB color ground truth and $\hat{C}_c(\mathbf{r})$,$\hat{C}_f(\mathbf{r})$ are the predicted color for the "coarse" and the "fine" model for ray $\mathbf{r}$.

As for some more specific details, they used a batch size of 4086 rays, each sampled at $N_c = 64$ points for the "coarse" model and $N_f = 128$ for the "fine"

model. They used the Adam optimizer with a learning rate beginning at $5x10^{-4}$ and decays exponentially to $5x10^{-}5$ over the course of optimization. Others parameters of the optimizer where left at default values: $\beta_1 = 0.9, \beta_2 = 0.999$ and $\epsilon = 10^{-7}$. Each scene took them around 100-300l iterations to converge on NVIDIA V100 GPU(about 1-2 days). Their tensorflow implementation is provided on `https://github.com/bmild/nerf`.

1.1.4 Results

To validate their results they generated and aggregated some datasets. The main distinction is given by synthetic scenes, denoted as *Diffuse Synthetic 360°* [47] and *Realistic Synthetic 360°*, and realistic scenes,*Real Forward Facing*. More in detail:

- *Diffuse Synthetic 360°*: contains four Lambertian objects with simple geometry. Each object is rendered at 512x512 pixels from viewpoints taken from the upper hemisphere.
- *Realistic Synthetic 360°*: new dataset that the authors introduce containing images of eight objects that exhibit complicated geometry and realistic non-Lambertian[2] materials. Six scenes were sampled from the upper hemisphere while the remaining two are rendered from a full sphere. For each scene 100 views are captured for training and 200 for testing, all at 800x800 pixels.
- *Real Forward Facing*: consists of 8 scenes captured with a handheld cellphone (5 taken from the LLFF [48] paper and 3 captured by them), captured with 20 to 62 images, holding out 1/8 of these for the test set. All images have a resolution of 1008x756 pixels.

The algorithm which have been compared were the top-performing techniques for view synthesis and are the following:

- **Neural Volumes (NV)** [49] synthesizes novel views of objects lying in a confined space with a distinct background(which must be captured alone). The model is based on a 3D convolutional network to predict a discretized RGBα voxel grid.
- **Scene Representation Networks (SRN)** [50] represent a continuous scene as an opaque surface, implicitly defined by an MLP that maps spatial coordinates to a feature vector. The color is then obtained by training a RNN along the ray.

[2]Lambertian reflectance is the property that defines an ideal "matte" or diffusely reflecting surface. The apparent brightness of a Lambertian surface to an observer is the same regardless of the observer's angle of view [**lambertian**].

- **Local Light Field Fusion (LLFF)** [48] is designed for producing photorealistic novel views for well-sampled forward facing scenes.

Quantitative results are reported from the paper in Table 1.1. The metrics used were: PSNR,SSIM and LPIPS (see Section **??** for more details). We can see that the method introduced outperformed past works in both realistic and synthetic scenarios.

	Diffuse Synthetic 360°			Realistic Synthetic 360			Real Forward-Facing		
Method	PSNR ↑	SSIM ↑	LPIPS ↓	PSNR ↑	SSIM ↑	LPIPS ↓	PSNR ↑	SSIM ↑	LPIPS ↓
SRN	33.20	0.963	0.073	22.26	0.846	0.170	22.84	0.668	0.378
NV	29.62	0.929	0.099	26.05	0.893	0.160	-	-	-
LLFF	34.38	0.985	0.048	24.88	0.911	0.114	24.13	0.798	**0.212**
NeRF	**40.15**	**0.991**	**0.023**	**31.01**	**0.947**	**0.081**	**26.50**	**0.811**	0.250

Table 1.1: Quantitative results In all datasets, for all metrics,except for the LPIPS, the NeRF method outperforms old methods.

A qualitative idea instead is given by Figure 1.6 and 1.7.

An additional validation of their design choices is given by an ablation study on the various parts that have been discussed in the implementation part. In particular the result of the study is reported in Table 1.2.

	Input	#Im.	L	(N_c, N_f)	PSNR ↑	SSIM ↑	LPIPS ↓
1) No PE, VD, H	xyz	100	-	$(256, -)$	26.67	0.906	0.136
2) No Pos. Encoding	$xyz\theta\phi$	100	-	(64,128)	28.77	0.924	0.108
3) No View Dependence	xyz	100	10	(64,128)	27.66	0.925	0.117
4) No Hierarchical	$xyz\theta\phi$	100	10	$(256, -)$	30.06	0.938	0.109
5) Far Fewer Images	$xyz\theta\phi$	25	10	(64,128)	27.78	0.925	0.107
6) Fewer Images	$xyz\theta\phi$	50	10	(64,128)	29.79	0.940	0.096
7) Fewer Frequencies	$xyz\theta\phi$	100	5	(64,128)	30.59	0.944	0.088
8) More Frequencies	$xyz\theta\phi$	100	15	(64,128)	30.81	0.946	0.096
9) Complete Model	$xyz\theta\phi$	100	10	(64,128)	**31.01**	**0.947**	**0.081**

Table 1.2: Quantitative results In all datasets, for all metrics,except for the LPIPS, the NeRF method outperforms old methods.

1.2 NeuralDiff: Segmenting 3D objects that move in egocentric videos

NeuralDiff is a neural radiance field adapted to distinguish three different parts of egocentric videos:1) *static*, that is everything that does not move,2) *foreground*,

which is everything that at some point of the video moves, and 3) actor, which comprehends the body parts of the person who is wearing the camera.

Anyway this is not an easy task. Infact motion in egocentric vision is a complex attribute to identify. We need to distinguish what is independent of the camera motion while dealing with the camera large viewpoint change and parallax that generate a large apparent motion.

In a static camera scenario the problem of separating the foreground from the static background would be easily solved by recurring to background subtraction techniques, but the parallax effect of a moving camera makes this technique useless. As an example provided in the paper we can think of a egocentric video of a person cooking: the actor behave in the scene by moving (and trasforming) objects. However, egomotion is the dominant effect since objects move only sporadically, and in a way that is hardly distinguishable from the much larger apparent motion induced by the viewpoint change, making it very difficult to segment dynamic objects automatically.

Even motion segmentation techniques struggles to separate a scene in different motion components, since they require correspondences, they reason locally, across a handful frames and usually avoid explicit 3D reasoning, making it difficult to treat egocentric videos with many small objects that move only occasionally throughout a long sequence.

With this work the authors take inspiration from neural rendering techniques [19] to create a motion analysis tool to obtain their goal. In particular they leverage the ability of reconstructing accurately 3D scenes for recovering the background and then build on top of it the dynamic parts. In fact 3D objects that are manipulated in the video also present a significant structure. They usally move in "bursts", changing their state when they are involved in an interaction, otherwise staying rigidly attached to the background. Exploiting this property they extend the neural render to reconstruct the omoving object appearance using a slowly-varying time encoding. The last part that show unique properties in the scene is the actor due to its continuous movement while occluding the scene and with a motion linked to the camera.

Summing up they ended up with a three-stream neural rendering architecture, where each stream models respectively: *static background*,*dynamic foreground* and the *actor*. These are then combined to explain the video as a whole. In Figure 1.8 is reported a general overview of NeuralDiff capabilities.

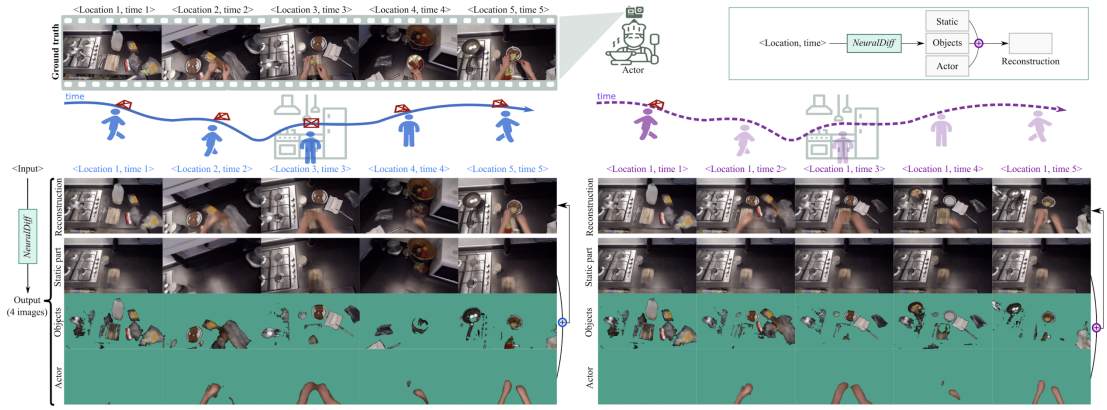

Figure 1.8: Given an egocentric video with camera reconstruction, NeuralDiff, a neural architecture, learns how to decompose each frame into a static background and a a dynamic foreground, which includes every object that sooner or later will move and the actor's body parts. Each of these streams is learned exploiting the characteristics of the scene that is going to be captured. Being a neural radiance field, NeuralDiff is also capable to render images from novel viewpoints as can be seen in the bottom right part of the scene.

Each stream is designed differently, in order to include inductive biases that match the statistics of each layer(background,foreground,actor). The resulting analysis-via-synthesis method shows that neural rendering techniques are also useful for analysis, and not just synthesis. In particular NeuralDiff was the first to demonstrate the effectiveness of these techniques in interpreting challenging egocentric videos, providing indications for the extraction of moving objects in scenes with a complex 3D structure and dynamics.

For evaluating their results, they augmented the EPIC-KITCHENS dataset [1] by manually segmenting all objects that move at some point. In this way they are able to assess the quality of the decompositino of the different components. In addition they defined a new benchmark for measuring progress in the challenging task of dynamic object segmentation in complex videos, aiming to promote research in the area.

1.2.1 Related Work

Background Subtraction

Background subtraction techniques have been used to detect moving objects in videos(see [51]). The simplest way to obtain the background would be to capture a background image that does not contain any foreground object. Unfortunately in some scenes it is not possible and it could be changed under critical situations

like illumination changes, objects appeearing and disappearing from the scene. Some expedients have been introduced that try to predict the background via a background initialization step, which base its guess on the first few frames of the video. Anyway egocentric videos still contains too much challenging problems(light change,multiple different objects,moving camera ,etc.) that do not allow background subtraction to be a viable solution.

Motion segmentation

Motion segmentation consists in decomposing a video into individually moving objects [52]. Amongst others, it has applications in robotics, traffic monitoring, sports analysis, inspection, video surveillance and compression. Anyway these techniques usually relies on optical flow, which can be subject to some ambiguities like, in our case,motion parallax. Even occlusion plays an important role. Motion segmentation struggles for example when an object moves in front of or behind other objects in the scene, leading to ambiguity in the flow field. Also, many methods fails when a dynamic object temporarilly remain static.

All these problems prevent to apply motion segmentation algorithms to egocentric videos.

Discovering and segmenting objects in videos

Discovering and segmenting objects in videos is related to background subtraction and motion segmentation. As instance moving objects can be segmented from the background by using a probabilistic model that acts on optical flow [53]. In [54], pixel trajectories and spectral clustering are combined to produce motion segments. Some recent works revisits classical motion segmentation techniques from a data-driven perspective [55], *e.g.* using physical motion cues to learn 3D representations or learning a scene representation using neural rendering 1.1.

With NeuralDiff they obtain a holistic representation capable of handling occlusions and detachable objects.

Neural rendering

This technique allows to synthesize novel views using a neural architecture and volume rendering. It was introduced with NeRF (see [19] or Section 1.1). It was designed for static scenes and it poorly handles dynamic scenes and occlusions, so research was performed to apply it to dynamic scenes(see NeRF-W [56]). Other research direction to model 3D geometry of one or more objects is studied in [57]. Another direction tried focusing on modeling dynamic scenes mostly with monocular videos as input [58, 59]. Most of this approaches use a canonical model in conjunction with a deformation network, or warp space [59], starting from a

canonical volume. Closer to the work presented is [58], where a static NeRF model is combined with a dynamic one.

Anyway none of the previous managed to segment 3D objects in such long and challenging videos.

1.2.2 Implementation

The goal of this method is to extract a mask from an input video sequence that discern foreground objects from background objects. It is worth noting that in this paper *background* is the part of the scene that remain static throughout the entire video; while *foreground* is defined as any object that moves independently of the camera in at least one frame.

The basic block upon which NeuralDiff is built upon is NeRF(see Section 1.1) that can predict the appearance of a static object,*i.e.* the background, from different viewpoints. The authors also suggest a modification of the basic NeRF which is capable of distinguish foreground objects and it will be described in the following sections.

Neural rendering

As reported in Section 1.1, a video x is a collection $(x_t)_{t\in[0,...,T-1]}$ of T RGB frames $x_t \in \mathbb{R}^{3xHxW}$. Each of these frames can be seen as a function $x_t = h(B, F_t, g_t)$, where: B is the static background, F_t is the variable Foreground and g_t is the moving camera. The motion and parameters of the camera are assumed to be known, usually being extracted via a SfM algorithm such as COLMAP [60] which is explained in detail in Section **??**. The background and foreground layers include the shape and reflectance of the 3D surfaces in the scene as well as the illumination.

Instead of trying to invert the function h to get B and F_t, which is known as *inverse rendering*, neural rendering directly learns h using a neural network f, $h(B, F_t, g_t) = f(g_t, t)$, provided the time and the camera viewpoint for the frame x_t. Viewpoint g and time t can be factorized by a careful desing of the function f, thus f can be used to generalize new unobserved viewpoints. In the basic NeRF implementation the main assumption is that the scene is static, meaning that F_t is empty and f can be rewritten as $f(g_t)$. The color of each frame is then obtained by a volumetric sampling process that simulates ray casting. More in detail the pixel color $x_{ut} \in \mathbb{R}^3$ of pixel $u \in \Omega = \{0, ..., H-1\} \times \{0, ..., W-1\}$ is obtained by averaging the color of the 3D points along the ray $g_t r_k$ weighted by the probability that a photon emanates from the point and reaches the camera.

A neural network, $(\sigma_k^b, c_k^b) = MLP^b(g_t r_k, d_t)$ retrieve the density $\sigma_k^b \in \mathbb{R}_+$ and the color $c_k^b \in \mathbb{R}^3$ of each point $g_t r_k$, where the superscript b refers to the fact that we are dealing with the static background and d_t is the unit-norm viewing

direction.

A photon while travelling along the segment (r_k, r_{k+1}) is transmitted with probability $T_k^b = e^{-\delta_k \sigma_k^b}$ where the quantity $\delta_k = |r_{k+1} - r_k|$ is the length of the segment. This definition allows to calculate the probability of transmission across several segments as the product of the individual transmission probabilities. The color of pixel u can thus be written as:

$$x_{ut} = f_u(g_t) = \sum_{k=0}^{M} v_k (1 - T_k^b) c_k^b, \quad v_k = \prod_{q=0}^{k-1} T_q^b \tag{1.8}$$

The model is trained by minimizing the reconstruction error $\|x - f(g_t)\|$.

Dynamic Components

To econstruct egocentric videos anyway we will deal with moving objects, so we can not neglect the foreground F_t. To capture this layer they propose to build on top of the background MLP^b a foreground-specific MLP^f,$(\sigma_k^f, c_k^f, \beta_k^f) = MLP^f(g_t r_k, z_t^f)$. Its outputs are a 'foreground' occupancy σ^f and color c^f. It also predict an uncertainty score β_k^f whose role is discussed in the next paragraphs. The variable z_t^f is introduced to capture the properties of the foreground that change over time.

The color x_{ut} of a pixel u is obtained by the composition of both background and foreground, so $\mathcal{S} = b, f$:

$$x_{ut} = f_u(g_t, z_t) = \sum_{k=0}^{M} v_k (\sum_{p \in \mathcal{S}} w^p(T_k) c_k^p) \quad where \quad v_k = \prod_{q=0}^{k-1} \prod_{p \in \mathcal{S}} T_q^p \tag{1.9}$$

The factor v_k reqquires a photon to be transmitted from the camera to point r_k through different materials. The weights $w^p(T_k)$ mix the colors based on points densities. As done in NeRF-W [21]:

$$w^p(T_k) = 1 - T_k^p \in [0,1] \tag{1.10}$$

Smooth Dynamics The model can now be optimized by minimizing the loss across all input frames:

$$\min_{f, z_1, \ldots z_T} \frac{1}{T|\Omega|} \sum_{t=1}^{T} \sum_{u \in \Omega} \|x_t - f(g_t, z_t)\|^2 \tag{1.11}$$

However there is a problem with the characteristics of foreground, because, even if dynamic, it does not change at every frame in fact most of objects spend most of their time rigidly attached to the background. This make the dependecy on independent frame-specific codes z_t almost useless. They came up with the idea of replacing it with a low-rank expansion of the trajectory of sates, taking $z_t = B(t)\Gamma$

where $B(t) \in \mathbb{R}^p$ is a fixed basis and the motion $\Gamma \in \mathbb{R}^{P \times D}$ are coefficients such that $P << T$. In particular $B(t) = [1, t, sin2\pi t, cos21pit, sin4\pi t, cos4\pi t, ...]$

Improved Geometry: capturing the actor The foreground layer can be divided again in objects manipulated by the actor, that can move sporadically, and the actor, which instead moves continually. To detect the actor a third MLP is integrated to the previous model. The actor's MLP is similar to the foreground MLP: $(\sigma_k^a, c_k^a.\beta_k^a) = MLP^a(r_k, z_t^a)$. The main difference is that this time the 3D point r_k is expressed with respect to the camera, and not to the world $(g_t r_k)$. This follows the physical properties of the recording stage, the camera is in fact attached to the actor head, making his body parts almost always present in front of the camera therefore it shows a reduced variability in the reference frame of the camera. On the contrary, the background and foreground is invariant with respect to the world reference system.

Improved Color Mixing A principled mixing model is proposed to replace Equation 1.10. It is obtained by dividing the segment δ_k in Pn sub-segments, alternating between the P different materials($P = |\mathcal{S}|$, $e.g. P = 3$ if all three layers are considered). They show that the probability that a photon is absorbed in a subsegment of material p is given by:

$$w^p(T_k) = \frac{\sigma_k^p}{\sum_{q=1}^{P} \sigma_k^q}(1 - \prod_{q=1}^{P} T_k^q) \tag{1.12}$$

where the first term represent the probability that a given material p is responsible for the absorption. The latter one instead is the probability that the photon is absorbed by all the materials involved.

Uncertainty and regularization

Uncertainty. Here we clarify the role of the previously announced β_k^p variable. It is predicted from each MLPs ($\beta_k^b = 0$ for the background) and represent the uncertainty of the color associated to each 3D point along the ray r_k for each layer p as pseudo-standard deviations(StDs). As reported in [56], the StD of a rendered color x_{ut} is given as the sum of all the StDs for each material: $\beta_{ut} = \sum_p \beta_{ut}^p$ where β_{ut}^p is obtained from Equation 1.9 by replacing c_k^p with β_k^p. Now we can introduce a probabilistic loss:

$$\mathcal{L}_{prob}(f, z_t | x_t, g_t, u) = \frac{\|x_{ut} - f_u(g_t, z_t)\|^2}{2\beta_{ut}^2} \tag{1.13}$$

Sparsity. The occupancy of the foreground and actor components is further penalized by using:

$$\mathcal{L}_{sparse}(f, z_t | x_t, g_t, u) = \sum_{p=1}^{P} \sum_{k=0}^{M} \sigma_k^p \tag{1.14}$$

Training Loss. Finally, the loss used for training the model is:

$$\mathcal{L} = \mathcal{L}_{prob} + \lambda \mathcal{L}_{sparse} \tag{1.15}$$

where $\lambda > 0$ is a weight set to 0.01.

1.2.3 EPIC-Diff benchmark

The goa of this paper was to identify any 'detachable' object, namely a onject that moves independently from the camera. An extension of EPIC-KITCHENS [1] was introduced to give an evaluation to their method.

Data Selection.10 videos sequences,or also called *scene*, were selected from EPIC-Kitchen, each lasting 14 minutes on average. Then 1000 frames where sampled from each and fed to COLMAP [60] to obtain camera reconstructions. The scenes followed these constraints. 1)The videos should contain different viewpoints and multiple manipulated objects.2) COLMAP should reconstruct the sequence with at least 600 frames. In the end the obtain 10 sequence with an average of 900 frames.

Data annotation Being an unsupervised algorithm, the only data annotations that collected were for testing and validation. They uniformly hold out 56 frames on average for validation(for setting parameters) and for testing. The latter were manually annotated with segmentation pixelwise binary masks to assess static/dynamic components. The test frames are not used for training such that they can be used also to also evaluate novel-view synthesis. In total they obtained 560 manual image-level segmentation masks. An example can be see in Figure 1.9

Evaluation. The task evaluated is background subtraction. To accomplish this they used standard segmentation metrics: each frames is demposed in its pixels and a mask is extracted from the predicted frame. Then it is compared with its ground-truth calculating average precision(AP). Each AP is then averaged eith eevery frame and scene to obtain mean average precision (mAP).

For novel view synthesis instead PSNR is used. Specifically they provided,using the ground-truth masks, the PSNR of the static and of the moving parts.

Results. The experimentts are based on a PyTTorch implementation of the model that has been published on `https://github.com/dichotomies/NeuralDiff`, more details can be found in the paper.

The Baselines compared are:

(1)**NeRF** [19] which uses a single stream to predict static scenes.

(2)**NeRF-BF** trains two NeRF models in parallel, one for the Background (B) and the other for the foreground (F). The latter stream is conditioned on time by passing a positional-encoded version of the time variable, that in this case corresponds with the frame number. This differ from NeuralDiff time encoding for Equation 1.11.

(3)**NeRF-W** [56] also contains two interlinked background and foreground streams. Anyway it was designed to deal with image collection, so it struggles to adapt to videos. Some adjustments were done to adapt it to this task, more details are reported in the paper.

In Table 1.3 are reported the results for the various model previously presented for the evaluation of their capacity to discover and segment 3D objects but also reconstruct dynamic scenes. It is worth noting that NeuralDiff largely improve the results with respect to other methods. Also the temporal information confirms to be fundamental for improving the performance over NeRF and NeRF-W. As a third observation each update proposed upon the vanilla NeuralDiff proveed to be succesful. Hoewever we have to remind that these metrics do not reflect the ability of the full model to separate foreground into objects and actor, since they are merged together in the annotated masks. This ability can be seen in Figure 1.8.

Method	mAP	PSNR	PSNR $_b$	PSNR $_f$
NeRF [19]	47.8	20.9	22.8	17.6
NeRF-W [21]	59.2	23.2	26.4	18.9
NeRF-BF	64.4	23.8	26.8	19.6
NeuralDiff	66.7	24.0	27.2	19.8
NeuralDiff+A	**69.1**	24.1	**27.3**	19.9
NeuralDiff+C	67.4	24.1	27.2	19.9
NeuralDiff+C+A	67.8	**24.2**	**27.3**	**20.0**

Table 1.3: Results

Figure 1.6: Comparison on test images from the newly introduced synthetic dataset. NeRF method is able to recover fine details in both geometry and appearance. LLFF exhibits some artifacts on the microphone and some ghosting artifact in the other scenes. SRN produces distorted and blurry rendering for every scene. Neural Volumesstruggle capturing details we can see from the ship reeconstruction.

Figure 1.7: Comparison on the test set of the real images. As expected LLFF is performing pretty well being projected for this specific use case(forward-facing captures of real scenes). Anyway NeRF is able to represent fine geometry more consistently across rendered views than LLFF as we can see in Fern's and in T-rex. NeRF is also able to reproduce partially occluded scene as in the second row. SRN instead completely fail to represent any high-frequency content.

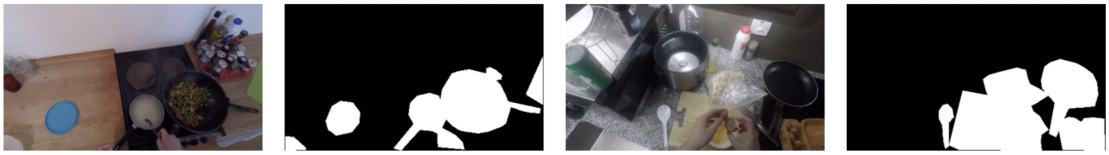

Figure 1.9: Examples of frames with their corresponding manually binary pixelwise mask

Appendix A

Galileo

```
1    import os
2    os.system("echo 1")
```

$\mathcal{O}(n \log n)$
`numpy`

Appendix B

Math Notation

$$\mathbf{a} \times \mathbf{b} = [\mathbf{a}]_{\times} \mathbf{b} = \begin{bmatrix} 0 & -a_3 & a_2 \\ a_3 & 0 & -a_1 \\ -a_2 & a_1 & 0 \end{bmatrix} \begin{bmatrix} b_1 \\ b_2 \\ b_3 \end{bmatrix}$$

$$\mathbf{a} \times \mathbf{b} = [\mathbf{b}]_{\times}^{T} \mathbf{a} = \begin{bmatrix} 0 & b_3 & -b_2 \\ -b_3 & 0 & b_1 \\ b_2 & -b_1 & 0 \end{bmatrix} \begin{bmatrix} a_1 \\ a_2 \\ a_3 \end{bmatrix}$$

Bibliography

[1] Dima Damen et al. «Scaling Egocentric Vision: The EPIC-KITCHENS Dataset». In: *European Conference on Computer Vision (ECCV)*. 2018 (cit. on pp. 13, 17).

[2] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. «Deep Residual Learning for Image Recognition». In: *CoRR* abs/1512.03385 (2015). arXiv: 1512.03385. URL: http://arxiv.org/abs/1512.03385.

[3] Shaoqing Ren, Kaiming He, Ross B. Girshick, and Jian Sun. «Faster R-CNN: Towards Real-Time Object Detection with Region Proposal Networks». In: *CoRR* abs/1506.01497 (2015). arXiv: 1506.01497. URL: http://arxiv.org/abs/1506.01497.

[4] Andrej Karpathy and Li Fei-Fei. «Deep Visual-Semantic Alignments for Generating Image Descriptions». In: *CoRR* abs/1412.2306 (2014). arXiv: 1412.2306. URL: http://arxiv.org/abs/1412.2306.

[5] Stanislaw Antol, Aishwarya Agrawal, Jiasen Lu, Margaret Mitchell, Dhruv Batra, C. Lawrence Zitnick, and Devi Parikh. «VQA: Visual Question Answering». In: *Proceedings of the IEEE International Conference on Computer Vision (ICCV)*. Dec. 2015.

[6] Mark Everingham, Luc Van Gool, Christopher K. I. Williams, John M. Winn, and Andrew Zisserman. «The Pascal Visual Object Classes (VOC) Challenge.» In: *Int. J. Comput. Vis.* 88.2 (2010), pp. 303–338. URL: http://dblp.uni-trier.de/db/journals/ijcv/ijcv88.html#EveringhamGWWZ10.

[7] Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. «ImageNet: A large-scale hierarchical image database». In: *2009 IEEE Conference on Computer Vision and Pattern Recognition*. 2009, pp. 248–255. DOI: 10.1109/CVPR.2009.5206848.

[8] Tsung-Yi Lin et al. «Microsoft COCO: Common Objects in Context». In: *CoRR* abs/1405.0312 (2014). arXiv: 1405.0312. URL: http://arxiv.org/abs/1405.0312.

[9] Bolei Zhou, Hang Zhao, Xavier Puig, Sanja Fidler, Adela Barriuso, and Antonio Torralba. «Scene Parsing through ADE20K Dataset». In: *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. 2017, pp. 5122–5130. DOI: 10.1109/CVPR.2017.544.

[10] Raghav Goyal et al. «The "something something" video database for learning and evaluating visual common sense». In: *CoRR* abs/1706.04261 (2017). arXiv: 1706.04261. URL: http://arxiv.org/abs/1706.04261.

[11] Sami Abu-El-Haija, Nisarg Kothari, Joonseok Lee, Paul Natsev, George Toderici, Balakrishnan Varadarajan, and Sudheendra Vijayanarasimhan. «YouTube-8M: A Large-Scale Video Classification Benchmark». In: *CoRR* abs/1609.08675 (2016). arXiv: 1609.08675. URL: http://arxiv.org/abs/1609.08675.

[12] Anna Rohrbach, Marcus Rohrbach, Niket Tandon, and Bernt Schiele. «A Dataset for Movie Description». In: *CoRR* abs/1501.02530 (2015). arXiv: 1501.02530. URL: http://arxiv.org/abs/1501.02530.

[13] Makarand Tapaswi, Yukun Zhu, Rainer Stiefelhagen, Antonio Torralba, Raquel Urtasun, and Sanja Fidler. «MovieQA: Understanding Stories in Movies through Question-Answering». In: *CoRR* abs/1512.02902 (2015). arXiv: 1512.02902. URL: http://arxiv.org/abs/1512.02902.

[14] David F. Fouhey, Weicheng Kuo, Alexei A. Efros, and Jitendra Malik. «From Lifestyle Vlogs to Everyday Interactions». In: *CoRR* abs/1712.02310 (2017). arXiv: 1712.02310. URL: http://arxiv.org/abs/1712.02310.

[15] Gunnar A. Sigurdsson, Gül Varol, Xiaolong Wang, Ali Farhadi, Ivan Laptev, and Abhinav Gupta. «Hollywood in Homes: Crowdsourcing Data Collection for Activity Understanding». In: *CoRR* abs/1604.01753 (2016). arXiv: 1604.01753. URL: http://arxiv.org/abs/1604.01753.

[16] Dima Damen et al. «Rescaling Egocentric Vision: Collection, Pipeline and Challenges for EPIC-KITCHENS-100». In: *International Journal of Computer Vision (IJCV)* 130 (2022), pp. 33–55. URL: https://doi.org/10.1007/s11263-021-01531-2.

[17] Dandan Shan, Jiaqi Geng, Michelle Shu, and David F. Fouhey. *Understanding Human Hands in Contact at Internet Scale*. 2020. arXiv: 2006.06669 [cs.CV].

[18] Rodrigo Benenson, Stefan Popov, and Vittorio Ferrari. *Large-scale interactive object segmentation with human annotators*. 2019. arXiv: 1903.10830 [cs.CV].

[19] 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*. 2020. arXiv: 2003.08934 [cs.CV] (cit. on pp. 3, 12, 14, 18, 19).

[20] Vadim Tschernezki, Diane Larlus, and Andrea Vedaldi. «NeuralDiff: Segmenting 3D objects that move in egocentric videos». In: *CoRR* abs/2110.09936 (2021). arXiv: 2110.09936. URL: https://arxiv.org/abs/2110.09936.

[21] Ricardo Martin-Brualla, Noha Radwan, Mehdi S. M. Sajjadi, Jonathan T. Barron, Alexey Dosovitskiy, and Daniel Duckworth. *NeRF in the Wild: Neural Radiance Fields for Unconstrained Photo Collections*. 2021. arXiv: 2008.02268 [cs.CV] (cit. on pp. 16, 19).

[22] Hang Gao, Ruilong Li, Shubham Tulsiani, Bryan Russell, and Angjoo Kanazawa. *Monocular Dynamic View Synthesis: A Reality Check*. 2022. arXiv: 2210.13445 [cs.CV].

[23] Charig Yang, Hala Lamdouar, Erika Lu, Andrew Zisserman, and Weidi Xie. *Self-supervised Video Object Segmentation by Motion Grouping*. 2021. arXiv: 2104.07658 [cs.CV].

[24] Ahmad Darkhalil, Dandan Shan, Bin Zhu, Jian Ma, Amlan Kar, Richard Higgins, Sanja Fidler, David Fouhey, and Dima Damen. *EPIC-KITCHENS VISOR Benchmark: VIdeo Segmentations and Object Relations*. 2022. arXiv: 2209.13064 [cs.CV].

[25] Kristen Grauman et al. *Ego4D: Around the World in 3,000 Hours of Egocentric Video*. 2022. arXiv: 2110.07058 [cs.CV].

[26] URL: https://justusthies.github.io/posts/neuralrenderingtutorial_cvpr/#:~:text=Neural%20rendering%20is%20a%20new,%2C%20appearance%2C%20and%20semantic%20structure (cit. on p. 3).

[27] Chiyu Max Jiang, Avneesh Sud, Ameesh Makadia, Jingwei Huang, Matthias Nießner, and Thomas Funkhouser. *Local Implicit Grid Representations for 3D Scenes*. 2020. arXiv: 2003.08981 [cs.CV] (cit. on p. 4).

[28] Jeong Joon Park, Peter Florence, Julian Straub, Richard Newcombe, and Steven Lovegrove. *DeepSDF: Learning Continuous Signed Distance Functions for Shape Representation*. 2019. arXiv: 1901.05103 [cs.CV] (cit. on p. 4).

[29] Kyle Genova, Forrester Cole, Avneesh Sud, Aaron Sarna, and Thomas Funkhouser. *Local Deep Implicit Functions for 3D Shape*. 2020. arXiv: 1912.06126 [cs.CV] (cit. on p. 4).

[30] Lars Mescheder, Michael Oechsle, Michael Niemeyer, Sebastian Nowozin, and Andreas Geiger. *Occupancy Networks: Learning 3D Reconstruction in Function Space*. 2019. arXiv: 1812.03828 [cs.CV] (cit. on p. 4).

[31] Michael Niemeyer, Lars Mescheder, Michael Oechsle, and Andreas Geiger. *Differentiable Volumetric Rendering: Learning Implicit 3D Representations without 3D Supervision*. 2020. arXiv: 1912.07372 [cs.CV] (cit. on p. 5).

[32] Vincent Sitzmann, Michael Zollhöfer, and Gordon Wetzstein. *Scene Representation Networks: Continuous 3D-Structure-Aware Neural Scene Representations*. 2020. arXiv: 1906.01618 [cs.CV] (cit. on p. 5).

[33] Marc Levoy and Pat Hanrahan. «Light field rendering». In: *Proceedings of the 23rd annual conference on Computer graphics and interactive techniques*. SIGGRAPH96. ACM, Aug. 1996. DOI: 10.1145/237170.237199. URL: http://dx.doi.org/10.1145/237170.237199 (cit. on p. 5).

[34] Steven Gortler, Radek Grzeszczuk, Richard Szeliski, and Michael Cohen. «The Lumigraph». In: *Proc. of SIGGRAPH 96* 96 (Aug. 2001). DOI: 10.1145/237170.237200 (cit. on p. 5).

[35] Michael Waechter, Nils Moehrle, and Michael Goesele. «Let There Be Color! Large-Scale Texturing of 3D Reconstructions». In: *European Conference on Computer Vision*. 2014. URL: https://api.semanticscholar.org/CorpusID:6085476 (cit. on p. 5).

[36] Chris Buehler, Michael Bosse, Leonard McMillan, Steven J. Gortler, and Michael F. Cohen. «Unstructured lumigraph rendering». In: *Proceedings of the 28th annual conference on Computer graphics and interactive techniques* (2001). URL: https://api.semanticscholar.org/CorpusID:215780580 (cit. on p. 5).

[37] Wenzheng Chen, Jun Gao, Huan Ling, Edward J. Smith, Jaakko Lehtinen, Alec Jacobson, and Sanja Fidler. *Learning to Predict 3D Objects with an Interpolation-based Differentiable Renderer*. 2019. arXiv: 1908.01210 [cs.CV] (cit. on p. 5).

[38] Kyle Genova, Forrester Cole, Aaron Maschinot, Aaron Sarna, Daniel Vlasic, and William T. Freeman. *Unsupervised Training for 3D Morphable Model Regression*. 2018. arXiv: 1806.06098 [cs.CV] (cit. on p. 5).

[39] Shichen Liu, Weikai Chen, Tianye Li, and Hao Li. *Soft Rasterizer: Differentiable Rendering for Unsupervised Single-View Mesh Reconstruction*. 2019. arXiv: 1901.05567 [cs.CV] (cit. on p. 5).

[40] K.N. Kutulakos and S.M. Seitz. «A theory of shape by space carving». In: *Proceedings of the Seventh IEEE International Conference on Computer Vision*. Vol. 1. 1999, 307–314 vol.1. DOI: 10.1109/ICCV.1999.791235 (cit. on p. 5).

[41] John Flynn, Michael Broxton, Paul Debevec, Matthew DuVall, Graham Fyffe, Ryan Overbeck, Noah Snavely, and Richard Tucker. *DeepView: View Synthesis with Learned Gradient Descent*. 2019. arXiv: 1906.07316 [cs.CV] (cit. on p. 5).

[42] Philipp Henzler, Volker Rasche, Timo Ropinski, and Tobias Ritschel. *Single-image Tomography: 3D Volumes from 2D Cranial X-Rays*. 2018. arXiv: 1710.04867 [cs.GR] (cit. on p. 5).

[43] Abhishek Kar, Christian Häne, and Jitendra Malik. «Learning a Multi-View Stereo Machine». In: *ArXiv* abs/1708.05375 (2017). URL: https://api.semanticscholar.org/CorpusID:19285959 (cit. on p. 5).

[44] James T. Kajiya and Brian Von Herzen. «Ray tracing volume densities». In: *Proceedings of the 11th annual conference on Computer graphics and interactive techniques* (1984). URL: https://api.semanticscholar.org/CorpusID:6722621 (cit. on p. 7).

[45] N. Max. «Optical models for direct volume rendering». In: *IEEE Transactions on Visualization and Computer Graphics* 1.2 (1995), pp. 99–108. DOI: 10.1109/2945.468400 (cit. on p. 7).

[46] Nasim Rahaman, Aristide Baratin, Devansh Arpit, Felix Draxler, Min Lin, Fred A. Hamprecht, Yoshua Bengio, and Aaron Courville. *On the Spectral Bias of Neural Networks*. 2019. arXiv: 1806.08734 [stat.ML] (cit. on p. 8).

[47] Vincent Sitzmann, Justus Thies, Felix Heide, Matthias Nießner, Gordon Wetzstein, and Michael Zollhöfer. *DeepVoxels: Learning Persistent 3D Feature Embeddings*. 2019. arXiv: 1812.01024 [cs.CV] (cit. on p. 10).

[48] Ben Mildenhall, Pratul P. Srinivasan, Rodrigo Ortiz-Cayon, Nima Khademi Kalantari, Ravi Ramamoorthi, Ren Ng, and Abhishek Kar. *Local Light Field Fusion: Practical View Synthesis with Prescriptive Sampling Guidelines*. 2019. arXiv: 1905.00889 [cs.CV] (cit. on pp. 10, 11).

[49] Stephen Lombardi, Tomas Simon, Jason Saragih, Gabriel Schwartz, Andreas Lehrmann, and Yaser Sheikh. «Neural volumes: learning dynamic renderable volumes from images». In: *ACM Transactions on Graphics* 38.4 (July 2019), pp. 1–14. ISSN: 1557-7368. DOI: 10.1145/3306346.3323020. URL: http://dx.doi.org/10.1145/3306346.3323020 (cit. on p. 10).

[50] Vincent Sitzmann, Michael Zollhöfer, and Gordon Wetzstein. *Scene Representation Networks: Continuous 3D-Structure-Aware Neural Scene Representations*. 2020. arXiv: 1906.01618 [cs.CV] (cit. on p. 10).

[51] Thierry Bouwmans. «Traditional and recent approaches in background modeling for foreground detection: An overview». In: *Computer Science Review* 11-12 (2014), pp. 31–66. ISSN: 1574-0137. DOI: https://doi.org/10.1016/j.cosrev.2014.04.001. URL: https://www.sciencedirect.com/science/article/pii/S1574013714000033 (cit. on p. 13).

[52] Jana Mattheus, Hans Grobler, and Adnan M. Abu-Mahfouz. «A Review of Motion Segmentation: Approaches and Major Challenges». In: *2020 2nd International Multidisciplinary Information Technology and Engineering Conference (IMITEC)*. 2020, pp. 1–8. DOI: 10.1109/IMITEC50163.2020.9334076 (cit. on p. 13).

[53] Pia Bideau and Erik Learned-Miller. *It's Moving! A Probabilistic Model for Causal Motion Segmentation in Moving Camera Videos*. 2016. arXiv: 1604.00136 [cs.CV] (cit. on p. 13).

[54] Thomas Brox and Jitendra Malik. «Object Segmentation by Long Term Analysis of Point Trajectories». In: *Computer Vision – ECCV 2010*. Ed. by Kostas Daniilidis, Petros Maragos, and Nikos Paragios. Berlin, Heidelberg: Springer Berlin Heidelberg, 2010, pp. 282–295. ISBN: 978-3-642-15555-0 (cit. on p. 13).

[55] Charig Yang, Hala Lamdouar, Erika Lu, Andrew Zisserman, and Weidi Xie. *Self-supervised Video Object Segmentation by Motion Grouping*. 2021. arXiv: 2104.07658 [cs.CV] (cit. on p. 14).

[56] Ricardo Martin-Brualla, Noha Radwan, Mehdi S. M. Sajjadi, Jonathan T. Barron, Alexey Dosovitskiy, and Daniel Duckworth. *NeRF in the Wild: Neural Radiance Fields for Unconstrained Photo Collections*. 2021. arXiv: 2008.02268 [cs.CV] (cit. on pp. 14, 17, 18).

[57] Karl Stelzner, Kristian Kersting, and Adam R. Kosiorek. *Decomposing 3D Scenes into Objects via Unsupervised Volume Segmentation*. 2021. arXiv: 2104.01148 [cs.CV] (cit. on p. 14).

[58] Zhengqi Li, Simon Niklaus, Noah Snavely, and Oliver Wang. *Neural Scene Flow Fields for Space-Time View Synthesis of Dynamic Scenes*. 2021. arXiv: 2011.13084 [cs.CV] (cit. on p. 14).

[59] Edgar Tretschk, Ayush Tewari, Vladislav Golyanik, Michael Zollhöfer, Christoph Lassner, and Christian Theobalt. *Non-Rigid Neural Radiance Fields: Reconstruction and Novel View Synthesis of a Dynamic Scene From Monocular Video*. 2021. arXiv: 2012.12247 [cs.CV] (cit. on p. 14).

[60] Colmap 2024. URL: https://colmap.github.io/index.html (cit. on pp. 14, 17).

[61] Wolf and Dewitt. 2000; McGlone, 2004.

[62] Mathworks 2023. URL: https://www.mathworks.com/help/vision/ug/structure-from-motion.html#:~:text=Structure%20from%20motion%20(SfM)%20is,localization%20and%20mapping%20(vSLAM)..

[63] OpenCV 2024. URL: https://docs.opencv.org/4.x/df/d54/tutorial_py_features_meaning.html.

[64] David G. Lowe. «Distinctive Image Features from Scale-Invariant Keypoints». In: *International Journal of Computer Vision* 60.2 (2004), pp. 91–110. DOI: 10.1023/B:VISI.0000029664.99615.94. URL: http://www.cs.ubc.ca/~lowe/papers/ijcv04.pdf.

[65] Herbert Bay, Andreas Ess, Tinne Tuytelaars, and Luc Van Gool. «Speeded-Up Robust Features (SURF)». In: *Computer Vision and Image Understanding* 110.3 (2008). Similarity Matching in Computer Vision and Multimedia, pp. 346–359. ISSN: 1077-3142. DOI: https://doi.org/10.1016/j.cviu.2007.09.014. URL: https://www.sciencedirect.com/science/article/pii/S1077314207001555.

[66] Michael Calonder, Vincent Lepetit, Christoph Strecha, and Pascal Fua. «BRIEF: Binary Robust Independent Elementary Features». In: *Computer Vision – ECCV 2010*. Ed. by Kostas Daniilidis, Petros Maragos, and Nikos Paragios. Berlin, Heidelberg: Springer Berlin Heidelberg, 2010, pp. 778–792. ISBN: 978-3-642-15561-1.

[67] Ethan Rublee, Vincent Rabaud, Kurt Konolige, and Gary Bradski. «ORB: An efficient alternative to SIFT or SURF». In: *2011 International Conference on Computer Vision*. 2011, pp. 2564–2571. DOI: 10.1109/ICCV.2011.6126544.

[68] Andrew W. Fitzgibbon, Simon W. Bowman, Andrew Zisserman, and Mark A. Fischler. «Bundle Adjustment – A Modern Synthesis». In: *Vision Algorithms: Theory and Practice*. Vol. 1883. LNCS. Springer-Verlag, 2000, pp. 298–372.

[69] Johannes Lutz Schönberger and Jan-Michael Frahm. «Structure-from-Motion Revisited». In: *Conference on Computer Vision and Pattern Recognition (CVPR)*. 2016.

[70] Johannes Lutz Schönberger, Enliang Zheng, Marc Pollefeys, and Jan-Michael Frahm. «Pixelwise View Selection for Unstructured Multi-View Stereo». In: *European Conference on Computer Vision (ECCV)*. 2016.

[71] René Ranftl, Alexey Bochkovskiy, and Vladlen Koltun. «Vision Transformers for Dense Prediction». In: *ICCV* (2021).

[72] René Ranftl, Katrin Lasinger, David Hafner, Konrad Schindler, and Vladlen Koltun. «Towards Robust Monocular Depth Estimation: Mixing Datasets for Zero-Shot Cross-Dataset Transfer». In: *IEEE Transactions on Pattern Analysis and Machine Intelligence* 44.3 (2022).

[73] Youngjung Kim, Hyungjoo Jung, Dongbo Min, and Kwanghoon Sohn. «Deep Monocular Depth Estimation via Integration of Global and Local Predictions». In: *IEEE Transactions on Image Processing* 27.8 (2018), pp. 4131–4144. DOI: 10.1109/TIP.2018.2836318.

[74] Zhengqi Li and Noah Snavely. *MegaDepth: Learning Single-View Depth Prediction from Internet Photos*. 2018. arXiv: 1804.00607 [cs.CV].

[75] Ke Xian, Chunhua Shen, Zhiguo Cao, Hao Lu, Yang Xiao, Ruibo Li, and Zhenbo Luo. «Monocular Relative Depth Perception with Web Stereo Data Supervision». In: *2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 2018, pp. 311–320. DOI: 10.1109/CVPR.2018.00040.

[76] Chaoyang Wang, Simon Lucey, Federico Perazzi, and Oliver Wang. *Web Stereo Video Supervision for Depth Prediction from Dynamic Scenes*. 2019. arXiv: 1904.11112 [cs.CV].

[77] Weifeng Chen, Zhao Fu, Dawei Yang, and Jia Deng. *Single-Image Depth Perception in the Wild*. 2017. arXiv: 1604.03901 [cs.CV].

[78] Thomas Schöps, Johannes L. Schönberger, Silvano Galliani, Torsten Sattler, Konrad Schindler, Marc Pollefeys, and Andreas Geiger. «A Multi-view Stereo Benchmark with High-Resolution Images and Multi-camera Videos». In: *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. 2017, pp. 2538–2547. DOI: 10.1109/CVPR.2017.272.

[79] Daniel J. Butler, Jonas Wulff, Garrett B. Stanley, and Michael J. Black. «A Naturalistic Open Source Movie for Optical Flow Evaluation». In: *Computer Vision – ECCV 2012*. Ed. by Andrew Fitzgibbon, Svetlana Lazebnik, Pietro Perona, Yoichi Sato, and Cordelia Schmid. Berlin, Heidelberg: Springer Berlin Heidelberg, 2012, pp. 611–625. ISBN: 978-3-642-33783-3.

[80] Andreas Geiger, Philip Lenz, and Raquel Urtasun. «Are we ready for autonomous driving? The KITTI vision benchmark suite». In: *2012 IEEE Conference on Computer Vision and Pattern Recognition*. 2012, pp. 3354–3361. DOI: 10.1109/CVPR.2012.6248074.

[81] Nathan Silberman, Derek Hoiem, Pushmeet Kohli, and Rob Fergus. «Indoor Segmentation and Support Inference from RGBD Images». In: *Computer Vision – ECCV 2012*. Ed. by Andrew Fitzgibbon, Svetlana Lazebnik, Pietro Perona, Yoichi Sato, and Cordelia Schmid. Berlin, Heidelberg: Springer Berlin Heidelberg, 2012, pp. 746–760. ISBN: 978-3-642-33715-4.

[82] Jürgen Sturm, Nikolas Engelhard, Felix Endres, Wolfram Burgard, and Daniel Cremers. «A benchmark for the evaluation of RGB-D SLAM systems». In: *2012 IEEE/RSJ International Conference on Intelligent Robots and Systems*. 2012, pp. 573–580. DOI: 10.1109/IROS.2012.6385773.