### [Training Neural Radiance Field (NeRF) Models with Keras/TensorFlow and DeepVision (stackabuse.com)](https://stackabuse.com/training-a-neural-radiance-field-nerf-model-with-keras-tensorflow-and-deepvision/)

### **Rendering**

Rendering is the process of creating an image from a 3D model. The model will contain features such as textures, shading, shadows, lighting, and viewpoints, and the role of the rendering engine is to process these features to create a realistic image.

Three common types of rendering algorithms are rasterization, which projects objects geometrically based on information in the model, without optical effects; ray casting, which calculates an image from a specific point of view using basic optical laws of reflection; and ray tracing, which uses Monte Carlo techniques to achieve a realistic image in a far shorter time

### **Volume Rendering**

Volume rendering enables you to create a 2D projection of a 3D discretely sampled dataset.

a volume rendering algorithm obtains the RGBα (Red, Green, Blue, and Alpha channel) for every voxels in the space through which rays from the camera are casted.

### **View Synthesis**

View synthesis is the opposite of volume rendering—it involves creating a 3D view from a series of 2D images. This can be done using a series of photos that show an object from multiple angles, create a hemispheric plan of the object, and place each image in the appropriate place around the object. A view synthesis function attempts to predict the depth given a series of images that describe different perspectives of an object.

NeRFs are used for novel view synthesis - creating new views of objects and images, given some views. In effect, you can think of novel view synthesis as 2D->3D conversion, and many approaches to solve this problem exist, some more successful than others.

The **biggest difference between a NeRF model and traditional neural networks** for 3D reconstruction is that NeRF is an instance-specific implicit representation of an object.

A NeRF network is trained to map directly from viewing direction and spatial location (5D input) to opacity and color (4D output), using volume rendering to render new views.

What is NeRF?

NeRF is a technique used in computer vision that allows us to reconstruct 3D scenes from 2D images. It uses deep [neural networks](https://saturncloud.io/glossary/neural-networks) to estimate the underlying 3D structure and appearance of a scene.

NeRF works by representing the scene as a continuous function called a radiance field. This function maps a 3D point in space to a color and opacity value.

## How Does NeRF Work?

## Our algorithm represents a scene using a fully-connected (non-convolutional) deep network, whose input is a single continuous 5D coordinate (spatial location (x,y,z) and viewing direction (θ,ϕ)) and whose output is the volume density and view-dependent emitted radiance at that spatial location. We synthesize views by querying 5D coordinates along camera rays and use classic volume rendering techniques to project the output colors and densities into an image.

## 

## NeRF works by training a neural network to approximate the radiance field of a scene. This is done by training the network to predict the color and opacity of each point in space. The network takes as input the 3D coordinates of a point and the viewing direction, and outputs the corresponding color and opacity values.

## https://github.com/bmild/nerf/raw/master/imgs/pipeline.jpg

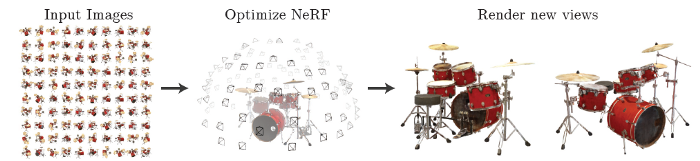
## To train the network, we need a dataset of images of the scene from different viewpoints. During training, the network is trained to minimize the difference between the predicted radiance values and the ground truth values from the dataset.

## How Neural Radiance Fields Work

A NeRF uses a sparse set of input views to optimize a continuous volumetric scene function. The result of this optimization is the ability to produce novel views of a complex scene. You can provide input for NeRF as a static set of images.

A continuous scene is a 5D vector-valued function with the following characteristics:

* Its input is a 3D location x = (x; y; z) and 2D viewing direction (θ; Φ)
* Its output is an emitted color c = (r; g; b) and volume density (α).



Here is how you can generate a NeRF from a specific viewpoint:

1. **Generate a sampled set of 3D points**—by marching camera rays through the scene.
2. **Produce an output set of densities and colors**—by inputting your sampled points with their corresponding 2D viewing directions into the neural network.
3. **Accumulate your densities and colors into a 2D image**—by using classical volume rendering techniques.

# **Starting with a Basic MLP**

## https://miro.medium.com/v2/resize:fit:700/1*cKChqiNKm0QTySQs2go3fg.png

## Figure 2. NeRF Training Overview. Image retrieved from the original NeRF paper by [Mildenhall](https://arxiv.org/abs/2003.08934" \t "_blank) et al.

## the input of this fully connected network is a single 5D coordinate (3 for location and 2 for viewing direction), and the output is the density and colour of the given location. In practice, density only matters with the location and not the viewing direction, and so only location is used to to predict the density, while viewing direction is combined with the location features to predict the colour seen.

# Optimising NeRF:

There are two implementation techniques to better improve NeRF in better representing complex scene — **Positional encoding and hierarchical volume sampling.**

## Improving NeRF Performance

The original NeRF model had several drawbacks—it was slow to train and render, only able to handle static scenes. It is also inflexible, because a NeRF model trained on one scene cannot be used for other scenes.

### **RegNeRF**

[RegNeRF (CVPR 2022)](https://arxiv.org/pdf/2112.00724.pdf) stands for regularizing neural radiance fields (RegNeRF) for view synthesis from sparse inputs. It helps address a problem in the performance of neural radiance fields (NeRF), when the number of input views is low.

## ****pixelNeRF****

The [pixelNeRF (CPVR 2021)](https://arxiv.org/pdf/2012.02190.pdf" \t "_blank) learning framework can predict a continuous neural scene representation based on one or several input images. Constructing NeRFs typically requires optimizing the representation of each scene independently, which involves many calibrated views and significant compute time.

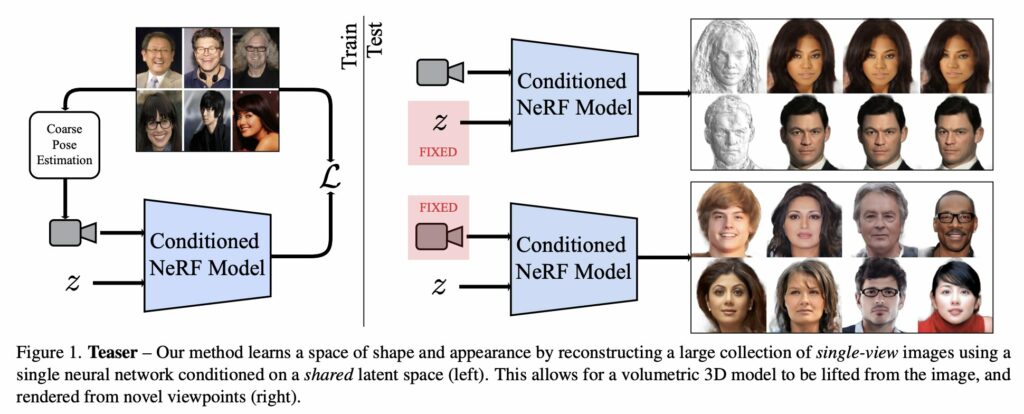
## ****Mega-NeRF****

[Mega-NeRF (CVPR 2022)](https://arxiv.org/pdf/2112.10703v2.pdf) is a learning framework that uses NeRFs to build interactive 3D environments from large-scale visual captures, such as buildings and multiple city blocks collected mainly by drones. Traditionally, NeRFs are evaluated using single-object scenes. However, this method poses several challenges.

Traditional NeRFs require modeling thousands of images with diverse lighting conditions, each capturing only a small part of a scene and making it difficult to achieve the fast rendering that enables interactive fly-throughs. Additionally, these prohibitively large models require capacities that make it infeasible to train on one GPU.

## ****LOLNeRF****

[LOLNeRF (CVPR 2022)](https://arxiv.org/pdf/2111.09996.pdf) stands for Learn from One Look (LOL). This learning method uses NeRF for generative 3D modeling, training only from data with primarily single views of each object. It helps produce the corresponding 3D structure of objects in a way they can be rendered from different views.



## ****Neural Sparse Voxel Fields (NSVF)****

[NSVF (NeurIPS 2020)](https://arxiv.org/abs/2007.11571) is a neural scene representation that enables fast, high-quality rendering that is not dependent on a specific viewpoint. It works by defining voxel-bounded implicit fields organized in a sparse network of cells, and progressively learns voxel structures in each cell of the network. It can render new views much faster by skipping voxels with no scene content—this technique makes NSVF over ten times faster than the original NeRF.

## ****Mip-NeRF****

[Mip-NerF](https://arxiv.org/abs/2103.13415) (ICCV 2021) extends the original NeRF model with the objective of reducing blurring effects and visual artifacts. NeRF used a single ray per pixel, which often caused blurring or aliasing at different resolutions. Mip-NeRF uses a geometrical shape known as a conical frustum to render each pixel, instead of a ray, which reduces aliasing, makes it possible to show fine details in an image, and reduces error rates by between 17-60%. The model is also 7% faster than NeRF.

## ****KiloNeRF****

[KiloNeRF](https://arxiv.org/abs/2103.13744) (2021) addresses the problem of slow rendering in NeRF, which is mainly related to the need to query a deep MLP network millions of times. KiloNeRF separates the workload among thousands of small MLPs, instead of one large MLP which needs to be queried many times. Each small MLP represents part of a scene, enabling 3X performance improvement with lower storage requirements, and comparable visual quality.

## ****Plenoxels****

[Plenoptic voxels (2021)](https://arxiv.org/abs/2112.05131) (Plenoxels) replace the MLP in the center of NeRF with a sparse 3D grid. Each query point is interpolated from its surrounding voxels. New 2D views are hence rendered without running a neural network, which greatly reduces complexity and computational requirements. Plenoxels provide a similar visual quality to NeRF while being two orders of magnitude faster.

## Follow up Works

## Positional Encoding

Fully-connected deep networks are biased to learn low frequencies faster. Surprisingly, applying a simple mapping to the network input is able to mitigate this issue. We explore these input mappings in a follow up work.

## Multiscale Representation

By efficiently rendering anti-aliased conical frustums instead of rays, our followup, mip-NeRF, reduces objectionable aliasing artifacts and significantly improves NeRF's ability to represent fine details, while also being 7% faster than NeRF and half the size.

## https://uploads-ssl.webflow.com/51e0d73d83d06baa7a00000f/6086e97ec8800aae5cc3c7c0_rays.jpg

## Learned Initializations

In a followup work we explore how meta-learning can be applied to speed up convergence and embed dataset specific priors.

## Relighting

We extend NeRF to enable the rendering of scenes from novel viewpoints under arbitrary lighting conditions.