Literature review on current state of implicit neural scene representations

Yash Savani,
Computer Science Department, Carnegie Mellon University,
ysavani@cs.cmu.edu

Introduction

Since it's release in 2020, the NeRF model (Mildenhall et al., 2020) has created a stir in the Machine Learning and Graphics communities. The paper showed that an implicit model in the form of an MLP could be used to convincingly represent complex 3-dimensional scenes as continuous vector fields. We examine some of the preliminary and contemporary work in the neural scene representation space and suggest potential future research directions to explore.

Many of the papers listed here were suggested in the fantastic lit review by Dellaert & Yen-Chen (2021) and Lin (2021a) in their Github collection.

NeRF (Mildenhall et al., 2020)

In this section we will go through the NeRF paper in some detail as it the foundation on which we consider the other work.

The authors introduce a method for implicitly representing a scene in a multi-layer perceptron (MLP). We can represent the MLP as a non-linear function F_Θ whose domain is $\mathbb{R}^5$ (3 dimensions for the spatial location $\mathbf{x} = (x, y, z)$ and two dimensions for the viewing angle $\mathbf{d} = (\theta, \phi)$) and whose co-domain is $\mathbb{R}^4$ (3 dimensions for color $\mathbf{c} = (r, g, b)$ and 1 dimension for the volume density or opacity σ) $F_\Theta : (\mathbf{x}, \mathbf{d}) \rightarrow (\mathbf{c}, \sigma)$.

Because volume rendering is differentiable, the only input required is a set of images with known camera poses. Training is carried out by using standard backpropagation with gradient descent to minimize the reconstruction loss between the predicted image from the camera and the actual image. Using multiple camera locations and viewing angles, the network is encouraged to converge to a consistent representation of the scene.

Since σ is not a function of the viewing angle, the MLP first processes $\mathbf{x}$ using

an 8 layer fully connected network with 1 skip connection at layer 5, where every layer is 256 dimensional, with ReLU activations to predict σ, rectified by another ReLU, and a 256 dimensional auxiliary feature vector. $\mathbf{c}$ is predicted using a concatenation of $\mathbf{d}$ and the 256 dimensional feature vector by passing it through a single 128 dimension fully-connected layer with ReLU activations and a final sigmoid layer.

The steps to generate the NeRF image are:

1. Extend rays from the camera through the scene and generate a sampled set of 3D points along the ray.
2. Use the points with the viewing angle as input to the MLP to get the colors and volume densities.
3. Integrate over the colors and volume densities to get the final color for the pixel represented for each ray.

Let $\sigma(\mathbf{x})$ be the differential probability of a ray terminating at location $\mathbf{x}$, and let $C(\mathbf{r})$ be the expected color of the camera ray $\mathbf{r}(t) = \mathbf{o} + t\mathbf{d}$, where t goes from t_n to t_f.

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

where $T(t) = \exp\left(-\int_{t_n}^{t} \sigma(\mathbf{r}(s))ds\right)$ is the accumulated transmittance along the ray i.e. the probability that the ray travels from t_n to t without hitting any other particle. To estimate $C(\mathbf{r})$, the authors use numerical quadrature with a stratified sampling approach that selects a point to sample uniformly at random from each one of N evenly-spaced bins. The quadrature rule used is:

$$\begin{aligned}
\hat{C}(\mathbf{r}) &= \sum_{i=[N]} T_i(1 - \exp(-\sigma_i\delta_i))\mathbf{c}_i \\
T_i &= \exp\left(-\sum_{j=[i-1]} \sigma_j\delta_j\right) \\
\delta_i &= t_{i+1} - t_i
\end{aligned}$$

Using a basic implementation of this method does not converge to a sufficiently high-resolution representation and is inefficient. Instead, the authors transform the 5D input using a positional encoding so the MLP can represent higher frequency functions. Also, a hierarchical sampling procedure is used to reduce the number of queries required for this high-frequency scene representation.

Neural networks are biased towards learning lower frequency functions so to capture the higher frequency details the authors map the inputs into a higher dimensional space using high frequency functions before passing them to the network. The mapping is $\gamma : \mathbb{R} \to \mathbb{R}^{2L}$, where

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

γ is applied separately to every coordinate in $\mathbf{x}$ and $\mathbf{d}$ ($\mathbf{x}$ and $\mathbf{d}$ are normalized so every element lies in $[-1, 1]$). The authors used $L = 10$ for $\gamma(\mathbf{x})$ and $L = 4$ for $\gamma(\mathbf{d})$.

Since many of the sampled points may lie in free space or occluded regions, the authors propose a more efficient hierarchical sampling technique that allocates samples proportional to the expected effect on the final rendering. They use a **coarse** and **fine** network. For the coarse method, they use the aforementioned stratified sampling technique for N_c locations. With the results from the coarse network, the fine network is trained based on

$$\hat{C}_c(\mathbf{r}) = \sum_{i \in [N_c]} w_i c_i$$
$$w_i = T_i(1 - \exp(-\sigma_i \delta_i)).$$

The normalized the weights $\hat{\mathbf{w}} = \mathbf{w}/||\mathbf{w}||$ produce a pdf along the ray. Sampling N_f locations from this distribution using inverse transform sampling, and evaluating the fine network at the union of both samples gives final rendered color.

The input data for an scene are the RGB images of the scene, the corresponding camera poses and the intrinsic parameters, and the scene bounds. The authors use the COLMAP package to estimate the parameters from real data. For each iteration a random batch of rays are sampled as a batch of pixels from the dataset. The hierarchical sampling is then carried out on the batch of rays. Let $\mathcal{R}$ be the set of rays in each batch. Then

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

The authors used a batch size of 4096 rays, each sampled with $N_c = 64$ and $N_f = 128$. They use an Adam optimizer with learning rate that begins at 5e-4 and decays exponentially to 5e-5 over the course of the optimization. The rest of the hyperparameters were left at default values. Convergence took around 100-300k iterations on a single NVIDIA V100 GPU.

One of the datasets they used for this task is the DeepVoxels dataset that is a synthetic set of 3D objects. Another dataset is a set of 8 scenes captured with a handheld cellphone, captured with 20 to 62 images and $\frac{1}{8}$ held out as a test set.

Pre-NeRF

There was a wealth of research in neural implicit surfaces that led up to the NeRF paper that we examine in some detail here.

- Occupancy Networks (Mescheder, Oechsle, Niemeyer, Nowozin, & Geiger, 2019)
  - Introduced the idea of an implicit model that learned a decision boundary that could be used to classify whether points were in the object or not.
  - The model was trained on a dataset of 3D scenes unlike NeRF, which is trained on images of 3D scenes.
  - The authors also train a VAE to learn a latent space for the 3D shapes that can be used to interpolate between the objects.
- Im-Net (Chen & Zhang, 2019)
  - Another work that trains an implicit network as a binary classifier that takes a feature vector as input along with a point coordinate and predicts whether the point is in the shape.
  - The feature vector is extracted by a shape encoder to represent the shape and can be interpolated to get intermediary shapes.
  - This model is also trained on 3D data.
- DeepSDF (J. J. Park, Florence, Straub, Newcombe, & Lovegrove, 2019)
  - This is the true parent of NeRF. The method trains a deep model to learn a deep signed distance function that takes a point as input and returns a scalar that represents the distance of the point from the boundary of the shape.
  - They also have a latent code to represent the shape.
  - This model is also trained on 3D data.
- PIFu (Saito et al., 2019)
  - Learns a 3D implicit model for human forms from images.
  - Uses a fully convolutional image encoder to get spatial features for every 3D point that can be used to see if a point is inside the surface.
  - It can also be used to learn RGB color information to perform texture painting.
  - They use a reconstruction loss and compare to a 3D human surfaces.

Beyond these initial forays into neural implicit surfaces that were still trained with 3D data, there was some work on generalizing to training from 2D images as well as work that built on top of the implicit functions to make them more efficient and more accurate.

- CvxNet (Deng, Genova, et al., 2020)
  - The authors proposed a technique to learn a convex decomposition of topology.
  - Any topology can be modeled using a point-wise maximum of a union of convex meshes.
  - The convex topologies can then be used to model physics since many

rigid body physics simulations require convex meshes.
    - The encoder network predicts the parameters for a convex polytope that can then be used to train the model.
    - The model is still trained on 3D data using a reconstruction loss along with other auxiliary losses to improve convergence.
- BSP-NET (Chen, Tagliasacchi, & Zhang, 2020)
    - Very similar to CvxNet in that it predicts a series of convex shapes to represent the object.
    - In contrast they use a binary space partition tree to create the convexes that ends up being much more efficient.
- Deep Local Shapes (Chabra et al., 2020)
    - Stores the DeepSDF weights in a voxel grid so it can represent large scenes more efficiently.
- Scene Representation Networks (Sitzmann, Zollhoefer, & Wetzstein, 2019)
    - Differentiable ray marching that can be used to learn from images.
- Differentiable Volumetric Rendering (Niemeyer, Mescheder, Oechsle, & Geiger, 2020)
    - Uses a differentiable renderer so it can be trained from images.
    - They do not integrate over volume like NeRF though.
- Implicit Differentiable Renderer (Yariv, Atzmon, & Lipman, 2020)
    - Similar to the above two, but the model proposed can be used to refine the camera pose during training. Essentially they are able to learn the camera details from the images in a differentiable way.
- Neural Articulated Shape Approximation (Deng, Lewis, et al., 2020)
    - Learns implicit models for deformable bodies.
    - Takes as input coordinate frames (bones) and learns an occupancy function that is a mixture of rigid compositions and a pose layer that allows for non-rigid deformations.
    - They get the deformations by convolving with a gaussian kernel

A contemporary of NeRF was Neural Volume Rendering (Lombardi et al., 2019) that was very similar to NeRF but still learned a 3D volume of density and color as a voxel-based representation.

Post-NeRF

After NeRF was able to create such spectacularly convincing results, there was a mass of excitement in the field. Since then several new papers have been released extending, improving and analyzing NeRF. We consider some of them in this section.

Analysis

- Fourier Features (Tancik et al., 2020)

Performance

- Neural sparse voxel fields (Liu, Gu, Lin, Chua, & Theobalt, 2020)
- NeRF++: Analyzing and Improving Neural Radiance Fields (Zhang, Riegler, Snavely, & Koltun, 2020)
- Derf: Decomposed radiance fields (Rebain et al., 2021)
- Autoint: Automatic integration for fast neural volume rendering (Lindell, Martel, & Wetzstein, 2021)
- Learned initializations for optimizing coordinate-based neural representations (Tancik et al., 2021)
- JaxNeRF ("Google-research/jaxnerf at master · google-research/google-research," n.d.)
- NeRF-pytorch (Lin, 2021b)

Dynamic Scenes

- Deformable neural radiance fields (K. Park et al., 2020)

Other Relevant Papers

- Structure-from-motion revisited (**schonberger__structure?**–motion_2016)
- Nerf in the wild: Neural radiance fields for unconstrained photo collections (Martin-Brualla et al., 2021)
- Self-Calibrating Neural Radiance Fields (Jeong et al., 2021)
- CodeNeRF: Disentangled Neural Radiance Fields for Object Categories (Jang & Agapito, 2021)
- SIMONe: View-Invariant, Temporally-Abstracted Object Representations via Unsupervised Video Decomposition (Kabra et al., 2021)
- NeRF-VAE: A Geometry Aware 3D Scene Generative Model (Kosiorek et al., 2021)
- Animatable Neural Radiance Fields for Human Body Modeling (Peng et al., 2021)

Potential Directions

While the field of implicit neural scene representations is relatively new, as you can see from the long and yet incomplete list of references there has already been a lot of work in this area. The prospect of attaining allocentric continuous 3D representations of the world through egocentric 2D images is tantalizing and offers several interesting directions. There are still several references that I have not had time to go through, and as a result it is possible that several of the ideas in this section may already have been explored. However, based on the literature review of the references I have provided above here are some suggestions for potential directions.

Exploiting NeRF-techniques for simulation and control.

NeRF is a fascinating tool for rendering the real world. However, most of the work has been focused on capturing the real world exclusively through images or video. One project idea is to look at potential ways to capture dynamics of the world through images and interaction. Rather than just represent the world as it is, we may be able to represent the world along with dynamics of how the world might look if interacted with. We could run these experiments in 3D interaction simulators like Unity Engine.

Uncertainty NeRF

As of right now, all the NeRF models still aren't perfect. It would be extremely useful to have a NeRF model that could generate uncertainty estimates for it's volume density and color predictions.

End-to-End Differentiable Pose Estimation

While (Jeong et al., 2021) has been able to have an end-to-end model that is able to learn intrinsic, extrinsic, and distortion paramaters from images of the scene, COLMAP (**schonberger_structure?**–motion_2016) is still used to find the poses of the camera. Potentially, we could learn the camera poses using a differentiable method. This would make the entire NeRF generation process differentiable and end-to-end learnable. We could also try to leverage any other meta-data we may have like accelerometer kinematic information to get better pose estimation.

NeRF meta-learning

(Tancik et al., 2021) have shown that using meta-learning initializations like MAML and Reptile can have an enormous impact on the convergence rate of NeRF. It is possible that other meta-learning techniques such as finding latent structure in the weights of the MLP or learning end-to-end trainable functions that are capable of adjusting the weights of the MLP for novel scenes might lead to even more significant results. Furthermore, it may be possible to learn homeomorphisms between different objects. It might be interesting to study the algebraic topology of an object through the lens of the weights of the network.

Using auxiliary architectures with NeRF

While NeRF does a great job of capturing the 3-dimensional richness of a scene, the architecture used to train the model is very simple. It would be interesting to see what would happen if we were to use an auxiliary architecture like a Transformer or a CNN along with the MLP model to make the model more performant.

Exploring some of the Theoretical aspects of NeRF further

(Tancik et al., 2020) has already done some theoretical work to show how the NeRF NLP acts as an NTK and how using randomized Fourier Features can improve the NTK convergence for the high-frequency components. We could extend some of this work and see if there is a way we may be able use techniques such as sketching to improve the convergence rate even more.

References

Chabra, R., Lenssen, J. E., Ilg, E., Schmidt, T., Straub, J., Lovegrove, S., & Newcombe, R. (2020). Deep local shapes: Learning local sdf priors for detailed 3d reconstruction (pp. 608–625). Springer.

Chen, Z., Tagliasacchi, A., & Zhang, H. (2020). Bsp-net: Generating compact meshes via binary space partitioning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition* (pp. 45–54).

Chen, Z., & Zhang, H. (2019). Learning implicit fields for generative shape modeling. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition* (pp. 5939–5948).

Dellaert, F., & Yen-Chen, L. (2021). Neural Volume Rendering: NeRF And Beyond. *arXiv:2101.05204 [Cs]*. Retrieved from http://arxiv.org/abs/2101.05204

Deng, B., Genova, K., Yazdani, S., Bouaziz, S., Hinton, G., & Tagliasacchi, A. (2020). Cvxnet: Learnable convex decomposition. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition* (pp. 31–44).

Deng, B., Lewis, J. P., Jeruzalski, T., Pons-Moll, G., Hinton, G., Norouzi, M., & Tagliasacchi, A. (2020). NASA neural articulated shape approximation. In *Computer Vision–ECCV 2020: 16th European Conference, Glasgow, UK, August 23–28, 2020, Proceedings, Part VII 16* (pp. 612–628). Springer.

Google-research/jaxnerf at master · google-research/google-research. (n.d.). *GitHub*. Retrieved from https://github.com/google-research/google-research

Jang, W., & Agapito, L. (2021). CodeNeRF: Disentangled Neural Radiance Fields for Object Categories. *arXiv:2109.01750 [Cs]*. Retrieved from http://arxiv.org/abs/2109.01750

Jeong, Y., Ahn, S., Choy, C., Anandkumar, A., Cho, M., & Park, J. (2021). Self-Calibrating Neural Radiance Fields. *arXiv Preprint arXiv:2108.13826*.

Kabra, R., Zoran, D., Erdogan, G., Matthey, L., Creswell, A., Botvinick, M., … Burgess, C. P. (2021). SIMONe: View-Invariant, Temporally-Abstracted

Object Representations via Unsupervised Video Decomposition. *arXiv Preprint arXiv:2106.03849*.

Kosiorek, A. R., Strathmann, H., Zoran, D., Moreno, P., Schneider, R., Mokrá, S., & Rezende, D. J. (2021). NeRF-VAE: A Geometry Aware 3D Scene Generative Model. *arXiv:2104.00587 [Cs, Stat]*. Retrieved from http://arxiv.org/abs/2104.00587

Lin, Y.-C. (2021a, September). Awesome Neural Radiance Fields. Retrieved from https://github.com/yenchenlin/awesome-NeRF

Lin, Y.-C. (2021b, September). NeRF-pytorch. Retrieved from https://github.com/yenchenlin/nerf-pytorch

Lindell, D. B., Martel, J. N., & Wetzstein, G. (2021). Autoint: Automatic integration for fast neural volume rendering. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition* (pp. 14556–14565).

Liu, L., Gu, J., Lin, K. Z., Chua, T.-S., & Theobalt, C. (2020). Neural sparse voxel fields. *arXiv Preprint arXiv:2007.11571*.

Martin-Brualla, R., Radwan, N., Sajjadi, M. S., Barron, J. T., Dosovitskiy, A., & Duckworth, D. (2021). Nerf in the wild: Neural radiance fields for unconstrained photo collections. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition* (pp. 7210–7219).

Mescheder, L., Oechsle, M., Niemeyer, M., Nowozin, S., & Geiger, A. (2019). Occupancy networks: Learning 3d reconstruction in function space. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition* (pp. 4460–4470).

Mildenhall, B., Srinivasan, P. P., Tancik, M., Barron, J. T., Ramamoorthi, R., & Ng, R. (2020). Nerf: Representing scenes as neural radiance fields for view synthesis. In *European conference on computer vision* (pp. 405–421). Springer.

Niemeyer, M., Mescheder, L., Oechsle, M., & Geiger, A. (2020). Differentiable volumetric rendering: Learning implicit 3d representations without 3d supervision. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition* (pp. 3504–3515).

Park, J. J., Florence, P., Straub, J., Newcombe, R., & Lovegrove, S. (2019). Deepsdf: Learning continuous signed distance functions for shape representation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition* (pp. 165–174).

Park, K., Sinha, U., Barron, J. T., Bouaziz, S., Goldman, D. B., Seitz, S. M., & Martin-Brualla, R. (2020). Deformable neural radiance fields. *arXiv Preprint arXiv:2011.12948*.

Peng, S., Dong, J., Wang, Q., Zhang, S., Shuai, Q., Bao, H., & Zhou, X. (2021). Animatable Neural Radiance Fields for Human Body Modeling. *arXiv:2105.02872 [Cs]*. Retrieved from http://arxiv.org/abs/2105.02872

Rebain, D., Jiang, W., Yazdani, S., Li, K., Yi, K. M., & Tagliasacchi, A. (2021). Derf: Decomposed radiance fields. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition* (pp. 14153–14161).

Saito, S., Huang, Z., Natsume, R., Morishima, S., Kanazawa, A., & Li, H. (2019). Pifu: Pixel-aligned implicit function for high-resolution clothed human digitization. In *Proceedings of the IEEE/CVF International Conference on Computer Vision* (pp. 2304–2314).

Sitzmann, V., Zollhoefer, M., & Wetzstein, G. (2019). Scene Representation Networks: Continuous 3D-Structure-Aware Neural Scene Representations. In *Advances in Neural Information Processing Systems* (Vol. 32). Curran Associates, Inc. Retrieved from https://papers.nips.cc/paper/2019/hash/b5dc4e5d9b495d0196f61d45b26ef33e-Abstract.html

Tancik, M., Mildenhall, B., Wang, T., Schmidt, D., Srinivasan, P. P., Barron, J. T., & Ng, R. (2021). Learned initializations for optimizing coordinate-based neural representations. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition* (pp. 2846–2855).

Tancik, M., Srinivasan, P. P., Mildenhall, B., Fridovich-Keil, S., Raghavan, N., Singhal, U., … Ng, R. (2020). Fourier features let networks learn high frequency functions in low dimensional domains. *arXiv Preprint arXiv:2006.10739*.

Yariv, L., Atzmon, M., & Lipman, Y. (2020). Universal differentiable renderer for implicit neural representations.

Zhang, K., Riegler, G., Snavely, N., & Koltun, V. (2020). NeRF++: Analyzing and Improving Neural Radiance Fields. *arXiv:2010.07492 [Cs]*. Retrieved from http://arxiv.org/abs/2010.07492