Sparse View Mesh Reconstruction of Plants

Harry Freeman (hfreeman), Gerard Maggiolino (gmaggiol), and David Russell (davidrus)
Carnegie Mellon University

May 6th, 2022

1 Introduction

3D models of plants can be used to address a variety of problems in agriculture. Plant breeding programs are quickly adopting high-throughput phenotyping as an important step in informing the selection of the next generation [1]. This replaces laborious manual measurements which often prevent the whole plot from being surveyed, forcing decisions under greater uncertainty. Some factors, such as the leaf color, are relatively easy to assess from a single viewpoint. Others, such as the angle of a leaf to a stem or the surface area of a leaf, require explicit 3D reasoning. As described in [2], watertight meshes are important for many applications such as characterizing leaf area or plant volume or integration into existing agricultural simulation platforms. To the best of our knowledge, all prior approaches to this problem rely either on images which surround the plant in a controlled environment or active sensing such as Lidar to obtain high-fidelity results.

In this work, we attempted to produce an accurate, watertight, and structurally-plausible mesh from sparse posed viewpoints of a plant. We did this by using neural surface reconstruction to reconstruct a mesh from sparse 2D images. In addition, we made our method generalizable by utilizing state of the art meta-learning techniques in order to create meshes for other crops with a small number of training images. This is a step toward generating high quality meshes in unstructured fields where highly controlled data collection is impractical. This capability can assist high-throughput phenotyping which can improve data-driven crop breeding outcomes for agricultural management.

2 Background

2.1 NeuS

NeuS [3] is a state of the art learning-based 3D reconstruction technique. It leverages the insight that most objects in the world can be represented by opaque surfaces rather than generic volumetric fields. Therefore, they parameterize an signed distances function (SDF) using an multilayer perceptron and extract the final surface as the 0 level-set of this function. This SDF, along with a color field, is optimized with differentiable rendering to match the colors from precicely posed images. A primary contribution of this work is a new volume rendering function which eliminates systematic bias which caused previous approaches to either over- or underestimate the 0 level set. The authors state that their approach can reconstruct well even with sparse viewpoints, which is likely improved by the strong regularization the continuous surface representation provides.

2.2 Reptile and Meta-NeRF

Tancik *et al.* [4] use the Reptile [5] meta-learning algorithm to obtain initial *meta-weights* over a number of independent reconstruction cases. These meta-weights enable single or sparse-view reconstruction with a limited number of iterations for any particular case of the same semantic category. We describe a batch version of Reptile in the context of NeuS [3] and provide pseudo-code (Algorithm 1).

Sub-processes are handed a reconstruction case to perform M iterations of NeuS optimization, initialized from the current meta-weights. The resulting weights from each case are averaged and their difference with the current meta-weights is taken as a gradient. This gradient can be scaled and subtracted from the

meta-weights to perform SGD, but [4] finds Adam [6] optimization substantially improves performance. In contrast to MAML [7] which is limited to 3-4 inner loop iterations, Reptile is a first-order method with no inherent computational constraint on the number of internal steps.

Algorithm 1 Synchronized Batch-Based Reptile with NeuS

$\Phi \leftarrow$ randomly as *meta-weights*
$N \leftarrow$ as outer loop iterations
$M \leftarrow$ as inner loop iterations
$B \leftarrow$ as number of parallel processes
for $n = 1, 2, \ldots, N$ **do**
    Sample B reconstruction cases
    Initialize B cases with Φ_n, run each for M NeuS iterations producing W_i for $i = 1, \ldots B$
    $\nabla\Phi_n = \Phi_n - \frac{1}{B}\sum_i^B W_i$
    Produce Φ_{n+1} with $\nabla\Phi_n$ using Adam
end for

Although Reptile is simple in theory, it's empirically sensitive to the choice of hyper-parameters within both loops. For example, while NerF [8] uses Adam and a moderate learning rate of 5e-4, [4] finds that vanilla SGD with an LR of 1 to 1e-1 performs best for inner loop optimization, depending on the dataset. More details of specific hyper-parameters chosen for NeuS optimization are given in the experiments section.

3 Experiments

3.1 Data

In this work we initially used the UNL-3DPPD [9] dataset which contained a large number of Maize and Sorghum plants imaged from ten viewpoints on a turntable. We were interested in this dataset because all the plants we very similar which would be an ideal situation to leverage meta learning. However, we found this dataset was challenging to work with because of imprecise calibration parameters and the limited number of viewpoints.

Therefore, we transitioned to the recently-released Common Objects in 3D (CO3D) dataset [10] , specifically the *plant* category. The authors collected this in-the-wild dataset from Amazon Turk, where they paid participants to record and object with a smartphone from a circular set of viewpoints. They run COLMAP on this data to produce intrinsic and extrinsic calibration and a sparse pointcloud. Additionally, they compute an approximate foreground mask which highlights the object. There are 571 instances of plants, and each one contains approximately 100 images.

3.2 Standard NeuS Training

To train NeuS we first needed to convert the CO3D dataset into the format that it required. We produced projection matrices from the data contained in the CO3D dataset. NeuS additionally required a scale matrix which made the object lie within a unit sphere. We obtained this scale factor from the noisy COLMAP pointclouds by the following heuristic: we assumed that the bounding sphere was 1.2 times the 95th percentile radius.

We then trained NueS using the default parameter values, except we trained for significantly fewer iterations than the default 300,000. Even on an NVIDIA RTX A6000, a single scene took approximately eight hours, and we thus limited iterations to 20,000 for standard training on all views, and 10,000 for standard and meta-initialized training on sparse views. We found this produced acceptable results and moderately low loss.

3.3 Sparse View Study

As described, CO3D gives us data rich reconstruction cases with 100+ images per plant. We evaluate sparse cases by reducing the number of images available for reconstruction to 30 and 10 views. For both cases the

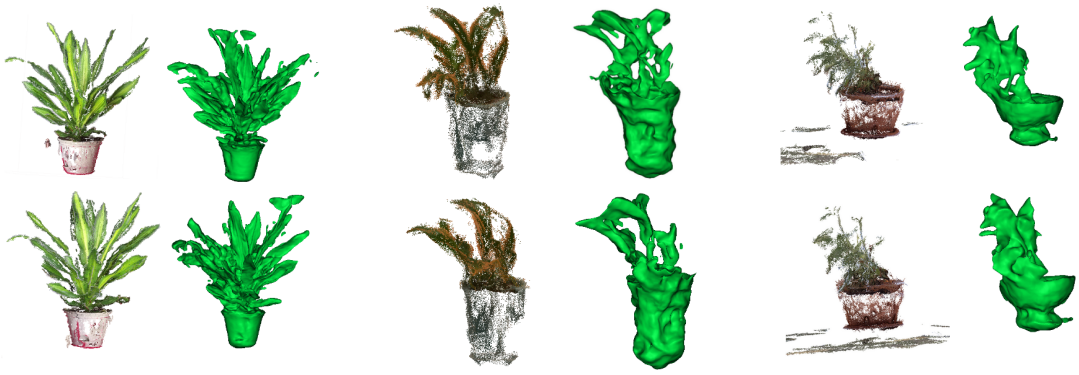


Figure 1: Reconstructions from NeuS trained on all viewpoints. Ground-truth point cloud (left) and extracted surface (right) from two views for three different plants.

views used were randomly sampled. We evaluate a number of plant reconstruction cases on the full (100+), 30, and 10 image views, and report statistics over each for random and meta-initialized weights.

3.4 Meta Learning

To perform meta-learning we follow the Reptile [5] approach, using parameters described in [4] with several exceptions. Notably, [4] trains NeRF for 100,000 outer loop iterations. Their inner loop uses 32 steps for ShapeNet and 64 steps for Phototourism with an SGD optimizer and learning rates from 1 to 0.1, up to 1e5 times greater than the original reconstruction method of NeRF.

NeuS is trained similarly to NeRF with Adam and an LR of 5e-4, but with a warmup period of 5,000 iterations and cosine scheduler. We find that using Meta-NeRF's SGD inner loop LRs of 1 to 0.1 results in severe instability and a complete lack of convergence. We use a reduced inner loop LR of 1e-2. We find that a single iteration of 64 steps takes approximately 8.5 seconds on an NVIDIA RTX A6000 GPU, indicating 100,000 outer loop iterations would take on the order of 200+ hours.

To speed up meta-training, we hand implement a parallelized batch version of Reptile, described in Algorithm 1. We initialize B sub-processes, each with their own GPU. A main process distributes reconstruction cases and places meta-weights into shared memory. Sub-processes initialize using the meta-weights, run the inner loop on their respective cases for 64 steps, and return the resulting weights. The main process then aggregates these weights, performs an update, and places the refreshed meta-weights into shared memory. Each sub-process has additional helper threads which setup and queue the next cases.

In contrast to Meta-NeRF, which uses an outer loop batch size of one case ($B = 1$), we scale up the *outer loop LR* near linearly [11] to the batch size to 5e-3 and are able to train with stability. Due to careful multiprocessing implementation, batched iterations have a negligible slowdown compared to single case updates.

We train Reptile as described across 20 plant cases for 4,000 iterations with $B = 8$ on 8x NVIDIA V100 GPUs, taking approximately 9 hours to complete. From the resulting meta-weights, held-out cases are trained in the standard NeuS matter for 10,000 iterations - this is in contrary to the procedure described in Meta-NeRF which applies the same inner loop optimization (SGD and LR of 1) for 1,000 iterations. We were unable to obtain strong results with the Meta-NeRF fine-tuning, which may be a consequence of our shorter training time or the general instability of NeuS at higher LRs.

4 Results

4.1 All Viewpoints

We first evaluate the performance of standard NeuS training using all viewpoints. Qualitative results of the meshes along with the ground-truth point clouds can be seen in Figure 1 from two different viewpoints.

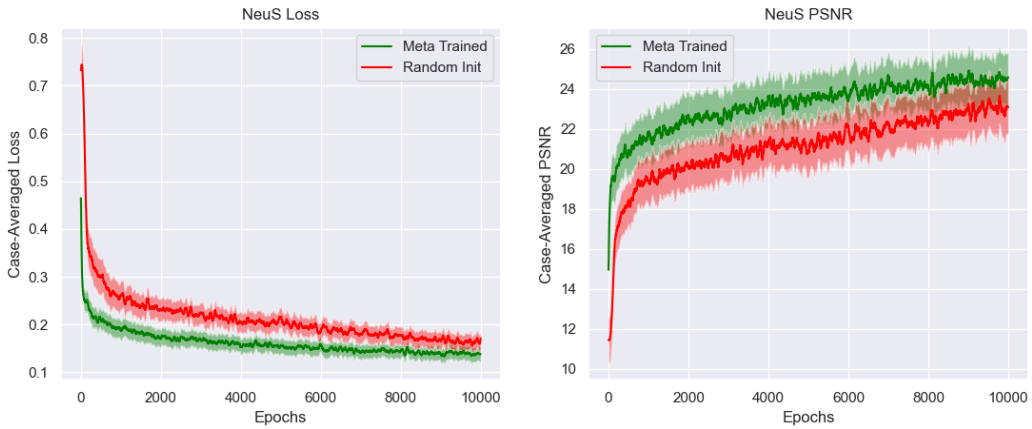


Figure 2: Loss comparison between normal training and meta-weight initialized training

The meshes clearly resemble the coarse components of the plants. However, there is difficulty in accurately representing the thin-dense structures, as seen in the third example on the right. Two possible reasons for this are the number of rays casted when training the SDF, as well as the resolution of the SDF-generated voxel grid for mesh reconstruction with marching cubes.

4.2 Sparse Viewpoints

Next, we compare the performance of NeuS training on a the sparse set of viewpoints, with and without meta-initialized weights. Training proceeds for 10,000 iterations - PSNR and total loss over this period can be seen in 2. Meta-initialized weights outperform the random initialization throughout the entirety of training. Qualitative results from two different viewpoints can be seen in Figure 3 and Figure 4. The first observation is that there is noticeable deterioration in the outputted meshes compared to training on a full set of viewpoints. This is expected as the number of viewpoints were reduced by two and ten times respectively. As well, the the full set of viewpoints were trained for twice the number of iterations. The more interesting result, however, is that the meta-initalized NeuS method significantly outperforms standard NeuS. For the first plant on the left, The standard NeuS extracted meshes do not resemble the correct structure at all, whereas the meta-inialized extracted meshes represent the coarse shape. For the second plant on the right, the meta-initialized result better captures both the coarse shape and the finer irregular surfaces of the upper plant section.

What is also interesting is that for the meta-initialized weights, there is little visual difference between using 30 vs 10 viewpoints, specifically for the first plant on the left. This demonstrates the future effectiveness of this method in agricultural applications where only a small number of viewpoints are accessible.

Further results using 10 viewpoints with meta-learning can be seen in 5. Again, the coarse structure of each plant is well-captured, where some of the finer details are missing. However, these outputs my be sufficient for common agricultural tasks such as volume estimation, sizing, and yield estimation. We also note that these meta-learned results are obtained with significantly less compute than the original papers, and loss continued to drop beyond the first and only 10 hours of batched meta-training. We believe it is reasonable to assume that meta-training over the entirety of plant cases in CO3D (approximately 500) and training for at least the full 100,000 outer loop iterations as described in Meta-NeRF would create meta-weights that are substantially more representative than our obtained weights.

4.3 Unseen View Synthesis

In addition to evaluating the quality of the mesh reconstructions, we also explored this representations ability to perform novel view synthesis. We trained models using the 10, 30, and full splits from random initialization learning and the 10 and 30 splits with meta learning initialization.

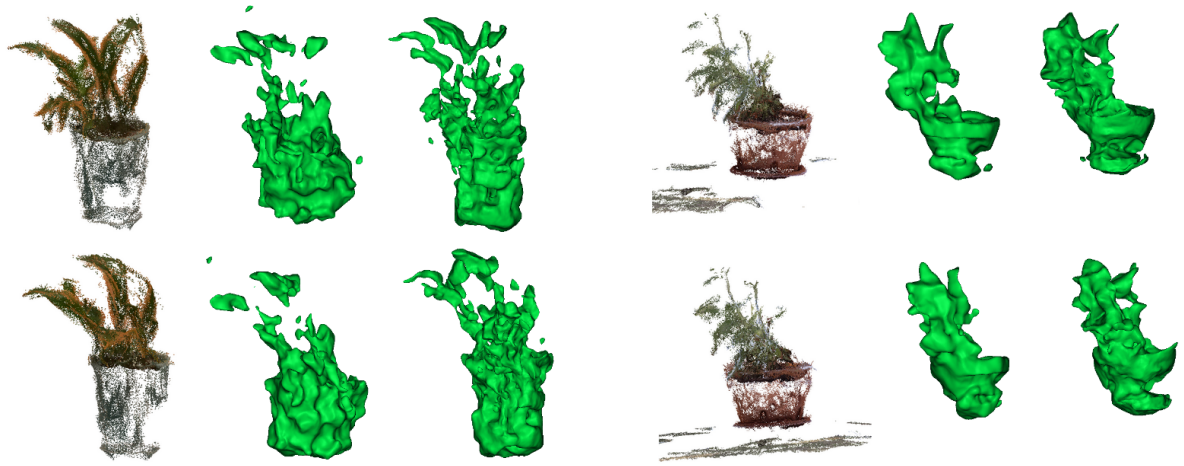


Figure 3: Reconstructions from NeuS trained on 30 viewpoints. Ground-truth point cloud (left), and extracted standard NeuS surface (middle), and extracted meta-initialized NeuS surface (right) from two views for two different plants

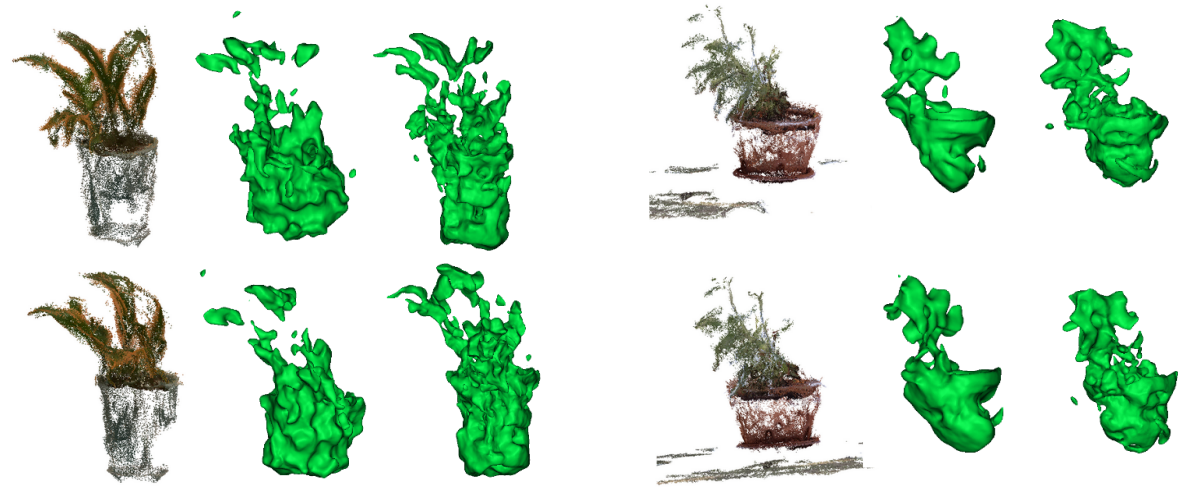
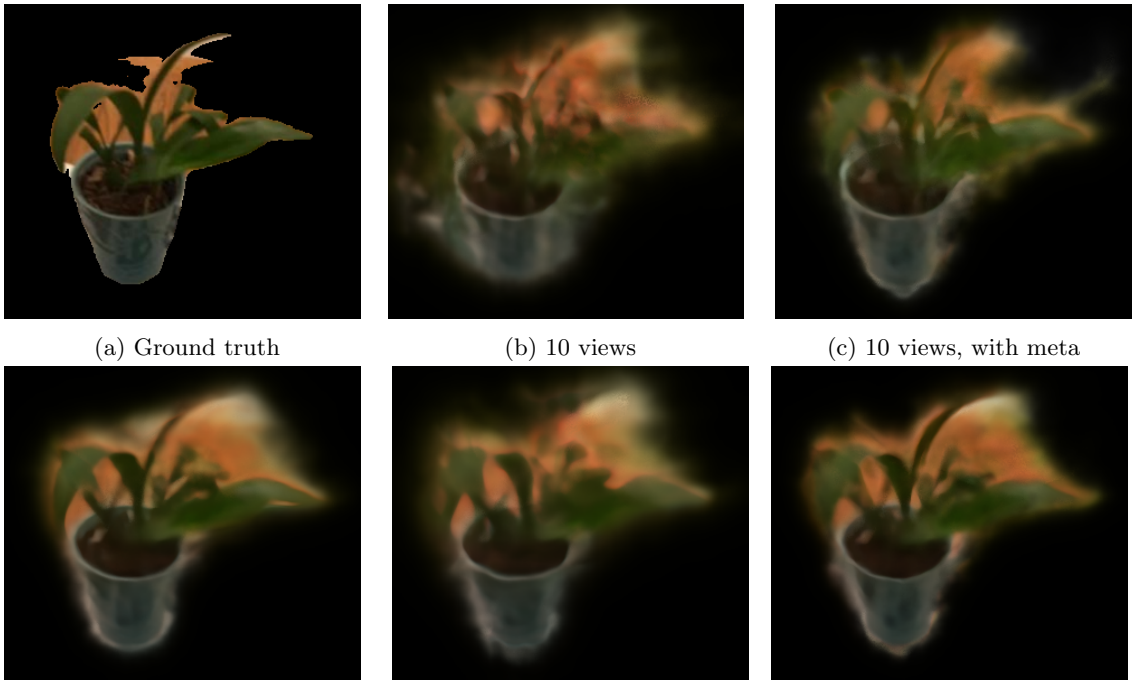


Figure 4: Reconstructions from NeuS trained on 10 viewpoints. Ground-truth point cloud (left), and extracted standard NeuS surface (middle), and extracted meta-initialized NeuS surface (right) from two views for two different plants

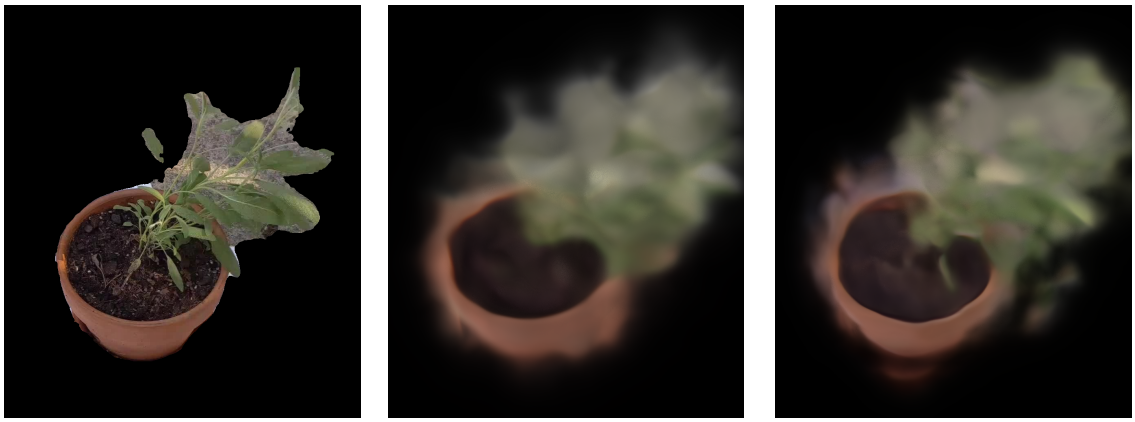
Figure 5: Multiple scenes reconstructed from ten views using meta-learning. Note that the far left instance was seen during meta learning training.



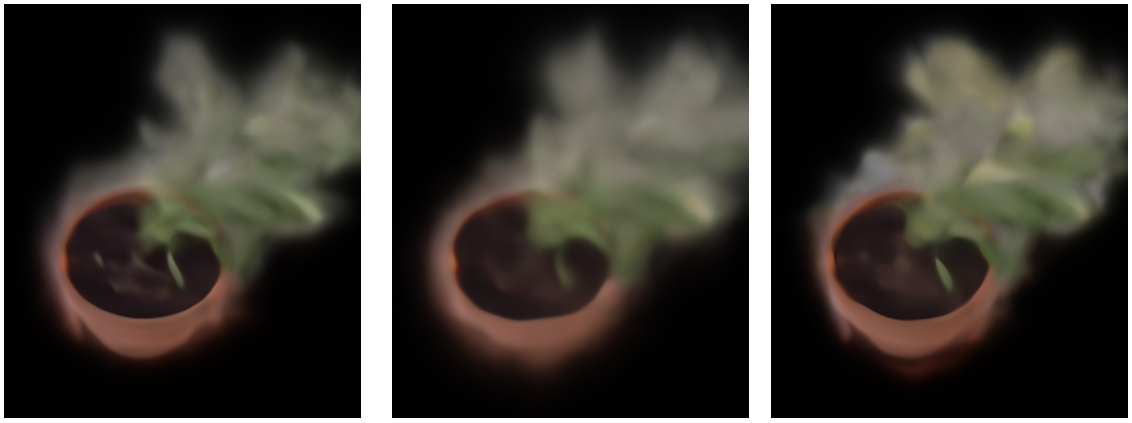
(a) Ground truth (b) 10 views (c) 10 views, with meta

(d) All views (e) 30 views (f) 30 views, with meta

Figure 6: Unseen view synthesis for *plant1*. Note that meta learning improves the results specifically for (f) versus (e), where the leaves are much sharper and the center leaf is properly captured.



(a) Ground truth (b) 10 views (c) 10 views, with meta



(d) All views (e) 30 views (f) 30 views, with meta

Figure 7: Unseen view synthesis for *plant2*. Even the reconstruction with all viewpoints fails, likely due to the thing structures. Meta learning is especially useful in the (c) where it allows the network to begin picking out some structure in the plant.

To evaluate the quality of these reconstructions, we render an image from a camera which was not seen during training and compare the result to the real image. The results of this can be seen for two different plants in Figure 6 (*plant1*) and Figure 7 (*plant2*).

Note that *plant1* is fairly well represented from a novel viewpoint while *plant2* is very blurry. This is likely because the former has much less detail and a lower resolution then the later. In both cases, meta-learning made the images sharper and produced results which better captured the structure of the leaves.

5 Conclusion

We explore the problem of reconstructing mesh representations of plants from sparse viewpoints. We leverage the insight that class-level priors should speed up the reconstruction process and make it more robust to sparse viewpoints. Both of these attributes are beneficial for in-the-wild deployment of learning-based reconstruction. To test this hypothesis, we implement a meta-learning approach which learns an initialization that is well-suited to a variety of tasks. We effectively demonstrate that meta learning does improve the quality of these reconstructions. In the future, we hope to provide more quantitative results using the Chamfer distance of the extracted mesh and the PSNR of novel synthesized views. We provide our data processing scripts [1] and metalearning implementation [2] to facilitate future work in this direction.

[1] `https://github.com/russelldj/leaf_reconstruction`

[2] `https://github.com/hfreecmu/NeuS`

References

[1] Eric Rodene et al. "A UAV-based high-throughput phenotyping approach to assess time-series nitrogen responses and identify trait-associated genetic components in maize". In: *The Plant Phenome Journal* 5.1 (2022), e20030.

[2] Anjana Deva Prasad et al. "Deep implicit surface reconstruction of 3D plant geometry from point cloud". In: *AI for Agriculture and Food Systems*. 2021.

[3] Peng Wang et al. "Neus: Learning neural implicit surfaces by volume rendering for multi-view reconstruction". In: *arXiv preprint arXiv:2106.10689* (2021).

[4] Matthew Tancik et al. "Learned initializations for optimizing coordinate-based neural representations". In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 2021, pp. 2846–2855.

[5] Alex Nichol, Joshua Achiam, and John Schulman. "On first-order meta-learning algorithms". In: *arXiv preprint arXiv:1803.02999* (2018).

[6] Diederik P Kingma and Jimmy Ba. "Adam: A method for stochastic optimization". In: *arXiv preprint arXiv:1412.6980* (2014).

[7] Chelsea Finn, Pieter Abbeel, and Sergey Levine. "Model-agnostic meta-learning for fast adaptation of deep networks". In: *International conference on machine learning*. PMLR. 2017, pp. 1126–1135.

[8] Ben Mildenhall et al. "Nerf: Representing scenes as neural radiance fields for view synthesis". In: *European conference on computer vision*. Springer. 2020, pp. 405–421.

[9] Sruti Das Choudhury et al. "Leveraging Image Analysis to Compute 3D Plant Phenotypes Based on Voxel-Grid Plant Reconstruction". In: *Frontiers in Plant Science* 11.December (2020), pp. 1–18. ISSN: 1664462X. DOI: 10.3389/fpls.2020.521431.

[10] Jeremy Reizenstein et al. "Common Objects in 3D: Large-Scale Learning and Evaluation of Real-life 3D Category Reconstruction". In: *International Conference on Computer Vision*. 2021.

[11] Priya Goyal et al. "Accurate, large minibatch sgd: Training imagenet in 1 hour". In: *arXiv preprint arXiv:1706.02677* (2017).