# 3D Reconstruction of Infant Keepsakes using Deep Neural Networks

Derek Benham and Ashton Palacios

***Abstract*— For this semester project we take the first steps in fleshing out our innovative idea: reconstructing infant keepsakes using advanced neural reconstruction techniques. Leveraging insights from Structure-from-Motion (SfM) and Neural Radiance Fields (NeRFs), the project aims to create high resolution models of an infant hand or foot. The exploration encompasses related works, data collection methodologies, and approaches for generating accurate hand models. With an eye on potential applications, including the possibility of a startup, the project taps into the current momentum in neural reconstructions. The subsequent sections detail the exploration process, methodologies, and our successful initial results that are leading us to explore the space further.**

## I. Introduction

As this semester project required we not double dip on research, we figured it was the perfect time to finally put some time towards the idea we've had for a couple of semesters—reconstructing baby hands. The idea began last Christmas when Derek was making a hand-print ornament of his daughter that he realized there must be a better way. We've had been exposed to Structure-from-Motion (SfM) through our research, however we knew that SfM would fail to create a high enough resolution model. Through the research and presentations of our colleagues, we were exposed to Neural Radiance Fields (NeRFs) but still didn't know enough about them because they are still (relatively) new. Through research and development on this semester project we realized that with the recent developments in the field of neural reconstructions, now is a unique time in which we can leverage some of the existing literature to potentially create a startup, or at least warrant further investigation.

Our semester project focuses around creating a full pipeline starting at taking a short, simple video on a smart phone to the finished 3D printed product. There are many stages that depend upon each other meaning any introduced errors will be magnified through the entire pipeline. Most of our project was spent trying out different approaches and refining certain processes to get good results. We begin with a related works section on some of the ideas we looked into and ultimately tried implementing. Next, we cover our approach to collecting data, solving camera poses, and prepping data to be used in surface reconstructions. And then we cover the different approaches we implemented to create hand models from our different data sets. Finally, we cover our results followed by a conclusion with future work.

## II. Related Works

Before diving into the related technical work, we want to first address some of the related methods that still deliver an infant keepsake for parents and grandparents.

D. Benham, A. Palacios are at Brigham Young University.

Fig. 1: Printed hand model using Neuralangelo.

### A. Manual Model Creation

The product that inspired this idea is a simple clay kit with a ribbon. A parent rolls out the clay and then presses their child's hand or foot in the center to leave an impression. It makes a great Christmas ornament or a decoration to hang on the wall in the nursery. Some of the problems that arise with this method is parents have to press hard on the child's hand to get an impression. This could lead to potentially harming the child. Despite the insistent pressing, the impression was still shallow and required painting to really pop. Lots of detail is lost in the impression as the main takeaway is the size of the child's hand. Regardless, it still makes a great gift for $20.

Another common approach is the physical cast. These can be purchased as a home kit, but hospitals and doctor's offices may also have the materials on hand to make a replica. For an at home kit, the product can be a bit messy and tedious, however from our analysis this product can give the best results. Like the clay impression, both kits are premeditated products that require more work from the parents. The act of purchasing a kit and casting are two separate occurrences, with the latter being rather time consuming. With a digital model, and if we can get faster results in the future, parents could get a digital turnaround time of half an hour with a physical cast mailed shortly after. Casting kits also sit in

the price range of $20 to $30 dollars. In contrast to the at home kit, casts in doctors offices or hospitals may be done to preserve the memory of a child who passed away too soon. It can be scary to trust our algorithm or the quality of a video from a nurse to create such a priceless keepsake, but in places where manual casting is not possible, a digital recreation could be the second best option.

One last interesting product we found is an embossed print of a baby's foot generated from the ink-print done by the hospital. There is no shortage of Etsy shops selling some variant of this including below average 3D prints, but there are many shops who sell a very well built frame and display case. Judging by the quantity of sellers and the amount of hits some of these shops get, we believe there is a realistic market for such a product, and no shortage of newborn babies or excited grandparents.

### B. Structure from Motion

Recreating an object from camera imagery is no new idea. SfM [1] is a common technique that aims to reconstruct the three-dimensional structure of a scene from a sequence of two-dimensional images or video frames. The fundamental idea behind SfM is to estimate the camera poses and the 3D positions of feature points observed in multiple images, thereby recovering the spatial relationships between objects in the scene. The process involves tracking distinctive features across frames, establishing correspondences between them, and utilizing these correspondences to compute the camera poses and reconstruct the 3D geometry. SfM has found widespread application in various fields, including robotics, augmented reality, and geographic information systems. Its ability to generate 3D reconstructions from ordinary image sequences makes it a valuable tool for tasks such as scene modeling, object recognition, and camera calibration, contributing significantly to the development of realistic and immersive visual environments.

Despite its versatility, SfM faces several challenges. One notable limitation is its sensitivity to factors such as lighting variations, occlusions, and repetitive patterns in the scene, which can lead to errors in feature matching and, consequently, inaccurate 3D reconstructions. Additionally, the resolution of generated models can be a concern. SfM typically uses triangulation between points to create a polygon mesh, which means it may struggle when recreating objects with low feature points at a high resolution. Forcing the algorithm to find more feature points will increase computational demands as well as induce more outliers. Robustness to outliers and the need for sufficient image overlap for reliable reconstruction are other challenges associated with SfM. Addressing these shortcomings often requires advanced techniques, such as incorporating additional sensor data, utilizing more sophisticated feature matching algorithms, or integrating SfM with other computer vision methods to enhance accuracy and robustness in challenging scenarios.

### C. Volume Rendering

To overcome some of the shortcomings of SfM, a new, groundbreaking field, has recently emerged in computer vision for synthesizing highly detailed and realistic 3D scenes. Introduced in a seminal paper, NeRF [2] utilizes neural networks to model voxels, assigning each voxel an opacity, density and color value. This neural representation allows for the generation of high-fidelity images by ray tracing, where rays are cast from a virtual camera through pixels, and the neural network is employed to predict radiance along these rays. NeRF excels in capturing intricate geometry, fine details, and realistic lighting. The versatility of NeRF extends to enabling novel views of scenes not present in the training data, making it a powerful tool for synthesizing visually compelling and immersive 3D environments.

Gaussian Splatting [3] is a newer technique that builds upon some of the ideas of NeRFs but instead represents the scene with 3D Gaussians that preserve the desirable properties of continues volumetric radiance fields. Additionally, it is optimized to avoid unnecessary computation in empty spaces. Gaussian Splatting has taken the field by storm as it can render high fidelity scenes better than NeRFs in a fraction of the time.

Like NeRFs, Gaussian Splatting is also a form of volumetric rendering. These methods can generate realistic images of complex scenes by simulating the interaction of light with a 3D volume of data, but struggle to recreate physical 3D meshes. Generating a mesh can be computationally intensive as it involves calculating the radiance at numerous points within the volume. Additionally, the resulting volumetric representations may lack the geometric precision and efficiency required for certain applications, such as virtual environments or model recreation. The conversion from volumetric data to a physical 3D mesh may also introduce artifacts, impacting the fidelity of the reconstructed geometry.

Surface Rendering Generating novel views with volume rendering doesn't inherently mean it has an understanding of the objects underlying structure. Ultimately, a different method than radiance fields is needed to create high fidelity 3D models, specifically a neural network needs to be optimized to produce a mesh rather than a novel view. This is where the idea of surface rendering has begun to fill the space. Instead of using a neural network to output opacity, density, and color values for each voxel, surface rendering models output a surface directly, typically done using the signed distance function (SDF).

One of the first papers to achieve high resolution surface modeling was NeuS [4]: Learning Neural Implicit Surfaces by Volume Rendering for Multi-view Reconstruction. NeuS addresses challenges in neural volume renderings by overcoming local minima and constraints related to varying depth changes and occlusions. It leverages both volume-based rendering, like NeRFs, and surface representations, using a signed distance function (SDF) to ensure robust constraints on level sets. The zero-level set of NeuS's neural implicit SDF represents the object's surface, where the func-

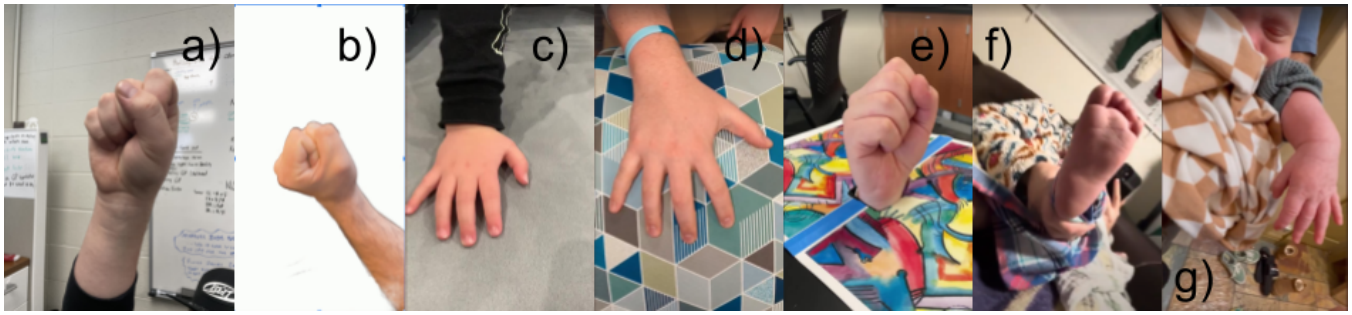


Fig. 2: Data set collection progression. A) is the original video collected. B) is segmenting out the background. C) is a hand laid flat on a plane. D) is laid flat but on a unique background. E) is a combination of what we learned on a custom background. F) is of a toddlers foot and G) is of a newborns hand while he was sleeping.

tion equals 0. To learn neural network weights, the authors introduce a novel volume rendering method, minimizing the difference between rendered and input images. The scene is represented by functions mapping 3D points to their SDF and encoding color based on spatial points and viewing direction. The authors introduce the S-density, combining SDF and a logistic density distribution, guiding the training of the SDF network through volume rendering with 2D input image supervision. Ultimately, NeuS aims to accurately reconstruct surfaces through this innovative integration of volume rendering and SDF-based neural representations.

One of the downsides of NeuS, and the field in general, is prolonged training times. NeuS can take upwards of 8 hours to train on a single GPU. NeuS2 [5] introduces optimizations for significant speedups, achieving a remarkable 100x speedup. The implementation utilizes hash encodings and multi-resolution hash tables of learnable feature vectors to parallelize the neural network encoded Signed Distance Function (SDF). The authors utilize CUDA and derive second-order derivatives tailored to ReLU-based MLPs to enhance training efficiency. A progressive training strategy updates the hash table from coarse to fine grade features, and an incremental learning method extends the approach to dynamic scenes. NeuS2 provides a notable solution by incorporating ray marching acceleration strategies and minimizing color differences for efficient volume rendering of hash-encoded SDF. The CUDA implementation significantly accelerates convergence, emphasizing simplicity in the learning algorithm to avoid local minima. Although we were unable to test NeuS and NeuS2 in this work, we plan to further explore them in the near future.

Like Neus2, the most recent and accurate models have been built around an encoding scheme called multi-resolution hash encoding and numerical gradients. This encoding directs the underlying optimization on the level of surface detail to model. These models optimize starting with coarse grain details and slowly move towards the finer details. Higher order gradients are used to be able to capture and model these finer details. Two examples of using this encoding scheme is InstantNGP [6] and Neuralango [7]. InstantNGP is more like the traditional NeRF and solely focuses on rendering visuals while Neuralango focuses on creating surface meshes. InstantNGP does not suffer from prolonged training times, but Neuralango does. What both struggle with is the amount of VRAM on the GPU it takes to store the multi-resolution hash encoding. Although Neuralango struggles with training time and the amount of training memory, Neuralango is the backbone of our project. It provides accurate models that can easily be post processed and printed.

## III. Input Processing

Many iterations were spent as we refined how we prepared data to be trained by a model. We originally planned to have a newborn hand modeled by the end of the project but much of our time was spent iterating between data collection of adult hands and model generation as we worked to refine the pipeline.

### *A. Solving Camera Poses*

Since part of the input to these neural methods are camera poses, they must be provided known poses. They don't simultaneously solve for the pose and the shapes structure like SfM. Our first approach to solving camera poses was to use the LiDAR in newer iPhone pro models. Using methods like the iterative closest point (ICP) algorithm, an optimizer can solve for the poses of the camera. Documentation exists for using the iPhone app Record3D to export a list of camera poses solved with the LiDAR. Unfortunately nobody on our team owns an iPhone with a LiDAR, but we were able to borrow one. Even so, the Record3D app only allowed a handful of free scans before requiring a subscription. We were able to record two different views of an adult hand, but due to the limited access we weren't able to fully flesh out this method as a feasible solution. Perhaps in the future when LiDARs in phones become commonplace, this solution would be more practical.

Nevertheless, traditional methods like SfM can still be used to solve for the initial camera poses and a point cloud that are fed into neural methods that can be used to generate novel views or a surface mesh. We tested out two different SfM software packages, Metashape by Agisoft, and COLMAP [1]. We ultimately went with COLMAP due to existing pipelines and that it is open source.

### B. Data Set Collection

For most of our data sets we collected 15 to 30 second videos of an adult hand. It was easier to acquire repeat videos of an adult's hand because the adult wouldn't inadvertently move their hand during data collection. We found that a longer video was unnecessary and that too short of a video wouldn't have enough coverage or motion blur. Collecting data was an iterative process as we believed that our neural reconstructions were failing due to various parameter or training reasons. Eventually, we realized it wasn't the neural reconstruction process failing, rather the old ML euphemism of "garbage in equals garbage out". We weren't providing our neural methods good starting data to begin with.

To explain how we came to the conclusion that our camera poses were rather poor, we'd like to share the process we went through to get there. An overview can be seen in Figure 2. We began by recording a raised fist in the air to easily segment out any background (Fig. 2a), but we learned that it was hard to solve for the camera poses when there was a large disparity between background and foreground. Continuing along the idea of trying to easily segment out the background we deployed a neural network to automatically remove the background from images (Fig. 2b). We tried to force the SfM to only find matching correspondences on the hand with the thought that we could potentially keep the camera in a still place and move instead twist the hand. This approach quickly failed, and we learned that the image background is necessary to aid in solving camera poses. To simplify the shape that had to be reconstructed, we then collected videos of a hand laid flat on a plane (Fig. 2c). This approach made the disparity between background and foreground much closer, but still was unsuccessful. We realized that the backgrounds were not unique enough and that hands alone don't provide enough unique feature descriptors either.

Building upon these ideas we found our first relative success when we laid a hand flat against a checkered pattern (Fig. 2d). We finally generated a hand model, albeit with some noise. This is when we finally realized our camera poses weren't well defined. Realizing we need some sort of unique background we made our own by printing out the results of "abstract art" from a google images search. We cut a hole in the middle and stuck a fist through before recording a video (Fig. 2e). This method worked the best and resulted in the final product as shown in Figure 1.

By providing the best input data possible we were able to find optimal parameters for the neural approach in order to create a 3D model. With that solved we were finally able to branch out and collect data more reminiscent of what our final consumer product could be. We collected a 10 second video of a very wiggly toddler's foot. She wasn't an ideal test subject but it's what we had to work with. Her mother wore a patterned shirt and we wrapped a blanket around her ankle to easily segment out the object of interest (2f). Since presenting our project, we also captured several short videos of a one month old's hand and foot (Fig. 2g). At the time of this report we've been unable to proceed with that data but it's something we'll test shortly

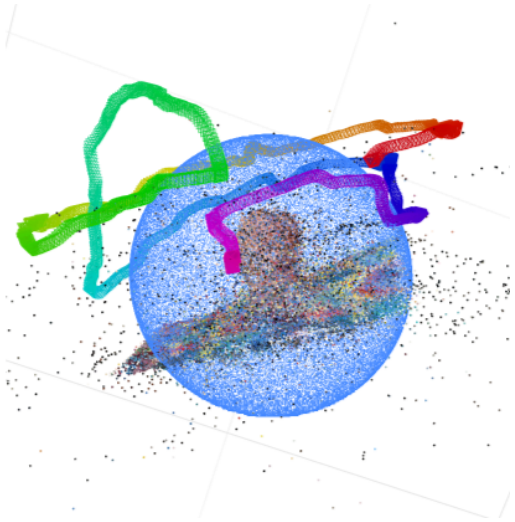

Fig. 3: Camera poses, point cloud, and bounding sphere generated from COLMAP and Python visualizer. Camera poses and bounding sphere is fed into model generation algorithm.

### C. Point Cloud Generation

Many NeRF and NeRF adjacent techniques utilize known camera poses and initial point cloud estimates. Like stated before, we use COLMAP to generate both of these for our pipeline. We do not directly use the point cloud generate by COLMAP in our model, but use it to direct the algorithm to the portion of the scene we want it to optimize around. We use either a simple 3D plot visualizer or Blender to view and manipulate a bounding sphere over the point cloud. Neuralango, out model generator, will only optimize the area within the bounding sphere. Our approach does not explicitly use the points within the sphere but many other techniques do. Gaussian Splatting for example starts with the point cloud as the initial locations of the gaussians. From there, the gaussians are split and optimized. When viewing the bounding sphere for our pipeline, we can also verify the camera poses are approximately where we expect them to be. If they are not, a new video needs to be acquired because the final model will not be accurate enough. Once the sphere is placed and camera poses verified, this information is fed into the model generation portion of our pipeline. Fig. 3 shows an example point cloud, bounding sphere, and camera poses for the model that is evaluated in Section V and shown in Fig. 1.

## IV. Model Generation

Model generation is the core idea behind our project. We tried many techniques to generate an accurate model of a fist from a simple video. The first we tried is Structure from Motion. From there we quickly moved onto neural methods. NeRFStudio [8] is a collection of old and new NeRF projects built into a single application. From this application we were

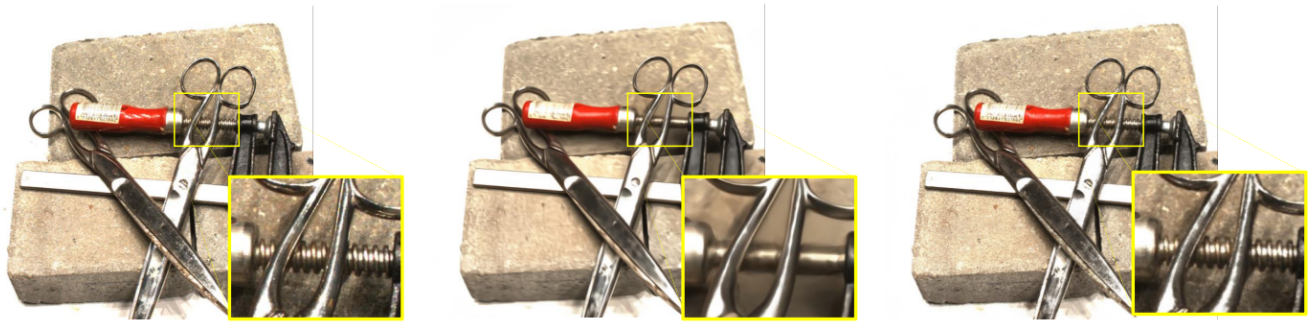

Fig. 4: Output of Neuralangelo compared to output of Neus algoirthm. Left) input image. Middle) Neus output. Right) Neuralangelo output.

able to try a few different NeRFs to generate our models. Finally, we landed on using Neuralangelo to generate our models.

### A. SfM

Structure from Motion is an ideal place to start for 3D reconstruction. The SfM algorithm generates camera poses and a sparse point cloud. The sparse point cloud does not contain enough detail to accurately model 3D objects typically. From the camera poses, a dense point cloud can be generated. From the dense point cloud 3D objects can start to form. These points can then be put into a mesh format to be printed. The biggest challenge for this method though is when there are occlusions and other introduced noise sources. Noise can be introduced from inaccurate camera poses to having short baselines between camera poses when triangulating points. All this noise make the 3D reconstructions noisy and not great models to be 3D printed. Once we discovered this, we moved onto the NeRF methods.

### B. NerfStudio

NeRFStudio is an application that houses a collection of NeRF projects that pushes forward NeRF research. Researchers can quickly experience and iterate on different NeRF projects. While NeRFStudio is a worth while project it still has a long way to go. There are many dependency issues for each of the NeRF projects that is supports that need to be resolved. We spent a long time trying to debug these issues to no avail for many of the projects. For the projects we were able to successfully run and train with we had terrible results. Traditional NeRFs do poorly on 3D reconstruction. In many cases, SfM would be the better solution. The underlying representation of environments in these NeRFs are not conducive to 3D mesh extraction. More recent work has proven to be more reliable and accurate at creating the types of models we are looking for.

### C. Neuralangelo

Neuralango was presented at CVPR 2023 and won Time's Best invention for 2023. It has made a large splash in the 3D reconstruction world with how much detail this model can extract from images. Neuralango is built upon multi-resolution hash encoding, SDF, and numerical gradients. Each of these ideals work in tandem with each other to enable coarse-to-fine detail optimization in the algorithm. The algorithm starts with the basic structures in the environment the user has selected. The hash encoding resolution starts at a large value. This effectively smooths over a large area. This generates the coarse details of the environment. As the optimization continues, the hash encoding resolution becomes finer. As it becomes finer, more detail starts to appear because the fine details is no longer being smoothed over. This approach is perfect for capture the large and small details of a hand or foot. The first epochs of the optimization roughly captures the outline of a hand, and then the later epochs slowly captures the wrinkle lines, finger nails, and even protruding veins. Neuralangelo uses a lot of VRAM on a GPU to store high resolution hash encodings while training. It also can take hours to train a single model. Future work will include optimizing both of these constraints. This will appear as new code be written and hyper-parameter tuning. Fig. 4 show a single snippet from the original Neuralangelo paper at the minute details that can be recovered the Neuralangelo approach.

### D. Model cleanup and Printing

Once our hand or foot model is generated, a small amount of post processing needs to be performed. We use a 3D modeling software named Blender. Blender is an open source, community driven modeling package that has been around for over 30 years. We used it to clean up sharp edges that are left over artifacts of training. The placement of the bounding sphere influences what sharp edges are created and how much post processing needs to be performed. If the bounding sphere includes an area that had no or insufficient image data the Neuralangelo algorithm does not know how to optimize leading to sharp or unusable geometry. We use Blender to smooth out these edges to prepare to print our models.

The two primary printing techniques are fused deposition modeling (FDM) and stereolithography (SLA). FDM printing is the traditional 3D printing method that involves melting plastic and extruding it in layers to create an object. SLA, or resin, printing uses a vat of UV light hardening resin and UV lasers to create objects. FMD printing is considerably cheaper, easier, and faster to work with, but comes at the cost

Fig. 5: Hand model printed with a FDM printer.

of object resolution. SLA on the other hand is completely the opposite. It is temperamental and prints slowly but captures a great amount of detail. We have access to both types of printers and can provide either service for potential customers.

## V. Results

We have two categories of metrics that we evaluated our pipeline against. The first is more subjective and evaluates solely the visual appeal of the final product. The second evaluates the model loss, VRAM use, and pipeline time use. We need a pipeline that is fast and efficient to generate lots of models.

### A. Visual

Our final models that we are able to create and print show off a decent amount of detail. We are able to capture the creases of skin on the palm of my hand. We are also able to capture fingernail outlines. If we let the model run long enough we are able to capture the slight bulge of the veins on the back of my hand. These fine details are shown in our SLA print in Fig. 1. When we print using FDM printers we are able to print at faster speeds but drop some of the detail as shown in Fig. 5. Either choice still shows off a great amount of visual detail.

Once the digital files are created for the object that is being reconstructed the object can be printed as many times as the user wants. This is a huge improvement over having a single cast or mold of an object. If the cast or molded object is ever lost or damaged there is usually no way to fully recover the priceless object. With our reconstruction method, we are simply able to reprint a new one. This also enables multiple people to enjoy the reconstructed object as well.

### B. Pipeline Efficiency

The second metric we evaluated is the training loss provided by Neuralangelo. Fig. 6 show that our model was

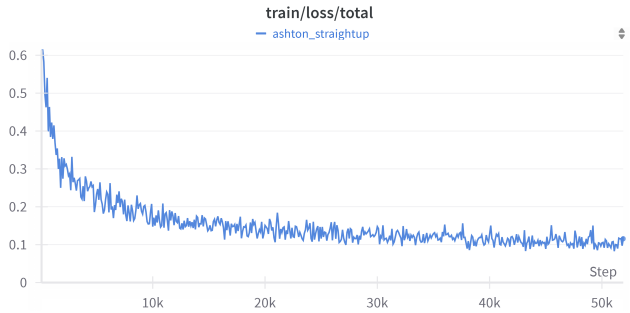


Fig. 6: Total loss of trained model.

optimizing correctly and hadn't started to overfit the model because the loss was still decreasing. We have made models that overfit according to the training loss that led major artifacts on the final model that are either unacceptable or needed a decent amount of post processing to be usable. We are still trying to find the balance between underfitting and using less computation time to overfitting and using lots of time. The model that corresponds to Fig. 6 took approximately 4 hours of training time to balance model accuracy and training time without overfitting. Another metric we looked at is how much VRAM we are using on the GPU. According to the data we almost used the full 20GBs of VRAM on our Nvidia A4500 GPU. This is not acceptable if this project is to be done at scale. This will be the first optimization we will look at. We will see how we can improve the memory footprint or switch to a similar model that is less accurate but uses less memory.

The final piece of our pipeline that could experience more optimization that we evaluated is our SfM process. Depending on the input video, the SfM portion of the project can take between 5 minutes up to 4 hours to calculate depending on the quality of the video and its length. Optimizing this will come down to how we collect short videos. We need a process that captures enough detail to recover camera poses and enough views of the object that can be reconstructed.

## VI. Conclusion

This semester project has been rewarding and challenging at the same time. We have fought with too many CUDA dependency issues to exhaustively name. We have learned that camera poses are more important than what is casually stated in many NeRF papers. We have learned how to use Blender. We have learned the invaluable place GPUs play in speeding up training machine learning models. But most importantly, we have learned that many people like the idea of having a printable keepsake of their child or grandchild. We did not estimate the amount of positive feedback we would get from the class or others we have shared this idea with. We will continue to work on this project and run it through the engineering college's Student Innovator of the Year experience. Our pipeline that we created for this project still needs a lot of optimization to make it work at scale, but throughout this semester we have been able to get off to a great start.

## REFERENCES


[1] J. L. Schönberger and J.-M. Frahm, "Structure-from-motion revisited," in *Conference on Computer Vision and Pattern Recognition (CVPR)*, 2016.

[2] B. Mildenhall, P. P. Srinivasan, M. Tancik, J. T. Barron, R. Ramamoorthi, and R. Ng, "Nerf: Representing scenes as neural radiance fields for view synthesis," 2020.

[3] B. Kerbl, G. Kopanas, T. Leimkühler, and G. Drettakis, "3d gaussian splatting for real-time radiance field rendering," *ACM Transactions on Graphics*, vol. 42, no. 4, July 2023. [Online]. Available: https://repo-sam.inria.fr/fungraph/3d-gaussian-splatting/

[4] P. Wang, L. Liu, Y. Liu, C. Theobalt, T. Komura, and W. Wang, "Neus: Learning neural implicit surfaces by volume rendering for multi-view reconstruction," *arXiv preprint arXiv:2106.10689*, 2021.

[5] Y. Wang, Q. Han, M. Habermann, K. Daniilidis, C. Theobalt, and L. Liu, "Neus2: Fast learning of neural implicit surfaces for multi-view reconstruction," in *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, 2023.

[6] T. Müller, A. Evans, C. Schied, and A. Keller, "Instant neural graphics primitives with a multiresolution hash encoding," *ACM Trans. Graph.*, vol. 41, no. 4, pp. 102:1–102:15, Jul. 2022. [Online]. Available: https://doi.org/10.1145/3528223.3530127

[7] Z. Li, T. Müller, A. Evans, R. H. Taylor, M. Unberath, M.-Y. Liu, and C.-H. Lin, "Neuralangelo: High-fidelity neural surface reconstruction," in *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2023.

[8] M. Tancik, E. Weber, E. Ng, R. Li, B. Yi, J. Kerr, T. Wang, A. Kristoffersen, J. Austin, K. Salahi, A. Ahuja, D. McAllister, and A. Kanazawa, "Nerfstudio: A modular framework for neural radiance field development," in *ACM SIGGRAPH 2023 Conference Proceedings*, ser. SIGGRAPH '23, 2023.