3D Object Reconstruction from 2D Images

Sameer Khan, Elena Kote, Kent Shibata

Introduction

3D object reconstruction is the process of estimating the model of a 3D object from multiple 2D images. This technology is used in many industries including robotics, film, civil engineering, and many other fields. Many approaches have been created to tackle the problem of 3D reconstruction ranging from the 1960's classical computer vision approaches to current day deep learning approaches. The purpose of our project is to compare different methods for 3D object reconstruction on their performance, complexity, limitations and more. Specifically, we will be comparing two classical computer vision approaches, structure from motion and reconstruction from disparity, as well as two neural network approaches, NeRF and NeRF--.

Methods

Structure from Motion (SfM)

- SfM tries to estimate 3D structure by using the relationship between consecutive images in an image sequence.
- The pipeline roughly goes:
    - Feature Matching
        - SIFT
    - Pose Estimation
        - Epipolar Geometry
        - Perspective N-point
    - Triangulation
    - Reprojection Error
- Camera intrinsic parameters and a correctly sequenced set of images are the only necessary inputs.
- This creates a sparse point cloud of the object using the feature points found in feature matching and triangulating them into 3D space.
- I primarily used OpenCV in python to construct the pipeline as well as the Middlebury MVS dataset as a test set.

Reconstruction with Disparity (RwD)

- The dataset used was the 2014 Middlebury stereo images. (Link below)
- https://vision.middlebury.edu/stereo/data/scenes2014/
- Method steps:
    - Preprocess stereo images with Gaussian Filter to smooth out features
    - With the two stereo images, calculate disparity using sum of absolute differences (SAD) method
    - Create disparity map using provided intrinsic parameters in the dataset
    - Apply averaging filter to the disparity map to smooth out the image
    - Calculate depth (z coordinates) of the pixels with the disparity map
    - Output a point cloud data file with the x, y, and z coordinates of the pixels
    - Plot the point cloud in 3D space
- Used OpenCV library to apply filters and output disparity maps to the stereo images
- Used CloudCompare (software) to visualize 3D point clouds

Neural Radiance Fields (NeRF and NeRF--)

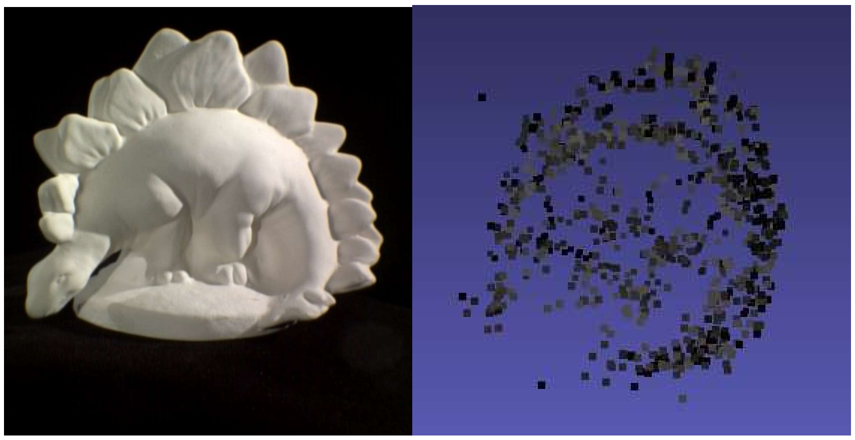
- The dataset used to train our NeRF was the NeRF synthetic dataset (https://drive.google.com/drive/folders/128yBriW1IG_3NJ5Rp7APSTZsJqdJdfc1)
    - We used lower quality versions of the pictures, so that we could train our model to convergence, with the computational resources we had at our disposal
    - The dataset consists of 360-degree images of objects with a blank background which helps with the learning task
- NeRF requires camera intrinsics and extrinsics to train. To do so, it first uses a classical Computer Vision tool called COLMAP, which does structure from motion and estimates the intrinsics and extrinsics of a set of images it is given. It also produces a point cloud of the object, which NeRF utilizes
- NeRF hierarchically samples points from the point cloud and then tries to project the rays, using the plenoptic function. The plenoptic function is what the model is trying to learn.
- We trained our model using PSNR loss, over 5000 epochs, until it reached convergence at PSNR = 24.5
- Our evaluation metrics for the model were PSNR (Peak Signal to Noise Ration) and SSIM (Structural Similarity Index), which allowed us to compare our model to the one in the original NeRF paper (https://arxiv.org/pdf/2003.08934)
- The dataset used to train NeRF—was the NeRF LLFF dataset (https://drive.google.com/drive/folders/128yBriW1IG_3NJ5Rp7APSTZsJqdJdfc1)

    - This dataset consists of pictures of outdoors objects including a background taken from a front-facing perspective
- NeRF—is a model that learns the plenoptic function (just like NeRF), as well as the intrinsics and extrinsics of the set of images given to it. So, in contrast to NeRF, it does not rely on COLMAP
- Because it is learning all these parameters, and because all the data that it was trained on does not have 360-degree views of objects, it does not perform well with 360-degree model recreations.
- It performs better with novel view synthesis of forward-facing scenes
- We used a pretrained model and then tested it on our data, using the same evaluation metrics as NeRF above
- Used Google Colab Pro to train the models with the Nvidia T4 GPU

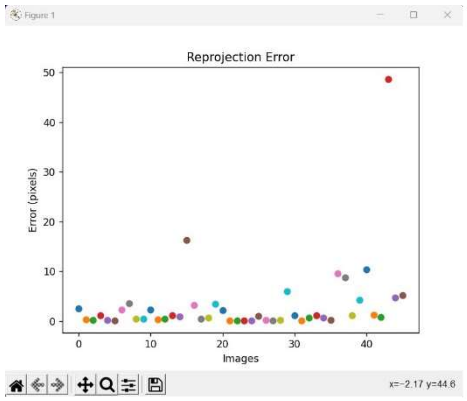
Results

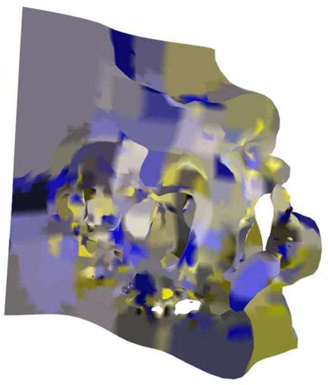
Structure from Motion (SfM)

- Middlebury MVS Dino Results using 48 Images.
- https://vision.middlebury.edu/mview/data/



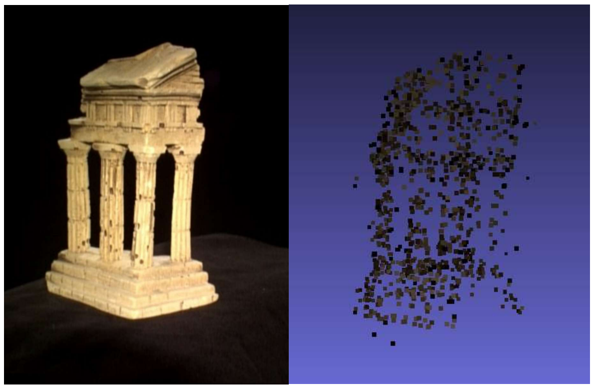
    - Left is ground truth model and right is the reconstruction point cloud



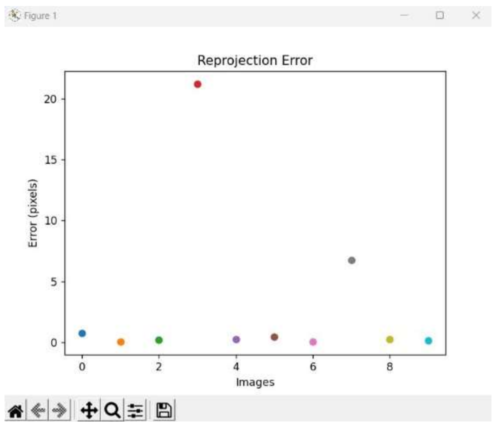


- Left is reprojection error after each image and right is mesh creation using Open3D.

- Middlebury MVS Temple Results using 12 images.



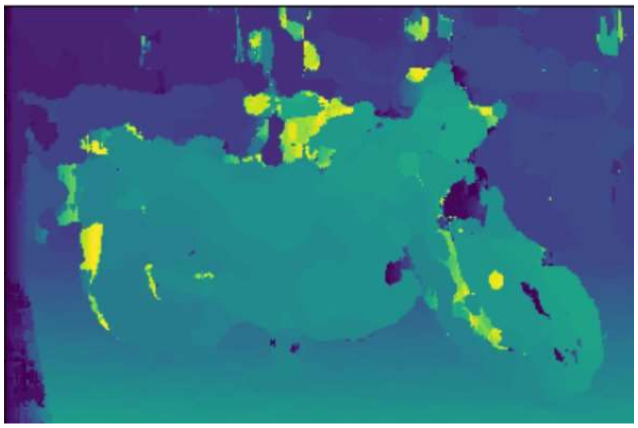
  - Left is ground truth model and right is the reconstruction point cloud



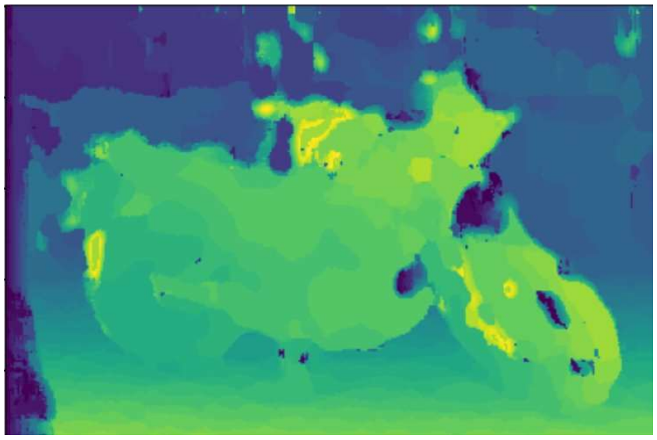
  - Left is reprojection error after each image and right is mesh creation using Open3D.

Reconstruction with Disparity (RwD)

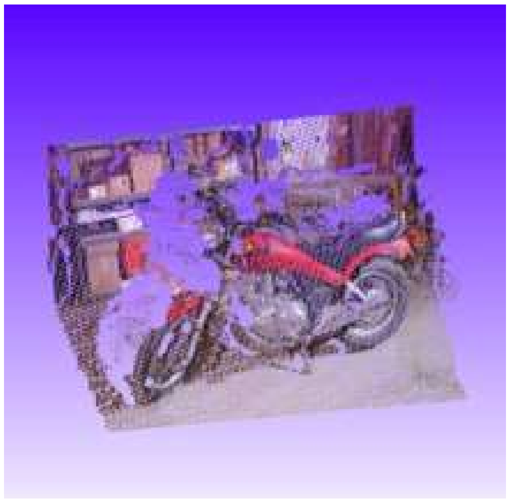
- Original disparity map output by calculating disparity using the sum of absolute differences method.



- Disparity map after averaging filter has been applied



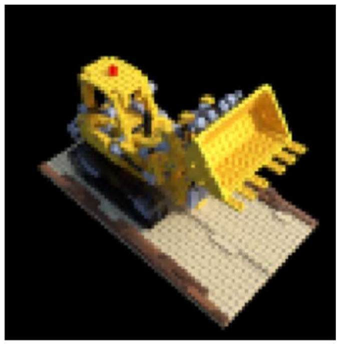
- Output of 3D reconstructed point cloud



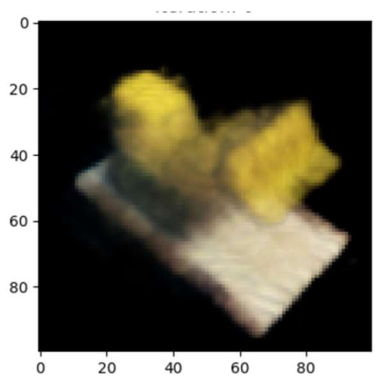
- From qualitative results, it was seen that the 3d point cloud was able to create plots with varying depth, however, this can be improved by smoothing the difference between background and object

Neural Radiance Fields (NeRF and NeRF--)

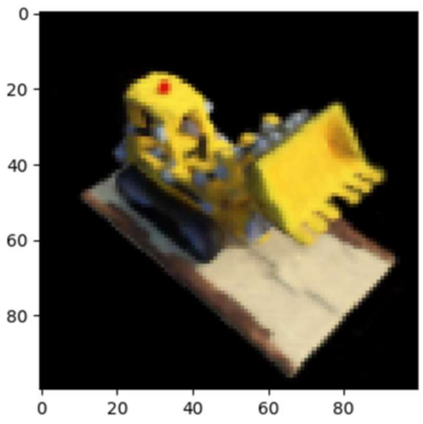
- Original view of the object as seen on the testing data

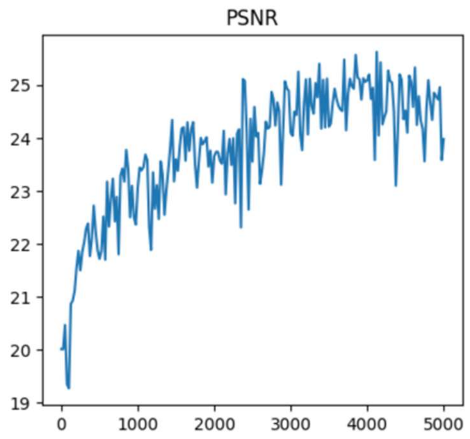


- Reconstruction of the object using the NeRF model from the 100th training iteration



- Reconstruction of the object using the final converged NeRF model and graph of training loss over epochs

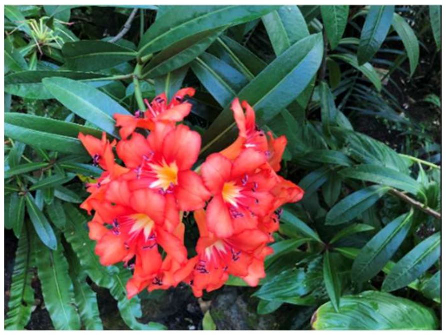


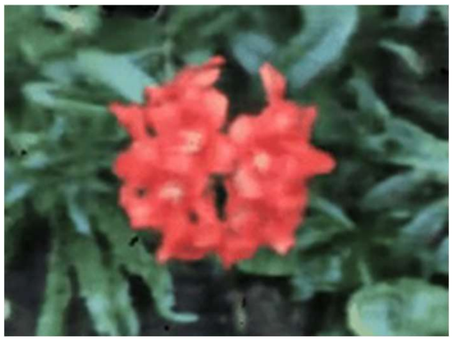


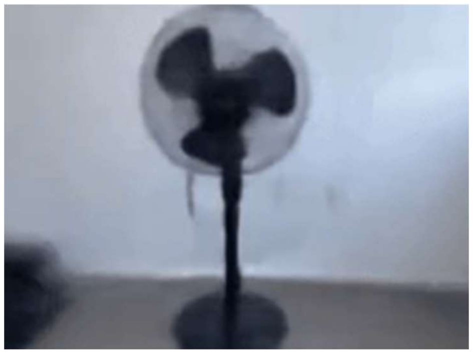
- Novel view synthesis of pretrained NeRF— with data with intrinsics and extrinsics similar to the data it was trained on (Word does not support GIFs so refer to PowerPoint presentation for a better visualization of the results)

Original data

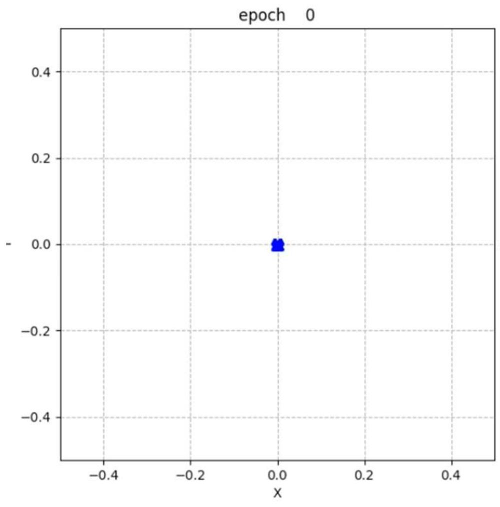
Novel View Synthesis of NeRF--

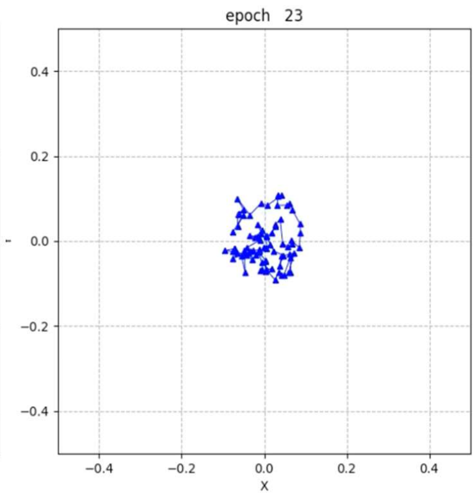


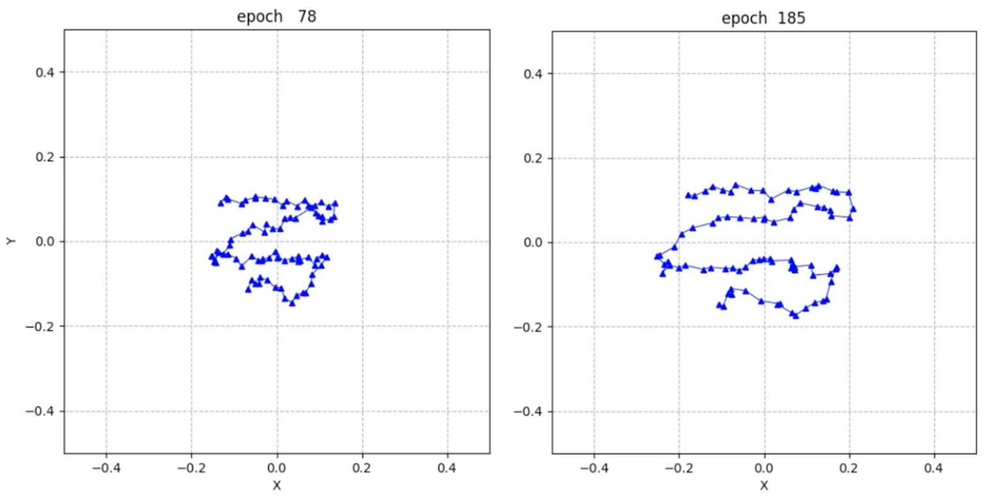




Evolution of learned extrinsics/trajectory over epochs







- Final evaluation metrics for NeRF and NeRF—on the same testing data:

Model	**PSNR**	**PSNR Standard Deviation**	**SSIM**	**SSIM Standard Deviation**
Our NeRF	25.92	1.33	0.91	0.02
Our NeRF --	20.22	0.89	0.60	0.01
Original NeRF	31.01	-	0.95	-
Original	22.48	-	0.63	-

Discussion

Summary

- SfM
    - Purely geometric method/ classical computer vision
    - Simple pipeline: Feature matching, pose estimation, triangulation, reprojection error
    - Evaluate model using reprojection error and mesh model accuracy
    - Used OpenCV and MVS data set to construct models
    - Had relatively good sparse reconstructions but poor mesh models
- RwD

    - Requires stereo data
    - Given well defined intrinsic and extrinsic parameters, it can perform 3D reconstruction fast and well
    - Needs lots of tuning and filtering to get good disparity maps, which also leads to good 3D reconstruction results
- NeRF
    - Uses classical CV methods to get camera parameters (COLMAP) and sparse point cloud of the object
    - Hierarchically sample points from the point cloud and uses the plenoptic function to project rays back on to the images
    - The model learns the plenoptic function and the training is done using PSNR loss
    - The model is evaluated using PSNR and SSIM
    - It only works if the data is in an extremely specific file format and structure
- NeRF—
    - Is doing joint optimization/learning of the intrinsics, the extrinsics as well as the plenoptic function
    - All it needs are the raw images of the object to be reconstructed
    - The pretrained model we worked with was only trained on the LLFF NeRF dataset, which only contains forward facing scenes of objects

Results Comparison

- SfM
    - Pros
        - Creates good sparse reconstructions
        - Relatively efficient, fast computation
        - Does not require too much data, as little as 10 images can work
    - Cons
        - Requires sequenced data
        - Requires tuning parameters and filtering outputs of erroneous points
        - Needs intrinsic parameters
        - Does not make very good mesh models from sparse reconstruction using modern mesh model creation software (needs orders of magnitude more points for good mesh model)
- RwD
    - To use this method, stereo camera image data is necessary
    - Works well only when the intrinsic and extrinsic parameters are well defined

    - When filters and algorithm parameters are tuned well, can output great results, however, takes time and multiple experimental iterations to find good tuning conditions
- NeRF
    - Remarkable results but has an exceptionally long training and inference time
    - Requires the data to be in a specific file format and structure
    - Requires a great amount of data for training
- NeRF --
    - Reasonable results for data that has extrinsics similar to what the training data did (so mainly forward - facing scenes)
    - Does not do well with 360-degree reconstructions
    - Requires an even greater amount of data than NeRF since it is learning a greater amount of parameters
    - The extrinsics it ends up learning end up being very similar/close to what COLMAP calculates, so I think it makes more sense to use COLMAP to get those parameters, which is what the original NeRF does. That way, you get just as precise results in less than a second
    -

Future Works

- SfM
    - Implement bundle adjustment
    - Improve point cloud and mesh construction
    - Find a ground truth point cloud for quantitative analysis
- RwD
    - Better disparity map methods
    - Better reconstruction for the “gaps” or difference of depth smoothness between background and object in 3D point cloud can be improved
    - Explore more methods for MVS disparity map creation
- NeRF/NeRF—
    - Train the models on better quality images
    - Add the positional encoding “trick” to the NeRF model, which helps with getting higher precision results
    - Try training our own NeRF – but with data that includes 360-degree reconstructions

Conclusion

- Given current results, we have determined 4 different 3D reconstruction methods that excel in their own specific use cases.
- SfM is an efficient model if you have video or sequenced data and have a way of turning a sparse point cloud to a reliable mesh.
- RwD is good if you have stereo data and if the intrinsic and extrinsic parameters are well defined.
- NeRF gives great results but it does require specific computational resources, like a GPU. You could use a pretrained NeRF, but even its inference time is quite long. To get great results, you also need a large number of pictures of the object from all possible perspectives.
- NeRF— is an interesting model, which was able to prove that camera intrinsics and extrinsics are learnable parameters. However, it is not able to outperform COLMAP in calculating this parameters in terms of precision. In addition to this, COLMAP yields results in under a second while NeRF – takes a considerable number of epochs to train. So, I believe it is a better choice to use COLMAP to get intrinsics and extrinsics, so that only the plenoptic function, which we don't have a closed-form solution for, is what needs to be learned.
- Depending on the use case, each of these methods has their benefits and they can each be further enhanced to improve results and generalize them better.