Novel View Synthesis of Transparent Objects using NeRFs

Bakari Hassan Leron Julian Anqi Yang
{bhassan, ljulian, anqiy1}@andrew.cmu.edu

1. Introduction & Background

Capturing and rendering a 3D photo that can be viewed from any viewpoint has drawn increasing interest in both the vision and graphics communities. While storing continuous viewpoints of a 3D scene is infeasible, capturing sparsely sampled viewpoints and synthesizing new viewpoints in between in real time becomes an intriguing solution. In this project, we are specifically interested in a scene with transparent objects. Given two set of 2D images capturing transparent objects and its corresponding background from different directions, our goal is to reconstruct shape of transparent objects, which can be used to render new viewpoints that are both photorealistic and faithful to the real world geometry.

Traditional view synthesis methods can be categorized into two types — geometry methods and image-based rendering [4]. The first class uses N-view geometry to reconstruct the geometry and appearance of a 3D scene and renders new views through camera projection. But reconstructing 3D scenes through reprojection is generally hard in real world scenarios. The second class of methods generate new views through compositing neighboring views without explicitly reconstruct the 3D scene. These methods generates artifacts around the fine texture in the images. More recently, Neural Radiance Field (NeRF) [5] sparkles a new stream of works that achieve astonishing results. They propose to implicitly represent 3D scenes as a mapping function that maps viewing direction and volume coordinates to volume density and colors, and then use classic volume rendering to synthesize a new image. Code is provided online: https://github.com/leronjulian/16822_Project

2. Motivation

High quality 3D scans and novel view synthesis have various modern applications in modeling technology. The main advantage is that the image of the object can be rendered as if viewed from a different camera location and lighting position. As a result, these renders can be used not only to model 3D scenes but for depth maps, mixed-reality applications, creating 3D meshes, and capturing a 360° scene with real data. However, there are limitations to current methods such as capturing transparent materials, enforcing geometrically accurate image transformations, and the numerous amounts of images needed for these 3D models. As a result, the impact of our project is proposing various solutions and improvements to the traditional NeRF.

3. Prior Works

NeRF, as stated before, is a current state of the art method for novel view synthesis that optimizes a radiance field represented as a mulit-layer perceptron (MLP) using sparse images. The MLP estimates radiance at 3D voxels by taking in as input - the 3D location and viewing direction of the camera and estimates the color and density of a pixel location. NeRF works very well at reconstructing complex scenes and objects with high fidelity mainly when these objects are non-lambertion, diffuse surfaces. NeRF fails when reconstructing transparent objects due to the fact that when NeRF casts a ray from the camera through a pixel onto the scene, it does not take into consideration the properties of the transparent object such as refraction. This becomes the basis of our project.

Recently, there have been numerous methods to improve on the limitations of NeRF. Using NeRF to capture non-rigid scenes such as using a front-facing cellphone camera to use "selfies" to reconstruct the human face will fail due to the non-rigid motion of humans between image captures along with the fine detail such as hair which are hard to reconstruct using NeRF. Deformable Neural Radiance Fields or "Nerfies" [6] improves reconstruction of non-rigid scenes by estimating a volumetric deformation field that warps scene points to the canonical 5D NeRF.

To use NeRF to reconstruct transparent objects, Dex-NeRF [2] uses a transparency aware method that searches the first sample along a ray for which a value σ is greater than some threshold to compute a depth map for the scene. To assist in representing the transparent objects, light is added to the scene to create more specular reflections on the transparent object due to the fact that NeRF does well at handling view-dependent light synthesis. This allows Dex-NeRF to capture depth of almost all transparent objects better than vanilla NeRF.

Opposed to using NeRF, Neural 3D Reconstruction of Transparent Shapes [3] uses a physically motivated deep network to reconstruct transparent shapes from a few unconstrained images that yields high quality 3D reconstruc-

tions that closely match scanned ground truth images.

4. Proposed Solutions

This leads us into our proposed solutions to improve traditional NeRF for novel view synthesis of transparent objects.

4.1. Shape from distortion

We propose shape from distortion by initially using a synthetic dataset consisting of background and distorted images. For this, we use the rendering program Blender to create a synthetic dataset of 2 main image categories. The first is transparent objects that are placed on a scene with some background which has some distortion in the transparent regions due to refraction. Then, we have that same background with the object not in the scene. Using a deepnet, we train the model with the scene inputs and output the depth-map for that scene. For the model we use a simple UNet.

Sample images are shown in 1. For each captured frame, the transparent objects are rotated or shifting randomly throughout the scene to add variation when training the model. The model is trained on a total of 5 unique objects and 10 unique backgrounds for 339 frames. Therefore, in total our training dataset includes 1695 images. At testing time, a synthetic or real image of a scene that includes the transparent object is captured, along with the object removed. The model then outputs the predicted depth map used for shape refinement.

4.2. Shape refinement

Given a set of depth maps corresponding to camera poses, the 3D scene can be reconstructed using non-linear least squares. Here, the point-to-plane iterative closest point (ICP) method is used to register the point clouds. Standard ICP begins with a reference (stationary) point cloud, and a template point cloud and iteratively solves for a similarity transformation that minimizes the sum of distances between the nearest neighbors of all points in the template cloud and the reference cloud. Standard ICP works well for fully overlapping point clouds, but some modifications were made for this case wherein clouds will only overlap partially when registered. Optimizing over all points favors solutions that colocate the cloud's centroids. We use a modified version of ICP via 1) a distance threshold for nearest neighbor section [7], and 2) a point-to-plane objective modifier to exploit point cloud structure [1].

The maximum distance threshold d_{max} excludes nearest neighbors from the objective cost that exceed the threshold. This encourages registration of small-scale cloud features and limits the magnitudes of similarity transformations. Choosing d_{max} has a strong effect on convergence rates and solutions. To avoid tuning this parameter, our implementation chooses d_{max} such that 25% of template cloud points are included in each optimization loop. The point-to-plane variant attempts to align planar surfaces by computing the inner products of errors with local normal vectors of the reference cloud. This is helpful when the scene has well-defined structure. The algorithm we used is shown in Algorithm 2 where the Levenberg–Marquardt algorithm was used in conjunction with the exponential map to constrain T to valid similarity transformations.

Algorithm 1 Point Cloud Registration

Require: N depth maps $\mathcal{D} = \{D^{(1)}, ..., D^{(N)}\}$ rendered using the learned neural representation.
input: Two depth maps: D, D' Initial transformation $T \in \mathbb{R}^{4\times4}$
output: Transformation T which aligns two point clouds C, C'.
$\mathcal{C} = \{C^{(1)}, ..., C^{(N)}\} \leftarrow$ `Backproject`$(\mathcal{D})$
Choose distance threshold d_{max} used to identify outliers when computing nearest neighbor distances between point clouds.
while not converged **do**
  for $j \leftarrow 1$**to**N **do**
    $m_j \leftarrow$ `NearestNeighborInC`$(T \cdot C')$
    if $\|m_j - T \cdot c'_j\|_2 < d_{max}$ **then**
      $w_j \leftarrow 1$
    else
      $w_j \leftarrow 0$
    end if
  end for
  $T \leftarrow \arg\min_T \sum_j w_j \|n_j \cdot (m_j - T \cdot c'_j)\|_2^2$
end while

where n_j is the surface normal at the point m_j.

4.3. Virtual camera alignment using NeRFs

In real world scenario, we need to obtain a pair of background and object images of the same camera pose and obtain multiple pairs from different viewpoints. In this project, we fix the camera poses for the object scene, and compute corresponding backgrounds through virtual cameras.

Specifically, We capture N viewpoints of a background scene $\mathcal{I}_B = \{I_B^{(1)}, .., I_B^{(N)}\}$ and M viewpoints of transparent objects with the same background $\mathcal{I}_O = \{I_O^{(1)}, .., I_O^{(N)}\}$ with the same camera. Note that the poses in two sequences are chosen freely and are not the same. We keep $\mathcal{I}_O$ as reference viewpoints and generate virtual background viewpoints $\hat{\mathcal{I}}_B$. This is achieved by training a NeRF $\mathcal{H}_B(\cdot)$ on background scene using $\mathcal{I}_B$, and querying specific viewpoints from $\mathcal{H}_B(\cdot)$.

One caveat of this approach is that camera poses of the two scenes are estimated separately and do not share the

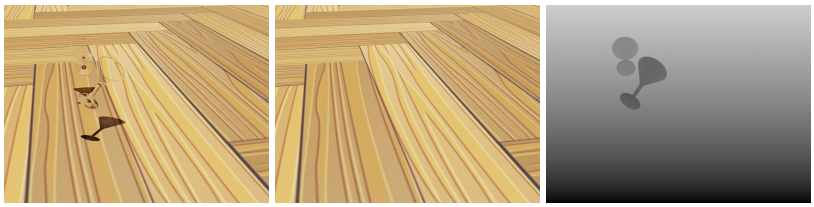

Figure 1. **Sample synthetic images.** Left-to-right: Transparent object with background. Background only without transparent object. Associated ground truth depth map.

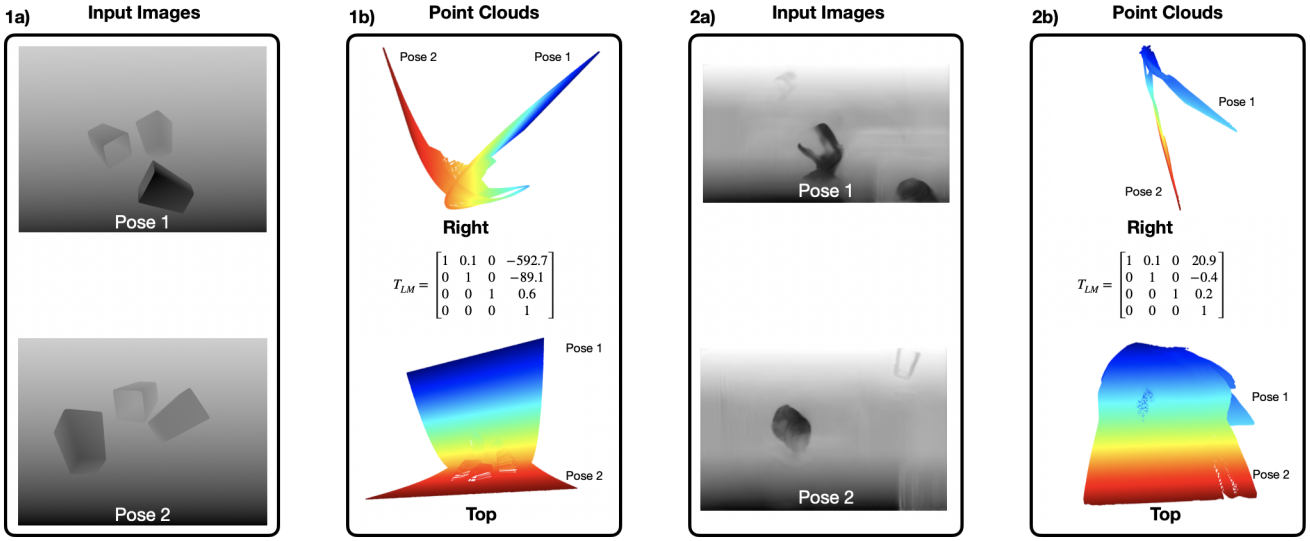

Figure 2. 1a,2a: Input images with learned transformation using the Levenberg–Marquardt algorithm for ideal, synthetic and learned depth maps respectively; 1b,2b: registered point clouds. Mesh registration for two poses from the synthetic image dataset shows the ICP algorithm attempts to align the cups for 25th precentile of points' nearest neighbors. However, background depth differences are too large for them to influence the solution. The algorithm performs worse on learned depth maps due to reduced quality and artifacts that appear at shallow depths.

same world coordinate, thus it is necessary for us to register world coordinates beforehand. The algorithm is summarized in Table 2. We first estimate camera parameters and sparse 3D points using SfM package COLMAP. As is shown in the left column in Figure 3, it gives us camera intrinsic matrix K, two sets of camera poses $[R_B^{(i)}|t_B^{(i)}], i = 1...N$ and $[R_B^{(j)}|t_B^{(j)}], j = 1...M$, and two sets of sparse 3D points P_B, P_O. Since two scenes share the same background, we can find a subset of 3D point correspondences $Q_B \in P_B, Q_O \in P_O$ and register them. To find 3D point correspondence, we compute 2D point correspondence using SIFT from two images and track 3D points that projects to these 2D points. Once we have Q_B, Q_O, we estimate a transformation R, t such that $Q_B = R * Q_O + t$ as illustrated in right column in Figure 3. The transformation can be solved by minimizing a least square,

$$\min \sum_{i=1}^{n} ||Q_B^{(i)} - R * Q_O^{(i)} - t||^2.$$

Rotation and translation can be computed by the following,

$$F = (Q_B - \bar{Q_B})(Q_O - \bar{Q_O})^T$$
$$F = U\Sigma V^T$$
$$R = VU^T$$
$$t = \bar{Q_B} - R * \bar{Q_O}$$

Next we apply the estimated transformation R, t to points and cameras in the object scene so that they are under the same world coordinate as the background scene.

$$P_O' = RP_O + t$$
$$M_O^{(i)'} = M_O^{(i)} \begin{bmatrix} R & t \\ \mathbf{0} & 1 \end{bmatrix}^{-1}$$

We use the registered camera projection matrices $M_O^{(i)'}$ to query background NeRF $\mathcal{H}_B$ and obtain N virtual background views $\mathcal{I}_B'$. $\mathcal{I}_B', \mathcal{I}_O$ form N pairs of background/object images that can be used for shape reconstruction as specified in Section 4.1 and 4.2. Figure 4 shows five

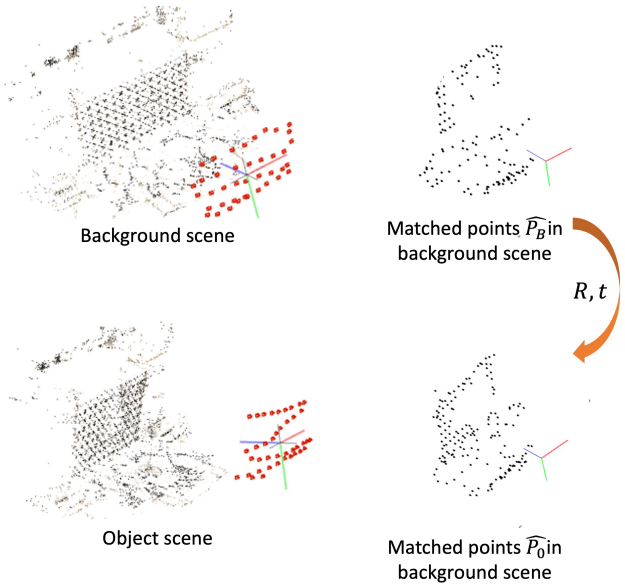

Figure 3. **Procedure of registering world coordinates of two scenes.** Left column shows COLMAP estimation results. Right column shows a set of point correspondences that need to be aligned by R, t.

registered image pairs from a checkboard background with glass cup.

Algorithm 2 World coordinate registration

Require: N view images $\mathcal{I}_B = \{I_B^{(1)}, ..., I_B^{(N)}\}$ capturing the background scene, and M view images $\mathcal{I}_O = \{I_O^{(1)}, ..., I_O^{(N)}\}$ capturing transparent objects with the same background.

Ensure: Both scene is captured by the same camera with the same intrinsic matrix K.

1. Use shape from motion to recover camera intrinsics, camera poses, and a sparse set of 3D points for both set of images. $K, [R_B|t_B], [R_O|t_O], P_B, P_O$.
2. Find point correspondences between two sets of 3D points Q_B, Q_O.
3. Estimating rotation and translation between them $Q_B = R * Q_O + t$.
4. Apply transformation R, t to points and camera poses in the object scene, so that the object scene share the same world coordinate as the background scene.

5. Results

Single-view depth estimation. Figure 5 presents sample images of predicted depth maps given a scene and its associated background. As the images display, our network does an adequate job at producing depth maps for transparent objects given unseen images. We attribute adequate results due to the small amount of training data fed into our model and a lack of complexity on the model and inputs.

Virtual camera alignment. Figure 4 showcase five real-captured transparent objects and corresponding background synthesized by virtual cameras. The alignment is reasonably robust. Figure 6 presents three pairs of real-captured cup on the floor, synthesized pure background images, and depth prediction. Virtual background is observing from the same camera rotation as in reference frame, but the translation estimation is offset by a small amount due to the textureless background. The depth prediction on real-captured images are less desired compared to simulated ones. This can be due to the discrepancies between simulated data and captured data, such as noise distribution, reflective properties of material, shape of transparent object, and etc. These discrepancies can be handled with more photorealistic simulation and a marginally larger dataset. Due to the limited time of a course project, we would like to list these modifications as future plans.

6. Challenges

Single-view depth estimation. For single-view depth estimation, the main challenge that we faced was producing concise accurate depth maps. We attribute this to the fact that our model was too simple for this task. We did not add any regularization or modifications to the inputs or models for more accurate depth prediction.

7. Conclusion

The goal of this project is to reconstruct shape of transparent objects. Our key insight is that a transparent object refracts light and reveals its shape by distorting the background. We propose a multiview reconstruction algorithm by leveraging single-view depth prediction from a background/object image pair and bundle adjusting estimation among multiple viewpoints. We also propose a camera registration method by synthesizing virtual viewpoints from NeRF for background scene. We achieve reasonable performance on simulated data.

References

[1] Y. Chen and G. Medioni. Object modelling by registration of multiple range images. *Image and vision computing*, 10(3):145–155, 1992. 2

[2] J. Ichnowski*, Y. Avigal*, J. Kerr, and K. Goldberg. DexNeRF: Using a neural radiance field to grasp transparent objects. In *Conference on Robot Learning (CoRL)*, 2020. 1

[3] Z. Li, Y.-Y. Yeh, and M. Chandraker. Through the looking glass: Neural 3d reconstruction of transparent shapes. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 1262–1271, 2020. 1

[4] B. Mildenhall, P. P. Srinivasan, R. Ortiz-Cayon, N. K. Kalantari, R. Ramamoorthi, R. Ng, and A. Kar. Local light field

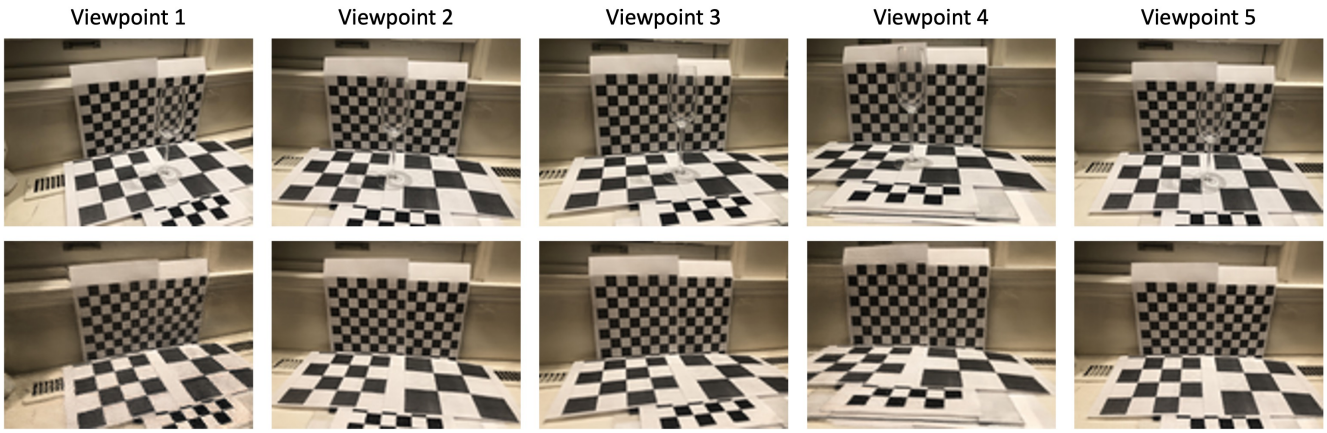

Figure 4. **Samples of real-captured background/object images.** Top row shows object images captured from four different viewpoints. Bottom row shows background images generated by NeRF with corresponding camera poses.

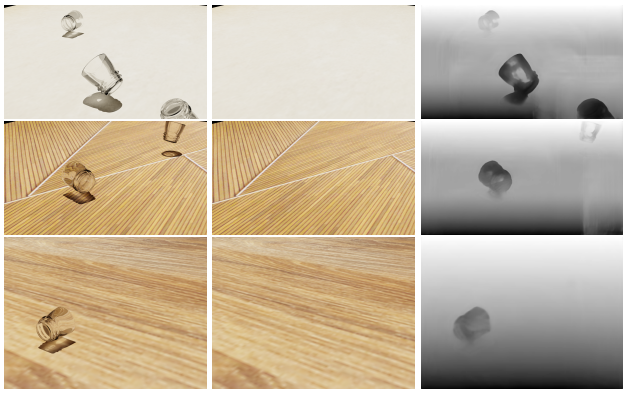

Figure 5. **Predicted Depth maps** Left-to-right: Transparent object with background. Background only without transparent object. Predicted depth map.

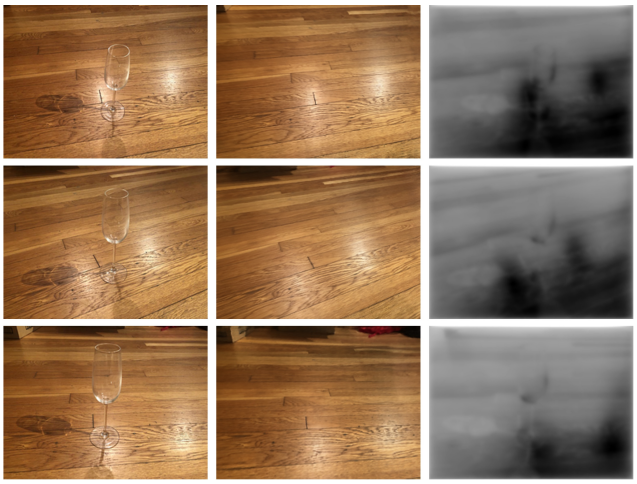

Figure 6. **Virtual camera registration and depth prediction.** From-left-to-right: real-captured glass cup with background, background generated by NeRF, and predicted depth map.

fusion: Practical view synthesis with prescriptive sampling guidelines. *TOG*, vol.38(4):pp.1–14, 2019. 1

[5] 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. In *ECCV*, 2020. 1

[6] K. Park, U. Sinha, J. T. Barron, S. Bouaziz, D. B. Goldman, S. M. Seitz, and R. Martin-Brualla. Nerfies: Deformable neural radiance fields. *ICCV*, 2021. 1

[7] A. Segal, D. Haehnel, and S. Thrun. Generalized-icp. In *Robotics: science and systems*, volume 2, page 435. Seattle, WA, 2009. 2