Consistent Depth Prediction for Transparent Object Reconstruction from RGB-D Camera

Yuxiang Cai Yifan Zhu Haiwei Zhang Bo Ren [*]
Nankai University
{caiyuxiang, zhuyifan}@mail.nankai.edu.cn, {zhhaiwei, rb}@nankai.edu.cn

Abstract

Transparent objects are commonly seen in indoor scenes but are hard to estimate. Currently, commercial depth cameras face difficulties in estimating the depth of transparent objects due to the light reflection and refraction on their surface. As a result, they tend to make a noisy and incorrect depth value for transparent objects. These incorrect depth data make the traditional RGB-D SLAM method fails in reconstructing the scenes that contain transparent objects. An exact depth value of the transparent object is required to restore in advance and it is essential that the depth value of the transparent object must keep consistent in different views, or the reconstruction result will be distorted. Previous depth prediction methods of transparent objects can restore these missing depth values but none of them can provide a good result in reconstruction due to the inconsistency prediction. In this work, we propose a real-time reconstruction method using a novel stereo-based depth prediction network to keep the consistency of depth prediction in a sequence of images. Because there is no video dataset about transparent objects currently to train our model, we construct a synthetic RGB-D video dataset with different transparent objects. Moreover, to test generalization capability, we capture video from real scenes using the RealSense D435i RGB-D camera. We compare the metrics on our dataset and SLAM reconstruction results in both synthetic scenes and real scenes with the previous methods. Experiments show our significant improvement in accuracy on depth prediction and scene reconstruction.

1. Introduction

Transparent objects frequently appear in our daily life, including glasses, goblets, plastic objects, vases, etc. Because of the nature of transparent materials, the light will reflect and refract on the surface of transparent objects, which will cause errors for the commonly used depth camera. Fig. 2 show this ambiguity: the depth captured by the camera is the distorted result of the object behind.

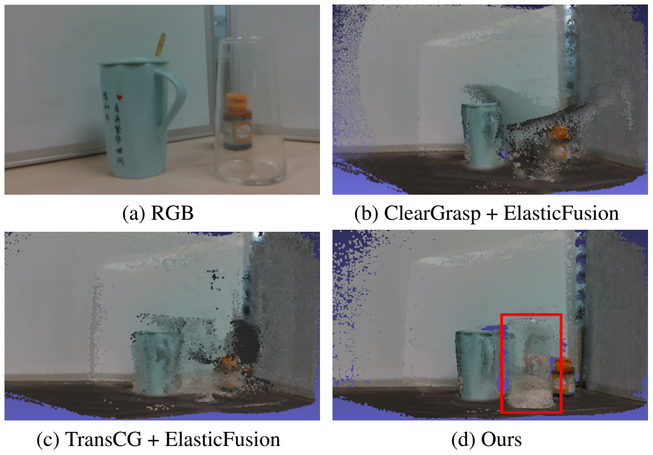
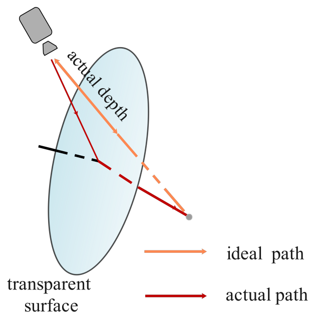


Figure 1. **RGB-D reconstruction result comparison in the real scene.** Subfigures (b) and (c) show the reconstruction result using ElasticFusion [38] with predicted depth data from the state-of-the-art transparent object depth prediction method: ClearGrasp [31] and TransCG [8]. Subfigure (d) shows our reconstruction result. In (b) and (c), the transparent object can not be well reconstructed. Compared with them, our model performs well in reconstruction.

SLAM(Simultaneous localization and mapping) is a popular topic in computer vision and robotic fields. There are lots of SLAM methods [7, 25, 38] that take RGB-D as their input to reconstruct the scene and recover the motion of the camera. The RealSense [19] and Kinect [12] are often used to capture the RGB-D data and are common input devices for RGB-D SLAM. However, due to the reason we state above, these cameras can not estimate the depth of transparent objects. So, restoring the correct depth of transparent objects is critical in reconstruction.

Many depth completion methods for indoor scenes train and test in the TUM dataset [34], which do not contain transparent objects. Also, there are many attempts [8, 31, 44] to try to predict the depth of transparent objects using different deep neural network models. However, these monocular-based methods perform poorly when using their predicted depth in

[*]Bo Ren is the corresponding author



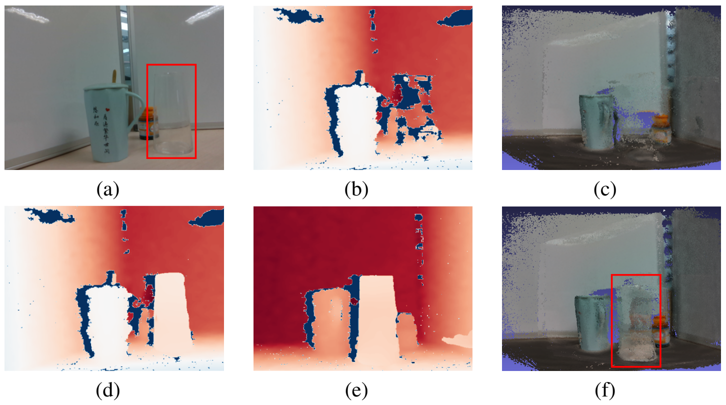


Figure 2. **Effect of the transparent object.** The left figure shows that the ray path changes when there is a transparent object, which causes inaccurate depth data. In (a), a red box shows where the transparent object is located. (b) shows the depth data from the depth camera. The region around the transparent object is noisy and incorrect and (c) shows the reconstruction result using these depth data. The transparent object is missing. (d) and (e) are our depth prediction results of the transparent object in different views. (f) is our reconstruction result.

the SLAM reconstruction. We attribute it to the depth inconsistency problem between frames. Estimating the consistent depth that can be used for SLAM is a challenge. Previous work like [3, 9, 23, 35] has mentioned the similar problem. Since depth between frames is strictly constrained by camera trajectory, monocular depth estimation can not ensure depth predicted under the same absolute scale. So instead of using a single frame to predict the depth of the transparent object, we choose multiple views as our input and build a connection between adjacent RGB images to keep consistency in prediction.

In this paper, we propose a stereo-based transparent depth prediction model based on DPSNet [17] to keep consistency in prediction, which can be directly used for the SLAM reconstruction of transparent objects. Our model uses the plane sweep stereo module to build a spatial consistency restriction between sequential input images. To obtain scale information on the more-accurate background depth, we separate the transparent object using the predicted mask and transform it to point clouds. We use a designed lite PointNet based on [27, 28] to learn this information and combine it into our model. We further design another surface normal prediction branch to assist and refine our depth prediction result. The detail of our model can be found in Sec. 3. To make a real-time reconstruction of the transparent object, we embed our network into ElasticFusion [38] as our pseudo-SLAM method in Sec. 4.

There are many datasets for the transparent object, such as ClearGrasp dataset [31], TransCG dataset [8], and Omniverse Object dataset [44]. However, all of them contain only single image segmentation data and do not have video segmentation data of transparent objects. In [45], a natural scene video dataset for the transparent object is proposed but needs the ground truth depth value of transparent objects for model training. So we use Blender to create a synthetic dataset for our depth prediction model. Besides, we use the RGB-D camera to capture real scene data to evaluate the generalization capability of our model and test the 3D reconstruction of these scenes. We also test the reconstruction result using the previous depth prediction method of the transparent object in our pseudo-SLAM pipeline. Fig. 1 shows the result comparison. Besides this figure, a variety of experiments in Sec. 5 demonstrate that our proposed method makes a significant improvement.

In summary, our contributions are:

- We propose a real-time 3D transparent object reconstruction method without any prior knowledge requirement of the scene using a stereo-based depth prediction model.
- We construct a new RGB-D dataset of transparent objects with ground truth depth data for model training to predict the depth of transparent objects in continuous RGB-D data sequences.

2. Related Work

2.1. Depth Prediction From RGB Images

Many prior works use the neural network to predict dense depth from RGB images. Earlier methods such as Eigen *et al.* [6] used convolution neural networks to learn the global structure of the scene from a single RGB image and predict depth. Other methods [15, 21, 32, 41] take advantage of the latest network structure to predict more accurate depth. Qi *et al.* [29] propose a depth-to-normal structure and a normal-to-depth structure to iteratively refine the prediction. However, monocular depth prediction suffers from the problem of scale consistency and depth consistency. Many works [3, 10, 11, 33, 46] train their model use the multi-view consistency loss to resolve it.

The stereo-based method is also popular for depth prediction tasks. These methods predict the depth map using two views or multiple views. Inspired by the traditional stereo matching techniques, [13, 17, 18, 42, 43] use the neural network to replace some processing progress to get an estimated disparity or depth. [5, 6, 21, 37] propose refinement modules to make a progressive reconstruction. However, none of them can predict the depth of transparent objects.

2.2. Depth Prediction of Transparent Object

Transparent objects usually have refractive and reflective properties, which cause errors and inaccuracy in obtaining depth data from RGB-D cameras. It is also a challenge for depth completion in scenes that contain transparent objects. Previously work focused on using the trained network to fix up the depth of transparent objects. Sajjan *et al.* [31] uses three models to predict mask, surface normal, and occlusion boundary separately, followed by a global optimization function to get the result. Zhu *et al.* [44] proposes that the depth of the transparent object can be inferred from the opaque object nearby using a trained local implicit depth function(LIDF). Fang *et al.* [8] uses an encoder-decoder model to predict depth using RGB and depth. Xu *et al.* [40] first restores the point cloud and then combines the point cloud with the RGB image to produce a complete depth map. However, the remaining point clouds are often too lacking and noisy to restore a complete shape. These methods perform well in single images but poorly in transparent object reconstruction. By contrast, our prediction method gets suitable depth data between frames that perform well in the reconstruction task.

2.3. Transparent Object Reconstruction

It is also a challenge for transparent object reconstruction, and there is no previous work on real-time reconstruction using end-to-end depth prediction. [22, 26, 30] draw support from additional known information and pre-computed environment information. Zhu *et al.* [45] reconstruct the scene and transparency respectively, which is time-consuming. Alt *et al.* [2] propose a method to reconstruct via depth camera. In their method, the background must be first estimated using a video sequence that excludes transparent objects. Unlike these existing methods, we introduce a stereo-based model for depth prediction and build a real-time transparent reconstruction method without any prior knowledge requirement of the scene.

2.4. Point-based 3D Learning

Traditional convolutional neural networks can not be directly applied in point clouds due to their unordered attribute, which means the point of objects can be ordered as any combination of sequences. Pioneered by PointNet [27], several works like [1] use the MLP layer for processing the point cloud. Because the MLP module lacks the capacity to learn geometric relationships between 3D points, [28, 36] propose a hierarchical architecture to learn from 3D geometric structures. However, MLP-based models, as discussed in [39], do not consider the connection and neighbouring relationship with other points as they usually use the max pooling function to get together the features of each point. Moreover, Mao *et al.* [24] propose that simply applying MLP network to point clouds may not always work in practice as we considered. [14, 20, 39] divide the space into 3D voxels or grids so that the points can be regularly arranged. But the voxelization operation leads to an inevitable loss of detailed geometric information. For our task, the original trusted point of the transparent objects is extremely sparse. It is hard to learn information using these points. Besides, the voxel-based or grid-based method is too complicated and slows in practice with a large resolution, so we choose the MLP-based PointNet and lightweight it to learn information from background points around the transparent object.

3. Transparent Object Prediction for SLAM Reconstruction

In Sec. 1, we have discussed that the consistent depth value in different views is essential for the RGB-D reconstruction of transparent objects. In order to make a consistent depth prediction for the transparent object, we take advantage of the spatial consistency of the RGB information between sequential input images and the scale information of the more-accurate background depth. We first obtain a cost volume from the plane sweep stereo method for the RGB channels using sequential images that contain transparent objects. The detail of this step can be found in Sec. 3.1. Then we aggregate it with a point feature that is extracted from the background depth of the current frame(i.e. depth out of the masked transparent region). The detail can be found in Sec. 3.2. Then a consistent transparent object depth can be reconstructed from a depth regression module. Inspired by [31], we further design another surface normal prediction branch parallel to the depth prediction branch in our network, which helps to provide stronger visual cues to assist the accuracy of depth prediction. We only use this module in the training process and disable this module in the inference process. The overview of our method is shown in Fig. 3.

3.1. Plane Sweep Stereo for Consistency Constraints

The plane sweep stereo proposed by [16] proposes to divide the depth space D into L average parts. The baseline of the plane sweep stereo method is: back projecting the image to successive virtual planes divided from the 3D space and measuring the cost on each plane. The costs then form a cost volume. Following [17], we use extracted features from a pyramid convolutional network with the pooling module to represent the image. We use the same equation as their

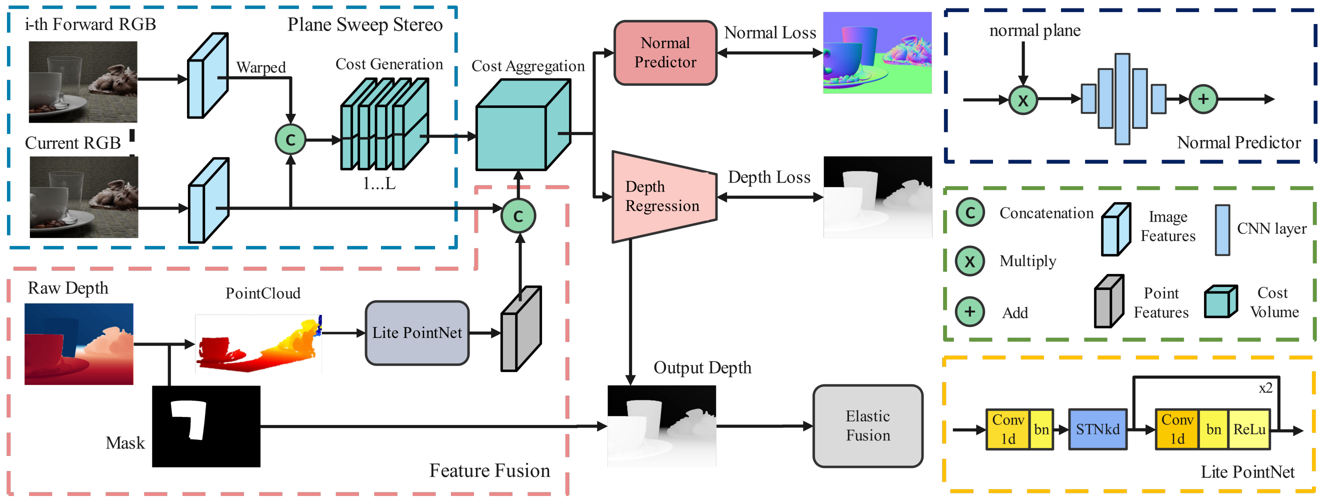


Figure 3. Overview of our proposed method. The model takes at least two RGB, raw depth data of the current frame captured by the depth camera, and a mask as input. The output depth is used in the RGB-D SLAM reconstruction method: ElasticFusion to make a scene construction. We introduce our depth prediction model in Sec. 3 and we introduce our pseudo-SLAM system in Sec. 4, which combinates our model and the ElascticFusion algorithm.

work for the warp function that describes the back-projecting process (1).

$$\tilde{p} \sim K\texttt{exp}(\hat{\xi})[(K^{-1}p)d_l \quad 1]^T \tag{1}$$

where the depth value for each plane l is represented as d_l. $\tilde{p}$ and p are the homogeneous coordinate of pixels in the warped view and current view. K refers to the camera's intrinsic matrix, and $\texttt{exp}(\hat{\xi})$ is the transformation matrix between these views represented in Lie algebra. The warped feature $\tilde{f}$ is the interpolated value of the neighborhood image feature using $\tilde{p}$.

Assumes that the shape of the feature extracted from the image is (B, C, H, W), where B represents the batch size, C represents the channel of the image feature, H, W is one-quarter of the input image height and width. For the later content, these symbols represent the same meaning as above. We combine the warped feature with the current feature at each plane in the second dimension to get a concatenated feature with a shape of $(B, 2C, L, H, W)$. For more than one neighboring image, we compute the concatenated feature between each neighboring image and the current image and take the average of them. Finally, we use the cost generation module in [17] to produce a cost volume with the shape of (B, L, H, W). Since the plane sweep stereo considers the motion between neighboring RGB frames, it brings a relative constraint to the subsequent modules and ensures overall consistency of depth prediction.

3.2. Point Cloud Feature Fusion for Aggregation

Since the plane sweep stereo only considers the relative depth scale, the above cost volume still lacks absolute scale information for the scene geometries to make an accurate prediction. We take advantage of the more-accurate background depth and design a lite point net module to inform the depth prediction network of the scaling information. The background depth is obtained by excluding the mask region of the transparent object in raw depth data. Unlike the previous method that directly combines raw depth data with RGB data to form input with four channels, we transform background depth data to point clouds. It is advantageous that point clouds contain spatial information about the scene, such as the arrangement of objects. We use the equation in (2) to generate point clouds from background depth data.

$$P(u, v) = (K^{-1}p)d \tag{2}$$

Given a pixel coordinate (u, v) and its depth d, p is the homogeneous coordinate of it, $P(\cdot)$ represents its point cloud vector, which can be computed using the camera's intrinsic matrix K. Through 1d convolution and feature transform module proposed in [27, 28], we obtain a matrix with S channel of each point cloud which represents point cloud features that contain spatial information. Inspired by [17], we fuse the point cloud feature with the RGB feature from current images into the above cost volume. The point cloud features are reshaped to (B, S, H, W) and combined with RGB features at the second channel to get the fusion feature with the shape of $(B, S + C, H, W)$. For every plane in cost volume, the cost value $(B, 1, H, W)$ is concatenated with these fusion features at the second dimension. After that, a dilated convolutional context network, like the structure in [17] will accumulate details and reduce the combined cost volume back to the original shape so that it can be fed into the prediction network.

3.3. Depth Prediction

The depth prediction module transforms the cost volume with the shape of (B, L, H, W) to a probability volume with the shape of (B, L, H, W) using the softmax function at the second dimension. The probability volume represents the probability for each pixel laying on each depth plane. Then we regress the final depth result using the method proposed in [18]. As the surface normal of the transparent object can provide stronger visual cues, proposed in [31], we design a normal predictor branch paralleling with the depth prediction branch to further restrains the result of the depth prediction branch. In this branch, we transform the depth plane to the normal plane using the equation in (3).

$$n_i = \langle P_i^w \times P_i^h \rangle; \tag{3}$$

where $P(\cdot)$ represents point cloud vector defined in (2). P^w is defined as $P(u+1, v) - P(u, v)$ and P^h is defined as $P(u, v+1) - P(u, v)$. To accumulate more details in the normal map, we use another five-layer convolutions network to process the normal map before it gets summed (4).

$$n = \sum_{i=0}^{L} \mathbf{CNN}(n_i \cdot \sigma(l_i)) \tag{4}$$

where $\sigma(l_i)$ is the probability value of plane l_i. We train our model using the loss function blow (5):

$$\mathcal{L} = ||\hat{D} - D^*||_H + \alpha||\hat{N} - N^*||_H \tag{5}$$

Where the first term represents the depth loss that penalizes depth inaccuracy. $\hat{D}$ and D^* denote the ground truth depth and the prediction result. Similar to depth loss, the second term penalizes normal inaccuracy. $\hat{N}$ and N^* denote the ground truth normal and our predicted result. $||\cdot||_H$ denotes the Huber norm function. α is the weight parameter. In practice, we find that the depth of transparent objects is concentrated only in a few adjacent planes. If we train our model with the loss function in the mask region of transparent objects, most planes are not well penalized and may lead to wrong predictions in the real scene. In order to learn to regress the accurate depth in the different planes, we predict the whole image but penalize our model with a weighted final loss. We compute the loss function (5) in the masked region of the transparent object and unmasked region. The final loss is defined as (6).

$$Loss = \beta\mathcal{L}_{masked} + (1-\beta)\mathcal{L}_{unmasked} \tag{6}$$

A higher β value means we care more about the region of transparent objects than its background.

Algorithm I: Pseudo-SLAM for transparent object reconstruction

```
Data: Image sequences $I$={$I_1, I_2, \dots, I_m$};
Depth data sequences $D$={$D_1, D_2, \dots, D_m$};
Number of input image $w$ ;
Camera intrinsic matrix $K$ ;
1  initialization $i = 1$ ;
2  repeat
3      predict mask of transparent object $\hat{M}$ using $I_i$ ;
4      exclude mask region and transform $D_i$ to point
         cloud $P_i$ ;
5      extract RGB feature $F_i$ using $I_i$ ;
6      foreach $j = 1, \dots, w-1$ do
7          compute $F_{i+j}$, $P_{i+j}$ same as above ;
8          compute relative motion $R_j$ and $t_j$ on $P_i, P_{i+j}$
             using ICP algorithm ;
9          warp $F_{i+j}$ to $\tilde{F_{i+j}}$ using $R_j, t_j$ ;
10         build aggregated feature $f$ using $\tilde{F_{i+j}}$ and $F_i$ ;
11     end
12     compute the average of $f$ and generate cost volume ;
13     cost aggregation using $F_i$ and extracted feature from
         $P_i$ ;
14     make a prediction of the transparent object and
         renew the original depth $D_i$ ;
15     go through ElasticFusion pipeline;
16     $i \leftarrow i + 1$ ;
17 until $i$ >($m - w + 1$);
```

4. Pseudo-SLAM System

To make a reconstruction of scenes, we embed our depth prediction model with the previous RGB-D SLAM framework: ElasticFusion. We use the prediction network proposed in [45] to get a mask of the transparent object. With this mask, we can strip the error depth data in the real scene to get the point cloud of the background, which can be used in estimating the rotation matrix, translation vector, and aggregation module in our model. The overall algorithm is shown in 17. We use a queue to cache results already computed in the previous steps to improve the efficiency of our pseudo-SLAM system.

5. Experiments

5.1. Data Creation

There are many datasets for the transparent object, such as ClearGrasp dataset [31], TransCG dataset [8], Omiverse Object dataset [44]. None of them contains sequences of images. Zhu *et. al* [45] propose a real scene dataset, which doesn't contain ground truth depth data for training.

Since relevance from different perspectives is key for re-

Method	Metrics					
	RMSE ↓	REL ↓	MAE ↓	$\delta_{1.05}$ ↑	$\delta_{1.10}$ ↑	$\delta_{1.25}$ ↑
ClearGrasp [31]	0.267	0.211	0.237	15.17	25.92	62.72
LIDF [44]	0.571	0.269	0.368	53.64	62.00	68.35
TransCG [8]	0.153	0.089	0.126	42.94	69.19	93.83
Ours	**0.148**	**0.077**	**0.122**	**50.30**	**77.76**	**95.92**

Table 1. **Metric comparison result for transparent object depth prediction.** ↑ means higher is better and ↓ means lower is better. Before evaluation, we re-train the previous network on our RGB-D video dataset using the pre-trained weight provided by them.

straining depth consistency, we choose to create a synthetic video dataset instead of training on existing datasets. Using the open-source model, we build different scenes in Blender. We create 15 scenes for training and 5 scenes for validation. Each scene contains about 300-400 RGB images with ground truth camera trajectories, masks, and depth of the transparent object. Our dataset contains various basic glass objects in daily life: glass cups, glass goblets, wine cups, and square transparent boxes. Further metrics comparison and reconstruction evaluation are made on our dataset to show the advance of our method. To test the generalization capability of our model, we recorded RGB images and depth data by RealSense D435i RGB-D camera at the frame rate of 30 Hz and 640x480 resolution in real scenes. Each scene contains about 400-600 frames and different transparent objects.

5.2. Implementation Details

The model is implemented using the PyTorch framework. The Pseudo-SLAM is implemented using C++ and the Libtorch framework. All experiments run on the Intel i7-8700K CPU and two NVIDIA GeForce GTX 1080Ti GPUs. We set $L = 50$ when initializing our model and $\alpha = 0.5$, $\beta = 0.9$ when initializing the loss function. Before training, we scale the input depth to the $[0, 1]$ range to fix depth space. For the training process, we set the training image and depth map resolution to 320x240 and the Adam optimizer with an initial learning rate of 0.0002. We use the batch size of 16 in training and validation. The processing speed of Pseudo-SLAM is about 15 FPS on average. The detail about real-time performance can be found in the supplemental material.

5.3. Depth Estimation Evaluation

We choose Root Mean Squared Error (RMSE), AbsoluteRelative Difference (REL), Mean Absolute Error (MAE), and Threshold δ as the main metrics to evaluate the precision of the predicted depth map. Their definition is below:

RMSE: $\sqrt{\frac{1}{|\hat{\mathcal{M}}|}\sum_{d\in\hat{\mathcal{M}}}\|d-d^*\|^2}$

REL: $\frac{1}{|\hat{\mathcal{M}}|}\sum_{d\in\hat{\mathcal{M}}}|d-d^*|/d^*$

MAE: $\frac{1}{|\hat{\mathcal{M}}|}\sum_{d\in\hat{\mathcal{M}}}|d-d^*|$

Threshold δ is the percentage of pixels for the predicted depth map that meets the condition: $\mathrm{Max}\left(\frac{d_i}{d_i^*}, \frac{d_i^*}{d_i}\right) < \delta$. Following [31], δ is set to 1.05, 1.10, 1.25. $\hat{\mathcal{M}}$ denotes the mask of transparent object in images. d and d^* denote ground truth depth data and predicted depth data. Note that we regard depth out of range [0, 3] as too far from the camera and remove them.

In our metrics comparison, we train ClearGrasp, LIDF, and TransCG using our dataset with the default training parameter provided in their configuration files. We use their validation API to compute these metrics. The quantitative result in Tab. 1 shows that our model improves all performance compared with previous depth prediction methods of the transparent object.

5.4. Transparent Object Reconstruction Evaluation

We test our Pseudo-SLAM method both in our synthesis dataset and in real captured scenes. In synthesis scenes, we use the ground truth mask and relative motion R, t. In real scenes, we use the prediction mask and estimated motion. For the previous depth prediction method of transparent objects, we replace our model with theirs in the Pseudo-SLAM pipeline to make a reconstruction. Qualitative comparison of reconstruction result and quantitative comparison of camera trajectories performance is shown in Tab. 2 and Fig. 4. The result of the reconstruction evaluation is shown in Tab. 3.

For computing camera trajectories based on our synthetic dataset, we use the visual odometry provided by ElasticFusion instead of an initial estimation through ICP. Metric ”ATE” represents absolute trajectory error, ”R.RPE” and ”T.TPE” represents rotation/translation relative pose error. For reconstruction evaluation, we use the mesh evaluation established by [4]. We regard camera trajectories and reconstruction results estimated by ground truth RGB-D data as the reference data. We align the reconstruction result of point clouds with reference one and transform them to mesh using ball pivoting in the Open3D library. Since the LIDF method doesn’t provide an inference API for their model, we skip

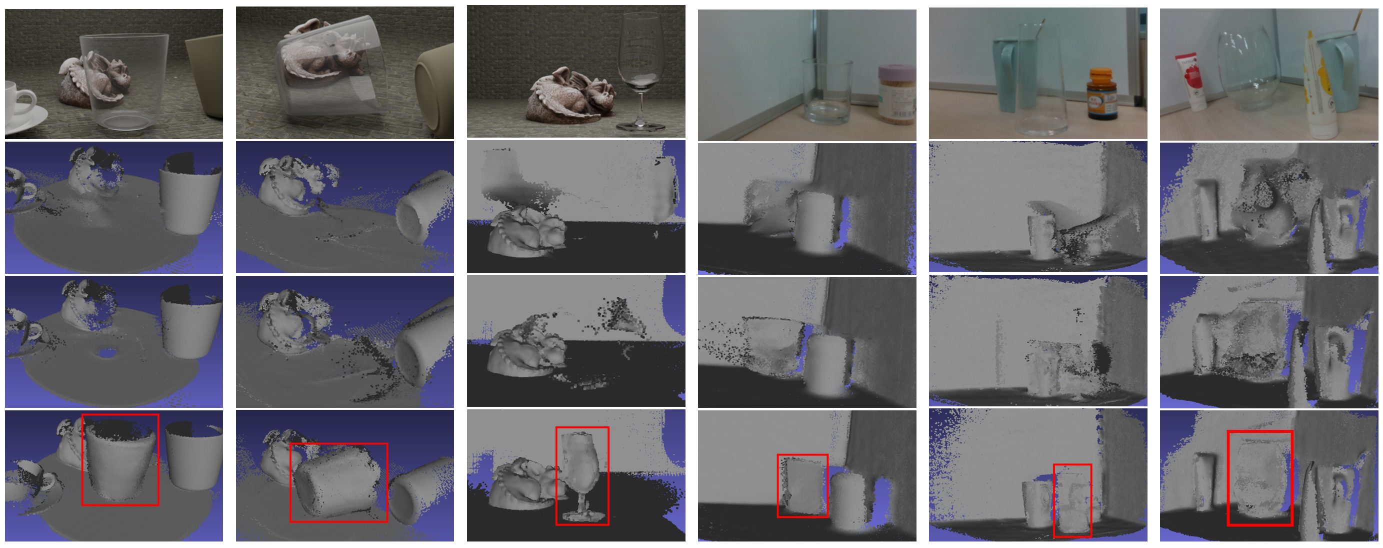


Figure 4. **Reconstruction result**. From top to bottom are the RGB image of this scene; the result of ClearGrasp and TransCG; our result. Our reconstruction result of the transparent object is boxed with the red line. (a), (b) and (c) are the synthetic scene in our dataset. (d), (e) and (f) are real scenes captured by the RGB-D camera.

	ClearGrasp	TransCG	Ours
Scene		ATE ↓	
Glass Cup	0.5210	0.2840	**0.0301**
Glass Cup(down)	0.4653	0.1580	**0.1120**
Goblet(part)	0.0152	**0.0210**	0.0736
Goblet	0.5892	0.6580	**0.1020**
Snack Plate	0.3237	0.6270	**0.0912**
		R.RPE ↓	
Glass Cup	0.00378	0.00201	**0.00143**
Glass Cup(down)	0.00649	0.00513	**0.00288**
Goblet(part)	**0.00129**	0.01880	0.00138
Goblet	0.00491	0.00516	**0.00255**
Snack Plate	0.00247	**0.00199**	0.00206
		T.RPE ↓	
Glass Cup	0.00318	0.00185	**0.00183**
Glass Cup(down)	0.00786	0.00564	**0.00345**
Goblet(part)	0.00113	0.00139	**0.00107**
Goblet	0.00291	0.00317	**0.00173**
Snack Plate	0.00169	0.00164	**0.00148**

Table 2. **Result of trajectory evaluation in all validation scenes.** A lower result means less impact for the transparent object and higher depth prediction accuracy. The trajectory unit in evaluation is *m*.

the experiment on their performance of reconstruction.

From Fig. 4, we can see the improvement of our reconstruction result of the transparent object. In Sec. 1, we have discussed that a consistent depth value in different views is essential for the RGB-D reconstruction of transparent objects. Previous depth prediction methods perform badly on depth prediction for the transparent object in different views, even with a mask. As shown in the second and third rows, the shape of the transparent object is disordered and the background objects are also affected by the wrong predicted depth of the transparent object. By contrast, our method can significantly improve the reconstruction result of the transparent object. Tab. 2 and Tab. 3 further show our improvement in metrics performance. Our lower reconstruction evaluation error means that the position of point clouds in our reconstruction result is close to the ground truth set of point clouds.

Method	Chamfer ↓	Prec ↑	Recall ↑	F-Score ↑
ClearGrasp	0.055	0.540	0.580	0.551
TransCG	0.046	0.627	0.651	0.618
Ours	**0.027**	**0.650**	**0.721**	**0.666**

Table 3. **Reconstruction evaluation result.** We use the mesh evaluation metrics proposed in [4] to compare the reconstruction result of scenes. Our result shows significant improvement compared with existing methods.

5.5. Ablation Study

Normal Prediction. In Tab. 4, we compare the model using the normal prediction module and without using this module. Surface normal provides a lot of geometric information about

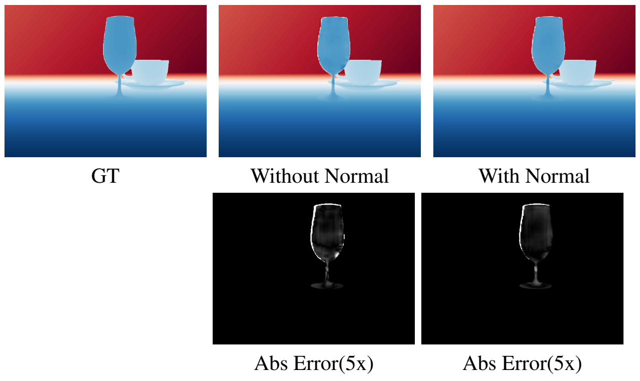


Figure 5. **Visualizing the depth data for goblet scene.** From left to right: ground true depth, depth predicted from model training without and with normal predictor module.

Normal	Trajectory	RMSE	REL	MAE	$\delta_{1.05}$	$\delta_{1.10}$	$\delta_{1.25}$
✓	×	0.156	0.083	0.130	35.69	65.35	94.62
×	✓	0.178	0.087	0.148	41.79	64.13	87.92
✓	✓	**0.148**	**0.077**	**0.122**	**50.30**	**77.76**	**95.92**

Table 4. **Ablation study on normal predictor and trajectory.** Metrics evaluated with/without the normal module or with/without a ground-truth trajectory in training are listed in this table.

an object, especially a transparent object. To further adjust the probability volume with geometric information, instead of an individual normal prediction model, we choose an additional branch to exploit the probability volume generated by the model. The visualization of the result in Fig. 5 shows that the normal loss restrains the depth prediction result, especially at the edge of an object. Note that the error in the left part of the edge is mainly caused by image resizing. With the additional normal predictor, the predicted depth is more accurate.

Trajectories. In the synthesis dataset, ground truth camera trajectories are known, through which we can easily obtain the rotation matrix and translation vector. In this situation, a precise cost volume can be generated. However, pre-estimation of camera trajectories is required in real scenes. We evaluate our model in the synthesis dataset using estimated camera trajectories instead of the ground truth one. The result is shown in Tab. 4. The result gets worse without the ground truth camera trajectories, but it is still decent since the synthesis data contain less noise.

Number of neighboring frames. We examine the performance of our model with different numbers of neighboring input images, we use w to represent this parameter. In this experiment, we test RMSE, MAE, and REL metrics under different w sizes from 1 to 10. The result is shown in Fig. 6. Note that, for $w = 1$, we set a copy of the current image as its neighborhood, an identity mat as the rotation matrix, and a

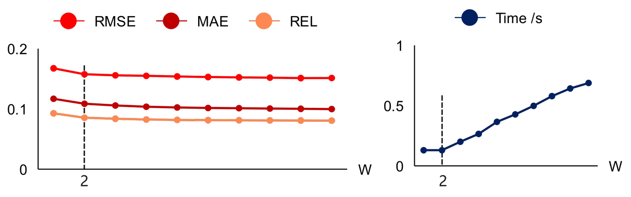


Figure 6. **Ablation study on selecting the number of neighboring images.** We test w from 1 to 10 to figure out the effect of the number of neighboring images for depth prediction and corresponding time consumption.

zero mat as the translation vector. From the left of this figure, we can find that more viewpoints are helpful to reduce the noise in cost volume. From the right of this figure, we can find that as the number of neighboring images increases, the inferring time rapidly grows. In 2-views, the time used for inference is about 0.2s. Image input for each additional view costs about an extra 0.08s in generating cost volume. The point of inflexion found in this figure is $w = 2$. The result is worse for $w = 1$, which doesn't consider the relative motion between neighboring images. For $w >= 2$, the growth trend on metrics slows down a lot, but time consumption is still increasing steadily. To balance the running time and prediction accuracy, we choose $w = 2$ in all experiments.

6. Conclusion

Compared with previous work on transparent object reconstruction, we propose a real-time 3D reconstruction method without any prior knowledge requirement of the scene. Our reconstruction method is based on ElasticFusion with a proposed stereo-based depth prediction model. In our model, we use plane sweep stereo to take advantage of the spatial consistency of the RGB information between sequential images and fuse our model with the scale information of the background depth. We design a surface normal prediction branch to further restrict the depth prediction result. Experiments show our model improves the metrics of depth prediction for transparent objects and the result of our 3D reconstruction method performs better in real scenes.

Limitation There are certain categories of transparent objects lacking in our current dataset such as handcrafts made of glass, which could be added in the following research. Our reconstructed results also show a lack of geometric details. Scenes that contain multiple transparent objects that occlude with each other are also hard to handle by our algorithm, which we will investigate in future works.

7. Acknowledgments

This work is supported by the Fundamental Research Funds for the Central Universities (Nankai University, No. 63233080) and the Natural Science Foundation of China (62132012).

References

[1] Panos Achlioptas, Olga Diamanti, Ioannis Mitliagkas, and Leonidas Guibas. Learning representations and generative models for 3d point clouds. In *International conference on machine learning*, pages 40–49. PMLR, 2018. 3

[2] Nicolas Alt, Patrick Rives, and Eckehard Steinbach. Reconstruction of transparent objects in unstructured scenes with a depth camera. In *2013 IEEE International Conference on Image Processing*, pages 4131–4135. IEEE, 2013. 3

[3] Jia-Wang Bian, Huangying Zhan, Naiyan Wang, Zhichao Li, Le Zhang, Chunhua Shen, Ming-Ming Cheng, and Ian Reid. Unsupervised scale-consistent depth learning from video. *International Journal of Computer Vision*, 129(9):2548–2564, 2021. 2

[4] Aljaz Bozic, Pablo Palafox, Justus Thies, Angela Dai, and Matthias Nießner. Transformerfusion: Monocular rgb scene reconstruction using transformers. *Advances in Neural Information Processing Systems*, 34:1403–1414, 2021. 6, 7

[5] David Eigen and Rob Fergus. Predicting depth, surface normals and semantic labels with a common multi-scale convolutional architecture. In *Proceedings of the IEEE international conference on computer vision*, pages 2650–2658, 2015. 3

[6] David Eigen, Christian Puhrsch, and Rob Fergus. Depth map prediction from a single image using a multi-scale deep network. *NIPS*, 27, 2014. 2, 3

[7] Jakob Engel, Thomas Schöps, and Daniel Cremers. Lsd-slam: Large-scale direct monocular slam. In *Computer Vision–ECCV 2014: 13th European Conference, Zurich, Switzerland, September 6-12, 2014, Proceedings, Part II 13*, pages 834–849. Springer, 2014. 1

[8] Hongjie Fang, Hao-Shu Fang, Sheng Xu, and Cewu Lu. Transcg: A large-scale real-world dataset for transparent object depth completion and a grasping baseline. *IEEE Robotics and Automation Letters*, 7(3):7383–7390, 2022. 1, 2, 3, 5, 6

[9] Dorian Gálvez-López, Marta Salas, Juan D Tardós, and José MM Montiel. Real-time monocular object slam. *Robotics and Autonomous Systems*, 75:435–449, 2016. 2

[10] Ravi Garg, Vijay Kumar Bg, Gustavo Carneiro, and Ian Reid. Unsupervised cnn for single view depth estimation: Geometry to the rescue. In *Computer Vision–ECCV 2016: 14th European Conference, Amsterdam, The Netherlands, October 11-14, 2016, Proceedings, Part VIII 14*, pages 740–756. Springer, 2016. 2

[11] Clément Godard, Oisin Mac Aodha, and Gabriel J Brostow. Unsupervised monocular depth estimation with left-right consistency. In *CVPR*, pages 270–279, 2017. 2

[12] Jungong Han, Ling Shao, Dong Xu, and Jamie Shotton. Enhanced computer vision with microsoft kinect sensor: A review. *IEEE transactions on cybernetics*, 43(5):1318–1334, 2013. 1

[13] Xufeng Han, Thomas Leung, Yangqing Jia, Rahul Sukthankar, and Alexander C Berg. Matchnet: Unifying feature and metric learning for patch-based matching. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 3279–3286, 2015. 3

[14] Binh-Son Hua, Minh-Khoi Tran, and Sai-Kit Yeung. Pointwise convolutional neural networks. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 984–993, 2018. 3

[15] Lam Huynh, Phong Nguyen-Ha, Jiri Matas, Esa Rahtu, and Janne Heikkilä. Guiding monocular depth estimation using depth-attention volume. pages 581–597, 2020. 2

[16] Sunghoon Im, Hyowon Ha, Gyeongmin Choe, Hae-Gon Jeon, Kyungdon Joo, and In So Kweon. Accurate 3d reconstruction from small motion clip for rolling shutter cameras. *IEEE transactions on pattern analysis and machine intelligence*, 41(4):775–787, 2018. 3

[17] Sunghoon Im, Hae-Gon Jeon, Stephen Lin, and In So Kweon. Dpsnet: End-to-end deep plane sweep stereo. *arXiv preprint arXiv:1905.00538*, 2019. 2, 3, 4

[18] Alex Kendall, Hayk Martirosyan, Saumitro Dasgupta, Peter Henry, Ryan Kennedy, Abraham Bachrach, and Adam Bry. End-to-end learning of geometry and context for deep stereo regression. In *Proceedings of the IEEE international conference on computer vision*, pages 66–75, 2017. 3, 5

[19] Leonid Keselman, John Iselin Woodfill, Anders Grunnet-Jepsen, and Achintya Bhowmik. Intel realsense stereoscopic depth cameras. In *Proceedings of the IEEE conference on computer vision and pattern recognition workshops*, pages 1–10, 2017. 1

[20] Shiyi Lan, Ruichi Yu, Gang Yu, and Larry S Davis. Modeling local geometric structure of 3d point clouds using geo-cnn. In *Proceedings of the IEEE/cvf conference on computer vision and pattern recognition*, pages 998–1008, 2019. 3

[21] Bo Li, Chunhua Shen, Yuchao Dai, Anton Van Den Hengel, and Mingyi He. Depth and surface normal estimation from monocular images using regression on deep features and hierarchical crfs. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 1119–1127, 2015. 2, 3

[22] Zhengqin Li, Yu-Ying Yeh, and Manmohan 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. 3

[23] Chao Liu, Jinwei Gu, Kihwan Kim, Srinivasa G Narasimhan, and Jan Kautz. Neural rgb (r) d sensing: Depth and uncertainty from a video camera. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 10986–10995, 2019. 2

[24] Jiageng Mao, Xiaogang Wang, and Hongsheng Li. Interpolated convolutional networks for 3d point cloud understanding. In *Proceedings of the IEEE/CVF international conference on computer vision*, pages 1578–1587, 2019. 3

[25] Raul Mur-Artal, Jose Maria Martinez Montiel, and Juan D Tardos. Orb-slam: a versatile and accurate monocular slam system. *IEEE transactions on robotics*, 31(5):1147–1163, 2015. 1

[26] H Murase. Surface shape reconstruction of an undulating transparent object. In *Proceedings Third International Conference on Computer Vision*, pages 313–314. IEEE Computer Society, 1990. 3

[27] Charles R Qi, Hao Su, Kaichun Mo, and Leonidas J Guibas. Pointnet: Deep learning on point sets for 3d classification

and segmentation. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 652–660, 2017. 2, 3, 4

[28] Charles Ruizhongtai Qi, Li Yi, Hao Su, and Leonidas J Guibas. Pointnet++: Deep hierarchical feature learning on point sets in a metric space. *Advances in neural information processing systems*, 30, 2017. 2, 3, 4

[29] Xiaojuan Qi, Renjie Liao, Zhengzhe Liu, Raquel Urtasun, and Jiaya Jia. Geonet: Geometric neural network for joint depth and surface normal estimation. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 283–291, 2018. 2

[30] Yiming Qian, Minglun Gong, and Yee Hong Yang. 3d reconstruction of transparent objects with position-normal consistency. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 4369–4377, 2016. 3

[31] Shreeyak Sajjan, Matthew Moore, Mike Pan, Ganesh Nagaraja, Johnny Lee, Andy Zeng, and Shuran Song. Clear grasp: 3d shape estimation of transparent objects for manipulation. In *2020 IEEE International Conference on Robotics and Automation (ICRA)*, pages 3634–3642. IEEE, 2020. 1, 2, 3, 5, 6

[32] Evan Shelhamer, Jonathan T Barron, and Trevor Darrell. Scene intrinsics and depth from a single image. pages 37–44, 2015. 2

[33] Tianwei Shen, Lei Zhou, Zixin Luo, Yao Yao, Shiwei Li, Jiahui Zhang, Tian Fang, and Long Quan. Self-supervised learning of depth and motion under photometric inconsistency. In *Proceedings of the IEEE/CVF International Conference on Computer Vision Workshops*, pages 0–0, 2019. 2

[34] J. Sturm, N. Engelhard, F. Endres, W. Burgard, and D. Cremers. A benchmark for the evaluation of rgb-d slam systems. In *Proc. of the International Conference on Intelligent Robot Systems (IROS)*, Oct. 2012. 1

[35] Keisuke Tateno, Federico Tombari, Iro Laina, and Nassir Navab. Cnn-slam: Real-time dense monocular slam with learned depth prediction. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 6243–6252, 2017. 2

[36] Lyne P Tchapmi, Vineet Kosaraju, Hamid Rezatofighi, Ian Reid, and Silvio Savarese. Topnet: Structural point cloud decoder. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 383–392, 2019. 3

[37] Xiaolong Wang, David Fouhey, and Abhinav Gupta. Designing deep networks for surface normal estimation. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 539–547, 2015. 3

[38] Thomas Whelan, Renato F Salas-Moreno, Ben Glocker, Andrew J Davison, and Stefan Leutenegger. Elasticfusion: Real-time dense slam and light source estimation. *The International Journal of Robotics Research*, 35(14):1697–1716, 2016. 1, 2

[39] Haozhe Xie, Hongxun Yao, Shangchen Zhou, Jiageng Mao, Shengping Zhang, and Wenxiu Sun. Grnet: Gridding residual network for dense point cloud completion. In *Computer Vision–ECCV 2020: 16th European Conference, Glasgow, UK, August 23–28, 2020, Proceedings, Part IX*, pages 365–381. Springer, 2020. 3

[40] Haoping Xu, Yi Ru Wang, Sagi Eppel, Alàn Aspuru-Guzik, Florian Shkurti, and Animesh Garg. Seeing glass: joint point cloud and depth completion for transparent objects. *arXiv preprint arXiv:2110.00087*, 2021. 3

[41] Guanglei Yang, Hao Tang, Mingli Ding, Nicu Sebe, and Elisa Ricci. Transformer-based attention networks for continuous pixel-wise prediction. In *ICCV*, pages 16269–16279, 2021. 2

[42] Sergey Zagoruyko and Nikos Komodakis. Learning to compare image patches via convolutional neural networks. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 4353–4361, 2015. 3

[43] Jure Zbontar and Yann LeCun. Computing the stereo matching cost with a convolutional neural network. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 1592–1599, 2015. 3

[44] Luyang Zhu, Arsalan Mousavian, Yu Xiang, Hammad Mazhar, Jozef van Eenbergen, Shoubhik Debnath, and Dieter Fox. Rgb-d local implicit function for depth completion of transparent objects. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 4649–4658, 2021. 1, 2, 3, 5, 6

[45] Yifan Zhu, Jiaxiong Qiu, and Bo Ren. Transfusion: A novel slam method focused on transparent objects. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 6019–6028, 2021. 2, 3, 5

[46] Yuliang Zou, Zelun Luo, and Jia-Bin Huang. Df-net: Unsupervised joint learning of depth and flow using cross-task consistency. In *Proceedings of the European conference on computer vision (ECCV)*, pages 36–53, 2018. 2