Monocular Depth Estimation for Glass Walls with Context: A New Dataset and Method

Yuan Liang, Bailin Deng, Wenxi Liu, Jing Qin, and Shengfeng He, *Senior Member, IEEE*

Abstract—Traditional monocular depth estimation assumes that all objects are reliably visible in the RGB color domain. However, this is not always the case as more and more buildings are decorated with transparent glass walls. This problem has not been explored due to the difficulties in annotating the depth levels of glass walls, as commercial depth sensors cannot provide correct feedbacks on transparent objects. Furthermore, estimating depths from transparent glass walls requires the aids of surrounding context, which has not been considered in prior works. To cope with this problem, we introduce the first Glass Walls Depth Dataset (GW-Depth dataset). We annotate the depth levels of transparent glass walls by propagating the context depth values within neighboring flat areas, and the glass segmentation mask and instance level line segments of glass edges are also provided. On the other hand, a tailored monocular depth estimation method is proposed to fully activate the glass wall contextual understanding. First, we propose to exploit the glass structure context by incorporating the structural prior knowledge embedded in glass boundary line segment detections. Furthermore, to make our method adaptive to scenes without structure context where the glass boundary is either absent in the image or too narrow to be recognized, we propose to derive a reflection context by utilizing the depth reliable points sampled according to the variance between two depth estimations from different resolutions. High-resolution depth is thus estimated by the weighted summation of depths by those reliable points. Extensive experiments are conducted to evaluate the effectiveness of the proposed dual context design. Superior performances of our method is also demonstrated by comparing with state-of-the-art methods. We present the first feasible solution for monocular depth estimation in the presence of glass walls, which can be widely adopted in autonomous navigation.

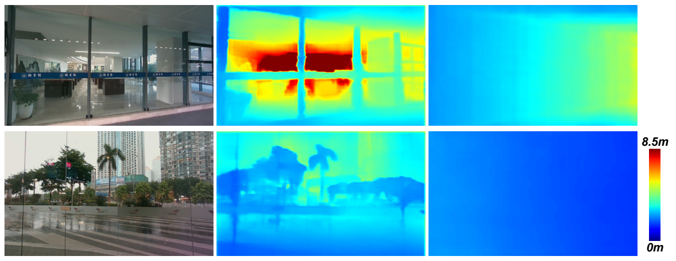
Index Terms—Monocular Depth Estimation, Glass Detection, Line Segment Detection.

1 Introduction

Glass walls, such as glass facades and glass doors, are common features in modern architecture. These design elements introduce new challenges for autonomous navigation systems. Distance measurements acquired by depth or stereo cameras on these surfaces can be unreliable due to their transparent and reflective properties. For instance, many commodity depth sensors use active range-sensing approaches such as time-of-flight and structured light, where the sensor emits a light pulse or a unique known pattern into the scene. This emitted light pulse or pattern may pass through transparent objects, preventing accurate feedback of glass walls to the receivers. Consequently, tasks that heavily rely on capturing or estimating depth information, such as visual navigation [1], [2], 3D object detection [3], [4] and autonomous driving [5], [6], [7], can easily fail in the scenarios involving glass walls (see Fig. 1(b)).

- *The work is supported by National Natural Science Foundation of China (No. 61972162); Guangdong Natural Science Funds for Distinguished Young Scholar (No. 2023B1515020097); and Singapore Ministry of Education Academic Research Fund Tier 1 (MSS23C002). Corresponding author: Shengfeng He.*

- *Yuan Liang is with the School of Computer Science and Engineering, South China University of Technology, Guangzhou, China. He is also with the School of Computing and Information Systems, Singapore Management University, Singapore. E-mail: yuanliang07@gmail.com.*
- *Bailin Deng is with the School of Computer Science and Informatics, Cardiff University, United Kingdom. E-mail: DengB3@cardiff.ac.uk.*
- *Wenxi Liu is with the College of Computer and Data Science, Fuzhou University, China. E-mail: wenxi.liu@hotmail.com.*
- *Jing Qin is with the School of Nursing, the Hong Kong Polytechnic University, Hong Kong, China. E-mail: harry.qin@polyu.edu.hk.*
- *Shengfeng He is with the School of Computing and Information Systems, Singapore Management University, Singapore. Email: shengfenghe@smu.edu.sg.*



(a) Input (b) NeWCRFs [8] (c) Ours

Fig. 1: Two typical scenarios involving glass walls that cannot be handled by existing autonomous navigation systems. The first row shows an indoor scene with multiple glass walls separated by opaque frames, while the second row presents an outdoor glass wall without a discernible frame. (b): depth estimation results from NeWCRFs [8] trained on the NYU Depth V2 dataset [9], where the glass wall depths are not accurately detected, potentially leading to mobile robots or autonomous cars crashing into them. (c): our results successfully estimate the depth levels of transparent glass walls by incorporating structure and reflection contexts.

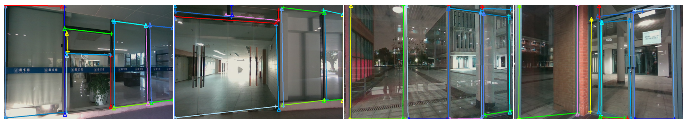
In this paper, we aim to estimate the depth levels of transparent glass walls from a pure monocular vision perspective. Several studies [10], [11], [12] have focused on depth estimation for small transparent objects such as bottles and cups. In these works, all transparent objects are placed on an opaque surface such as a table, and the depth maps are captured at relatively close distances, making them more suitable for robotic manipulation tasks [11], [12]. Other glass related datasets [13], [14] include annotated glass masks but do not provide depth annotations. Currently, there is a lack of RGB-D datasets that capture diverse glass walls and include well-annotated depth maps for both glass and other reflective surfaces (*e.g.*, walls and floors with ceramic tiles) to accurately represent their true distances from depth sensors.

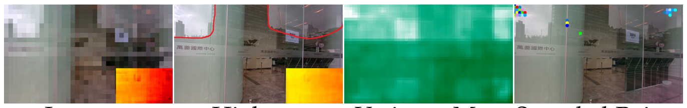
To address this gap, we construct the first RGB-D dataset with glass walls and other glass structures, such as facades, windows, doors, and railings, called the Glass Walls Depth Dataset (GW-Depth Dataset). It includes 1,200 densely-annotated images captured in various environments, using two types of commodity depth sensors. The main challenge in developing such a dataset is correcting depth values in areas where they are missing or incorrect due to reflection and transparency. We note that glass walls typically consist of one or more flat glass panels, thus missing or incorrect depths can be completed or corrected using neighboring points that are coplanar with them. To achieve this, we manually select multiple points with accurate depths to form a convex polygon, enclosing an area with missing or incorrect depth. These selected depth points are then used to interpolate the ground-truth depths within the polygon's interior.

Even with our well-annotated dataset, estimating depths of transparent glass walls with monocular vision remains challenging. Humans, unlike other species (such as birds that frequently collide with transparent glass walls or windows), can distinguish transparent obstacles due to their understanding of the surrounding context. This context cannot be obtained merely by enlarging the receptive fields as prior works have done [8], [15], [16], [17]. Instead, transparent glass walls possess unique characteristics that require specialized expert knowledge to describe. To this end, we explore two types of contexts: glass structure context and reflection context. The glass structure context is defined by the outer frames of glass walls, which are more visually discernible than the transparent interior (see Fig. 2(a)). To leverage this structural information, we formulate a line segment detection task to depict the structure of glass walls. High-scoring line segments are selected to sample structure-embedded features, which are then fed into a tailored point-guided transformer to predict both depth levels and glass segments. In addition, the reflection context is proposed to supplement scenarios where the glass boundary is either absent in the image or too narrow to be reliably recognized. We identify glass walls by integrating reflection cues based on a key observation: reflection properties can serve as indicators of glass presence [14], [18], but these properties are only available in images with sufficient resolution. As shown in Fig. 2(b), the depth estimation results exhibit significant differences (brighter regions in the variance map) in reflection regions. We use this clue to reveal glass walls by collecting a series of depth reliable points to inject reflection understanding to complement the structure context.

Extensive experiments are conducted on our GW-Depth dataset for depth estimation and glass segmentation tasks. Our methodology on glass wall contextual understanding achieves superior performances compared to state-of-the-art methods trained on our dataset. We not only propose the first glass walls depth dataset, but also demonstrate the first feasible solution on monocular depth estimation that can potentially benefit autonomous navigation.

In summary, the contribution of this paper is fourfold:


(a) Structure Context



(b) Reflection Context

Fig. 2: We reveal transparent glass walls using two types of contexts. The structure context (shown in the top row) is based on detecting outer frames (displayed with colored lines) of glass walls, which depicts the glass structure. The reflection context (shown in the second row) is based on the variation of reflective appearance across different resolutions. Here we show a close-up view of the glass, where the boundaries are too narrow to be reliably recognized. Reflections can be observed in this close-up view with high-resolution details (marked by red curves), but these illumination properties may be lost in a lower-resolution version of the image. The depth estimations at two resolutions (shown at the bottom-right corners) exhibit significant differences in reflection regions (brighter regions in "Variance Map"). We leverage this reflection clue to detect glass walls by sampling depth-reliable points (denoted by colored circles in "Sampled Points") on the depth variance map.

- We introduce the first glass walls RGB-D dataset, which captures diverse real-world scenarios and contains 1,200 images with well-annotated depth maps, glass masks, and glass outlines.
- We develop a depth interpolation pipeline to generate well-annotated depths, addressing the issue of missing or incorrect depth values in transparent or reflective regions.
- We propose a dual-context approach for understanding glass walls: the structure context is used to leverage information from the glass outer frames, while the reflection context handles scenarios where visible glass structures are either absent or cannot be reliably recognized.
- Our work is the first feasible method for transparent glass wall monocular depth estimation, and outperforms state-of-the-art methods even when trained with the same dataset.

2 Related Work

2.1 Monocular Depth Estimation

Recent work on monocular depth estimation employs deep learning methods to extract multi-scale features, with varied choices on training architecture, loss functions, and auxiliary tasks. Given the significant differences between supervised and self-supervised monocular depth estimation, we provide separate summaries for each approach in the following.

Architecture choices and loss functions. Many self-supervised methods focus on the correlations between estimated depths and reconstructed RGB images derived from adjacent monocular video frames [19] or stereo camera pairs [20]. The photometric loss [19], [20] is typically used to indirectly supervise the estimated depths. Recent works [21], [22], [23], [24] aim to address the limitations of photometric loss, including issues related to ego motion, occlusion, and scale ambiguity, etc.. Godard *et al.* [21] apply photometric loss only on pixels with the minimum appearance reconstruction error, which are more likely to be in non-occlusion areas. Guizilini *et al.* [22] sample appearance-matching points along depth-discretized epipolar lines to mitigate ambiguities and local minima caused by a lack of texture and ego motion. RM-Depth [23] integrates predictions of motion fields and camera poses to address artifacts caused by moving objects.. Petrovai *et al.* [24] use a scale-invariant loss function in a teacher-student network to alleviate scale ambiguity issues.

In supervised monocular depth estimation, various architecture choices have been explored. Eigen *et al.* [15] use a CNN network with a coarse-scale encoder and fine-scale decoder. They propose a scale-invariant loss to address the scale ambiguity issue, encouraging each pair of predicted depths to have a similar scale of difference to their corresponding pair in the ground truth. Following this, Eigen and Fergus [25] add depth gradient errors to the scale-invariant loss to promote predictions with similar local structures. Residual block-based CNN networks [26] bring more local-global context relations, which are widely used in depth estimation. With these CNN encoders, implicit geometry restrictions like coplanarity [16], [27] are introduced to facilitate depth estimation. Lee *et al.* [16] add a local-planar guidance module into the decoder, where 4D plane coefficients represent the relationship within local grids, providing adaptive local relations between different feature resolutions. Similar to [16], P3Depth [27] independently learns plane coefficients and resamples offsets. Initial and refined depth maps are produced using plane coefficients and resampled plane coefficients, respectively. A confidence map is predicted to adaptively fuse the two depth maps for refined depth estimation. Taking advantage of the larger context enabled by transformer networks, Yang *et al.* [17] and NeWCRFs [8] use transformer blocks in the decoder part; a gated attention module [17] or a neural window fully-connected CRFs module [8] is appended at each level of the decoder to combine multi-scale feature maps. Another line of works [28], [29], [30] treats depth estimation as an ordinal regression problem. Depth labels are discretized into predefined [28], [29] or learned [30] intervals, with classification scores predicted for each depth interval. These scores are then used to generate weighted average estimated depths.

Similar to Yang *et al.* [17], our proposed method adopts a transformer decoder on minimum scale features (*e.g.* $\frac{1}{32}$ resolution). However, instead of directly using vision transformer (ViT) blocks, we design a novel attention layer that incorporates glass structure features derived from the glass boundary line segments.

Depth estimation with auxiliary tasks. In self-supervised monocular depth estimation, auxiliary tasks such as segmentation [31], simultaneous localization and mapping (SLAM) [32], optical flow [32], [33], multiscale fusion [34], and relative depth estimation [35], are incorporated to help improve correspondence between multiple frames [32], [33] or maintain consistency between global structures and local details [31], [34], [35].

Recent works [36], [37], [38], [39] aim to introduce prior knowledge to address the challenges in supervised depth estimation by exploring joint tasks like semantic segmentation, surface normal estimation, and depth estimation. Xu *et al.* [36] design an encoder that generates intermediate multi-task predictions, which are then refined using a distillation module. TRL [37] and PAP [38] focus on creating task-specific attention maps or affinity matrices, connected using gated attention [37] or adaptive combination [38]. Lu *et al.* [39] train multi-task models independently but link them with a consistency loss based on physical and logical constraints.

Our multi-task architecture also includes auxiliary tasks including line segment detection and semantic segmentation. These tasks assist the dual context exploration: glass boundary line segment detection provides structure context, while glass segmentation helps correct potential inaccuracies in structure context, especially in scenes where the glass outline is absent or hard to recognize reliably.

2.2 Transparent Object Understanding

Traditional computer vision tasks (like image classification and object detection) assume the input visual information to be sufficiently reliable. However, tasks such as visual navigation, robotic manipulation, and novel view synthesis can face unreliable observations and abnormal results due to transparent objects. Recent work on transparent glass mainly focuses on glass segmentation, depth estimation, and 3D reconstruction or novel view synthesis.

Transparent or glass-like object segmentation. Transparent or glass-like object segmentation [13], [14], [40], [41], [42], [43] has made significant progress in recent years, with new glass segmentation datasets featuring real-world scenes being introduced [13], [14], [42]. Glass segmentation methods require either the object's intrinsic properties or a large context to identify cues that indicate glass areas. Kalra *et al.* [41] use a polarization camera to capture polarized imagery, making transparent textures more visible. Xu *et al.* [40] estimate the likelihood of a pixel belonging to a transparent object using light-field linearity. Mei *et al.* [13] design a model with a larger context by adopting separate convolutions to extract abundant contexts from a large field. Other works [14], [42], [43] use glass boundaries as prior knowledge to facilitate segmentation, with dilated convolutions [14], [42] or graph convolution networks [43] employed to extract multi-scale, large context features.

Depth estimation on smaller transparent objects. Several studies [10], [11], [12], [44] investigate depth estimation for smaller transparent objects such as bottles or cups. Wang *et al.* [10] use depth cues to localize semi-transparent objects. ClearGrasp [11] introduces a synthetic RGB-D dataset for transparent objects and refines the initial depth using a multi-task architecture, which includes normal estimation, boundary detection, and glass masking. Zhu *et al.* [12] address the non-transparent contact dependency and inference inefficiency of ClearGrasp by learning a local implicit depth function (LIDF) for ray-voxel pairs, improving inference efficiency and generalization to unseen data. While these works [11], [12] primarily train on synthetic datasets, the Booster Dataset [44] presents a stereo RGB-D dataset featur-

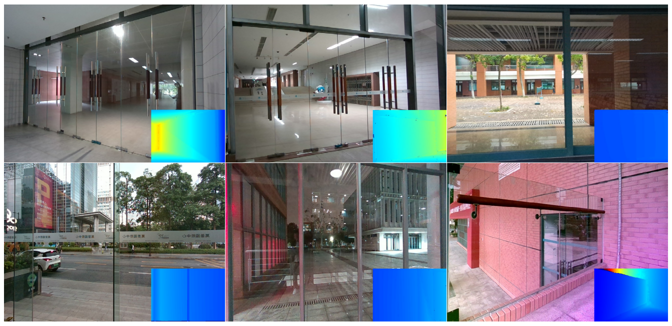


Fig. 3: Six captured images and interpolated depths in our GW-Depth dataset. The depth map is visualized in the bottom-right corners.

ing indoor scenes with transparent and specular surfaces to address real-world challenges.

Transparency-aware object reconstruction. To reconstruct transparent objects, Transfusion [45] uses a segmentation network to first eliminate the adverse influence of transparent objects, resulting in more accurate camera poses. Transparent objects are then reconstructed using a visual hull-based method. To address inaccurate depth estimation and rendering in neural radiance fields (NeRF) models for transparent objects, NeRFReN [46] separately models transmitted and reflected parts of the scene with distinct neural radiance fields. A dedicated network is also employed to leverage geometry priors.

These works primarily focuses on smaller transparent objects like glass cups, bottles, and vases, which does not represent real-world scenarios involving large-scale glass walls encountered by visual navigation systems. Moreover, most studies treat depth estimation of transparent objects as an auxiliary task to aid segmentation or 3D reconstruction tasks. In contrast, we introduce an RGB-D dataset for standard glass walls and propose a dual context embedded deep neural network specifically designed to tackle the glass depth estimation problem.

3 Glass Walls Depth Dataset

Our dataset aims to fill the gap in RGB-D datasets featuring large glass walls, where typical depth cameras may be unreliable. We use two commodity depth cameras, the Intel RealSense Depth Camera D455 and the Microsoft Azure Kinect DK, to capture RGB and depth images. In this section, we present and analyze our dataset construction method.

3.1 Image and Depth Capturing

To include realistic scenes where commodity depth cameras may be unreliable, the images were captured along corridors with glass facades and glass guardrails. Our dataset includes 66 indoor and outdoor scenes, and the capturing period include both daytime and nighttime. Each scene features various distances and views of glass walls, capturing the impact of reflections on depth maps under different conditions.

For each scene, the depth maps are aligned with the RGB images according to the intrinsic device parameters. As explained later, our depth annotation approach relies on

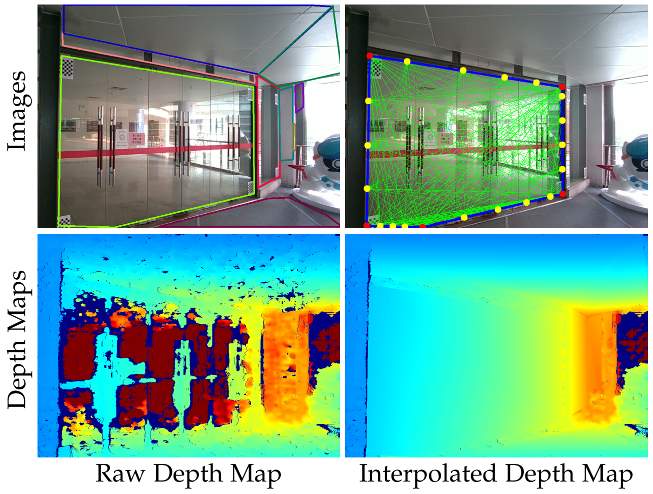


Fig. 4: Depth interpolation on a captured image. The areas with invalid depth are enclosed with polygons (see top left, where different polygons are displayed in different colors). Valid depth points for one polygon are shown as red circles in the top-right image. Depth interpolation occurs along polygon edges (blue lines) and connecting line segments between edges (green lines). The raw depth map and interpolated depth map are shown in the bottom row.

the availability of valid depth points that can be used to interpolate the depth values in areas where the raw depth map is unreliable. However, for some scenes with a large area of transparent/specular materials, there may not be sufficient valid depth points for our depth annotation. Therefore, where necessary, we manually pasted thin opaque covers on corners of glass walls, to provide valid depth points for interpolation. In most cases, we also captured the same scene without covers to preserve the original appearance. Fig. 3 shows examples of captured images and our annotated depths.

3.2 Depth Correction and Dense Annotations

Invalid or missing depth values often appear in transparent or specular areas, like glass walls or tiled floors (see an example of a raw depth map in Fig. 4). Our key observation is that such areas usually have a piecewise planar shape. For example, a glass wall often consists of flat glass panels, while a tiled surface is typically covered by coplanar tiles. In addition, accessories on glass walls such as door handles typically occupy a small area in the depth map and can be ignored in many applications. Therefore, starting with a few valid depth values on a planar piece, we can compute true depth values for the rest of the area through interpolation. We also include segmentation masks to differentiate glass from other materials and annotate area boundaries. Our corrected depth maps and dense annotations are created in three steps:

1) We first label polygons to enclose areas with incorrect depth values, connecting vertices with valid depth values.
2) We then perform depth interpolation to correct depth values within the polygons.
3) Finally, we crop out areas that still contain invalid depth values due to insufficient valid depth points for interpolation.

The details for each step are explained below.

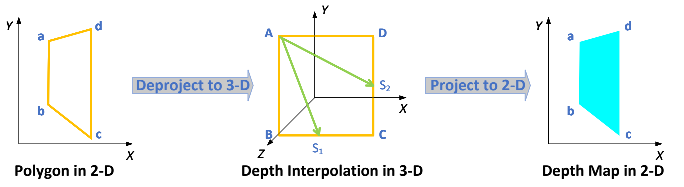


Fig. 5: Depth interpolation pipeline. Left: a polygon that encloses an area with invalid depth values is first deprojected into 3D space. Middle: the depth interpolation is conducted on polygon edges (denoted by orange edges) and then on the interior area (with green arrows connecting different edges). Right: finally, the generated 3D points are projected back into 2D space to assign depth values for the corresponding pixels.

3.2.1 Labeling Polygons

We visually inspect the raw depth map and use Labelme[1] to manually label convex polygons around planar transparent or specular regions. These polygons have vertices located in areas with valid depth (see Fig. 4 for an example). As explained in Sec. 3.1, we add opaque covers to corners of such areas to ensure enough valid depth points for polygon vertices. For consistency, the vertices of each polygon are labeled in a counter-clockwise order. We also record all polygons enclosing glass regions and use them to create dense segmentation masks for those regions.

3.2.2 Depth Interpolation and Image Cropping

Using the labeled polygons and the transformation from 2D pixels to 3D coordinates, we generate ground-truth depth maps by interpolating along the line segments using valid depth values at their endpoints in the 3D space. The interpolation pipeline has three steps, as shown in Fig. 5.

First, we create 2D polygons with vertices that have valid depth values, as explained in Sec. 3.2.1. we use the camera's intrinsic parameters to determine the 3D coordinates of these vertices based on their pixel coordinates and depth values (see Fig. 5, middle). With the assumption that the polygon is in a flat area, the segments of each 2D polygon are mapped to 3D straight line segments connecting the vertices. Therefore, for a polygon segment connecting vertices $A, B \in \mathbb{R}^3$, we compute a set of intermediate points $\{P_k \in \mathbb{R}^3\}$ on the segment as follows:

$$P_k = (1-\alpha_k)A + \alpha_k B, \qquad k = 1, 2 \ldots, \tag{1}$$

where $\alpha_k = (U \cdot k)/\|A - B\|$, and U is a unit distance (*e.g.*, 1mm) that controls the density of the intermediate points.

Afterward, 3D points for the interior regions of the polygons are generated by computing points on line segments with endpoints on different polygon edges (*e.g.*, line segments $\overline{\mathrm{AS}_1}$ and $\overline{\mathrm{AS}_2}$ in the middle figure of Fig. 5). Using the z-coordinate as the depth value, these 3D points are then projected back onto 2D pixels.

It is worth noting that each interpolated 3D point is a convex combination of the valid 3D points used for interpolation. Therefore, if the underlying surface area is truly coplanar, then the error of the interpolated depth will not exceed the maximum depth error among the valid points.

1. https://github.com/wkentaro/labelme

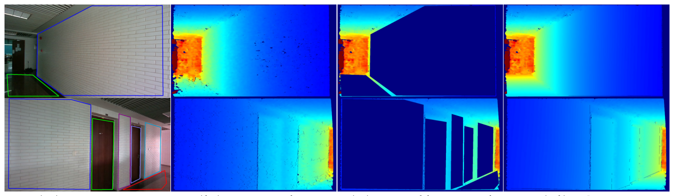


Fig. 6: Two example images for testing the interpolation settings with a sample ratio of 50% and a unit distance of 6 millimeters. The interpolation areas are enclosed by polygons in (a). The captured depth maps are visualized in (b). The interpolation areas on the depth maps are hollowed (shown in dark blue) in (c). (d) shows the completed depth maps produced by our depth interpolation procedure.

Similarly, in nearly coplanar areas, the interpolated depth will have not have a large error, provided that the depth errors at the valid points are small. This property ensures the reliability of our interpolation approach. See also Sec. 3.4 for further validation.

Due to the large amount of line segments connecting different edges, Interpolating at a millimeter-scale in 3D space can be time-consuming. Moreover, multiple adjacent collinear points may project to the same pixel due to limited image resolution. To address this, we sample a predefined ratio of points on each boundary edge and connect them with intermediate points and vertices on other edges to form line segments for interior interpolation. To further reduce computation, we set a large enough unit distance parameter U to decrease the number of computed points while ensuring sufficient density for the final depth map.

To evaluate the impact of different sample ratios and unit distance parameters, we captured three non-glass scenes with valid depth maps (two example images are shown in Fig. 6). We took two images for each scene, enclosed some flat areas with polygons, and set the depth values inside the polygons to zero to simulate invalid depth areas. We then used our interpolation method to correct the depth values inside these polygons, and compared the original depth maps as ground-truth to assess the accuracy of our interpolated results. We used the root mean squared error (RMS) and average relative error (REL) as evaluation metrics. Specifically, let $\mathcal{P}$ be the set of valid depth points in the ground-truth depth map, and $y_p, \widehat{y}_p$ be the ground-truth and estimated depth values for a point $p \in \mathcal{P}$, respectively. The two metrics are defined as:

- *Root mean squared error* (RMS):

$$\mathrm{RMS} = \sqrt{\left(\sum\nolimits_{p\in\mathcal{P}} (y_p - \widehat{y}_p)^2\right) / |\mathcal{P}|}. \tag{2}$$

- *Average relative error* (REL):

$$\mathrm{REL} = \left(\sum\nolimits_{p\in\mathcal{P}} |y_p - \widehat{y}_p|/y_p\right) / |\mathcal{P}|. \tag{3}$$

Tab. 1 compares the RMS, REL and computational time (in seconds) for various sampling ratio and unit distance settings. It shows that the accuracy improves with a higher sample ratio and lower unit distance, but the computational time can grow exponentially. As a result, we set the unit distance to 6 millimeters and the sample ratio to 50% to strike a balance between accuracy and efficiency.

TABLE 1: Comparison of depth interpolation accuracy and computational time using different settings of unit distance and sampling ratio.

Dist.	*Ratio* = 0.1			*Ratio* = 0.2			*Ratio* = 0.5		
	RMS	REL	Time	RMS	REL	time	RMS	REL	Time
20	0.108	0.011	83	0.108	0.011	85	0.108	0.011	85
10	0.092	**0.009**	336	0.092	**0.009**	339	0.091	**0.009**	447
6	0.091	**0.009**	960	0.091	**0.009**	1005	**0.090**	**0.009**	1815
3	**0.090**	**0.009**	4978	**0.090**	**0.009**	6957	**0.090**	**0.009**	15695

TABLE 2: The number of instances and pixel ratios for four types of glass in our glass RBG-D dataset.

Labels	Glass Wall	Window	Glass Door	Glass Railing	Total
Inst. num	3278	404	817	53	4552
Pixel ratio	0.37	0.06	0.2	0.04	0.67

In some areas of the raw depth map, we may not be able to correct the depth values due to insufficient valid depth points for interpolation. Thus, as the final step, we crop the completed depth maps and the corresponding segmentation masks to remove such areas as much as possible.

3.3 Dataset Split

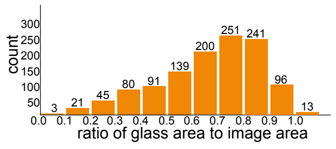
In total, our GW-Depth dataset has 1,200 images from 66 scenes, with varying numbers of images per scene. We randomly split the images into the training set and the test set, while ensuring the scenes in the training set do not appear in the test set. The training set has 55 scenes with 1,018 images, and the test set has 11 scenes with 182 images.

Our dataset mainly features school buildings and shopping malls with four types of glass elements: glass walls, glass doors, glass windows, and glass railings. Tab. 2 shows the total number of instances for each element type and the ratio of total pixels for each type to all image pixels. Glass walls, often found in public buildings and embedded in small frames, make up 37% of all pixels and 72% of all glass element instances. Glass doors are larger than other glass elements, accounting for 20% of all pixels but only 817 instances. Glass windows and guardrails represent smaller proportions in the dataset but contribute to its diversity.

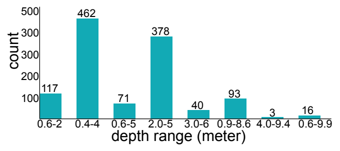
Fig. 7(a) shows a histogram of the ratio between the total area of glass elements and the total image area in our dataset. For most images, glass elements make up more than 60% of the image area. Fig. 7(b) further shows the distribution of depth ranges for glass elements in the images. Most glass elements have depth values no larger than 6 meters, but some glass walls with depth values beyond 6 meters pose more challenges to depth estimation.

3.4 Depth Annotation Analysis

To evaluate the quality of our annotated depth maps in real scenes, we additionally capture 12 images from 6 scenes of glass walls. For each scene, the original glass walls and depths are captured at first. With the same camera position, we then manually cover the glass walls with opaque flats (*e.g.*, thin cardboard and thin mat) and capture the image and depth map thereafter, which is considered as the ground truth of our annotation. We measure the depth differences between the captured depths of covered glass walls and our annotated depths (only the covered areas are computed). The depth errors are 0.087, 0.076, and 0.043 for metrics of RMS, RMS_{log}, and REL, respectively. All inputs for metrics have a unit of meter. Thus, the error of our annotated depths is generally less than 0.09 meters, which is consistent with the thickness of our used opaque flats. Three examples of captured glass walls are shown in Fig. 8.



(a) Glass Area Distribution



(b) Glass Depth Distribution

Fig. 7: Statistics of our GW-Depth dataset. It shows that glass areas make up a significant portion of image pixels, increasing the challenge of depth estimation. Most depth values are within 6 meters, reflecting real-world scenarios.

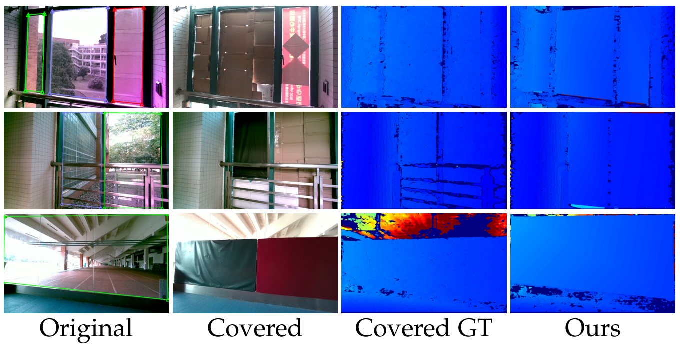


Fig. 8: Examples of captured covered GT and our generated depths. The first column ("Original") shows the original glass walls, where we highlight the areas to conduct depth interpolation by colored polygons. Our annotations show minor differences with the manually covered GT.

4 Methodology

Our proposed model consists of a ResNet encoder and a dual context embedded decoder, which includes a structure context stage and a reflection context stage. The outputs from ResNet's convolution layers (denoted by light green rectangles in Fig. 9(a)) are fed into their corresponding decoder layers (illustrated in Fig. 9(a) with red rectangles for the structure context stage and deep green rectangles for the reflection context stage). Each stage of the decoder operates at different resolutions, with the structure context stage at $\frac{1}{32}$ of the input resolution (*i.e.*, $\frac{W}{32} \times \frac{H}{32}$ for an input in resolution $W \times H$), and the reflection context stage at $\frac{1}{16}$, $\frac{1}{8}$ and $\frac{1}{4}$ of the input resolution respectively.

As shown in Fig. 9(a), the glass boundary line segments are predicted in the structure context stage, and their endpoints are used as sampling points to collect point features representing the glass structure. To create the structure context, we propose a point-guided transformer (PGT) block that takes unrefined global features and sampled glass structure features as input. The attention map within the PGT block is created between global features and each sampled

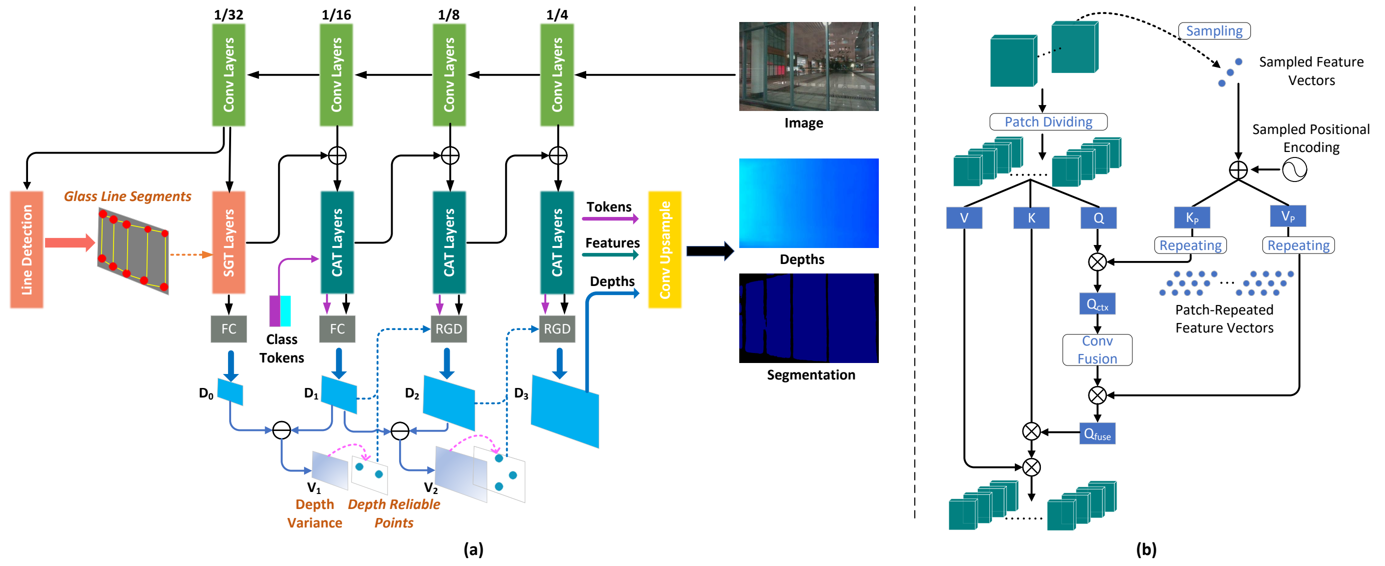


Fig. 9: (a) Overview of our proposed method. With multiscale features from ResNet (in light green), our structure context stage (in orange) predicts glass line segments. The line segment endpoints guided transformer layers (*SGT Layers*) refine features with glass structure context and produce an initial depth estimation $\boldsymbol{D_0}$. The reflection context stage (in deep green) uses refined features and *Class Tokens* to generate higher-resolution spatially refined features. The first *CAT Layers* produce depth estimation $\boldsymbol{D_1}$ at 1/16 input size. The variance map $\boldsymbol{V_1}$, calculated using the square of differences (denoted by $\ominus$) between $\boldsymbol{D_1}$ and upsampled $\boldsymbol{D_0}$, helps construct the reflection context and enables reflection context guided depth estimation (*RGD*) at higher resolutions. Final class tokens, features, and depths combine for full-size depth estimation and glass segmentation via multiple convolution layers (*Conv Upsample*). (b) Architecture of the structure context guided transformer block (SGT).

structure point feature (shown in Fig. 9(b)), which is then used as structure guidance to refine the global features.

As the structure context can be invalid when the glass boundaries either are absent or cannot be reliably recognized, the proposed reflection context stage can provide compensation at higher resolutions. Depth estimation with large variances between adjacent layers (*e.g.*, depth estimations at resolutions of $\frac{W}{16} \times \frac{H}{16}$ and $\frac{W}{8} \times \frac{H}{8}$ where $W \times H$ is the input resolution) are leveraged as anchor points. Using long dependency embedded attention weights in the reflection context, global depths at higher resolutions are estimated through the weighted summation of the anchor point depths.

4.1 Structure Context

Line segment detection for glass walls. Most glass walls have visible outer frames that are roughly coplanar with the glass they enclose. Detecting these glass outlines can help with glass wall identification and depth estimation. Rather than using the dense glass outer frame, we use corner points to represent the glass structure, as each frame segment can be formed by joining two corner points. These corner points are used to create the structure context (as explained in Sec. 4.1).

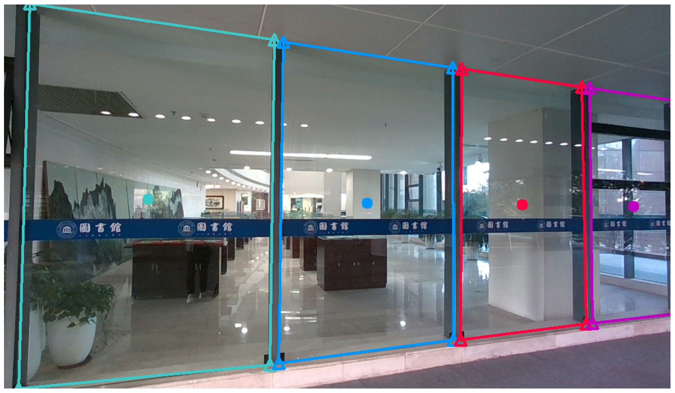
Specifically, we use an end-to-end, proposal-free line segment detection method LETR [47] as our glass line segment detection model. LETR employs a transformer-based multi-scale encoder/decoder and multi-scale prediction heads to generate S line segment predictions with a size of $S \times P_{coor} \times 2$ and their classification scores with a size of $S \times P_{cls} \times 2$, where S is a pre-defined number. Each line segment prediction contains the 2D coordinates for the two endpoints, while the classification scores are the outputs of an MLP layer and indicate the confidence of the predicted line segments being true line segments. We refer to these classification scores as the confidence scores.

In our approach, each line segment is represented by its two endpoints and the center point of the glass wall it belongs to, which helps promote learning of glass structures. Typically, the predefined number S of predicted line segments is larger than the number of ground truth line segments in an image. Therefore, a bipartite (Hungarian) matching operation is performed between the prediction set and the ground truth set to determine which predictions correspond to actual line segments.

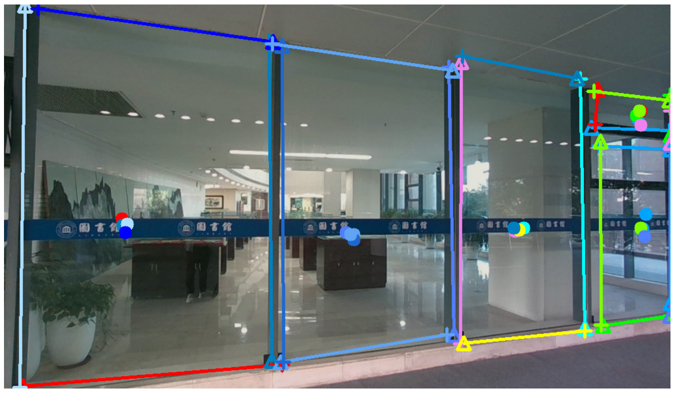
Let the set of ground-truth line segments be denoted by $\{L^{(j)} \mid j = 1, ..., T\}$, and the set of predicted line segments be denoted by $\{\hat{L}^{(i)} \mid i = 1, ..., S\}\}$ where each line segment $\hat{L}^{(i)}$ has a confidence score $c^{(i)}$. We use an injective function $\sigma : \{1, \ldots, T\} \mapsto \{1, \ldots, S\}$ to represent the matching between the ground-truth line segments and their corresponding predictions, such that the ground truth L^i corresponds to the prediction $\hat{L}^{(\sigma(i))}$. We compute the best matching by solving the following optimization problem

$$\min_{\sigma} \sum_{i=1}^{S} \lambda_1 d(\widehat{L}^{(i)}, L^{(\sigma(i))}) - \lambda_2 c^{(i)},$$

where $d(\cdot, \cdot)$ denotes the ℓ_1 distance between the coordinates of line segments, and λ_1, λ_2 are balancing weights. The optimization aims at finding the best bipartite matching that reduces the total distance between the ground truth and the prediction while increasing highest total confidence scores. The problem is solved using the Hungarian algorithm.



(a) Ground-truth Line Segments for Glass Walls



(b) Detected Line Segments

Fig. 10: An example of ground-truth and detected line segments: In the ground-truth (a), each glass instance is enclosed by at least three line segments, with its geometry center displayed as a circle matching the line segment color. Line segment endpoints are marked with triangles. For detected line segments (b), show the line segments with the top 22 classification scores, marking their endpoints with upward triangles and crosses. Each line segment has a predicted glass center, also shown as a circle.

Fig. 10 shows an example of ground-truth line segments and their corresponding predictions.

Glass structure guided transformer block. Our glass structure guided transformer (SGT) block is based on the Swin Transformer (ST) block [48], which computes representations within windows. The SGT block has two branches (see Fig. 9(b)): (1) a vanilla multi-head self-attention embedded patch branch that divides input features into non-overlapping local patches to learn relationships within the patches (shown on the left of Fig. 9(b)); (2) a structure point branch that establishes relationships between glass corner points and patches (shown on the right of Fig. 9(b)). We only use the ST block without the patch merging procedure after layers of ST block as proposed in [48], since it is typically designed for encoder models rather than decoder models. We use a multi-layer perceptron (MLP) layer to process the output of ST blocks.

An SGT block takes in standard CNN feature maps with C channels, $f \in \mathbb{R}^{C\times H\times W}$ (omitting the batch dimension for clarity). Two operations are performed on f for each branch: patch dividing for the patch branch and bilinear sampling for the structure point branch.

For the patch branch, the input features f are divided along the spatial dimension using a patch size of $M\times M$. This conversion changes the spatial size from $H\times W$ to $\frac{H}{N_H}\times\frac{W}{N_W}\times M\times M$, where $\frac{H}{N_H}$ and $\frac{W}{N_W}$ represent the number of patches in the H and W dimensions, respectively. With $N=\frac{H}{N_H}\times\frac{W}{N_W}$, the divided feature patches $f_w\in\mathbb{R}^{C\times N\times M\times M}$ are fed into an MLP to generate key, query, and value feature maps (denoted as K, Q, and V respectively in Fig. 9(b)), each having the same size as f_w.

For the point branch, to select sampling points beforehand, the detected line segments are first sorted by their confidence scores (as described in Sec. 4.1). The top P_L predicted line segments are selected for glass structure guidance. Using the endpoints of the selected line segments, the structure feature vectors $f_p\in\mathbb{R}^{C\times 2\times P_L}$ are sampled accordingly by bilinear interpolation from the input feature map f. Note that the second dimension of f_p has a size of 2 for the two endpoints of each line. Since the position relationship is lost during sampling, we add sinusoidal position embeddings [49] to f_p. Thereafter, f_p is reshaped to $C\times 1\times P$ where $P=2\times P_L$, and we call its last dimension the point dimension. f_p is fed into an MLP layer to generate structure key features K_P and structure value features V_P, both with the same size as f_p.

To build structure context between patches Q and structure feature vectors K_P and V_P, the latter two feature vectors are repeated along the point dimension, expanding their size from $C\times 1\times P$ to $C\times N\times P$. This ensures that each patch of Q (with a size of $C\times N\times M\times M$) has corresponding structure features. Next, a matrix multiplication is performed between Q and K_P across the divided channel groups, which have C channels split into h groups, each having $d=\frac{C}{h}$ channels. This multiplication generates structure context relation embedded query feature maps $Q_{ctx}\in\mathbb{R}^{h\times N\times M^2\times P}$. The structure context Q_{ctx} generation can be described as:

$$Q_{ctx}=\frac{Q\otimes(\mathrm{RP}(\mathrm{Norm}(K_P)))^T}{\sqrt{d}}, \tag{4}$$

where $\mathrm{Norm}(\cdot)$ denotes layer normalization [50] and $\mathrm{RP}(\cdot)$ represents the repetition of point feature vectors for each patch of Q. $\otimes$ denotes the matrix multiplication between Q and the results of $\mathrm{RP}(\cdot)$. To further combine the structure context relation map Q_{ctx} and consider more cross-patch correlations, Q_{ctx} is first reshaped with all patches grouped into one dimension and all point features grouped into another dimension, resulting in $Q_{ctx}\in\mathbb{R}^{h\times\widehat{H}\times P}$, where $\widehat{H}=N\times M^2$. Next, Q_{ctx} is fed into a convolution-based fusion model (denoted as *Conv Fusion* in Fig. 9(b)). This model is partly inspired by object-centric learning [51], which projects global features into predefined slots representing objects or entities in the input. They use a Gated Recurrent Unit (GRU) [52] to globally construct the fusion model. In our SGT block, our goal is to recalibrate the structure context by seeking more cross-patch relations, assigning proper attention weights to previously selected structure points already considering global structure. Therefore, we choose a regular convolution-based model instead of a GRU layer to iteratively perform local-global aggregation:

$$Q^i_{ctx}=Q^{i-1}_{ctx}+\mathrm{GELU}(\mathrm{Norm}(\mathrm{Conv}(Q^{i-1}_{ctx}))).$$

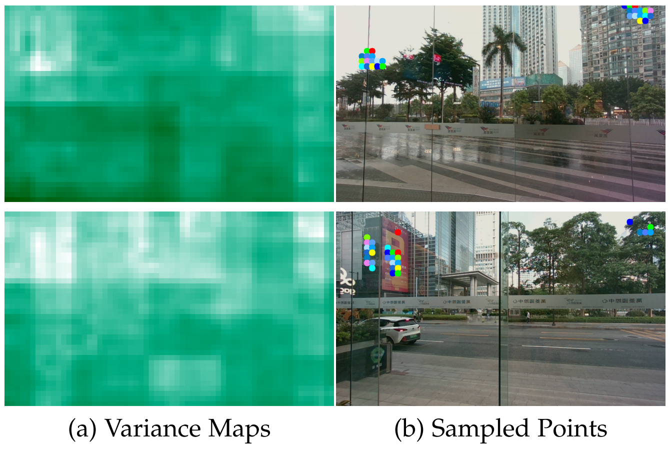


Fig. 11: Two visualized variance maps and sampled depth reliable points. The variance maps are calculated between depth estimations at 1/32 and 1/16 input resolutions. The "Sampled Points" are the points with top 30 values in corresponding variance maps.

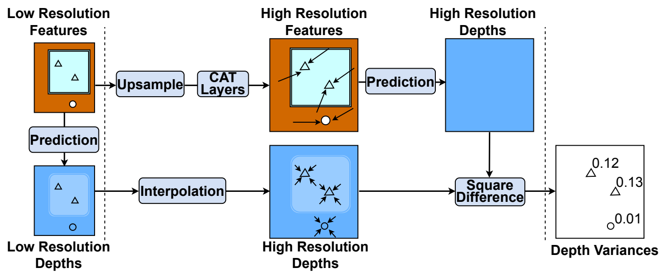


Fig. 12: Diagram for the generation of depth variance. Small triangles and circles in feature maps represent pixels in glass areas (light green) and non-glass opaque areas (brown). "CAT Layers" denotes class attention transformer layers.

The convolution-based aggregation is performed multiple times (three times in our experiments) to generate fusion bias, which is added to the result of the previous step.

After the fusion process, the structure context relation maps Q_{ctx} are normalized using the softmax function across the last dimension, creating structure context attentions. This normalization ensures that the attention coefficients for all structure points add up to one, helping the trained model identify the most relevant structure points for each patch. Next, the output Q_{fuse} undergoes another multi-head self-attention (MSA) [49] operation with the query patches K, and the subsequent calculations resemble those of the standard Swin Transformer [48] block. Finally, the patches are adjusted using the guidance from the glass structure context.

4.2 Reflection Context

In cases where glass structures are missing or hard to detect, we use reflection context to reveal glass walls. As discussed and shown in Fig. 2(b) and Fig. 11, the reflection context takes advantage of the reflection differences seen in low- and high-resolution images. Areas with significant depth changes between different resolutions of depths are more likely to be reflection areas, making their features more correlated to the glass rather than the background scene.

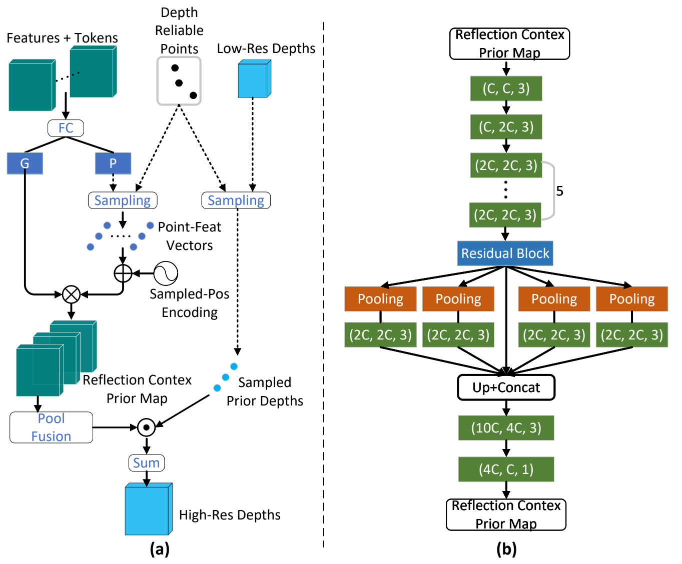


Fig. 13: (a) Reflection context guided depth estimation architecture, with $\odot$ representing element-wise product. (b) Pooling-based convolution fusion model diagram ("Pool Fusion" in (a)). Green rectangles labeled (C_1, C_2, K) represent convolution layers with input channel C_1, output channel C_2, and kernel size K. "Up" denotes bilinear upsampling.

To leverage the reflection context, we first select a set of reliable depth points based on the differences between initial depth estimates at different resolutions (*e.g.*, estimations at $\frac{1}{32}$ and $\frac{1}{16}$ of the input resolution, see Fig. 12). These points serve two purposes: to build the reflection context and to sample glass prior depths from low-resolution depth maps. We predict higher resolution depths using the reflection context weighted sum of sampled prior depths, as shown in the pipeline in Fig. 13(a).

We will now explain the three main components of the reflection context stage: the class attention based transformer layers, the depth variance based reliable point sampling, and depth estimation using reflection context and prior depths.

Class attention based transformer layers. After the structure context stage, in the higher resolution part of the decoder (called dense decoder), we use the patch branch of the transformer block mentioned in Sec. 4.1 and add a class attention layer [53] for depth estimation and glass segmentation. This helps incorporate more glass prior knowledge and address potential incorrect structure context, especially when glass boundaries cannot be reliably recognized. Different from [53], which has single class token initialized for image classification, our dense decoder performs depth estimation and glass segmentation simultaneously. Therefore, we initialize separate depth and segmentation tokens with a spatial size of $\frac{1}{32}$ of the input and 64 channels. We call the transformer layers in the dense decoder Class Attention Transformer (CAT) layers. CAT layers are placed at $\frac{1}{16}$, $\frac{1}{8}$, and $\frac{1}{4}$ of the input resolution (shown as deep green rectangles in Fig 9(a)). We refer to these three layers as the first, second, and third substages of the reflection context, respectively.

The dense decoder takes the output from the structure context stage, which has a size of $\frac{1}{32}$ of the input and 512 channels. The outputs of the three substages include feature maps $F_R^i \in \mathbb{R}^{C_r^i \times H^i \times W^i}$, depth tokens $T_D^i \in \mathbb{R}^{C_t \times H^i \times W^i}$, and segmentation tokens $T_S^i \in \mathbb{R}^{C_t \times H^i \times W^i}$, where $i = 1, 2, 3$ corresponds to the three substages. The feature channels C_r^i are halved between substages, becoming 256, 128, and 64 respectively. The class token channels C_t remain the same. The spatial size H^i and W^i are doubled, with three substages producing output feature maps at resolutions of $\frac{1}{16}$, $\frac{1}{8}$, and $\frac{1}{4}$ of the input, respectively.

Reliable points sampling. Reliable points are sampled from depth variances, which are calculated between two resolutions of depth estimations created by CAT layers. As shown in Fig. 9(a), the first two depth estimations (blue parallelograms denoted by "D_0" and "D_1") are generated directly by multiple MLP layers ("FC"), while the last two depth estimations ("D_2" and "D_3") are produced by depth reliable points guided reflection context ("RGD"). The computation of two depth variance maps ("V_1" and "V_2" in Fig. 9(a)) can be formulated as

$$D_0 = \mathrm{Sig}(\mathrm{MLP}(F_s)), \tag{5}$$
$$D_1 = \mathrm{Sig}(\mathrm{MLP}(F_R^1, T_D^1)), \tag{6}$$
$$D_j = \mathrm{Sig}(\mathrm{RGD}(F_R^j, T_D^j)), \quad j = 2, 3, \tag{7}$$
$$V_k = ||D_k - Up^{\uparrow}(D_{k-1})||^2, \quad k = 1, 2, \tag{8}$$

where F_s are the output features from the structure context stage. F_R^j, T_D^j, and D_j are the backbone features, depth token, and depth estimation at the j-th substage, respectively. $\mathrm{MLP}(\cdot)$ denotes multiple MLP layers that generate depth logits, which are then normalized by a sigmoid function $\mathrm{Sig}(\cdot)$. $\mathrm{RGD}(\cdot)$ is the reflection context guided depth estimation and will be discussed in a later subsection. Depth variance maps V_k (with $k = 1, 2$) are calculated by comparing depths D_k at larger resolutions and bilinearly upsampled (denoted by $Up^{\uparrow}$) depths D_{k-1} at lower resolutions.

The depth variance map V_k is used to sample reliable points, which are then utilized to construct reflection context in the next substage. A simple approach for selecting reliable points is to choose those with the highest variances. However, this may lead to points being densely clustered in one area, resulting in a biased reflection context. To obtain a more diverse sampling, we take inspiration from recent multi-view stereo methods [54], [55]. We first discretize the depth estimation with higher resolution (D_k in Eq. (8)) into predefined number of depth ranges, and then perform the sampling within each depth range. We use the term "bin" for simplicity when referring to depth ranges. Suppose there are N_B bins, and we want to sample S_P points from a depth variance map with T_P total points. After dividing the depth estimation D_k into N_B bins, the b-th bin contains P_b points. We then calculate the number of reliable points S_b to be sampled from the b-th bin based on the ratio of P_b to T_P:

$$S_b = S_P \cdot \frac{P_b}{T_P}.$$

For each bin, we select the points with the top S_b variances as reliable points. If the number of sampled points is less than S_P, we repeat the points to reach the predefined size.

Reflection context guided depth estimation. As shown in Fig. 13(a), we use the selected reliable depth points to sample point feature vectors from depth features (labeled as "P") and prior depths from depth estimations (labeled as "Low-Res Depths"). We multiply global features (labeled as "G" in Fig. 13(a)) with point feature vectors produce a point-to-global correlation embedded reflection context prior map. With the reflection context prior map aggregated weights regarding each sampled point, a new depth map with higher resolution is predicted by taking a weighted average of the prior depths.

Since the point feature vectors are sparsely sampled using bilinear interpolation, it is necessary to explicitly consider long-range dependencies to enhance the reflection context prior map. While convolution-based fusion models are effective for lower-resolution features in the structure context stage, they are less efficient at higher resolutions in the reflection context stage. To save computation and GPU memory, we use multiple layers of average pooling with varying kernel sizes to efficiently capture multi-scale global information, as shown in Fig. 13(b). The input for the reflection context prior map first goes through layers of convolutions and residual blocks [26]. Then, the outputs are fed into four average pooling layers, each followed by an MLP layer. These four outputs are upsampled using bilinear interpolation and concatenated along the channel dimension. Finally, convolution layers process the concatenated features and restore the feature channels to their original inputs.

Afterward, a softmax activation is applied to the reflection context prior map along the point dimension to obtain attention weights from each sampled point. These weights are then multiplied with prior depths and summed over the point dimension to generate a new depth estimation. Since our sampling-based dual contexts focus on long-distance correlations, we use traditional bilinear interpolation and convolution layers to efficiently estimate full-size depth and segmentation, resulting in smooth details. To fuse depth features with backbone features, depth tokens, and previously estimated low-resolution depths, we combine them along the channel dimension and project them to a fixed number of channels (64 in our case) using an MLP layer. After two rounds of bilinear interpolation and convolution layers, we obtain the full-size depth estimation. We use a separate but identical module for glass segmentation.

4.3 Loss Function

We use the scale-invariant pixel-wise depth loss from [15] to supervise multi-scale depth estimation results. For semantic segmentation, we use cross entropy loss. The line segment detection loss is the same as in LETR [47]. Based on the selected positive line segments from the minimized bipartite match, we compute the line segment classification loss using cross entropy loss, and the associated line segment endpoints loss using the ℓ_1 loss.

Specifically, let the semantic segmentation prediction for a pixel z in an image with T_{pixel} pixels be denoted by $\widehat{s}_z$, and the ground-truth label be denoted by s_z. Assuming that M_{line} line segments are selected as positive prediction, and the predicted two endpoints of line segment and classification scores are $\widehat{g}_{e_1}^j$, $\widehat{g}_{e_2}^j$, and $\widehat{g}_c^j$ for line segment j. The corresponding ground-truth values are denoted as $g_{e_1}^j$,

$g_{e_2}^j$, and g_c^j respectively. The semantic segmentation loss and line segment detection loss are defined as

$$\mathcal{L}_{seg} = \left(\sum\nolimits_z H(\widehat{s_z}, s_z)\right)/T_{pixel},$$
$$\mathcal{L}_{line} = \left(\sum\nolimits_j (|\widehat{g}_{e_1}^j - g_{e_1}^j| + |\widehat{g}_{e_2}^j - g_{e_2}^j| + H(\widehat{g}_c^j, g_c^j))\right)/M_{line},$$

where H is the cross entropy loss

$$H(x,y) = -(y\log(x) + (1-y)\log(1-x)),$$

with x being the predicted logits and y being the ground-truth label with a value 0 or 1.

Additionally, we have estimated depth values at four resolutions ($\frac{1}{16}$, $\frac{1}{8}$, $\frac{1}{4}$ and full size). For each resolution r, let n_r be the number of valid depth points in that resolution. The depth estimation loss for the resolution is defined as:

$$\mathcal{L}_{depth}^r = \frac{1}{n_r}\sum_i d_i^2 - \frac{1}{n_r^2}(\sum_i d_i)^2,$$

where d_i is an error measure between the predicted depth p_i and the ground-truth depth $\widehat{p}_i$ of pixel i:

$$d_i = \log p_i - \log \widehat{p}_i.$$

The total loss is defined as

$$\mathcal{L} = \alpha\mathcal{L}_{seg} + \beta\mathcal{L}_{line} + \sum_{r\in\mathcal{R}} \lambda_r \mathcal{L}_{depth}^r,$$

where $\mathcal{R}$ denotes the set of four resolutions, and α, β, λ_r are weights for different losses. We set $\alpha = 2$, $\beta = 1$, and λ_r as 0.25, 0.25, 0.25 and 1 for the resolutions of $\frac{1}{16}$, $\frac{1}{8}$, $\frac{1}{4}$ and full size respectively.

5 Experiments

In this section, we first explain the network structure and training details. Next, we compare our method with other depth estimation approaches. Finally, we investigate the effectiveness of our proposed models with ablation studies.

5.1 Training Details

Our network is implemented using PyTorch [59] and trained in an end-to-end manner. The network backbone is initialized with pretrained weights from DETR [60]. We apply AdamW [61], [62] as the optimizer with weight decay 10^{-4}. The initial learning rate is set to 10^{-5} for training the network backbone and 10^{-4} for all the decoder layers. The learning rate is reduced by a factor of 10 every 70 epochs. The model is trained for 200 epochs on 4 Nvidia RTX 2080Ti GPUs.

5.2 Evaluation Metrics

We evaluate the performance of depth estimation using a set of standard metrics. Let $\mathcal{P}$ be the set of valid depth points in the ground-truth depth map, y_p and $\widehat{y}_p$ be the ground-truth and estimated depth values for a point $p \in \mathcal{P}$, respectively. Then we use the RMS and REL as defined in Eqs. (2) and (3) to measure the accuracy. In addition, we adopt the following metrics in our evaluation:

- *Root mean squared log error* (RMS_{log}):

$$\text{RMS}_{\text{log}} = \sqrt{\left(\sum\nolimits_{p\in\mathcal{P}} (\log(y_p) - \log(\widehat{y}_p))^2\right)/|\mathcal{P}|}.$$

- *Average* $\log_{10}$ *error* (Log_{10}):

$$\text{Log}_{10} = \left(\sum_{p\in\mathcal{P}} |\log_{10}(y_p) - \log_{10}(\widehat{y}_p)|\right)/|\mathcal{P}|.$$

- *Accuracy with threshold*, defined as the percentage of points in $\mathcal{P}$ where the ratio between the estimated and ground-truth depth is smaller than a given threshold t:

$$A_t = |\{p \mid \max(y_p/\widehat{y}_p,\ \widehat{y}_p/y_p) < t\}|/|\mathcal{P}|.$$

We use three thresholds, 1.25, 1.25^2 and 1.25^3 for our evaluation. We denote the three resulting metrics as σ_1, σ_2 and σ_3 respectively, i.e.,

$$\sigma_1 = A_{1.25}, \qquad \sigma_2 = A_{1.25^2}, \qquad \sigma_3 = A_{1.25^3}.$$

For glass segmentation, we adopt the widely used Mean IoU and pixel accuracy [37], [63], [64] as evaluation metrics.

5.3 Comparison with State-of-the-Art Methods

To demonstrate the performance of traditional depth estimation methods [8], [16], [17], [30], [56], [57], [58] on transparent glass walls, we evaluate the models trained on NYU depth V2 [9] and tested on our GW-Depth dataset. The results in Tab. 3 show that their generalization ability is reduced in scenes with glass walls. Additionally, methods that incorporate structure prior from larger complementary datasets [56] or uncertainty [58] derived from image gradients tend to generalize better in these scenarios.

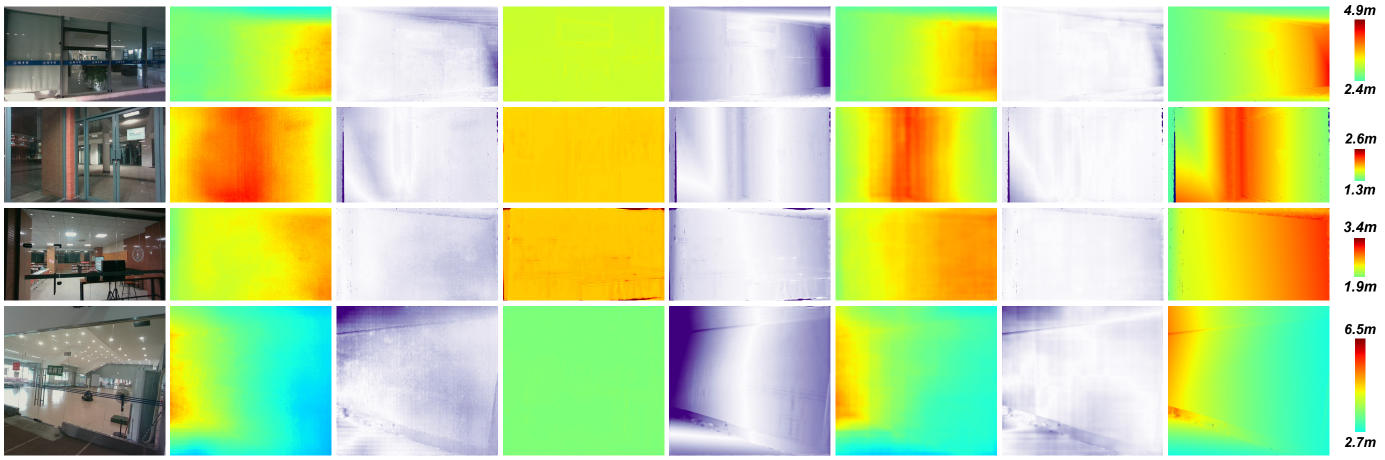
We further compare our method with recent depth estimation methods [8], [17], [27], [30] by fine-tuning them on the GW-Depth training set. The evaluation results on the GW-Depth test set (Tab. 3) show significant improvements compared to models trained on NYU Depth V2, indicating the better annotation quality of our GW-Depth dataset. Our method achieves better performance with fewer parameters compared to other methods. For a fair comparison, we also train our model on NYU Depth V2 without the structure context. Specifically, to adapt our model for depth estimation only, we remove the line detection decoder and replace structure context guided transformer blocks in SGT layers with standard Swin Transformer blocks. We also remove the class token for segmentation in the reflection context. The evaluation results (denoted by “Ours (w/o SC)” in Tab. 4) show acceptable performance despite the limited parameters, with a frame rate of 19.8 FPS on a single 3090Ti GPU.

To further evaluate the applicability of our method, we switch back to using SGT layers in the decoder. The line segments of the NYU Depth V2 dataset are obtained beforehand by the off-the-shelf line segment detection method LETR [47], which provides structure points and classification scores for the SGT layers. We further replace ResNet50 with swin transformer backbone, and the resulting method (“Ours (w/ SGT, SB)”) achieves the best performance among compared methods.

Several results are shown in Fig. 14. The top two rows show glass walls with clear structure context, and the bottom two rows show large glass walls with less noticeable boundaries. Our method effectively estimates their depths, especially at greater distances from the camera, while maintaining consistent depth estimations across the glass walls.

TABLE 3: Depth estimation comparison results on our proposed GW-Depth test set. The second row shows models trained on NYU Depth V2 and evaluated on our GW-Depth test set. The third row displays off-the-shelf model performance fine-tuned on our GW-Depth training set. The last row presents results from our proposed method.

	Method	σ_1 ↑	σ_2 ↑	σ_3 ↑	REL ↓	RMS ↓	RMS$_{\log}$ ↓	Log$_{10}$ ↓	Parameters ($\times 10^6$)
Trained on NYU Depth V2	BTS [16] (arXiv 19)	0.319	0.604	0.797	0.538	1.637	0.507	0.184	47
	SANet [56] (ECCV 20)	0.462	0.845	0.957	0.246	0.71	0.16	0.117	163
	AdaBins [30] (CVPR 21)	0.368	0.655	0.847	0.48	1.378	0.446	0.163	78
	DPT [57] (ICCV 21)	0.303	0.521	0.671	0.771	2.027	0.580	0.219	123
	TransDepth [17] (ICCV 21)	0.394	0.653	0.791	0.589	1.765	0.503	0.176	247
	NeWCRFs [8] (CVPR 22)	0.368	0.633	0.792	0.6	1.957	0.528	0.182	270
	GrUMoDepth [58] (Drop Model) (ECCV 22)	0.439	0.711	0.868	0.357	1.04	0.431	-	-
Trained on GW-Depth	AdaBins [30] (CVPR 21)	0.682	0.946	0.994	0.165	0.435	0.172	0.071	78
	TransDepth [17] (ICCV 21)	0.792	**0.989**	0.994	0.152	0.414	0.159	0.064	247
	NeWCRFs [8] (CVPR 22)	0.851	0.965	0.997	0.123	0.324	0.13	0.052	270
	P3Depth [27] (CVPR 22)	0.864	0.974	0.997	0.115	0.313	0.123	0.05	88
	Ours	**0.9**	**0.989**	**0.999**	**0.1**	**0.276**	**0.112**	**0.043**	**66**



(a) Input (b) NeWCRFs [8] (c) P3Depth [27] (d) Ours (e) GT

Fig. 14: Visualized depth comparisons for methods trained on our GW-Depth dataset. Results predicted by the model with distinct glass structures are in the first two rows, while those with inconspicuous glass structures are in the last two rows. Visualized depths and depth error maps (where darker colors indicate larger errors) are shown from left to right in "(b)", "(c)", and "(d)", respectively.

TABLE 4: Depth estimation comparison results on the NYU Depth V2 test set. All methods are trained on NYU Depth V2. "Ours (w/o SC)" is our model trained without the glass structure context and with the ResNet50 backbone. "Ours (w/ SGT, SB)" is our model trained with SGT Layers and swin transformer backbone [48]. Column "P" lists the number of parameters needed for each method.

Method	σ_1 ↑	σ_2 ↑	σ_3 ↑	REL ↓	RMS ↓	Log$_{10}$ ↓	P
AdaBins [30]	0.698	0.937	0.988	0.178	0.595	0.078	78
SANet [56]	0.899	0.983	0.996	**0.098**	0.376	**0.042**	163
TransDepth [17]	0.900	0.983	0.996	0.106	0.365	0.045	247
P3Depth [27]	0.898	0.981	0.996	0.104	0.356	0.043	88
Ours (w/o SC)	0.894	0.979	0.996	0.119	0.405	0.051	45
Ours (w/ SGT, SB)	**0.917**	**0.990**	**0.998**	**0.098**	**0.339**	**0.042**	237

TABLE 5: Depth estimation comparison for point guided transformer block and reflection context guided depth estimation: "PGT" represents the point guided transformer block replacing the vanilla Swin Transformer block in the baseline model at the smallest scale of 1/32. "S" indicates class attention (depth and segmentation) incorporated at scales 1/16, 1/8, and 1/4. "D" refers to the convolution-based fusion model in the point guided transformer block. "RGD" denotes reflection context guided depth estimation.

Method	σ_1↑	σ_2↑	σ_3↑	REL↓	RMS↓	Log$_{10}$↓	FPS
Baseline	0.847	0.98	0.995	0.12	0.342	0.053	15.4
PGT	0.879	0.985	0.997	0.113	0.317	0.049	13.2
PGT+S	0.877	0.983	0.997	0.108	0.304	0.047	13.1
PGT+S+D	0.894	0.985	0.997	0.102	0.292	0.044	12.9
PGT+S+D+RGD	**0.9**	**0.989**	**0.999**	**0.1**	**0.276**	**0.043**	11.4

5.4 Ablation Studies

We also conduct ablation studies to examine the effectiveness of our proposed model components and determine the optimal parameters.

Baseline model. To demonstrate the effectiveness of our proposed components in the ablation studies, we first establish a baseline model. This model includes a ResNet50 backbone and a depth estimation decoder composed of three types of layers for different resolutions. After the backbone, four Swin Transformer blocks are used to obtain features at 1/32 of the original resolution. Multi-head class attention

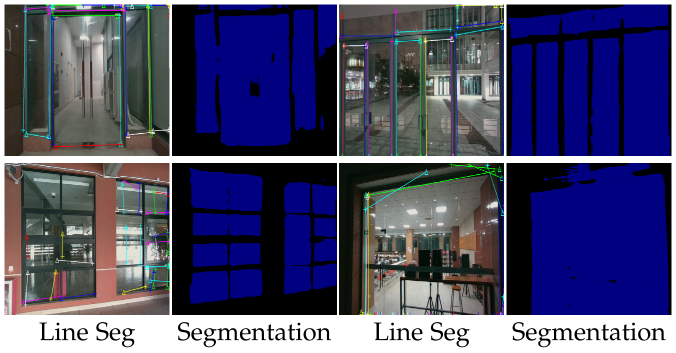


Fig. 15: Glass line detection results with the top 56 scores (first and third columns) and glass segmentation. The first row shows accurate glass line segment detections. The bottom left image has mostly false detections due to heavy reflection from sunshine and small glass frames. Additionally, some inaccurate line segments appear in the bottom right image because of dark surroundings and ambiguous glass frames.

(CA) layers [53] are then employed for features at 1/16, 1/8, and 1/4 of the original resolution, with 2, 2, and 1 layer(s) for each resolution, respectively. Lastly, a convolution-based upsampling module is added to achieve depth estimation and glass segmentation at full resolution.

Structure context guided transformer. To assess the effectiveness of structural prior knowledge, we replace the normal MSA layers at 1/32 of the original resolution in the baseline model with the structure context-guided transformer blocks from Sec. 4.1. We generate a predefined number of line segments (set at 100) from the line segment detection decoder. We then select a fixed number of line segments with the top N classification scores, which include $2 \times N$ endpoints as the glass structure guidance. These points are used to sample glass structure feature vectors, as detailed in Sec. 4.1. The performance comparison between the baseline model and the one with structure-guided transformer is shown in Tab. 5. We observe significant improvements in all metrics, which highlights the effectiveness of glass structure guidance in depth estimation.

Challenging factors such as small glass structures, heavy reflections, or inconspicuous boundaries can lead to incorrect detection of selected line segments, as shown in the second row of Fig. 15. However, glass segmentation is more robust to these conditions due to distinct glass appearances, potentially correcting the adverse influence of incorrect glass structures. Moreover, the convolution-based fusion model adjusts the weights of the structure context-embedded point-to-global attention by considering more cross-patch correlations. Ablation studies on glass segmentation and the convolution-based fusion model are presented in Tab. 5. We observe that the glass segmentation-based model ("PGT+S") reduces depth error, and the fusion model ("PGT+S+D") generally enhances threshold accuracy.

We also examine the impact of the number of selected line segments in Tab. 6. The model achieves the best performance with 28 line segments. One possible reason is that too many line segments may introduce more redundant or incorrect line segments, while too few points cannot represent the complete glass structure. In subsequent experiments, we maintain the number of selected line segments at 28.

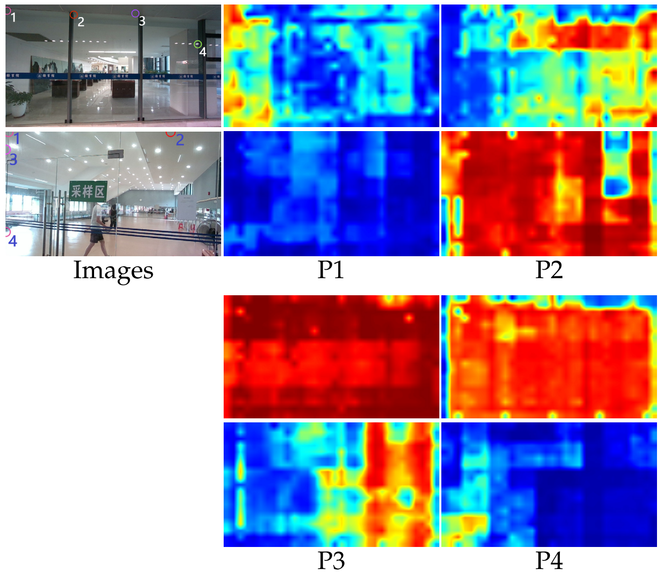


Fig. 16: Visualized attention maps for images with maximum attention weights concentrated on only four glass structure points (marked by colored circles with numbers 1-4 in "Images"). "P1"-"P4" are attention maps computed using maximum values along the channel dimension for points 1-4, respectively. Brighter colors indicate larger values. The comparison shows that most attention weights are clustered on one or two points only.

TABLE 6: Ablation study on the number of line segments for constructing the structure context. This experiment is conducted without considering the reflection context.

# Line Seg.	$\sigma_1\uparrow$	$\sigma_2\uparrow$	$\sigma_3\uparrow$	REL$\downarrow$	RMS$\downarrow$	Log$_{10}\downarrow$	FPS
50	0.886	**0.989**	**0.998**	0.107	0.293	0.046	13.0
28	**0.894**	0.985	0.997	**0.102**	**0.292**	**0.044**	13.1
20	0.89	0.987	0.997	0.105	0.297	0.046	13.1

Despite using multiple glass lines to construct the structure context, we find that for a specific image, the most relevant points tend to be concentrated on a few structure points. As described in Sec. 4.1, the glass structure attention maps have the shape of $C \times H \times W \times N_{pnt}$ (N_{pnt} denotes the number of structure points). By calculating the maximum values along the C dimension, we obtain attention weights between each pixel and each structure point. For the testing set of the GW-Depth dataset, the statistics show that the largest attention values for all images are concentrated on less than 11 points on average. We show two examples of images in Fig. 16, where each image has all its largest structure attention weights concentrated on four structure points only (denoted by colored circles in "Images"). With the brighter colors indicating larger values, Fig. 16 shows that most large attentions are clustered in one or two points (the nearly full red attention maps). We conjecture that this phenomenon is caused by the flat geometry of common glass walls and the limited number of line segments of the glass structures.

Reflection context guided depth estimation. The Reflection

TABLE 7: Ablation study on reflection context guided depth estimation."w/o RGD" denotes our full model without reflection context guided depth estimation ("PGT+S+D" in Tab. 5). "R w/o P" and "R w/ P" denotes "RGD" without or with pooling-based global correlation fusion module.

Areas	Methods	$\sigma_1\uparrow$	$\sigma_2\uparrow$	$\sigma_3\uparrow$	REL$\downarrow$	RMS$\downarrow$	Log$_{10}\downarrow$
Global	w/o RGD	0.894	0.985	0.997	0.102	0.292	0.044
	R w/o P	0.876	0.987	0.998	0.110	0.289	0.048
	R w/ P	**0.9**	**0.989**	**0.999**	**0.1**	**0.276**	**0.043**
Glass	w/o RGD	0.915	0.992	0.998	0.093	0.264	**0.04**
	R w/o P	0.9	0.994	**0.999**	0.102	0.26	0.044
	R w/ P	**0.916**	**0.994**	**0.999**	**0.093**	**0.258**	**0.04**
None-Glass	w/o RGD	**0.891**	0.978	0.996	**0.105**	0.312	**0.046**
	R w/o P	0.869	0.986	**0.999**	0.113	0.323	0.05
	R w/ P	0.886	**0.988**	**0.999**	0.106	**0.31**	**0.046**

TABLE 8: Ablation study on reliable points sampling depth range. For simplicity, the sampling range "(a, b, c)" denotes that sampling are performed within the depth ranges of (min, a], (a, b], (b, c], (c, max), respectively. "(0)" denotes global sampling without depth range.

Range	$\sigma_1\uparrow$	$\sigma_2\uparrow$	$\sigma_3\uparrow$	REL$\downarrow$	RMS$\downarrow$	Log$_{10}\downarrow$
(1,2,3,4,5,6,7,8,9)	0.892	0.986	0.998	0.11	0.302	0.047
(1,3,5,7,9)	0.9	**0.989**	**0.999**	**0.1**	**0.276**	**0.043**
(1,5,9)	0.882	0.986	0.995	0.108	0.298	0.047
(5)	**0.903**	**0.989**	0.998	**0.1**	0.284	**0.043**
(0)	0.894,	0.988	0.998	0.106,	0.294	0.045

Context Guided Depth Estimation (RGD) primarily includes depth reliable points sampling and a pooling-based fusion module. We first compare the performance of models with and without RGD and then conduct an ablation study on the points sampling settings.

The performance of the model with RGD is shown in the last row of Tab. 5. We observe that the RGD-adopted model effectively reduces the RMS error. To specifically investigate the influence of RGD and its pooling component, we compare their performances in glass and non-glass areas in Tab. 7. It is evident that the RGD model primarily improves performance in glass areas. We deduce that the pooling-based global correlation fusion module helps establish long-distance correlations in glass areas while also mitigating potential negative effects for non-glass areas.

The depth reliable points sampling is performed in multiple predefined depth ranges. To find the best setting for depth range, we first keep the number of sampling points at 30 and 80 for the first and second substages of the reflection context stage, respectively. The results for different depth ranges are shown in Tab. 8. We can see that too dense (the second row of Tab. 8) or too sparse ranges (the last row) result in inferior performance. This could be because the optimal sampling intervals can collect more points with a more favorable influence on global depth estimation (see the third row of Tab. 8). Thus, we maintain the depth range of (1,3,5,7,9) as the default in further ablation studies.

Generally, having too many points can result in unreliable depth estimation due to the inclusion of depth-unreliable points, and can also increase computation and memory costs. Conversely, having too few sampling points can result in too sparse point-to-global correlation, which is insufficient to generate global depth estimation. We conduct experiments

TABLE 9: Ablation study on sampling reliable points number for the first and second substages of reflection context stage. For the setting "$(P1, P2)$", $P1$ and $P2$ denote sampling points number for the first and second substages respectively.

Point Number	$\sigma_1\uparrow$	$\sigma_2\uparrow$	$\sigma_3\uparrow$	REL$\downarrow$	RMS$\downarrow$	Log$_{10}\downarrow$
(60, 160)	**0.9**	0.986	0.997	0.106	0.3	0.047
(40, 100)	0.89	0.985	**0.999**	0.106	0.298	0.046
(30, 80)	**0.9**	**0.989**	**0.999**	**0.1**	**0.276**	**0.043**
(15, 40)	0.874	0.98	0.996	0.114	0.314	0.049

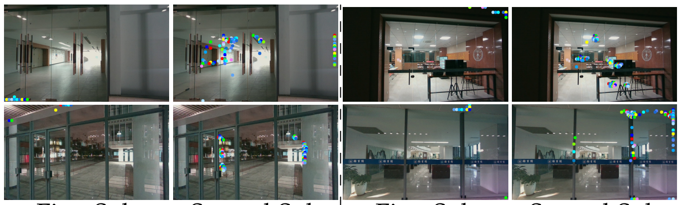


Fig. 17: Visualized depth reliable points in colors on the first and second substages of reflection context stage. 30 and 80 points are sampled for the two substages, respectively.

to determine the optimal number of sampling points for the first and second substages of the reflection context stage. The results are shown in Tab. 9. We notice that a small number of points (30 and 80 points) are sufficient to generate satisfactory global depths, striking a balance between effectiveness and efficiency.

Several images with sampled depth reliable points are shown in Fig. 17. We observe that the sampled points are more likely to locate at non-glass areas on the first substage. On the second substage, more depth reliable points are sampled within the glass areas adjacent to glass outer frames or reflection regions.

5.5 Discussion

Effects of multi-task learning. As our network is devised in a multi-task learning manner, it is necessary to quantitatively verify the influence of each auxiliary task based on our complete model. To evaluate glass line segment detection, we remove the line detection branch and replace the structure context guided transformer blocks with vanilla Swin Transformer blocks. We keep the reflection context guided decoder with the optimal settings from Sec. 5.4, using a depth range of (1, 3, 5, 7, 9) and setting the number of sampling points to (30, 80). Glass segmentation models are in class attention layers alongside depth attention layers. To evaluate glass segmentation, we remove all segmentation related models and only keep the depth attention models. Tab. 10 shows the multi-task learning comparison results. Both auxiliary tasks improve depth estimation, with glass line detection providing a more significant performance boost.

Parameters on different components. The effectiveness of the proposed components is also evident in the number of parameters. Tab. 11 shows the parameter count for different methods and settings. Three conclusions can be drawn from the results. First, the structure context guided transformer ("PGT") improves performance with a 54.2% increase in parameters, which is the main increase in our full

TABLE 10: Ablation study on the influence of auxiliary tasks. "✓" and "-" denotes the network is trained with or without the corresponding task.

Line	Seg.	$\sigma_1\uparrow$	$\sigma_2\uparrow$	$\sigma_3\uparrow$	REL↓	RMS↓	$\text{Log}_{10}\downarrow$
-	-	0.847	0.98	0.995	0.124	0.348	0.054
✓	-	0.881	0.984	**0.999**	0.11	0.301	0.047
-	✓	0.87	0.982	0.995	0.115	0.328	0.05
✓	✓	**0.9**	**0.989**	**0.999**	**0.1**	**0.276**	**0.043**

TABLE 11: Quantity of parameters for the variants of the models with different components and settings. "PGT" denotes the point guided transformer adopted method. "Seg" denotes the incorporating of glass segmentation. "Fusion" denotes the convolution based fusion model adopted in "PGT". The "PGT+Seg+Fusion" model is simplified as "PGT-Full". "RGD" denotes our full reflection context model, and its variant without pooling-based fusion model is denoted as "RGD (no Pool)". The methods with sampled number of depth reliable points $P1$ and $P2$ for the first and second substages are denoted as "RGD ($P1$, $P2$)".

Method	Parameters ($\times 10^6$)	$\sigma_1\uparrow$	RMS ↓
Baseline	38.32	0.847	0.342
PGT	58.47	0.879	0.317
PGT + Seg	59.09	0.877	0.304
PGT + Seg + Fusion	59.10	0.894	0.292
PGT-Full + RGD (no Pool)	59.16	0.876	0.303
PGT-Full + RGD (15, 40)	60.93	0.874	0.314
PGT-Full + RGD (30, 80)	66.21	**0.9**	**0.276**
PGT-Full + RGD (40, 100)	70.39	0.89	0.298
PGT-Full + RGD (60, 160)	87.42	**0.9**	0.3

model. Second, multi-task learning ("PGT+Seg") enhances performance with an added 1.4M parameters, highlighting the importance of a well-annotated dataset for supervised depth estimation. Finally, the reflection context guided depth estimation component requires a local-global reasoning module (pooling-based fusion module) for better point-to-global relations, increasing the network parameters by 7M (comparing "PGT-Full+RGD (no Pool)" to the optimal setting "PGT-Full+RGD (30,80)").

Generalization on glass and transparent objects datasets. We tested our method's generalization by evaluating our model on Trans10K [42], a segmentation dataset containing transparent objects captured in the wild. Fig. 18 shows several qualitative results, indicating successful transparent glass detection and generally accurate depth estimation. It is important to note that our model, trained on only 1,018 images, demonstrates improved generalization thanks to our multi-task learning approach.

We further evaluate our model on the ClearGrasp dataset [11], an RGB-D dataset for transparent objects such as cups and bottles. Due to the large difference in depth scale between GW-Depth (0.4-9.9 meters) and ClearGrasp (0.2-2 meters), our model trained on GW-Depth can effectively predict the depth of supporting planar regions (e.g., tables, basins, etc.), but struggled with transparent objects. The performance comparisons on the ClearGrasp dataset are shown in Tab. 12. Notably, while both ClearGrasp [11] and LIDF [12] use the captured raw depths as input, our method only requires RGB images and achieves comparable

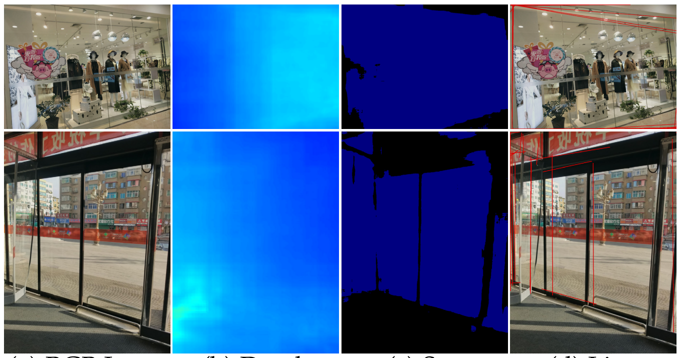


Fig. 18: Our results on transparent objects segmentation dataset Trans10K [42]. The model is trained only on our GW-Depth dataset. (b) & (c): visualized depth estimations and glass segmentations. (d): detected glass line segments with the top 16 confidence scores are displayed in red.

TABLE 12: Performance comparisons on the synthetic split of the ClearGrasp dataset [11]. "Our GW-D" and "Our CG" are our models without glass structure context trained on the GW-Depth and ClearGrasp datasets, respectively. "SPF" denotes the number of seconds taken by each method for processing one image.

Method	$\sigma_1\uparrow$	$\sigma_2\uparrow$	$\sigma_3\uparrow$	REL↓	RMS↓	$\text{Log}_{10}\downarrow$	SPF
CG [11]	0.6943	0.8917	0.9674	0.055	0.041	0.031	1.82
LIDF [12]	0.9479	0.9852	0.9967	**0.017**	**0.012**	**0.009**	0.09
Our GW-D	0.1353	0.2889	0.4543	0.4965	0.4839	0.4405	**0.04**
Our CG	**0.9910**	**0.9991**	**0.9995**	0.060	0.049	0.043	**0.04**

performances (the best accuracy under thresholds σ_1, σ_2, σ_3). Two visualized depths of our method on the ClearGrasp datasets are shown in Fig. 19. This shows the reflection context is also useful in scenes dominated by light refraction.

Limitations and future work. Our approach still has some limitations. In most cases, the required context prior knowledge is readily available. However, in situations where strong light conditions make glass walls ambiguous, our model may incorrectly predict glass walls in enclosed areas where no glass is present. Fig. 20 illustrates this with two examples from the NYU Depth V2 dataset. In the first row, our model predicts reasonable depths for a glass structure with clear lighting. In the second row, it incorrectly predicts

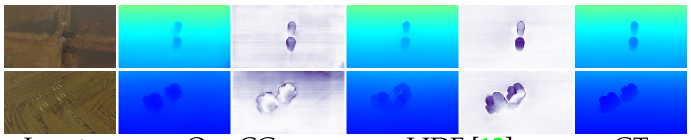


Fig. 19: Visualized depth comparison on the transparent object dataset ClearGrasp [11]. Our model ("Our CG") is trained only with reflection context and without raw depths as input. Both "Our CG" and "LIDF" contain visualized depths and depth error maps (where darker colors indicate larger errors). "GT" denotes the ground-truth depths.

a window frame without glass as a glass wall. Additionally, our model may treat reflections in mirrors as real scenes, as shown in the last row of Fig. 20. These limitations arise because the glass wall scenes in our dataset mainly consist of transparent walls, which has an unbalanced distribution in such cases. This is also a common problem for monocular depth estimation, as the networks might overly rely on the contextual information learned from specific datasets. It highlights the need to improve dataset diversity and further develop new methods to address depth estimation challenges in scenes where depth cameras are unreliable.

To address the above limitations, we plan to capture more diverse indoor scenes that contain both glass walls and mirrors. To make the dataset distributed in a more balanced way, we opt to also capture the common indoor scenes without glass and mirror. To improve the capturing efficiency, we plan to set up a sliding track to carry the depth camera, which will enable us to save time by not having to frequently attach and remove the opaque covers in areas where depth cameras can be unreliable. Furthermore, to leverage existing glass detection datasets, mirror segmentation datasets, and emergent large models like CLIP [65], we plan to design a model that can be pretrained on those datasets or distilled from trained large models, which could potentially help our method more robustly distinguish scenes where depth cameras are unreliable.

6 Conclusion

We created an RGB-D dataset for glass walls and introduced a dual-context embedded method for transparent glass wall depth estimation. The dataset features real-world scenes with detailed annotations, including depth maps, glass segmentations, and glass line segments. Our network uses glass structure context from top-scored line segments as prior knowledge, constructing a local-to-structure attention map. We also utilize reflection context by selecting depth-reliable points based on variance between two estimated depth maps, enhancing depth estimation in scenes with absent or unclear glass structures. The reflection context is embedded in point-to-global attention between depth-reliable point feature vectors and global feature maps, resulting in a higher resolution depth map. Our extensive experiments show the effectiveness of this dual-context design.

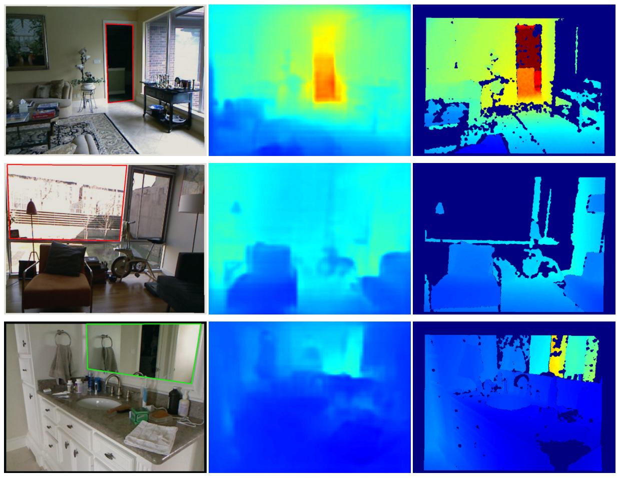


Fig. 20: Our results on two special cases: enclosed areas without glass (first and second rows) and mirror (last row). All images are from NYU Depth V2. “Ours” shows the results of our model trained on the GW-Depth dataset, while “NYU GT” denotes the ground truth depths from NYU Depth V2.

References

[1] B. Mayo, T. Hazan, and A. Tal, “Visual navigation with spatial attention,” in *CVPR*, 2021, pp. 16 898–16 907. 1
[2] F. Zhu, X. Liang, Y. Zhu, Q. Yu, X. Chang, and X. Liang, “Soon: scenario oriented object navigation with graph-based exploration,” in *CVPR*, 2021, pp. 12 689–12 699. 1
[3] Z. Liu, Z. Zhang, Y. Cao, H. Hu, and X. Tong, “Group-free 3d object detection via transformers,” in *ICCV*, 2021, pp. 2949–2958. 1
[4] X. Pan, Z. Xia, S. Song, L. E. Li, and G. Huang, “3d object detection with pointformer,” in *CVPR*, 2021, pp. 7463–7472. 1
[5] P. Li and J. Jin, “Time3d: End-to-end joint monocular 3d object detection and tracking for autonomous driving,” in *CVPR*, 2022, pp. 3885–3894. 1
[6] Y.-N. Chen, H. Dai, and Y. Ding, “Pseudo-stereo for monocular 3d object detection in autonomous driving,” in *CVPR*, 2022, pp. 887–897. 1
[7] X. Ye, M. Shu, H. Li, Y. Shi, Y. Li, G. Wang, X. Tan, and E. Ding, “Rope3d: The roadside perception dataset for autonomous driving and monocular 3d object detection task,” in *CVPR*, 2022, pp. 21 341–21 350. 1
[8] W. Yuan, X. Gu, Z. Dai, S. Zhu, and P. Tan, “Neural window fully-connected crfs for monocular depth estimation,” in *CVPR*, 2022, pp. 3916–3925. 1, 2, 3, 11, 12
[9] P. K. Nathan Silberman, Derek Hoiem and R. Fergus, “Indoor segmentation and support inference from rgbd images,” in *ECCV*, 2012, pp. 746–760. 1, 11
[10] T. Wang, X. He, and N. Barnes, “Glass object localization by joint inference of boundary and depth,” in *ICPR*, 2012, pp. 3783–3786. 1, 3
[11] S. Sajjan, M. Moore, M. Pan, G. Nagaraja, J. Lee, A. Zeng, and S. Song, “Clear grasp: 3d shape estimation of transparent objects for manipulation,” in *ICRA*, 2020, pp. 3634–3642. 1, 3, 15
[12] L. Zhu, A. Mousavian, Y. Xiang, H. Mazhar, J. van Eenbergen, S. Debnath, and D. Fox, “Rgb-d local implicit function for depth completion of transparent objects,” in *CVPR*, 2021, pp. 4649–4658. 1, 3, 15
[13] H. Mei, X. Yang, Y. Wang, Y. Liu, S. He, Q. Zhang, X. Wei, and R. W. Lau, “Don’t hit me! glass detection in real-world scenes,” in *CVPR*, 2020, pp. 3687–3696. 1, 3
[14] J. Lin, Z. He, and R. W. Lau, “Rich context aggregation with reflection prior for glass surface detection,” in *CVPR*, 2021, pp. 13 415–13 424. 1, 2, 3
[15] D. Eigen, C. Puhrsch, and R. Fergus, “Depth map prediction from a single image using a multi-scale deep network,” *NeurIPS*, vol. 27, 2014. 2, 3, 10
[16] J. H. Lee, M.-K. Han, D. W. Ko, and I. H. Suh, “From big to small: Multi-scale local planar guidance for monocular depth estimation,” *arXiv preprint arXiv:1907.10326*, 2019. 2, 3, 11, 12
[17] G. Yang, H. Tang, M. Ding, N. Sebe, and E. Ricci, “Transformer-based attention networks for continuous pixel-wise prediction,” in *ICCV*, 2021, pp. 16 269–16 279. 2, 3, 11, 12
[18] M. Osadchy, D. W. Jacobs, and R. Ramamoorthi, “Using specularities for recognition.” in *ICCV*, vol. 9, 2003, pp. 45–46. 2
[19] T. Zhou, M. Brown, N. Snavely, and D. G. Lowe, “Unsupervised learning of depth and ego-motion from video,” in *CVPR*, 2017, pp. 1851–1858. 2
[20] C. Godard, O. Mac Aodha, and G. J. Brostow, “Unsupervised monocular depth estimation with left-right consistency,” in *CVPR*, 2017, pp. 270–279. 2
[21] C. Godard, O. Mac Aodha, M. Firman, and G. J. Brostow, “Digging into self-supervised monocular depth estimation,” in *ICCV*, 2019, pp. 3828–3838. 3

[22] V. Guizilini, R. Ambruș, D. Chen, S. Zakharov, and A. Gaidon, "Multi-frame self-supervised depth with transformers," in *CVPR*, 2022, pp. 160–170. 3

[23] T.-W. Hui, "Rm-depth: Unsupervised learning of recurrent monocular depth in dynamic scenes," in *CVPR*, 2022, pp. 1675–1684. 3

[24] A. Petrovai and S. Nedevschi, "Exploiting pseudo labels in a self-supervised learning framework for improved monocular depth estimation," in *CVPR*, 2022, pp. 1578–1588. 3

[25] D. Eigen and R. Fergus, "Predicting depth, surface normals and semantic labels with a common multi-scale convolutional architecture," in *ICCV*, 2015, pp. 2650–2658. 3

[26] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *CVPR*, 2016, pp. 770–778. 3, 10

[27] V. Patil, C. Sakaridis, A. Liniger, and L. Van Gool, "P3depth: Monocular depth estimation with a piecewise planarity prior," in *CVPR*, 2022, pp. 1610–1621. 3, 11, 12

[28] Y. Cao, Z. Wu, and C. Shen, "Estimating depth from monocular images as classification using deep fully convolutional residual networks," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 28, no. 11, pp. 3174–3182, 2017. 3

[29] H. Fu, M. Gong, C. Wang, K. Batmanghelich, and D. Tao, "Deep ordinal regression network for monocular depth estimation," in *CVPR*, 2018, pp. 2002–2011. 3

[30] S. F. Bhat, I. Alhashim, and P. Wonka, "Adabins: Depth estimation using adaptive bins," in *CVPR*, 2021, pp. 4009–4018. 3, 11, 12

[31] S. Y. Kim, J. Zhang, S. Niklaus, Y. Fan, S. Chen, Z. Lin, and M. Kim, "Layered depth refinement with mask guidance," in *CVPR*, 2022, pp. 3855–3865. 3

[32] P. Ji, Q. Yan, Y. Ma, and Y. Xu, "Georefine: Self-supervised online depth refinement for accurate dense mapping," in *ECCV*. Springer, 2022, pp. 360–377. 3

[33] Z. Zhang, F. Cole, R. Tucker, W. T. Freeman, and T. Dekel, "Consistent depth of moving objects in video," *ACM Transactions on Graphics*, 2021. 3

[34] S. M. H. Miangoleh, S. Dille, L. Mai, S. Paris, and Y. Aksoy, "Boosting monocular depth estimation models to high-resolution via content-adaptive multi-resolution merging," in *CVPR*, 2021, pp. 9685–9694. 3

[35] C.-Y. Wu, J. Wang, M. Hall, U. Neumann, and S. Su, "Toward practical monocular indoor depth estimation," in *CVPR*, 2022, pp. 3814–3824. 3

[36] D. Xu, W. Ouyang, X. Wang, and N. Sebe, "Pad-net: Multi-tasks guided prediction-and-distillation network for simultaneous depth estimation and scene parsing," in *CVPR*, 2018, pp. 675–684. 3

[37] Z. Zhang, Z. Cui, C. Xu, Z. Jie, X. Li, and J. Yang, "Joint task-recursive learning for semantic segmentation and depth estimation," in *ECCV*, 2018, pp. 235–251. 3, 11

[38] Z. Zhang, Z. Cui, C. Xu, Y. Yan, N. Sebe, and J. Yang, "Pattern-affinitive propagation across depth, surface normal and semantic segmentation," in *CVPR*, 2019, pp. 4106–4115. 3

[39] Y. Lu, S. Pirk, J. Dlabal, A. Brohan, A. Pasad, Z. Chen, V. Casser, A. Angelova, and A. Gordon, "Taskology: Utilizing task relations at scale," in *CVPR*, 2021, pp. 8700–8709. 3

[40] Y. Xu, H. Nagahara, A. Shimada, and R.-i. Taniguchi, "Transcut2: Transparent object segmentation from a light-field image," *IEEE Transactions on Computational Imaging*, vol. 5, no. 3, pp. 465–477, 2019. 3

[41] A. Kalra, V. Taamazyan, S. K. Rao, K. Venkataraman, R. Raskar, and A. Kadambi, "Deep polarization cues for transparent object segmentation," in *CVPR*, 2020, pp. 8602–8611. 3

[42] E. Xie, W. Wang, W. Wang, M. Ding, C. Shen, and P. Luo, "Segmenting transparent objects in the wild," in *ECCV*, 2020, pp. 696–711. 3, 15

[43] H. He, X. Li, G. Cheng, J. Shi, Y. Tong, G. Meng, V. Prinet, and L. Weng, "Enhanced boundary learning for glass-like object segmentation," in *ICCV*, 2021, pp. 15 859–15 868. 3

[44] P. Z. Ramirez, F. Tosi, M. Poggi, S. Salti, S. Mattoccia, and L. Di Stefano, "Open challenges in deep stereo: the booster dataset," in *CVPR*, 2022, pp. 21 168–21 178. 3

[45] Y. Zhu, J. Qiu, and B. Ren, "Transfusion: A novel slam method focused on transparent objects," in *ICCV*, 2021, pp. 6019–6028. 4

[46] Y.-C. Guo, D. Kang, L. Bao, Y. He, and S.-H. Zhang, "Nerfren: Neural radiance fields with reflections," in *CVPR*, 2022, pp. 18 409–18 418. 4

[47] Y. Xu, W. Xu, D. Cheung, and Z. Tu, "Line segment detection using transformers without edges," in *CVPR*, 2021, pp. 4255–4264. 7, 10, 11

[48] Z. Liu, Y. Lin, Y. Cao, H. Hu, Y. Wei, Z. Zhang, S. Lin, and B. Guo, "Swin transformer: Hierarchical vision transformer using shifted windows," in *ICCV*, 2021, pp. 10 012–10 022. 8, 9, 12

[49] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, Ł. Kaiser, and I. Polosukhin, "Attention is all you need," *NeurIPS*, vol. 30, 2017. 8, 9

[50] J. L. Ba, J. R. Kiros, and G. E. Hinton, "Layer normalization," *arXiv preprint arXiv:1607.06450*, 2016. 8

[51] F. Locatello, D. Weissenborn, T. Unterthiner, A. Mahendran, G. Heigold, J. Uszkoreit, A. Dosovitskiy, and T. Kipf, "Object-centric learning with slot attention," *NeurIPS*, vol. 33, pp. 11 525–11 538, 2020. 8

[52] K. Cho, B. Van Merriënboer, C. Gulcehre, D. Bahdanau, F. Bougares, H. Schwenk, and Y. Bengio, "Learning phrase representations using rnn encoder-decoder for statistical machine translation," *arXiv preprint arXiv:1406.1078*, 2014. 8

[53] H. Touvron, M. Cord, A. Sablayrolles, G. Synnaeve, and H. Jégou, "Going deeper with image transformers," in *ICCV*, 2021, pp. 32–42. 9, 13

[54] S. Cheng, Z. Xu, S. Zhu, Z. Li, L. E. Li, R. Ramamoorthi, and H. Su, "Deep stereo using adaptive thin volume representation with uncertainty awareness," in *CVPR*, 2020, pp. 2524–2534. 10

[55] G. Bae, I. Budvytis, and R. Cipolla, "Multi-view depth estimation by fusing single-view depth probability with multi-view geometry," in *CVPR*, 2022, pp. 2842–2851. 10

[56] T. Chen, S. An, Y. Zhang, C. Ma, H. Wang, X. Guo, and W. Zheng, "Improving monocular depth estimation by leveraging structural awareness and complementary datasets," in *ECCV*, 2020, pp. 90–108. 11, 12

[57] R. Ranftl, A. Bochkovskiy, and V. Koltun, "Vision transformers for dense prediction," in *ICCV*, 2021, pp. 12 179–12 188. 11, 12

[58] J. Hornauer and V. Belagiannis, "Gradient-based uncertainty for monocular depth estimation," *arXiv preprint arXiv:2208.02005*, 2022. 11, 12

[59] A. Paszke, S. Gross, F. Massa, A. Lerer, J. Bradbury, G. Chanan, T. Killeen, Z. Lin, N. Gimelshein, L. Antiga *et al.*, "Pytorch: An imperative style, high-performance deep learning library," *NeurIPS*, vol. 32, 2019. 11

[60] N. Carion, F. Massa, G. Synnaeve, N. Usunier, A. Kirillov, and S. Zagoruyko, "End-to-end object detection with transformers," in *ECCV*, 2020, pp. 213–229. 11

[61] I. Loshchilov and F. Hutter, "Decoupled weight decay regularization," *arXiv preprint arXiv:1711.05101*, 2017. 11

[62] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," *arXiv preprint arXiv:1412.6980*, 2014. 11

[63] G. Lin, A. Milan, C. Shen, and I. Reid, "Refinenet: Multi-path refinement networks for high-resolution semantic segmentation," in *CVPR*, 2017, pp. 1925–1934. 11

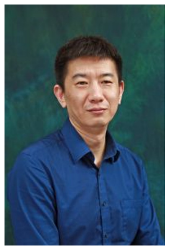
[64] A. Kendall, V. Badrinarayanan, and R. Cipolla, "Bayesian segnet: Model uncertainty in deep convolutional encoder-decoder architectures for scene understanding," *arXiv preprint arXiv:1511.02680*, 2015. 11

[65] A. Radford, J. W. Kim, C. Hallacy, A. Ramesh, G. Goh, S. Agarwal, G. Sastry, A. Askell, P. Mishkin, J. Clark *et al.*, "Learning transferable visual models from natural language supervision," in *International conference on machine learning*. PMLR, 2021, pp. 8748–8763. 16

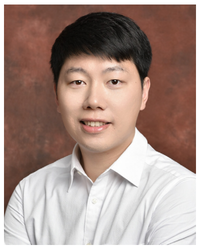
Yuan Liang is currently pursuing the Ph.D. degree with the School of Computer Science and Engineering, South China University of Technology, Guangzhou, China. His research interests include visual understanding and machine vision.

Bailin Deng is a Senior Lecturer in the School of Computer Science and Informatics at Cardiff University. He received the BEng degree in computer software (2005) and the MSc degree in computer science (2008) from Tsinghua University (China), and the PhD degree in technical mathematics (2011) from Vienna University of Technology (Austria). His research interests include geometry processing, numerical optimization, computational design, and digital fabrication. He is a member of the IEEE, and an Associate Editor of IEEE Computer Graphics and Applications.

Wenxi Liu received the Ph.D. degree from the City University of Hong Kong. He is currently a Professor with the College of Computer and Data Science, Fuzhou University. His research interests include computer vision, robot vision, and image processing.



Jing Qin is currently an associate professor in Centre for Smart Health, School of Nursing, The Hong Kong Polytechnic University. His research focuses on creatively leveraging advanced virtual and augmented reality (VR/AR) and artificial intelligence (AI) techniques in healthcare and medicine applications and his achievements in relevant areas has been well recognized by the community. He has developed dozens of VR-based simulators for surgical training, planning, and intraoperative guidance. Recently, he has proposed hundreds of deep learning models for various medical image analysis tasks. He won the Hong Kong Medical and Health Device Industries Association Student Research Award for his PhD study on VR-based simulation systems for surgical training and planning. He won 5 best paper awards for his research on AI-driven medical image analysis and computer-assisted surgery. He served as a local organization chair for MICCAI 2019, technical program committee (TPC) members for dozens of academic conferences, speakers for many invited talks, and referees for most prestigious journals and conferences in relevant fields.



Shengfeng He (Senior Member, IEEE) is an associate professor in the School of Computing and Information Systems, Singapore Management University. He was on the faculty of the South China University of Technology, from 2016 to 2022. He obtained B.Sc. and M.Sc. degrees from Macau University of Science and Technology in 2009 and 2011 respectively, and a Ph.D. degree from City University of Hong Kong in 2015. His research interests include visual understanding and generative models. He is a senior member of IEEE and CCF. He serves on the editorial board of Neurocomputing.