A Multi-modal Garden Dataset and Hybrid 3D Dense Reconstruction Framework Based on Panoramic Stereo Images for a Trimming Robot

Can Pu, Chuanyu Yang, Jinnian Pu, Radim Tylecek, Robert B. Fisher

Abstract

Recovering an outdoor environment's surface mesh is vital for an agricultural robot during task planning and remote visualization. Image-based dense 3D reconstruction is sensitive to large movements between adjacent frames and the quality of the estimated depth maps. Our proposed solution for these problems is based on a newly-designed panoramic stereo camera along with a hybrid novel software framework that consists of three fusion modules: disparity fusion, pose fusion, and volumetric fusion. The panoramic stereo camera with a pentagon shape consists of 5 stereo vision camera pairs to stream synchronized panoramic stereo images for the following three fusion modules. In the disparity fusion module, rectified stereo images produce the initial disparity maps using multiple stereo vision algorithms. Then, these initial disparity maps, along with the intensity images, are input into a disparity fusion network to produce refined disparity maps. Next, the refined disparity maps are converted into full-view (360°) point clouds or single-view (72°) point clouds for the pose fusion module. The pose fusion module adopts a two-stage global-coarse-to-local-fine strategy. In the first stage, each pair of full-view point clouds is registered by a global point cloud matching algorithm to estimate the transformation for a global pose graph's edge, which effectively implements loop closure. In the second stage, a local point cloud matching algorithm is used to match single-view point clouds in different nodes. Next, we locally refine the poses of all corresponding edges in the global pose graph using three proposed rules, thus constructing a refined pose graph. The refined pose graph is optimized to produce a global pose trajectory for volumetric fusion. In the volumetric fusion module, the global poses of all the nodes are used to integrate the single-view point clouds into the volume to produce the mesh of the whole garden. The proposed framework and its three fusion modules are tested on a real outdoor garden dataset to show the superiority of the performance. The whole pipeline takes about 4 minutes on a desktop computer to process the real garden dataset, which is available at: `https://github.com/Canpu999/Trimbot-Wageningen-SLAM-Dataset`.

Keywords

3D reconstruction, stereo vision, disparity fusion, pose graph optimization, point cloud registration, volumetric fusion

1 Introduction

An economical but robust online 3D reconstruction approach for outdoor environments is vital for the remote visualization of the scene and robot task planning. Recovering the dense 3D structure (e.g. mesh) of an outdoor garden with only image input quickly and robustly is challenging because of lighting changes, texture similarity, shadow interference, limited computation and network resources, etc. Figure 1 (a) shows a real outdoor garden for our gardening robot Trimbot's[1] navigation and plant pruning. In real applications, there are two big challenges[2] for image-based dense 3D reconstruction with high fidelity: 1) Movement (rotation or translation) between adjacent

[1] Trimbot2020 project URL: `http://trimbot2020.webhosting.rug.nl/`

[2] For more description about the challenges in the real world, please read the following file: `https://github.com/Canpu999/Trimbot-Wageningen-SLAM-Dataset/blob/main/Real-challenges.pdf`

frames is big because of e.g. a gardening robot's fast speed (1 m/s translation or 90 deg/s rotation), the temporal downsampling ratio of the frames[3] (1/10), and the image sensors' low frame rate (12 FPS); 2) Disparity maps[4] from existing methods in real outdoor environments are not accurate, dense and robust enough because of texture similarity, lighting changes, and shadows.

According to a recent survey (Chen et al., 2022), SLAM[5] systems are classified by the input data source or the sensors used. Figure 2 shows the SLAM classification results. Given that the input data to our proposed framework is from a newly-designed panoramic stereo camera (see Figure 1 (b), Figure D.1 or Figure D.3) and there is no existing similar work as far as we know, we have defined a new branch in the pure visual SLAM class called 'Panoramic Stereo SLAM' with abbreviation 'PS-SLAM'. **Panoramic stereo SLAM is a class of pure visual SLAM methods with panoramic stereo images as input (e.g. our ring of stereo vision cameras to achieve 360° perception)**. Our proposed framework belongs to PS-SLAM which is in the pure visual SLAM area.

Instead of following the existing main-stream pure visual SLAM technique [pose recovery by feature extraction and mapping (e.g. Campos et al. (2021); Schonberger and Frahm (2016)); pose recovery by minimizing the pixel-wise photo-metric error (e.g. Engel et al. (2017))], we start a new approach (slightly similar to the RGB-D SLAM algorithm Kinectfusion (Izadi et al., 2011)) to do pose recovery by using the point clouds rather than the features or pixel intensities (which are sensitive to illumination, scene appearance, and shadows). To guarantee point cloud quality outdoors, a new disparity fusion algorithm is first introduced into the SLAM pipeline, whose outputs are then improved by some practical techniques. To deal with fast motion or rotation of the robot, an innovative multi-stage pose trajectory estimation method with joint information (Algorithm 1) is developed based on loop closure (LC), view switching, and global & local information transition. The integration of multi-level fusion modules, various supporting algorithms and different innovative strategies make the proposed hybrid framework unique and able to cope with the real challenges mentioned above, on which the traditional SLAM frameworks (e.g. Orbslam3 (Campos et al., 2021), Open3D reconstruction system (Zhou et al., 2018), and the commercial software 'ContextCapture') perform badly.

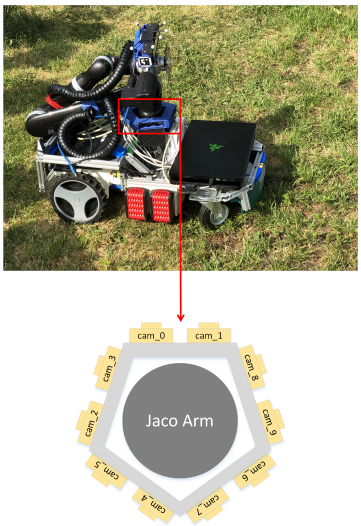
More specifically, to solve the two big challenges above in a real garden for the trimming robot, the TrimBot2020 project team designed a new hardware configuration called the 'panoramic stereo camera' along with a novel 3D reconstruction software framework containing three fusion modules to compute accurate disparity maps, estimate relative pose, and geometrically integrate the maps. Figure 1 (b) shows the panoramic stereo camera which is mounted on the TrimBot2020 robot, and which is primarily used for navigation and visual servoing when the vehicle is near to plants to be trimmed. The diagram in Figure 1 (b) shows the panoramic stereo camera with 5 stereo vision cameras (10 image sensors 'Cam0' - 'Cam9') arranged in a pentagon shape. The panoramic stereo camera streams the synchronized panoramic stereo images (see Figure 1 (c)) from the 10 image sensors ('Cam0' - 'Cam9') for the following three modules to deal with. First, in the disparity fusion module, rectified stereo images are combined to compute the initial disparity maps by multiple stereo vision algorithms. Then the initial disparity maps along with the image information are input into a disparity fusion network to produce a refined disparity map. Next, the refined disparity map is converted into a full-view (360°) point cloud or a single-view (72°) point cloud for the pose fusion module (see Algorithm 1). In the first stage of pose fusion, each two 360° local point clouds are registered by a global point cloud matching algorithm to get the corresponding transformation for the global pose graph's edge, which realizes loop closure (LC) essentially. The global pose graph is then optimized to produce a coarse global pose trajectory of the robot's path through the garden. In the second stage, a refined pose graph is computed based

[3]Because of the mobile network speed, we pick one out of every ten frames to transfer to the server for online 3D reconstruction.

[4]In a stereo configuration, disparity and depth are interchangeable measures: $depth = focal\_length \times baseline/disparity$. When input data is from a depth sensor like Lidar or time of flight sensor, the depth information can be converted into disparity information by using a constant baseline and focal length. Thus, in this paper we regard the two terms as the same and won't distinguish them.

[5]The abbreviations in this manuscript are listed in Appendix A List of Abbreviations. And the frequently-used symbols in this manuscript are listed in Appendix B List of Symbols.

(a) The TrimBot2020 test garden



(b) A panoramic stereo camera

(c) 10 synchronized images from the panoramic stereo camera consisting of 5 stereo vision cameras (10 image sensors in total)

Figure 1: Figure (a) shows the TrimBot2020 outdoor test garden; Figure (b) shows the newly-designed 'panoramic stereo camera' hardware, which is mounted on the trimming robot. The panoramic stereo camera with a pentagon shape consists of 5 stereo vision cameras (10 image sensors, 'Cam0' - 'Cam9'). Figure (c) shows ten synchronized raw images from the 5 stereo camera pairs (the left image sensors in the stereo configuration are color cameras and the right image sensors are gray scale). The images in this figure are from the Trimbot2020 project. For viewing small details in this figure, readers are recommended to view the electronic version.

on the coarse global pose graph. Local point cloud registration along with the coarse global pose trajectory from the first stage jointly update all the available edges in the refined pose graph, which is then optimized to estimate an accurate global sensor pose trajectory. Lastly, in the volumetric fusion module, the global poses of all the nodes in the refined pose graph are used to integrate the corresponding depth maps or point clouds into a volume to produce the surface mesh of the whole garden, which could be used for task planning and the remote visualization. Figure 3 gives an overview of the whole fusion pipeline.

In conclusion, there are three major contributions which could be regarded as the foundation of the PS-SLAM approach. The **major contributions** are :

(1) First real garden dataset (Figure 6) for future PS-SLAM research, which contains the ground truth of the fully-dense depth maps, the semantic maps, the global poses, the rectified stereo images, sparse Lidar scans, and the semantic 3D model;

(2) First hybrid 3D dense reconstruction framework based on panoramic stereo images, which could be regarded as the initial baseline framework (Figure 3) for future PS-SLAM research;

(3) First two-stage full-view-to-single-view global-coarse-to-local-fine pose trajectory estimation method (Algorithm 1), which is robust to fast or large transformations between adjacent frames.

Additionally, there are three notable minor contributions to solve the related problems or improve the related performance in this paper:

1) Theoretical proof (Appendix C.1) that the Frobenius-norm-based transformation difference loss function (Equation 4) is a special case of the maximum likelihood loss function when applied to the pose graph optimization problem;

2) Two practical strategies (in Section 3.1.2) with the theoretical proof (Appendix C.2) to improve the disparity fusion accuracy by setting the maximum distance of interest (denoted by 'Maximum Distance') and up-and-down resolution transformation (denoted by 'High Definition');

3) Three rules (in Section 3.2.2) to optimize the edge set which constrains the pose graph's loss function, boosting the estimated pose's accuracy.

The remainder of this paper is structured as follows. Section 2 presents previous research about SLAM classification, influence factors, and the dataset. Section 3 presents the proposed multi-level fusion framework including the disparity fusion module, the pose fusion module, and the volumetric fusion module. Section 4 describes the real garden dataset and demonstrates the performance of the fusion framework including the disparity fusion module, the pose fusion module and the volumetric fusion module on the real garden dataset. Section 5 presents a discussion and summary of the work.

2 Related Works

This section reviews the existing SLAM classification and positioning the proposed new SLAM framework within it. Secondly, we analyse factors which influence the framework's performance. Lastly, the outdoor datasets used for visual SLAM are reviewed.

2.1 SLAM Classification

According to a recent survey (Chen et al., 2022), SLAM systems are classified by the input data source or the sensors used. Figure 2 shows the classification of different SLAM systems. SLAM systems have been divided into two main categories: Lidar SLAM (e.g. Lego-Loam (Shan and Englot, 2018)) and visual SLAM. Within the visual SLAM category, there are two sub-categories: semantic visual SLAM (e.g. Blitz-slam (Fan et al., 2022)) and traditional visual SLAM. RGB-D SLAM (e.g. Kinectfusion (Izadi et al., 2011), Elasticfusion (Whelan et al., 2016)), pure visual SLAM and visual-inertial SLAM (e.g. Vins-mono (Qin et al., 2018)) constitute the traditional visual SLAM family. Monocular SLAM with a single image sensor (e.g. Colmap (Schonberger and

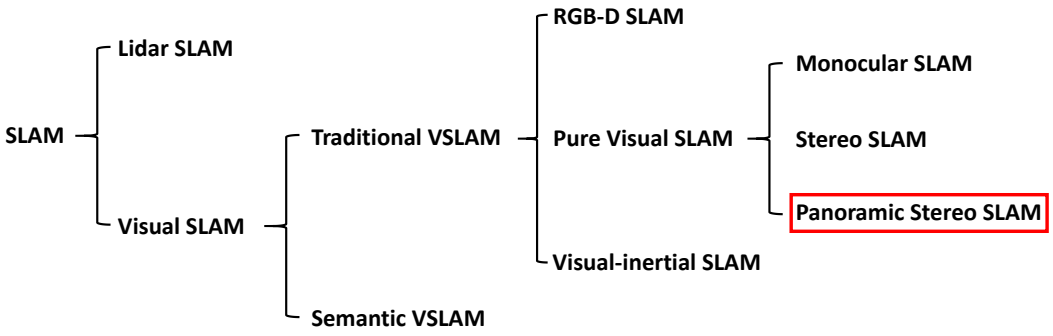


Figure 2: The figure shows SLAM classification (Chen et al., 2022) based on the input data source or the used sensors. As far as we know, our proposed framework is the first work in the branch 'Panoramic Stereo SLAM'.

Frahm, 2016)), stereo SLAM with a stereo vision camera (e.g. Stereo LSD-SLAM (Engel et al., 2015)), and our proposed framework with a panoramic stereo camera belong to the pure visual SLAM class.

While the newly-designed panoramic stereo camera (a ring of 5 synchronized stereo vision cameras mounted in a pentagon shape) extends the category of stereo SLAM, there are significant differences which stem from the specifics of the panoramic arrangement of multiple cameras. Thus, we define a new traditional pure visual SLAM branch 'Panoramic Stereo SLAM' (PS-SLAM), which uses a panoramic stereo camera - a ring of synchronized stereo vision cameras - to input a 360° view. Although there is a stereo panoramic vision system (Guo et al., 2022) that uses a stereo vision camera containing two panoramic vision sensors with a wide field of view (FOV) , that approach still largely follows the classic stereo SLAM concept with some improvements to the stereo SLAM framework. As far as we know, our proposed framework (Figure 3) is the first true PS-SLAM research.

Compared with mainstream panoramic SLAM algorithms (Chen et al., 2021, 2019; Ji et al., 2020; Wang et al., 2022; Zhang and Huang, 2021; Zhao et al., 2022; Zhu et al., 2019), one difference between ours and theirs is that they cannot provide dense global depth information because there is only one monocular camera at each viewpoint inside their panoramic camera whereas in our case multiple stereo images from different perspectives create the panoramic image. The second difference is that they still follow the typical visual SLAM pipeline (e.g. SFM (Schonberger and Frahm, 2016), Orb-slam (Mur-Artal et al., 2015)): feature extraction and mapping, pose estimation by triangulation, loop closure detection, and global optimization, which makes them sensitive to lighting changes unlike our proposed method. Additionally, these algorithms only produce a sparse reconstruction of the scene based on matched feature points as compared to our dense reconstruction.

Meanwhile, some researchers (Ahmadi et al., 2023; Kang et al., 2021) projected the point clouds from a Lidar scanner to the image plane of a panoramic image from a panoramic camera to form a panoramic RGB-D image first. Then, the panoramic RGB-D images were input into OpenVSLAM (Sumikura et al., 2019), an open-source third-party library containing commonly used visual SLAM algorithms (e.g. RGB-D SLAM algorithm in Orb-slam framework (Campos et al., 2021; Mur-Artal and Tardós, 2017)). Compared with our solution, their Lidar sensor is expensive and does not produce a dense point cloud. In summary, the existing panoramic SLAM algorithms follow the traditional visual SLAM pipeline and ours has a different theoretical framework. Based on our newly-designed panoramic stereo camera, we provide an economic dense reconstruction solution which is robust to light changes and scene appearance.

Different from previous frameworks (Chen et al., 2022; Kazerouni et al., 2022; Xu et al., 2022; Zhang et al., 2021), the proposed framework is designed specifically for a panoramic stereo camera set and concentrates on depth quality improvement under some challenging conditions

(illumination changes, similar texture, etc.) and accurate global pose trajectory estimation while coping with the robot's fast motion or rotation. Thus, some new features are proposed to enhance the SLAM framework, such as the disparity fusion module, the field of view switching (360° versus 72°), the novel multi-stage global pose trajectory estimation algorithm 1.

To summarize, the proposed framework is the first to do 3D dense reconstruction in the PS-SLAM research subfield, and so is a baseline to facilitate the progress of PS-SLAM. Compared with the popular SLAM frameworks (e.g. OrbSLAM3 (Campos et al., 2021), Open3D reconstruction system (Zhou et al., 2018), and the commercial software 'ContextCapture'), the integration of multi-level fusion modules, various supporting algorithms, and different innovative strategies makes the proposed framework both unique and capable of performing well even given the two real challenges facing any real outdoor robot: depth data quality and fast robot motion.

2.2 Performance Factors

2.2.1 Depth Quality

Compared with Lidar scanners (expensive and their point cloud is sparse) and ToF sensors (sensitive to infrared light outdoors), etc., image-based depth estimation methods (e.g. stereo vision algorithms) are economical and produce dense depth map indoors and outdoors robustly. In our proposed framework, the stereo vision algorithms estimate the raw disparity maps and the disparity fusion algorithm is used to refine the raw disparity maps from the stereo vision algorithms to get a refined disparity map.

The most well-known classical stereo vision algorithm is the semi-global matching method (Hirschmuller, 2005), which conducts pixel-wise matching using mutual information with a global smoothness approximation. With the rise of deep neural networks, Flownet (Dosovitskiy et al., 2015) is the first to use an end-to-end convolutional neural network to estimate the disparity map between two images. A recent survey (Poggi et al., 2021) gives an overview of the latest progress of stereo vision algorithms. Although stereo vision algorithms have made huge progress recently, a single stereo vision algorithm still has different advantages and disadvantages, and fails to estimate disparity maps accurately at all pixels in all scenes. Disparity fusion is a good method for refining the initial raw disparity maps (from the same viewpoint) from several individual disparity estimation algorithms to estimate a more accurate and robust disparity map based on their complementary properties. The majority of classical disparity fusion methods (Marin et al., 2016; Poggi et al., 2019; Zakeri et al., 2020) share the same pipeline: estimate the disparity map and a confidence map from different sensors, and then use a specific fusion method to fuse the disparity maps using the confidence maps as weights. Because it is hard to estimate the confidence map and disparity distribution accurately, these classical methods have a lower precision compared with deep-learning-based methods (Pu and Fisher, 2019; Pu et al., 2019; Sandström et al., 2022). To highlight, Sdf-man (Pu et al., 2019) is the first to input multiple initial disparity maps with auxiliary information (e.g. RGB, gradients) into the refiner network to produce a refined disparity map. It used a discriminator to classify the refined disparity map and the ground truth disparity map as real or fake to improve the refined disparity map's accuracy.

In existing SLAM system surveys (Chen et al., 2022; Kazerouni et al., 2022; Xu et al., 2022; Zhang et al., 2021), the emerging concept of 'disparity fusion' was not mentioned and we saw no mention of using a disparity fusion algorithm in the SLAM system to improve the 3D dense reconstruction accuracy. We are the first to encode the disparity fusion part in the front end of our proposed SLAM system based on Sdf-man (Pu et al., 2019). Although Sdf-man achieved state-of-art real-time performance in an outdoor garden, its error rate still lies at ˜10 *cm* level. In this paper, we propose two new practical strategies (Section 3.1.2) along with a proof (Appendix C.2) to improve the disparity fusion accuracy, as demonstrated by experiments in Section 4.2.

2.2.2 Pose Accuracy

Compared with image feature matching (e.g. SIFT (Ng and Henikoff, 2003), ORB (Rublee et al., 2011)) to estimate the 6D pose between different views, estimation based on point clouds can produce a more reliable and accurate result. Currently, there are three classes of point cloud matching algorithms to estimate the relative 6D pose. A recent survey (Huang et al., 2021) has an overview. The first class of algorithms (e.g. Gu et al. (2022); Junior et al. (2022); Segal et al. (2009)) is derived from ICP (Besl and McKay, 1992), which calculates the relative 6D pose between two point clouds by finding the closest corresponding points in two point clouds and minimizing their Euclidean distance. Exactly corresponding points seldom exist in the real cases, so the ICP-based methods have low accuracy (and initialization issues). The second class is feature-based algorithms (Ao et al., 2021; Wu et al., 2021; Zeng et al., 2017; Zhou et al., 2016). They extract local descriptors from two point clouds first and then do feature matching to estimate the relative 6D pose between the two point clouds. This class is sensitive to noisy and sparse point clouds which may lead to inaccurate local descriptors and could even make the algorithm collapse when the density is too sparse or the noise is too strong. The third class (Huang et al., 2022; Liu et al., 2021; Myronenko and Song, 2010; Pu et al., 2018) treats point cloud registration as a probability matching problem. They use probabilistic models to describe the geometric distribution of the two point clouds first and then maximize the likelihood of two probabilistic models overlap to calculate the relative 6D pose of the two point clouds. This class of algorithms aligns point clouds more accurately and robustly compared with the previous two classes, but is slow because of their computational complexity.

Pairwise point cloud registration algorithms only compute the relative 6D pose between two local segments within a whole pose trajectory, and do not guarantee estimation of the global optimum of the whole global pose trajectory. That is, registering point clouds sequentially produces a sensor pose trajectory which inherently drifts over time because of the accumulated error. Building an optimized pose graph (Barath et al., 2021; Grisetti et al., 2010; Mendes et al., 2016) could reduce the accumulated error and give an optimized global solution. Ordinarily, loop closure helps resolve this issue, but here the 360 degree point clouds allow many overlapping point sets. Based on the full-view and single-view point clouds, we are the first to develop a two-stage full-view-to-single-view global-coarse-to-local-fine pose trajectory estimation method (Algorithm 1), which can cope well with the fast motion of the real robot outdoors.

2.2.3 Volumetric Fusion

With a range of depth maps and their corresponding global poses, volumetric fusion methods (Curless and Levoy, 1996; Zhou and Koltun, 2013) integrate the surface geometry information into a volume that represents the 3D space in the world coordinate system. Using volumetric integration to build the 3D model of the garden and the marching cube technique (Grosso and Zint, 2022; Lewiner et al., 2003) to extract the surface mesh and its corresponding point cloud is a good option to remove outliers and noise, which could result in good quality when reconstructing the 3D garden model. We use an existing volumetric fusion technique (Section 3.3) for completeness and visualization purposes and do not claim a contribution for this part.

2.3 Dataset

There are multiple outdoor datasets for visual SLAM research, such as Kitti (Geiger et al., 2013; Menze and Geiger, 2015) and Cityscapes (Cordts et al., 2016) for autonomous driving in the city. Besides our dataset, some other datasets (e.g. Alam et al. (2022); Chebrolu et al. (2017); Hu et al. (2022); Polvara et al. (2022)) for agricultural robots have recently been announced. For example, LettuceMOT (Hu et al., 2022) and TobSet (Alam et al., 2022) have only semantic information for lettuce, tobacco crop, weed detection and tracking. The Sugar Beets Dataset (Chebrolu et al., 2017) contains the data from an RGB-D sensor (Kinect v2), a 4-channel multi-spectral camera

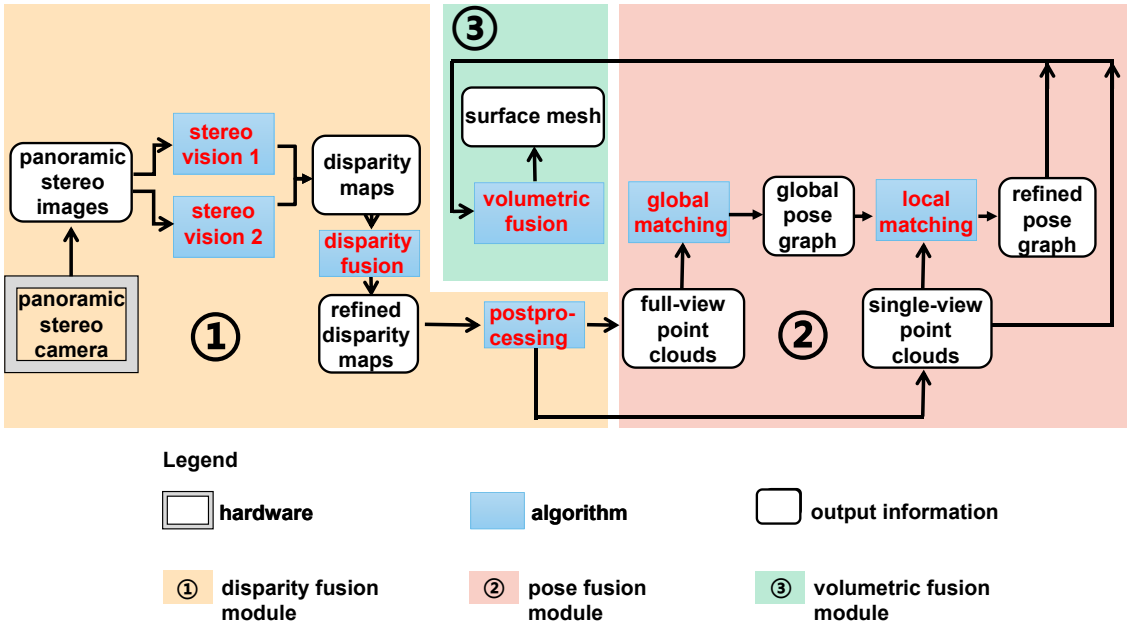


Figure 3: The proposed hybrid multi-level fusion framework based on panoramic stereo images for 3D dense reconstruction

(JAI AD-130GE), two on-board Lidar scanners (Velodyne VLP-16 Puck) and two GPS sensors (Leica RTK GPS and Ublox GPS) as well as wheel encoders to facilitate the research relevant to plant classification, localization and mapping in a sugar beet field. The BLT Dataset[6] (Polvara et al., 2022) contains the data from two RGB-D sensors (ZED), an IMU (RSX-UM7), a 2D Lidar scanner (SICK MRS1000) and a 3D Lidar scanner (Ouster OS1-16) for long-term mapping and localization in a vineyard.

Compared with all the previous datasets, the obvious difference in our dataset is the inclusion of fully dense ground truth depth maps, a fully dense ground truth semantic 3D model and the synchronized joint panoramic stereo information (including RGB & intensity, fully-dense depth, semantic labels, sparse laser scan and global pose) for the first time. Our released dataset could facilitate multiple research topics in SLAM, including sensor calibration, depth estimation, semantic segmentation, pose estimation, and all types of SLAM frameworks (Lidar SLAM, semantic visual SLAM, and traditional visual SLAM). To the best of our knowledge, this is the first public dataset (Section 4.1) in the panoramic stereo SLAM domain.

3 Methodology

Figure 3 shows the proposed hybrid 3D dense reconstruction framework based on images from a panoramic stereo camera rig. Five calibrated binocular cameras inside the panoramic stereo camera (each with a FOV of 72°) stream five pairs of rectified stereo images from the related left and right cameras synchronously into the disparity fusion module. The stereo image pairs are fed into two separate stereo vision algorithms to compute disparity maps in the same view. The disparity maps are used by the disparity fusion algorithm to produce the refined disparity map, which can be converted into the corresponding full-view (FOV = 360°) or single-view (FOV = 72°) point cloud. The point clouds' outliers are removed by post-processing.

In the pose fusion module, the full-view point clouds from different frame times are first registered with each other by a global point cloud matching algorithm for a coarse global pose

[6]BLT dataset: `https://lcas.lincoln.ac.uk/wp/research/data-sets-software/blt/`

estimate, which becomes the corresponding edge's transformation in the global pose graph. Then, the global pose graph is optimized to produce a coarse global sensor pose trajectory. In the second pose fusion stage, the refined pose graph is initialized by the global pose graph. Each pose graph edge's transformation is an input into q local point cloud registration algorithm to obtain a more accurate global pose, which will be used to update each edge's transformation in the refined pose graph. Finally, the refined pose graph is optimized to output a more accurate global pose trajectory, which transforms each single-view point cloud (or depth map) into the global coordinate system in the volumetric fusion module to create a mesh of the whole garden.

In the following, Section 3.1 introduces the disparity fusion module including the stereo vision algorithms, the disparity fusion algorithm, and the post-processing step. Section 3.2 introduces the pose fusion module, including the global pose graph and refined pose graph. Section 3.3 introduces the volumetric fusion module.

3.1 Disparity Fusion Module

3.1.1 Disparity Estimation

In this stage, stereo vision algorithms with complementary properties estimate the initial disparity maps from the stereo images. Based on common sense and experience, classical stereo vision algorithms (e.g. Hirschmuller (2005)) perform better at the edges and small objects while the methods based on deep learning (e.g. Mayer et al. (2016)) perform better at other aspects (e.g. flat planes, close shots). We have chosen DispNet (Mayer et al., 2016) and Semi-global matching (Hirschmuller, 2005) as suitable representatives to compute the initial disparity maps in our project, but other stereo vision algorithms can be used as well. In the following, the initial disparity maps and auxiliary information (intensity and gradient information) are fed into the disparity fusion network to get a refined disparity map.

3.1.2 Disparity Fusion

In order to obtain a more accurate disparity map robustly, fusing disparity maps from multiple sources is a good solution considering cost and performance, under the assumption that the initial disparity inputs are from the same viewpoint at the same time. Fusing multiple input disparity maps to get a refined disparity map output is called disparity fusion, and we base it on Sdf-man (Pu et al., 2019) with some small differences, motivated by a machine learning ensemble approach. As demonstrated in (Pu et al., 2019), the disparity fusion algorithm Sdf-man can refine the initial disparity inputs effectively and produce a more accurate disparity map robustly compared with its initial disparity inputs, even where the disparity inputs are inaccurate on their own.

Similar to the GAN approach, Sdf-man (Pu et al., 2019) consists of two adversarial networks (refiner and discriminator) to perform a mini-max two-player game strategy to make the refiner network output a more accurate disparity map. However, unlike standard GANs (Goodfellow et al., 2014), the input is the initial disparity maps plus intensity and gradient information rather than random noise, and its output is deterministic during inference.

For the sake of readability, we summarize the Sdf-man (Pu et al., 2019) method; more background and details can be found in the original paper.

The refiner neural network R (which is similar to the generator G in (Goodfellow et al., 2014)) is trained to output a refined disparity map that is not classified as "fake" by the discriminator network D. The discriminator network D is trained simultaneously to conclude that the input disparity map from the ground truth is real and the input disparity map from the refiner network R is fake. With a minimax two-player game strategy, it leads the output distribution from the refiner to approximate the real disparity data distribution. The full system pipeline is shown in Figure 4.

To train the refiner network and discriminator network, the following loss function is used in a fully supervised way:

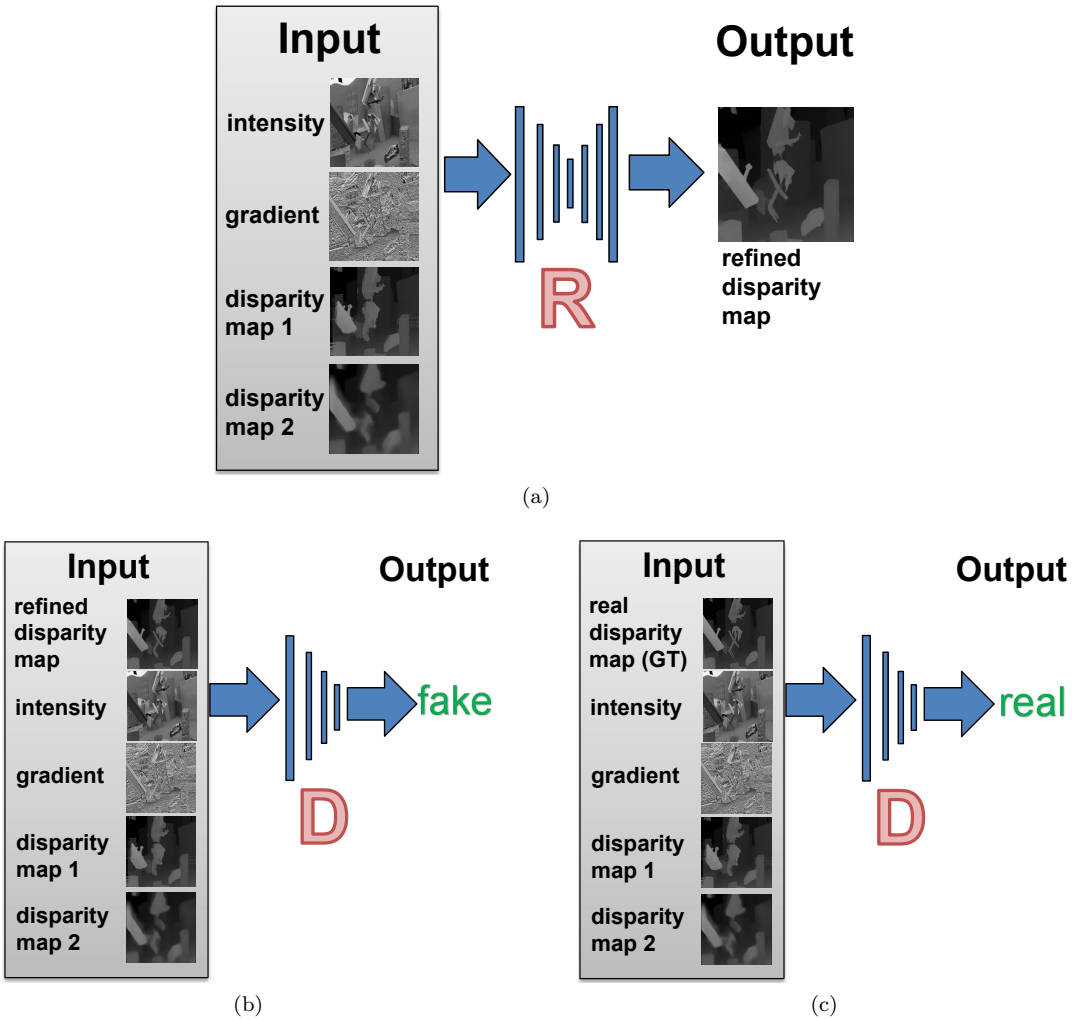


Figure 4: Overview of the disparity fusion algorithm Sdf-man. The refiner network R is trained to map initial disparity maps (*disparity map 1* and *disparity map 2*) from two stereo vision algorithms to the ground truth disparity map based on the corresponding image information (intensity, gradient). The refiner network R attempts to produce a refined disparity map, which is closer to the ground truth. The discriminator network D attempts to discriminate whether its input is the real (real disparity map (GT) from the ground truth) or a fake (refined disparity map from R). The refiner network and discriminator network are updated alternately. (**a**) Refiner: a network to produce a refined disparity map; (**b**) Negative examples (fake): a discriminator network with refined disparity maps as input; (**c**) Positive examples (real): a discriminator network with real disparity maps (GT) as input. For small details in this figure, readers are recommended to view the electronic version.

$$\begin{aligned}\mathcal{L}(R,D) \quad &= \theta_1 \mathcal{L}_{L_1}^{Ld}(R) \\ &+ \theta_2 \mathcal{L}_{sm}^{Ld}(R) + \theta_3 \textstyle\sum_{i=1}^{M} \mathcal{L}_{GAN}^{Ld}(R, D_i)\end{aligned} \tag{1}$$

where θ_1, θ_2, θ_3 are the weight values of the different loss terms. M is the number of scale levels used. R represents the refiner network and D represents the discriminator network. $\mathcal{L}_{L_1}^{Ld}(R)$ is a gradient-based L_1 distance training loss, which applies a bigger weight to the disparity information at the scene edges to avoid blurring at scene edges. $\mathcal{L}_{sm}^{Ld}(R)$ is a gradient-based smoothness term, which is used to propagate more accurate disparity values from scene edges to other areas, assuming that the disparity values of neighboring pixels should be close if their image intensities are similar. $\mathcal{L}_{GAN}^{Ld}(R, D_i)$ is a disparity relationship training loss, which assists the refiner in outputting a disparity map whose distribution is closer to the real distribution. Ld is the labelled data, which is fed in the supervised learning process.

In this paper, we updated the original method (Pu et al., 2019) to improve its performance in the real robot application with the following two practical strategies:

(1) **Maximum Distance** strategy: The disparity fusion network does not require to output all the disparity information in the source stereo images because the mobile robot needs more accurate depths of the nearby surroundings (rather than the remote scene). Thus, a maximum distance threshold max_dist constrains the output of the disparity fusion network rather than the maximum disparity threshold max_disp. More specifically, in the initial stages, at the end of the refiner network, it uses the function $tanh$ to output an intermediate map w and uses the function $initial\_disp = \frac{max\_disp \cdot (w+1)}{2}$ to convert the intermediate map w into the disparity map $initial\_disp$. The intermediate map's size (width, height, channel) is identical to the disparity map's. In this paper, the difference is that we use a modified function $new\_disp = \frac{2fb}{max\_dist \cdot (w+1)}$ to map the $tanh$ output to the new disparity map new_disp where f and b are the focal length and baseline of the stereo vision camera. This strategy effectively reduces the disparity fusion error (see the experiments in Section 4.2). The theoretical proof can be found in Appendix C.2.

(2) **High Definition** strategy: The initial disparity fusion network (Pu et al., 2019) outputs a disparity map with the same resolution as that of the stereo images, which will result in small details being lost in the fused disparity map. To produce a more detailed result in the fused disparity map, the new disparity fusion network is required to output an HD (High Definition) disparity map first. The ratio between the HD width resolution and the initial width resolution is ϕ_w and the ratio between the HD height resolution and the initial height resolution is ϕ_h. Then the HD disparity map is downsampled to the initial resolution (same as the input stereo images). The refiner and the discriminator from Sdf-man are able to adjust their networks adaptively to any resolution of images, as shown in Sdfman (Pu et al., 2019) (Figures 2,3). What is done differently here is: 1) upscaling the data input first and then inputing the upscaled data into the networks to train autonomously; 2) downscaling the disparity output from the refiner network to the resolution of the initial stereo images as the final result. The reason why the up-and-down resolution transformation strategy works is that Sdf-man will include more neurons in the refiner and discriminator network structure to capture more small details autonomously when the input resolution becomes higher. Experiments in Section 4.2 demonstrate this is an effective strategy.

After disparity fusion, a refined disparity map is produced registered to the left view in the stereo configuration. Given that there are still some outliers in the refined disparity map, the refined disparity map is converted into a local point cloud with the outliers removed in the next stage.

3.1.3 Post-processing

The disparity post-processing part consists of three steps: 1) converting the disparity map into a depth map; 2) converting the depth map into a local point cloud using the camera calibration parameters; 3) removing the point cloud outliers.

In the first step, the refined disparity map is converted into the depth map using Equation 2.

$$depth = \frac{focal\_length \times baseline}{disparity} \tag{2}$$

In the second step, the depth map is back-projected into a 3D point cloud using the camera intrinsic parameters (Szeliski, 2010). In Equation 3, (u, v) is the coordinate of the 2D point on the image plane and (X, Y, Z) is the corresponding 3D point in the camera space. f_x, f_y are the focal lengths on the x, y axes and c_x, c_y are the coordinates of the principle point on the x, y axes. D is the depth value.

$$D \begin{bmatrix} u \\ v \\ 1 \end{bmatrix} = \begin{bmatrix} f_x & 0 & c_x \\ 0 & f_y & c_y \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} X \\ Y \\ Z \end{bmatrix} \tag{3}$$

In the third step, any points that 1) have less than N_p neighboring points in a given sphere with the radius *radius* or 2) are farther away from their N_n neighboring points than a threshold distance ratio *dist_ratio* (which is equal to the mean distance to their N_n neighboring points divided by the distance standard deviation to the N_n neighboring points) are treated as outliers and are removed.

After post-processing stage, the local single-view (72°) and full-view (360°) point clouds in the current frame are produced and used in the following pose fusion module.

3.2 Pose Fusion Module

3.2.1 First Stage: Global Pose Graph

As a single-view point cloud (72°) has limited features for point cloud matching, we combine the local point clouds from five views (5 stereo vision cameras) at the same time to form a full-view (360°) point cloud for point cloud registration. This representation improves tracking robustness against fast or big transformations, by using the full-view (360°) point clouds for global registration. The full-view (360°) point cloud X_i^m, the single-view (72°) point cloud X_i^s and their corresponding global pose P_i in the world frame will constitute a pose graph node V_i. The global pose P_i is also the pose of the node V_i. Every pair of nodes V_i and V_j have an edge E_{ij} containing a transformation matrix T_{ij} that aligns their full-view point clouds X_i^m and X_j^m. The nodes V_i and the edges E_{ij} form a global pose graph $G(V, E)$. V is the set of nodes and E is the set of edges.

Every pair of full-view point clouds in different nodes are registered to get the corresponding edge's transformation matrix by using the feature-based fast global registration algorithm (Zhou et al., 2016), which essentially implements loop closure (LC). The global pose graph $G(V, E)$ is then optimized to produce a coarse global pose trajectory $\{P_1, ..., P_n\}$ by minimizing loss:

$$\mathcal{L}(G(V, E)) = \underset{\{P_1,...,P_n\} \in SE(3)^n}{\arg\min} \sum_{(i,j) \in E} ||T_{ij} - P_i P_j^{-1}||_F^2 \tag{4}$$

$||\bullet||_F$ is the Frobenius norm, $SE(3)$ is the special Euclidean group in 3 dimensions and n is the number of the pose graph nodes. The loss function represented by Equation 4 derives from the maximum likelihood estimation formula in (Moreira et al., 2021a) when setting the uncertainty of the translations to be identical to the rotations. We use the optimization method in (Moreira et al., 2021a) to minimize the loss function based on the implementation available at: `https://github.com/gabmoreira/maks`.

The derivation and proof of Equation 4 can be found in Appendix C.1. The same deduction and optimization methods can be applied to Equation 6 below.

3.2.2 Second Stage: Refined Pose Graph

The global pose graph has edges between every pair of nodes possibly, even if there is little or no overlap between their corresponding single-views. This can lead to local distortions. This stage optimizes the global pose graph by using the poses from overlapping views. The refined pose graph $\tilde{G}(\tilde{V}, \tilde{E})$ is initialized by the global pose graph $G(V, E)$.

Since the fast global registration algorithm (Zhou et al., 2016) is not accurate enough compared with local registration algorithms (e.g. GICP (Segal et al., 2009), DUGMA (Pu et al., 2018)), we input each edge's transformation matrix T_{ij} (from E_{ij} in the global pose graph $G(V, E)$) into the local registration algorithm GICP (Segal et al., 2009) as a global initialization to align the corresponding single-view (72°) point clouds[7] X_i^s and X_j^s. The local registration algorithm GICP (Segal et al., 2009) outputs a new estimated transformation matrix T_{ij}^l for the corresponding edge.

Then, every pair of point clouds X_i^s and X_j^s are transformed into the same coordinate system using the transformation matrix T_{ij}^l. We calculate the number of the corresponding point pairs within a distance threshold (similar to finding corresponding closest point pairs in ICP). The overlap percentage of one point cloud after registration is equal to the number of the corresponding pairs divided by the number of the points in the point cloud. The overlap percentage of the pair of point clouds after registration is equal to the overlap percentage of the point cloud with the fewest points. Based on the registration results above and the coarse global pose trajectory from the first stage, the edges $\tilde{E}_{ij}$ in the refined pose graph are updated using the following three rules:

(1) **Prune**: If the overlap percentage of the two point clouds after registration is lower than the threshold OL_{min}, prune the edge (remove the edge between the two nodes).

(2) **Update**: If the overlap percentage of the two point clouds after registration is higher than threshold OL_{max} and if the transformation $P_i P_j^{-1}$ (whose 6D pose is denoted as the 6D vector $\vec{v}_P$) between the two nodes (V_i, V_j) is similar to the newly calculated transformation T_{ij}^l (whose 6D pose is denoted as the 6D vector $\vec{v}_T$), update the edge.

(3) **Keep**: As for the 'else' case, keep but do not update the edge transformation.

Equation 5 gives the precise logic for the three rules above:

$$\tilde{T}_{ij} = \begin{cases} Null & \beta < OL_{min} \quad [rule1] \\ T_{ij}^l & \beta > OL_{max} \ \& \ |\vec{v}_P - \vec{v}_T| < \vec{v}_{th} \quad [rule2] \\ T_{ij} & else \quad [rule3] \end{cases} \tag{5}$$

In Equation 5, $\tilde{T}_{ij}$ is the transformation matrix of the edge $\tilde{E}_{ij}$ in the refined pose graph. β is the overlap percentage of the two point clouds after registration. $\vec{v}_{th}$ is 6D pose threshold in vector format and $|\bullet|$ means getting the absolute value of each element to form a new vector. $Null$ denotes "deleting this edge". The two rules (Prune and Update) act on the edge set to constrain the loss function - Equation 6. An accurate constraint could give a more accurate global pose estimation, which is demonstrated by the ablation study in Section 4.3.1. After edge refinement, the refined pose graph $\tilde{G}(\tilde{V}, \tilde{E})$ is optimized using Equation 6 to produce a more accurate global pose trajectory $\{\tilde{P}_1, ..., \tilde{P}_n\}$. The refined accurate global pose $\tilde{P}_i$ of each node $\tilde{V}_i$ and their corresponding single-view point cloud (or depth map) will be used in the volumetric fusion process to construct the surface mesh of the whole garden.

$$\mathcal{L}(\tilde{G}(\tilde{V}, \tilde{E})) = \underset{\{\tilde{P}_1, ..., \tilde{P}_n\} \in SE(3)^n}{\arg\min} \sum_{(i,j) \in \tilde{E}} ||\tilde{T}_{ij} - \tilde{P}_i \tilde{P}_j^{-1}||_F^2 \tag{6}$$

To conclude, we have proposed a two-stage full-view-to-single-view global-coarse-to-local-fine pose trajectory estimation method in this subsection. We name this proposed method for pose tra-

[7] When using the extrinsic transformation matrices to merge the point clouds from the five stereo vision cameras on the camera ring, the error from the extrinsic parameters will cause the full-view (360°) point cloud to be not as accurate as the single-view point cloud.

jectory estimation as 'Multi-stage Pose Trajectory Estimation with Joint Information (MPTEJI)'. Algorithm 1 shows the pseudocode of the proposed algorithm MPTEJI, which gives a formal overview of the whole proposed method.

Algorithm 1 Multi-stage Pose Trajectory Estimation with Joint Information (MPTEJI)

Input: single-view (72°) and full-view (360°) point clouds

procedure WITH **full view** ▷ 1^{st} *stage*
  global registration (feature-based) → Loop Closure
  coarse pose graph $G(V, E)$ ← Equation 4
procedure WITH **single view** ▷ 2^{nd} stage
  pose graph inheritance ← $G(V, E)$
  local registration (ICP-based) ← T_{ij}
  edge refinement ← Equation 5
  refined pose graph $\tilde{G}(\tilde{V}, \tilde{E})$ ← Equation 6

Output: an accurate global pose trajectory $\{\tilde{P}_1, ..., \tilde{P}_n\}$

3.3 Volumetric Fusion Module

Fusing the range images (containing depth information) into a voxel-based volumetric scene representation is called volumetric fusion (Curless and Levoy, 1996). The refined accurate global pose trajectory $\{\tilde{P}_1, ..., \tilde{P}_n\}$ gives where to integrate the associated RGB-D range images projected from the single-view point clouds into a voxel-grid-based TSDF (Truncated Signed Distance Field) volume. The value of each voxel here represents the signed distance to the closest surface interface in the global space, which is in turn used to obtain the mesh of the reconstructed scene, using the marching cubes (Lorensen and Cline, 1987) algorithm.

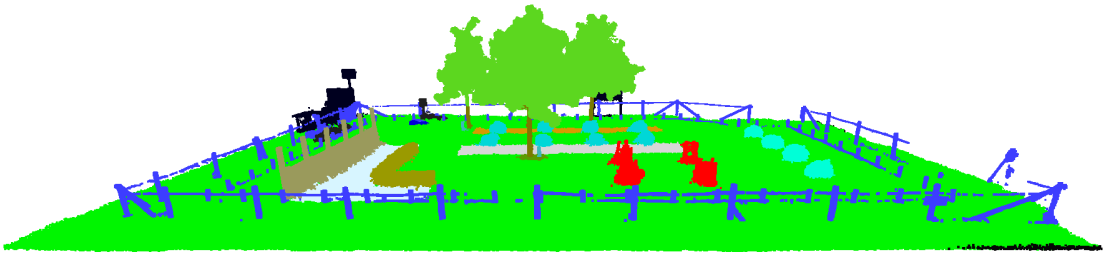
More specifically, the single-view point clouds are projected back into the image planes to get the related depth maps first, creating again RGB-D images. We use the refined single-view point clouds to compute the depth maps rather than use the original depth maps from the disparity fusion (Section 3.1.2) directly because the single-view point clouds after the third step 'outlier removal' in the post-processing section (Section 3.1.3) are more accurate. Then the pairwise data (the RGB-D images and the corresponding global poses) are integrated into the global TSDF volume using the technique from Izadi et al. (2011); Zhou and Koltun (2013). Finally, we extract the surface mesh using marching cubes (Lewiner et al., 2003; Lorensen and Cline, 1987), based on a publicly available implementation[8].

The volumetric fusion module produces a smooth and watertight 3D mesh of the reconstructed scene in the global coordinate system. The corresponding dense 3D point cloud of the reconstructed scene can be produced by extracting all the vertexes of the 3D mesh above. Simply put, the volumetric fusion performs like a weighted average filter in the 3D global space to reduce the noise and remove the outliers from multiple local segments by using the joint global information in the global coordinate system. That is the reason why we use the volumetric fusion to extract the mesh and the corresponding point cloud sequentially, rather than stitching the single-view point clouds together using their corresponding pose directly.

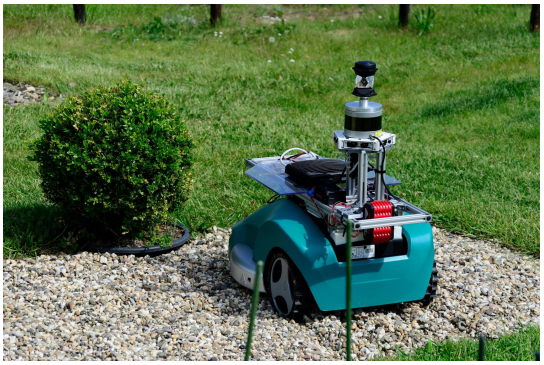
4 Experiments

All the experiments in this section are conducted on a machine with Intel Core i7-12700KF processor (12 cores, 20 threads, 25 MB cache, up to 5 GHz) and Nvidia GeForce GTX 1080 Ti. Section 4.1 gives the description of the real outdoor garden dataset we released and used in this

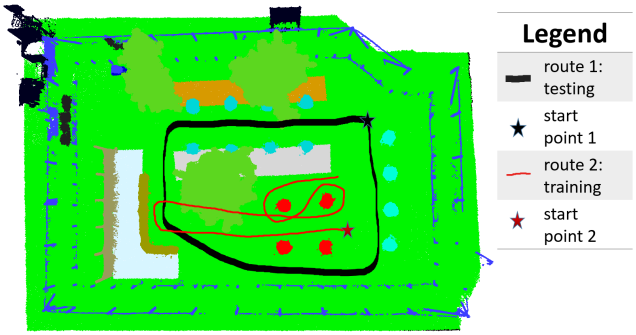
[8] `https://github.com/qianyizh/ElasticReconstruction/tree/master/Integrate`



(a) Manually labeled 3D semantic model of the whole garden. Legend: grass (bright green), rail fence (blue), tree (dark green), hedge (brown), board fence (soil color), rose (red), boxwood (dark blue), potted plant (light blue) (Sattler et al., 2017)



(b) The robot platform used for collecting data



(c) Route for the training & testing dataset in the garden

Figure 5: Figure (a) shows the 3D model of the real garden; Figure (b) shows the robot platform for collecting data; Figure (c) shows the routes for training and testing dataset.

paper. Section 4.2 evaluates the performance improvement of the disparity fusion module quantitatively compared with the initial disparity inputs (Hirschmuller, 2005; Mayer et al., 2016), the ground truth of DSF (Poggi and Mattoccia, 2016) and the initial version of Sdf-man (Pu et al., 2019). Section 4.3 evaluates the global pose trajectory's accuracy from the pose fusion module quantitatively compared with ORB-SLAM3 (Campos et al., 2021) and the "reconstruction system" in the latest version (0.15.1) of Open3D (Zhou et al., 2018). Section 4.4 gives a view of the reconstructed point cloud from the volumetric fusion module qualitatively and quantitatively compared with Open3D (Zhou et al., 2018).

4.1 Dataset Description

Figure 5 shows the 3D model of the outdoor garden, the robot platform and the route path for collecting the raw data. All the data in our dataset were recorded within the same half day to avoid interference from vegetation growth. The raw data is from the "test_around_garden" bagfile[9] in the Trimbot Garden 2017 dataset (Sattler et al., 2017; Tylecek and Fisher, 2020). The raw data was divided into two parts in the post-processing step: one for network training and one for testing. Figure 5 (c) shows the robot navigation path for the training and testing datasets. In Figure 5 (c), the "route 1" trajectory (black loop curve) around the whole garden is for the SLAM testing and the "route 2" trajectory[10] (red curve) is for the network training (e.g. depth estimation, semantic segmentation, etc.). We use a robot (See Figure 5 b) equipped with a ring of 5 stereo vision cameras (for live operations), Velodyne Puck (VLP-16) Lidar sensor (for sparse lidar scans

[9] `https://www.research.ed.ac.uk/en/datasets/trimbot2020-dataset-for-garden-navigation-and-bush-trimming`.
[10] All the scenes in "route 1" can be seen in "route 2".

collection), STIM300 IMU sensor and Topcon PS Series Robotic Total Station position tracking system (for ground-truth positions) to collect the raw images, sparse Lidar scans and the global pose of the robot. The raw stereo vision images are calibrated and rectified using the Kalibr package[11]. The sparse Lidar scan from the Velodyne Puck (VLP-16) Lidar sensor is projected to the camera plane of each left camera in the 5 stereo settings. Robot navigation poses were recorded in the coordinate system of Topcon PS Series Robotic Total Station along with STIM300 IMU sensor first and then transformed into each image sensor's global pose. Structure-from-motion (Schonberger and Frahm, 2016) is used to refine each image sensor's pose subsequently. The 3D model of the whole garden is collected using Leica ScanStation P15 equipment and is semantically labelled manually. Figure 5 (a) shows the semantic 3D model of the whole garden. Using the semantic 3D model and each camera's pose in the garden, the dense depth map and semantic map are acquired by projecting the semantic 3D model into each camera's plane. More details about the data collection process can be found in Appendix D.

Table 1: Parameters of the *Trimbot Wageningen SLAM Dataset*

Parameter Name	**Parameter Value**
The number of panoramic stereo camera rigs	1;
The number of stereo vision cameras	5;
The number of image sensors	10;
The number of panoramic frames - 360°	In the training subset: 68; In the test subset: 67;
The number of stereo vision frames - 72°	In the training subset: 340; In the test subset: 335;
Image resolution	752 × 480 pixels (width × height);
The mean relative pose between adjacent frames ([translation on x axis, translation on y axis, translation on z axis, roll, pitch, yaw])	[0.29 m, 0.21 m, 0.00 m, 9.04 deg, 0.97 deg, 1.16 deg];
The standard deviation of the relative pose between adjacent frames ([translation on x axis, translation on y axis, translation on z axis, roll, pitch, yaw])	[0.18 m, 0.18 m, 0.00 m, 13.32 deg, 0.76 deg, 0.89 deg];
The maximum translation value on each axis between adjacent frames	X axis:0.47 m; Y axis: 0.67 m; Z axis: 0.02 m;
The maximum rotation value on each axis between adjacent frames	X axis: 81.64 deg; Y axis: 3.21 deg; Z axis: 4.46 deg;
Data support	RGB — intensity, dense depth, sparse lidar, semantics, pose, point cloud, calibration.

Figure 6 shows frames from the new dataset "The Trimbot Wageningen SLAM Dataset", which is the augmentation of the Trimbot Garden 2017 dataset used in the semantic reconstruction challenge of ICCV 2017 workshop "3D Reconstruction meets Semantics" (Sattler et al., 2017). In the new Trimbot Wageningen SLAM dataset, we release all the rectified images from the 10 image sensors (5 stereo vision cameras) ranging from cam_0 to cam_9 (See Figure 1b for the position of the 10 image sensors). The newly released sparse Lidar scan (from the onboard Lidar sensor - Velodyne Puck (VLP-16)), dense depth map and semantic map are in the coordinate system of each left image sensor (Cam 0, Cam 2, Cam 4, Cam 6, Cam 8). Each image sensor's global pose in the garden, their intrinsic parameters and distortion models are available in the new dataset. We subsample one out of every 10 frames from the initial raw data bagfile to form the new dataset. Table 1 lists the key dataset properties and their corresponding values in the Trimbot Wageningen SLAM Dataset. Figure 6 gives an overview of the new dataset. See the dataset website for more details: `https://github.com/Canpu999/Trimbot-Wageningen-SLAM-Dataset`.

[11] https://github.com/ethz-asl/kalibr

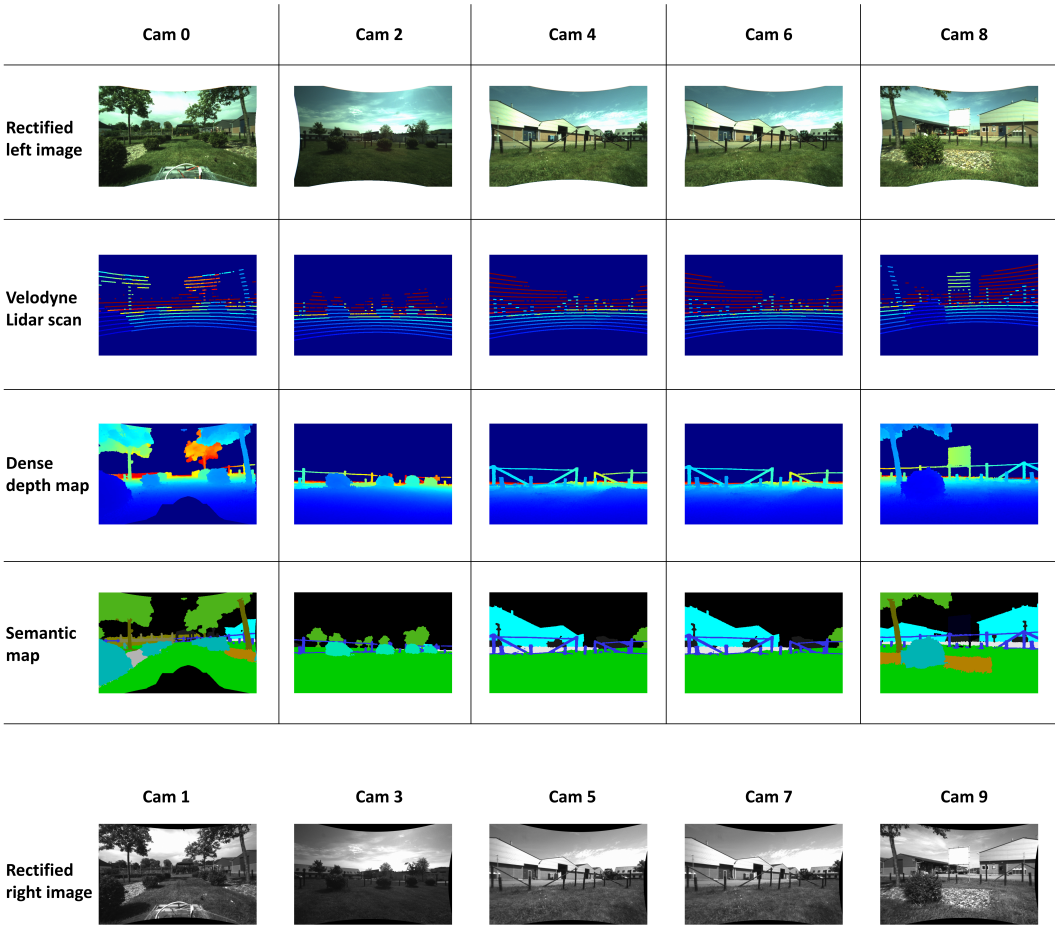


Figure 6: Trimbot Wageningen SLAM Dataset

4.2 Disparity Fusion Module

In the disparity fusion module, we use the SGM (Hirschmuller, 2005) (with Matlab implementation[12]) and Dispnet[13] (Mayer et al., 2016) stereo vision algorithms to get the initial disparity maps. With the initial disparity maps and auxiliary information (left intensity image and left gradient information), we train our supervised disparity fusion network on our outdoor real garden dataset - "Trimbot Wageningen SLAM Dataset". All 340 samples ($\approx 50\%$) in the training set are used to train and all 335 samples ($\approx 50\%$) in the test set are used to test. The initial supervised method (Pu et al., 2019) is named "Sdfman-initial" and the updated method using the two practical strategies presented in Section 3.1 is named "Sdfman-star". The parameter max_dist is set to 5 meters[14]. The parameters ϕ_w and ϕ_h (resolution ratio between the HD and initial image width and height) are set to 2. Disparity fusion algorithms DSF (Poggi and Mattoccia, 2016) and "Sdfman-initial" (Pu et al., 2019) are compared to the new method "Sdfman-star". Additionally, an ablation study is conducted by adding two internal comparison algorithms ("Sdfman-max-dist" and "Sdfman-HR"). "Sdfman-max-dist" is an internal comparison algorithm that only applies the "Maximum Distance" strategy to "Sdfman-initial". "Sdfman-HR" is an internal comparison algorithm that only applies the "High Definition" strategy to "Sdfman-initial". Table 2 summarizes the algorithms' names with the corresponding strategies.

Table 2: Algorithm definition.

Algorithm Name	**Strategy**
Sdfman-initial	Default
Sdfman-max-dist	Maximum Distance
Sdfman-HR	High Definition
Sdfman-star	Maximum Distance + High Definition

When calculating the error of each algorithm, we omit the pixels whose ground truth depth exceeds the maximum distance threshold $max\_dist = 5$ m. Table 3 shows the accuracy of each algorithm. We use meter (m) rather than pixel disparity as the units for the error to give a more intuitive sense of the error magnitudes. With the initial input (Matlab SGM (Hirschmuller, 2005), Dispnet (Mayer et al., 2016)), DSF (Poggi and Mattoccia, 2016) reduces the mean absolute depth error from 0.40 m (Matlab SGM) and 0.24 m (Dispnet) to 0.18 m (DSF), which is larger than that of Sdfman-initial (0.09 m). Compared with Sdfman-initial (0.09 m), Sdfman-max-dist, Sdfman-HR and Sdfman-star are more accurate, which demonstrates that each of the proposed strategies contributes to improving the fusion accuracy. Algorithm Sdfman-star performs best with the mean absolute depth error (0.03 m) and achieves this at 34.21 frames per second. In the following experiments, we omit the two internal algorithms (Sdfman-max-dist and Sdfman-HR) because they are only used for the ablation study.

Table 3: Mean absolute depth error of the disparity fusion on Trimbot Wageningen SLAM Dataset (SD=Sdfman)

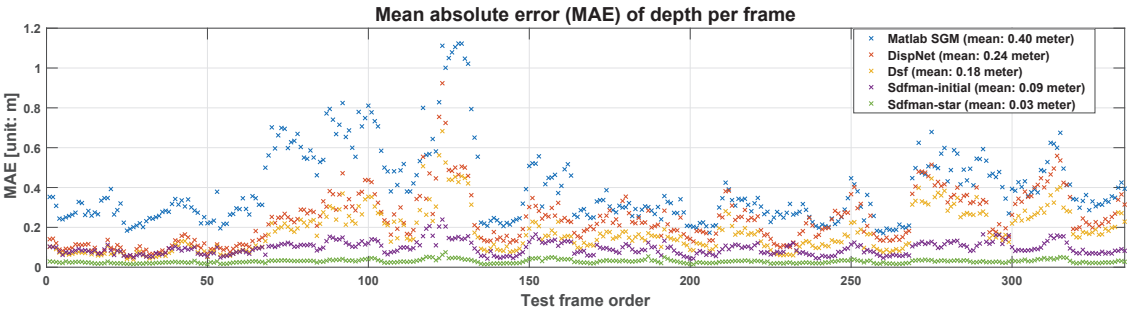
	Inputs		**Comparison**		**Ablation Study**		
	Matlab SGM	Dispnet	DSF	SD-initial	SD-max-dist	SD-HR	SD-star
Error	0.40 m	0.24 m	0.18 m	0.09 m	0.07 m	0.08 m	**0.03 m**

Figure 7 (a) compares the mean absolute error of each frame's depth map in the test dataset from all the algorithms. The accuracy of Sdfman-star is better than the other algorithms at all

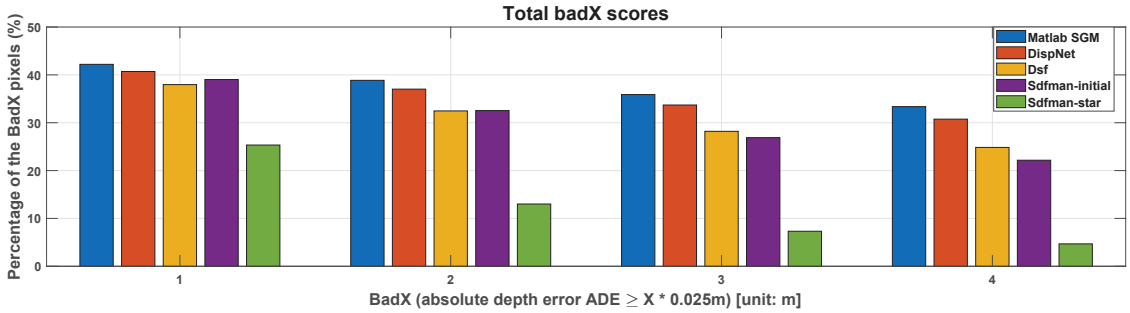
[12]Matlab Implementation URL:`https://www.mathworks.com/help/vision/ref/disparitysgm.html`

[13]The authors of Dispnet (Mayer et al., 2016) were our project partners and they trained Dispnet on the project dataset to get their best performance.

[14]As the focal length and baseline are fixed, depth estimation is inversely proportional to its corresponding disparity value - see Equation 2. When the depth is 5 meters, the corresponding disparity is about 3 pixels. Estimated depth values larger than 5 meters (i.e. disparity value smaller than 3 pixels) have a larger error compared with depths closer than 5 meters.



(a)

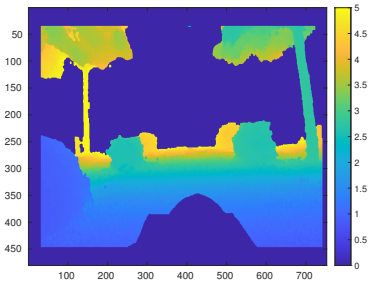


(b)

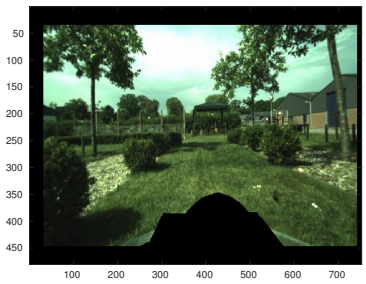
Figure 7: The mean absolute error (MAE) of the estimated depth map per frame and the total bad X scores

frames, which shows the robustness of Sdfman-star. Define parameter badX to be the percentage of pixels whose absolute depth errors in the depth map are bigger than X * 0.025 m (X is a positive number). Figure 7 (b) shows the percentage of badX pixels for different badX thresholds (values of X). Compared with the other algorithms, Sdfman-star has fewer pixels whose absolute depth error is bigger than 0.025 m, 0.05 m, 0.075 m and 0.1 m respectively. More than 95% of the pixels (bad4) from Sdfman-star have an absolute depth error less than 0.1 m.

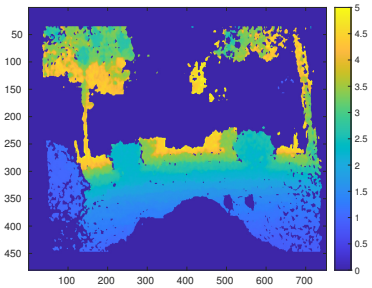
Figure 8 shows one qualitative result from one image sensor (Cam 0). Compared with the other algorithms, Sdfman-star is more accurate globally and also preserves small details more vividly (e.g. object edges, the trees' trunks).



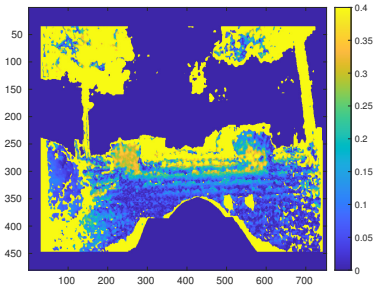
(a) Ground truth



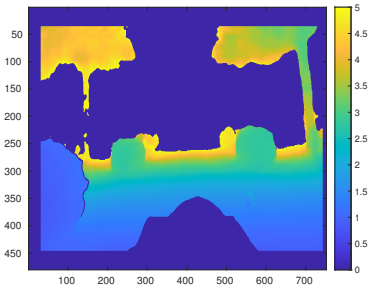
(b) RGB image



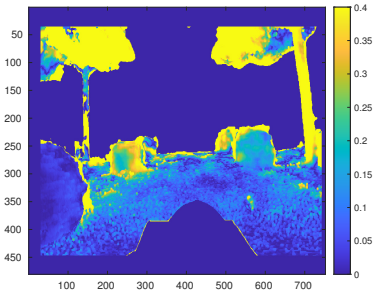
(c) Matlab SGM



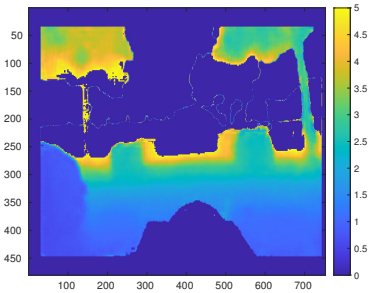
(d) SGM error



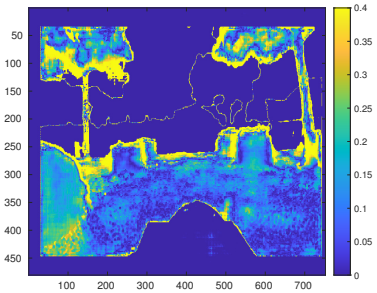
(e) Dispnet



(f) Dispnet error

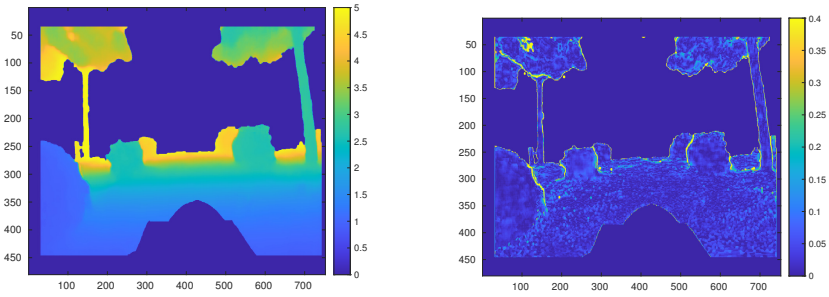


(g) Sdfman-initial



(h) Sdfman-initial error

Figure 8: *See Continuation*



(i) Sdfman-star (j) Sdfman-star error

Figure 8: One qualitative result for disparity fusion. The lighter pixels in (**d**,**f**,**h**,**j**) represent bigger depth error.

In the post-processing step, N_p is set as 20 and *radius* is set as 0.05 m. N_n is set as 20 and *dist_ratio* is set as 1.5. Figure 9 shows one example of the point clouds from the depth maps after outlier removal. Compared with the ground truth, the remote objects (e.g. trunk) in the point clouds from Sdfman-star are noisy, which can be expected.

4.3 Pose Fusion Module

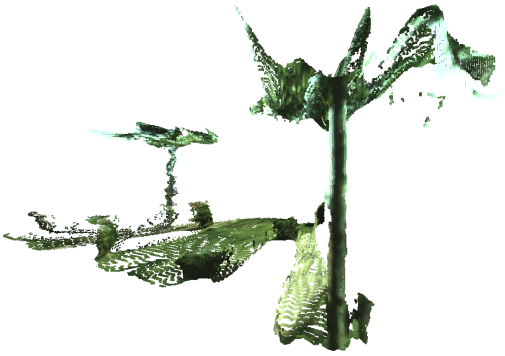
Section 4.3.1 presents results from an ablation study to show that the strategies proposed in Section 3.2 are effective. Section 4.3.2 compares Orbslam3 (Campos et al., 2021) and Open3D (Zhou et al., 2018) with the proposed pose fusion method.

Two methods are used to evaluate the 6D pose estimate accuracy. The first one uses the 6D pose vector $[tx, ty, tz, r, p, y]$ ([translation on X axis, translation on Y axis, translation on Z axis, roll, pitch, yaw]). The unit for tx, ty, tz is meters and the unit for r, p, y is degrees. The absolute difference between the ground truth and the estimated 6D vector is a measure of the 6D pose's **accuracy on each axis**. The second method uses the rotation matrix and translation vector. The **overall accuracy** of the 6D poses is computed using Equation 7 and Equation 8 (Huynh, 2009)

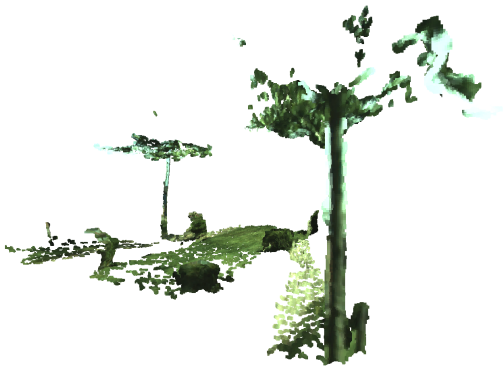
$$E_R = ||\boldsymbol{I} - \boldsymbol{R_{gt}}\boldsymbol{R_{est}^{-1}}||_F \tag{7}$$

$$E_t = ||\boldsymbol{t_{gt}} - \boldsymbol{t_{est}}||_F \tag{8}$$

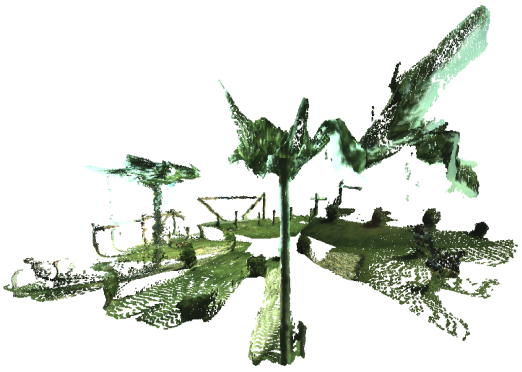
where $||\bullet||_F$ is the Frobenius norm. $\mathbf{R_{gt}}, \mathbf{t_{gt}}$ are the ground truth and $\mathbf{R_{est}}, \mathbf{t_{est}}$ are the estimated values, respectively. Equation 7 does not have a physical unit although smaller is better and is a measure of better point cloud overlap. Equation 8 is the distance between the two coordinate systems' origins (the ground truth and estimated coordinate system) and its unit is meter. Both ways of the above methods evaluate the 6D pose's accuracy, although their error values and their error estimation methods are different.



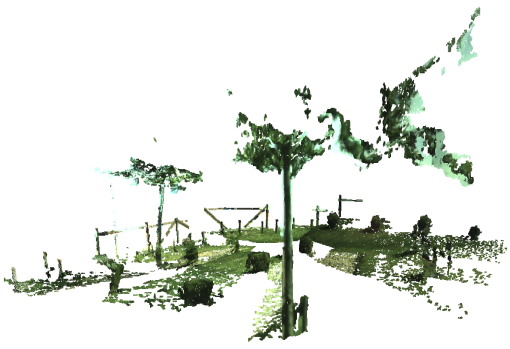
(k) Single-view point cloud from Sdfman-star



(l) Single-view point cloud from ground truth



(m) Full-view point cloud from Sdfman-star



(n) Full-view point cloud from ground truth

Figure 9: A single-view (72°) and full-view (360°) point cloud from Sdfman-star and ground truth

4.3.1 Ablation Study

Table 4: Model definition.

Model Name	**Strategy**
Ours	the proposed method in Section 3.2
Ours-global	disable the refined pose graph in Section 3.2.2
Ours-prune	disable rule 1: "Prune"
Ours-update	disable rule 2: "Update"
Ours-gtdepth	replace the depth from Sdfman-star with the depth from GT depth
Ours-single-stereo	input the stereo images from the front stereo camera only

We define six models to compare each proposed strategy's effectiveness. Table 4 lists the defined model names and the strategies. The model "Ours" is the proposed method in Section 3.2 with the point cloud input from Sdfman-star and is the baseline model, from which the other models are derived by changing only one strategy or factor. Model "Ours-global" disables the second-stage pose graph - the refined pose graph in Section 3.2.2. Model "Ours-prune" disables rule 1 - "Prune" and thus will not prune any edge, no matter what the edge's reliability is. Model "Ours-update" disables rule 2 - "Update" and thus will not update the transformation matrix of each edge. Model "Ours-gtdepth" replaces the input point clouds from Sdfman-star with the point clouds from the ground truth depth. Model "Ours-single-stereo" only inputs the stereo images from the front stereo camera (which consists of image sensors Cam0 and Cam1) rather than the panoramic stereo images from the ring of synchronized stereo cameras.

For all models, the overlapping rate threshold $OL_{min} = 0.33$ and $OL_{max} = 0.35$. The 6D pose vector $\vec{v}_{th}$ is set to [0.4 m, 0.4 m, 0.4 m, 15°, 15°, 15°] ([translation on X axis, translation on Y axis, translation on Z axis, roll, pitch, yaw]). Figure 10 shows the 2D trajectories from all the models when looking downward from above at the whole garden.

The trajectory of Ours-gtdepth is closest to the ground truth. The trajectories of Ours and Ours-prune are similar and rank 2^{nd} together. The remaining models perform worse. In particular note that Ours-single-stereo did not work correctlyin the latter part of the global pose trajectory, with completely wrong pose estimates. All the models except Ours-single-stereo have similar performance on the rotation factor, but perform on the translation factor variously. Table 5 shows the overall performance of each model by using the metrics in Equation 7 and Equation 8.

Table 5: Ablation study for pose fusion

Metric	Ours	Ours-global	Ours-prune	Ours-update	Ours-gtdepth	Ours-single-stereo
$\overline{E_R}$	0.11	0.08	0.12	0.08	0.08	0.98
σ_{E_R}	0.07	0.04	0.07	0.04	0.05	0.99
$\overline{E_t}$ (m)	0.33	0.48	0.34	0.52	0.27	2.47
σ_{E_t} (m)	0.18	0.32	0.20	0.29	0.16	1.99
$Time$ (s)	233.24	210.86	233.14	233.15	230.41	179.28

The running time of all the models are close (about 230 s) except Ours-global (210.86 s) and Ours-single-stereo (179.28 s). From the quantitative aspect, it is obvious that Ours-single-stereo fails compared with the other models. Thus, using the panoramic stereo images from the ring of synchronized stereo vision cameras in the proposed framework is vital to overcome the challenges of the fast or large transformations between adjacent frames when a real robot navigates in a real outdoor environment. The reason is that the 360° field of view makes the overlap between

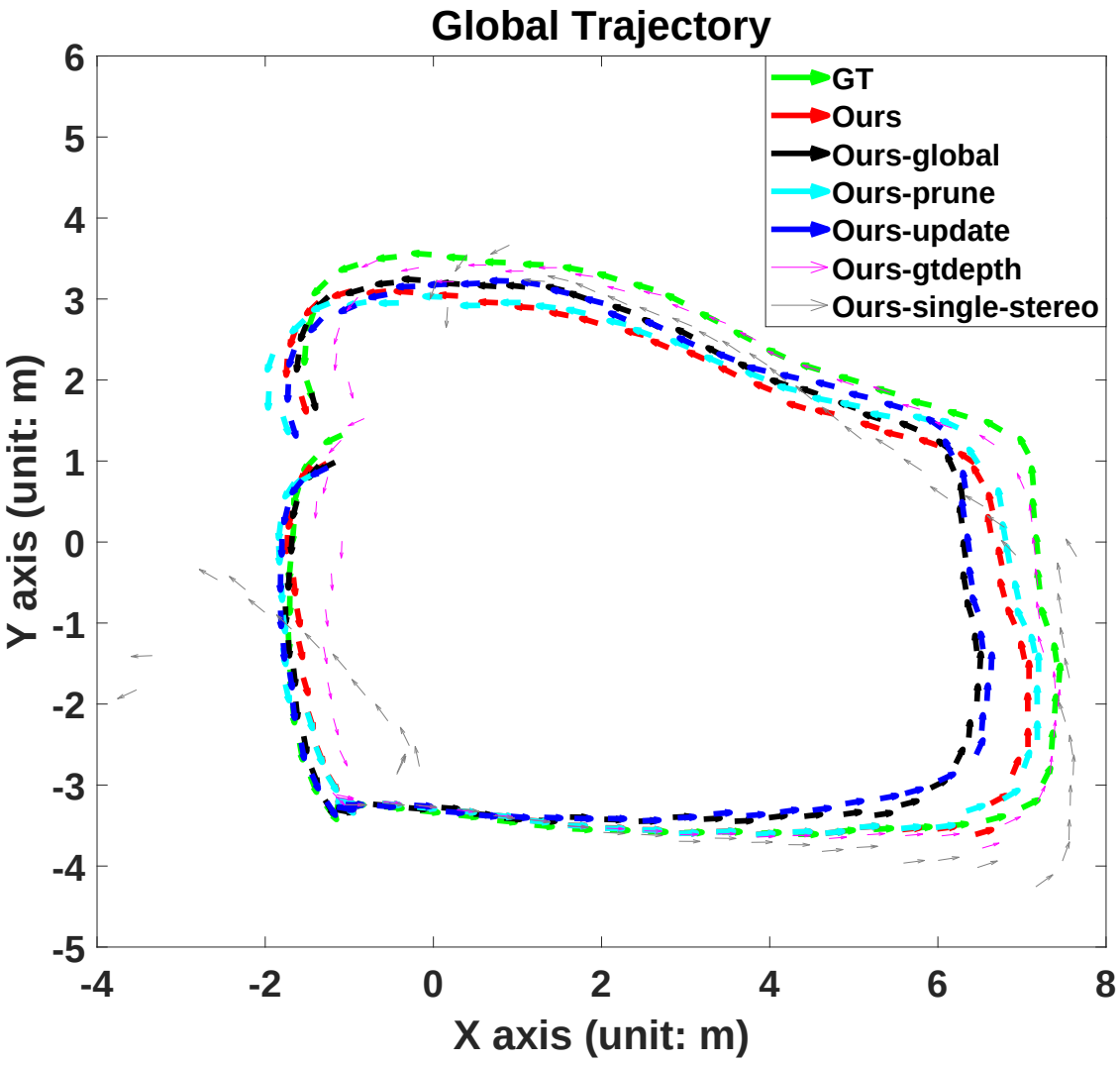


Figure 10: The estimated trajectory using each model

successive views high, which ensures the success of the global point cloud matching in the first stage of the pose fusion - global coarse pose graph optimization - to avoid the possibility of the whole 3D reconstruction framework collapsing. This strongly supports our major contribution (3) because we are the first to combine the two-stage full-view-to-single-view global-coarse-to-local-fine pose graph optimization with a ring of synchronized stereo vision cameras simultaneously to handle the robot's fast movement in the real world. The performance the other models (except Ours-single-stereo) on the rotation factor are similar (0.08 - 0.12) but the performance on the translation factor fluctuates (0.27 m - 0.52 m). The translation accuracy of the model "Ours-global" and "Ours-update" is much lower than that of the model "Ours", which demonstrates the two-stage from-coarse-to-fine pose graph optimization and rule 2 - "Update" are effective. The accuracy of the model "Ours-prune" is slightly worse than that of the model "Ours" on both rotation and translation factors, which shows that rule 1 - "Prune" is effective. Rule 1 "Prune" and rule 2 "Update" improve the performance because they make the transformation of the edge set $\tilde{E}$ more accurate and reliable which improves the constraint encoded in the loss function (see Equation 6). A more accurate constraint leads to to a more accurate pose estimate. The accuracy of model "Ours-gtdepth" is better than that of the model "Ours", which demonstrates that better recovery of the input point clouds leads to better pose fusion accuracy. Thus, one topic for future work is to continue improving the accuracy of the input point clouds.

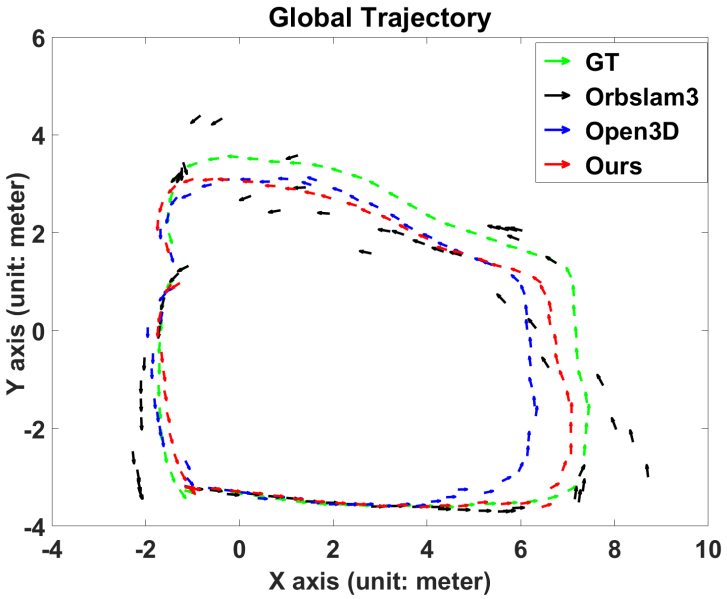
4.3.2 Comparison with Existing Methods

We compare the model "Ours" with existing state-of-the-art algorithms, represented by the RGBD SLAM algorithm in Orbslam3 (Campos et al., 2021) and the reconstruction system in Open3d (Zhou et al., 2018) as available online[15]. The depth maps that all the algorithms receive as input are the output from the Sdfman-star fusion algorithm. The parameter setting in model "Ours" is the same as that in Section 4.3.1. For Orbslam3, we set the number of features per image "ORBextractor.nFeatures" as 10000. The number of levels in the scale pyramid "ORBextractor.nLevels" is 15. The fast threshold "ORBextractor.iniThFAST" is 5 and "ORBextractor.minThFAST" is 3. The number of camera frames per second is 1. The rest of the parameters are the same as those in their released code. As the Orbslam3 framework does not support panoramic data, we input the RGB images and the corresponding depth maps from the image sensor 'Cam0' into the RGBD SLAM algorithm in the Orbslam3 framework. To make Orbslam3 work better on the difficult dataset "Trimbot Wageningen SLAM Dataset", we additionally provide the ground truth pose to Orbslam3 when the adjacent frames have a large rotation and Orbslam3 lost tracking (at all the corners of the trajectory). More specifically, at Frames 20, 31, 50, 54, 56, we provide the corresponding ground truth pose to Orbslam3. See Figure 11 (b) and Figure 11 (c) where both the rotation and translation error of Orbslam3 are equal to 0. Open3D failed to work if we only input the single-view point clouds. To make Open3D perform better, we modified its initial code to make it use our full-view and single-view point clouds. We also provide the comparison results under the same conditions in Appendix E.2 Fair Comparison With More Open-source Frameworks. Readers can test their own code on the "Trimbot Wageningen SLAM dataset".

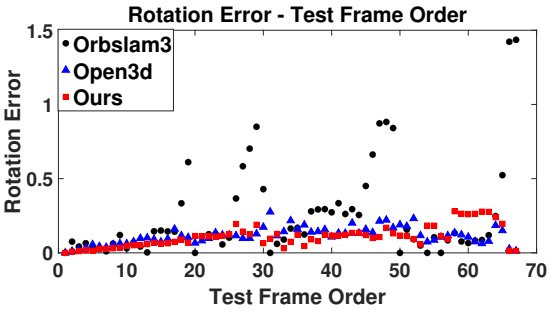
Figure 11 (a) shows the estimated global trajectory from GT, Orbslam3, Open3D, and Ours. Figure 11 (b) and (c) show the rotation and translation error at each frame time in the test dataset using Equation 7 and Equation 8. From Figure 11 we could see Open3D and Ours perform much more accurately and robustly than Orbslam3. Open3D and Ours have similar performance on the rotation and Ours performs more accurately than Open3D on the translation.

If we use the absolute difference between the ground truth and the estimated 6D vector to describe the 6D pose accuracy, Table 6 compares the performance of the algorithms on each axis. Our approach's mean bias and the related standard deviation of the translation on the x, y, and z axis are generally smaller than those of Orbslam3 and Open3D. The rotation performance of Ours and Open3D on x, y, and z axis is more accurate and robust than that of Orbslam3. Our

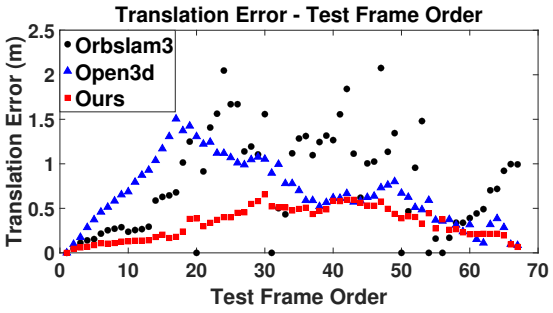
[15] Orbslam3: `https://github.com/UZ-SLAMLab/ORB_SLAM3` and Open3D: `https://github.com/isl-org/Open3D`



(a) Global Pose Trajectories from Different Algorithms



(b) Overall Rotation error



(c) Overall Translation error

Figure 11: Figure (a) shows the global pose trajectory of the sensor in the real garden; Figure (b), (c) show the overall rotation and translation error at each frame in the test dataset.

rotation performance on x, y, and z axis is similar to that of Open3D.

Table 6: Performance comparison of the algorithms on each axis

Metric	*Orbslam3*	*Open3D*	*Ours*
$\overline{tx}$ (m)	0.42	0.55	**0.18**
σ_{tx} (m)	0.36	0.44	**0.14**
$\overline{ty}$ (m)	0.44	0.21	**0.20**
σ_{ty} (m)	0.50	**0.15**	0.18
$\overline{tz}$ (m)	0.26	0.18	**0.11**
σ_{tz} (m)	0.35	0.14	**0.08**
$\overline{r}$ (deg)	8.40	**2.77**	3.00
σ_r (deg)	12.28	**1.95**	3.34
$\overline{p}$ (deg)	1.96	2.40	**1.03**
σ_p (deg)	3.14	1.82	**1.03**
$\overline{y}$ (deg)	3.50	2.12	**1.91**
σ_y (deg)	4.98	1.58	**1.52**

Table 7 shows the overall performance of each algorithm using Equation 7 and Equation 8. Ours performs best, although it increases the running time slightly. Given the bad performance of Orbslam3 on the real outdoor garden dataset, we will omit Orbslam3 in the following text and compare Ours with Open3D in Section 4.4 "Volumetric Fusion Module" only.

Table 7: Overall performance comparison with external algorithms

Metric	Orbslam3	Open3D	Ours
$\overline{E_R}$	0.25	0.12	**0.11**
σ_{E_R}	0.31	**0.06**	0.07
$\overline{E_t}$ (m)	0.78	0.68	**0.33**
σ_{E_t} (m)	0.57	0.37	**0.18**
$Time$ (s)	**67.82**	229.76	233.24

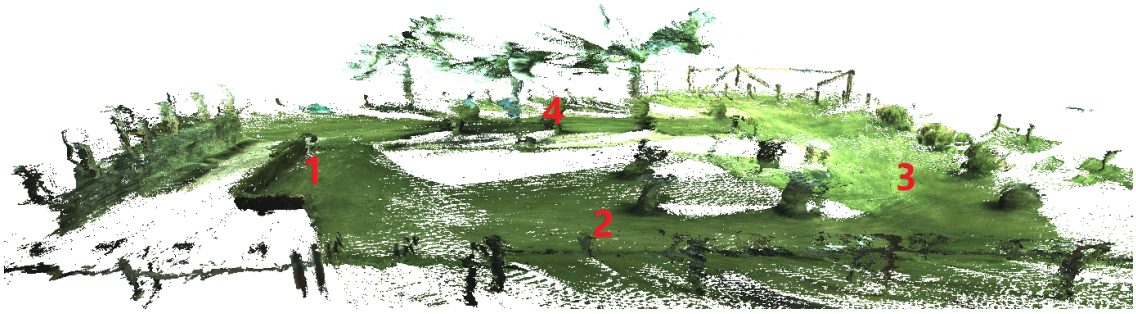
4.4 Volumetric Fusion Module

In this part, we set the maximum depth for integrating as 5 meters. The size of TSDF (Truncated Signed Distance Field) cube is 10 meters. The length of each voxel is 0.01 m (1 cm). The truncation value for the signed distance function (SDF) is set to 0.06.

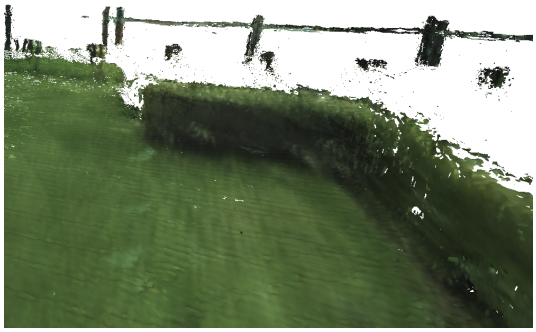
Figure 12 shows the mesh of the reconstructed garden and its details at different sites. Figure 12 (a) shows the overview of the reconstructed whole garden. Figure 5 (a) shows the ground truth. Figure 12 (b) (c) (d) (e) show close-up views at sites 1, 2, 3, 4 in Figure 12 (a). The white blank areas in all the figures are regions that have not been scanned during driving. These regions did not have target plants for the trimming robot and thus were not scanned. From the details, the reconstructed scene is good enough for the remote visualization and coarse robot task planning. A video that shows the reconstructed garden is at: `https://youtu.be/zGxcj0_NXCA`.

In the following, the reconstructed gardens from all the algorithms are compared to the ground truth 3D model of the whole garden in the same world coordinate system. The evaluation method consists of estimating the mean and standard deviation of the minimum distance between each point of the reconstructed garden and its closest point in the ground truth garden model. Thus, this metric measures how close the reconstructed garden is, on average, with respect to the ground truth garden model.

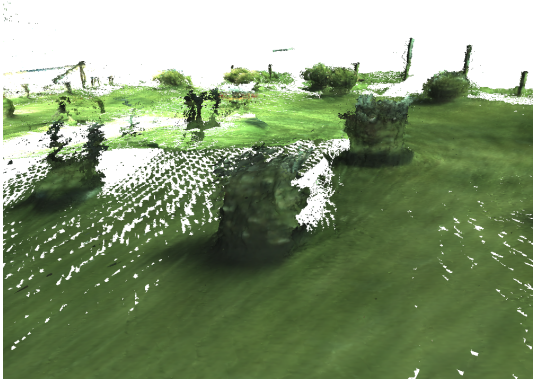
Table 8 shows the mean $\overline{dist}$ and standard deviation σ_{dist} of the minimum distance between the corresponding closest points. The mean and standard deviation of the minimum distance's absolute bias on x, y, and z axis are ($\overline{dx}$, σ_{dx}), ($\overline{dy}$, σ_{dy}) and ($\overline{dz}$, σ_{dz}) respectively. The maximum



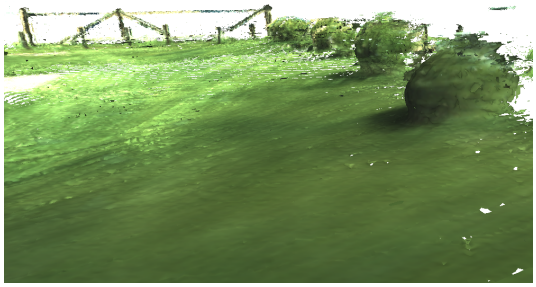
(a) Reconstructed 3D Garden



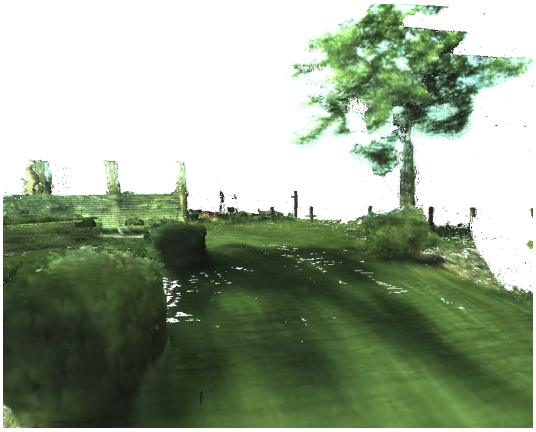
(b) Close-up shot 1



(c) Close-up shot 2



(d) Close-up shot 3



(e) Close-up shot 4

Figure 12: Figure (a) shows the reconstructed 3D model of the real garden; Figure (b) - (e) shows close-up shots for sites 1 - 4.

of the minimum distance's absolute mean bias on all the three axes is $\overline{dz}$ (0.15 m) and the mean of the minimum distance $\overline{dist}$ is 0.18 m, which are good enough for the user's remote visualization and robot global task planning[16] on the reconstructed global model.

Table 8: Reconstruction Accuracy

Metric	Open3D	Ours
$\overline{dx}$ (m)	0.06	**0.05**
σ_{dx} (m)	**0.09**	**0.09**
$\overline{dy}$ (m)	**0.05**	**0.05**
σ_{dy} (m)	0.09	**0.07**
$\overline{dz}$ (m)	0.20	**0.15**
σ_{dz} (m)	0.18	**0.12**
$\overline{dist}$ (m)	0.24	**0.18**
σ_{dist} (m)	0.19	**0.14**
Time /s	8.48	**8.23**

Figure 13 (a) (c) (e) show the reconstructed gardens from Open3D, Ours, and the ground truth garden model. Figure 13 (b) (d) (f) show the details on the same site in the real garden. Compared with ours, we could see Open3D fails to align the point clouds of the same tree, and makes it seem that there were two trees on that site.

In addition to the experiments above, there are two other experiments. In the first experiment, the proposed framework is successfully tested with scene appearance and sunlight change. More details can be found in Appendix E.1. The second experiment compares our proposal with a popular commercial software application 'ContextCapture' on the Trimbot Wageningen SLAM Dataset. The proposed approach again has better performance. For more details, see Appendix E.3.

5 Conclusion and Discussion

This paper presented an improved approach for recovering accurate outdoor 3D scene reconstruction, based on disparity fusion, pose fusion and volumetric fusion, and demonstrated its performance by reconstructing a real outdoor garden containing a variety of different natural and man-made structures. Avoiding the need for expensive and sparse Lidar scans, the proposed approach inputs the disparity maps from two different stereo vision algorithms into a disparity fusion network to produce accurate disparity maps, which is a cheap, accurate and robust solution to get higher quality depth data. The depth data is converted into point clouds, whose outliers are removed, and then input into the pose fusion module. The pose fusion module uses a two-stage from-global-coarse-to-local-fine pose graph optimization to estimate a more accurate global pose trajectory. More specifically, in the first stage, we use fast global point cloud registration (Zhou et al., 2016) and full-view (360°) point clouds to build a coarse global pose graph, which is robust to fast motion and big transformations between two consecutive frames. In the second stage, a local point cloud registration algorithm GICP (Segal et al., 2009) extended with three domain rules optimizes the refined pose graph, which produces a more accurate global pose trajectory. With the accurate global pose trajectory and the fused depth maps, the mesh of the whole garden can be reconstructed by volumetric fusion, as demonstrated on a real outdoor dataset.

The key to a good 3D reconstruction of the real garden is the accurate depth map and global pose trajectory. In future work, more advanced disparity fusion networks will be explored to continue to improve the disparity accuracy. The accuracy of the 6D pose that registers the point clouds affects the accuracy of the edges in the pose graph, which in turn influence the optimized global pose trajectory. More advanced and faster global and local point cloud registration algorithms

[16]Our trimming robot did coarse global task planning on the reconstructed global model first. When the robot arrives at the specific location for trimming, the robot arm will move the depth cameras on the robot arm to scan the target locally and build the accurate local 3D model with the precise pose update from the robot arm's joints.

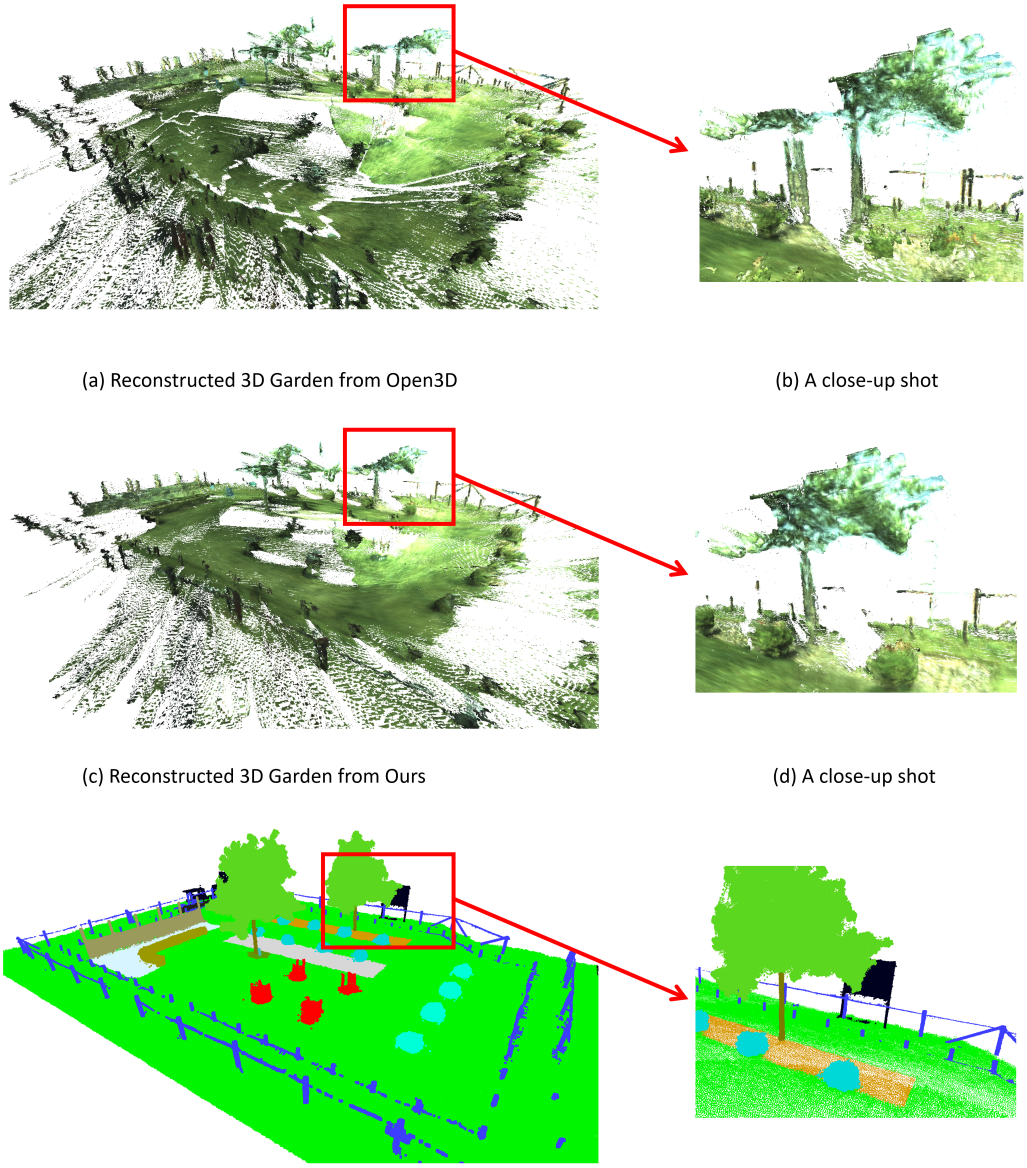


Figure 13: Comparison of the reconstructed garden from Open3d, Ours and GT.

that are robust against strong noise and large occlusions will be explored to get more accurate initial 6D pose estimation. The research proposed in this paper is adapted for use by a robot in a garden environment, but it can be generalized to different outdoor application scenarios. However, this requires the related ground truth for the network training and performance evaluation. Expanding into related domains of robot applications is our priority for the near future.

Acknowledgements

Before 2020, the research was funded by the TrimBot2020 project (Grant Agreement No. 688007, URL: `http://trimbot2020.webhosting.rug.nl/`) from the European Union Horizon 2020 programme. After 2020, the research funding is from Shenzhen Amigaga Technology Co. Ltd. by the Gagabot2022 project (Grant Agreement No. P987001), from the Human Resources and Social Security Administration of Shenzhen Municipality by Overseas High-Caliber Personnel project (Grant NO. 202102222X, Grant NO. 202107124X) and from Human Resources Bureau of Shenzhen Baoan District by High-Level Talents in Shenzhen Baoan project (Grant No. 20210400X, Grant No. 20210402X). We thank Gabriel Moreira from Moreira et al. (2021a,b) for the help with pose graph optimization. We thank all the partners from TrimBot2020 consortium for their help when we did this work, and for their contributions to test garden design, robot and sensor design and construction, data collection, and ground truthing, as well as many other contributions to the TrimBot2020 project.

Appendix A. List of Abbreviations

FOV	Field Of View
GAN	Generative Adversarial Network
HD	High Definition
ICP	Iterative Closest Point
LC	Loop Closure
MPTEJI	Multi-stage Pose Trajectory Estimation with Joint Information
ORB	Oriented FAST and Rotated BRIEF
PS-SLAM	Panoramic Stereo SLAM
SDF	Signed Distance Function
SFM	Structure From Motion
SIFT	Scale Invariant Feature Transform
SLAM	Simultaneous Localization And Mapping
TOF	Time of Flight
TSDF	Truncated Signed Distance Field

Appendix B. List of Symbols

max_dist	The maximum distance threshold
ϕ_w	The width resolution ratio between the HD and the initial image
ϕ_h	The height resolution ratio between the HD and the initial image
N_p	The threshold number of the neighboring points in a sphere
$radius$	The radius of the sphere
N_n	The number of the points in the neighborhood
$dist\_ratio$	The distance ratio to remove the points
OL_{min}	The minimum overlapping rate threshold
OL_{max}	The maximum overlapping rate threshold
$\vec{v}_{th}$	6D pose threshold in vector format

Appendix C. Formula derivation

C.1 Loss Function

In this subsection, we will prove that Equation 4 in this paper is equal to equation 4 in the paper (Moreira et al., 2021a) under the assumption that the uncertainty of the rotation is the same as the translation's. Although 'Equation 1' in the paper (Moreira et al., 2021a) is similar to our Equation 4, the authors (Moreira et al., 2021a) did not prove that the Frobenius-norm-based transformation difference loss function (Equation 4) is a special case of the maximum likelihood loss function in the pose graph optimization, which is the motivation for this section.

In Equation 4, $G(V, E)$ is a connected pose graph with $|V| = n$ poses (or vertices). The rigid transformation T_{ij} (here, computed using a point cloud registration algorithm) from the i^{th} pose (denoted by P_i) to the j^{th} pose (denoted by P_j) could be written as $\{\tilde{\mathbf{R}}_{ij}, \tilde{t}_{ij}\}$ for the edge $(i, j) \in E$. E is the edge set. $\tilde{\mathbf{R}}_{ij}$ and $\tilde{t}_{ij}$ are the corresponding relative rotation and translation estimates. The i^{th} pose P_i could be written as $\{\mathbf{R}_i, t_i\}_{i=1,\dots,n}$. In the following, we will use block-matrix notation to represent Equation 4.

As the transformation is rigid, thus:

$\tilde{\mathbf{R}}_{ij}\tilde{\mathbf{R}}_{ij}^T = \mathbf{I}$, $\mathbf{R}_i\mathbf{R}_i^T = \mathbf{I}$, $\mathbf{R}_j\mathbf{R}_j^T = \mathbf{I}$.
Let us write:
$T_{ij} = \begin{bmatrix} \tilde{\mathbf{R}}_{ij} & \tilde{t}_{ij} \\ \mathbf{0} & 1 \end{bmatrix}$, $P_i = \begin{bmatrix} \mathbf{R}_i & t_i \\ \mathbf{0} & 1 \end{bmatrix}$,
$P_j = \begin{bmatrix} \mathbf{R}_j & t_j \\ \mathbf{0} & 1 \end{bmatrix}$, $P_j^{-1} = \begin{bmatrix} \mathbf{R}_j^T & -\mathbf{R}_j^T t_j \\ \mathbf{0} & 1 \end{bmatrix}$;
Thus, Equation 4 can be transformed into:

$$\begin{aligned} &\arg\min \sum_{(i,j)\in E} \left\| T_{ij} - P_i P_j^{-1} \right\|_F^2 \\ =&\arg\min \sum_{(i,j)\in E} \left\| \begin{bmatrix} \tilde{\mathbf{R}}_{ij} - \mathbf{R}_i\mathbf{R}_j^T & \tilde{t}_{ij} - t_i + \mathbf{R}_i\mathbf{R}_j^T t_j \\ \mathbf{0} & 0 \end{bmatrix} \right\|_F^2 \\ =&\arg\min \sum_{(i,j)\in E} \left(\left\| \tilde{\mathbf{R}}_{ij} - \mathbf{R}_i\mathbf{R}_j^T \right\|_F^2 + \left\| \tilde{t}_{ij} - t_i + \mathbf{R}_i\mathbf{R}_j^T t_j \right\|_F^2 \right) \end{aligned} \tag{C.1}$$

According to the definition, Frobenius norm of a matrix $\mathbf{A}$ is defined as the square root of the sum of the absolute squares of its elements in the matrix, which is equal to the square root of the matrix trace of $\mathbf{A}\mathbf{A}^T$. Additionally, $tr(A) = tr(A^T)$.
Expanding the term $\left\| \tilde{\mathbf{R}}_{ij} - \mathbf{R}_i\mathbf{R}_j^T \right\|_F^2$ in Equation C.1:

$$\begin{aligned} &\left\| \tilde{\mathbf{R}}_{ij} - \mathbf{R}_i\mathbf{R}_j^T \right\|_F^2 \\ =&tr\{(\tilde{\mathbf{R}}_{ij} - \mathbf{R}_i\mathbf{R}_j^T)(\tilde{\mathbf{R}}_{ij} - \mathbf{R}_i\mathbf{R}_j^T)^T\} \\ =&tr(\tilde{\mathbf{R}}_{ij}\tilde{\mathbf{R}}_{ij}^T + \mathbf{R}_i\mathbf{R}_j^T\mathbf{R}_j\mathbf{R}_i^T - \tilde{\mathbf{R}}_{ij}\mathbf{R}_j\mathbf{R}_i^T - \mathbf{R}_i\mathbf{R}_j^T\tilde{\mathbf{R}}_{ij}^T) \\ =&tr(\mathbf{I} + \mathbf{I}) - tr(\tilde{\mathbf{R}}_{ij}\mathbf{R}_j\mathbf{R}_i^T) - tr\{(\tilde{\mathbf{R}}_{ij}\mathbf{R}_j\mathbf{R}_i^T)^T\} \\ =&6 - 2tr(\tilde{\mathbf{R}}_{ij}\mathbf{R}_j\mathbf{R}_i^T) \end{aligned} \tag{C.2}$$

Substitute the term $\left\| \tilde{\mathbf{R}}_{ij} - \mathbf{R}_i\mathbf{R}_j^T \right\|_F^2$ in Equation C.1 with Equation C.2 and neglect the constant term:

$$\begin{aligned} &\arg\min \sum_{(i,j)\in E} \left\| T_{ij} - P_i P_j^{-1} \right\|_F^2 \\ =&\arg\min \sum_{(i,j)\in E} \{ \left\| \tilde{t}_{ij} - t_i + \mathbf{R}_i\mathbf{R}_j^T t_j \right\|_F^2 - 2tr(\tilde{\mathbf{R}}_{ij}\mathbf{R}_j\mathbf{R}_i^T) \} \end{aligned} \tag{C.3}$$

In Equation C.3 we minimize the loss function to get the estimated rotation and translation by maximizing its negative. Thus, divide the right part of the equal sign by the negative constant $-2\sigma_R^2$, we get:

$$\begin{aligned} &\arg\min \sum_{(i,j)\in E} \left\| T_{ij} - P_i P_j^{-1} \right\|_F^2 \\ =&\arg\max \{ -\frac{1}{2\sigma_R^2} \sum_{(i,j)\in E} \left\| \tilde{t}_{ij} - t_i + \mathbf{R}_i\mathbf{R}_j^T t_j \right\|_F^2 \\ &+ \frac{1}{\sigma_R^2} \sum_{(i,j)\in E} tr(\tilde{\mathbf{R}}_{ij}\mathbf{R}_j\mathbf{R}_i^T) \} \end{aligned} \tag{C.4}$$

Compare our term Equation C.4 with the log-likelihood term equation 4 in the paper (Moreira

et al., 2021a), which is shown in the following Equation C.5:

$$\begin{aligned} & logL(\theta|y) \\ = & -\frac{1}{2\sigma_t^2}\sum_{(i,j)\in E}\left\|\tilde{t}_{ij}-t_i+\mathbf{R}_i\mathbf{R}_j^T t_j\right\|_F^2 \\ & +\frac{1}{\sigma_R^2}\sum_{(i,j)\in E} tr(\tilde{\mathbf{R}}_{ij}\mathbf{R}_j\mathbf{R}_i^T) \end{aligned} \tag{C.5}$$

When the noise level for the rotation and translation in the paper (Moreira et al., 2021a) is assumed to be equal (i.e. $\sigma_t = \sigma_R$), Equation C.4 (in our paper) and Equation C.5 (which is same with equation 4 in Moreira et al. (2021a)) are completely the same. Thus, we could use the optimization method[17] in Moreira et al. (2021a) to optimize our error function. Finding the optimum rotation parameters first and then solving for the translation parameters turns it into a least-squares problem. To deduce the Equation C.5, refer to Page 9 - 10 in (URL: `https://drive.google.com/file/d/1ML7mkLSIALm3x5DtID7ozHD3YL7S1iNC/view?usp=sharing`) or the most related papers (Carlone et al., 2015a,b; Moreira et al., 2021a,b).
To conclude, Equation 4 can be turned into a maximum likelihood estimation problem under the assumption of the proper noise level for rotation and translation. From another aspect, there is a more intuitive way to express the physical meaning of Equation 4. That is: estimate the pose of each node accurately, which in turn makes the existing relative pose measurements between different nodes closer to the post-calculated relative pose between different nodes based on their estimated global pose.

C.2 Maximum Distance

In the initial work Sdf-man (Pu et al., 2019), at the end of the refiner network (see Figure 2 on page 7 in Pu et al. (2019)) the method uses the function 'tanh' to output an intermediate map w and each value in w is in $(-1, 1)$.

$$initial\_disp = \frac{max\_disp \cdot (w+1)}{2} \tag{C.6}$$

Then it uses Equation C.6 to convert the intermediate map w to the disparity map $initial\_disp$. max_disp is the maximum disparity threshold. Converting the disparity map $initial\_disp$ into a depth map using Equation 2 gives Equation C.7.

$$initial\_depth = \frac{2fb}{max\_disp \cdot (w+1)} \tag{C.7}$$

f and b are the focal length value and baseline value of the stereo vision camera. As w ranges from -1 to 1, the $initial\_depth$ values will range from $\frac{fb}{max\_disp}$ to $+\infty$.

In this paper, the difference is that we use a new Equation C.8 to map the intermediate map w to the new disparity map new_disp rather than Equation C.6.

$$new\_disp = \frac{2fb}{max\_dist \cdot (w+1)} \tag{C.8}$$

max_dist is set as the maximum distance threshold of interest. Converting the new disparity map new_disp into the depth map format using Equation 2 gives Equation C.9, whose value domain is $(0, max\_dist)$.

$$new\_depth = \frac{max\_dist \cdot (w+1)}{2} \tag{C.9}$$

[17] The URL of the released code: `https://github.com/gabmoreira/maks`

Comparing the value domain of *new_depth* in Equation C.9 and *initial_depth* Equation C.7, the domain $(\frac{fb}{max\_disp}, +\infty)$ of *initial_depth* is larger than the domain $(0, max\_dist)$ of *new_depth* considerably, though their definition domains are the same. Thus, as for noise with the same granularity in the input, *new_depth* from the proposed strategy will output a more robust and accurate result. By setting the maximum distance threshold of interest and the new mapping function Equation C.8, we effectively narrow the value domain of the depth output to increase depth accuracy. This alternative approach has also been confirmed by the experiment results in Section 4.2.

Appendix D. More Details about Data Collection

As Figure 5 (b) shows, the trimming robot Trimbot[18] navigates around the outdoor garden[19] to collect the raw data. The top of the robot resembles a 'tower' (see Figure D.1), which consists of a prism retroreflector, a Velodyne VLP16, and the panoramic stereo camera (a ring of 5 stereo cameras).

The prism reflector on the robot is used to reflect the laser beam from Topcon PS Series Robotic Total Station (See Figure D.2), to estimate the robot's position in 3D space. According to the datasheet of the Topcon PS Series Robotic Total Station[20], the distance measurement accuracy with the prism could be down to $1.5mm + 2ppm$. To estimate the robot's orientation [roll, pitch, yaw], a STIM300 IMU sensor inside the trimming robot is used to record the acceleration and rotation rate measurements. According to the datasheet of STIM300 IMU sensor[21], the gyroscope input range is $\pm 400deg/sec$ and its angular random walk is $0.15deg/\sqrt{hr}$. The accelerometer range is $\pm 10g$ and its velocity random walk is $0.06m/s/\sqrt{hr}$. The in-run bias stability of the gyroscope and accelerometer is $0.5deg/hr$ and, $0.05mg$ respectively. Based on the precise measurements from the STIM300 IMU sensor, the orientation of the robot is estimated in an offline post-processing step by strap-down integration. The 3D position $[tx, ty, tz]$ from Topcon PS Series Robotic Total Station and the orientation $(roll, pitch, yaw)$ are appended to constitute the 6-DoF pose of the robot. Then, by calibration, the 6-DoF pose of the robot is transformed to get the pose of each rigidly placed image sensor in the panoramic stereo camera. Finally, structure-from-motion (Schonberger and Frahm, 2016) is used to refine each image sensor's pose to form the ground truth pose of each image sensor, particularly to fix poses where the line of sight between the Topcon and prism was interrupted by obstacles.

The Velodyne VLP16 lidar sensor is mounted on top of the panoramic stereo camera to record a reference point cloud from the Trimbot robot's perspective. The lidar sensor has a 360° horizontal field of view with an angle resolution spanning from 0.1° to 0.4°, which corresponds to the rotation rate from 5 Hz to 20 Hz. In the Trimbot Wageningen SLAM Dataset, the angular resolution is set to 0.2° and the rotation rate is set to $10Hz$. The Velodyne VLP16 lidar sensor has 16 horizontal rays, which are distributed within a vertical field of view of ±15°. According to its datasheet[22], its scanning range could be up to 100 m with an accuracy of $\pm 3cm$. Then the lidar point cloud is projected to the image planes of the 10 image sensors in the panoramic stereo camera to form sparse depth maps.

The panoramic stereo camera[23] is built with ten MT9V024 CMOS image sensors from ON-Semiconductors[24]. The housing of the panoramic stereo camera has a pentagon shape and is

[18] Trimbot's hardware was mainly developed by our Trimbot2020 consortium member Robert Bosch GmbH based on the Bosch Indigo lawn mower.

[19] The garden was constructed by our Trimbot2020 consortium member Wageningen Research in Netherlands.

[20] Topcon PS Series Robotic Total Station: `https://drive.google.com/file/d/1Z54jDM0fkqhNgYq1SBsZRFX-XG14alIC/view?usp=sharing`

[21] STIM300 IMU: `https://drive.google.com/file/d/1PhFtSSABCs0msnu2Gwza0EhAg4mpUZar/view?usp=sharing`

[22] Velodyne VLP16: `https://drive.google.com/file/d/1ZYrYHf7wqI5PjuuKpxO7SPzDfmb33b1Y/view?usp=sharing`

[23] The panoramic stereo camera hardware was developed by our Trimbot2020 consortium member ETH Zürich based on their previous work (Honegger et al., 2017).

[24] MT9V024 datasheet URL:`https://github.com/Canpu999/Trimbot-Wageningen-SLAM-Dataset/blob/main/`

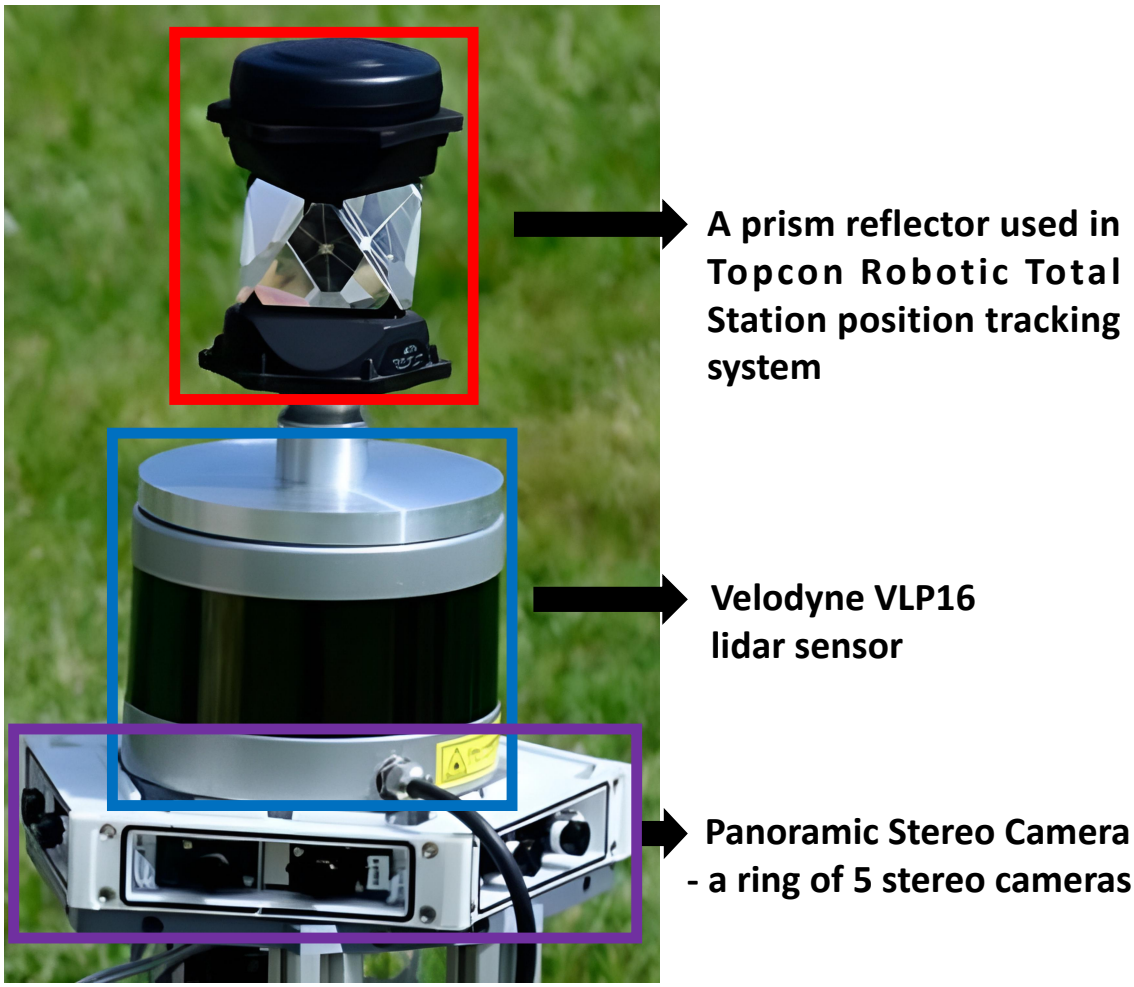


Figure D.1: Some sensors for collecting the data

Figure D.2: Topcon PS Series Robotic Total Station estimated the robot's 3D position by emitting laser beams and receiving the beams reflected by he prism reflector on the top of the trimming robot Trimbot.

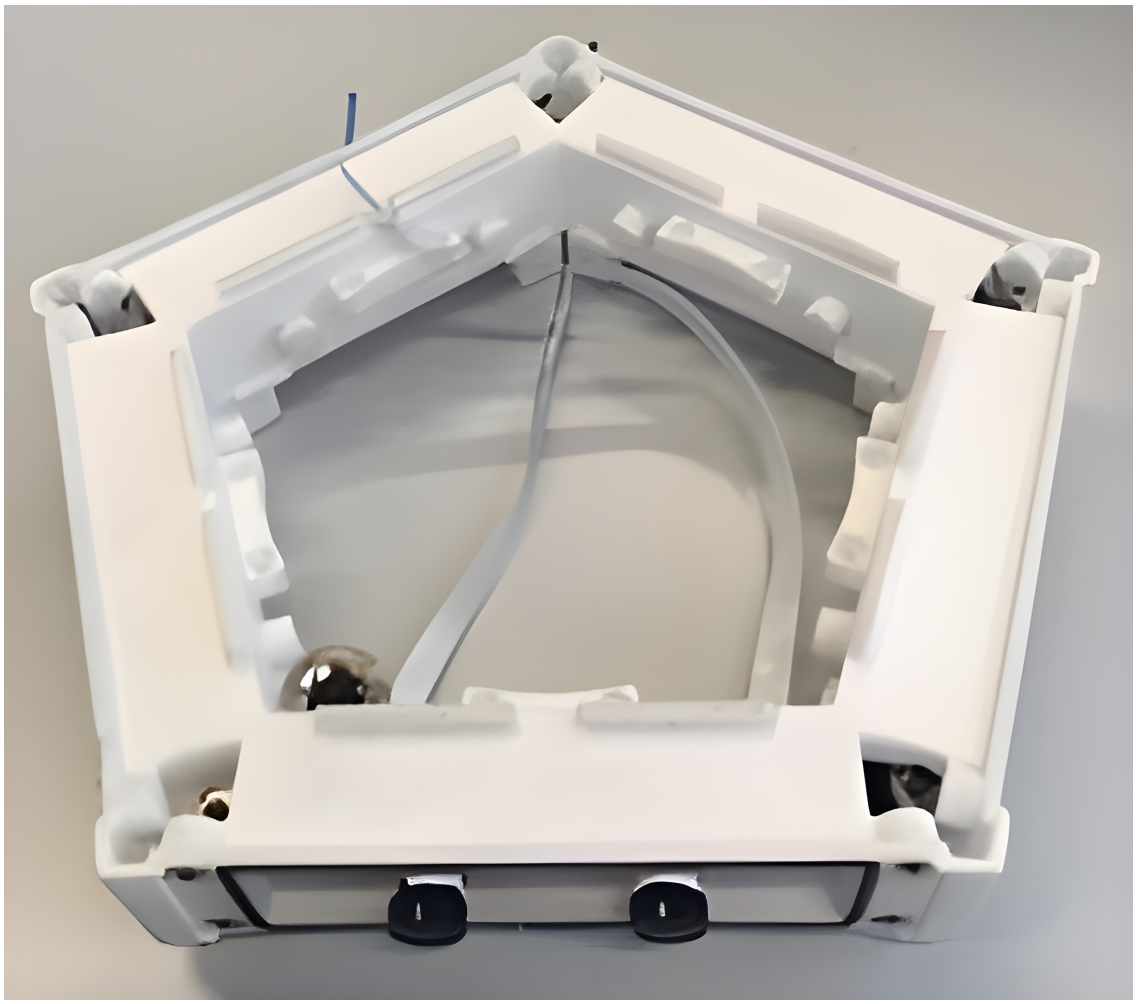
manufactured by 3D printing. Figure D.3 shows the panoramic stereo camera from the top and side views. The image sensors have 752×480 ($Horizontal \times Vertical$) pixels resolution with global shutter. The maximum frame rate of a single image sensor could be up to 60 FPS at full resolution. The five synchronized stereo vision cameras (10 image sensors) could only stream the synchronized panoramic stereo images (10 synchronized images) at 12 HZ because it is limited by the bandwidth of the panoramic stereo camera's data bus. The image sensor's operating temperature ranges from $-40°C$ to $+100°C$ ambient. The active imager size is $4.51mm(H) \times 2.88mm(V)$, whose diagonal size is 5.35 mm. The pixel size is 6.0 μm $\times$ 6.0 μm. We calibrated the image sensor using the *Kalibr* package[25] by setting the camera model as pinhole and the distortion model as radial-tangential. The calibration accuracy of image sensors in terms of a mean reprojection error was 0.00 pixels on the X and Y axis, with standard deviation ranging from 0.1 pixels to 0.2 pixels. Given that the image sensors' temperature and long operating time influences the robustness of the commercial hardware's imaging quality, which will possibly influence the calibration accuracy, more engineering robustness tests would be future work.

The 3D point cloud of the whole garden was collected by a Leica ScanStation P15[26]. The Leica ScanStation P15 measures the distance by a laser scanner and incorporates the corresponding RGB data from a color camera into the laser scanner's coordinate system. The laser scanner's working distance range varies from $[0.4m, 40m]$. The 3D position accuracy could be low to 3 mm at 40 meters and the linearity error is smaller than 1 mm. The angular accuracy is 8" in the horizontal direction and 8" in the vertical direction. Figure D.4 shows the garden's point clouds from different views with RGB data and colored height (different colors represent different heights). As the distance measurement and the RGB measurement are conducted sequentially,

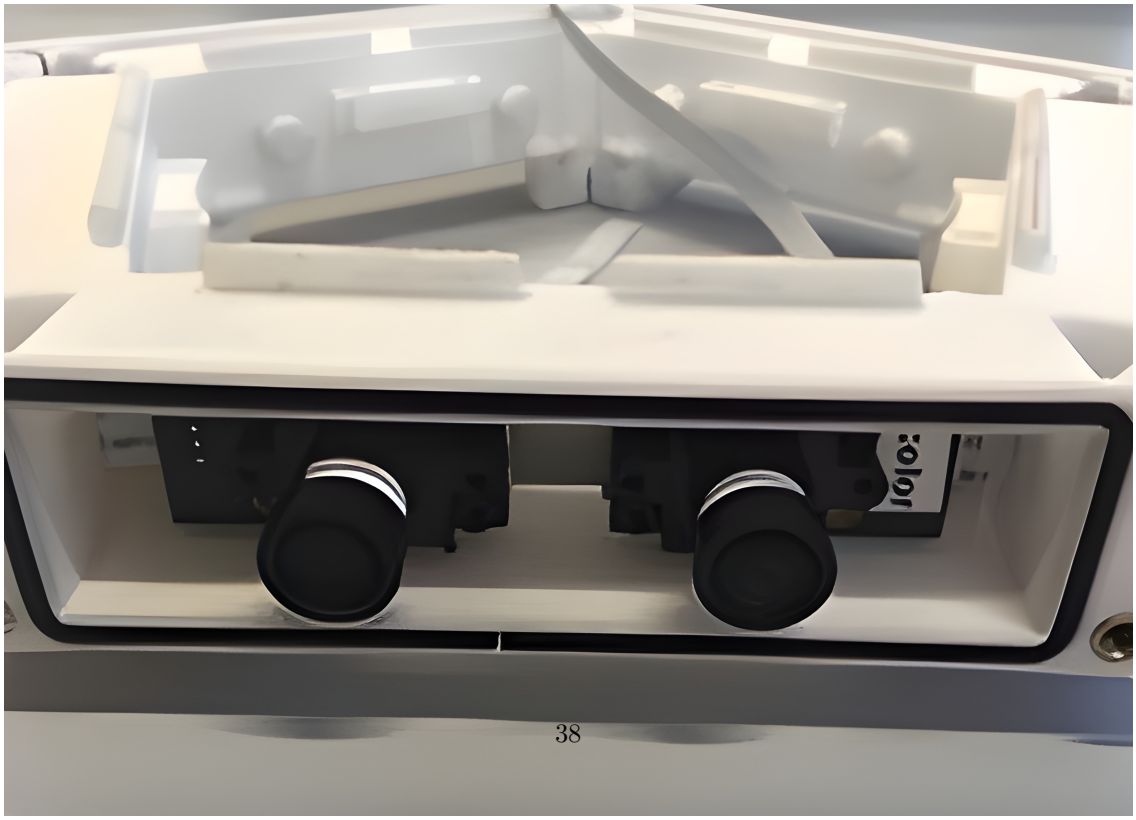
Image-sensor-MT9V024-datasheet-ON_Semiconductor.pdf

[25] https://github.com/ethz-asl/kalibr

[26] Leica ScanStation P15 Datasheet URL: https://github.com/Canpu999/Trimbot-Wageningen-SLAM-Dataset/blob/main/Leica_ScanStation_P15_datasheet.pdf



(a) Top View



(b) Side View

Figure D.3: Figure (a) (b) shows the panoramic stereo camera from the top view and side view

both measurements are not exactly synchronized. As a consequence, there may be wrong RGB values on moving objects' point clouds. A moving leaf's point cloud (taken at one timestamp) will show the sky's color (taken at another timestamp) if the leaf moved at one timestamp (when the laser scanner was acquiring data) to another place at another timestamp (when the RGB sensor was acquiring data). Reflectance properties of glossy leaves can also result in color changes. Thus, in Figure D.4, the tree's point clouds show a mixed color (green and sky-white). That is the reason why we did not include the point cloud with RGB data into the Trimbot Wageningen SLAM Dataset.

Appendix E. Additional Qualitative Results

E.1 Robustness to Environment Change

As sunlight and the scene appearance outdoors will influence an image's appearance directly, thus we vary sunlight from different time periods and scene appearance from different places in the following experiment to demonstrate the robustness of the proposed framework. Figure E.1 shows the experimental setting for the robustness test. In Scene 1, the robot drove straight along route A under sunlight A. In Scene 2, the robot drove straightly along route B (whose travelled distance is same with that of route A) under the same sunlight A. In Scene 3, the robot drove straightly along the route A, but under the different sunlight B. Since factors other than scene appearance and lighting should not influence the image appearance directly, we classify the other factors as Uncontrolled Random Factors. Table E.1 gives the precise value of each factor, and the values of the uncontrolled random factors are from a weather website [27].

Figure E.2 shows the three scenes and the related 3D reconstruction results in surface mesh format by the proposed framework. As acquiring ground truth requires considerable expense and effort by a big team, we only show the corresponding qualitative results. Compare the shadows, trees, posts and bushes in the example images (Figure E.2 a c e) with the corresponding shapes in the reconstructed 3D meshes (Figure E.2 b d f). The high quality and believable shapes, even with different lighting and viewpoint, demonstrate the high quality of the 3D reconstruction by our proposed framework. Further robustness tests against other factors (e.g. temperature, wind speed, long operating time, humidity, foggy or snowy weather) are future work.

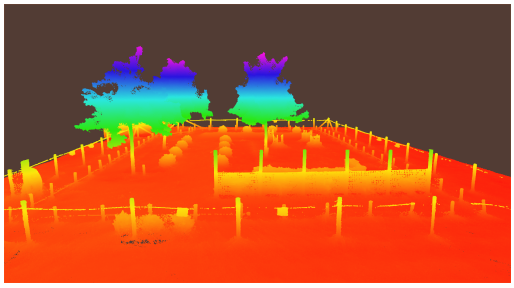
Table E.1: Scene Definition with Different Condition Setting

	Controlled Var		**Uncontrolled Random Factors**						
Scene	**route**	**sun-light**	**time**	**date**	**temp**	**wind speed**	**humid**	**dew point**	**pressure**
1	A	A	$\sim 15:30$	2017-05-17	$22.5°C$	12.7 km/h	76%	$15.7°C$	995.6Mb
2	B	A	$\sim 15:30$	2017-05-17	$22.5°C$	12.7 km/h	76%	$15.7°C$	995.6Mb
3	A	B	$\sim 11:00$	2018-06-27	$20.5°C$	12.7 km/h	75%	$12.7°C$	1003.4Mb

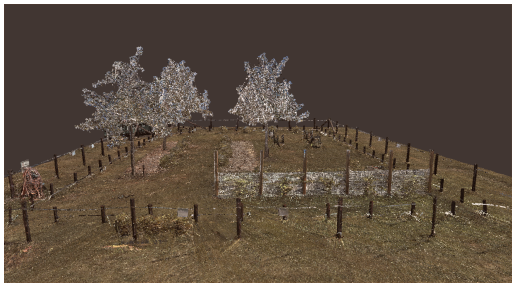
E.2 Fair Comparison with More Open-source Frameworks

Orbslam3 (Campos et al., 2021) and Open3D (Zhou et al., 2018) needed hints in Section 4.3.2 to make them work. In this subsection, we will compare all the frameworks when using only one stereo vision camera (which consists of the image sensor Cam0 and Cam1) without any hints provided to any framework. We use the initial parameter setting of Open3D for a test (denoted by Open3D-initial). For Orbslam3, we test their RGBD SLAM algorithm (denoted by Orbslam3-rgbd), their stereo SLAM algorithm (denoted by Orbslam3-stereo), and their monocular SLAM algorithm (denoted by Orbslam3-monocular). For the parameter setting of each SLAM algorithm

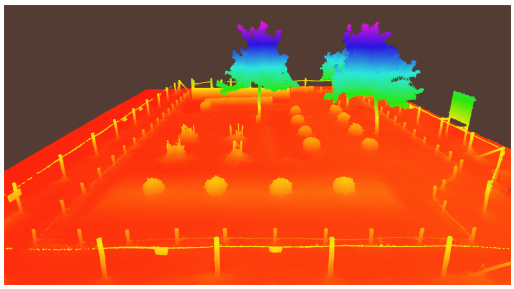
[27] `https://tcktcktck.org/netherlands/gelderland/wageningen`



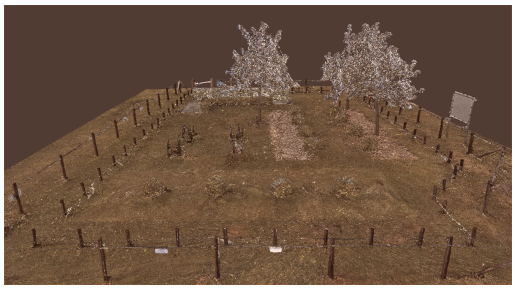
(a) The point cloud from view 1 with colored height



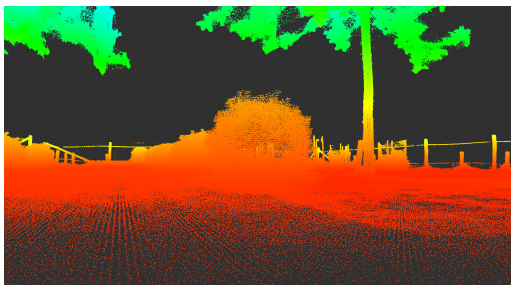
(b) The point cloud from view 1 with registered RGB data



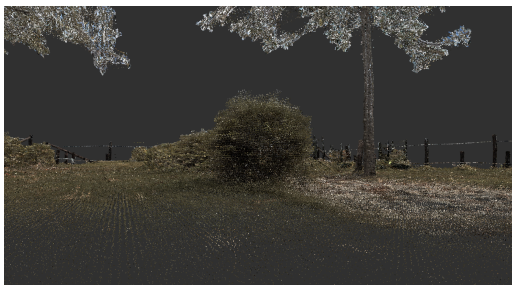
(c) The point cloud from view 2 with colored height



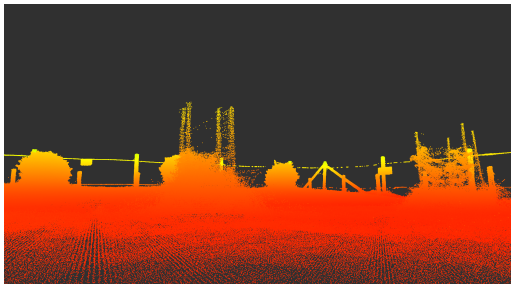
(d) The point cloud from view 2 with registered RGB data



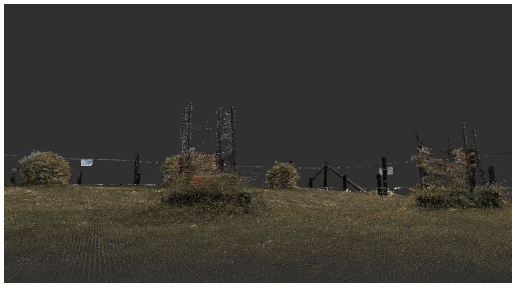
(e) The point cloud from view 3 with colored height



(f) The point cloud from view 3 with registered RGB data



(g) The point cloud from view 4 with colored height



(h) The point cloud from view 4 with registered RGB data

Figure D.4: Figure (a) (c) (e) (g) show the point clouds with colored height. Figure (b) (d) (f) (h) show the point clouds with registered RGB data. All point clouds are from the Leica ScanStation P15. The images are from our partner Robert Bosch GmbH in the Trimbot2020 project.

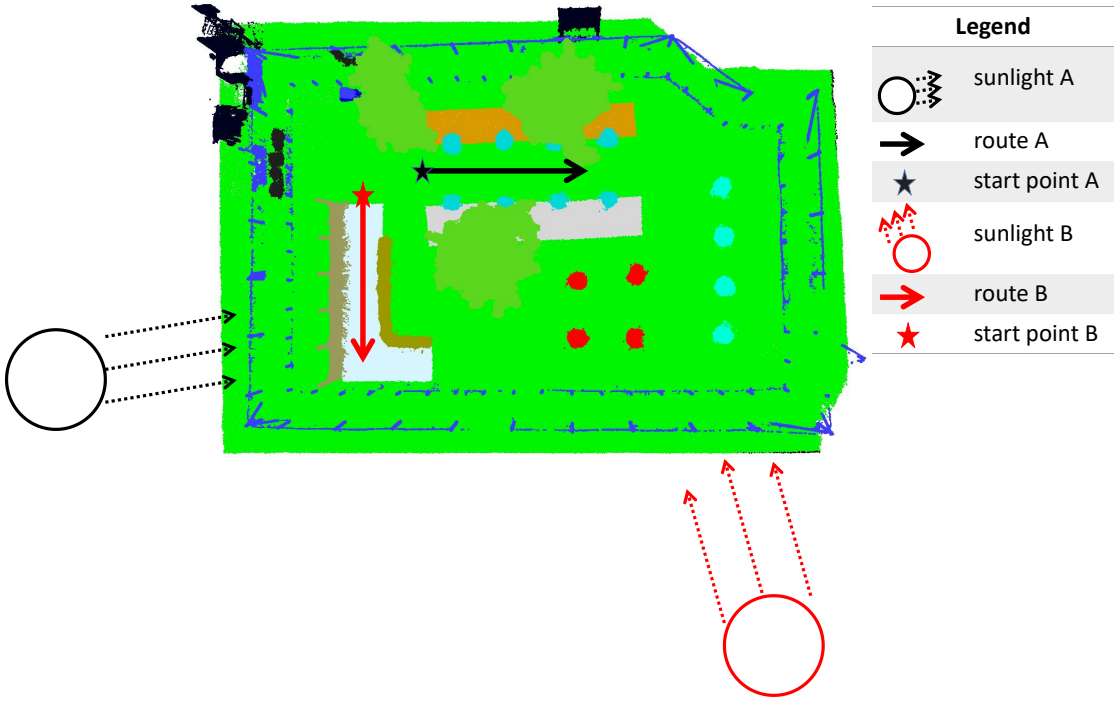


Figure E.1: The experiment setting in the robustness test

from Orbslam3, please refer to the video at `https://youtu.be/4luADtHNbuA`. In this video[28], we also show the performance of each algorithm from Orbslam3 on our Trimbot Wageningen SLAM dataset when having a random test[29].

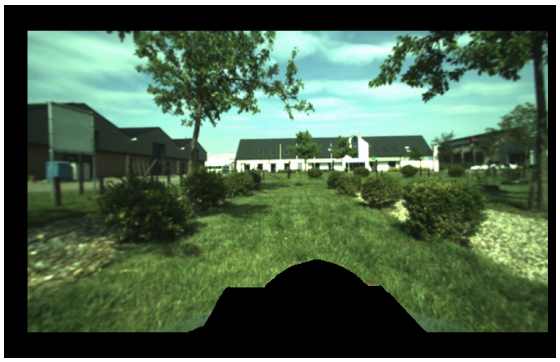
Figure E.3 shows the global pose trajectory of each framework. The results of Orbslam3 are from the best trials among tens of random trials. Method "Ours" is the method proposed in this paper with panoramic stereo images as input. Method "Ours-single-stereo" denotes a method which is derived from the "Ours" method, but with the input from only one stereo camera. Figure E.3 shows that all of the algorithms sucessfully estimate poses initially, but eventually fail, except "Ours" when traversing the whole dataset. "Ours-single-stereo" seems to work better compared with the external counterparts, although it still fails at the latter part of the global pose trajectory. The trajectory of Orbslam3-monocular (cyan trajectory) has only one dot to show at the starting point, near the coordinate (0 m, -1 m), because Orbslam3-monocular almost lost tracking every frame. Given the bad performance of each external algorithm, it is meaningless to calculate the corresponding quantitative results. It is also the reason why we needed to provide "some extra help" to the external counterparts to get results to compare with in Section 4.3.2.

E.3 Comparison with Software 'ContextCapture'

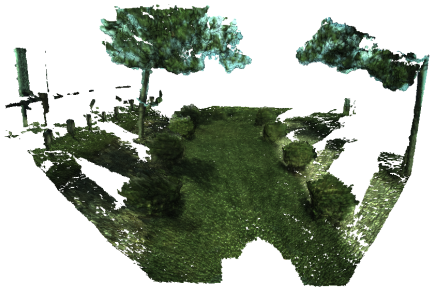
Above, we compared our proposed framework with the latest popular frameworks Orbslam3 (Campos et al., 2021) and Open3D (Zhou et al., 2018). The experimental results have shown our framework's performance superiority over the two open-source frameworks. Here, we compare the proposed framework with the commercial software called "ContextCapture Center" from Bent-

[28]There are many false extracted features near the edge of the robot in the above video. We refined the data input of Orbslam3 by masking more areas near the robot and tuned the parameters to maximize Orbslam3's performance. The modified video (URL: `https://youtu.be/9TaayXl4mJ4`) shows the corresponding performance.

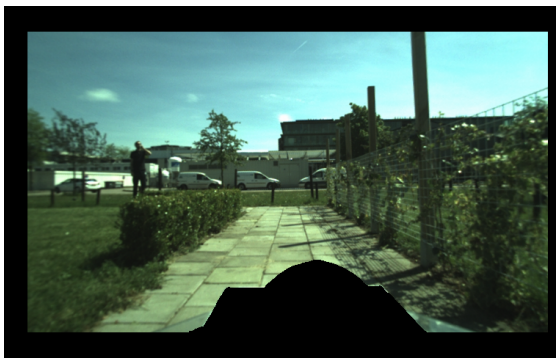
[29]Due to the stochastic property of the algorithm RANSAC (Fischler and Bolles, 1981) in the framework Orbslam3, the results from Orbslam3 are different each time.



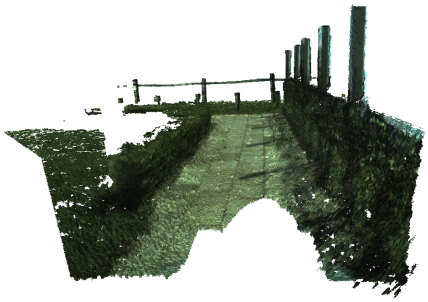
(a) Scene 1



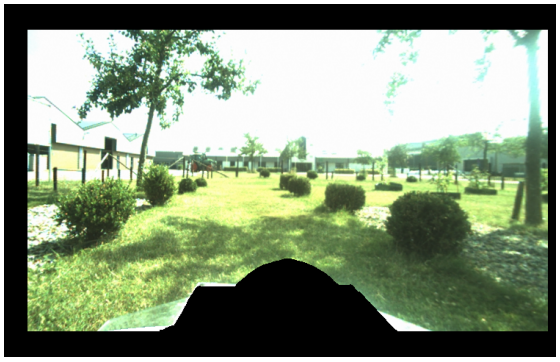
(b) Reconstructed Model 1



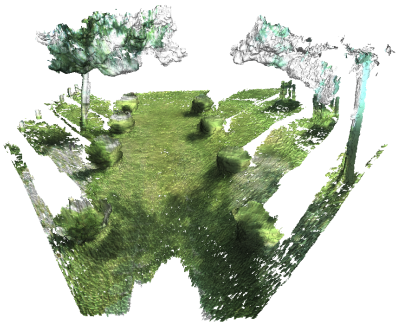
(c) Scene 2



(d) Reconstructed Model 2



(e) Scene 3



(f) Reconstructed Model 3

Figure E.2: Figure (a) (c) (e) shows the scenes and Figure (b) (d) (f) shows the reconstructed models.

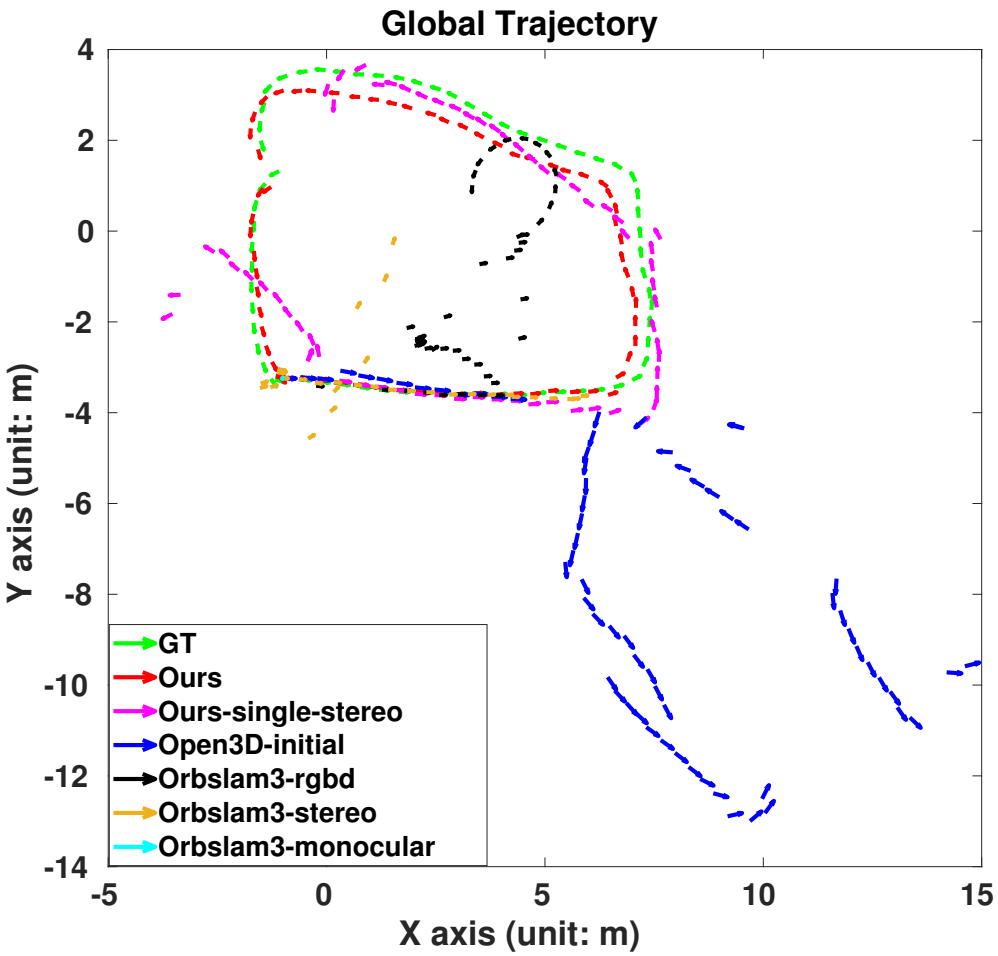
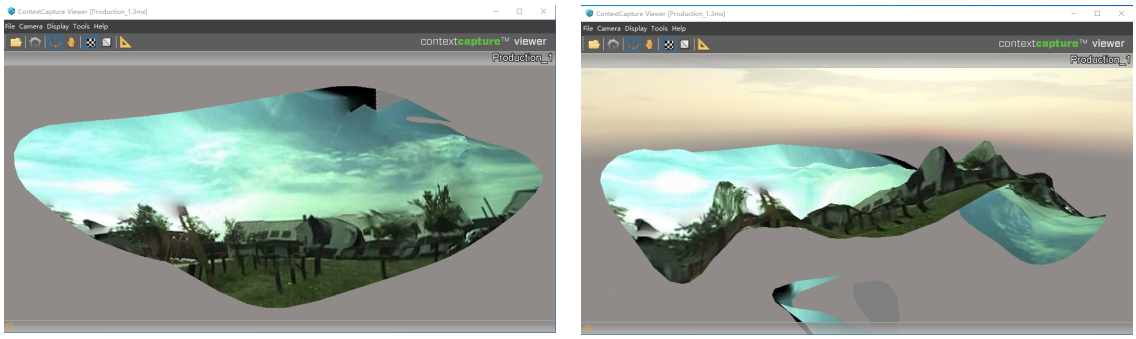


Figure E.3: The global pose trajectory from each algorithm



(a) One View (b) Another View

Figure E.4: Figures (a) and (b) show the reconstructed models from the professional 3D reconstruction Software 'ContextCapture'.

ley[30], which is known for its image-based 3D reconstruction and aerial photogrammetry. Because the commercial software ContextCapture's 3D reconstruction framework does not support 360° image-based 3D reconstruction, we could only input the RGB images from our image sensor Cam0 in our Wageningen SLAM Dataset. Figure E.4 shows the reconstructed meshes from two views. From the strange reconstruction results, we could easily see that the ContextCapture reconstruction system failed at our task. The reason is that the real robot's fast movement and large transformation between frames when navigating in the real outdoor garden lead ContextCapture to lose tracking (please read the acquisition report[31] from ContextCapture). The quality report[32] from the software ContextCapture reveals more reconstruction details about its failure.

References

Ahmadi, M., Naeini, A.A., Arjmandi, Z., Zhang, Y., Sheikholeslami, M.M., Sohn, G., 2023. Hdpv-slam: Hybrid depth-augmented panoramic visual slam for mobile mapping system with tilted lidar and panoramic visual camera. arXiv preprint arXiv:2301.11823 .

Alam, M.S., Alam, M., Tufail, M., Khan, M.U., Güneş, A., Salah, B., Nasir, F.E., Saleem, W., Khan, M.T., 2022. Tobset: A new tobacco crop and weeds image dataset and its utilization for vision-based spraying by agricultural robots. Applied Sciences 12, 1308.

Ao, S., Hu, Q., Yang, B., Markham, A., Guo, Y., 2021. Spinnet: Learning a general surface descriptor for 3d point cloud registration, in: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, pp. 11753–11762.

Barath, D., Mishkin, D., Eichhardt, I., Shipachev, I., Matas, J., 2021. Efficient initial pose-graph generation for global sfm, in: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, pp. 14546–14555.

Besl, P.J., McKay, N.D., 1992. Method for registration of 3-d shapes, in: Sensor fusion IV: control paradigms and data structures, Spie. pp. 586–606.

Campos, C., Elvira, R., Rodríguez, J.J.G., Montiel, J.M., Tardós, J.D., 2021. Orb-slam3: An accurate open-source library for visual, visual–inertial, and multimap slam. IEEE Transactions on Robotics 37, 1874–1890.

[30] https://www.bentley.com/software/contextcapture/

[31] ContextCapture's acquisition report: https://drive.google.com/file/d/1k_oUyolomh2LmJv3ATyrUnxA5sc--bxE/view?usp=sharing

[32] ContextCapture's quality report: https://drive.google.com/file/d/1Q5DqU8XOqrO1ORWt9dWxitlhvU81vRjQ/view?usp=sharing

Carlone, L., Rosen, D.M., Calafiore, G., Leonard, J.J., Dellaert, F., 2015a. Lagrangian duality in 3d slam: Verification techniques and optimal solutions, in: 2015 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), IEEE. pp. 125–132.

Carlone, L., Tron, R., Daniilidis, K., Dellaert, F., 2015b. Initialization techniques for 3d slam: a survey on rotation estimation and its use in pose graph optimization, in: 2015 IEEE international conference on robotics and automation (ICRA), IEEE. pp. 4597–4604.

Chebrolu, N., Lottes, P., Schaefer, A., Winterhalter, W., Burgard, W., Stachniss, C., 2017. Agricultural robot dataset for plant classification, localization and mapping on sugar beet fields. The International Journal of Robotics Research 36, 1045–1052.

Chen, H., Hu, W., Yang, K., Bai, J., Wang, K., 2021. Panoramic annular slam with loop closure and global optimization. Applied Optics 60, 6264–6274.

Chen, H., Wang, K., Hu, W., Yang, K., Cheng, R., Huang, X., Bai, J., 2019. Palvo: visual odometry based on panoramic annular lens. Optics express 27, 24481–24497.

Chen, W., Shang, G., Ji, A., Zhou, C., Wang, X., Xu, C., Li, Z., Hu, K., 2022. An overview on visual slam: From tradition to semantic. Remote Sensing 14, 3010.

Cordts, M., Omran, M., Ramos, S., Rehfeld, T., Enzweiler, M., Benenson, R., Franke, U., Roth, S., Schiele, B., 2016. The cityscapes dataset for semantic urban scene understanding, in: Proceedings of the IEEE conference on computer vision and pattern recognition, pp. 3213–3223.

Curless, B., Levoy, M., 1996. A volumetric method for building complex models from range images, in: Proceedings of the 23rd annual conference on Computer graphics and interactive techniques, pp. 303–312.

Dosovitskiy, A., Fischer, P., Ilg, E., Hausser, P., Hazirbas, C., Golkov, V., Van Der Smagt, P., Cremers, D., Brox, T., 2015. Flownet: Learning optical flow with convolutional networks, in: Proceedings of the IEEE international conference on computer vision, pp. 2758–2766.

Engel, J., Koltun, V., Cremers, D., 2017. Direct sparse odometry. IEEE transactions on pattern analysis and machine intelligence 40, 611–625.

Engel, J., Stückler, J., Cremers, D., 2015. Large-scale direct slam with stereo cameras, in: 2015 IEEE/RSJ international conference on intelligent robots and systems (IROS), IEEE. pp. 1935–1942.

Fan, Y., Zhang, Q., Tang, Y., Liu, S., Han, H., 2022. Blitz-slam: A semantic slam in dynamic environments. Pattern Recognition 121, 108225.

Fischler, M.A., Bolles, R.C., 1981. Random sample consensus: A paradigm for model fitting with applications to image analysis and automated cartography. Commun. ACM 24, 381–395. URL: https://doi.org/10.1145/358669.358692, doi:10.1145/358669.358692.

Geiger, A., Lenz, P., Stiller, C., Urtasun, R., 2013. Vision meets robotics: The kitti dataset. The International Journal of Robotics Research 32, 1231–1237.

Goodfellow, I., Pouget-Abadie, J., Mirza, M., Xu, B., Warde-Farley, D., Ozair, S., Courville, A., Bengio, Y., 2014. Generative adversarial nets, in: Advances in neural information processing systems, pp. 2672–2680.

Grisetti, G., Kümmerle, R., Stachniss, C., Burgard, W., 2010. A tutorial on graph-based slam. IEEE Intelligent Transportation Systems Magazine 2, 31–43.

Grosso, R., Zint, D., 2022. A parallel dual marching cubes approach to quad only surface reconstruction. The Visual Computer 38, 1301–1316.

Gu, X., Tang, C., Yuan, W., Dai, Z., Zhu, S., Tan, P., 2022. Rcp: Recurrent closest point for point cloud, in: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, pp. 8216–8226.

Guo, X.d., Wang, Z.b., Zhu, W., He, G., Deng, H.b., Lv, C.x., Zhang, Z.h., 2022. Research on dso vision positioning technology based on binocular stereo panoramic vision system. Defence Technology 18, 593–603.

Hirschmuller, H., 2005. Accurate and efficient stereo processing by semi-global matching and mutual information, in: Computer Vision and Pattern Recognition, 2005. CVPR 2005. IEEE Computer Society Conference on, IEEE. pp. 807–814.

Honegger, D., Sattler, T., Pollefeys, M., 2017. Embedded real-time multi-baseline stereo, in: 2017 IEEE International Conference on Robotics and Automation (ICRA), IEEE. pp. 5245–5250.

Hu, N., Wang, S., Wang, X., Cai, Y., Su, D., Nyamsuren, P., Qiao, Y., Jiang, Y., Hai, B., Wei, H., 2022. Lettucemot: A dataset of lettuce detection and tracking with re-identification of re-occurred plants for agricultural robots. Frontiers in Plant Science 13.

Huang, X., Li, S., Zuo, Y., Fang, Y., Zhang, J., Zhao, X., 2022. Unsupervised point cloud registration by learning unified gaussian mixture models. IEEE Robotics and Automation Letters .

Huang, X., Mei, G., Zhang, J., Abbas, R., 2021. A comprehensive survey on point cloud registration. arXiv preprint arXiv:2103.02690 .

Huynh, D.Q., 2009. Metrics for 3d rotations: Comparison and analysis. Journal of Mathematical Imaging and Vision 35, 155–164.

Izadi, S., Kim, D., Hilliges, O., Molyneaux, D., Newcombe, R., Kohli, P., Shotton, J., Hodges, S., Freeman, D., Davison, A., et al., 2011. Kinectfusion: real-time 3d reconstruction and interaction using a moving depth camera, in: Proceedings of the 24th annual ACM symposium on User interface software and technology, pp. 559–568.

Ji, S., Qin, Z., Shan, J., Lu, M., 2020. Panoramic slam from a multiple fisheye camera rig. ISPRS Journal of Photogrammetry and Remote Sensing 159, 169–183.

Junior, E.M.O., Santos, D.R., Miola, G.A.R., et al., 2022. A new variant of the icp algorithm for pairwise 3d point cloud registration. American Academic Scientific Research Journal for Engineering, Technology, and Sciences 85, 71–88.

Kang, J., Zhang, Y., Liu, Z., Sit, A., Sohn, G., 2021. Rpv-slam: Range-augmented panoramic visual slam for mobile mapping system with panoramic camera and tilted lidar, in: 2021 20th International Conference on Advanced Robotics (ICAR), IEEE. pp. 1066–1072.

Kazerouni, I.A., Fitzgerald, L., Dooly, G., Toal, D., 2022. A survey of state-of-the-art on visual slam. Expert Systems with Applications , 117734.

Lewiner, T., Lopes, H., Vieira, A.W., Tavares, G., 2003. Efficient implementation of marching cubes' cases with topological guarantees. Journal of graphics tools 8, 1–15.

Liu, W., Wu, H., Chirikjian, G.S., 2021. Lsg-cpd: Coherent point drift with local surface geometry for point cloud registration, in: Proceedings of the IEEE/CVF International Conference on Computer Vision, pp. 15293–15302.

Lorensen, W.E., Cline, H.E., 1987. Marching cubes: A high resolution 3d surface construction algorithm. ACM siggraph computer graphics 21, 163–169.

Marin, G., Zanuttigh, P., Mattoccia, S., 2016. Reliable fusion of tof and stereo depth driven by confidence measures, in: European Conference on Computer Vision, Springer. pp. 386–401.

Mayer, N., Ilg, E., Hausser, P., Fischer, P., Cremers, D., Dosovitskiy, A., Brox, T., 2016. A large dataset to train convolutional networks for disparity, optical flow, and scene flow estimation, in: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp. 4040–4048.

Mendes, E., Koch, P., Lacroix, S., 2016. Icp-based pose-graph slam, in: 2016 IEEE International Symposium on Safety, Security, and Rescue Robotics (SSRR), IEEE. pp. 195–200.

Menze, M., Geiger, A., 2015. Object scene flow for autonomous vehicles, in: Proceedings of the IEEE conference on computer vision and pattern recognition, pp. 3061–3070.

Moreira, G., Marques, M., Costeira, J.P., 2021a. Fast pose graph optimization via krylov-schur and cholesky factorization, in: Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision, pp. 1898–1906.

Moreira, G., Marques, M., Costeira, J.P., 2021b. Rotation averaging in a split second: A primal-dual method and a closed-form for cycle graphs, in: Proceedings of the IEEE/CVF International Conference on Computer Vision, pp. 5452–5460.

Mur-Artal, R., Montiel, J.M.M., Tardos, J.D., 2015. Orb-slam: a versatile and accurate monocular slam system. IEEE transactions on robotics 31, 1147–1163.

Mur-Artal, R., Tardós, J.D., 2017. Orb-slam2: An open-source slam system for monocular, stereo, and rgb-d cameras. IEEE transactions on robotics 33, 1255–1262.

Myronenko, A., Song, X., 2010. Point set registration: Coherent point drift. IEEE transactions on pattern analysis and machine intelligence 32, 2262–2275.

Ng, P.C., Henikoff, S., 2003. Sift: Predicting amino acid changes that affect protein function. Nucleic acids research 31, 3812–3814.

Poggi, M., Agresti, G., Tosi, F., Zanuttigh, P., Mattoccia, S., 2019. Confidence estimation for tof and stereo sensors and its application to depth data fusion. IEEE Sensors Journal 20, 1411–1421.

Poggi, M., Mattoccia, S., 2016. Deep stereo fusion: combining multiple disparity hypotheses with deep-learning, in: 2016 Fourth International Conference on 3D Vision (3DV), IEEE. pp. 138–147.

Poggi, M., Tosi, F., Batsos, K., Mordohai, P., Mattoccia, S., 2021. On the synergies between machine learning and binocular stereo for depth estimation from images: a survey. IEEE Transactions on Pattern Analysis and Machine Intelligence .

Polvara, R., Molina, S., Hroob, I., Papadimitriou, A., Konstantinos, T., Giakoumis, D., Likothanassis, S., Tzovaras, D., Cielniak, G., Hanheide, M., 2022. Blt dataset: acquisition of the agricultural bacchus long-term dataset with automated robot deployment. Journal of Field Robotics, Agricultural Robots for Ag 4.0 Under Review.

Pu, C., Fisher, R.B., 2019. Udfnet: Unsupervised disparity fusion with adversarial networks, in: 2019 IEEE International Conference on Image Processing (ICIP), IEEE. pp. 1765–1769.

Pu, C., Li, N., Tylecek, R., Fisher, B., 2018. Dugma: Dynamic uncertainty-based gaussian mixture alignment, in: 2018 International Conference on 3D Vision (3DV), IEEE. pp. 766–774.

Pu, C., Song, R., Tylecek, R., Li, N., Fisher, R.B., 2019. Sdf-man: Semi-supervised disparity fusion with multi-scale adversarial networks. Remote Sensing 11, 487.

Qin, T., Li, P., Shen, S., 2018. Vins-mono: A robust and versatile monocular visual-inertial state estimator. IEEE Transactions on Robotics 34, 1004–1020.

Rublee, E., Rabaud, V., Konolige, K., Bradski, G., 2011. Orb: An efficient alternative to sift or surf, in: 2011 International conference on computer vision, Ieee. pp. 2564–2571.

Sandström, E., Oswald, M.R., Kumar, S., Weder, S., Yu, F., Sminchisescu, C., Van Gool, L., 2022. Learning online multi-sensor depth fusion. arXiv preprint arXiv:2204.03353 .

Sattler, T., Tylecek, R., Brox, T., Pollefeys, M., Fisher, R.B., 2017. 3d reconstruction meets semantics–reconstruction challenge 2017, in: ICCV Workshop, Venice, Italy, Tech. Rep.

Schonberger, J.L., Frahm, J.M., 2016. Structure-from-motion revisited, in: Proceedings of the IEEE conference on computer vision and pattern recognition, pp. 4104–4113.

Segal, A., Haehnel, D., Thrun, S., 2009. Generalized-icp., in: Robotics: science and systems, Seattle, WA. p. 435.

Shan, T., Englot, B., 2018. Lego-loam: Lightweight and ground-optimized lidar odometry and mapping on variable terrain, in: 2018 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), IEEE. pp. 4758–4765.

Sumikura, S., Shibuya, M., Sakurada, K., 2019. Openvslam: A versatile visual slam framework, in: Proceedings of the 27th ACM International Conference on Multimedia, pp. 2292–2295.

Szeliski, R., 2010. Computer vision: algorithms and applications. Springer Science & Business Media.

Tylecek, R., Fisher, R.B., 2020. Trimbot2020 dataset for garden navigation and bush trimming, in: Edinburgh DataVault. 10.7488/9f9de786-5e58-4bca-9279-f1d7ffddda41.

Wang, D., Wang, J., Tian, Y., Hu, K., Xu, M., 2022. Pal-slam: a feature-based slam system for a panoramic annular lens. Optics Express 30, 1099–1113.

Whelan, T., Salas-Moreno, R.F., Glocker, B., Davison, A.J., Leutenegger, S., 2016. Elasticfusion: Real-time dense slam and light source estimation. The International Journal of Robotics Research 35, 1697–1716.

Wu, B., Ma, J., Chen, G., An, P., 2021. Feature interactive representation for point cloud registration, in: Proceedings of the IEEE/CVF International Conference on Computer Vision, pp. 5530–5539.

Xu, X., Zhang, L., Yang, J., Cao, C., Wang, W., Ran, Y., Tan, Z., Luo, M., 2022. A review of multi-sensor fusion slam systems based on 3d lidar. Remote Sensing 14, 2835.

Zakeri, F.S., Sjöström, M., Keinert, J., 2020. Guided optimization framework for the fusion of time-of-flight with stereo depth. Journal of Electronic Imaging 29, 053016.

Zeng, A., Song, S., Nießner, M., Fisher, M., Xiao, J., Funkhouser, T., 2017. 3dmatch: Learning local geometric descriptors from rgb-d reconstructions, in: Proceedings of the IEEE conference on computer vision and pattern recognition, pp. 1802–1811.

Zhang, S., Zheng, L., Tao, W., 2021. Survey and evaluation of rgb-d slam. IEEE Access 9, 21367–21387.

Zhang, Y., Huang, F., 2021. Panoramic visual slam technology for spherical images. Sensors 21, 705.

Zhao, X., Hu, Q., Zhang, X., Wang, H., 2022. An orb-slam3 autonomous positioning and orientation approach using 360-degree panoramic video, in: 2022 29th International Conference on Geoinformatics, IEEE. pp. 1–7.

Zhou, Q.Y., Koltun, V., 2013. Dense scene reconstruction with points of interest. ACM Transactions on Graphics (ToG) 32, 1–8.

Zhou, Q.Y., Park, J., Koltun, V., 2016. Fast global registration, in: European conference on computer vision, Springer. pp. 766–782.

Zhou, Q.Y., Park, J., Koltun, V., 2018. Open3d: A modern library for 3d data processing. arXiv preprint arXiv:1801.09847 .

Zhu, X.X., Yu, Y., Wang, P.F., Lin, M.J., Zhang, H.R., Cao, Q.X., 2019. A visual slam system based on the panoramic camera, in: 2019 IEEE International Conference on Real-time Computing and Robotics (RCAR), IEEE. pp. 53–58.