sensors

MDPI

Article

FaSS-MVS: Fast Multi-View Stereo with Surface-Aware Semi-Global Matching from UAV-Borne Monocular Imagery

Boitumelo Ruf [1,*], Martin Weinmann [2] and Stefan Hinz [2]

[1] Fraunhofer Institute of Optronics, System Technologies and Image Exploitation IOSB, 76131 Karlsruhe, Germany
[2] Institute of Photogrammetry and Remote Sensing, Karlsruhe Institute of Technology, 76131 Karlsruhe, Germany; martin.weinmann@kit.edu (M.W.); stefan.hinz@kit.edu (S.H.)
* Correspondence: boitumelo.ruf@iosb.fraunhofer.de

Abstract: With FaSS-MVS, we present a fast, surface-aware semi-global optimization approach for multi-view stereo that allows for rapid depth and normal map estimation from monocular aerial video data captured by unmanned aerial vehicles (UAVs). The data estimated by FaSS-MVS, in turn, facilitate online 3D mapping, meaning that a 3D map of the scene is immediately and incrementally generated as the image data are acquired or being received. FaSS-MVS is composed of a hierarchical processing scheme in which depth and normal data, as well as corresponding confidence scores, are estimated in a coarse-to-fine manner, allowing efficient processing of large scene depths, such as those inherent in oblique images acquired by UAVs flying at low altitudes. The actual depth estimation uses a plane-sweep algorithm for dense multi-image matching to produce depth hypotheses from which the actual depth map is extracted by means of a surface-aware semi-global optimization, reducing the fronto-parallel bias of Semi-Global Matching (SGM). Given the estimated depth map, the pixel-wise surface normal information is then computed by reprojecting the depth map into a point cloud and computing the normal vectors within a confined local neighborhood. In a thorough quantitative and ablative study, we show that the accuracy of the 3D information computed by FaSS-MVS is close to that of state-of-the-art offline multi-view stereo approaches, with the error not even an order of magnitude higher than that of COLMAP. At the same time, however, the average runtime of FaSS-MVS for estimating a single depth and normal map is less than 14% of that of COLMAP, allowing us to perform online and incremental processing of full HD images at 1–2 Hz.

Keywords: multi-view stereo; plane-sweep multi-image matching; semi-global optimization; surface-awareness; online processing; oblique aerial imagery; UAVs

Citation: Ruf, B.; Weinmann, M.; Hinz, S. FaSS-MVS: Fast Multi-View Stereo with Surface-Aware Semi-Global Matching from UAV-Borne Monocular Imagery. *Sensors* **2024**, *24*, 6397. https://doi.org/10.3390/s24196397

Academic Editors: Marios Antonakakis and Michalis Zervakis

Received: 29 August 2024
Revised: 25 September 2024
Accepted: 29 September 2024
Published: 2 October 2024

1. Introduction

The image-based estimation of depth maps and geometry by dense image matching (DIM) and multi-view stereo (MVS) is one of the fundamental tasks in photogrammetry, remote sensing and computer vision. It enables a wide range of high-level applications such as navigation and path planning for autonomous mobile robot (AMR) systems, urban planning and monitoring, simulation and 3D modeling, as well as virtual, mixed and augmented reality. The ongoing development and increasing availability of commercial off-the-shelf (COTS) UAVs is opening up new possibilities and applications for image-based 3D mapping, both offline and online. In recent years, for example, the use of COTS UAVs by emergency services such as firefighters and medical rescue services has been steadily increasing, which in turn facilitates rapid and large-scale situation assessment or enables monitoring of areas inaccessible to ground forces [1,2]. In this context, image-based techniques and photogrammetry based on aerial reconnaissance are a key element in assisting the rescue workers, provided that the environmental conditions, e.g., weather and daytime, allow a visual inspection [2].

Meanwhile, there is a large collection of software toolboxes, such as COLMAP [3,4] or OpenMVS (http://cdcseacave.github.io, accessed on 27 August 2024), for performing offline photogrammetric 3D reconstruction, allowing to accurately reconstruct the disaster site from aerial imagery. Highly accurate 3D reconstructions can be used to accurately assess the damage caused by an incident or the structural integrity of a partially collapsed building. First responders, however, require rapid and extensive 3D mapping of the disaster site in a short period of time, rather than a highly accurate 3D reconstruction. This allows them to quickly assess the situation, prioritize resources, and plan routes and operations through hard-to-reach areas.

MVS methods can be divided into three categories based on the resulting scene representation, namely volumetric, point cloud-based, and depth map-based [5]. While the first two categories typically work on the full extent of the scene, depth map-based methods typically separate the depth estimation process from the depth map fusion process. This makes such methods typically more versatile, especially with respect to online and iterative 3D mapping, since the depth map estimation can be performed on a locally limited set of images, resulting in separate depth maps that can be subsequently fused into different scene representations, e.g., true orthophoto and 2.5D height map [6], 3D point cloud or mesh [7]. With this in mind, this work proposes and investigates an approach for fast multi-view stereo, by combining the SGM algorithm with a true multi-image matching approach. In it, we propose to:

- use an efficient plane-sweep sampling to perform hierarchical dense multi-image matching;
- adopt the SGM algorithm to work with depth hypotheses generated by plane-sweep sampling;
- extend the SGM algorithm to favor not only fronto-parallel surfaces in the computation of dense depth maps, by incorporating a surface-aware regularization based on local surface normals;
- implement and deploy it on modern GPU hardware to efficiently compute dense depth, normal and confidence maps online from image sequences.

Our fast, surface-aware semi-global optimization approach for multi-view stereo (FaSS-MVS) is designed to assist specialized first responders in deploying a high-end COTS UAV in combination with a ground control station (GCS) to rapidly assess the situation through aerial reconnaissance. It is assumed that the image data are streamed down to the GCS during the operation of the UAV, where they can be processed by more powerful hardware. Even though this approach is proposed with the above use case in mind, it is not limited to airborne data and can also be used to perform incremental and online 3D mapping by a ground-based robot or sensor system. Although learning-based approaches using deep neural networks [5,8–10] have made significant improvements in recent years, with FaSS-MVS we still rely on a traditional processing pipeline to ensure a high reliability and explainability in a practical application. We evaluate FaSS-MVS on two public datasets for dense MVS with accurate ground truth, and on two use-case-specific datasets. It combines and extends our previous work presented in Ruf et al. [11,12], by:

- a more detailed description of the algorithms used;
- extending the plane-sweep multi-image matching to use non-fronto-parallel plane orientations;
- improving the surface-aware regularization of the SGM algorithm;
- using a different confidence measure for estimating the confidence map;
- a thorough evaluation and ablation study with respect to different aspects and configurations of the approach;
- providing a detailed discussion with respect to the support of rescue workers by aerial reconnaissance.

An earlier and in some parts more elaborate version of this work has already been published as part of the PhD thesis by Ruf [13]. In contrast to the first publication, we have

extended the evaluation and comparison of FaSS-MVS with respect to related approaches from the literature.

1.1. Paper Outline

This paper is organized as follows. In Section 1.2, the related work on incremental image-based 3D mapping for online processing as well as the reconstruction of non-fronto-parallel surfaces using DIM and MVS are briefly summarized. In this, it is also delineated how the presented approach differs from those presented in the related work. In Section 2, the entire processing pipeline of the presented approach is illustrated and outlined with a short overview. This is followed by a detailed description of the implementation and methodology of each step of the processing pipeline as well as a description of the datasets and error metrics used for evaluation. The results of the conducted experiments are presented in Section 3. Subsequently, the results are discussed in Section 4 and put into context of the considered use case, before a summary and concluding remarks, as well as a short outlook on future work, are given in Section 5.

1.2. Related Work

Due to the ever-increasing demand for detailed 3D models, the research in the fields of photogrammetry, remote sensing and computer vision has brought up a number of software suites and applications that focus on estimating accurate and dense depth and geometry information from a large set of input images using DIM and MVS. Prominent and widely used representatives of such applications are MVE [14], PMVS [15], SURE [16,17], COLMAP [4], ACMMP [18], and OpenMVS, to name a few. However, these approaches are designed for offline processing, with the goal of accuracy and completeness of the resulting 3D model, assuming that all input data are available at the time of reconstruction and that there are no critical constraints on computation time or hardware resources.

In contrast, the goal of FaSS-MVS is to extract dense depth and geometry information from image sequences as they are acquired, or at least while the image data stream is being received, in case direct processing is not possible due to the acquisition by a small UAV and its limited hardware resources. The focus is therefore on incremental and online processing of the input data by DIM and MVS.

1.2.1. Incremental Camera-Based Mapping for Online Processing

Early work on incremental and online camera-based mapping of the local environment was primarily by robotics and augmented reality (AR) applications [19–21]. The main goal was to robustly localize the camera pose, and thus the sensor carrier, with respect to its environment in order to navigate through the environment or to augment the camera images with additional information. Since the focus of these so-called simultaneous localization and mapping (SLAM) algorithms is on estimating the camera pose and trajectory, the detailed and dense mapping of the environment was rather of secondary interest. In turn, these approaches relied mainly on point features for tracking and mapping rather than direct pixel matching. Since a dense and detailed model of the environment is essential for a convincing AR experience, subsequent work [22,23] has proposed dense mapping simultaneously with image acquisition and camera localization. However, these approaches aim at reconstructing rather small-scale environments and thus use short baseline video clips for image matching, which in turn allows relying on dense optical flow methods to find dense pixel correspondences [22]. In contrast, the input to the approach presented in this paper is assumed to be image data captured by a UAV, typically flying several tens of meters away from the object of interest. The approach presented here is thus designed to densely map a large-scale environment, which in turn requires image matching on a wide baseline, rather than tracking pixel-by-pixel correspondences between successive frames.

The works of Gallup et al. [24] and Pollefeys et al. [25] are part of the early approaches to camera-based mapping and reconstruction of urban environments. They used the plane-sweep algorithm [26] for true multi-image matching to map and reconstruct building

facades in real time from images captured by a vehicle-mounted camera. They rely on vanishing points detected in the input images and data from an additional inertial measurement unit (IMU) to recover the orientations of the building facades and the ground plane relative to the camera. To find the optimal plane configuration for each pixel and, in turn, extract a depth map from the results of the DIM, Pollefeys et al. [25] employ a Bayesian formulation with a subsequent selection of the winner-takes-it-all (WTA) solution, while Gallup et al. [24] minimize a formulated energy functional. Other approaches to urban reconstruction from ground-based imagery, such as those of Furukawa et al. [27], Sinha et al. [28], and Gallup et al. [29], perform piecewise planar reconstruction by fitting multiple differently oriented planes into the scene and optimizing photometric consistency. They minimize an energy functional using a graph-cut algorithm that takes a few minutes on a commodity CPU.

More recent approaches to online camera-based 3D mapping are presented by Kern et al. [6] and Zhao et al. [30]. In their work, the authors propose complete processing pipelines for online and real-time 3D mapping from aerial imagery, which are highly relevant for the use case outlined in this paper. The design of the processing pipelines is similar to the approach of Pollefeys et al. [25], consisting of camera pose estimation, DIM and depth map fusion. For the image-based DIM, Kern et al. [6] relies on the so-called PlaneSweepLib [31], which is based on the work of Gallup et al. [24] and Pollefeys et al. [25]. In contrast, the focus of RTSfM [30] is on efficient and globally consistent Structure-from-Motion (SfM) in real time. For the task of DIM, RTSfM relies on the two-view stereo approach ELAS [32], which is run on image pairs to estimate the depth maps.

The presented approach also uses the plane-sweep algorithm to perform efficient dense multi-image matching. The use of a plane-sweep algorithm for the task of DIM is mainly motivated by its ability to generate depth hypotheses by matching an arbitrary number of input images, as well as the fact that it can be efficiently optimized for massively parallel execution on GPUs, making it particularly suitable for online processing. The use of COTS UAVs as a sensor carrier introduces the need to efficiently handle large scene depths and thus potentially large sampling spaces, due to the relatively low flight altitude and the ability to freely pitch the camera. To limit the sampling space and thus the number of depth hypotheses generated, we embed the plane-sweep algorithm in a hierarchical processing scheme.

1.2.2. Efficient Dense Image Matching Accounting for Non-Fronto-Parallel Surfaces

The so-called SGM algorithm proposed by Hirschmüller [33,34] has become one of the most widely used approaches for both online and offline DIM due to its efficiency and convincing results [16,17,35–37]. In their work, Sinha et al. [38] combine plane-sweep multi-image matching with the SGM algorithm to estimate dense and highly accurate disparity maps. In contrast to the presented approach, Sinha et al. [38] use local slanted planes extracted from feature correspondences to generate disparity hypotheses and use the SGM algorithm to recover a disparity map. They evaluate their approach on a high-resolution stereo benchmark and achieve a significant improvement over the standard SGM algorithm in terms of both runtime and accuracy. The runtime improvement is attributed to the fact that the local plane-sweep allows us to test a locally limited part of the full disparity range for each pixel, thus reducing the computational complexity of the optimization within the SGM algorithm. Similar improvements to overcome the problem of high computational complexity due to the large disparity range inherent in oblique aerial imagery were made by Haala et al. [37] by embedding the SGM in a hierarchical coarse-to-fine processing.

Although many urban environments can be well abstracted by piecewise planar reconstructions, not all structures are fronto-parallel, i.e., their surface orientations are not parallel to the image plane. The original formulation of the SGM algorithm, however, only models a first-order smoothness term and thus favors fronto-parallel surfaces, leading to staircase artifacts when reconstructing slanted surfaces. This should be avoided, especially if the goal is a visually appealing reconstruction of the environment. While Hermann et al. [39]

and Ni et al. [40] propose to include a second-order smoothness assumption in the formulation of the SGM energy function, Scharstein et al. [41] proposes a simpler yet effective improvement to address this issue. Specifically, plane priors, which can be recovered from normal maps or point correspondences, are used to adjust the zero-cost transition within the path aggregation of the SGM, thus penalizing deviations from the surface orientation represented by the prior. The major advantage over the other approaches is that the pixel-wise offset for the zero-cost transition can be computed in advance.

In this work, we use an improved implementation of the SGM algorithm to regularize the cost volume and efficiently extract an accurate dense depth map from the pixel-wise depth hypotheses generated by the plane-sweep DIM. We also adopt the approach presented by Scharstein et al. [41] to account for non-fronto-parallel surfaces by adjusting the zero-cost transition based on surface information stored in a normal map. Moreover, we also propose to reduce the fronto-parallel bias of the SGM algorithm by adjusting the zero-cost transition in the path aggregation based on the gradient of the minimum-cost path. And just like Haala et al. [37], we also embed the SGM in a hierarchical coarse-to-fine processing. Very similar to FaSS-MVS seems to be the approach of Roth and Mayer [42]. They also rely on the improvements proposed by Scharstein et al. [41] and combine the SGM with a plane-sweep DIM. However, their work focuses on estimating disparity images from ground-based stereo image pairs and has only been evaluated on synthetic scenes.

1.2.3. Learning of Dense Image Matching and Multi-View Stereo Reconstruction

Due to the success of deep-learning-based methods in other areas of computer vision and photogrammetry, the technological advances gained have also been transferred and applied to the task of DIM and MVS, resulting in approaches [5,8,9,43] that outperform state-of-the-art model-based approaches on numerous common benchmarks. Despite recent improvements and highly accurate results, all of these approaches are trained in a supervised manner and thus require datasets with appropriate ground truth. However, the availability and versatility of appropriate datasets is not very high, especially with respect to real-world scenarios, which still greatly hinders the practical use of deep-learning-based MVS approaches. To overcome this problem, recent approaches [10,44] attempt to train models in an unsupervised, or sometimes referred to as self-supervised, manner. But again, their practical use and ability for generalization still needs more studies [44]. These limitations are the reasons why learning-based approaches for the task of MVS are not yet practical for the considered use case, namely to reliably support emergency forces in incremental and online mapping of the operational area. In addition, we believe that there are still a number of aspects related to traditional MVS approaches that need to be addressed, such as runtime or fronto-parallel bias, which we aim to address in this work.

2. Materials and Methods

The processing pipeline of FaSS-MVS is outlined in Figure 1. Given an input bundle $(\mathcal{I}, \mathrm{P})_k \in \Omega$, consisting of k input images $\mathcal{I}$ extracted in sequential order from an image sequence, and corresponding camera poses P, our approach computes depth, normal, and confidence maps $(\mathcal{D}, \mathcal{N}, \mathcal{C})$ for a defined reference image $\mathcal{I}_{\mathrm{ref}}$, which is typically the center image of the input bundle Ω. We assume that the input is calibrated, i.e., that the images are free of lens distortion, and that the full projection matrices $\mathrm{P}_k = \mathrm{K}[\mathrm{R}\ \mathrm{t}]$ are known.

Before any processing, a Gaussian image pyramid with n pyramid levels is computed for each image of the input bundle, allowing hierarchical processing. The lowest pyramid levels ($l = 0$) contain the input images with their original image size. This results in an expansion of the input bundle Ω by $n - 1$ additional sets. In the following, a superscript is used to mark the results and processes at a particular pyramid level. The pipeline is initialized at the level with the smallest image size and executes three successive computations at each pyramid level.

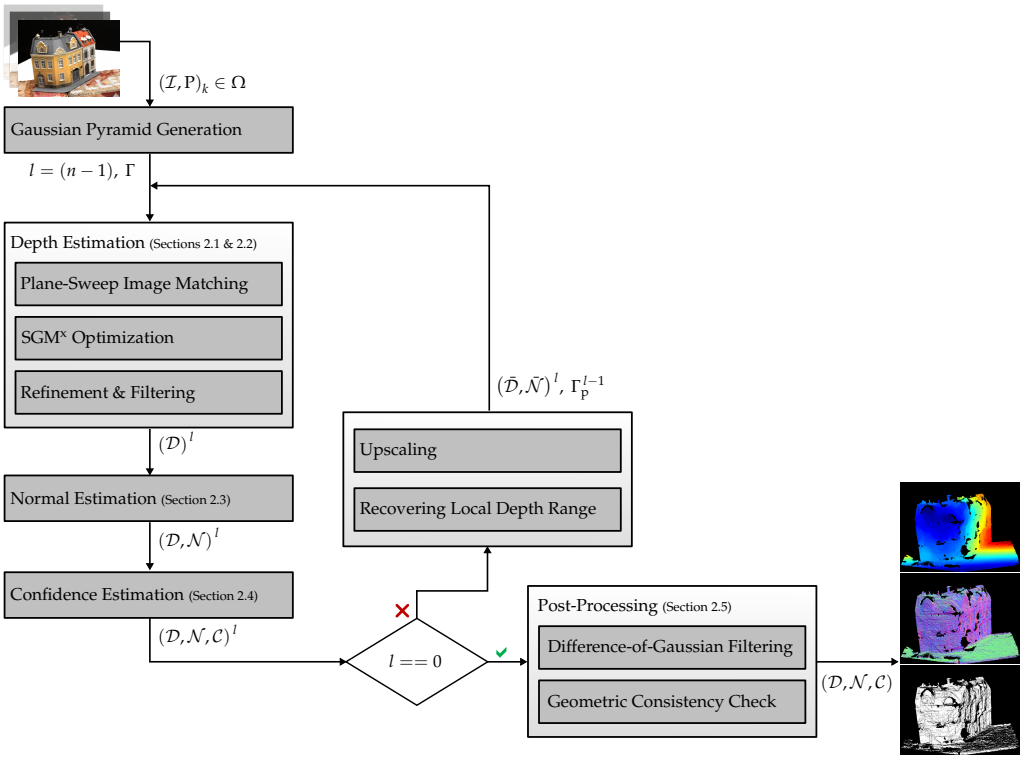

Figure 1. Overview of the processing pipeline for FaSS-MVS. Given a bundle of images and corresponding camera poses $(\mathcal{I},\mathrm{P})_k$ of an input sequence, a hierarchical MVS estimation is performed to recover a depth, normal and confidence map $(\mathcal{D},\mathcal{N},\mathcal{C})$. Adapted from [12,45].

The first part of the actual processing, the depth estimation, computes a depth map $\mathcal{D}^l$ and is in turn subdivided into a plane-sweep multi-image matching (Section 2.1), which generates depth hypotheses, and the SGM^x optimization (Section 2.2), which extracts the optimal depth from the set of hypotheses. The latter adopts the SGM algorithm [33,34] to the plane-sweep matching and extends it to account for non-fronto-parallel surface structures. A concluding depth refinement and median filter with a kernel size of 5×5 pixels is used to remove small outliers in the resulting depth map.

In the second and third computational parts of the hierarchical processing, a normal map $\mathcal{N}^l$ (Section 2.3) and a confidence map $\mathcal{C}^l$ (Section 2.4) are estimated from the previously computed depth map $\mathcal{D}^l$. The confidence map contains pixel-wise confidence values in the interval $[0,1]$ with respect to the depth estimates. These confidence scores are computed based on the surface orientation at the considered pixel.

Inherent to a hierarchical coarse-to-fine processing, the depth map $\mathcal{D}^l$ and the normal map $\mathcal{N}^l$ computed at level l are used to initialize the depth map estimation at the next pyramid level $l-1$, as long as the lowest level of the image pyramid has not yet been reached, i.e., while $l > 0$. Here, $\mathcal{D}^l$ and $\mathcal{N}^l$ are upscaled to the image size of the next pyramid level by nearest neighbor interpolation, yielding $\bar{\mathcal{D}}^l$ and $\bar{\mathcal{N}}^l$. Then, $\bar{\mathcal{D}}^l$ is first used to compute the pixel-wise sampling range Γ_p^{l-1} of the multi-image plane-sweep algorithm at the next pyramid level. Here, the Γ_p^{l-1} is computed separately for each pixel p based on the previous depth estimate $\bar{d}_\mathrm{p}^l = \bar{D}^l(\mathrm{p})$ and a predefined window with a radius of Δd around $\bar{d}_\mathrm{p}^l$:

$$\begin{aligned} \Gamma_\mathrm{p}^{l-1} &= \left[d_{\mathrm{p,min}}^{l-1}, d_{\mathrm{p,max}}^{l-1} \right], \text{with} \\ d_{\mathrm{p,min}}^{l-1} &= \bar{d}_\mathrm{p}^l - \Delta d, \\ d_{\mathrm{p,max}}^{l-1} &= \bar{d}_\mathrm{p}^l + \Delta d. \end{aligned} \tag{1}$$

In the first iteration, the sampling range is set equally for all pixels and parameterized by the minimum and maximum scene depth: $\Gamma = [d_{\min}, d_{\max}]$. The upscaled normal map $\mathcal{N}^l$ is used by one of the proposed SGM extensions to account for surface orientation within the scene. The final depth, normal, and confidence maps are the result of the processing at the lowest pyramid level. They are labeled $\mathcal{D}$, $\mathcal{N}$, and $\mathcal{C}$, respectively, and have the same image size as the input images.

In a final post-processing step (Section 2.5), we use a Difference-of-Gaussian (DoG) filter [46], as well as a geometric filtering to remove remaining outliers by masking out regions with little image texture and enforcing geometric consistency.

2.1. Real-Time Dense Multi-Image Matching with Plane-Sweep Sampling

Given a multi-camera setup c_i and an additional scene plane $\Pi = (\mathrm{n}, \delta)$ positioned in the field of view of the cameras, the image point $\mathrm{p}^{\mathrm{ref}}$ in the image of a preselected reference camera is mapped directly to the pixel p^k in any other camera image via the homography H induced by the plane Π. The scene plane Π is parameterized by its normal vector n and its distance δ from the reference camera. Together with the corresponding camera poses, this homographic projection is formulated by:

$$\begin{aligned} \mathrm{p}^k &= \mathrm{H}(\Pi, \mathrm{P}_{\mathrm{ref}}, \mathrm{P}_k) \cdot \mathrm{p}^{\mathrm{ref}}, \quad \text{with} \\ \mathrm{H}(\Pi, \mathrm{P}_{\mathrm{ref}}, \mathrm{P}_k) &= \mathrm{K}_k \cdot \frac{\mathrm{R} - \mathrm{t}\mathrm{n}^\intercal}{\delta} \cdot \mathrm{K}_{\mathrm{ref}}^{-1}. \end{aligned} \tag{2}$$

Here, $\mathrm{K}_{\mathrm{ref}}$ and K_k denote the intrinsic matrices of both cameras, and $[\mathrm{R}\ \mathrm{t}]$ denotes the relative transformation matrix of the neighboring pose P_k with respect to the reference pose $\mathrm{P}_{\mathrm{ref}}$. As shown in Figure 2, Equation (2) is interpreted geometrically by casting a viewing ray through the pixel $\mathrm{p}^{\mathrm{ref}}$ and intersecting it with the scene plane Π, yielding a scene point x_Π, which is then projected into the second camera, resulting in the image point p^k [47].

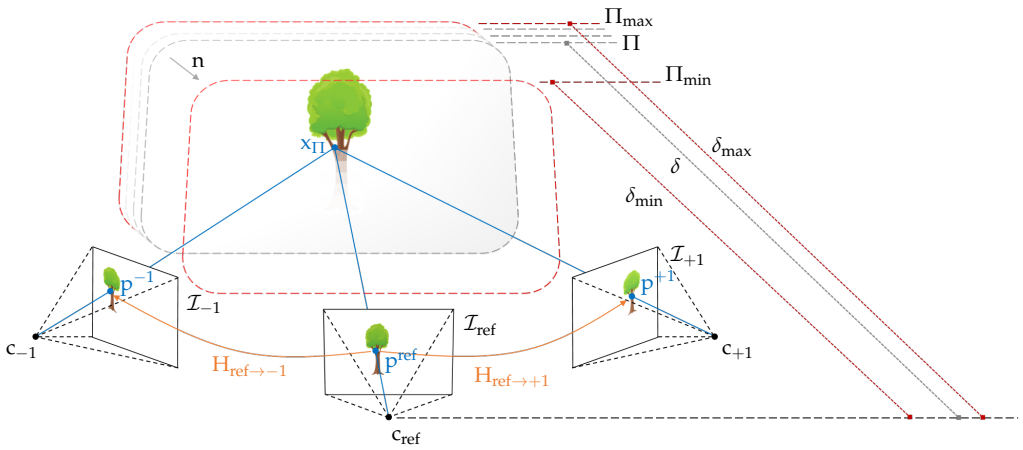

Figure 2. Illustration of the plane-sweep algorithm for multi-image matching. A scene is sampled by a plane $\Pi = (\mathrm{n}, \delta)$, where n is the normal vector of the plane and δ is the orthogonal distance of the plane from $\mathrm{c}_{\mathrm{ref}}$. The plane is swept through space along its normal vector between two bounding planes $\Pi_{\max}$ and $\Pi_{\min}$. For each distance δ of Π, the reference pixel $\mathrm{p}^{\mathrm{ref}}$ is projected by the plane-induced homography $\mathrm{H}_{\mathrm{ref}\to k}$ into an arbitrary number of viewpoints where it is matched with the corresponding pixel in $\mathcal{I}_k$.

2.1.1. The Hierarchical Plane-Sweep Algorithm for Real-Time Multi-Image Matching

Based on the relationship between two or more cameras and a scene plane, Collins [26] proposed an algorithm for true multi-image matching. This algorithm samples the scene space between two bounding planes $\Pi_{\min}$ and $\Pi_{\max}$, located at $\delta_{\min}$ and $\delta_{\max}$, by sweeping a plane along its normal vector n through space and matching the input images according to Equation (2) for each distance $\delta \in [\delta_{\min}, \delta_{\max}]$ of the plane relative to the reference camera. For each position of the plane, an arbitrary number of matching images are warped

by the plane-induced homography $\mathrm{H}^{-1}_{\mathrm{ref}\to k}$ into the view of the reference camera, where they are matched against the reference image. If the scene plane is close to a three-dimensional structure, then the corresponding image regions of the warped matching images overlap with those in the reference image, allowing the scene depth of the corresponding object to be derived from the parameterization of the corresponding plane (Figure 2). Initially referred to as the space-sweep algorithm, it has been adopted by numerous studies on multi-image matching and MVS [24,25,38], finally called the plane-sweep algorithm. This algorithm has proven to be very efficient in generating pixel-wise depth hypotheses and is therefore still widely used for the task of depth estimation [5,44,48] or novel view synthesis [49]. The presented hierarchical multi-image matching approach is based on the plane-sweep algorithm introduced by Pollefeys et al. [25] and described in Algorithm 1.

Algorithm 1: Plane-sweep multi-image matching executed at a specific pyramid level l of the proposed hierarchical processing scheme.

Data: a calibrated image bundle Ω^l at the pyramid level l, a set of planes Π with a normal vector n and varying distances δ as well as a local depth sampling range $\Gamma^l_{\mathrm{p}} = [d^l_{\mathrm{p,min}}, d^l_{\mathrm{p,max}}]$.

Result: three-dimensional cost volume $\mathcal{S}$, holding the pixel-wise matching score for each pixel $\mathrm{p}^{\mathrm{ref}} \in \mathcal{I}^l_{\mathrm{ref}}$ and plane Π.

determine bounding planes Π_{min} and Π_{max} located at δ_{min} and δ_{max}, so that the local depth range Γ^l_{p} is completely sampled (see Section 2.1.2).

foreach *pixel* $\mathrm{p}^{\mathrm{ref}} \in \mathcal{I}^l_{\mathrm{ref}}$ **and** *distance* $\delta \in [\delta_{\mathrm{min}}, \delta_{\mathrm{max}}]$ **do**

Configure scene plane $\Pi = (\mathrm{n}, \delta)$.

Determine pixels p^k in all matching images $\mathcal{I}^l_k \in \Omega^l \setminus \mathcal{I}^l_{\mathrm{ref}}$:

$$\mathrm{p}^k = \mathrm{H}\left(\Pi, \mathrm{P}^l_{\mathrm{ref}}, \mathrm{P}^l_k\right) \cdot \mathrm{p}^{\mathrm{ref}}.$$

Warp local image patches $\mathcal{P}^l_k \in \mathcal{I}^l_k$ around p^k, with the same size as the support region of the matching cost function $C(\cdot)$, into $\mathcal{I}^l_{\mathrm{ref}}$:

$$\tilde{\mathcal{P}}^l_k = \mathrm{H}\left(\Pi, \mathrm{P}^l_{\mathrm{ref}}, \mathrm{P}^l_k\right)^{-1} \cdot \mathcal{P}^l_k.$$

Compute the matching cost $s(\mathrm{p}, \Pi)$ between reference patch $\mathcal{P}^l_{\mathrm{ref}} \in \mathcal{I}^l_{\mathrm{ref}}$ and $\tilde{\mathcal{P}}^l_k$ for left and right subset of cameras separately:

$$s^{\mathrm{L}}(\mathrm{p}, \Pi) = \sum_{k<\mathrm{ref}} C\left(\mathcal{P}^l_{\mathrm{ref}}, \tilde{\mathcal{P}}^l_k\right),$$

$$s^{\mathrm{R}}(\mathrm{p}, \Pi) = \sum_{k>\mathrm{ref}} C\left(\mathcal{P}^l_{\mathrm{ref}}, \tilde{\mathcal{P}}^l_k\right).$$

Store the minimum of left and right matching cost (accounting for occlusions as described by Kang et al. [50]) into three-dimensional cost volume $\mathcal{S}$:

$$\mathcal{S}^l(\mathrm{p}, \Pi) = \min\{s^{\mathrm{L}}(\mathrm{p}, \Pi), s^{\mathrm{R}}(\mathrm{p}, \Pi)\}.$$

end

As part of the actual image matching, the Hamming distance of the census transform (CT) [51] and a negated, truncated and scaled form of the normalized cross-correlation (NCC) [38,41] are used and evaluated as cost functions $C(\cdot)$. And since the approach considers a bundle of input images with an equal number of matching images on either side of the reference image, the approach presented by Kang et al. [50] is adopted to account

for occlusions, using the minimum aggregated matching cost of the left and right subset of matching images. The resulting three-dimensional cost volume $\mathcal{S}^l$ is of size $w^l \times h^l \times |\delta^l|$, where w^l and h^l are the width and height of the reference image and $|\delta^l|$ is the number of plane positions at which the matching is performed, all with respect to the current pyramid level l. The cost volume $\mathcal{S}^l$ is implemented as a dynamic cost volume [37] for all but the top pyramid level, since the sampling range Γ_{p}^l is determined independently for each pixel p. Nevertheless, the complete set of plane distances $\delta \in [\delta_{\min}, \delta_{\max}]$, deduced from Γ, are precomputed for each pyramid level l and are the same for all pixels. This in turn allows us to precompute the homographic mappings for all planes Π.

2.1.2. Determining the Bounding Planes Corresponding to the Given Depth Range

As described before, it is assumed that two bounding planes, namely $\Pi_{\min}$ and $\Pi_{\max}$ with corresponding distances $\delta_{\min}$ and $\delta_{\max}$, between which the scene is to be sampled, are known. In the case of a fronto-parallel sampling strategy, i.e., $\mathrm{n} = (0\ 0\ -1)^\intercal$ with respect to the local camera coordinate system, the distances $\delta_{\min}$ and $\delta_{\max}$ are equal to the minimum and maximum depths, namely $d_{\min}$ and $d_{\max}$. This does not hold for non-fronto-parallel plane orientations. To find the bounding planes for slanted plane orientations, first a view-frustum is constructed, which corresponds to the reference camera for which the depth is to be estimated. This view-frustum is represented by a pyramid similar to the field of view of the camera, truncated by two fronto-parallel near and far planes located at $d_{\min}$ and $d_{\max}$. Given the four corner points of the view-frustum on the near plane $\mathrm{x}_i^{\mathrm{near}}$ and the four on the far plane $\mathrm{x}_i^{\mathrm{far}}$, the minimum and maximum distances $\delta_{\min}$ and $\delta_{\max}$ are determined as follows:

$$\begin{aligned} \delta_{\min} &= \min_i(|\mathrm{n}^\intercal \cdot \mathrm{x}_i^{\mathrm{near}}|)\ , \ \text{and} \\ \delta_{\max} &= \max_i(|\mathrm{n}^\intercal \cdot \mathrm{x}_i^{\mathrm{far}}|). \end{aligned} \tag{3}$$

To avoid an orientation flip of the images, all camera centers c_i must lie before $\Pi_{\min}$ with respect to the sweeping direction. Thus, for all cameras, $\mathrm{n}^\intercal \cdot \mathrm{c}_i + \delta_{\min} > 0$ must hold.

2.1.3. Finding the Sampling Steps by Utilizing the Cross-Ratio

As stated by Equation (2), the sampling planes Π of the plane-sweep algorithm are parameterized by two parameters, namely the normal vector n, which denotes the orientation and the sweeping direction of the plane, and the orthogonal distance δ from the optical center of the reference camera $\mathrm{c}_{\mathrm{ref}}$. The latter one determines the step size with which the scene is sampled. A simple approach would be to set the step size to sample the scene with a desired resolution, i.e., sweeping the planes at equidistant unit intervals through the scene space. However, there is no guarantee that a thorough sampling of the scene with a small step size will result in higher accuracy. If the step size is not chosen in accordance with the camera positions of the input images and the baseline between the cameras, the matching results of two or more successive plane positions may not reveal enough difference, thus introducing ambiguities between multiple plane hypotheses. Furthermore, for efficiency, it is important to vary the sampling rate in scene space with respect to the plane distance relative to the reference camera, since perspective projection requires an increasingly smaller step size as the plane moves closer to the camera.

Therefore, a common approach is to select the sampling positions of the planes according to the disparity change induced by two successive planes. The pixel-wise motion between the distorted images of two successive planes should not exceed an absolute value of 1 [25,52]. In this approach, we derive the distances of the sampling planes directly from the correspondences in image space by relying on the cross-ratio, which is invariant under perspective projection. Our approach to computing the distances δ of the sampling planes Π with respect to the reference camera was published in [11] and is illustrated in Figure 3 and summarized by Algorithm 2.

Algorithm 2: Finding plane distances δ by utilizing the cross-ratio.

Data: two cameras with full projection matrices $\mathrm{P}_{\mathrm{ref}}$ and P_k, an image point $\mathrm{p}^{\mathrm{ref}}$ inducing largest disparity when warped from $\mathcal{I}_{\mathrm{ref}}$ to $\mathcal{I}_k$, as well as two bounding planes Π_{min} and Π_{max}.

Result: list of orthogonal plane distances δ relative to $\mathrm{c}_{\mathrm{ref}}$, such that the maximum pixel displacement between the warped images of two consecutive planes is less than or equal to 1.

1 Calculate the viewing ray $\mathrm{v}_{\mathrm{p}}^{\mathrm{ref}}$, going through $\mathrm{c}_{\mathrm{ref}}$ and $\mathrm{p}^{\mathrm{ref}}$, and intersect it with Π_{min} and Π_{max}, yielding the scene points $\mathrm{x}_{\mathrm{min}}$ and $\mathrm{x}_{\mathrm{max}}$.

2 Project the optical center $\mathrm{c}_{\mathrm{ref}}$, as well as $\mathrm{x}_{\mathrm{min}}$ and $\mathrm{x}_{\mathrm{max}}$ onto the image plane of the second camera, yielding the epipole $\mathrm{e}_{\mathrm{ref}}^k$ and the two image points $\mathrm{p}_{\mathrm{min}}^k$ and $\mathrm{p}_{\mathrm{max}}^k$, all lying on the epipolar line $\mathrm{l}_{\mathrm{p}}^k$.

3 Determine the unit vector $\mathrm{k} = \frac{\mathrm{p}_{\mathrm{min}}^k - \mathrm{p}_{\mathrm{max}}^k}{||\mathrm{p}_{\mathrm{min}}^k - \mathrm{p}_{\mathrm{max}}^k||}$, being the normalized direction of $\mathrm{l}_{\mathrm{p}}^k$ and pointing from $\mathrm{p}_{\mathrm{max}}^k$ to $\mathrm{p}_{\mathrm{min}}^k$.

4 **for** $\mathrm{p}_i^k \leftarrow \mathrm{p}_{\mathrm{max}}^k$ **to** $\mathrm{p}_{\mathrm{min}}^k$ **by** $\mathrm{p}_{i+1}^k = \mathrm{p}_i^k + \mathrm{k}$ **do**

5 Given the viewing rays $\mathrm{v}_{\mathrm{e}_{\mathrm{ref}}}^k$, $\mathrm{v}_{\mathrm{p}_{\mathrm{min}}}^k$, $\mathrm{v}_{\mathrm{p}_i}^k$ and $\mathrm{v}_{\mathrm{p}_{\mathrm{max}}}^k$ going through the optical center of c_k and $\mathrm{e}_{\mathrm{ref}}^k$, $\mathrm{p}_{\mathrm{min}}^k$, p_i^k and $\mathrm{p}_{\mathrm{max}}^k$ respectively, apply the cross-ratio to compute $\mathrm{x}_i \in \mathrm{v}_{\mathrm{p}}^{\mathrm{ref}}$ according to:

$$Q(\mathrm{v}_{\mathrm{e}_{\mathrm{ref}}}^k, \mathrm{v}_{\mathrm{p}_{\mathrm{min}}}^k, \mathrm{v}_{\mathrm{p}_i}^k, \mathrm{v}_{\mathrm{p}_{\mathrm{max}}}^k) = \frac{\sin(\alpha(\mathrm{v}_{\mathrm{e}_{\mathrm{ref}}}^k, \mathrm{v}_{\mathrm{p}_i}^k)) \cdot \sin(\alpha(\mathrm{v}_{\mathrm{p}_{\mathrm{min}}}^k, \mathrm{v}_{\mathrm{p}_{\mathrm{max}}}^k))}{\sin(\alpha(\mathrm{v}_{\mathrm{e}_{\mathrm{ref}}}^k, \mathrm{v}_{\mathrm{p}_{\mathrm{max}}}^k)) \cdot \sin(\alpha(\mathrm{v}_{\mathrm{p}_{\mathrm{min}}}^k, \mathrm{v}_{\mathrm{p}_i}^k))} = \frac{\Delta(\mathrm{c}_{\mathrm{ref}}, \mathrm{x}_i) \cdot \Delta(\mathrm{x}_{\mathrm{min}}, \mathrm{x}_{\mathrm{max}})}{\Delta(\mathrm{c}_{\mathrm{ref}}, \mathrm{x}_{\mathrm{max}}) \cdot \Delta(\mathrm{x}_{\mathrm{min}}, \mathrm{x}_i)}.$$

6 Since $Q(\mathrm{c}_{\mathrm{ref}}, \mathrm{x}_{\mathrm{min}}, \mathrm{x}_i, \mathrm{x}_{\mathrm{max}}) = Q(\mathrm{c}_{\mathrm{ref}}, \delta_{\mathrm{min}}, \delta, \delta_{\mathrm{max}})$, derive δ relative to $\mathrm{c}_{\mathrm{ref}}$ according to:

$$\frac{\delta \cdot (\delta_{\mathrm{max}} - \delta_{\mathrm{min}})}{\delta_{\mathrm{max}} \cdot (\delta - \delta_{\mathrm{min}})} = \frac{\sin(\alpha(\mathrm{v}_{\mathrm{e}_{\mathrm{ref}}}^k, \mathrm{v}_{\mathrm{p}_i}^k)) \cdot \sin(\alpha(\mathrm{v}_{\mathrm{p}_{\mathrm{min}}}^k, \mathrm{v}_{\mathrm{p}_{\mathrm{max}}}^k))}{\sin(\alpha(\mathrm{v}_{\mathrm{e}_{\mathrm{ref}}}^k, \mathrm{v}_{\mathrm{p}_{\mathrm{max}}}^k)) \cdot \sin(\alpha(\mathrm{v}_{\mathrm{p}_{\mathrm{min}}}^k, \mathrm{v}_{\mathrm{p}_i}^k))}.$$

7 **end**

For c_k, we choose the camera that will induce the largest image offset, thus giving an upper bound on the disparity range. As noted by Pollefeys et al. [25], this is typically the camera farthest away from the reference camera. Similarly, we choose $\mathrm{p}^{\mathrm{ref}}$ as the pixel that induces the largest disparity when warped from $\mathcal{I}_{\mathrm{ref}}$ to $\mathcal{I}_k$ via $\mathrm{H}(\Pi_{\mathrm{min}}, \mathrm{P}_{\mathrm{ref}}, \mathrm{P}_k)$, typically one of the four corners. Furthermore, to account for all possible setups of $\mathrm{c}_{\mathrm{ref}}$ and c_k, it is important to use $Q(\mathrm{v}_{\mathrm{e}_{\mathrm{ref}}}^k, \mathrm{v}_{\mathrm{p}_{\mathrm{min}}}^k, \mathrm{v}_{\mathrm{p}_i}^k, \mathrm{v}_{\mathrm{p}_{\mathrm{max}}}^k)$ in Algorithm 2, since $\mathrm{e}_{\mathrm{ref}}^k$ would flip to the side of $\mathrm{p}_{\mathrm{max}}^k$ if the focal plane of the reference camera is behind c_k. This approach is computationally efficient and is not restricted to a fronto-parallel orientation of the sampling planes, as long as the optical axis of the reference camera intersects the planes and the sweeping vector has a component that is parallel to the optical axis.

2.2. Depth Map Computation with Surface-Aware Semi-Global Matching

The hierarchical plane-sweep algorithm for multi-image matching produces a three-dimensional cost volume $\mathcal{S}^l(\mathrm{p}, \Pi)$ at each pyramid level, containing pixel-wise matching costs, given plane Π located at a distance δ orthogonal to the location $\mathrm{c}_{\mathrm{ref}}$ of the reference camera. In the second stage of the depth estimation within FaSS-MVS, the cost volume is regularized by a semi-global optimization scheme, yielding a dense depth map $\mathcal{D}^l$. It is based on the Semi-Global Matching (SGM) algorithm proposed by Hirschmüller [33,34] for the task of disparity estimation as part of the stereo normal case. It uses dynamic

programming to efficiently minimize a two-dimensional Markov Random Field (MRF) energy function by aggregating the matching costs within the cost volume along numerous concentric one-dimensional paths.

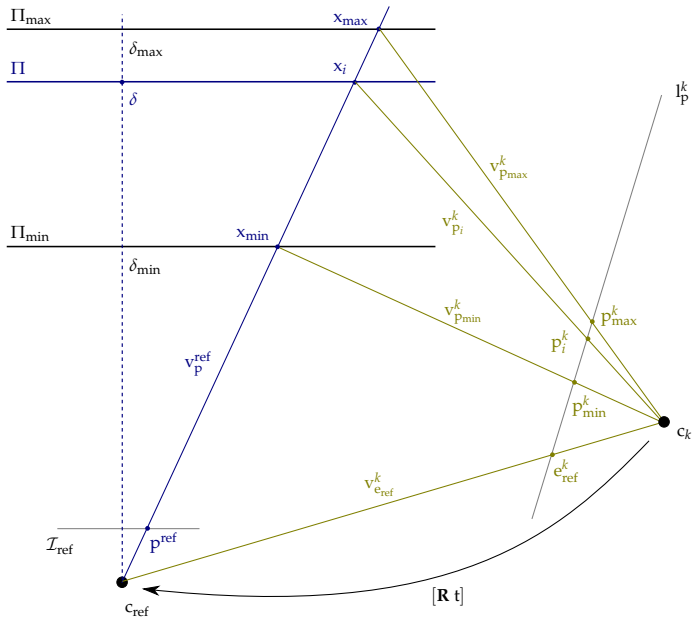

Figure 3. Illustration of determining the orthogonal distance parameter of the sampling planes of the plane-sweep multi-image matching by using the cross-ratio and epipolar geometry. Here, c_{ref} and c_k represent the positions of the optical centers of the two cameras. Adapted from [11].

Building on the original SGM approach, we propose three different optimization schemes (SGM^x). Apart from a straightforward adaptation of the matching cost aggregation to plane-sweep sampling, we also adopt the approach of Scharstein et al. [41] to also favor slanted surfaces by taking into account surface information available in the form of surface normals. Furthermore, we investigate a third extension that penalizes deviations from the gradient of the minimum-cost path within the SGM optimization scheme. The subsequent extraction of the depth map $\mathcal{D}$ is performed analogously to the extraction of the disparity map within the SGM algorithm, where disparity is replaced by depth. If a fronto-parallel plane orientation is considered during the plane-sweep, the depth can be extracted directly from the plane parameterization. For non-fronto-parallel orientations, however, $\mathcal{D}$ is computed by a pixel-wise intersection of the viewing rays with the corresponding WTA solutions.

2.2.1. Resolving Plane Hypotheses with Semi-Global Matching

Since the plane-sweep algorithm does not compute hypotheses on disparities, but rather pixel-wise plane distances relative to the reference camera and thus depth, the first SGM extension we propose is a straightforward adaption of the standard SGM algorithm to a multi-view plane-sweep sampling. In this, the formulation of the SGM path aggregation is modified to

$$L_r(p,\Pi) = \mathcal{S}(p,\Pi) + \min_{\delta'}\Big(L_r(p-r,\Pi') + V_\Pi(\Pi,\Pi')\Big), \tag{4}$$

where Π is the sampling plane at distance δ. The smoothness term V_Π now penalizes the selection of different planes between adjacent pixels along the path L_r, instead of disparities. It is formulated as:

$$V_\Pi(\Pi, \Pi') = \begin{cases} 0 & \text{, if } I(\Pi) = I(\Pi') \\ \varphi_1 & \text{, if } |I(\Pi) - I(\Pi')| = 1 \\ \varphi_2 & \text{, if } |I(\Pi) - I(\Pi')| > 1, \end{cases} \tag{5}$$

where $I(\cdot)$ is a function that returns the index of Π within the set of sampling planes (see Figure 4, Column 2). We denote this extension as *plane-wise* SGM (SGM$^\Pi$). In our previous publication [12], we have referred to this extension as *fronto-parallel* SGM (SGM$^\mathrm{fp}$), since we have only considered a fronto-parallel sweeping direction so far. However, the extension is not restricted to a fronto-parallel plane orientation in the plane-sweep sampling and will also be evaluated with slanted planes in the scope of this work. Given a pixel-wise WTA plane parameterization, the corresponding depth is extracted by intersecting the viewing ray through pixel $\mathrm{p} = (p_x\ p_y)^\intercal$ with the corresponding plane:

$$d_\mathrm{p} = \frac{-\delta}{\mathrm{n}^\intercal \cdot \mathrm{K}^{-1} \cdot (p_x\ p_y\ 1)^\intercal}. \tag{6}$$

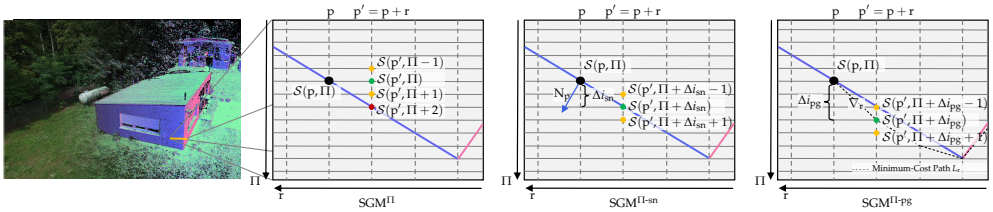

Figure 4. Illustration of the different path aggregation strategies along one path direction r within the three presented SGM$^\mathrm{x}$ optimization schemes. Column 1: Reference image and normal map of a building. Illustrated area is marked with yellow line. Column 2: SGM$^\Pi$ path aggregation. The blue and pink lines represent the blue and pink surface orientations on the building facade. When aggregating the path costs for pixel p at plane Π, SGM$^\Pi$ will include the previous costs at the same plane position (green) without additional penalty. The previous path costs at $\Pi \pm 1$ (yellow) will be penalized with φ_1. The previous path costs located at $\Pi + 2$ (red), which is actually located on the corresponding surface, will be penalized with the highest penalty φ_2. Column 3: SGM$^\mathrm{\Pi\text{-}sn}$ uses the normal vector $\mathrm{n_p}$, which encodes the surface orientation at pixel p, and computes a discrete index jump Δi_sn, which ideally adjusts the zero-cost transition so that the previous path costs at Π+2 are not penalized. Column 4: Similar to SGM$^\mathrm{\Pi\text{-}sn}$, SGM$^\mathrm{\Pi\text{-}pg}$ adjusts the zero-cost transition. However, the discrete index jump Δi_pg is derived from the running gradient $\nabla \mathrm{r}$ of the minimum-cost path. Adapted from [12].

2.2.2. Incorporating Surface Normals to Adjust the Zero-Cost Transition

The smoothness term of the initial SGM algorithm is formulated with discrete disparity differences, penalizing discrete disparity jumps between neighboring pixels. In its optimization scheme, it does not consider subpixel disparities and thus favors fronto-parallel surface structures, leading to staircase artifacts if no post-processing is applied [41]. The same applies to our first extension, SGM$^\Pi$. Although plane-sweep sampling also supports non-fronto-parallel plane orientations, the smoothness term of SGM$^\Pi$ (see Equation (5)) does not, and strongly penalizes index jumps in the sampling planes greater than 1. While this is desired if the plane orientation coincides with the surface orientation, it will still lead to staircasing artifacts if the surface and plane orientations do not align. To overcome the favoring of fronto-parallel structures and to adjust the smoothness term of SGM to surfaces that are slanted with respect to the sampling direction, Scharstein et al. [41] suggest adding an offset to the smoothness term. This offset can be extracted from additional information about the surface orientation, e.g., surface normals, which will make the zero-cost transi-

tion coincide with the surface orientation. We adopt this approach as part of our second extension, and thus call it *surface normal* SGM (SGM$^{\Pi\text{-sn}}$).

In our hierarchical approach, we extract the normal vectors from the normal map $\mathcal{N}^{l+1}$, which was estimated in the previous level of the pyramid (see Figure 1). The pixel-wise normal vectors $\mathrm{n_x} = \mathcal{N}^{l+1}(\mathrm{p})$ indicate the surface orientation at the scene point x, which is computed by intersecting the viewing ray through p with the plane Π. From this, the discrete index jump Δi_{sn} through the set of sampling planes can be calculated, which is caused by the tangent plane to $\mathrm{n_x}$. Since the plane-sweep sampling is not restricted to fronto-parallel plane orientations, the index jump Δi_{sn} must be calculated based on the difference between the tangent plane at x_Π and the orientation of the sampling planes in the direction r of the currently considered aggregation path. With Δi_{sn}, the smoothness term used by our extension SGM$^{\Pi\text{-sn}}$ is adjusted according to

$$V_{\Pi\text{-sn}}(\Pi, \Pi') = \begin{cases} 0 & , \text{ if } I(\Pi) + \Delta i_{\mathrm{sn}} = I(\Pi') \\ \varphi_1 & , \text{ if } |I(\Pi) + \Delta i_{\mathrm{sn}} - I(\Pi')| = 1 \\ \varphi_2 & , \text{ if } |I(\Pi) + \Delta i_{\mathrm{sn}} - I(\Pi')| > 1, \end{cases} \tag{7}$$

This allows the zero-cost transition of the SGM path aggregation to be aligned with the surface orientation of the scene (see Figure 4, Column 3). The pixel-wise discrete index jumps can be computed once for each pixel p and each path direction r, as also noted by Scharstein et al. [41], with little computational overhead.

2.2.3. Penalizing Deviations from the Gradient of the Minimum-Cost Path

Instead of relying on additional information, e.g., normal vectors, the third of our proposed extensions computes the running gradient $\nabla \mathrm{r}$ of the minimum-cost path in scene space in order to adjust the zero-cost transition in the aggregation of path costs. Hence, it is denoted as *path gradient* SGM (SGM$^{\Pi\text{-pg}}$).

The gradient vector $\nabla \mathrm{r} = \mathrm{x} - \mathrm{x}'$ in scene space is computed dynamically while traversing the path r. Again, x is the scene point found by intersecting the viewing ray through p with Π, while x' is the scene point parameterized by p' and the plane $\hat{\Pi}'$. Here, $\mathrm{p}' = \mathrm{p} + \mathrm{r}$ represents the predecessor of p along the path r and $\hat{\Pi}'$ denotes the plane at distance $\hat{\delta} = \arg\min_\delta L_{\mathrm{r}}(\mathrm{p}', \Pi)$ associated with the previous minimum costs.

From this, a discrete index jump Δi_{pg} is computed, which is again used to account for possibly slanted surfaces in scene space by adjusting the zero-cost transition of the smoothness term according to

$$V_{\Pi\text{-pg}}(\Pi, \Pi') = \begin{cases} 0 & , \text{ if } I(\Pi) + \Delta i_{\mathrm{pg}} = I(\Pi') \\ \varphi_1 & , \text{ if } \left|I(\Pi) + \Delta i_{\mathrm{pg}} - I(\Pi')\right| = 1 \\ \varphi_2 & , \text{ if } \left|I(\Pi) + \Delta i_{\mathrm{pg}} - I(\Pi')\right| > 1, \end{cases} \tag{8}$$

This implicitly penalizes deviations from the running gradient between two scene points corresponding to two consecutive pixels on the aggregation path r (see Figure 4, Column 4).

2.3. Extraction of Surface Normals from Depth Maps

From the estimated depth map $\mathcal{D}$, our approach computes a normal map $\mathcal{N}$, which holds the local surface orientations in the form of three-dimensional normal vectors. The surface normal vectors $\mathrm{n_p} = \mathrm{h_p} \times \mathrm{v_p}$ are computed using the cross-product, where $\mathrm{h_p}$ is the difference vector between the reprojected scene points of two neighboring pixels to p in horizontal direction and $\mathrm{v_p}$ is the difference vector in vertical direction.

Using only the cross-product to compute the surface orientation does not include any local smoothness assumption. Therefore, we use an appearance-based weighted Gaussian

smoothing in a local two-dimensional window $\mathcal{W}_p$ around p, which adjusts the smoothing strength depending on the intensity difference between $q \in \mathcal{W}_p$ and p:

$$\mathcal{N}(p) = \frac{\bar{n}_p}{|\bar{n}_p|}, \tag{9}$$

with

$$\bar{n}_p = n_p + \sum_{q \in \mathcal{W}_p} n_q \cdot \frac{1}{\sqrt{2\pi\sigma^2}} \cdot \exp\left(-\frac{(q-p)^2}{2\sigma^2} - \frac{\Delta\mathcal{I}_{pq}}{\beta} \right). \tag{10}$$

In this, β is set to 10, while σ is fixed to the radius of $\mathcal{W}_p$.

2.4. Estimation of Confidence Measures Based on Surface Orientation

Besides the depth map $\mathcal{D}$ and the normal map $\mathcal{N}$, the presented approach also computes confidence measures for the depth estimates in the range of $[0,\ 1]$ and stores them in a confidence map $\mathcal{C}$. Such confidence measures allow subsequent reasoning about the certainty of the corresponding estimates and thus improve further processing. Thus, confidence maps are useful by-products for subsequent steps such as depth map fusion or scene interpretation. Furthermore, they allow us to gain more insight into the effects of different configurations of the presented approach.

The computation of the pixel-wise confidence measures is based on the geometric properties of the estimated depth map and is derived from the normal vectors stored inside the normal map $\mathcal{N}$ and the plane orientations of the plane-sweep sampling. In particular, the geometric confidence measure is based on the enclosed angles between the local surface orientation stored inside the normal map $n_p = \mathcal{N}(p)$, the orientation of the sampling plane n_Π, and the inverted viewing direction v. This is taken from the geometric weighting factor proposed by Kolev et al. [53]. They argue that a depth estimate is more accurate when the surface orientation of the observed geometry is fronto-parallel to the image plane of the camera, and less accurate when the camera observes slanted surfaces. This correlation is modeled by the scalar product between the surface orientation and the inverted viewing direction. Since image warping, as part of image matching, can be aligned with the surface orientation by adjusting the normal vector of the plane-sweep algorithm, the plane orientation is also taken into account. Thus, the geometry-based weighting factor is calculated as follows:

$$\mathcal{C}(p) = \begin{cases} \frac{\langle n_p, n_\Pi \rangle \langle n_\Pi, v \rangle - \cos\rho}{1 - \cos\rho} & \text{, if } \{\sphericalangle(n_p, n_\Pi) \wedge \sphericalangle(n_\Pi, v)\} \leq \rho \\ 0 & \text{, otherwise .} \end{cases} \tag{11}$$

All of the above vectors are assumed to be normalized and given with respect to the local coordinate system of the camera; thus, $v = (0\ 0\ -1)^\intercal$. As in the work of Kolev et al. [53], a critical angle $\rho = 60°$ is used to mark the measurements, for which the enclosed angles exceed this threshold, as unreliable. The additional consideration of n_Π in Equation (11) implicitly models the indirect matching of the input images via the plane-induced homography.

2.5. Post-Processing and Depth Map Filtering

In a final post-processing step, remaining outliers are removed from the depth, normal and confidence maps by applying a Difference-of-Gaussian (DoG) filtering (Section 2.5.1) and an outlier removal based on geometric consistency (Section 2.5.2).

2.5.1. Difference-of-Gaussian Filtering

As proposed by Wenzel [46], the DoG filter allows us to remove estimates of $\mathcal{D}$, $\mathcal{N}$, and $\mathcal{C}$ by masking pixels in image regions that provide little textural information (e.g., blurred or overexposed areas). It is assumed that image matching in such regions is ambiguous and leads to less accurate results. The DoG filter is used to detect weakly textured areas

within the reference image $\mathcal{I}_{\text{ref}}$ and to build a binary image mask that is used to remove the estimates from the corresponding maps. Algorithm 3 provides an overview of the implementation of the DoG filter used, which is similar to the one proposed in [46].

Algorithm 3: The Difference-of-Gaussian filter to invalidate all image pixels belonging to weakly textured areas.

Data: unfiltered depth, normal and confidence maps ($\mathcal{D}$, $\mathcal{N}$ and $\mathcal{C}$) as well as corresponding reference image $\mathcal{I}_{\text{ref}}$.
Result: filtered $\mathcal{D}$, $\mathcal{N}$ and $\mathcal{C}$, in which all estimates corresponding to weakly-textured areas in $\mathcal{I}_{\text{ref}}$ are removed.

1 Use a Gaussian filter with a kernel of 7×7 pixels to smooth the reference frame $\mathcal{I}_{\text{ref}}$, yielding $\mathcal{I}_{\text{ref}}^{\text{smooth}}$.
2 Compute the DoG image depicting local image gradients, according to: $\mathcal{I}_{\text{ref}}^{\text{DoG}} = \mathcal{I}_{\text{ref}} - \mathcal{I}_{\text{ref}}^{\text{smooth}}$.
3 Apply a binary threshold to compute the DoG mask $\mathcal{M}^{\text{DoG}}$, marking all image areas in which the intensity change is greater than 0.5.
4 Remove activation areas smaller than 7 pixels in $\mathcal{M}^{\text{DoG}}$ by applying a speckle filter.
5 Dilate $\mathcal{M}^{\text{DoG}}$ with a kernel size of 3×3 pixels to fill small holes in activation areas.
6 Remove deactivation areas smaller than 21 pixels by applying a speckle filter to the inverted DoG mask $\mathcal{M}^{\text{DoG-inv}} = 1 - \mathcal{M}^{\text{DoG}}$.
7 Invalidate pixels in $\mathcal{D}$, $\mathcal{N}$ and $\mathcal{C}$ for which $\mathcal{M}^{\text{DoG}} = 1$.

2.5.2. Geometric Consistency Based on Mutual Reprojection Error

If multiple depth maps $\mathcal{D}_k$ with corresponding projection matrices P_k are available, e.g., when performing reconstruction by MVS or when considering a sequence of images as input and a temporal consistency is to be established, a geometric consistency check can be performed by relying on the mutual reprojection error. As formulated by Schönberger et al. [4], each pixel p^{ref} of a selected reference depth map $\mathcal{D}_{\text{ref}}$ with a depth estimate $d_{\mathrm{p}}^{\text{ref}}$ is projected into the view of another depth map $\mathcal{D}_k$ by H_{p}, according to $d_{\mathrm{p}}^{\text{ref}}$ and the corresponding projection matrices P_{ref} and P_k, resulting in the image point p^k. Given p^k and the corresponding depth d_{p}^k from $\mathcal{D}_k$, the image point p^k is projected back into the view of $\mathcal{D}_{\text{ref}}$ by $\mathrm{H}_{\mathrm{p}}^k$, resulting in $\tilde{\mathrm{p}}^{\text{ref}}$. Finally, if the Euclidean distance between p^{ref} and $\tilde{\mathrm{p}}^{\text{ref}}$, i.e., the reprojection error $\epsilon_{\mathrm{r}}^k(\mathrm{p})$, exceeds a given threshold η_{r}, the estimate at p^{ref} is invalidated.

We adopt this approach to perform a final geometry-based filtering between a set of depth maps within a sliding window. This is not part of the actual hierarchical processing pipeline, but rather a separate post-processing step, since it requires the results of other image bundles of the input sequence. If possible, the middle depth map of the sliding window Ψ is chosen as the reference view on which the filtering is performed. At the beginning or end of the sequence, where the sliding window would exceed the boundaries, the window is shifted to either side of the reference view, so that it is always within the boundaries of the sequence and no depth map is filtered multiple times. Besides the threshold of the reprojection error ϵ_{r}^k, another criterion is introduced to evaluate the geometric consistency, namely the number of neighboring views for which the reprojection error is within the threshold, i.e., the number of hits: $\epsilon_{\mathrm{h}}(\mathrm{p}) = \sum_k [\epsilon_{\mathrm{r}}^k(\mathrm{p}) < \eta_{\mathrm{r}}]$, where $[\cdot]$ is the Iverson bracket. Algorithm 4 gives an overview of geometric consistency and the corresponding filtering of the depth, normal, and confidence maps. In this work, the sliding window size is empirically set to $|\Psi| = 5$, the reprojection error threshold to $\eta_{\mathrm{r}} = 10$, and the consistency threshold to $\eta_{\mathrm{h}} = 3$.

Algorithm 4: The geometric consistency filter for multiple depth maps.

Data: depth, normal and confidence maps ($\mathcal{D}_k$, $\mathcal{N}_k$ and $\mathcal{C}_k$) within a sliding window Ψ of the input sequence as well as corresponding projection matrices P_k.

Result: filtered $\mathcal{D}_{\mathrm{ref}}$, $\mathcal{N}_{\mathrm{ref}}$ and $\mathcal{C}_{\mathrm{ref}}$ of reference view, in which all estimates that are not geometrically consistent are removed.

Select $\mathcal{D}_{\mathrm{ref}}$, $\mathcal{N}_{\mathrm{ref}}$ and $\mathcal{C}_{\mathrm{ref}}$ corresponding to the center-most view within the sliding window Ψ.

foreach *pixel* $\mathrm{p}^{\mathrm{ref}} \in \mathcal{D}_{\mathrm{ref}}$ **and** *neighboring view* $k \in \Psi$ **do**

Calculate number of hits for which reprojection error is below threshold:

$$\epsilon_{\mathrm{h}}(\mathrm{p}) = \sum_k [\epsilon_{\mathrm{r}}^k(\mathrm{p}) < \eta_{\mathrm{r}}], \text{ with } \epsilon_{\mathrm{r}}^k(\mathrm{p}) = \left| \mathrm{p} - \mathrm{H}_{\mathrm{p}}^k \cdot \mathrm{H}_{\mathrm{p}} \cdot \mathrm{p} \right|.$$

If $\epsilon_{\mathrm{h}} < \eta_{\mathrm{h}}$, invalidate pixel p in $\mathcal{D}_{\mathrm{ref}}$, $\mathcal{N}_{\mathrm{ref}}$ and $\mathcal{C}_{\mathrm{ref}}$, by setting it to 0.

end

2.6. Evaluation Datasets

The presented approach is quantitatively evaluated on two public datasets, namely the DTU Robot MVS dataset [54,55] and the 3DOMcity Benchmark dataset [56], which also provide appropriate ground truth. These two datasets are based on images of scale modeled buildings and an urban landscape from which an accurate ground truth is acquired. For a qualitative evaluation and discussion of the applicability of the presented approach for online dense image matching and 3D reconstruction, two privately captured datasets of real-world scenes are used, hereafter referred to as the TMB and FB datasets. In the following, the characteristics of these datasets are briefly introduced. In particular, we discuss which parts of the datasets are used and what kind of ground truth is available for the evaluation. The key characteristics are summarized in Table 1.

2.6.1. DTU Robot MVS Dataset

The DTU Robot MVS dataset (Figure 5, Column 1) consists of 124 different tabletop scenes, of which we used 21 scans of different building models, as these scenes are closest to the target use case. For each scene, there are input images taken from 49 locations distributed in an orbital pattern around the tabletop scene. In addition, a ground truth is provided for each scene in the form of a detailed point cloud captured by a structured-light scanner. For the quantitative evaluation, the already undistorted images with a resolution of 1600×1200 pixels, together with the provided intrinsic and extrinsic camera projection matrices, were used as input data for the approach. Since the focus of this work is on the estimation of depth and normal maps only, corresponding ground truth data are rendered from the detailed point cloud data.

2.6.2. 3DOMcity Benchmark Dataset

Depending on the aircraft and its environment, an orbital trajectory as shown by the DTU benchmark data may not always be feasible or desirable. The data provided as part of the 3DOMcity Benchmark [56] (Figure 5, Column 2), however, simulate a grid flight where the aircraft flies linearly over the area of interest with a fixed camera orientation relative to the sensor carrier. In this case, images of a scaled urban scene consisting of buildings of various sizes and shapes, as well as roads and vegetation, are captured with a DSLR camera that is moved in parallel lines over the model along a rigid bar. To use the data of the 3DOMcity Benchmark for a quantitative evaluation of the performance of the presented approach, the already undistorted images are first downscaled to a size of 1798×1200 pixels, preserving the initial aspect ratio, before the intrinsic camera parameters are estimated with the help of COLMAP [3]. The extrinsic camera data are extracted from

the reference provided as part of the benchmark. To evaluate the accuracy of the depth maps, the reference point cloud computed by the semi-global dense image matching (DIM) algorithm is rendered from the viewpoints of the input images, as in the case of the DTU Benchmark dataset.

2.6.3. Real-World Use-Case-Specific TMB and FB Datasets

The strength and purpose of the DTU and 3DOMcity benchmark datasets is their small size and the associated ability to record or compute accurate reference data, which in turn facilitates a quantitative evaluation of the accuracy of the evaluated algorithms. However, these datasets were recorded in controlled environments and do not fully address the use case targeted by the presented approach. In order to perform a qualitative evaluation on real data, appropriate test data were collected in private datasets.

This dataset is two-fold. The first part, the TMB dataset (Figure 5, Column 3), consists of four sequences captured by a DJI Phantom (DJI, Shenzhen, China) 3 Professional flying around a freestanding house and containers at altitudes between 8 m and 15 m. The second part, referred to as the Fire Brigade (FB) dataset (Figure 5, Column 4), was captured during a fire training exercise around a large industrial building. The data were collected using a DJI Matrice 200 with a Zenmuse XT2 sensor flying linearly over the area where the exercise was conducted. For all sequences, images were captured at a frame rate of approximately 1 FPS and downsampled to an image size of 1920 × 1080 pixels. Images that are not suitable as input for the presented approach, e.g., by providing too little offset, are discarded.

Figure 5. Overview of the datasets used for performance evaluation of FaSS-MVS. Column 1: Two building models from the DTU Robot MVS dataset. Column 2: Example images in oblique and nadir view from the 3DOMcity Benchmark dataset. Column 3: Excerpt of the privately acquired TMB dataset. Column 4: Use-case-specific dataset acquired during an exercise of the local fire brigade.

Table 1. Summary of the key characteristics of the four evaluation datasets. *(i): intrinsic calibration, (e): extrinsic calibration.*

Dataset	# Images	Image Size	Calibration	Reference	Scene	Flight Pattern
DTU	1029	1600 × 1200	(i) pre-calibration, (e) pre-calibration	structured-light sensor	scale-modeled buildings	orbital
3DOMcity	245	1798 × 1200	(i) COLMAP, (e) pre-calibration	semi-global offline DIM	scale-modeled urban area	linear
TMB	2013	1920 × 1080	(i) COLMAP, (e) COLMAP	COLMAP (geometric depth)	rural area	orbital
FB	202	1920 × 1080	(i) COLMAP, (e) COLMAP	COLMAP (geometric depth)	industrial area	linear

2.7. Error Measures

To directly quantify error between the estimated depth map $\mathcal{D}_{\text{est}}$ and the corresponding ground truth $\mathcal{D}_{\text{gt}}$ during the experiments, absolute and relative L1 measures are used:

$$\text{L1-abs}(\mathcal{D}_{\text{est}}, \mathcal{D}_{\text{gt}}) = \frac{1}{|\mathcal{V}|} \sum_{\text{p} \in \mathcal{V}} |\mathcal{D}_{\text{est}}(\text{p}) - \mathcal{D}_{\text{gt}}(\text{p})| \,, \text{ and} \tag{12}$$

$$\text{L1-rel}(\mathcal{D}_{\text{est}}, \mathcal{D}_{\text{gt}}) = \frac{1}{|\mathcal{V}|} \sum_{\text{p} \in \mathcal{V}} \frac{|\mathcal{D}_{\text{est}}(\text{p}) - \mathcal{D}_{\text{gt}}(\text{p})|}{\mathcal{D}_{\text{gt}}(\text{p})} \,. \tag{13}$$

Here, $\mathcal{V}$ denotes the set of pixels for which both $\mathcal{D}_{\text{est}}$ and $\mathcal{D}_{\text{gt}}$ have valid depth measurements. While L1-abs provides an absolute and thus interpretable insight into the mean error of the estimated depth map, it is rather unsuitable for comparing results across multiple datasets with different depth ranges. This is because the error of depth measurements typically increases with depth, resulting in a higher absolute error for datasets with greater scene depth. To compensate for this effect, the relative L1-rel measure normalizes the absolute difference by the depth stored at the corresponding ground truth pixel. This reduces the effect that erroneous pixels in distant areas of the scene have on the error score, while increasing the weight of pixels close to the camera.

The two error measures introduced above provide a simple strategy for evaluating the error of the estimates. However, they do not allow one to reason about the completeness and density of the estimated depth map. Since the focus of this work is on dense MVS, it is also of great interest to know how many pixels of $\mathcal{D}_{\text{est}}$ are actually filled with correct estimates. Two closely related error measures are used for this, namely the accuracy (Acc_θ) and the completeness (Cpl_θ). These scores are typically used to evaluate classification tasks, but in recent years, they have also been used to evaluate range measurements [57,58]. On the one hand, the accuracy Acc_θ indicates the number of pixels within the estimated depth map $\mathcal{D}_{\text{est}}$ for which the corresponding depth value is within a given threshold θ to the ground truth:

$$\text{Acc}_\theta(\mathcal{D}_{\text{est}}, \mathcal{D}_{\text{gt}}) = \frac{1}{|\mathcal{E}|} \sum_{\text{p} \in \mathcal{V}} \left[\max\left(\frac{\mathcal{D}_{\text{est}}(\text{p})}{\mathcal{D}_{\text{gt}}(\text{p})}, \frac{\mathcal{D}_{\text{gt}}(\text{p})}{\mathcal{D}_{\text{est}}(\text{p})} \right) < \theta \right] . \tag{14}$$

The completeness Cpl_θ, on the other hand, indicates the fraction of the ground truth pixels for which estimates exist that are within the given distance threshold to the reference:

$$\text{Cpl}_\theta(\mathcal{D}_{\text{est}}, \mathcal{D}_{\text{gt}}) = \frac{1}{|\mathcal{G}|} \sum_{\text{p} \in \mathcal{V}} \left[\max\left(\frac{\mathcal{D}_{\text{est}}(\text{p})}{\mathcal{D}_{\text{gt}}(\text{p})}, \frac{\mathcal{D}_{\text{gt}}(\text{p})}{\mathcal{D}_{\text{est}}(\text{p})} \right) < \theta \right] . \tag{15}$$

Again, $\mathcal{V}$ holds the set of pixels for which both $\mathcal{D}_{\text{est}}$ and $\mathcal{D}_{\text{gt}}$ have valid depth measurements. Similarly, $\mathcal{E}$ denotes the set of pixels with valid estimates, while $\mathcal{G}$ holds the pixels with valid ground truth values. In both Equations (14) and (15), the operator $[\cdot]$ refers to the Iverson bracket. The threshold θ is given as a percentage of the corresponding ground truth value. For example, $\text{Acc}_{1.25}$ and $\text{Cpl}_{1.25}$ give the fraction of pixels with respect to the $\mathcal{D}_{\text{est}}$ and $\mathcal{D}_{\text{gt}}$ for which the difference between the estimate and the ground truth is less than 25% of the corresponding ground truth depth. These two measures are combined into a single score, the F_θ score, which is the harmonic mean of Acc_θ and Cpl_θ:

$$\text{F}_\theta(\mathcal{D}_{\text{est}}, \mathcal{D}_{\text{gt}}) = 2 \cdot \frac{\text{Acc}_\theta \cdot \text{Cpl}_\theta}{\text{Acc}_\theta + \text{Cpl}_\theta} \,. \tag{16}$$

Thus, a high F_θ-score indicates a good trade-off between the achieved accuracy of the depth map and its completeness with respect to the ground truth.

3. Results

The following sections present the results of the experiments conducted. They evaluate and analyze different aspects of the presented approach, such as accuracy, efficiency, and application-specific usability. First, the chosen configuration of the hyperparameters, i.e., those of the hierarchical processing scheme, similarity metrics and cost function, is outlined in Section 3.1. In Section 3.2, the ability of the three SGM extensions to reconstruct non-fronto-parallel surface structures is evaluated and compared to the effects of using a non-fronto-parallel plane orientation within plane-sweep sampling. An evaluation of the improvements obtained by post-filtering is presented in Section 3.3. In Section 3.4 and Section 3.5, FaSS-MVS is evaluated and compared with related approaches from the literature, both in terms of accuracy and runtime. Finally, the results of use-case-specific experiments are presented and qualitatively illustrated in Section 3.6.

The entire processing pipeline of the presented approach, except for the generation of the Gaussian image pyramids and the parameterization of the plane-sweep algorithm, is implemented in CUDA and thus optimized for massively parallel computing by general purpose computation on a GPU (GPGPU), which in turn is embedded in a C++ application. All experiments and timing measurements were performed on an NVIDIA Titan X (Santa Clara, CA, USA) GPU and an Intel XEON CPU E5-2650 (Santa Clara, CA, USA) running at 2.20 GHz. Although the CPU is designed for a server architecture, only a small part of our approach is run on the CPU, and thus its superiority over commodity desktop hardware is insignificant.

3.1. Configuration of the Hierarchical Plane-Sweep Dense Multi-Image Matching

To find the best parameterization for the hierarchical DIM, i.e., the optimal number of pyramid levels, the best similarity measure and cost function for the plane-sweep DIM, as well as the appropriate plane orientation, we performed several ablation studies as described in Appendix A, Appendix B, and Appendix C, respectively. In summary, the input bundle size is set to $|\Omega| = 5$ and the pyramid height is set to $n = 3$ for the DTU dataset and $n = 2$ for the 3DOMcity dataset. A fronto-parallel plane orientation, i.e., $\mathrm{n} = (0\ 0\ -1)^\intercal$, is used for the plane-sweep sampling. As a similarity measure and cost function in the DIM, the truncated, inverted and scaled NCC with a support region of 5×5 pixels is used. Although the NCC with a support region of 9×9 pixels achieves the best results, $\mathrm{NCC}_{5\times5}$ is chosen for further experiments, since the error increase is small, but the computational complexity is significantly lower and the throughput higher than that of $\mathrm{NCC}_{9\times9}$ as measured by Ruf et al. [36]. Based on the chosen cost function, we set the SGM penalty φ_1 to 100. To preserve depth discontinuities at object boundaries, we adaptively adjust the second penalty φ_2 based on the absolute intensity difference between two neighboring pixels, as formulated by Scharstein et al. [41], with $\alpha = 8$ and $\beta = 10$. And since the presented approach uses multiple matching images, the SGM penalties are multiplied by the number of input images within the left and right subsets with respect to $\mathcal{I}_{\mathrm{ref}}$, since the matching costs are summed within these image sets. These hyperparameters will be used for all subsequent experiments.

3.2. Evaluation of the Surface-Aware Extensions to SGM

As described in Section 2.2, in addition to the straightforward combination of SGM with plane-sweep sampling (SGM^{Π}), this work includes two further surface-aware extensions to SGM, namely the incorporation of surface normals to adjust the zero-cost transition in SGM path aggregation ($\mathrm{SGM}^{\Pi\text{-sn}}$) and the penalization of deviations from the gradient of the minimum-cost path ($\mathrm{SGM}^{\Pi\text{-pg}}$). In the following, the results obtained by $\mathrm{SGM}^{\Pi\text{-sn}}$ and $\mathrm{SGM}^{\Pi\text{-pg}}$ in combination with a fronto-parallel sampling plane orientation are evaluated and compared with those obtained by SGM^{Π}.

The quantitative results presented in Table 2 show only small differences in the L1 error between the different implementations of the SGM optimization. While for the DTU dataset, the best results are achieved by the $SGM^{\Pi\text{-pg}}$ implementation, for the 3DOMcity dataset, the standard adaptation of the SGM optimization to plane-sweep sampling, i.e., SGM^{Π}, achieves the lowest error. The relative L1 error shows no difference. This is due to the fact that the individual L1-rel values only start to differ after the fourth decimal place. Nevertheless, the ranking of the L1-rel scores is the same as that of the L1-abs scores. In a qualitative comparison, Figure 6 shows that $SGM^{\Pi\text{-sn}}$ leads to a seemingly smoother depth and normal map (e.g., on the ground plane), but at the same time loses small details and increases unwanted depth discontinuities in some areas, such as the building facade. A close comparison of the normal maps between SGM^{Π} and $SGM^{\Pi\text{-pg}}$ shows slightly less staircasing artifacts in the case of $SGM^{\Pi\text{-pg}}$, which also supports the slightly lower error in Table 2. However, a qualitative comparison of the results on the 3DOMcity dataset in Figure 7 does not show any noticeable differences between the different implementations. The reason for the small L1-abs error achieved by SGM^{Π} on the 3DOMcity dataset is thought to be due to the fact that the 3DOMcity dataset also contains a subset of nadir images in which there are few slanted surfaces and the fronto-parallel orientation of the sampling planes coincides with most of the scene structure.

Table 2. Quantitative comparison of the results obtained by different implementations and adaptations of the SGM algorithm in combination with a fronto-parallel sweeping direction. The best results are underlined.

Dataset	Metric	SGM^{Π}	$SGM^{\Pi\text{-sn}}$	$SGM^{\Pi\text{-pg}}$
DTU	L1-abs (in mm)	19.832 ±16.225	19.768 ±16.192	<u>19.684</u> ±16.154
	L1-rel	0.027 ±0.021	0.027 ±0.021	<u>0.027</u> ±0.021
3DOMcity	L1-abs (in mm)	<u>14.615</u> ±6.254	14.673 ±6.229	15.074 ±6.133
	L1-rel	<u>0.012</u> ±0.007	0.012 ±0.007	0.012 ±0.006

To further quantify the strengths and weaknesses of the three different SGM aggregation strategies, three receiver operating characteristic (ROC) curves, one for each extension, are plotted for each dataset in Figure 8. These curves illustrate the error rate achieved by the corresponding SGM extension as a function of increasing density of the estimated depth map. The density of the depth map is varied by sampling the number of pixels in steps of 5% based on their ordered confidence stored in $\mathcal{C}$, going from a high to a low confidence estimate. The average error rate is quantified by $1\text{-}Acc_{1.05}$ (see Equation (14)) and indicates the number of sampled pixels in $\mathcal{D}$ whose absolute difference from the ground truth exceeds 5% of the ground truth value. Thus, at a low density of $\mathcal{D}$, i.e., a high confidence threshold, the error rate should ideally be at its minimum and then increase with increasing density, reaching the total error of $\mathcal{D}$ at a density of 100%. The plots start at a density of 5%, since the error rate at a density of 0% is undefined. However, analyzing the ROC curves of each method individually is not very meaningful. So in Table 3, we also provide data on the area under curve (AUC) along with the optimal AUC (AUC-Opt.) and the difference between the two (ΔAUC) for the three different SGM implementations, as discussed by Mehltretter and Heipke [59], to quantitatively assess the accuracy of the estimated depth and confidence map. Since the AUC-Opt. represents the area under curve for an optimal confidence map, the smaller the difference ΔAUC, the more accurate the confidence map. The curves in Figure 8 as well as the results in Table 3 support the superiority of the surface-aware SGM extensions over the standard SGM adaptation to plane-sweep sampling. For both datasets, the curves and the difference of the AUC to the AUC-Opt. of $SGM^{\Pi\text{-sn}}$ and $SGM^{\Pi\text{-pg}}$ are lower than those of SGM^{Π}, indicating lower error

rates. However, the fact that most of the ROC curves start with a high error rate at a density of 5%, and then drop before rising again, suggests that the estimated confidence values do not adequately represent the certainty of the depth estimates. The reasons for this are many and will be discussed further in Section 4.4.

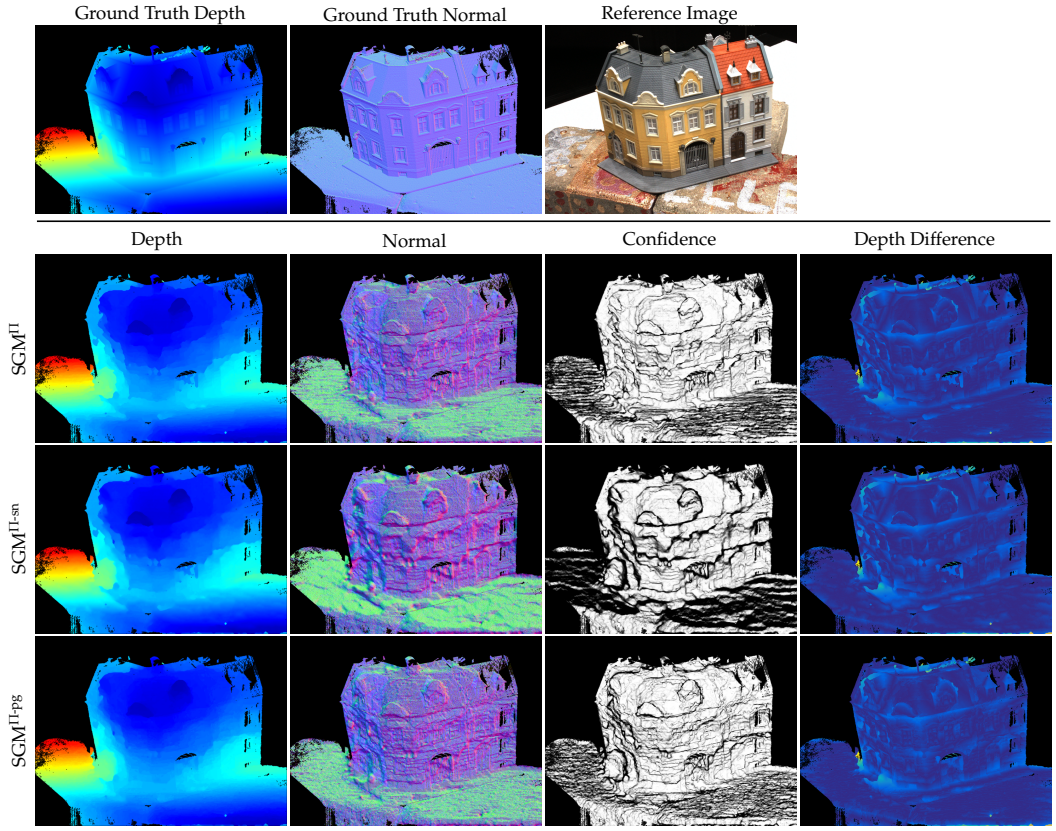

Figure 6. Qualitative comparison of the results achieved by the three different SGM implementations on the DTU dataset. Row 1: Reference data from the dataset, i.e., the ground truth depth and normal map, as well as the reference image for which the data are computed. Rows 2–4: Data, i.e., depth, normal and confidence maps, computed by SGM^{Π}, $SGM^{\Pi\text{-sn}}$ and $SGM^{\Pi\text{-pg}}$, respectively. Furthermore, difference maps are provided which hold the pixel-wise absolute difference between the estimated depth map and the ground truth. The color encoding reaches from dark blue (low error) via green to yellow (high error). The depth range within the depth maps reaches from 580 mm (blue) to 830 mm (red). The estimated maps are masked according to the ground truth.

Table 3. The AUC together with the AUC-Opt. and the difference between those two (ΔAUC) for the three different SGM implementations. The best results are underlined.

Dataset	Metric	SGM^{Π}	$SGM^{\Pi\text{-sn}}$	$SGM^{\Pi\text{-pg}}$
DTU	AUC	1245.5	1157.5	<u>1150.3</u>
	AUC-Opt.	180.5	180.5	180.8
	ΔAUC	1065.0	977.0	<u>969.6</u>
3DOMcity	AUC	338.0	312.2	<u>290.0</u>
	AUC-Opt.	192.6	192.6	193.0
	ΔAUC	145.4	119.6	<u>96.9</u>

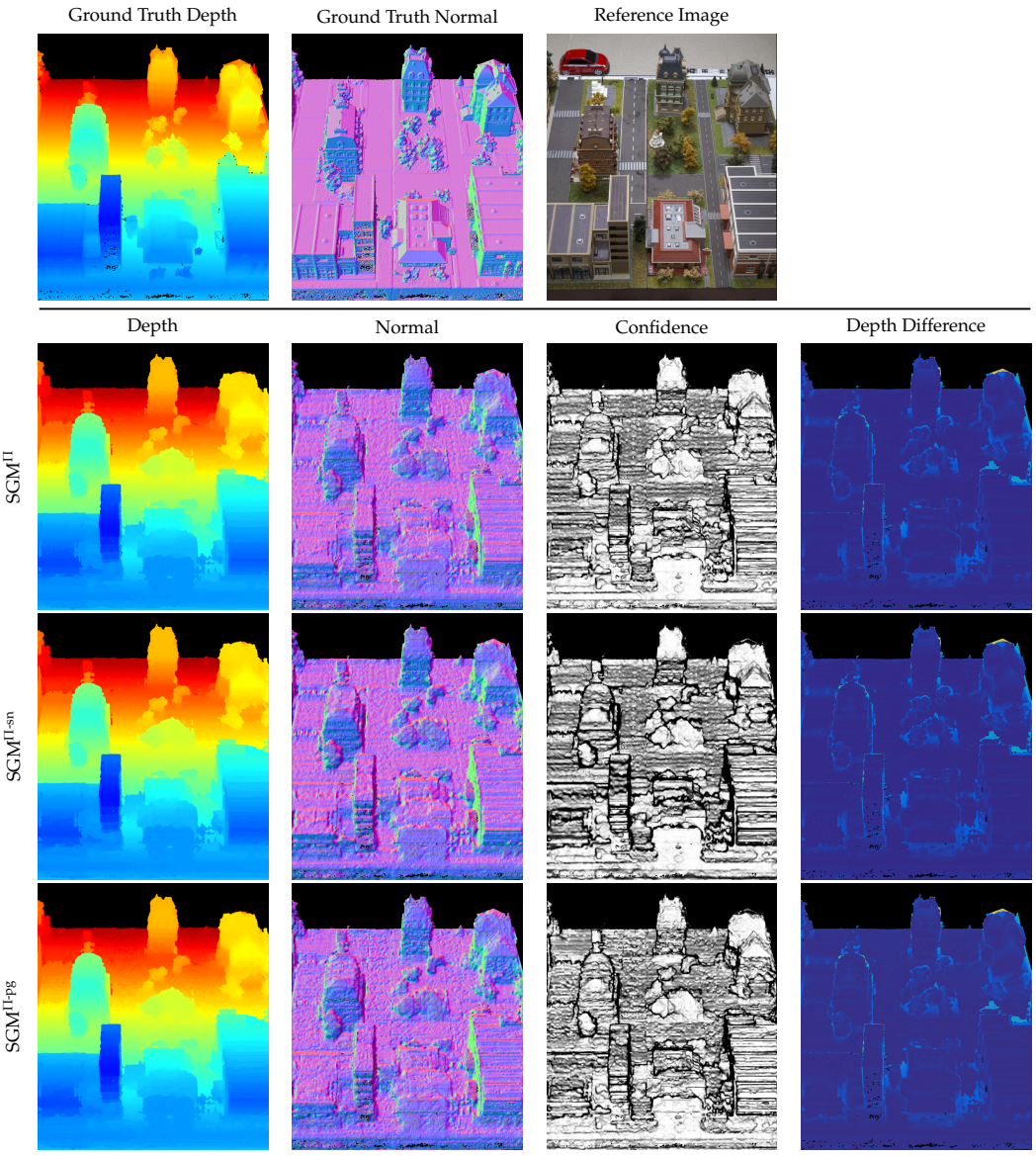

Figure 7. Qualitative comparison of the results achieved by the three different SGM implementations on the 3DOMcity dataset. Row 1: Reference data from the dataset, i.e., the ground truth depth and normal map, as well as the reference image for which the data are computed. Rows 2–4: Data, i.e., depth, normal and confidence maps, computed by SGM^{Π}, $SGM^{\Pi\text{-sn}}$ and $SGM^{\Pi\text{-pg}}$, respectively. Furthermore, difference maps are provided which hold the pixel-wise absolute difference between the estimated depth map and the ground truth. The depth range within the depth maps reaches from 1 m (blue) to 1.8 m (red). The estimated maps are masked according to the ground truth. For visualization in this figure, the resulting images have been rotated counterclockwise by 90°. Thus, the color encoding of the normal maps differs from that used in the other figures. Here, red represents an upwards orientation, while green represents an orientation to the left.

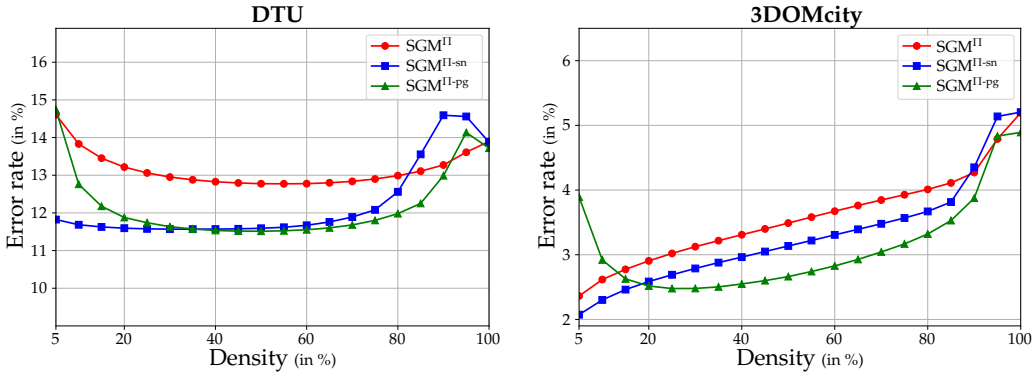

Figure 8. ROC curves illustrating the error rate achieved by the three different SGM implementations as a function of increasing density of the estimated depth map.

3.3. Improvements Gained by Post-Filtering

In the following, the effects of the implemented post-filtering methods to remove remaining outliers and supposedly wrong estimates by Difference-of-Gaussian (DoG) filtering (see Section 2.5.1) and a geometric consistency check (GCC) (see Section 2.5.2) are examined. While the latter relies on the actual estimates, the DoG filter is based on the assumption that image regions with low texture could lead to ambiguities in the image matching and, in turn, incorrect estimates. However, this can lead to the erroneous removal of good or even correct estimates.

Instead of using the absolute and relative L1 metrics to quantitatively evaluate the results achieved when using post-filtering, the effects are evaluated using the accuracy measure Acc_θ (see Equation (14)) and the completeness measure Cpl_θ (see Equation (15)). This is because they indirectly contain information about the density of the resulting depth maps, which should ideally be as high as possible. Since the individual sequences of the 3DOMcity dataset consist of too few images to perform a GCC with the parameterization mentioned in Section 2.5.2, this experiment is only performed on the DTU benchmark dataset. Figure 9 shows the results of different post-filtering strategies, i.e., DoG filtering, GCC, and a combination of both, performed in combination with the three different SGM extensions and a fronto-parallel sampling. For reference, the accuracy–completeness curves resulting from the corresponding configurations without post-filtering are also shown. When constructing the curves, the threshold θ is varied within the list of $\{1.25, 1.20, 1.15, 1.10, 1.05, 1.01\}$. Note that, as the threshold decreases, the accuracy and completeness rates also decrease. The highest values are obtained with $\theta = 1.25$.

Most evidently, Figure 9 again shows that there is not much difference in the overall error between the three SGM implementations. However, the accuracy–completeness curves clearly show the differences between the post-filtering strategies. Unsurprisingly, the reference configuration with no filtering achieves the highest completeness, since no estimates are removed from the predicted depth map, which results in the lowest accuracy. The use of DoG filtering significantly improves this, as it is likely to remove a significant number of false estimates from poorly textured areas. However, as expected, the DoG filter probably also removes a number of correct estimates, as the use of filtering based on geometric consistency achieves a similar completeness, but with a higher accuracy. In particular, looking at the values for $\theta = 1.01$, i.e., the lower left end of each curve, the use of a GCC achieves an increase in completeness of about 5%, while exceeding the accuracy of the DoG filter by more than 10%. However, a clear recommendation as to which filter to use cannot be made, since both filtering strategies have their strengths and weaknesses, especially with respect to online processing, as discussed in Section 4.3. A combination of both filters is not motivated. Although the accuracy increases slightly, the completeness decreases by more than 20% in some cases. Moreover, this effect can also be achieved by lowering the threshold of the reprojection error η_r in the geometric consistency check, which will probably increase the accuracy even more.

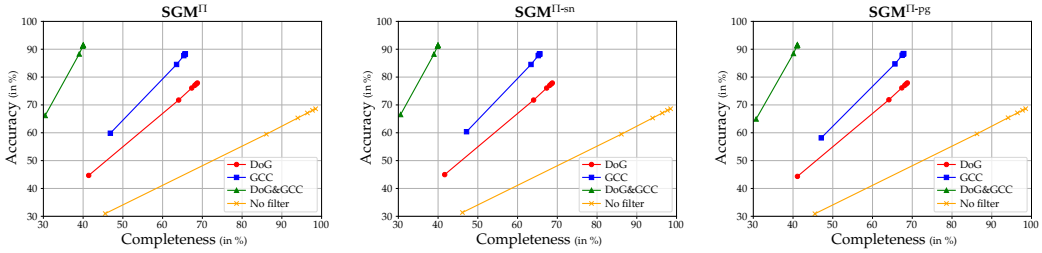

Figure 9. Accuracy–completeness curves of different post-filtering strategies, i.e., DoG filtering, GCC as well as a combination of both, executed in combination with the three different SGM extensions and a fronto-parallel sampling. In this, the threshold θ is varied within the list of $\{1.25, 1.20, 1.15, 1.10, 1.05, 1.01\}$. By decreasing θ, the accuracy and completeness rates drop.

Finally, to directly compare the different SGM extensions in combination with the GCC that gives the best results, the corresponding F-scores (see Equation (16)) for each evaluated θ are listed in Table 4. Just like the results shown in Table 2, the F-scores reveal the superiority of SGM$^{\Pi\text{-pg}}$ over the other two implementations, since for all θ but one, SGM$^{\Pi\text{-pg}}$ achieves the highest F-score.

Table 4. F-scores achieved by the SGM^{x} approaches together with the post-filtering based on GCC. The best results are underlined.

Approach	$F_{1.25}$ (in %)	$F_{1.20}$ (in %)	$F_{1.15}$ (in %)	$F_{1.10}$ (in %)	$F_{1.05}$ (in %)	$F_{1.01}$ (in %)
SGM$^{\Pi}$	74.2	74.1	74.0	73.7	71.5	51.9
SGM$^{\Pi\text{-sn}}$	74.1	74.1	74.0	73.6	71.5	<u>52.3</u>
SGM$^{\Pi\text{-pg}}$	<u>75.6</u>	<u>75.5</u>	<u>75.4</u>	<u>75.1</u>	<u>72.9</u>	51.4

3.4. Comparison to Related Approaches from Literature

Based on the previous experiments and the knowledge gained about the best performing configuration, we now perform a series of experiments on the DTU dataset to compare FaSS-MVS with related approaches from the literature. On the one hand, we compare our results with those of a related approach for online dense MVS, namely the PlaneSweepLib (PSL) [31], which is also used by OpenREALM [6]. The algorithm provided by the PSL is very similar to ours, but does not have hierarchical processing and does not perform post-processing based on geometric queues. In addition, the PSL uses a Bayesian formulation to extract the depth map from the generated depth hypotheses, while FaSS-MVS relies on optimizing an MRF using dynamic programming. In the following experiments, we configure the PSL to also use five input images, 128 planes to generate depth hypotheses, the NCC as similarity measure, and the reference split [50] to account for occlusions. We compare the performance of the PSL with different sized support regions for the NCC, namely with a neighborhood size of 5×5 pixels and 11×11 pixels. And since the PSL does not include a geometric verification of the estimates, we also combine it with the filtering based on GCC in the same configuration as described above.

We also evaluate FaSS-MVS against two offline MVS approaches, namely the widely used and open source COLMAP toolbox and the more recent ACMMP [18]. While COLMAP provides the complete reconstruction pipeline, i.e., including the estimation of camera poses by SfM and the fusion of the depth maps into a 3D point cloud, only the geometric depth maps estimated by the provided MVS approach [4] with the default configuration are used for comparison. Like many other MVS techniques, COLMAP as well as FaSS-MVS and PSL have difficulty estimating reliable pixel correspondences and thus depth values in poorly textured image regions. The recent ACMMP approach enhances MVS depth estimation with multi-scale geometric consistency and a planar prior to reduce ambiguity in image regions with little texture information, resulting in denser depth maps.

The results achieved by FaSS-MVS with its three different SGM strategies, as well as the results obtained by the three other approaches, are listed in Table 5. While $SGM^{\Pi\text{-pg}}$ outperforms the other two SGM extensions in terms of F-score, $SGM^{\Pi\text{-sn}}$ has the lowest L1 error. This can be explained by the density of the depth maps. When $SGM^{\Pi\text{-pg}}$ is used, more estimates pass the geometric consistency check, resulting in depth maps that are slightly more dense than those produced by SGM^{Π} and $SGM^{\Pi\text{-sn}}$, increasing the F-score but also increasing the L1 error. Quantitatively speaking, however, the difference is only marginal, and a conclusion as to whether one particular SGM extension should be preferred over the others depends on the use case and should be drawn based on qualitative comparisons.

Table 5. Quantitative comparison of FaSS-MVS with its three SGM extensions, combined with post-filtering based on geometric consistency checking, with related approaches from the literature on the data of the DTU benchmark. As a reference, the results of the PSL [31] for online MVS, with differently sized support regions for the NCC, as well as with GCC are given. The results of two offline MVS approaches, namely COLMAP [4] and ACMMP [18], are provided for reference. The best results are underlined.

Approach	L1-abs (in mm)	L1-rel	$F_{1.25}$ (in %)	$F_{1.20}$ (in %)	$F_{1.15}$ (in %)	$F_{1.10}$ (in %)	$F_{1.05}$ (in %)	$F_{1.01}$ (in %)
FaSS-MVS-SGM^{Π}	8.549 ± 7.509	0.012 ± 0.011	74.2	74.1	74.0	73.7	71.5	51.9
FaSS-MVS-$SGM^{\Pi\text{-sn}}$	8.479 ± 7.559	0.012 ± 0.011	74.1	74.1	74.0	73.6	71.5	52.3
FaSS-MVS-$SGM^{\Pi\text{-pg}}$	8.722 ± 7.255	0.013 ± 0.010	75.6	75.5	75.4	75.1	72.9	51.4
PSL-$NCC_{5\times5}$	73.924 ± 23.686	0.106 ± 0.027	67.3	62.5	56.5	48.4	35.3	9.9
PSL-$NCC_{5\times5}$-GCC	2.32 ± 1.26	0.003 ± 0.002	32.5	32.5	32.5	32.5	32.4	31.2
PSL-$NCC_{11\times11}$	51.229 ± 28.209	0.071 ± 0.035	72.2	69.5	66.0	61.1	51.0	21.9
PSL-$NCC_{11\times11}$-GCC	$\underline{2.17} \pm 1.23$	$\underline{0.003} \pm 0.002$	61.1	61.1	61.1	61.1	61.0	58.9
COLMAP	3.745 ± 5.498	0.006 ± 0.004	$\underline{80.2}$	$\underline{80.2}$	$\underline{80.1}$	$\underline{80.0}$	$\underline{79.6}$	$\underline{74.4}$
ACMMP	12.963 ± 13.379	0.018 ± 0.018	77.5	77.2	76.6	75.5	73.0	55.2

Compared to PSL in the configuration proposed by Häne et al. [31], i.e., without geometric post-processing, FaSS-MVS performs significantly better. This is due to the fact that the PSL does not include outlier removal, resulting in a high L1 error and a lower F-score. When combined with outlier filtering based on geometric consistency, the L1 of the resulting depth maps is the lowest of all the approaches evaluated, even lower than that of the two offline MVS approaches. However, the F-score is also significantly reduced due to the low completeness of the depth maps, as can be seen in Figure 10.

Offline MVS approaches are said to be superior to online approaches due to the availability of more input images and the absence of runtime constraints. And while COLMAP is slightly outperformed by PSL in combination with GCC in terms of the L1 error, it clearly achieves overall superiority in terms of the F-score and thus the trade-off between accuracy and completeness. When comparing the mean density of the resulting depth maps, ACMMP outperforms COLMAP by more than 24%. Surprisingly, however, ACMMP has a high L1 error and a low F-score. This suggests that the estimates computed by ACMMP in low-texture areas, where the other approaches do not provide estimates, are not very accurate. The significance of a comparison between online and offline MVS approaches can be questioned, however, since the two types of approaches make different assumptions and focus on different aspects within the processing, as further discussed in Section 4.1.

3.5. Runtime Comparison

As motivated above, the presented approach aims at incremental and online processing, i.e., the computation should ideally keep up with the input stream. Therefore, the total runtime of FaSS-MVS with its three SGM extensions compared to the comparable approach of PSL is evaluated in Table 6. In addition to the standard use of eight aggregation paths within the SGM optimization, which achieves the lowest error and has been used in previous experiments, the runtime and accuracy reduction of using only four aggre-

gation paths is listed. This is motivated by a number of studies [35,36,60] that show that reducing the number of aggregation paths from eight to four can significantly reduce the computational time of SGM aggregation, while only marginally increasing its error. All measurements were conducted without any post-processing, i.e., DoG filtering or filtering based on geometric consistency.

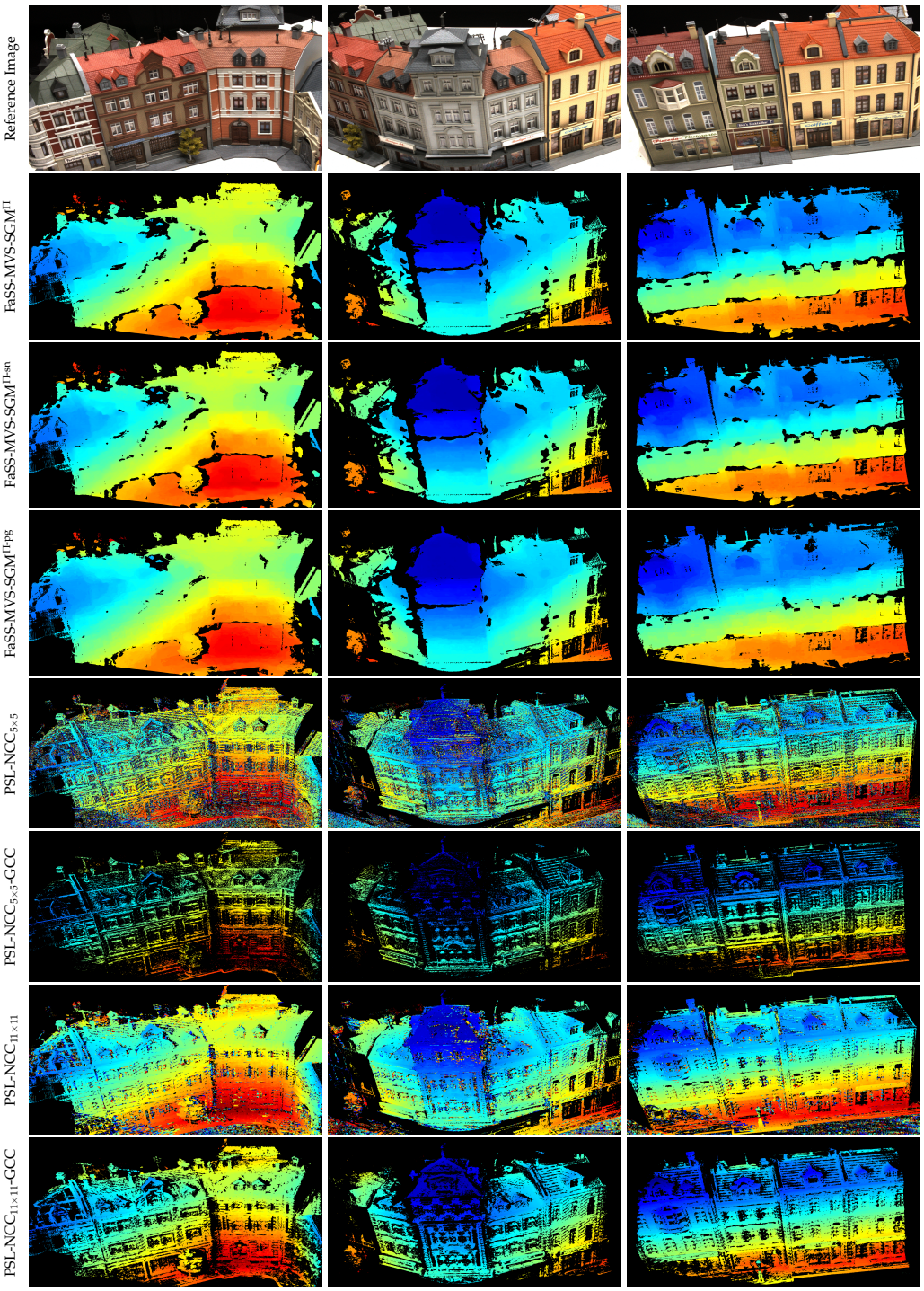

Figure 10. Qualitative comparison of FaSS-MVS with its three SGM extensions and GCC, the PSL with differently sized support regions for the NCC, as well as with GCC.

Table 6. Runtime comparison of FaSS-MVS with its three SGM extensions and the comparable approach from the PSL [31]. Measurements were performed on the DTU benchmark dataset and represent the average runtime required by the different approaches to estimate a single depth map. With respect to FaSS-MVS, the difference between using 8 and 4 aggregation paths within the SGM optimization is also evaluated. The best results are underlined.

Metric	FaSS-MVS-SGM$^{\Pi}$		FaSS-MVS-SGM$^{\Pi\text{-sn}}$		FaSS-MVS-SGM$^{\Pi\text{-pg}}$		PSL
	8-Path	4-Path	8-Path	4-Path	8-Path	4-Path	NCC$_{5\times5}$
Runtime (in ms)	640	413	895	546	2079	1132	<u>139</u>
Δ L1-abs (in %)		+6.3		+6.7		<u>+6.2</u>	

The measurements clearly show that the PSL is much faster than FaSS-MVS. However, as the use-case-specific experiments in the following section show, the runtime of FaSS-MVS can vary greatly due to the variable number of sampling planes, depending on the distance between the input images and thus the observable depth range. The measurements also show that especially the SGM$^{\Pi\text{-pg}}$ extension introduces a large computational complexity compared to SGM$^{\Pi}$ and SGM$^{\Pi\text{-sn}}$. However, reducing the number of aggregation paths has a large impact on the runtime, reducing it by up to 45%, while having only a marginal impact on the error. Whether the listed runtime is sufficient for online processing is further discussed in Section 4.3.

3.6. Use-Case-Specific Experiments Conducted on Real-World Datasets

Finally, to demonstrate the performance of the presented approach on use-case-specific and real-world datasets, experiments are performed using SGM$^{\Pi\text{-pg}}$ with four aggregation paths and the above configuration on the TMB and FB datasets. Samples of the computed depth, normal, and confidence maps is shown in Figure 11, along with the corresponding depth maps estimated by COLMAP as a reference. The average processing time for the TMB dataset is 690 ms, but can vary between 320 ms and 1218 ms depending on the arrangement of the input data and the number of plane distances δ at which the scene is sampled. For the FB dataset, the average processing time is 800 ms, again varying between 514 ms and 1419 ms depending on the arrangement of the input images.

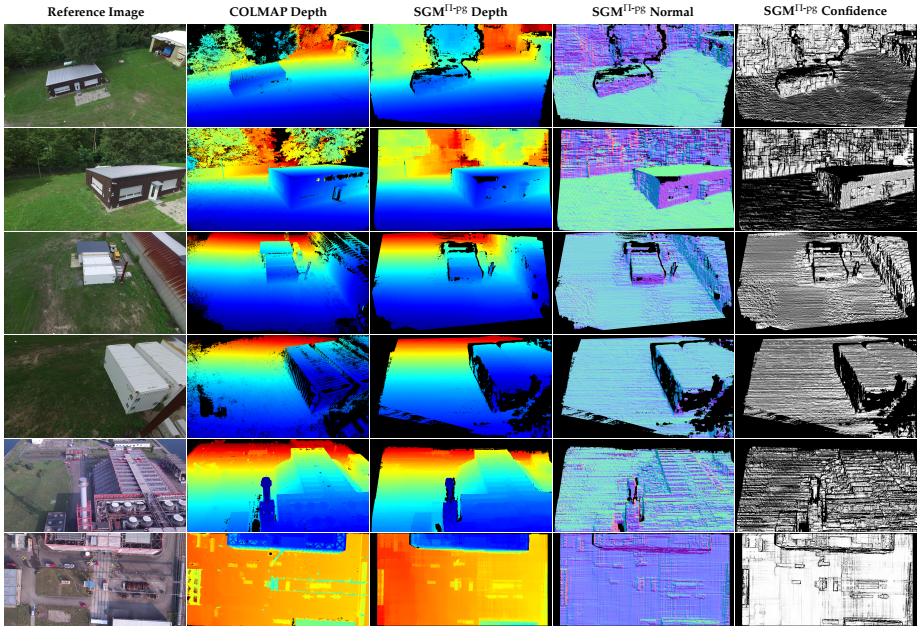

Figure 11. Qualitative results of SGM$^{\Pi\text{-pg}}$ with 4 aggregation paths achieved on the two real-world and use-case-specific datasets, namely the TMB dataset and the FB dataset. As comparison, the corresponding depth maps estimated by COLMAP are also visualized. Rows 1 and 2: TMB Building scene captured from an altitude of 15 m and 8 m, respectively. Rows 3 and 4: TMB Container scene. Rows 5 and 6: Two excerpts from the FB dataset.

4. Discussion

In the following, the results of the conducted experiments are discussed with respect to different aspects, namely the overall accuracy (Section 4.1), the ability of the presented approach to reconstruct slanted surface structures (Section 4.2), the runtime and support for online processing (Section 4.3), and the effects of the post-filtering algorithms used and the relevance of the confidence estimates (Section 4.4).

4.1. Overall Accuracy

In Table 5, FaSS-MVS is evaluated against a comparable online MVS approach from the PSL. Considering the results achieved by the approach provided by the PSL, it is clearly outperformed by FaSS-MVS. The comparison is somewhat unfair, however, since the results presented by FaSS-MVS have undergone filtering based on a GCC. For this reason, additional experiments were performed in which the same filtering was applied to the PSL results, showing a superiority of the PSL with GCC over FaSS-MVS with respect to the L1 error. However, as suggested by the lower F-score and as shown by the extracts in Figure 10, the post-filtering leads to the removal of quite a few estimates in the depth maps computed by PSL. This again underlines the strength of FaSS-MVS in computing dense depth maps with high consistency and accuracy.

Furthermore, as the results in Table 5 show, the overall accuracies of the depth maps estimated by the presented approach are lower than those achieved by the two offline MVS approaches, i.e., COLMAP and ACMMP. This is not surprising, since the procedure and assumptions involved are very different between online approaches, such as the one presented in this paper, and offline approaches. Offline approaches assume that all input images are available at the time of reconstruction, allowing them to optimize the set of input images considered for the reconstruction of a given viewpoint. In contrast, online approaches, which perform MVS incrementally, only consider input images within a temporally limited window, at most all images acquired up to a certain point in time. In addition, offline approaches typically do not have time constraints either. Nevertheless, the quantitative differences between the results obtained with the presented approach and COLMAP are not that large, less than an order of magnitude, and even exceed those obtained with ACMMP. Furthermore, a qualitative comparison on use-case-specific input data makes the results of the presented approach very satisfactory. Compared to the geometric depth maps of COLMAP, the depth maps of $\text{SGM}^{\Pi\text{-pg}}$ lack the fine-grained details, such as the roof structures in rows 5 and 6 of Figure 11, which are caused by the coarse-to-fine processing. However, larger structures are well represented and the quality of their reconstruction is comparable, as is the overall density. Although the fronto-parallel bias of SGM is reduced, some artifacts of fronto-parallel sampling are still visible, especially in the normal maps of Figure 11.

4.2. Ability to Account for Non-Fronto-Parallel Surfaces

To further increase the accuracy of the reconstruction of slanted, non-fronto-parallel surface structures, this work proposes, in addition to SGM^{Π}, two extensions to the SGM algorithm that should reduce the fronto-parallel bias. Namely, the incorporation of surface normals to adjust the zero-cost transition in the SGM path aggregation ($\text{SGM}^{\Pi\text{-sn}}$) and the penalization of deviations from the gradient of the minimum-cost path ($\text{SGM}^{\Pi\text{-pg}}$). The experiments conducted show that these extensions provide only a slight quantitative improvement over the standard SGM adaptation (SGM^{Π}) to plane-sweep sampling. This finding is in contrast to the experiments of Scharstein et al. [41]. There are at least two reasons for this discrepancy. First, Scharstein et al. [41] demonstrate their implementation on a two-view stereo dataset, where the input images are captured by two cameras mounted on a fixed rig and oriented in the same direction. In addition, prior to processing, the images are rectified, i.e., transformed, so that both lie in the same image plane and the epipolar lines coincide with the image rows. Thus, in the dense image matching process, the images are sampled equidistantly with a step size of 1 pixel. In the case of the presented approach,

however, the distances of the sampling planes and thus the sampling points are chosen in such a way that the disparity shift along the epipolar line between two consecutive planes is less than or equal to 1. This results in sampling with a much higher density, which already reduces the staircase effect in the case of SGM^{Π}. And secondly, Scharstein et al. [41] proposes to use a ground truth normal map to adjust the zero-cost transition, whereas in the presented approach, the upscaled normal map of the previous iteration of the hierarchical processing is used. This is bootstrapped with SGM^{Π} at the highest pyramid level, which introduces inaccuracies that probably cannot be fully compensated for. However, the qualitative analysis shows that $SGM^{\Pi\text{-sn}}$ and $SGM^{\Pi\text{-pg}}$ clearly lead to smoother normal maps and reduce staircase artifacts in the depth maps, which is why only $SGM^{\Pi\text{-pg}}$ is considered in the use-case-specific experiments.

In addition to reducing the fronto-parallel bias in the SGM path aggregation, the plane-sweep algorithm in our approach allows us to adapt the image matching to the surface structures in the scene by selecting appropriate normal vectors and sweeping directions. In a short qualitative experiment (see Figure A1), the effects of a horizontal plane sampling compared to a fronto-parallel sampling, both in combination with SGM^{Π}, are investigated. The results show that horizontal sampling leads to more consistent depth estimates with little or no staircasing artifacts in areas where the surface structure coincides with the plane orientation, e.g., the ground plane. However, in areas where the surface structure is not horizontal, non-fronto-parallel sampling introduces significant errors. To overcome this effect, one can consider dividing the scene into local regions that are individually sampled with different plane orientations, similar to the local-plane-sweep approach presented by Sinha et al. [38]. However, this comes at the cost of higher computational complexity. Another remedy is to repeat the plane-sweep image matching several times on the whole image domain prior to the SGM optimization, with different sweeping directions, and to perform a pixel-wise pre-selection of the best plane orientation based on the matching costs, similar to the approach of Pollefeys et al. [25]. This results in a smaller increase in computational complexity compared to the first option.

4.3. Runtime and Online Processing

Given the runtime measurements in Table 6, the presented approach is obviously not capable of real-time and low-latency processing, in the sense that for each input frame, a depth map is computed at similar frame rates as given for the input stream. However, considering the nature of the approach and the expected input data, the runtime is generally sufficient for online processing, which will be explained in the following section. The presented approach takes a bundle of three or more input images, with a bundle size of five images actually yielding better results, and performs MVS on a reference image of the input bundle, typically the middle one. While these input images could be provided by individual cameras, it is assumed that the images are extracted from an input sequence captured by a single camera moving around a static scene. In addition, not every frame of the input sequence can be used, since a suitable baseline must lie between each input frame to enable scene depth estimation. This, of course, depends on the depth range to be sampled and the scene structure. In the case of the TMB dataset, the average distance between the individual input images is 1.8 m and 1.03 m for a flight altitude of 15 m and 8 m, respectively. This increases at higher altitudes due to greater scene depth. Modern COTS rotor-based UAVs can fly up to a speed of over 10 m/s. However, the typical flight speed for image acquisition is closer to 1–3 m/s [61,62]. Thus, if the sets of input images are disjoint, then an estimation needs to be performed at least every 3 s, considering a low flight altitude together with a high flight speed of about 3 m/s and an input bundle size of three images. If a maximum overlap between the input bundles is desired, i.e., a new depth map estimation is triggered with each new suitable input frame and it reuses four images from the previous bundle, the required runtime is significantly lower. However, as the use-case-specific experiments for the TMB and FB datasets show, the average processing time of $SGM^{\Pi\text{-pg}}$, which is the most computationally expensive variant, is between 1 and

2 Hz, depending on the arrangement of the input images. Another way to reduce the runtime is to use higher Gaussian pyramids, which again comes at the cost of a reduced level of detail, as already pointed out in the discussion on overall accuracy (see Section 4.1). In short, there are a number of possible settings both in the acquisition of the input data, e.g., regarding the flight speed or the size and overlap of the input bundles, and in the configuration of the presented approach, e.g., regarding the Gaussian pyramid height, the depth range or the optimization strategy, which allow us to adapt the runtime to the rate of the input images and thus to allow online processing.

The emergence of high-performance systems-on-a-chip (SoCs) with embedded GPUs, such as the NVIDIA Jetson series, allows approaches like FaSS-MVS to be brought directly to the sensor platform, e.g., the UAV, for on-board processing. To evaluate the feasibility of running FaSS-MVS on-board an embedded device, some additional runtime measurements were performed on the NVIDIA Jetson AGX, equipped with an 8-core 64-bit ARMv8.2 CPU and a 512-core Volta GPU. On an excerpt of the TMB dataset with an image size of 1920×1080 pixels, FaSS-MVS with SGM^{Π} achieves an average runtime of 727 ms on the Jetson AGX, compared to an average runtime of about 403 ms on the NVIDIA Titan X. As already discussed, the runtime can be further reduced by increasing the pyramid height to $n = 4$ and $n = 5$, for example, while accepting a decrease in the quality of the results. This results in average runtimes of 444 ms and 385 ms, respectively. These experiments show that FaSS-MVS is capable of on-board processing using a high-performance embedded SoC such as the NVIDIA Jetson AGX. This may be of particular interest when considering deployment on a sensor carrier that does not suffer from severe power constraints.

4.4. Post-Filtering and the Relevance of the Estimated Confidence Values

A comparison of the L1 errors in Table 5 with those listed in Table 2 shows that using post-filtering based on geometric consistency can drastically reduce the mean errors by about 40%. The trade-off for this improvement is a loss of density in the depth map and an increase in latency between input and results. The latter is due to the additional sliding window introduced by geometric consistency-based filtering. In addition to the bundle of input images for which only one set of estimates is produced, the geometric filter also requires two or more depth maps for processing. The geometric filter is also more computationally expensive than the DoG filter. Again, whether to use the DoG or the geometric filter depends on the application. For example, if the presented approach is used for the task of online 3D reconstruction, i.e., a subsequent depth map fusion step is used [7], the geometric-consistency-based filtering is typically performed in the depth map fusion and can thus be omitted. The DoG filter, on the other hand, is very efficient and does not introduce any additional latency. However, as mentioned in the experiments, the DoG filter may also remove potentially good estimates, since it is performed only on the data provided by the input image. Nevertheless, the DoG filter is of great benefit, especially when working with input data containing many homogeneous areas with little or no texture, e.g., a clear or cloudy sky in case of extreme viewing angles.

Finally, as a third output, the presented approach computes a confidence map containing pixel-wise confidence values corresponding to the depth estimates. In this work, these confidence measures are used to perform a comparison between the different SGM extensions based on an ROC analysis (see Figure 8). As noted above, the fact that some of the curves are not monotonically increasing suggests that the confidence values do not adequately represent the certainty of the estimates. For example, the fact that the scene in row 6 of Figure 11 consists mostly of fronto-parallel structures leads to a confidence map with high certainty values, while the confidence map in row 2 of Figure 11 makes the estimation of the roof of the building, which appears qualitatively very accurate, completely uncertain. A similar observation can be seen in the confidence maps shown in Figure 6. Only because the ground plane is highly tilted with respect to the image plane, the confidence of the corresponding estimates becomes very low, even though they do not appear qualitatively more accurate than the estimates on the building facade. The most likely reason is that

modeling a confidence score based on surface orientation alone is not very meaningful. Incorporating additional heuristics based on internal properties of the algorithm, as carried out in previous work [12], could improve the confidence estimation, but this still requires a cumbersome empirical study of the hyper-parameters. In recent years, however, the performance of learning-based approaches to confidence estimation [63,64] has improved significantly. They are often agnostic to the internals of the algorithm and can be trained on any data for which both estimated and reference depth maps are available.

5. Conclusions

In conclusion, we present an approach for multi-view stereo (MVS) from UAV-borne imagery that allows for fast, dense, and incremental 3D mapping. This approach consists of a hierarchical processing scheme that estimates dense depth maps and corresponding normal and confidence maps. For the depth map computation, dense multi-image matching using the plane-sweep algorithm is used to generate pixel-wise depth hypotheses. From these hypotheses, a dense depth map is extracted using the optimization scheme of the widely used Semi-Global Matching (SGM) algorithm. Here, the SGM algorithm is not only adapted to work with the multi-image matching of the plane-sweep algorithm, but also extended to reduce the fronto-parallel bias and to account for slanted surface structures by introducing two additional regularization schemes. The successive normal and confidence map estimation is performed separately on the results of the depth estimation. In a final filtering step, geometric consistency is enforced over multiple depth maps, which greatly increases the overall accuracy of the resulting depth maps.

The performance of our approach is quantitatively evaluated on two public datasets containing image data of model-scaled scenes captured from an aerial perspective and providing accurate ground truth. The experiments show that for the best configuration, the estimated depth maps have a mean absolute L1 error of only 8.5 mm on the DTU dataset, or 1%, with respect to the maximum depth of the reconstructed scene. In comparison, on the same dataset, the geometric depth maps from COLMAP, a widely used open-source toolbox for offline MVS, have a mean absolute error of 3.8 mm. Thus, even though the presented approach does not have all image data of the input sequence available at the time of reconstruction and is subject to runtime constraints to ensure fast and online processing, its quantitative results are not too far off from state-of-the-art offline approaches. While the quantitative results do not show a significant improvement by the presented SGM extensions to account for slanted surface structures, a qualitative comparison reveals their ability to account for non-fronto-parallel surfaces. Thus, in the case of oblique aerial imagery containing many slanted surfaces, the presented SGM extension, which penalizes deviations from the gradient of the minimum-cost path, i.e., $\text{SGM}^{\Pi\text{-pg}}$, is the best choice, despite its higher computational complexity. Final experiments on real-world and use-case-specific datasets have shown that the presented approach is well suited for online processing in terms of runtime, achieving a processing rate of 1–2 Hz, meaning that it keeps up with the monocular input stream and allows for incremental 3D mapping as input data are received. Fast 3D mapping, in turn, can facilitate other important applications or tasks, such as rapid assessment of inaccessible areas by emergency responders, e.g., after a flood or earthquake, to perform disaster relief or search and rescue missions.

Finally, there are also some aspects to consider for future work. Although the approach supports different plane orientations in plane-sweep multi-image matching, each estimation is performed with only one orientation. In the future, the approach should be extended to use multiple plane orientations within the computation of a single depth map. This will allow for smoother reconstruction of large planar surfaces such as the ground plane, but will also allow for higher accuracy in other regions by using fronto-parallel sampling. Furthermore, while we note that the processing speed is sufficient, a further reduction in runtime and a more efficient use of GPU resources would free up more opportunities for other concurrent tasks, such as depth map fusion or orthophoto generation. Therefore, further optimization in terms of runtime and utilization of processing resources

is an ongoing task. In addition, due to the ongoing development and rapid advancement of deep learning-based approaches for the task of MVS, we want to investigate whether individual steps or even the entire approach can be replaced by an appropriate learning-based approach, while maintaining the reliability for use in the context of critical applications. Finally, the work of Nex and Rinaudo [65] has shown that the complementary use of Light Detection and Ranging (LiDAR) and image-based techniques for photogrammetric tasks has great potential. In addition, with the improvements in LiDAR sensors and the ability to equip commercial UAVs with such sensors, such as the Zenmuse L1, their use to facilitate fast and incremental 3D mapping will inevitably be considered in future work.

Author Contributions: Conceptualization, B.R.; methodology, B.R.; investigation, B.R.; writing—original draft preparation, B.R.; writing—review and editing, B.R. and M.W.; supervision, S.H. All authors have read and agreed to the published version of the manuscript.

Funding: The APC was funded by the Fraunhofer Publication Fund.

Institutional Review Board Statement: Not applicable.

Informed Consent Statement: Not applicable.

Data Availability Statement: Publicly available datasets were analyzed in this study. This data can be found here: https://roboimagedata.compute.dtu.dk/?page_id=36 (accessed on 27 August 2024) and https://3dom.fbk.eu/3domcity-benchmark (accessed on 27 August 2024).

Acknowledgments: We would like to thank the team from Arbeiter-Samariter-Bund (ASB) Baden-Württemberg e.V. from Karlsruhe for providing us with the dataset of the fire brigade exercise.

Conflicts of Interest: The authors declare no conflicts of interest.

Appendix A. Finding the Optimal Number of Input Images and Pyramid Levels

In this section, we demonstrate the need for and the importance of the hierarchical processing scheme within the presented approach and evaluate different configurations on the size of the input bundle. In this, a couple of aspects are considered in order to find the best configuration. The objective is to find the appropriate number n of Gaussian pyramid levels and the size $|\Omega|$ of the input bundle, providing a good trade-off between:

- the error of the resulting depth maps, measured by L1-abs and L1-rel;
- the sampling density and the entailed resources needed for the computation;
- the resulting processing runtime.

For this experiment, a fronto-parallel plane orientation is used as part of the plane-sweep image matching and the NCC with a support region of 5×5 pixels is set as similarity measure. The optimization of the cost volume and the extraction of the optimal depth map is performed by employing the SGM^{Π} scheme, which is the adoption of the standard SGM optimization to the use of plane-sweep image matching (see Section 2.2). The smoothness penalty within the SGM optimization is set to $\varphi_1 = 100$, while the adaptive φ_2 penalty is used. This, together with the 5×5 sized NCC as matching cost, was chosen in accordance with the work of Scharstein et al. [41]. To find the appropriate height of the Gaussian pyramids, the size of the input bundle, i.e., the number of input images, is set to $|\Omega| = 3$.

Table A1 lists the mean errors of the estimated depth maps when evaluated with different numbers of pyramid levels on the datasets of both the DTU and 3DOMcity benchmark. In this, the absolute and relative L1 measures are used, averaged over all depth maps within each dataset. It is to be expected that the omission of any hierarchical processing, i.e., the use of only one pyramid level and thus no coarse-to-fine processing, would lead to the smallest error between the estimate and ground truth. However, the results reveal that in the case of the DTU dataset, the smallest mean error, even if it is only slightly smaller, is achieved when setting $n = 3$, while the best result in the case of the 3DOMcity dataset is achieved at $n = 1$.

Table A1. Mean errors achieved on the DTU and 3DOMcity datasets for a different number of Gaussian pyramid levels (n) as part of the hierarchical processing scheme. The error metrics used are the absolute L1-abs, measured in mm, as well as the relative L1-rel measure. Both are averaged over all evaluated depth maps within each dataset. The best results are underlined.

Dataset	Metric	$n = 1$	$n = 2$	$n = 3$	$n = 4$	$n = 5$
DTU	L1-abs (in mm)	26.394 ±24.262	26.221 ±23.835	<u>23.473</u> ±19.656	25.045 ±19.298	29.676 ±19.436
	L1-rel	0.036 ±0.032	0.036 ±0.032	<u>0.032</u> ±0.026	0.034 ±0.026	0.041 ±0.026
3DOMcity	L1-abs (in mm)	<u>12.789</u> ±6.916	14.936 ±6.754	21.801 ±8.010	32.458 ±9.408	47.422 ±22.292
	L1-rel	<u>0.010</u> ±0.006	0.012 ±0.006	0.017 ±0.007	0.026 ±0.009	0.037 ±0.014

As described in Section 2.1.3, the plane distances within the plane-sweep sampling, and thus the sampling points, are selected in such a way that two consecutive planes induce a maximum disparity difference of 1 pixel. Depending on the capturing setup, i.e., the relative poses between the images and their obliqueness and, in turn, the range of the scene depth, this can lead to a very high number of sampling points and with it to a large memory consumption, as the dimensions of the three-dimensional cost volume need to be set accordingly. Thus, in order to not exceed the memory limit, the maximum number of sampling points for the highest pyramid level is restricted to 256 in the implementation of the approach. In case of the camera setup of the DTU dataset and the configuration of this experiment, i.e., having a bundle size of $|\Omega| = 3$, a pyramid height of 3 is the smallest height at which the number of sampling points at the highest level does not reach or exceed the set limit, as Table A2 shows. Comparing Tables A1 and A2 further reveals that on both datasets, the best results are achieved when the highest pyramid level has a maximum of 128 sampling planes.

Table A2. Processing runtime measured for different configurations of the pyramid height on the DTU and 3DOMcity datasets. In addition, the maximum number of sampling planes with which the scene was sampled at the highest pyramid level is stated.

Dataset	Metric	$n = 1$	$n = 2$	$n = 3$	$n = 4$	$n = 5$
DTU	Runtime (in ms)	2365 ±15	1315 ±10	386 ±2	220 ±2	187 ±1
	max. num. planes	256	256	128	64	32
3DOMcity	Runtime (in ms)	613 ±3	431 ±3	225 ±1	196 ±1	192 ±1
	max. num. planes	128	64	32	16	8

Another criterion which is used to deduce the best configuration on the height of the Gaussian pyramid is the runtime needed to estimate a single depth map. Table A2 additionally lists the corresponding measurements taken, i.e., the number of milliseconds it takes to estimate a single depth map given a certain number of pyramid levels, as well as the number of planes used for sampling the scene space at the highest pyramid level. The measurements again show that, up to $n = 3$ in the case of the DTU dataset, the number of sampling planes at the highest pyramid level is equal to the limit of 256 and that with a smaller amount of sampling points, the runtime decreased drastically. Furthermore, the significant drop of one second in runtime between using a pyramid height of 2 and 3 suggests that the decreasing use of processing resources on the GPU increases the processing speed and that going from $n = 2$ to $n = 3$ makes a significant improvement in its efficiency. Because the use of a higher number of pyramid levels does not only reduce the amount of sampling points, but also the image size at the highest pyramid level and

with it the amount of pixels that need to be matched, depending on the camera setup, a hierarchical processing is very important in order to ensure a high sampling density of the scene space, while at the same time efficiently utilizing the processing hardware and, in turn, alleviating high processing speeds. In the case of the DTU dataset, this experiment shows that the best number of pyramid levels to be used is $n = 3$, which will thus be set for the successive experiments. In case of the 3DOMcity dataset, Table A1 suggests that the best configuration is to use the original image size. A hierarchical processing scheme is needed, however, in order to use $\text{SGM}^{\Pi\text{-sn}}$, the extension of the SGM algorithm to consider local surface orientations in order to account for slanted surfaces. Thus, in the case of the 3DOMcity dataset, the successive experiments will be executed with $n = 2$, which induces only a slightly higher mean error compared to the best configuration.

In the second part of this experiment, the effects of a different number of input images and, in turn, the optimal size $|\Omega|$ of the input bundle are evaluated. Here, the settings for the plane-sweep image matching and the subsequent SGM optimization are kept the same as before. The height of the Gaussian pyramids is fixed to $n = 3$ in the case of the DTU dataset and $n = 2$ in the case of the data from the 3DOMcity dataset. Table A3 lists the mean errors achieved on both datasets with a different number of input images, as well as the difference in runtime with respect to the best configuration of the first part of the experiment. The results reveal that the best accuracies are achieved when five input images are used for image matching, even though, in the case of the 3DOMcity dataset, it is only a marginal improvement. As expected, the utilization of more input images in the process of image matching also leads to an increase in runtime, since more pixels are matched. At the same time, however, there is more time available to keep up with the image acquisition, as discussed in Section 4.3. In conclusion, in the subsequent experiments, the size of the input bundle is set to $|\Omega| = 5$, while the height of the Gaussian image pyramids is set to $n = 3$ and $n = 2$ in the case of the DTU and 3DOMcity datasets, respectively.

Table A3. Mean errors achieved on the DTU and 3DOMcity datasets for different input bundle sizes ($|\Omega|$), i.e., number of images. In addition, the differences in runtime, with respect to the measurements of the first part (i.e., $|\Omega| = 3$), are stated. The best results are underlined.

Dataset	Metric	$\|\Omega\| = 3$	$\|\Omega\| = 5$	$\|\Omega\| = 7$
DTU	L1-abs (in mm)	23.473 ±19.656	<u>19.832</u> ±16.225	21.843 ±21.605
	L1-rel	0.032 ±0.026	<u>0.027</u> ±0.021	0.031 ±0.031
	Δ Runtime (in ms)		+271	+302
3DOMcity	L1-abs (in mm)	14.936 ±6.754	<u>14.615</u> ±6.254	16.514 ±7.569
	L1-rel	<u>0.012</u> ±0.006	0.012 ±0.007	0.014 ±0.009
	Δ Runtime (in ms)		+360	+410

Appendix B. Evaluating Different Similarity Measures in the Process of Dense Multi-Image Matching

As part of the plane-sweep multi-image matching, this approach comprises two different similarity measures and cost functions: the Hamming distance of the census transform (CT) as well as the truncated, inverted and scaled normalized cross-correlation (NCC). While the CT is computationally less expensive than the NCC and is thus more suitable for real-time or online processing, it is less discriminative, which might result in a more ambiguous set of matched pixel correspondences. When working with a stereo normal case, in which the input images suffer only from a little perspective distortion induced by homographic transformations, the CT outperforms the NCC in both runtime

and accuracy [36]. However, as the results in Table A4 show, the perspective distortion, resulting from the warping of images from converging cameras by means of the plane-induced homography within the plane-sweep algorithm, leads to a significant increase in error when using the CT as a similarity measure instead of the NCC.

Table A4. Mean errors achieved on the DTU and 3DOMcity datasets when using different similarity measures and cost functions with different support regions. The best results are underlined.

Dataset	Metric	$CT_{3\times3}$	$CT_{5\times5}$	$CT_{9\times7}$	$NCC_{3\times3}$	$NCC_{5\times5}$	$NCC_{9\times9}$
DTU	L1-abs (in mm)	42.494 ±39.112	42.136 ±37.958	42.305 ±36.394	26.229 ±17.816	19.832 ±16.225	<u>19.667</u> ±16.453
	L1-rel	0.056 ±0.049	0.056 ±0.048	0.057 ±0.046	0.037 ±0.024	0.027 ±0.021	<u>0.027</u> ±0.021
3DOMcity	L1-abs (in mm)	29.149 ±17.272	22.128 ±14.218	26.005 ±14.106	26.678 ±10.377	14.615 ±6.254	<u>13.789</u> ±5.962
	L1-rel	0.024 ±0.016	0.019 ±0.014	0.022 ±0.014	0.023 ±0.010	0.012 ±0.007	<u>0.011</u> ±0.006

Apart from the two different similarity measures, the effects of different support regions are also evaluated in the scope of this experiment. In this, for each similarity measure, the most commonly used configurations were tested. A support region of a size of 5×5 pixels represents a good trade-off between uniqueness and computational complexity, while, in the case of the CT, a support region of a size of 9×7 pixels is the biggest size for which the bit-string still fits into a single 64-bit integer. The configuration of the plane-sweep algorithm and the SGM optimization is set in accordance with the values from the first experiment (see Appendix A). In terms of the SGM penalties, φ_1 is set to 100 for all $NCC_{3\times3}$, $NCC_{5\times5}$ and $NCC_{9\times9}$, since the maximum matching cost of the NCC is normalized to 255, independent of the support region. For $CT_{3\times3}$, $CT_{5\times5}$ and $CT_{9\times7}$, however, φ_1 is set to 3, 9 and 24, respectively, which is equivalent to the configuration for NCC, when considering the ratio between φ_1 and the maximum matching cost.

Appendix C. Effects of Non-Fronto-Parallel Plane Orientations in DIM

In addition to adjusting the SGM optimization to account for non-fronto-parallel surface structures, the plane orientations within the plane-sweep sampling for DIM can be adjusted in accordance with the scene structure by selecting an appropriate normal vector, and with it a corresponding sweeping direction. Thus, in the following, the use of non-fronto-parallel plane orientations within the plane-sweep sampling is investigated. In this, an additional horizontal orientation with respect to the reference coordinate system of the scene is selected and compared to the fronto-parallel sampling direction. For this, a subset of the DTU dataset, in which the camera is looking in a more downwards direction, is selected. As reference, results with a fronto-parallel sampling were computed separately. The quantitative results reveal a major increase in error when non-fronto-parallel plane orientations are used for sampling, as can be seen in the difference maps in Figure A1. Simultaneously, Figure A1 also reveals that in areas where the surface structure coincides with the sampling direction, e.g., the ground plane, the depth map is very smooth and consistent.

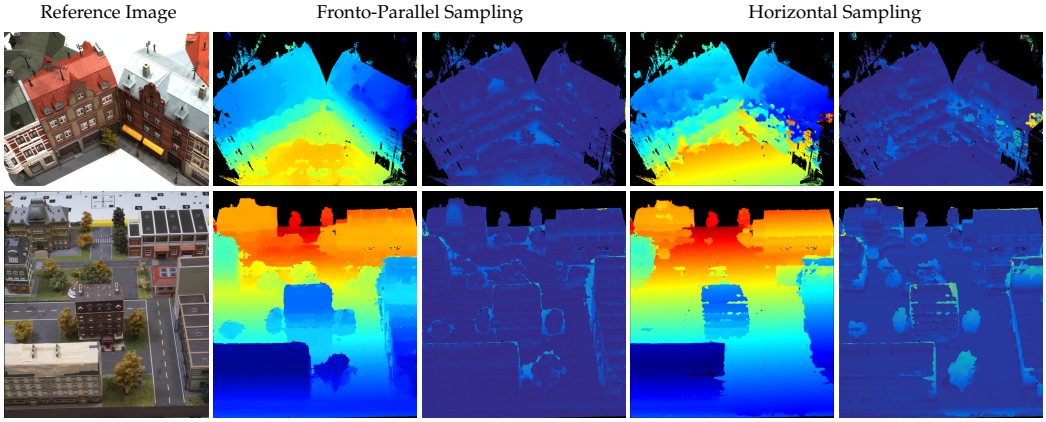

Figure A1. Qualitative comparison between the use of a fronto-parallel and non-fronto-parallel sampling direction in combination with SGM$^{\Pi}$. Columns 2 and 4: Corresponding estimated depth map. Columns 3 and 5: Difference map holding the pixel-wise absolute difference between the estimated depth map and the ground truth. The color encoding reaches from dark blue (low error) via green to yellow (high error). The estimated depth maps and the difference maps are masked according to the ground truth.

References

1. Restas, A. Drone applications for supporting disaster management. *World J. Eng. Technol.* **2015**, *3*, 316–321. [CrossRef]
2. Furutani, T.; Minami, M. Drones for disaster risk reduction and crisis response. In *Emerging Technologies for Disaster Resilience*; Springer: Singapore, 2021; pp. 51–62.
3. Schönberger, J.L.; Frahm, J.M. Structure-from-motion revisited. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Las Vegas, NV, USA, 27–30 June 2016; pp. 4104–4113.
4. Schönberger, J.L.; Zheng, E.; Frahm, J.M.; Pollefeys, M. Pixelwise view selection for unstructured multi-view stereo. In Proceedings of the European Conference on Computer Vision, Amsterdam, The Netherlands, 11–14 October 2016; pp. 501–518.
5. Wang, F.; Galliani, S.; Vogel, C.; Pollefeys, M. IterMVS: Iterative Probability Estimation for Efficient Multi-View Stereo. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, New Orleans, LA, USA, 18–24 June 2022; pp. 8606–8615.
6. Kern, A.; Bobbe, M.; Khedar, Y.; Bestmann, U. OpenREALM: Real-time Mapping for Unmanned Aerial Vehicles. In Proceedings of the International Conference on Unmanned Aircraft Systems, Athens, Greece, 1–4 September 2020; pp. 902–911.
7. Hermann, M.; Ruf, B.; Weinmann, M. Real-time dense 3D reconstruction from monocular video data captured by low-cost UAVs. *Int. Arch. Photogramm. Remote Sens. Spat. Inf. Sci.* **2021**, *XLIII-B2-2021*, 361–368. [CrossRef]
8. Cheng, S.; Xu, Z.; Zhu, S.; Li, Z.; Li, L.E.; Ramamoorthi, R.; Su, H. Deep stereo using adaptive thin volume representation with uncertainty awareness. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, Seattle, WA, USA, 13–19 June 2020; pp. 2524–2534.
9. Gu, X.; Fan, Z.; Zhu, S.; Dai, Z.; Tan, F.; Tan, P. Cascade cost volume for high-resolution multi-view stereo and stereo matching. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, Seattle, WA, USA, 13–19 June 2020; pp. 2495–2504.
10. Huang, B.; Yi, H.; Huang, C.; He, Y.; Liu, J.; Liu, X. M3VSNET: Unsupervised multi-metric multi-view stereo network. In Proceedings of the IEEE International Conference on Image Processing, Virtual Conference, 19–22 September 2021; pp. 3163–3167.
11. Ruf, B.; Erdnuess, B.; Weinmann, M. Determining plane-sweep sampling points in image space using the cross-ratio for image-based depth estimation. *Int. Arch. Photogramm. Remote Sens. Spat. Inf. Sci.* **2017**, *XLII-2/W6*, 325–332. [CrossRef]
12. Ruf, B.; Pollok, T.; Weinmann, M. Efficient surface-aware semi-global matching with multi-view plane-sweep sampling. *ISPRS Ann. Photogramm. Remote Sens. Spat. Inf. Sci.* **2019**, *IV-2/W7*, 137–144.
13. Ruf, B. Fast Dense Depth Estimation from UAV-Borne Aerial Imagery for the Assistance of Emergency Forces. Ph.D. Thesis, Karlsruher Institut für Technologie (KIT), Karlsruhe, Germany, 2022.
14. Goesele, M.; Snavely, N.; Curless, B.; Hoppe, H.; Seitz, S.M. Multi-view stereo for community photo collections. In Proceedings of the IEEE International Conference on Computer Vision, Rio De Janeiro, Brazil, 14–21 October 2007.
15. Furukawa, Y.; Ponce, J. Accurate, dense, and robust multiview stereopsis. *IEEE Trans. Pattern Anal. Mach. Intell.* **2010**, *32*, 1362–1376. [CrossRef]
16. Rothermel, M.; Wenzel, K.; Fritsch, D.; Haala, N. SURE: Photogrammetric surface reconstruction from imagery. In Proceedings of the LowCost3D Workshop, Berlin, Germany, 6–7 December 2012.
17. Wenzel, K.; Rothermel, M.; Haala, N.; Fritsch, D. SURE—The IFP software for dense image matching. In Proceedings of the Photogrammetric Week, Stuttgart, Germany, 9–13 September 2013; pp. 59–70.

18. Xu, Q.; Kong, W.; Tao, W.; Pollefeys, M. Multi-Scale Geometric Consistency Guided and Planar Prior Assisted Multi-View Stereo. *IEEE Trans. Pattern Anal. Mach. Intell.* **2022**, *45*, 4945–4963. [CrossRef]
19. Klein, G.; Murray, D. Parallel tracking and mapping for small AR workspaces. In Proceedings of the IEEE and ACM International Symposium on Mixed and Augmented Reality, Nara, Japan, 13–16 November 2007; pp. 225–234.
20. Davison, A.J.; Reid, I.D.; Molton, N.D.; Stasse, O. MonoSLAM: Real-time single camera SLAM. *IEEE Trans. Pattern Anal. Mach. Intell.* **2007**, *29*, 1052–1067. [CrossRef]
21. Eade, E.; Drummond, T. Scalable monocular SLAM. In Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition, New York, NY, USA, 17–22 June 2006; Volume 1, pp. 469–476.
22. Newcombe, R.A.; Davison, A.J. Live dense reconstruction with a single moving camera. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, San Francisco, CA, USA, 13–18 June 2010; pp. 1498–1505.
23. Newcombe, R.A.; Lovegrove, S.J.; Davison, A.J. DTAM: Dense tracking and mapping in real-time. In Proceedings of the IEEE International Conference on Computer Vision, Barcelona, Spain, 6–13 November 2011; pp. 2320–2327.
24. Gallup, D.; Frahm, J.M.; Mordohai, P.; Yang, Q.; Pollefeys, M. Real-time plane-sweeping stereo with multiple sweeping directions. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Minneapolis, MN, USA, 17–22 June 2007.
25. Pollefeys, M.; Nistér, D.; Frahm, J.M.; Akbarzadeh, A.; Mordohai, P.; Clipp, B.; Engels, C.; Gallup, D.; Kim, S.J.; Merrell, P.; et al. Detailed real-time urban 3d reconstruction from video. *Int. J. Comput. Vis.* **2008**, *78*, 143–167. [CrossRef]
26. Collins, R.T. A space-sweep approach to true multi-image matching. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, San Francisco, CA, USA, 18–20 June 1996; pp. 358–363.
27. Furukawa, Y.; Curless, B.; Seitz, S.M.; Szeliski, R. Manhattan-world stereo. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Miami, FL, USA, 20–25 June 2009; pp. 1422–1429.
28. Sinha, S.N.; Steedly, D.; Szeliski, R. Piecewise planar stereo for image-based rendering. In Proceedings of the IEEE International Conference on Computer Vision, Kyoto, Japan, 29 September–2 October 2009; pp. 1881–1888.
29. Gallup, D.; Frahm, J.M.; Pollefeys, M. Piecewise planar and non-planar stereo for urban scene reconstruction. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, San Francisco, CA, USA, 13–18 June 2010; pp. 1418–1425.
30. Zhao, Y.; Chen, L.; Zhang, X.; Xu, S.; Bu, S.; Jiang, H.; Han, P.; Li, K.; Wan, G. RTSfM: Real-Time Structure from Motion for Mosaicing and DSM Mapping of Sequential Aerial Images with Low Overlap. *IEEE Trans. Geosci. Remote Sens.* **2022**, *60*, 5607415. [CrossRef]
31. Häne, C.; Heng, L.; Lee, G.H.; Sizov, A.; Pollefeys, M. Real-time direct dense matching on fisheye images using plane-sweeping stereo. In Proceedings of the IEEE International Conference on 3D Vision, Tokyo, Japan, 8–11 December 2014; pp. 57–64.
32. Geiger, A.; Roser, M.; Urtasun, R. Efficient Large-Scale Stereo Matching. In *Computer Vision—ACCV 2010*; Springer: Berlin/Heidelberg, Germany, 2011; pp. 25–38.
33. Hirschmüller, H. Accurate and efficient stereo processing by semi-global matching and mutual information. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, San Diego, CA, USA, 20–25 June 2005; pp. 807–814.
34. Hirschmüller, H. Stereo processing by semiglobal matching and mutual information. *IEEE Trans. Pattern Anal. Mach. Intell.* **2008**, *30*, 328–341. [CrossRef]
35. Hernandez-Juarez, D.; Chacón, A.; Espinosa, A.; Vázquez, D.; Moure, J.C.; López, A.M. Embedded real-time stereo estimation via semi-global matching on the GPU. *Procedia Comput. Sci.* **2016**, *80*, 143–153. [CrossRef]
36. Ruf, B.; Mohrs, J.; Weinmann, M.; Hinz, S.; Beyerer, J. ReS^{2}tAC—UAV-borne real-time SGM stereo optimized for embedded ARM and CUDA devices. *Sensors* **2021**, *21*, 3938. [CrossRef]
37. Haala, N.; Rothermel, M.; Cavegn, S. Extracting 3D urban models from oblique aerial images. In Proceedings of the IEEE Joint Urban Remote Sensing Event, Lausanne, Switzerland, 30 March–1 April 2015.
38. Sinha, S.N.; Scharstein, D.; Szeliski, R. Efficient high-resolution stereo matching using local plane sweeps. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Columbus, OH, USA, 23–28 June 2014; pp. 1582–1589.
39. Hermann, S.; Klette, R.; Destefanis, E. Inclusion of a second-order prior into semi-global matching. In Proceedings of the Pacific-Rim Symposium on Image and Video Technology, Tokyo, Japan, 13–16 January 2009; pp. 633–644.
40. Ni, J.; Li, Q.; Liu, Y.; Zhou, Y. Second-order semi-global stereo matching algorithm based on slanted plane iterative optimization. *IEEE Access* **2018**, *6*, 61735–61747. [CrossRef]
41. Scharstein, D.; Taniai, T.; Sinha, S.N. Semi-global stereo matching with surface orientation priors. In Proceedings of the International Conference on 3D Vision, Qingdao, China, 10–12 October 2017; pp. 215–224.
42. Roth, L.; Mayer, H. Reduction of the fronto-parallel bias for wide-baseline semi-global matching. *ISPRS Ann. Photogramm. Remote Sens. Spat. Inf. Sci.* **2019**, *IV-2/W5*, 69–76. [CrossRef]
43. Zhang, Z.; Peng, R.; Hu, Y.; Wang, R. GeoMVSNet: Learning multi-view stereo with geometry perception. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, Vancouver, BC, Canada, 18–22 June 2023; pp. 21508–21518.
44. Khot, T.; Agrawal, S.; Tulsiani, S.; Mertz, C.; Lucey, S.; Hebert, M. Learning unsupervised multi-view stereopsis via robust photometric consistency. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition Workshops, Long Beach, CA, USA, 16–17 June 2019.
45. Hermann, M.; Weinmann, M.; Nex, F.; Stathopoulou, E.K.; Remondino, F.; Jutzi, B.; Ruf, B. Depth estimation and 3D reconstruction from UAV-borne imagery: Evaluation on the UseGeo dataset. *ISPRS Open J. Photogramm. Remote Sens.* **2024**, *13*, 100065. [CrossRef]
46. Wenzel, K. Dense Image Matching for Close Range Photogrammetry. Ph.D. Thesis, University of Stuttgart, Stuttgart, Germany, 2016.

47. Hartley, R.; Zisserman, A. *Multiple View Geometry in Computer Vision*; Cambridge University Press: Cambridge, UK, 2004.
48. Yao, Y.; Luo, Z.; Li, S.; Fang, T.; Quan, L. MVSNet: Depth inference for unstructured multi-view stereo. In Proceedings of the European Conference on Computer Vision, Munich, Germany, 8–14 September 2018; pp. 767–783.
49. Chen, Y.; Xu, H.; Zheng, C.; Zhuang, B.; Pollefeys, M.; Geiger, A.; Cham, T.J.; Cai, J. MVSplat: Efficient 3D Gaussian Splatting from Sparse Multi-View Images. *arXiv* **2024**, arXiv:2403.14627.
50. Kang, S.B.; Szeliski, R.; Chai, J. Handling occlusions in dense multi-view stereo. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Kauai, HI, USA, 8–14 December 2001; Volume 1, pp. 103–110.
51. Zabih, R.; Woodfill, J. Non-parametric local transforms for computing visual correspondence. In Proceedings of the European Conference on Computer Vision, Stockholm, Sweden, 2–6 May 1994; pp. 151–158.
52. Szeliski, R.; Scharstein, D. Sampling the disparity space image. *IEEE Trans. Pattern Anal. Mach. Intell.* **2004**, *26*, 419–425. [CrossRef]
53. Kolev, K.; Tanskanen, P.; Speciale, P.; Pollefeys, M. Turning mobile phones into 3D scanners. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Columbus, OH, USA, 23–28 June 2014; pp. 3946–3953.
54. Jensen, R.; Dahl, A.; Vogiatzis, G.; Tola, E.; Aanæs, H. Large scale multi-view stereopsis evaluation. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Columbus, OH, USA, 23–28 June 2014; pp. 406–413.
55. Aanæs, H.; Jensen, R.R.; Vogiatzis, G.; Tola, E.; Dahl, A.B. Large-Scale Data for Multiple-View Stereopsis. *Int. J. Comput. Vis.* **2016**, *120*, 153–168. [CrossRef]
56. Özdemir, E.; Toschi, I.; Remondino, F. A Multi-Purpose Benchmark for Photogrammetric Urban 3D Reconstruction in a Controlled Environment. *Int. Arch. Photogramm. Remote Sens. Spat. Inf. Sci.* **2019**, *XLII-1/W2*, 53–60. [CrossRef]
57. Schöps, T.; Schönberger, J.; Galliani, S.; Sattler, T.; Schindler, K.; Pollefeys, M.; Geiger, A. A multi-view stereo benchmark with high-resolution images and multi-camera videos. In Proceedings of the IEEE Conference on Computer Vision and Pattern, Honolulu, HI, USA, 21–26 July 2017; pp. 3260–3269.
58. Knapitsch, A.; Park, J.; Zhou, Q.Y.; Koltun, V. Tanks and temples: Benchmarking large-scale scene reconstruction. *ACM Trans. Graph.* **2017**, *36*, 78. [CrossRef]
59. Mehltretter, M.; Heipke, C. Aleatoric uncertainty estimation for dense stereo matching via CNN-based cost volume analysis. *ISPRS J. Photogramm. Remote Sens.* **2021**, *171*, 63–75. [CrossRef]
60. Banz, C.; Hesselbarth, S.; Flatt, H.; Blume, H.; Pirsch, P. Real-time stereo vision system using semi-global matching disparity estimation: Architecture and FPGA-implementation. In Proceedings of the International Conference on Embedded Computer Systems: Architectures, Modeling and Simulation, Samos, Greece, 19–22 July 2010; pp. 93–101.
61. DJI. Matrice 200 V2-Series—User Manual. 2020. Available online: https://dl.djicdn.com/downloads/m200_v2/20200630/M200_Series_V2_User_Manual_en4.pdf (accessed on 27 August 2024).
62. DJI. Matrice 2 Pro/Zoom—User Manual. 2020. Available online: https://dl.djicdn.com/downloads/Mavic_2/Mavic_2_Pro_Zoom_User_Manual_v2.2_en.pdf (accessed on 27 August 2024).
63. Poggi, M.; Tosi, F.; Mattoccia, S. Learning a confidence measure in the disparity domain from O(1) features. *Comput. Vis. Image Underst.* **2020**, *193*, 102905. [CrossRef]
64. Heinrich, K.; Mehltretter, M. Learning Multi-Modal Features for Dense Matching-Based Confidence Estimation. *Int. Arch. Photogramm. Remote Sens. Spat. Inf. Sci.* **2021**, *XLIII-B2-2021*, 91–99. [CrossRef]
65. Nex, F.; Rinaudo, F. LiDAR or photogrammetry? Integration is the answer. *Eur. J. Remote Sens.* **2011**, *43*, 107–121. [CrossRef]