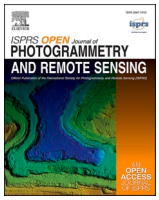
ISPRS Open Journal of Photogrammetry and Remote Sensing 13 (2024) 100065

Contents lists available at ScienceDirect

ISPRS Open Journal of Photogrammetry and Remote Sensing

journal homepage: www.journals.elsevier.com/isprs-open-journal-of-photogrammetry-and-remote-sensing



# Depth estimation and 3D reconstruction from UAV-borne imagery: Evaluation on the UseGeo dataset

M. Hermann [a,b,*], M. Weinmann [b], F. Nex [c], E.K. Stathopoulou [d], F. Remondino [d], B. Jutzi [b], B. Ruf [a]

[a] Fraunhofer IOSB, Karlsruhe, Germany
[b] Institute of Photogrammetry and Remote Sensing, Karlsruhe Institute of Technology (KIT), Karlsruhe, Germany
[c] ITC Department of Earth Observation Science, Faculty ITC, University of Twente, Enschede, the Netherlands
[d] 3D Optical Metrology (3DOM) Unit, Bruno Kessler Foundation (FBK), Trento, Italy

ARTICLE INFO

*Keywords:*
Depth estimation
3D reconstruction
UAV
UseGeo
MVS
SMDE
NeRF

ABSTRACT

Depth estimation and 3D model reconstruction from aerial imagery is an important task in photogrammetry, remote sensing, and computer vision. To compare the performance of different image-based approaches, this study presents a benchmark for UAV-based aerial imagery using the UseGeo dataset. The contributions include the release of various evaluation routines on GitHub, as well as a comprehensive comparison of baseline approaches, such as methods for offline multi-view 3D reconstruction resulting in point clouds and triangle meshes, online multi-view depth estimation, as well as single-image depth estimation using self-supervised deep learning. With the release of our evaluation routines, we aim to provide a universal protocol for the evaluation of depth estimation and 3D reconstruction methods on the UseGeo dataset. The conducted experiments and analyses show that each method excels in a different category: the depth estimation from COLMAP outperforms that of the other approaches, ACMMP achieves the lowest error and highest completeness for point clouds, while OpenMVS produces triangle meshes with the lowest error. Among the online methods for depth estimation, the approach from the Plane-Sweep Library outperforms the FaSS-MVS approach, while the latter achieves the lowest processing time. And even though the particularly challenging nature of the dataset and the small amount of training data leads to a significantly higher error in the results of the self-supervised single-image depth estimation approach, it outperforms all other approaches in terms of processing time and frame rate. In our evaluation, we have also considered modern learning-based approaches that can be used for image-based 3D reconstruction, such as NeRFs. However, due to the significantly lower quality of the resulting 3D models, we have only included a qualitative comparison between NeRF-based and conventional approaches in the scope of this work.

## 1. Introduction

Depth estimation and the reconstruction of 3D models from aerial imagery is a fundamental task in the fields of photogrammetry, remote sensing, and computer vision. Especially due to the high availability of commercial off-the-shelf (COTS) unmanned aerial vehicles (UAVs), which has greatly increased over the past years, the interest and the studies conducted in this field of research have greatly intensified. Especially the use of image-based techniques to create such models becomes increasingly popular, due to the cost-efficient and simple acquisition of image data and the recent advancements achieved in the field of photogrammetric 3D modeling. With the recently presented UseGeo dataset[1](Nex et al., 2024), a new possibility to quantitatively and qualitatively evaluate approaches for image-based depth estimation and 3D reconstruction from UAV-borne imagery is now available to the scientific community. It has been specifically acquired for a rigorous assessment of the performance of approaches for 3D reconstruction from aerial UAV-borne imagery (in terms of either single-image depth estimation or multi-view 3D reconstruction).

In this paper, we present an extensive evaluation on the UseGeo

* Corresponding author.
*E-mail addresses:* max.hermann@kit.edu (M. Hermann), martin.weinmann@kit.edu (M. Weinmann), f.nex@utwente.nl (F. Nex), estathopoulou@fbk.eu (E.K. Stathopoulou), remondino@fbk.eu (F. Remondino), boris.jutzi@kit.edu (B. Jutzi), boitumelo.ruf@iosb.fraunhofer.de (B. Ruf).

[1] https://usegeo.fbk.eu, https://github.com/3DOM-FBK/usegeo.

https://doi.org/10.1016/j.ophoto.2024.100065
Received 19 December 2023; Received in revised form 7 March 2024; Accepted 22 April 2024
Available online 4 May 2024

dataset. We provide evaluation routines for the quantitative assessment of depth maps as well as 3D point clouds and 3D meshes generated from UAV-borne imagery. Furthermore, with these evaluation routines, we investigate the performance of six different approaches for depth estimation and 3D reconstruction on the UseGeo dataset, namely COLMAP (Schönberger and Frahm, 2016; Schönberger et al., 2016), OpenMVS,[2] ACMMP (Xu et al., 2022), Plane-Sweep Library (Häne et al., 2014), FaSS-MVS (Ruf et al., 2021), and a self-supervised monocular depth estimation approach (Hermann et al., 2020). Thereby, we examine three offline multi-view stereo (MVS) approaches that allow the generation of a full 3D model, two approaches for online dense image matching (DIM) and depth estimation, as well as one learning-based approach for single-image depth estimation. Online depth estimation from UAV-borne imagery facilitates a wide variety of applications, such as mapping and monitoring (Zhang et al., 2023a; Fanta-Jende et al., 2023), precision agriculture (Botta et al., 2022), and disaster management (Ruf, 2022). In this context, we use the terminology "online" to denote fast processing without setting hard time constraints, but ideally keeping up with the frame rate of the input data. In this, the user shouldn't have to wait a couple of hours for the processing to finish but instead would get a first impression of the results right away. In contrast, "offline" MVS processing is independent with respect to the actual acquisition of the input data. Such approaches have full access to the available input data and are focused on the accuracy and completeness of the resulting product, rather than on fast execution and quick availability of the results. Detailed and accurate 3D point clouds or meshed models are of increasing importance for a large number of applications. Prominent examples of such applications are architecture and civil engineering (Tsoraeva et al., 2021), environmental monitoring (Bayomi and Fernandez, 2023), preserving cultural heritage (Pepe et al., 2022) as well as disaster relief and mission planning (Kerle et al., 2020; Furutani and Minami, 2021).

In summary, our contributions are as follows:

- We evaluate the performance of approaches for 3D reconstruction and image-based depth estimation from UAV-borne aerial imagery on the recently presented UseGeo dataset.
- We provide a comprehensive comparison of baseline approaches evaluated on the UseGeo dataset. This includes approaches for offline multi-view 3D reconstruction and online multi-view depth estimation, as well as approaches for single-image depth estimation and modern learning-based approaches that can be used for image-based 3D reconstruction, such as NeRFs.
- We release the routines used to evaluate the quality of derived depth maps, 3D point clouds, and 3D meshes on GitHub.[3] This aims to provide a universal protocol for the evaluation of depth estimation and 3D reconstruction methods on the UseGeo dataset.

This paper is organized as follows. In Section 2, we summarize related work. In this regard, we briefly revisit related work on 3D reconstruction with offline multi-view stereo approaches and subsequently address recent progress regarding the tasks of online dense image matching and respective depth estimation as well as single-image depth estimation. In Section 3, we provide details about the studied approaches for offline and online 3D reconstruction as well as for single-image depth estimation. In Section 4, we focus on the evaluation conducted on the basis of depth maps, point clouds, and triangle meshes, and we describe the used evaluation metrics. In Section 5, we provide a comprehensive comparison of results achieved by the applied approaches on the UseGeo dataset. In Section 6, we discuss the achieved results in detail. Finally, in Section 7, we provide concluding remarks as well as suggestions for future work.

## 2. Related work

In the following section, we briefly discuss related work in the area of depth estimation and 3D reconstruction. For this purpose, we will first summarize approaches regarding offline 3D reconstruction in Section 2.1, followed by methods for online depth estimation in Section 2.2. Finally, in Section 2.3, we present related work regarding methods for depth estimation based on deep learning.

### 2.1. 3D model generation with offline multi-view stereo

The software suite *PhotoTourism* (Snavely et al., 2006), later known as *Bundler*, was one of the first open-source software toolkits to perform an image-based 3D reconstruction and point cloud generation from a set of internet photos without preliminary knowledge of the scene or camera geometry. While it was first restricted to a rather small-scaled scene, like a single building, it was further extended and demonstrated by Agarwal et al. (2011) that the approach can also be used for large-scale city reconstruction. Despite the impressive results, these approaches were not able to create a fully dense reconstruction of the scene as they only matched and triangulated point image features between the input images. With *VisualSfM* (Wu, 2011, 2013), an easy-to-use software suite was published that, as one of the first, integrated an end-to-end processing pipeline to generate dense 3D point clouds directly from image data by means of MVS. In this, it integrated the dense multi-image matching approach *PMVS/CMVS* (Furukawa and Ponce, 2009; Furukawa et al., 2010) to densify the point cloud from the triangulation of the point image features.

Since these early achievements, a variety of structure-from-motion (SFM) and MVS approaches and libraries have been released, each targeting different aspects of the full processing pipeline. Prominent examples are *OpenMVG* (Moulon et al., 2016) and *OpenSFM* (Adorjan, 2016) for the task of structure-from-motion, as well as *MVE* (Fuhrmann et al., 2015), *OpenMVS*, *Gipuma* (Galliani et al., 2015) and *ACMMP* (Xu et al., 2022) for the task of multi-view stereo. These approaches and libraries typically provide interoperability with each other or can be used with the aforementioned toolkits to build a custom processing pipeline. The most recent, all-in-one open-source software suite that implements a full end-to-end SFM and MVS pipeline is *COLMAP* (Schönberger and Frahm, 2016; Schönberger et al., 2016). It allows the computation of highly accurate 3D point clouds and meshes directly from image data. Furthermore, it is actively maintained and extensively used by the photogrammetry and computer vision community. Since it is also equipped with a graphical user interface and an automatic reconstruction pipeline, it is suitable for both expert and non-expert users.

In the scope of this study, we evaluate the accuracy of the results achieved with COLMAP due to its popularity and extensive use. We also evaluate the accuracy of the meshes computed by OpenMVS, since these meshes are optimized with respect to the number of triangles and, thus, more practical to use. Lastly, we also evaluate the depth maps and point clouds produced by ACMMP, since this approach aims at estimating more complete depth maps and, in turn, point clouds with higher point density.

### 2.2. Online dense image matching and depth estimation

The approaches presented by Gallup et al. (2007) and Pollefeys et al. (2008) are part of the first scientific work that tackles the task of online camera-based mapping of urban surroundings. In this, they use images captured from vehicle-mounted cameras to reconstruct building facades, while the vehicle is moving through the streets. In order to allow for online depth estimation, they employ the plane-sweep algorithm (Collins, 1996) for dense multi-image matching and optimize it for concurrent execution on GPU hardware. And since most objects in urban

[2] https://github.com/cdcseacave/openMVS.

[3] https://github.com/UseGeoEvaluation/DepthEstimationAnd3DReconstruction.

scenery can be approximated well by planar structures, DIM based on the plane-sweep algorithm is a well-suited approach for this task. This advantage is also exploited by other approaches, such as those from Furukawa and Ponce (2009), Sinha et al. (2009), or Gallup et al. (2010), which fit multiple, differently orientated planes into the scene to implement a piece-wise planar reconstruction of urban scenes. The best composition of the planes is then found by minimizing an energy function and optimizing photometric consistency.

Initially intended for two-view stereo depth estimation, the so-called Semi-Global Matching (SGM) algorithm was proposed by Hirschmueller (2005, 2008). Due to its proven efficiency and convincing results, it quickly evolved into one of the most widely used approaches for real-time, online, and offline DIM for both two-view (Hernandez-Juarez et al., 2016; Spangenberg et al., 2014; Zhao et al., 2020) and multi-view stereo (Rothermel et al., 2012; Sinha et al., 2014; Haala et al., 2015).

More recently, approaches as those presented by Kern et al. (2020), Hermann et al. (2021) and Zhao et al. (2022) present full processing pipelines for online 3D mapping from aerial imagery. These pipelines generally consist of a tracking and camera pose estimation step, followed by dense depth estimation from a bundle of selected input images, as well as a depth map fusion and a number of post-processing steps. The approaches of Kern et al. (2020) and Hermann et al. (2021) estimate dense depth maps by performing dense multi-image matching using *PlaneSweepLib* (Häne et al., 2014) and *FaSS-MVS* (Ruf et al., 2021), respectively. Both are part of the evaluation in the scope of this study. *RTSfM* (Zhao et al., 2022) on the other hand, rather focuses on an efficient and globally consistent SFM and pose estimation in real-time. For the estimation of dense depth maps, the two-view stereo approach ELAS (Geiger et al., 2011) is executed on pairs of images. However, since the estimation of dense depth maps from aerial imagery is not constrained to only using two views for the depth estimation, unless one is working with an actual two-view stereo camera, a MVS approach seems more appropriate. This is because such approaches rely on three or more input images for the DIM which, in turn, can increase the reliability of the estimates and allow for more complete depth maps due to a higher number of observations and vantage points, resulting in less occluded image areas.

### 2.3. Learning of image-based depth estimation

Due to the advancements in the field of deep learning and the results achieved by deep convolutional neural networks (CNNs) for computer vision and photogrammetric tasks, the use of learning-based approaches for DIM and depth estimation has also thrived in recent years. First approaches (Häne et al., 2014; Hartmann et al., 2017; Zbontar and LeCun, 2016) aim at learning a similarity score between image patches with the help of CNNs and use this to compute disparity hypotheses from which depth maps can be extracted by means of conventional optimization strategies, such as SGM. Succeeding approaches, such as MVSNet (Yao et al., 2018) and DeepMVS (Huang et al., 2018) build upon these insights of learning a similarity score from multi-image matching. But instead of using conventional approaches to regularize the cost-volume, they both aim at establishing an end-to-end learning-based approach and thus rely on the U-Net (Ronneberger et al., 2015) to extract the resulting depth map. Despite their convincing results, a great disadvantage of such approaches is their need for ground truth data for supervised training. Approaches aiming at an unsupervised training of such models (Khot et al., 2019; Huang et al., 2021), try to overcome this problem by using the projection error between different viewpoints as a training signal. Although their results are typically superior to those achieved by conventional approaches, their ability for generalization and, in turn, their practical use in partially unknown environments still need to be proven.

A great advantage of learning-based approaches for image-based depth estimation over approaches that rely on conventional DIM, however, is their ability to also learn image cues based on the input data, which are typically difficult to model. Similar to the empirical knowledge of human beings, it is assumed that approaches relying on deep CNNs are able to learn how to predict relative scene depth even from a single image. This constitutes a major advantage over two-view or multi-view stereo approaches since it would lead to reduced latency, allow the prediction of depth for dynamic scenes, and remove the constraining relationship between baseline and maximum depth. Thus, great effort has recently been put into studying approaches for single-image depth estimation. In particular, self-supervised approaches are of great interest, as they do not rely on special training data but can learn to predict depth from solely a tuple of input images by formulating the training task as a novel view synthesis and image reconstruction problem.

First approaches, such as those presented by Flynn et al. (2016) and Xie et al. (2016) aim at synthesizing images from new viewpoints. In this, the model learns how to predict the depth from this new vantage point in order to correctly sample new image data. Focusing on the prediction of depth and considering the novel-view-synthesis as an intermediate stage, this methodology can be utilized to learn how to predict depth from a single view, as proposed by Godard et al. (2017). By relying on a stereo camera setup during training, the approach of Godard et al. (2017) does not require predicting the extrinsic transformation between the input images but can use the data of a preliminary calibration. Succeeding approaches like (Mahjourian et al., 2018; Wang et al., 2018; Zhao et al., 2016; Godard et al., 2019) make use of the monocular MVS, using images of a single moving camera during training. However, this also requires learning how to predict the relative transformation between the input images. This is usually implemented using a neural network, which estimates the camera movement in six degrees of freedom using the images as input. While the aforementioned architectures are mostly based on conventional convolution layers, there are an increasing number of approaches that rely on vision transformers (Dosovitskiy et al., 2021) as backbone to increase performance under certain circumstances (Ranftl et al., 2021) or alternatively to reduce the model size (Zhang et al., 2023b).

As one of the first, Knöbelreiter et al. (2018) demonstrate the feasibility of using a self-supervised learning approach for two-view stereo reconstruction from aerial imagery, which is significantly different from images of street scenes as used by the previous approaches, due to the higher degrees of freedom in the camera movement. The approaches of Madhuanand et al. (2021) and Hermann et al. (2020) aim at predicting the depth from a single moving camera attached to a COTS UAV. In contrast to the approach of Hermann et al. (2020), which is evaluated in the scope of this study, Madhuanand et al. (2021) rely on a 2D CNN encoder and a 3D CNN decoder and further introduce two additional loss functions.

Similar to the self-supervised learning approaches for depth estimation, NeRF-based approaches learn the representation of a scene by synthesizing new images. By optimizing a continuous volumetric scene representation, Mildenhall et al. (2021) predict the color and density for a given coordinate and viewing direction in 3D space using a fully connected neural network. In order to obtain a discrete point cloud, this latent space can be systematically sampled, whereby all points whose density lies above a threshold value are selected as valid points. In order to be able to reconstruct large scenes and to be able to parallelize the reconstruction, Turki et al. (2022) divide the scene into sub-modules by clustering the camera poses and representing them by individual NeRF models. Mip-NeRF (Barron et al., 2021) extends the NeRF concept by using conical frustums instead of rays to reduce aliasing artifacts, which increases accuracy and speed. Tancik et al. (2022) also divide the scene into sub-modules utilizing the Mip-NeRF approach for the individual models and adding a visibility check for higher consistency. Instead of relying on a continuous space, Sun et al. (2022) use discrete voxel grids and thus achieves higher speed with comparable quality.

## 3. Evaluated methods

In this chapter, different approaches for offline depth estimation, as well as point cloud and mesh generation (Sections 3.1 to 3.3) are presented. Furthermore, methods for online depth estimation (Sections 3.4 and 3.5) and a self-supervised monocular approach for depth estimation from a single image (Section 3.6) are introduced.

### 3.1. COLMAP

To the best of our knowledge, *COLMAP*[4] (Schönberger and Frahm, 2016; Schönberger et al., 2016) is one of the few open-source software suites that implements a full end-to-end SFM and MVS pipeline, while there are a couple of methods that cover parts of the pipeline. For camera pose estimation and sparse scene modeling within the SFM part of the processing pipeline, COLMAP first extracts Scale-Invariant Feature Transform (SIFT) image features (Lowe, 2004). These features are then matched against each other to establish correspondences between the individual images. In this, the user can choose from a number of different matching techniques, e.g. exhaustive, sequential, or vocabulary tree matching. From these correspondences, a scene graph is created enforcing geometric consistency between the matched image features. The estimation of the camera poses and the simultaneous sparse scene reconstruction is then realized by an incremental SFM approach implementing a next-best view selection followed by multi-view point triangulation for image resection. To avoid drifting of the model and to improve the camera pose estimation, a global bundle adjustment is employed every few iterations using the CERES[5] solver. In addition to the estimation of extrinsic camera parameters during the sparse reconstruction, COLMAP also allows for the simultaneous estimation of intrinsic camera parameters. For this, different models for focal length, principal point, and distortion parameters are supported.

After the estimation of the camera poses and the sparse scene representation, a dense point cloud can be estimated using an offline MVS pipeline. In this, a probabilistic patch-match stereo approach is used, which extends the approach presented by Zheng et al. (2014) to jointly estimate the per-pixel depth and surface normals in order to avoid stair-casing artifacts on oblique structures. Generally, MVS approaches based on patch-match stereo create surface hypotheses for each pixel by performing dense multi-image matching based on a plane-induced homography with respect to individual 3D plane patches, from which depth estimates are recovered. Schönberger et al. (2016) further improve the depth and normal estimation by employing a number of different priors, i.e. triangulation, resolution, and incident prior, to select the best views for MVS. In the estimation of the pixel-wise depth and normal information, a photometric consistency based on a bilaterally weighted normalized cross correlation (NCC) similarity score, as well as a geometric consistency is implemented. In a final step, the geometric consistent depth maps are fused into a dense point cloud using visibility constraints to handle occluded or duplicated areas. Since the points of the fused point clouds are enhanced with a normal vector, the point cloud can directly be meshed into a surface model using the Poisson reconstruction algorithm proposed by Kazhdan and Hoppe (2013).

### 3.2. OpenMVS

While the MVS Pipeline of COLMAP contains the functionality to create a dense mesh from the reconstructed point cloud, it is not a main area of focus of the framework and is not really suitable for large-scale reconstructions. This is because the Poisson meshing inside COLMAP directly computes a meshed model from the dense point cloud, not incorporating a mesh decimation step which, in turn, leads to large models with respect to the required memory resources, due to a large number of surface triangles. Furthermore, only the vertices are colored by COLMAP, which is why the color resolution depends on the number of vertices. Texturing of the triangles, on the other hand, is not performed. The *OpenMVS*[6] library aims to fill this gap by providing a set of algorithms to reconstruct a dense point cloud from a set of input images with corresponding camera poses by means of MVS. With this, it can recover a detailed and yet useable meshed surface model by minimizing the number of surface triangles, while at the same time preserving structural details. In addition, the created triangle meshes can be textured with high-resolution imagery.

The estimation of depth maps by means of MVS is based on the approach presented by Shen (2013). This approach is also based on patch-match stereo matching to generate the depth maps with a subsequent refinement process to enforce consistency between multiple views. Again, as a similarity score to recover the optimal surface patches, the NCC is used. To generate a surface mesh from the dense point cloud, OpenMVS implements a Delaunay triangulation method, followed by mesh decimation and refinement to fuse neighboring triangles in areas with less structural details while simultaneously enhancing fine structures. In a subsequent step, the meshed model is further refined with the provided images by calculating the error for the reprojection between neighboring images. OpenMVS further provides functionalities to generate an accurate and high-resolution mesh texture from the input images in order to colorize the resulting surface model. Due to the computed mesh texture and the varying level-of-detail, the resulting surface model is accurate, yet visually appealing.

As already mentioned, OpenMVS only provides MVS functionalities and, thus, requires the input images to be enhanced with projection matrices $\mathbf{P} = \mathbf{K}[\mathbf{R}|\mathrm{t}]$, consisting of the intrinsic camera matrix $\mathbf{K}$ and the extrinsic camera pose $[\mathbf{R}|\mathrm{t}]$. In this, the input images are to be free of lens distortion. While this data could be recovered by a variety of open-source SFM libraries or toolboxes, in the scope of this study we use the intrinsic and extrinsic camera calibration provided with the UseGeo dataset.

### 3.3. ACMMP

Inherent to all MVS techniques, including COLMAP and OpenMVS, that rely on a photometric consistency during dense image matching, is the difficulty in estimating reliable correspondences in low-textured image regions. With ACMMP,[7] Xu et al. (2022) aim at tackling this difficulty by proposing a multi-scale geometric consistency guided and planar prior assisted MVS. It combines ACMH, ACMM, and ACMP (Xu and Tao, 2019, 2020) into a single framework.

ACMMP also employs patch-match stereo matching for accurate depth map estimation. However, instead of relying on a probabilistic view selection strategy, as done by COLMAP, ACMMP implements an adaptive checkerboard sampling and multi-hypothesis view selection (ACMH). This is supposed to increase efficiency by increasing the parallelism, while at the same time improving accuracy by adaptively increasing the support region. A multi-scale processing is introduced as part of ACMM to reduce ambiguities in low-textured areas. In this, the estimates from the coarser scales are propagated to finer scales to constrain the local depth estimation. Since this coarse-to-fine propagation often results in a loss of detail on finer scales, difference maps between adjacent scales with respect to the photometric consistency are used to restore details. Lastly, with ACMP a planar prior is used to assist the patch-match MVS to further increase the effectiveness to also compute depth estimates in low-textured image regions. Just as COLMAP, ACMMP also uses a bilaterally weighted NCC as a similarity

[4] https://github.com/colmap/colmap.
[5] https://github.com/ceres-solver/ceres-solver.
[6] https://github.com/cdcseacave/openMVS.
[7] https://github.com/GhiXu/ACMMP.

measure in the process of dense image matching.

### 3.4. Plane-Sweep Library

The use of the so-called plane-sweep algorithm is a prominent approach for online depth estimation based on MVS (Gallup et al., 2007; Pollefeys et al., 2008; Sinha et al., 2014). First proposed by Collins (1996), it allows for true multi-image matching by warping an arbitrary number of matching images $\mathscr{I}_k$ into the view of a reference camera, via a set of imaginary scene planes, and matching them against the reference image $\mathscr{I}_{\text{ref}}$. In this, all images are assumed to be free of lens distortion. If the scene plane, which is used for the image warping, is close to an actual geometry, the corresponding image regions between $\mathscr{I}_{\text{ref}}$ and the warped matching image $\mathscr{I}'_k$ will match, yielding high photometric similarity. Thus, with the plane-sweep algorithm, the scene is sampled with a set of imaginary planes implicitly providing depth hypotheses from which the final scene depth can be reconstructed.

The scene planes are parameterized by their orientation in the form of a normal vector n, as well as their distance $d$ from the reference camera along their normal vector. Together with the intrinsic camera matrices $\mathbf{K}_k$ and the extrinsic relative poses $[\mathbf{R}|\mathbf{t}]$ between the matching images and the reference image, the plane-induced homography $\mathbf{H}$ can be computed according to Equation (1) with which the matching images $\mathscr{I}_k$ are warped into the view of the $\mathscr{I}_{\text{ref}}$.

$$\mathbf{H} = \mathbf{K}_{\text{ref}} \cdot \frac{\mathbf{R} - \mathbf{t} \cdot \mathbf{n}^\intercal}{d} \cdot \mathbf{K}_k^{-1} \tag{1}$$

From a selected set of depth hypotheses which should make up the final depth map $\mathscr{D}$, the pixel-wise scene depth can be reconstructed by doing a ray cast for each pixel $\mathrm{p} = (u\ v)^\intercal$ and intersecting the viewing ray with the corresponding scene plane according to:

$$\mathscr{D}(\mathrm{p}) = \frac{-d}{\mathbf{n}^\intercal \cdot \mathbf{K}_{\text{ref}}^{-1} \cdot (u\ v\ 1)^\intercal}. \tag{2}$$

The warping of images via a plane-induced homography can be efficiently implemented using the projective texture mapping functionality of graphic processing units (GPUs) and together with the capabilities of modern hardware, the plane-sweep algorithm can be executed with real-time processing rates and is, thus, well suited for online MVS depth estimation.

The *PlaneSweepLib*[8] provides an open-source implementation of the plane-sweep algorithm for online depth estimation using MVS. While the presented study only considers images from a camera with a standard pinhole camera model, the Plane-Sweep Library (PSL) also allows performing depth estimation using images from a fisheye camera. The PSL implementation of the plane-sweep algorithm follows the description given by Gallup et al. (2007). In this, they use a bundle of five input images $\mathscr{I}$ and corresponding camera projection matrices $\mathbf{P}$ as input to the algorithm. The sampling planes are parameterized with multiple sweeping directions, i.e. multiple sets of plane families with the same normal vector so that they are similar to the prominent planes of the scene which is to be reconstructed. The distances of the individual planes from the reference camera are selected in such a way that the maximum disparity shift between two consecutive planes is less or equal to one pixel. As a similarity measure and matching cost, the PSL implements the sum of absolute differences (SAD) as well as the NCC, with the latter one being used in the scope of this study. In their work, Gallup et al. (2007) and Häne et al. (2014) assume that the input images are part of an image sequence with a dominant motion direction. Thus, in order to account for occlusions, the PSL relies on the approach of Kang et al. (2001) which splits the input images into two disjoint sets to the left and right of the $\mathscr{I}_{\text{ref}}$ and separately performs the dense image matching in each of the two sets. From the plane hypotheses, a final depth map is extracted using a graph-cut optimization method.

[8] https://www.cvg.ethz.ch/research/planeSweepLib.

### 3.5. FaSS-MVS – fast multi-view stereo with Surface-Aware Semi-Global Matching

As illustrated by Fig. 1, the approach for fast multi-view stereo with surface-aware Semi-Global Matching (*FaSS-MVS*) (Ruf et al., 2021) uses an input bundle $\Omega$, consisting of $k$ input images $\mathscr{I}$ and corresponding full cameras projection matrices $\mathbf{P}$, to compute a set of depth, normal and confidence maps $(\mathscr{D}, \mathscr{N}, \mathscr{C})$ for a selected reference image, typically the middle one inside $\Omega$. Again, the input images are to be free of lens distortion and should be provided in sequential order, depicting the scene from slightly different viewpoints. $\mathbf{P}_k = \mathbf{K}[\mathbf{R}|\mathbf{t}]_k\ [\mathbf{R}|\mathbf{t}]$

In order to allow for online processing, the computational complexity is reduced by employing a hierarchical processing scheme. This allows to restrict the search space for depth hypotheses and with it the overall run-time, in particular for scenes with a large depth range, e. g. oblique aerial imagery. To do so, first, a Gaussian image pyramid is created for each input image, halving the image size in both image

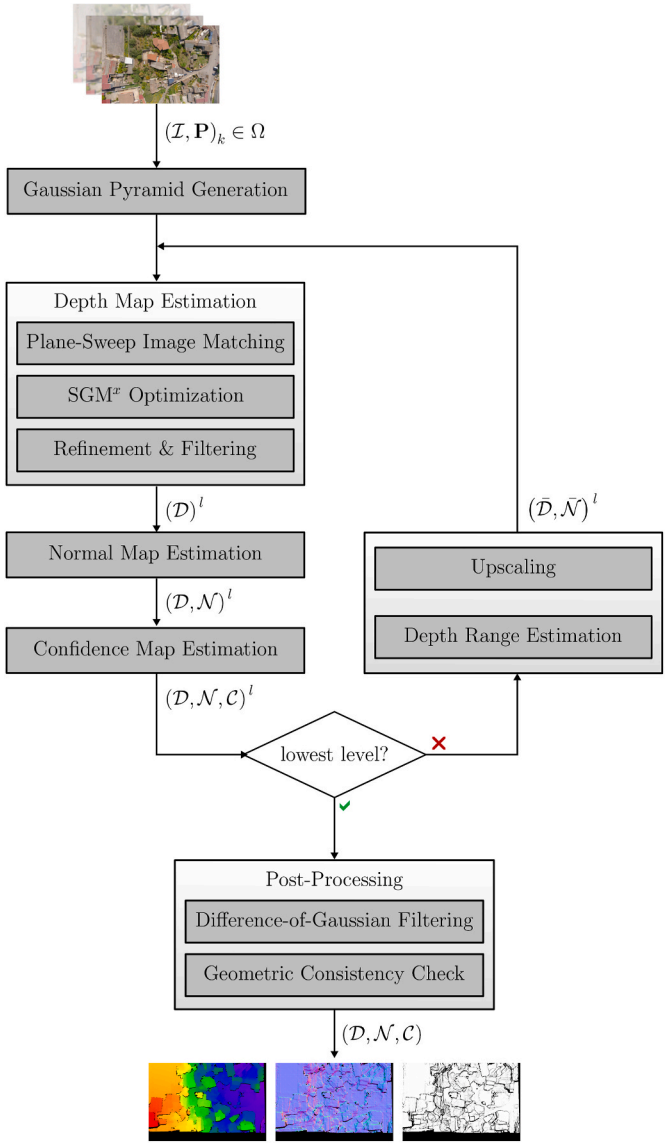



**Fig. 1.** Overview of the processing pipeline for Fast Multi-View Stereo with Surface-Aware Semi-Global Matching (FaSS-MVS). Adapted from (Ruf et al., 2021).

dimensions on each pyramid level with respect to the previous level. Starting off on the highest pyramid level, i.e. the one with the smallest resolution and, thus, coarsest image, the algorithm is initialized to sample the complete depth range for depth hypotheses. In the subsequent stages, the sampling of the scene space is initialized with the results of the previous pyramid level by upscaling the depth and normal maps and recovering a pixel-wise local depth range around the available depth estimated. This results in coarse-to-fine processing since the sizes of the depth sampling steps are determined based on the disparity shift in image space and are, thus, increasing when moving down the pyramid.

At each pyramid level, the scaled input images and camera projection matrices are used to first estimate a depth map $\mathscr{D}$ for the selected reference image $\mathscr{I}_{\text{ref}}$. In this, a plane-sweep algorithm for multi-image matching, similar to that implemented in the PSL (cf. Section 3.4), is used to generate depth hypotheses. As also done by the PSL, a separate image matching is done for the left and right subset of the input images with respect to $\mathscr{I}_{\text{ref}}$ in order to account for occlusions. This is followed by an estimation of a single depth map using an adaptation of the Semi-Global Matching (SGM) algorithm (Hirschmueller, 2005, 2008) to account for the plane-wise depth hypotheses generated by the plane-sweep algorithm. A final depth refinement and filtering step with a $5 \times 5$ pixels sized median kernel removes remaining outliers and increases the smoothness of the depth map.

The depth estimation is then followed by a normal and confidence map estimation. The normal map $\mathscr{N}$ is computed based on the depth estimates given in the depth map $\mathscr{D}$, by considering the surface gradient within a local neighborhood. With a subsequent appearance-based weighted Gaussian smoothing, a more extensive consistency and smoothness within the normal map are enforced. Given the normal map $\mathscr{N}$ and the depth map $\mathscr{D}$, the confidence map $\mathscr{C}$ is computed based on the pixel-wise incidence angle of the viewing ray on the corresponding surface.

When the bottom of the image pyramid, and with it, the original image size is reached, a final post-processing step is performed to further remove the remaining outliers. This includes Difference-of-Gaussian (DoG) filtering which unmasks non-distinctive image regions with little to no texture, i.e. homogeneous image regions, that lead to unreliable image matching. Assuming that the input images belong to an image sequence, a geometric consistency check with respect to previously computed depth maps of the same object or area can be performed to increase the overall consistency between consecutive depth maps.

In the scope of this benchmark evaluation, we have used the NCC with a support window of $9 \times 9$ pixels. In the SGM optimization we have relied on the SGM$^{\Pi}$ with a $P1 = 80$ and an adaptive $P2$ based on the image-gradient. Even though FaSS-MVS also estimates normal and confidence maps, only the accuracy of the depth maps are evaluated in the scope of this study.

### 3.6. Depth estimation from a single image by self-supervision

Due to the technological advancements achieved in the field of deep learning in recent years, the use and extensive study of learning-based approaches with deep CNNs have immensely increased in all fields of computer vision and photogrammetry. The use of deep CNNs to predict a depth map from a single input image is of particular interest since it would remedy a number of disadvantages of depth estimation by two-view or multi-view stereo. For example, depth estimation from a single image allows for predicting the depth of dynamic scenes without being subjected to a maximum depth range due to the restrictions in the stereo baseline. With respect to MVS from images captured by a single camera, for example, the depth estimation from a single image has sufficiently lower latency, since it does not require an ego-motion of the camera. The process of predicting a depth map from a single image is referred to as *monocular* or *single-view* depth estimation.

In the scope of this work, we evaluate an approach for self-supervised monocular depth estimation (SMDE), namely the one proposed by Hermann et al. (2020)[9] Learning depth estimation in a self-supervised manner means that, during the training process, the task of depth estimation is posed as a view synthesis and image reconstruction problem. This allows the approach, unlike approaches based on supervised learning, to not be dependent on ground truth depth maps for learning how to predict the scene depth. In contrast, it only requires a bundle of input images depicting the scene from slightly different vantage points. Similar to the approaches presented in Sections 3.4 and 3.5, the network learns to predict the depth for a reference image $\mathscr{I}_{\text{ref}}$ by synthesizing a new image for the same view based on the neighboring images in the input bundle and comparing it with the actual image. When the synthesized image $\mathscr{I}'$ and the actual image $\mathscr{I}_{\text{ref}}$ are most similar with respect to their matching costs, it is assumed that the network has learned to correctly predict the depth. Because the scene depth, together with the relative transformation between the images, is required to correctly synthesize a new view.

Fig. 2 illustrates the steps executed in each training iteration and for each image bundle. i) First, an encoder-decoder network is used to predict a depth map $\mathscr{D}$ corresponding to $\mathscr{I}_{\text{ref}}$. ii) In the second step, the relative transformations $\mathbf{E}_{\text{ref}\to k} = [\mathbf{R}|\mathbf{t}]$ between the reference image and the matching images are predicted. iii) With the predicted depth map $\mathscr{D}$ and the relative transformations $\mathbf{E}_{\text{ref}\to k}$, a synthetic reference image is sampled from the matching images in step three, using a spatial transformer network (STN) (Jaderberg et al., 2015). iv) Lastly, the training loss, comprised of a similarity and a smoothness score, is computed by comparing the synthetic reference image to the actual $\mathscr{I}_{\text{ref}}$ and is backpropagated through the CNN. At the beginning of the training, the predictions of $\mathscr{D}$ and $\mathbf{E}_{\text{ref}\to k}$ will be of low quality. But with more and more iterations, the network will learn appropriate image cues to correctly synthesize a visually correct appearing reference image from the matching images and with it correct depth maps and relative transformations.

Other than described by Hermann et al. (2020), we do not share the weights between the encoders of both networks. We also pre-train the camera movement estimation network on the WildUAV dataset (Florea et al., 2021). Both help a lot in the convergence of the whole network. To make the best use of the limited data, we heavily rely on data augmentation. At this point, the high resolution of the images helps, as random cropping and resizing can be used to create many variants of the original images. As backbone for the pose estimation network, we use the ResNet18 architecture (He et al., 2016) in all experiments. For the depth estimation backbone, we investigate ResNet18, ResNet50 and PackNet (Guizilini et al., 2020). We use ADAM as optimizer with the standard parameterization, a learning rate of 0.0001, and a batch size of 5 for all experiments.

## 4. Evaluation

The evaluation of the methods described in the previous sections is divided into the evaluation of depth maps, point clouds, and triangle meshes. To ensure reproducibility, the code used for this evaluation is available on GitHub.[10] In the following, the metrics and evaluation routines are briefly described.

### 4.1. UseGeo dataset

The UseGeo dataset (Nex et al., 2024; ISPRS Scientific Initiative UseGeo, 2023a, 2023b) has been acquired in the scope of one of the ISPRS Scientific Initiatives 2021–2022 to specifically allow for a rigorous assessment of the performance of approaches for 3D

[9] https://github.com/Max-Hermann/SelfSupervisedAerialDepthEstimator.
[10] https://github.com/UseGeoEvaluation/DepthEstimationAnd3DReconstruction.

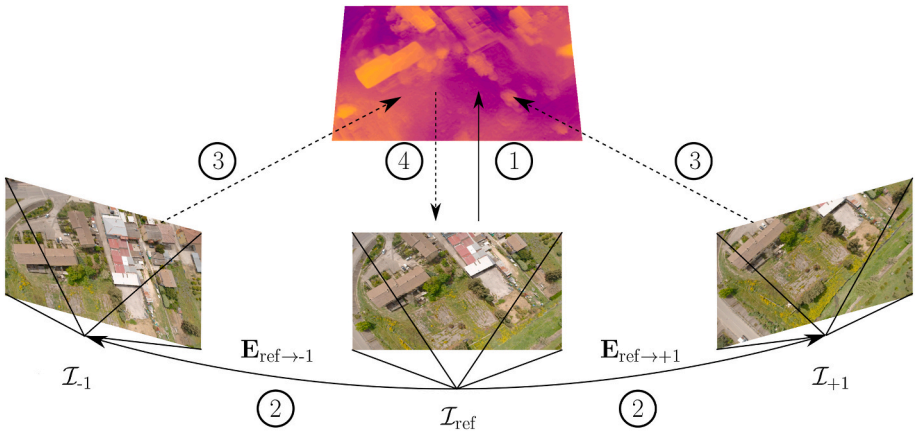



**Fig. 2.** Illustration of the self-supervised training process for the task of single-view depth estimation. Adapted from (Hermann et al., 2020).

reconstruction from imagery (in terms of either single image depth estimation or multi-view 3D reconstruction).

More specifically, the data was acquired with a UAV equipped with a SONY ILCE-7RM3 camera and a RIEGL miniVUX-3UAV scanner. Flights were performed for three different urban and peri-urban areas to address scenarios of different complexity, and a total of 829 images (with 7952 × 5304 pixels) were collected, whereby the average height above ground was about 80 m. Thus, the ground sampling distance of acquired imagery amounts to approximately 2 cm, and the laser point cloud has a point density of about 51 pt/m$^2$, in some areas even higher. For details on the data preprocessing, we refer to (Nex et al., 2024). The laser point cloud may serve as a reference for evaluating the quality of derived 3D reconstructions.

The UseGeo dataset presents a particular challenge for learning-based methods, as a total of only 829 images with nadir perspectives are available. Even though they cover a diverse scene with high resolution, the overlap between the individual images is relatively small. This is problematic because for example the self-supervised method used here performs both depth estimation and estimation of relative camera movement at the same time. For a reasonable result, both parts of the network have to converge, which is especially difficult for a camera pose estimation network.

In the following, the data acquired for each flight are referred to as *Dataset-1*, *Dataset-2* and *Dataset-3*, respectively.

### 4.2. Evaluation of depth maps

To evaluate the depth maps derived with the methods presented in Section 3, the supplied ground truth is used, which was generated by projecting the points of the LiDAR point cloud into the image plane. However, since the provided ground truth is in the form of range maps, we first transform these into depth maps by converting the range values into orthogonal distance values. Methods that do not use the ground truth extrinsic camera parameters, such as the self-supervised approach, do not provide depth maps that have the same scaling as the ground truth. In this case, they are aligned using median scaling, where the estimated depth map is scaled by the median of the depth values in the ground truth. As metrics, we use both absolute and relative metrics, and based on these, the L1 error as well as accuracy (Acc) and completeness (Cpl). The latter two help to investigate the trade-off between sparse but accurate depth maps on the one hand, and more complete depth maps, with potentially higher error, on the other hand.

$$\text{L1}-\text{abs}\left(\mathscr{D}_{est},\mathscr{D}_{gt}\right)=\frac{1}{|\mathscr{V}|}\sum_{\text{p}\in\mathscr{V}}|\mathscr{D}_{est}(\text{p})-\mathscr{D}_{gt}(\text{p})| \tag{3}$$

$$\text{L1}-\text{rel}\left(\mathscr{D}_{est},\mathscr{D}_{gt}\right)=\frac{1}{|\mathscr{V}|}\sum_{\text{p}\in\mathscr{V}}\frac{|\mathscr{D}_{est}(\text{p})-\mathscr{D}_{gt}(\text{p})|}{\mathscr{D}_{gt}(\text{p})} \tag{4}$$

$$\text{Acc}_{\theta}\left(\mathscr{D}_{est},\mathscr{D}_{gt}\right)=\frac{1}{|\mathscr{E}|}\sum_{\text{p}\in\mathscr{V}}\max\left(\frac{\mathscr{D}_{est}(\text{p})}{\mathscr{D}_{gt}(\text{p})},\frac{\mathscr{D}_{gt}(\text{p})}{\mathscr{D}_{est}(\text{p})}\right)<\theta \tag{5}$$

$$\text{Cpl}_{\theta}\left(\mathscr{D}_{est},\mathscr{D}_{gt}\right)=\frac{1}{|\mathscr{G}|}\sum_{\text{p}\in\mathscr{V}}\max\left(\frac{\mathscr{D}_{est}(\text{p})}{\mathscr{D}_{gt}(\text{p})},\frac{\mathscr{D}_{gt}(\text{p})}{\mathscr{D}_{est}(\text{p})}\right)<\theta \tag{6}$$

For the L1-metrics, the per-pixel depth estimate $\mathscr{D}_{est}(\text{p})$ is compared with the ground truth depth $\mathscr{D}_{gt}(\text{p})$ and averaged over the number of valid pixels $\mathscr{V}$. In the case of the two L1 error metrics, the set of valid pixels is made up of the pixels for which both $\mathscr{D}_{est}(\text{p})$ and $\mathscr{D}_{gt}(\text{p})$ exist. The accuracy and completeness, on the other hand, are normalized by the number of valid pixels $\mathscr{E}$ in the estimated depth map and by the number of non-zero pixels $\mathscr{G}$ in the ground truth depth map, respectively. Originating from the domain of image-based classification, the accuracy and completeness metric determines the ratio of correct pixels in the estimated depth map by comparing it to the corresponding measurement in the ground truth depth map, within a predefined threshold $\theta$. This particular measurement is commonly employed by the KITTI (Menze and Geiger, 2015) and ETH3D (Schöps et al., 2017) benchmarks as well. In contrast, in the case of the accuracy using the absolute error, a threshold value in meters is used instead of a percentage deviation.

The quantitative results are divided into two tables: one presenting the results based on the relative error and another displaying the results based on the absolute error. The relative error measure favors close objects by giving less weight to pixels with higher ground truth depth, resulting in tree tops and roofs being weighted differently than for example roads due to the nadir perspective. The absolute error metric, on the other hand, is unaffected by this bias and simplifies the interpretation.

To evaluate the performance of the self-supervised trained network, the UseGeo dataset was divided into training and test data. The majority of the first sub-dataset was used for the test dataset, while the remaining two sub-datasets were designated as training data. To avoid data leakage caused by the overlapping nature of the three sub-datasets, certain portions of the first dataset had to be excluded. Furthermore, a minimum requirement of three consecutive images with sufficient overlap was necessary for training purposes, resulting in the exclusion of additional images. In total, the training and test datasets encompassed 510 and 192 images, with 127 images being excluded. We use 768 × 448 pixels as the image resolution to train the self-supervised method. The division of dataset into continuous sequences and into training and test areas is publicly available for the purpose of reproducibility.

**Table 1**
Results of the depth estimation methods using offline processing, evaluated with the metrics based on the relative error. The results are listed by dataset, with the second-best result per column highlighted underlined and the best result additionally by bold print. The different COLMAP methods use either no prior knowledge $\text{COLMAP}_{SFM+MVS}$ or the given intrinsic and extrinsic camera calibration $\text{COLMAP}_{MVS}$. For comparison, we also provide corresponding results for the case when a higher resolution is considered $\text{COLMAP}_{MVS+8K}$.

| Dataset-1 | ↓L1-rel | ↑$\text{Acc}_{1.1}$ | ↑$\text{Cpl}_{1.1}$ | ↑$\text{Acc}_{1.05}$ | ↑$\text{Cpl}_{1.05}$ | ↑$\text{Acc}_{1.01}$ | ↑$\text{Cpl}_{1.01}$ |
|---|---|---|---|---|---|---|---|
| $\text{COLMAP}_{SFM+MVS}$ | 0.0047 | 0.9949 | 0.9515 | 0.9829 | 0.9400 | 0.9215 | 0.8815 |
| $\text{COLMAP}_{MVS}$ | **0.0044** | 0.9950 | 0.9537 | 0.9832 | 0.9424 | **0.9255** | 0.8873 |
| OpenMVS | **0.0044** | **0.9966** | 0.9404 | **0.9863** | 0.9307 | 0.9228 | 0.8711 |
| ACMMP | 0.0101 | 0.9802 | **0.9771** | 0.9652 | **0.9622** | 0.9042 | **0.9015** |
| $\text{COLMAP}_{MVS+8K}$ | 0.0035 | 0.9960 | 0.8952 | 0.9873 | 0.8874 | 0.9466 | 0.8509 |
| Dataset-2 | ↓L1-rel | ↑$\text{Acc}_{1.1}$ | ↑$\text{Cpl}_{1.1}$ | ↑$\text{Acc}_{1.05}$ | ↑$\text{Cpl}_{1.05}$ | ↑$\text{Acc}_{1.01}$ | ↑$\text{Cpl}_{1.01}$ |
| $\text{COLMAP}_{SFM+MVS}$ | 0.0050 | 0.9959 | 0.9573 | 0.9818 | 0.9439 | 0.9047 | 0.8701 |
| $\text{COLMAP}_{MVS}$ | **0.0048** | 0.9961 | 0.9557 | 0.9827 | 0.9429 | **0.9073** | 0.8709 |
| OpenMVS | 0.0051 | **0.9970** | 0.9391 | **0.9855** | 0.9284 | 0.9017 | 0.8498 |
| ACMMP | 0.0074 | 0.9865 | **0.9845** | 0.9702 | **0.9683** | 0.8899 | **0.8883** |
| $\text{COLMAP}_{MVS+8K}$ | 0.0044 | 0.9950 | 0.9537 | 0.9832 | 0.9424 | 0.9255 | 0.8873 |
| Dataset-3 | ↓L1-rel | ↑$\text{Acc}_{1.1}$ | ↑$\text{Cpl}_{1.1}$ | ↑$\text{Acc}_{1.05}$ | ↑$\text{Cpl}_{1.05}$ | ↑$\text{Acc}_{1.01}$ | ↑$\text{Cpl}_{1.01}$ |
| $\text{COLMAP}_{SFM+MVS}$ | 0.0052 | 0.9955 | 0.9580 | 0.9824 | 0.9454 | 0.8944 | 0.8612 |
| $\text{COLMAP}_{MVS}$ | **0.0050** | 0.9952 | 0.9555 | 0.9826 | 0.9434 | **0.9028** | 0.8670 |
| OpenMVS | 0.0051 | **0.9969** | 0.9419 | **0.9864** | 0.9320 | 0.8977 | 0.8484 |
| ACMMP | 0.0074 | 0.9865 | **0.9840** | 0.9702 | **0.9677** | 0.8865 | **0.8843** |
| $\text{COLMAP}_{MVS+8K}$ | 0.0037 | 0.9962 | 0.8613 | 0.9873 | 0.8536 | 0.9362 | 0.8097 |

### 4.3. Evaluation of point clouds

Unlike depth maps, the estimated point clouds lack a specific order, which presents challenges in finding precise correspondences between estimated points and their respective ground truth equivalent. Consequently, for each estimated point, first, a search is conducted to find its nearest neighbor in the ground truth. Subsequently, the error is calculated based on the point-to-point distance instead of point-to-plane, as the ground truth does not include normal vectors and the estimation could lead to uncertainties. For the experiments that do not use the given extrinsic and intrinsic camera parameters and in which SFM is performed, an alignment to the ground truth is performed. Initially, a coarse manual alignment is performed, followed by a more refined registration using the iterative-closest-point (ICP) algorithm (Besl and McKay, 1992). Additionally, ICP with scaling as an additional parameter is employed. Following the approach outlined in the work of Seitz et al. (2006), all errors are calculated at a completeness level of 90 % to assign less significance to outliers. A threshold of 0.2 m is used to determine completeness, indicating that all points in the ground truth point cloud within this distance of an estimated point are considered when calculating the completeness metric. The absolute error (L1-abs), root-mean-square error (RMSE), and completeness are employed as metrics for evaluation:

$$\text{L1}-\text{abs}(\mathscr{P}_{est}, \mathscr{P}_{gt}) = \frac{1}{|\mathscr{E}|}\sum_{\text{p}\in\mathscr{E}} |\mathscr{P}_{est}(\text{p}) - \mathscr{P}_{gt}(\text{p})| \tag{7}$$

$$\text{RMSE}(\mathscr{P}_{est}, \mathscr{P}_{gt}) = \sqrt{\frac{1}{|\mathscr{E}|}\sum_{\text{p}\in\mathscr{E}} \left(\mathscr{P}_{est}(\text{p}) - \mathscr{P}_{gt}(\text{p})\right)^2} \tag{8}$$

$$\text{Cpl}_{\theta}(\mathscr{P}_{est}, \mathscr{P}_{gt}) = \frac{1}{|\mathscr{G}|}\sum_{\text{p}\in\mathscr{G}} |\mathscr{P}_{est}(\text{p}) - \mathscr{P}_{gt}(\text{p})| < \theta \tag{9}$$

Here, $\mathscr{P}_{est}$ represents the estimated point cloud and $\mathscr{P}_{gt}$ the ground truth point cloud. For L1-abs and RMSE, the distance to the corresponding point in the ground truth point cloud $\mathscr{P}_{gt}(\text{p})$ is calculated for each estimated point $\mathscr{P}_{est}(\text{p})$ and normalized by the number of points in the estimated point cloud $\mathscr{E}$. By contrast in the case of the completeness, the distance to the corresponding point $\mathscr{P}_{est}(\text{p})$ in the estimate is calculated for every point $\mathscr{P}_{gt}(\text{p})$ in the ground truth point cloud and normalized by its number of points $\mathscr{G}$. If this distance is below the threshold value $\theta$, it is considered a valid point that contributes to the completeness. As a correspondence, the spatially closest point of the other point cloud is used in each case.

### 4.4. Triangle meshes

Triangle meshes, like point clouds, lack a specific order. However, in

**Table 2**
Quantitative results of the depth estimation methods using offline processing, evaluated with the metrics based on the absolute error. The results are listed by dataset, with the second-best result per column highlighted underlined and the best result additionally by bold print. The different COLMAP methods use either no prior knowledge $\text{COLMAP}_{SFM\ +\ MVS}$ or the given intrinsic and extrinsic camera calibration $\text{COLMAP}_{MVS}$. For comparison, we also provide corresponding results for the case when a higher resolution is considered $\text{COLMAP}_{MVS+8K}$.

| Dataset-1 | ↓L1-abs | ↑$\text{Acc}_{0.5}$ | ↑$\text{Cpl}_{0.5}$ | ↑$\text{Acc}_{0.1}$ | ↑$\text{Cpl}_{0.1}$ | ↑$\text{Acc}_{0.05}$ | ↑$\text{Cpl}_{0.05}$ |
|---|---|---|---|---|---|---|---|
| $\text{COLMAP}_{SFM+MVS}$ | 0.3724 | 0.8807 | 0.8430 | 0.4720 | 0.4533 | 0.2653 | 0.2550 |
| $\text{COLMAP}_{MVS}$ | **0.3500** | **0.8890** | 0.8526 | 0.5395 | 0.5183 | 0.3212 | 0.3086 |
| OpenMVS | 0.3507 | 0.8689 | 0.8205 | 0.4419 | 0.4186 | 0.2347 | 0.2227 |
| ACMMP | 0.7408 | 0.8718 | **0.8692** | **0.5695** | **0.5680** | **0.3617** | **0.3608** |
| $\text{COLMAP}_{MVS+8K}$ | 0.2765 | 0.9181 | 0.8254 | 0.6476 | 0.5837 | 0.4258 | 0.3842 |
| Dataset-2 | ↓L1-abs | ↑$\text{Acc}_{0.5}$ | ↑$\text{Cpl}_{0.5}$ | ↑$\text{Acc}_{0.1}$ | ↑$\text{Cpl}_{0.1}$ | ↑$\text{Acc}_{0.05}$ | ↑$\text{Cpl}_{0.05}$ |
| $\text{COLMAP}_{SFM+MVS}$ | 0.4397 | 0.8514 | 0.8193 | 0.4454 | 0.4298 | 0.2464 | 0.2379 |
| $\text{COLMAP}_{MVS}$ | **0.4238** | **0.8581** | 0.8240 | 0.4771 | 0.4588 | 0.2808 | 0.2702 |
| OpenMVS | 0.4482 | 0.8221 | 0.7751 | 0.3321 | 0.3136 | 0.1672 | 0.1578 |
| ACMMP | 0.6360 | 0.8447 | **0.8432** | **0.5197** | **0.5189** | **0.3198** | **0.3194** |
| $\text{COLMAP}_{MVS+8K}$ | 0.3224 | 0.9040 | 0.8160 | 0.6309 | 0.5706 | 0.4004 | 0.3624 |
| Dataset-3 | ↓L1-abs | ↑$\text{Acc}_{0.5}$ | ↑$\text{Cpl}_{0.5}$ | ↑$\text{Acc}_{0.1}$ | ↑$\text{Cpl}_{0.1}$ | ↑$\text{Acc}_{0.05}$ | ↑$\text{Cpl}_{0.05}$ |
| $\text{COLMAP}_{SFM+MVS}$ | 0.4489 | 0.8273 | 0.7969 | 0.3800 | 0.3670 | 0.2045 | 0.1976 |
| $\text{COLMAP}_{MVS}$ | **0.4307** | **0.8396** | 0.8066 | 0.4173 | 0.4014 | 0.2328 | 0.2240 |
| OpenMVS | 0.4413 | 0.8017 | 0.7579 | 0.2867 | 0.2714 | 0.1380 | 0.1306 |
| ACMMP | 0.6177 | 0.8271 | **0.8252** | **0.4510** | **0.4502** | **0.2648** | **0.2644** |
| $\text{COLMAP}_{MVS+8K}$ | 0.3166 | 0.8925 | 0.7721 | 0.5688 | 0.4938 | 0.3485 | 0.3023 |

addition to vertices, they also consist of triangles that compose the actual mesh structure. To assess the quality of the meshes, it is necessary to determine the minimum distance for each triangle to the ground truth point cloud, which is computationally time-consuming. To accomplish this with a reasonable run-time, a raycasting technique is employed, originating from every point within the ground truth point cloud. This approach enables efficient determination of the closest triangle for each point, along with the precise intersection point on the triangle. The distance between the position of the ground truth point and the point on the triangle serves as an error measure. Similar to the evaluation of point clouds, the metrics of L1-abs (Eq. (7)), RMSE (Eq. (8)), and Cpl (Eq. (9)) are employed.

## 5. Experimental results

In the following analysis, we review the results achieved by the presented methods on the dataset. Initially, we examine the performance in relation to depth estimation, assessing the approaches through the application of both relative and absolute error metrics. At this point, we distinguish between offline and online processing methods. Subsequently, we evaluate the reconstructed point clouds, and where applicable, the reconstructed triangle meshes.

### 5.1. Offline depth estimation

Approaches for offline depth estimation and 3D reconstruction focus primarily on the best possible results by using offline processing. This implies that the methods have access to all images during processing and can thus use several viewpoints to estimate the depth of an image and benefit from subsequent geometric filtering with multiple spatially close images, which all methods implement. As a result, all techniques yield significantly improved outcomes compared to the results achieved by online methods (cf. Section 5.2). The similarity of the results of the L1-rel error between the offline and online depth estimation approaches suggests that the L1-relative error metric is not appropriate for the comparison of these two different types of methods. However, as soon as $\text{Acc}_\theta$ or especially the absolute error is considered, the difference is apparent. Nonetheless, the offline methods also differ considerably from each other.

In this section the offline depth estimation, i.e. the precision of the depth maps estimated by COLMAP, OpenMVS, and ACMMP is evaluated. Unless otherwise specified, the experiments performed rely on the given resolution of 1989 × 1320 pixels with the standard parameterization of the evaluated toolbox. As mentioned above, OpenMVS and ACMMP only provide algorithms for dense depth estimation and 3D reconstruction with MVS and, thus, to evaluate these two approaches the given intrinsic and extrinsic camera parameters were used. In the case of COLMAP, we have evaluated three different configurations of input

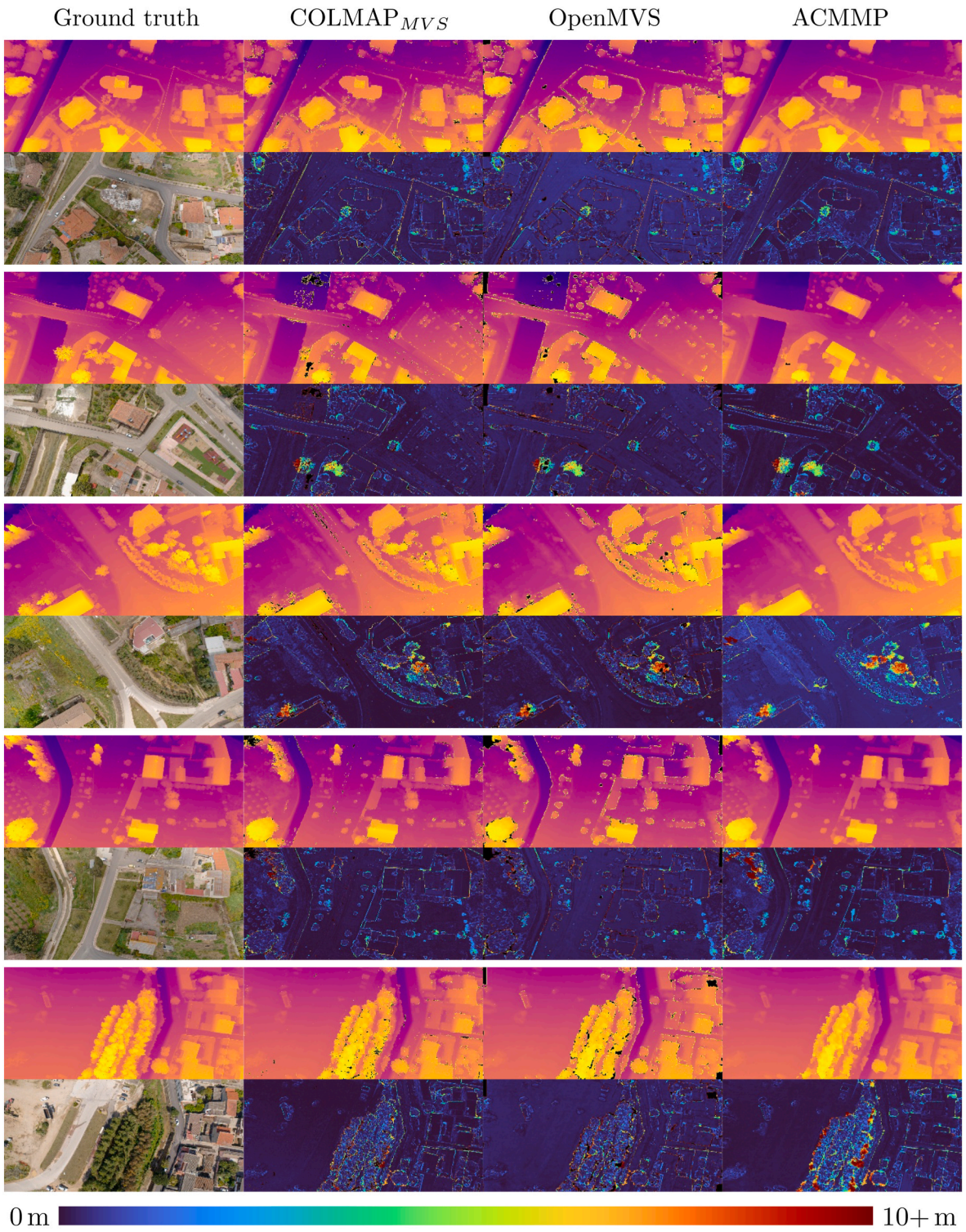



**Fig. 3.** Results of the depth estimation methods using offline processing. For each section, the depth maps are displayed in the first row, and the color-coded error maps with respect to the available ground truth in the second row. (For interpretation of the references to color in this figure legend, the reader is referred to the Web version of this article.)

data. In the configuration denoted as COLMAP$_{SFM+MVS}$, we have solely provided the down-scaled input images as input and let COLMAP estimate the intrinsic and extrinsic camera parameters by means of SFM. For COLMAP$_{MVS}$ the camera parameters given within the UseGeo dataset were used, just as with the other two approaches. And lastly, for COLMAP$_{MVS+8K}$ we have used the full-resolution imagery. But again, we rely on the provided intrinsic and extrinsic camera data. As the approach with the highest image resolution, namely COLMAP$_{MVS+8K}$, always yields the best quantitative results, we have also highlighted the second-best result in the tables below.

The importance of the quality of camera poses and camera calibration becomes apparent when comparing the depth maps of COLMAP with and without the given extrinsic and intrinsic calibration. As shown in Table 2, the L1-abs error of COLMAP$_{MVS}$ is up to 6 % lower compared

**Table 3**
Quantitative results of the depth estimation methods using online processing, evaluated with the metrics based on the relative error. Results are listed by dataset, with those from the self-supervised approach available only for Dataset-1. The second-best result per column is underlined, while the best result is additionally printed in bold script. All the methods use the provided intrinsic camera calibration, while the FaSS-MVS and PSL methods make use of the ground truth extrinsic parameters.

| Dataset-1 | ↓L1-rel | ↑$Acc_{1.1}$ | ↑$Cpl_{1.1}$ | ↑$Acc_{1.05}$ | ↑$Cpl_{1.05}$ | ↑$Acc_{1.01}$ | ↑$Cpl_{1.01}$ |
|---|---|---|---|---|---|---|---|
| FaSS-MVS$_{GPP}$ | 0.0096 | 0.9793 | 0.7689 | 0.9492 | 0.7451 | 0.8111 | 0.6355 |
| PSL$_{SplitOcc}$ | 0.0556 | 0.8025 | 0.8025 | 0.7621 | 0.7621 | 0.6734 | 0.6734 |
| PSL$_{SplitOcc\ GPP}$ | 0.0060 | 0.9841 | 0.5432 | 0.9742 | 0.5377 | 0.9031 | 0.4980 |
| PSL$_{BestKOcc}$ | 0.0300 | 0.8957 | 0.8957 | 0.8693 | **0.8693** | 0.8006 | **0.8006** |
| PSL$_{BestKOcc\ GPP}$ | **0.0040** | **0.9903** | 0.6563 | **0.9822** | 0.6509 | **0.9307** | 0.6163 |
| SMDE$_{ResNet18}$ | 0.0614 | 0.9399 | 0.9399 | 0.7436 | 0.7436 | 0.1994 | 0.1994 |
| SMDE$_{ResNet50}$ | 0.0350 | 0.9485 | 0.9485 | 0.7665 | 0.7665 | 0.2149 | 0.2149 |
| SMDE$_{PackNet01}$ | 0.0345 | 0.9498 | **0.9498** | 0.7689 | 0.7689 | 0.2186 | 0.2186 |
| Dataset-2 | ↓L1-rel | ↑$Acc_{1.1}$ | ↑$Cpl_{1.1}$ | ↑$Acc_{1.05}$ | ↑$Cpl_{1.05}$ | ↑$Acc_{1.01}$ | ↑$Cpl_{1.01}$ |
| FaSS-MVS$_{GPP}$ | 0.0092 | 0.9794 | 0.7669 | 0.9541 | 0.7473 | 0.7623 | 0.5959 |
| PSL$_{SplitOcc}$ | 0.0606 | 0.7974 | 0.7974 | 0.7508 | 0.7508 | 0.6398 | 0.6398 |
| PSL$_{SplitOcc\ GPP}$ | 0.0069 | 0.9776 | 0.5243 | 0.9646 | 0.5174 | 0.8674 | 0.4646 |
| PSL$_{BestKOcc}$ | 0.0353 | 0.8854 | **0.8854** | 0.8535 | **0.8535** | 0.7627 | **0.7627** |
| PSL$_{BestKOcc\ GPP}$ | **0.0046** | **0.9843** | 0.6314 | **0.9737** | 0.6246 | **0.9011** | 0.5772 |
| Dataset-3 | ↓L1-rel | ↑$Acc_{1.1}$ | ↑$Cpl_{1.1}$ | ↑$Acc_{1.05}$ | ↑$Cpl_{1.05}$ | ↑$Acc_{1.01}$ | ↑$Cpl_{1.01}$ |
| FaSS-MVS$_{GPP}$ | 0.0088 | 0.9725 | 0.7771 | 0.9481 | 0.7575 | 0.7564 | 0.6036 |
| PSL$_{SplitOcc}$ | 0.0594 | 0.8114 | 0.8114 | 0.7731 | 0.7731 | 0.6646 | 0.6646 |
| PSL$_{SplitOcc\ GPP}$ | 0.0066 | 0.9717 | 0.5564 | 0.9607 | 0.5502 | 0.8614 | 0.4934 |
| PSL$_{BestKOcc}$ | 0.0358 | 0.8853 | **0.8853** | 0.8563 | **0.8563** | 0.7676 | **0.7676** |
| PSL$_{BestKOcc\ GPP}$ | **0.0047** | **0.9764** | 0.6422 | **0.9668** | 0.6359 | **0.8936** | 0.5877 |

**Table 4**
Quantitative results of the depth estimation methods using online processing, evaluated with the metrics based on the absolute error. Results are listed by dataset, with those from the self-supervised approach available only for Dataset-1. The second-best result per column is underlined, while the best result is additionally printed in bold script. All the methods use the provided intrinsic camera calibration, while the FaSS-MVS and PSL methods make use of the ground truth extrinsic parameters.

| Dataset-1 | ↓L1-abs | ↑$Acc_{0.5}$ | ↑$Cpl_{0.5}$ | ↑$Acc_{0.1}$ | ↑$Cpl_{0.1}$ | ↑$Acc_{0.05}$ | ↑$Cpl_{0.05}$ |
|---|---|---|---|---|---|---|---|
| FaSS-MVS$_{GPP}$ | 0.7486 | 0.7620 | 0.5965 | 0.3612 | 0.2819 | 0.1943 | 0.1517 |
| PSL$_{SplitOcc}$ | 4.5773 | 0.6291 | 0.6291 | 0.3042 | 0.3042 | 0.1663 | 0.1663 |
| PSL$_{SplitOcc\ GPP}$ | 0.4884 | 0.8458 | 0.4659 | 0.4187 | 0.2291 | 0.2299 | 0.1257 |
| PSL$_{BestKOcc}$ | 2.3804 | 0.7611 | **0.7611** | 0.3735 | **0.3735** | 0.2009 | **0.2009** |
| PSL$_{BestKOcc\ GPP}$ | **0.3217** | **0.8894** | 0.5885 | **0.4565** | 0.3014 | **0.2468** | 0.1630 |
| SMDE$_{ResNet18}$ | 4.9545 | 0.1233 | 0.1233 | 0.0248 | 0.0248 | 0.0124 | 0.0124 |
| SMDE$_{ResNet50}$ | 2.8260 | 0.1332 | 0.1332 | 0.0269 | 0.0269 | 0.0134 | 0.0134 |
| SMDE$_{PackNet01}$ | 2.7866 | 0.1362 | 0.1362 | 0.0275 | 0.0275 | 0.0138 | 0.0138 |
| Dataset-2 | ↓L1-abs | ↑$Acc_{0.5}$ | ↑$Cpl_{0.5}$ | ↑$Acc_{0.1}$ | ↑$Cpl_{0.1}$ | ↑$Acc_{0.05}$ | ↑$Cpl_{0.05}$ |
| FaSS-MVS$_{GPP}$ | 0.7681 | 0.6549 | 0.5110 | 0.2455 | 0.1912 | 0.1310 | 0.1021 |
| PSL$_{SplitOcc}$ | 5.2157 | 0.5842 | 0.5842 | 0.2592 | 0.2592 | 0.1395 | 0.1395 |
| PSL$_{SplitOcc\ GPP}$ | 0.5870 | 0.7925 | 0.4238 | 0.3562 | 0.1891 | 0.1920 | 0.1018 |
| PSL$_{BestKOcc}$ | 2.9696 | 0.7103 | **0.7103** | 0.3178 | **0.3178** | 0.1686 | **0.1686** |
| PSL$_{BestKOcc\ GPP}$ | **0.3949** | **0.8437** | 0.5395 | **0.3896** | 0.2478 | **0.2074** | 0.1318 |
| Dataset-3 | ↓L1-abs | ↑$Acc_{0.5}$ | ↑$Cpl_{0.5}$ | ↑$Acc_{0.1}$ | ↑$Cpl_{0.1}$ | ↑$Acc_{0.05}$ | ↑$Cpl_{0.05}$ |
| FaSS-MVS$_{GPP}$ | 0.7624 | 0.6474 | 0.5163 | 0.2330 | 0.1855 | 0.1217 | 0.0969 |
| PSL$_{SplitOcc}$ | 5.3428 | 0.5963 | 0.5963 | 0.2425 | 0.2425 | 0.1288 | 0.1288 |
| PSL$_{SplitOcc\ GPP}$ | 0.5795 | 0.7723 | 0.4424 | 0.3183 | 0.1826 | 0.1695 | 0.0973 |
| PSL$_{BestKOcc}$ | 3.1116 | 0.7029 | **0.7029** | 0.2904 | **0.2904** | 0.1529 | **0.1529** |
| PSL$_{BestKOcc\ GPP}$ | **0.4137** | **0.8227** | 0.5409 | **0.3516** | 0.2309 | **0.1859** | 0.1220 |

to COLMAP$_{SFM+MVS}$. Increasing the resolution to 8K, which corresponds to about 16 times the number of pixels, leads to an expected greatly improved result, particularly with respect to the absolute error, which suggests that distant objects can be estimated more accurately. The depth maps estimated by OpenMVS achieve a slightly lower quality in terms of accuracy and completeness than comparable depth maps estimated by COLMAP. As displayed in Table 1, the depth maps of ACMMP, however, show significantly higher relative errors, which is probably due to considerably reduced filtering, since the completeness is higher. This effect is also visible in Fig. 3. The depth maps of ACMMP are clearly more complete, but it is evident from the error maps that the error is high compared to those estimated by COLMAP and OpenMVS in areas that are not filtered out. Interestingly, this flips when considering the absolute error, as can be seen from the rather strict criterion $Acc_{0.05}$. Here, ACMMP performs significantly better, suggesting that distant pixels are better estimated. Compared to the others, the depth maps of OpenMVS in some cases show more noise at object edges, which could explain the results of slightly lower quality.

### 5.2. Online depth estimation

Online methods are characterized by their ability to process image streams directly, without relying on global operations such as global bundle adjustment. Consequently, these methods typically utilize only a limited number of consecutive images or, in the case of the self-supervised approach, estimate depth solely based on the current image. As a result, the primary focus of these techniques lies in their execution speed, rather than the best possible result. Results for relative and absolute error are shown in Table 3 and Table 4. They are broken down for the individual subsets of the dataset. Here, the results of the self-supervised method are only available for the first sub-dataset, as the other two are part of the training data. Fig. 4 shows the results of the

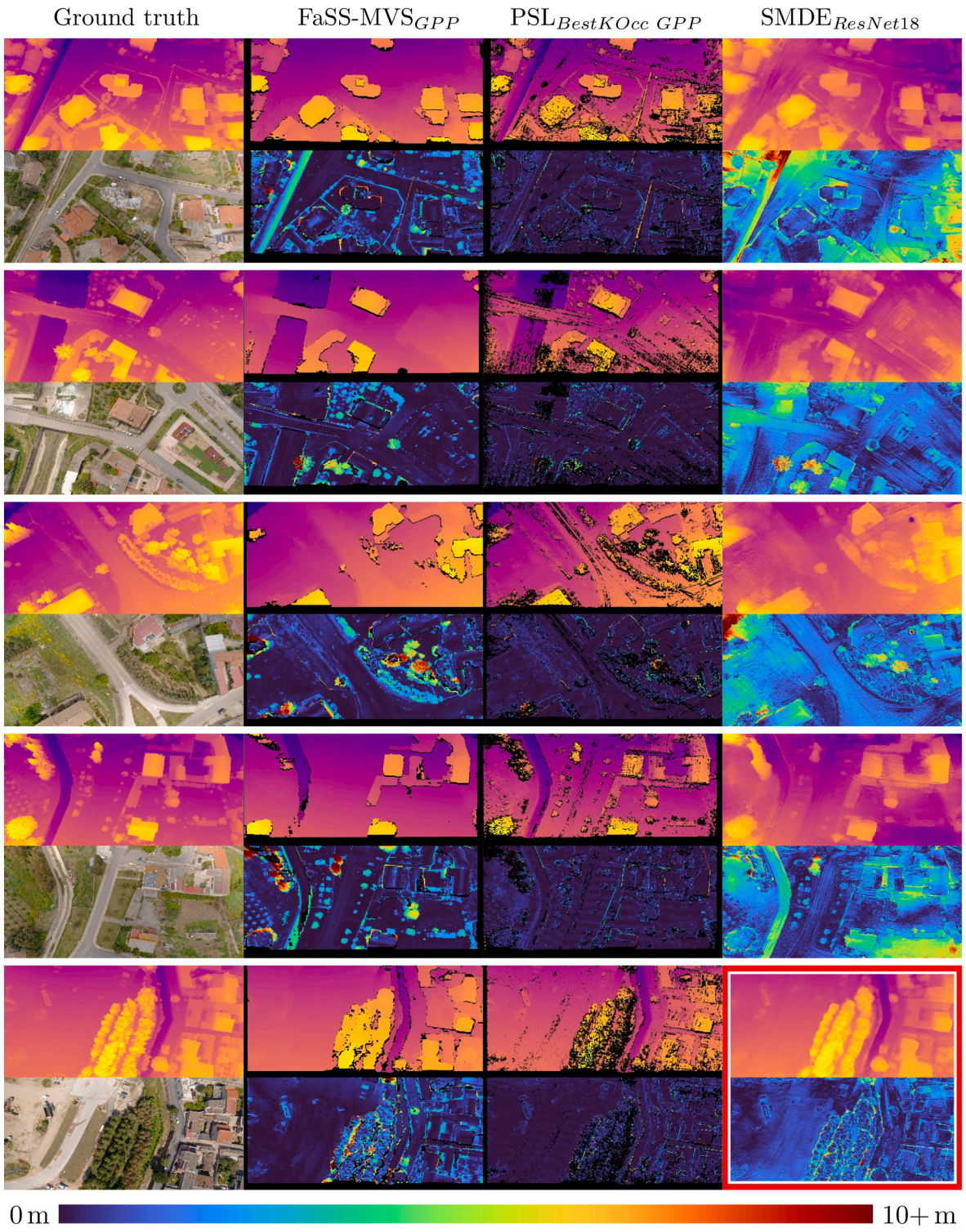



**Fig. 4.** Results of the depth estimation methods using online processing. For each section, the depth maps are displayed in the first row and the color-coded error maps with respect to the available ground truth in the second row. The red highlighted result of the self-supervised method in the last row is taken from the training data to show the potential of a more diverse training set. (For interpretation of the references to color in this figure legend, the reader is referred to the Web version of this article.)

online depth estimation methods. Each section includes the estimated depth maps in the first row and a color-coded error map representing the L1-abs error in the next row. Black pixels in the depth map indicate pixels that have been filtered out, resulting in lower completeness. The methods FaSS-MVS and PSL use the image resolution of 1989 × 1320 pixels and the self-supervised method uses a resolution of 768 × 448 pixels.

In the following, the specific parameterization of the methods is indicated by their subscript. The labels SplitOcc and BestKOcc indicate how occlusions are handled in the PSL approach. If geometric filtering is applied in post-processing, this is shown by the additional label geometric postprocessing (GPP). For the learning-based methods, the

**Table 5**
Quantitative results of the fused point clouds. The results are listed by dataset, with the second-best result per column highlighted underlined and the best result additionally by bold print. In each case, the point-to-point distance between the calculated and the ground truth point cloud is used as the measurement method. The different methods use either no prior knowledge like $COLMAP_{SFM+MVS}$ or the given intrinsic and extrinsic camera calibration denoted with *MVS*. For comparison, we also provide corresponding results for the case when a higher resolution is considered with $COLMAP_{MVS+8K}$.

| Dataset-1 | ↓L1-abs in m | ↓RMSE in m | ↑Cpl. | No. points |
|---|---|---|---|---|
| $COLMAP_{SFM+MVS}$ | 0.0778 | 0.0912 | 0.5165 | 13,753,122 |
| $COLMAP_{MVS}$ | 0.0609 | 0.0700 | 0.5911 | 13,787,242 |
| OpenMVS | 0.0765 | 0.0898 | 0.5682 | 23,014,725 |
| ACMMP | **0.0473** | **0.0541** | **0.6331** | 53,033,375 |
| $COLMAP_{MVS+8K}$ | 0.0453 | 0.0510 | 0.6743 | 190,358,894 |
| **Dataset-2** | **↓L1-abs in m** | **↓RMSE in m** | **↑Cpl.** | **No. points** |
| $COLMAP_{SFM+MVS}$ | 0.1599 | 0.1983 | 0.3339 | 20,215,643 |
| $COLMAP_{MVS}$ | 0.0690 | 0.0812 | 0.5965 | 20,332,931 |
| OpenMVS | 0.0976 | 0.1159 | 0.5469 | 34,001,002 |
| ACMMP | **0.0491** | **0.0569** | **0.6436** | 75,096,267 |
| $COLMAP_{MVS+8K}$ | 0.0445 | 0.0492 | 0.7041 | 271,763,480 |
| **Dataset-3** | **↓L1-abs in m** | **↓RMSE in m** | **↑Cpl.** | **No. points** |
| $COLMAP_{SFM+MVS}$ | 0.1294 | 0.1611 | 0.3897 | 17,087,339 |
| $COLMAP_{MVS}$ | 0.0782 | 0.0921 | 0.5418 | 17,199,568 |
| OpenMVS | 0.1061 | 0.1247 | 0.4865 | 28,418,405 |
| ACMMP | **0.0566** | **0.0664** | **0.5840** | 55,845,638 |
| $COLMAP_{MVS+8K}$ | 0.0514 | 0.0581 | 0.6510 | 210,699,535 |

backbone used for the encoder of the depth network is specified.

As can be seen in Table 3 and Table 4, $PSL_{BestKOcc\ GPP}$ performs best in terms of accuracy, followed by $PSL_{SplitOcc\ GPP}$ and $FaSS\text{-}MVS_{GPP}$. Especially noteworthy here is the effectiveness of the geometric verification in post-processing (GPP), as this significantly improves the results compared to the basic methods. However, this inevitably reduces the completeness, which means that $PSL_{BestKOcc}$ is the best overall method in terms of this metric. As visible in the quantitative results, the different scenery of the individual sub-datasets also affects the outcome of the approaches, as the results differ considerably. For example, the L1-abs of $PSL_{BestKOcc\ GPP}$ is 22.2 % higher on Dataset-3 than on Dataset-1.

Despite $PSL_{BestKOcc\ GPP}$ achieving superior quantitative outcomes, it requires by far the longest processing time, with an average 9300 ms, while $PSL_{SplitOcc\ GPP}$ requires 527 ms and $FaSS\text{-}MVS_{GPP}$ only 442 ms of computation time per image using a NVIDIA RTX A2000 GPU.

In contrast to FaSS-MVS and PSL, the self-supervised method requires only one image to estimate the depth map, which is the reason why no geometric verification can be performed. However, as can be seen from the relative error in Table 3, and especially from the absolute error in Table 4, the performance is inferior compared to the non-learning-based methods. The performance seems to depend directly on the amount of training data, since experiments using only Dataset-2 for training already achieve significantly worse results. It can therefore be assumed that a larger training dataset with similar properties could lead to much better results. In the field of autonomous driving, with considerably larger datasets available, self-supervised monocular approaches already achieve results in the range of supervised trained methods (Guizilini et al., 2020). Another indication that more training data could improve the results is that the complexity of the encoder does not have a great influence on the results. Although an encoder based on a ResNet50 has more than twice as many parameters as a ResNet18, the results regarding $Acc_\theta$ do not improve significantly. In fact, the jump to a much more complex and advanced architecture like PackNet (Guizilini et al., 2020) results in almost no improvement in performance. The last image in the bottom-right corner of Fig. 4, which is highlighted with a red frame, shows an example from the training dataset. Both visually and from the error map, a significantly better result is visible, representing the potential performance achievable with a better training dataset.

### 5.3. Offline 3D reconstruction

Regarding the reconstruction of a dense point cloud, we have only considered offline-based methods in this work. For this reason, only the results for COLMAP, OpenMVS and ACMMP are examined in the following, as can be seen in Table 5. For COLMAP, results with higher resolution as well as with SFM are additionally shown. Results in the form of color-coded error maps with respect to the available ground truth are shown in Fig. 5. All experiments were conducted on a server with 4 Titan X GPUs and a 24 core CPU, using the default parameters for

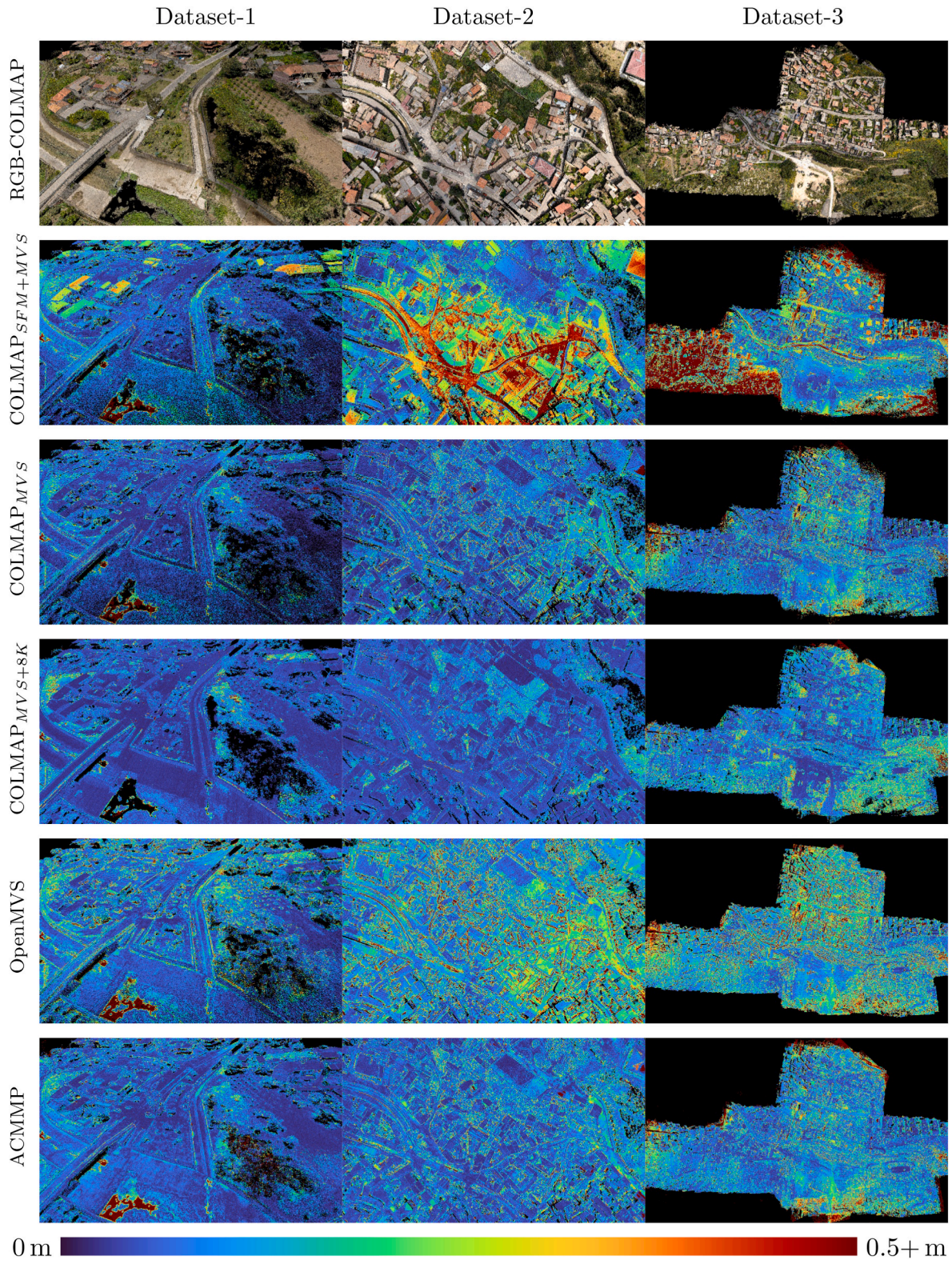



**Fig. 5.** Results of the reconstructed point clouds. The first row shows a colored point cloud for reference and the following rows show the error maps.

each framework.

As visible in Tables 5, if the same configuration is considered, i.e. the same image resolution and the use of the given extrinsic and intrinsic camera parameters, then ACMMP clearly outperforms COLMAP and OpenMVS. ACMMP achieves both lower errors in terms of L1-abs and RMSE, as well as higher completeness. This means that ACMMP delivers the best result for the three sub-datasets in each case and also delivers more points in the point cloud than the other two methods combined. Only the COLMAP reconstruction which uses the full image resolution of 8K, i.e. COLMAP$_{MVS+8K}$, achieves slightly better results than ACMMP in terms of L1-abs, RMSE, and Cpl. However, the number of points of the resulting point cloud is also significantly higher. As with the depth maps,

**Table 6**
Quantitative results of the reconstructed triangle meshes. The results are listed by dataset, with the second-best result per column highlighted underlined and the best result additionally by bold print. In each case, the point-to-point distance between the individual triangles and the ground truth point cloud is used for error assessment. The different methods use either no prior knowledge like COLMAP$_{SFM\ +\ MVS}$ or the given intrinsic and extrinsic camera calibration denoted with *MVS*. For comparison, we also provide corresponding results for the case when a higher resolution is considered with COLMAP$_{MVS+8K}$.

| Dataset-1 | ↓L1-abs in m | ↓RMSE in m | ↑Cpl. | No. triangles |
|---|---|---|---|---|
| COLMAP$_{SFM+MVS}$ | 0.1011 | 0.1295 | 0.5223 | 59,195,510 |
| COLMAP$_{MVS}$ | 0.0794 | 0.1049 | **0.5976** | 57,747,344 |
| OpenMVS$_{no\ refine}$ | 0.0816 | 0.1173 | 0.5282 | 7,450,170 |
| OpenMVS | **0.0261** | **0.0531** | 0.5918 | 1,467,494 |
| COLMAP$_{MVS+8K}$ | 0.0754 | 0.0980 | 0.6607 | 109,727,922 |
| **Dataset-2** | **↓L1-abs in m** | **↓RMSE in m** | **↑Cpl.** | **No. triangles** |
| COLMAP$_{SFM+MVS}$ | 0.2221 | 0.2791 | 0.3239 | 78,134,162 |
| COLMAP$_{MVS}$ | 0.0984 | 0.1336 | **0.5981** | 77,404,013 |
| OpenMVS$_{no\ refine}$ | 0.1127 | 0.1558 | 0.4896 | 11,637,590 |
| OpenMVS | **0.0704** | **0.1218** | 0.5425 | 2,394,468 |
| COLMAP$_{MVS+8K}$ | 0.0868 | 0.1172 | 0.6966 | 120,631,107 |
| **Dataset-3** | **↓L1-abs in m** | **↓RMSE in m** | **↑Cpl.** | **No. triangles** |
| COLMAP$_{SFM+MVS}$ | 0.2044 | 0.2571 | 0.3265 | 55,706,150 |
| COLMAP$_{MVS}$ | 0.1144 | 0.1515 | **0.5442** | 65,912,047 |
| OpenMVS$_{no\ refine}$ | 0.1243 | 0.1687 | 0.4339 | 10,329,882 |
| OpenMVS | **0.0714** | **0.1219** | 0.4850 | 2,195,850 |
| COLMAP$_{MVS+8K}$ | 0.1015 | 0.1332 | 0.6337 | 112,077,677 |

the point clouds also show that performing SFM instead of using the ground truth camera parameters leads to a significantly worse result, even though the point clouds are aligned using ICP. This is particularly visible in the results in Fig. 5 on Dataset-2 and Dataset-3. Here, clearly more points are colored with red tones, which indicates a higher L1-abs error. In these exemplary images, OpenMVS also shows the erroneous points that presumably lead to poorer quantitative results. However, it should be noted that, unlike ACMMP, OpenMVS is intended for the reconstruction of a triangle mesh and accordingly the point cloud is only an intermediate product.

As can be seen in Table 5, all methods achieve a relatively low completeness score, which is probably due to the fact that the ground truth point cloud extends well beyond the areas covered by the images (cf. Fig. 8).

Regarding run-time complexity across all three sub-datasets, OpenMVS required on average 1 h 25 min, COLMAP 2 h 59 min and ACMMP 5 h 08 min to reconstruct a dense point cloud.

### 5.4. Offline mesh reconstruction

Since only COLMAP and OpenMVS support the reconstruction of triangle meshes, the following evaluation focuses on these two methods. To investigate the improvement of the OpenMVS refinement step in quality, we evaluate the reconstruction step by itself in addition to the final result. Similar to the evaluation of the point clouds, we use the LiDAR point cloud as ground truth to determine the deviation of the triangles. For this purpose, we use the point on each triangle that corresponds to the minimum distance to a point in the ground truth point cloud as a distance metric. The quantitative results obtained in this regard are shown in Table 6. One example from the results is shown in Fig. 6. For each method, there is a shaded rendering of the mesh at the evaluated level-of-detail, as well as a color-coded error maps with respect to the available ground truth showing the L1-abs error per triangle.

As Table 6 and Fig. 6 show, the triangle meshes generated by OpenMVS and COLMAP differ not only in terms of their error metrics but also in their level-of-detail. The final models of OpenMVS have only 3 % of the number of triangles compared to the mesh of COLMAP using the same image and camera configuration, despite the fact that the point clouds of OpenMVS have almost twice as many points as COLMAP. This is because the number of triangles is automatically reduced during the generation of the mesh. This reduction can be seen in Fig. 6 in the smoother surface, for example, at the railing of the bridge. Despite this massive reduction in the level-of-detail, the reconstructions using OpenMVS perform best in terms of error metrics, which is particularly striking since the COLMAP reconstruction with 8K image resolution is also outperformed in almost all cases. The strong mesh decimation seem to be only noticeable with respect to the completeness, as COLMAP performs up to 6 % better here. At this point, the mesh refinement step of OpenMVS should be highlighted, as it significantly improves the

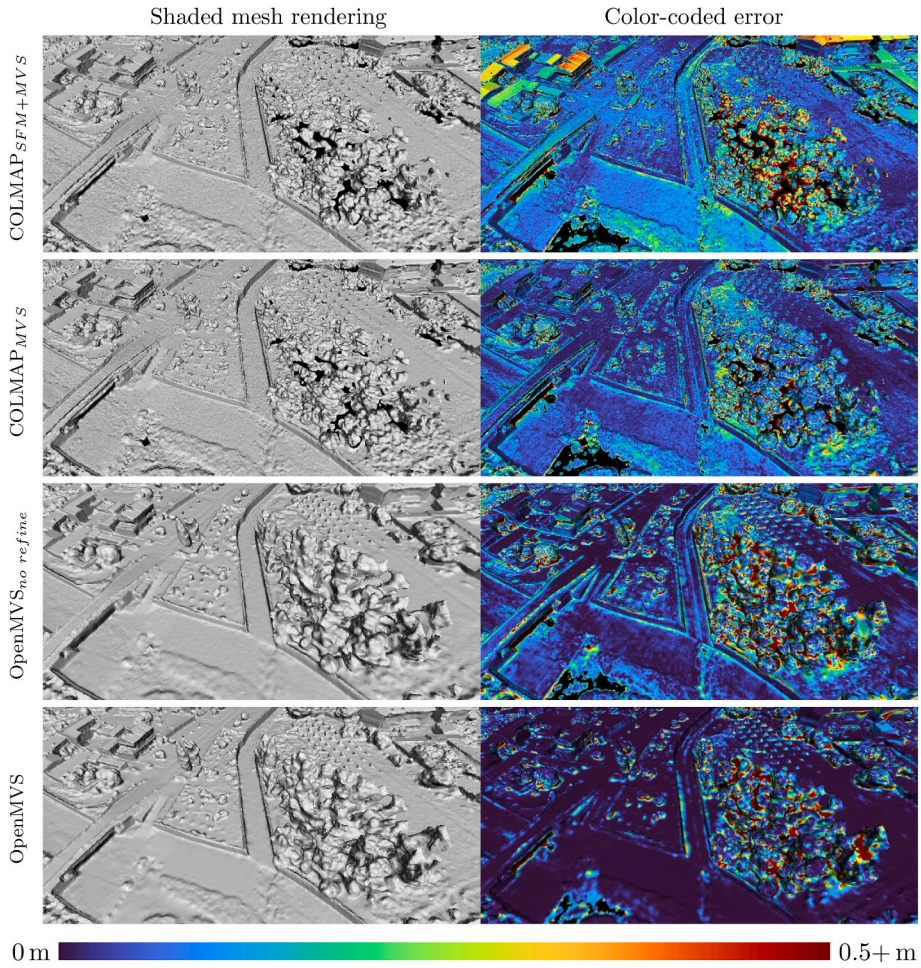



**Fig. 6.** Results of the reconstructed triangle meshes. The left column shows a rendered image of the mesh and the right column shows the error for each triangle color-coded compared to the ground truth point cloud. (For interpretation of the references to color in this figure legend, the reader is referred to the Web version of this article.)

**Table 7**
Overview table for the online approaches. The different methods denoted with *GPP* use geometric filtering. The symbols represent a ranking of the individual methods from poor (–) to high (++).

| Online approaches | Run-time Complexity | Depth map Acc | Depth map Cpl |
|---|---|---|---|
| FaSS-MVS$_{GPP}$ | + | + | + |
| PSL$_{SplitOcc}$ | + | o | + |
| PSL$_{SplitOcc\ GPP}$ | + | + | o |
| PSL$_{BestKOcc}$ | - - | o | + |
| PSL$_{BestKOcc\ GPP}$ | - - | ++ | o |
| SMDE | ++ | - - | o |

**Table 8**
Overview table for the offline approaches. The different methods use either no prior knowledge like COLMAP$_{SFM\ +\ MVS}$ or the given intrinsic and extrinsic camera calibration denoted with *MVS*. ACMMP does not implement a mesh reconstruction step. The symbols represent a ranking of the individual methods from poor (–) to high (++).

| Offline approaches | Run-time complexity | Depth map | Point cloud | Triangle mesh |
|---|---|---|---|---|
| COLMAP$_{SFM\ +\ MVS}$ | o | + | – | o |
| COLMAP$_{MVS}$ | + | ++ | + | + |
| OpenMVS | ++ | + | o | ++ |
| ACMMP | – | + | ++ | N/A |

result both in terms of error metrics and completeness, which can also be seen in Fig. 6. The error map of the final result appears in a darker blue, which indicates an overall better result. As already stated in the previous evaluations, the use of SFM also leads to worse results in terms of mesh generation. It is important to emphasize that COLMAP focuses on SFM and MVS and does not prioritize the meshing of the point clouds. OpenMVS, on the other hand, focuses strongly on the generation and refinement of triangle meshes, which is also reflected in the results.

## 6. Discussion

In the preceding section, it became evident that there are several significant differences between the presented online and offline approaches. In the case of the online methods, only depth estimation was examined. Table 7 provides a short overview of the results from the previous chapter in the categories of run-time complexity, accuracy and completeness. The symbols represent a ranking of the individual methods from poor (–) to high (++). As can be seen here, PSL$_{BestKOcc\ GPP}$ achieves the best result in terms of accuracy, followed by PSL$_{SplitOcc\ GPP}$ and FaSS-MVS$_{GPP}$. This impression is further emphasized when examining the absolute error and thus weighting farther away objects equally with those that are close by. In terms of completeness, the ranking of the methods remains the same, whereby the variants without geometric filtering show better results. At this point, the effect of the geometric filtering in post-processing becomes apparent, which significantly increases the accuracy, but also results in lower completeness. Another


**Fig. 7.** Results of the NeRF based methods. The COLMAP dense reconstruction and the NeRF-volume-rendering-based reconstructions. The examples shown in this figure used three sub-modules for Mega-NeRF and Block-NeRF.

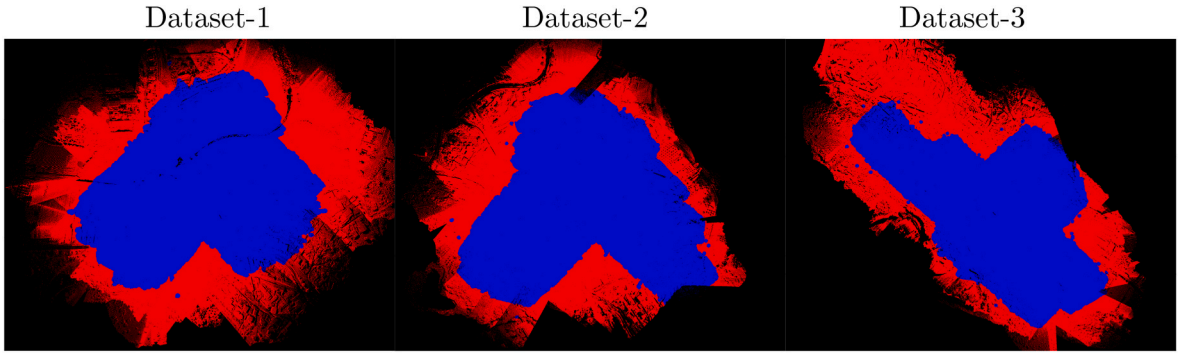



**Fig. 8.** Filtered ground truth. The red area shows the provided LiDAR point clouds, which extend beyond the area covered by the images. The blue area represents our adjusted ground truth. (For interpretation of the references to color in this figure legend, the reader is referred to the Web version of this article.)

aspect to consider is the trade-off between execution speed and quantitative results. Which is why PSL$_{SplitOcc\ GPP}$ and FaSS-MVS$_{GPP}$ are probably the better compromise, as PSL$_{BestKOcc\ GPP}$ takes significantly longer, as shown above. Fast execution is a major criterion, particularly for real-time processing applications, which are the inherent focus of online methods. This aligns with the findings of the self-supervised method, which notably exhibits the fastest execution speed and can efficiently handle real-time video streams even on low-performance hardware. Furthermore, unlike other online methods, the self-supervised approach does not require image bundles for inference since it estimates depth using only a single image. However, these advantages come at the cost of significantly poorer accuracy and lower completeness. The big difference between training and test errors also suggests that a larger and more diverse dataset may be helpful in this situation. In addition, the setting of the dataset proved to be quite challenging, to which the nadir perspective of the images probably contributed in particular. Thus, in order for the investigated architectures to converge reliably on the dataset, a pre-training of the network has been necessary in order to correctly estimate the camera movement.

In contrast, offline approaches, where processing times are in the order of minutes or hours rather than milliseconds or seconds, do not place as much emphasis on speed. For these methods, the depth estimation, the reconstruction of a dense point cloud and, if available, the generation of a triangle mesh are examined. It is shown that each method prioritizes different domains, which is also reflected in the overview in Table 8. In terms of depth estimation, COLMAP achieves the best result, followed by ACMMP and OpenMVS, if the same prerequisites such as image resolution and given extrinsic and intrinsic data are applied. ACMMP has a considerably higher completeness, which probably also explains the comparatively high L1 error. However, ACMMP reconstructs far away objects better, which is revealed by the higher $Acc_{0.1}$ and $Acc_{0.05}$ shown in Table 2. If the absolute L1 error is analyzed for the depth maps, some online methods perform slightly better than

offline methods, which is particularly true for the methods with geometric post processing. This suggests that geometric filtering is used much more aggressively here, as the resulting completeness values are considerably lower. The effect is relativized when the results for accuracy are compared, as the offline methods perform significantly better. When the individual point clouds are compared, ACMMP achieves the best results ahead of COLMAP in terms of both accuracy and completeness, while at the same time, the point cloud is by far the densest. However, if the triangle mesh is evaluated, OpenMVS achieves the best result in terms of accuracy, but with slightly lower completeness compared to COLMAP. This is probably due to the significantly lower number of triangles in the mesh since OpenMVS decimates the reconstructed mesh in its standard configuration. The mesh refinement step of OpenMVS is particularly noteworthy, as it significantly improves the result. The ACMMP method does not include any mesh generation.

In terms of depth estimation and reconstruction of the point clouds, the experiments with $\text{COLMAP}_{MVS+8K}$, which uses the full image resolution of $7953 \times 5279$ pixels, stand out. However, since the advantage in point cloud reconstruction is small compared to ACMMP with the standard resolution and OpenMVS delivers better results in mesh generation, it is questionable whether the significantly longer computation time is justified. At this point, it would be interesting to see the results achieved by the other two methods, i.e. OpenMVS and ACMMP, using the high-resolution imagery. This, however, is challenging due to hardware limitations and very long processing times, due to their less sophisticated parallelization compared to COLMAP.

### 6.1. Influence of the given camera calibration on the reconstruction quality

Since the datasets have both extrinsic and intrinsic camera calibration, comparative studies can be performed that additionally emphasize the SFM pipeline. In contrast to the other experiments, $\text{COLMAP}_{SFM + MVS}$ does not use any prior knowledge in the form of intrinsic and extrinsic camera calibration, so SFM must be performed to determine the camera parameters. Due to the fact that $\text{COLMAP}_{SFM + MVS}$ has no prior knowledge of the scene, the resulting 3D model is also not metric and cannot be directly compared to the ground truth. Instead, we use a coarse manual alignment followed by ICP and an additional ICP with scale estimation as refinement. Compared to the other results, the results drop significantly when SFM has to be performed, which is especially visible in the results for the point clouds and the meshes. However, if for the reconstruction with COLMAP the intrinsic camera calibration is fixed and only the extrinsic calibration is estimated, the results almost reach the level of the reconstruction with fixed intrinsic and extrinsic calibration. This shows that a good intrinsic calibration, which is much easier to obtain than the extrinsic parameters, already leads to significantly better results. Accordingly, a large part of this drop-off in performance could be due to the inaccurate estimation of the intrinsic camera calibration, if, in addition, the extrinsic parameters have to be estimated as well. Especially in the case of the three datasets considered here with nearly nadir perspectives on the scene, a correct estimation of the focal length is challenging. Due to the fact that an inaccurately estimated focal length causes the resulting 3D reconstruction to be deformed, the comparison with the ground truth is difficult, which probably also contributes to the worse result. One way to account for the inaccurate estimation of the focal length would be to scale the point cloud along the vertical axis (here Z-axis), which improves the result. But we decided not to display these results because we wanted to evaluate the out-of-the-box performance of the frameworks.

In addition to the experiments listed in the tables above, we have also investigated whether the initialization with the given camera calibration, followed by SFM to estimate the extrinsic camera data, as well as to refine the given intrinsic parameters, improves the result. By using the full calibration as a starting point and then refining both the focal length and the principal point, we achieve slightly better results on all three datasets. On Dataset-1, the difference is the smallest with 0.2 % and 0.3 % lower L1-abs error on point cloud and mesh, respectively, and a 0.1 % higher $\text{Acc}_{0.05}$ for the depth maps. However, on Dataset-2 and Dataset-3, the improvements are much more significant at 8.4 %, 13.0 %, 2.8 %, and 2.8 %, 5 %, 5 %, respectively for point clouds, meshes, and depth maps. This could be due to the run-to-run variances. However, it is noticeable that the results for all three datasets are better, albeit marginally so for Dataset-1. An alternative explanation is that the improved intrinsic calibration works better for COLMAP as a framework, which does not necessarily mean that the calibration is universally better.

### 6.2. Comparison with methods based on volume rendering

Since the work of Mildenhall et al. (2021), approaches based on neural rendering have gained a lot of attention. Even though the actual goal is to synthesize new perspectives, they can also be used to reconstruct point clouds or implicit surfaces. For this purpose, the continuous space is usually sampled at equidistant steps and a density threshold is used to select valid points. As part of this work, we carried out extensive investigations on all three sub-datasets and used three different methods: Mega-NeRF (Turki et al., 2022), Block-NeRF (Tancik et al., 2022) and Direct Voxel Grid Optimization (Sun et al., 2022). However, as Fig. 7 evidently shows, the results fall short of the performance of conventional methods. This also applies to the quantitative results, which is why we limit the discussion here to qualitative examples. For the examples shown in Fig. 7 three sub-modules for Mega-NeRF and Block-NeRF were used. A larger number of sub-modules did not provide any added value in our experiments. Row one shows the reconstruction with $\text{COLMAP}_{MVS}$ as a reference. As long as the point clouds are viewed from the nadir perspective, the models look visually appealing. But as soon as this viewing angle is changed, this is no longer the case. This is particularly visible in the last column, which displays the side perspective of a big slope in Dataset-1. Here, it can be seen that the NeRF-based approaches have great difficulties in reconstructing the correct geometry. This is also consistent with the findings of Nex et al. (2023) using NeRF-based approaches on the UseGeo dataset.

Similar to the SMDE method, the low number of images and their nadir perspective on the scene could be responsible for the rather poor results. In addition, most NeRF-based methods do not focus on reconstructing large-scale outdoor scenes or, as in the case of Mega-NeRF, usually use large amounts of oblique imagery. The fact that the resulting point clouds look good from a nadir perspective is probably because these approaches are mainly trained and used for creating new viewpoints which are similar to the original images.

### 6.3. Overlap between camera images and LiDAR ground truth

Since all areas covered by the camera are also covered by the LiDAR ground truth, it is possible to use the nearest ground truth point for each point or triangle to calculate the accuracy of the point clouds and meshes. For the completeness score, however, we need to assign a correspondence in the estimate to each point in the ground truth point cloud, which is problematic because the LiDAR scan extends well beyond the area covered by images. For this reason, the results presented earlier obtained in terms of completeness appear rather low. To get a more realistic value, we have additionally filtered the ground truth by removing all points at the edge. For this purpose, we use a distance threshold compared to our densest reconstruction, the COLMAP reconstruction using images with 8K resolution. To prevent photogrammetrically challenging regions, such as vegetation and weakly textured areas that are often not included in the reconstruction from also being filtered out of the LiDAR point cloud, we project both point clouds onto the horizontal plane (here XY-plane). Then we remove all points that are farther than 5 m from a point of the photogrammetric point cloud. In this way, we remove between 14 % and 18 % of the points from the datasets.

Fig. 8 visualizes this filtering process by showing the adjusted LiDAR point cloud in blue and the excluded points in red. However, as particularly visible in Dataset-1 and Dataset-2, this procedure leads to clusters of points around outliers of the photogrammetric reconstruction. On average, evaluating with this adjusted ground truth leads to a 10 % higher completeness.

## 7. Conclusion

In summary, we have extensively evaluated multiple approaches on the UseGeo dataset. In doing so, we have created a baseline in the field of offline and online depth estimation, as well as dense reconstruction of point clouds and triangle meshes from UAV-borne aerial imagery using current state-of-the-art methods. The evaluation routines used for this are freely available on GitHub,[11] facilitating a universal and comparable evaluation of depth estimation and 3D reconstruction methods on the UseGeo dataset.

Our analyses of this dataset have shown that, depending on the scenario, different methods excel in distinct categories. Among the offline methods, COLMAP gives the lowest overall error in terms of depth estimation, but the depth maps of ACMMP are more complete. However, when evaluating the point clouds, ACMMP produces both the lowest error and the highest completeness while also providing the densest result. But when considering the quality of the triangle meshes, OpenMVS produces meshes with the lowest error while, at the same time, decimating the meshes to reduce their size. In the future, it could be interesting to investigate whether a combination of the three approaches would lead to a better overall result. In terms of volume-based neural rendering approaches, the results were quite poor, which might be due to the nadir perspective and total number of images.

In the approaches for online depth estimation, the effect of geometric verification is particularly evident, as it significantly reduces the error at the cost of completeness. Although the variant $\mathrm{PSL}_{BestKOcc\ GPP}$ achieves the best result compared to $\mathrm{PSL}_{SplitOcc\ GPP}$ and FaSS-$\mathrm{MVS}_{GPP}$, it also has by far the highest run time. The self-supervised method has the advantage that the depth is only estimated from a singular image, which leads to its fast execution speed, but the error is significantly higher when compared to using the conventional methods. One factor here is probably the particularly challenging nature and small amount of the training data, which is also reflected in the fact that the deeper architectures offer little added value.

Even though the focus of this work was on depth estimation and dense reconstruction, the importance of good camera calibration is noticeable. The experiments that conducted SFM to estimate intrinsic and extrinsic camera parameters instead of using the ground truth information yielded significantly worse results. However, this can be largely mitigated if at least the intrinsic parameters are specified. In the future, it would be interesting to investigate the performance of SFM using different approaches in more detail by comparing the reconstructed camera trajectory with the ground truth. We have only indirectly evaluated this by observing a drop in performance in terms of depth estimation and dense reconstruction.

## Declaration of competing interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

## Acknowledgments

We acknowledge support by the KIT-Publication Fund of the Karlsruhe Institute of Technology.

[11] https://github.com/UseGeoEvaluation/DepthEstimationAnd3DReconstruction.

## References

Adorjan, M., 2016. OpenSfM: A Collaborative Structure-From-Motion System. Ph.D. thesis. Wien. URL: https://github.com/mapillary/OpenSfM.

Agarwal, S., Furukawa, Y., Snavely, N., Simon, I., Curless, B., Seitz, S.M., Szeliski, R., 2011. Building Rome in a day. Commun. ACM 54, 105–112.

Barron, J.T., Mildenhall, B., Tancik, M., Hedman, P., Martin-Brualla, R., Srinivasan, P.P., 2021. Mip-nerf: a multiscale representation for anti-aliasing neural radiance fields. In: Proc. IEEE International Conference on Computer Vision, pp. 5855–5864.

Bayomi, N., Fernandez, J.E., 2023. Eyes in the sky: drones applications in the built environment under climate change challenges. Drones 7, 637.

Besl, P.J., McKay, N.D., 1992. Method for registration of 3-D shapes. In: Proc. Sensor Fusion IV: Control Paradigms and Data Structures, pp. 586–606. SPIE.

Botta, A., Cavallone, P., Baglieri, L., Colucci, G., Tagliavini, L., Quaglia, G., 2022. A review of robots, perception, and tasks in precision agriculture. Appl. Mech. 3, 830–854.

Collins, R.T., 1996. A space-sweep approach to true multi-image matching. In: Proc. IEEE Conference on Computer Vision and Pattern Recognition, pp. 358–363.

Dosovitskiy, A., Beyer, L., Kolesnikov, A., Weissenborn, D., Zhai, X., Unterthiner, T., Dehghani, M., Minderer, M., Heigold, G., Gelly, S., Uszkoreit, J., Houlsby, N., 2021. An image is worth 16x16 words: transformers for image recognition at scale. In: Proc. International Conference on Learning Representations. URL: https://openreview.net/forum?id=YicbFdNTTy.

Fanta-Jende, P., Steininger, D., Kern, A., Widhalm, V., Apud Baca, J.G., Hofstätter, M., Simon, J., Bruckmüller, F., Sulzbachner, C., 2023. Semantic real-time mapping with UAVs. PFG–journal of photogrammetry. Remote Sens. Geoinf. Sci. 1–14.

Florea, H., Miclea, V.C., Nedevschi, S., 2021. WildUAV: monocular UAV dataset for depth estimation tasks. In: Proc. IEEE International Conference on Intelligent Computer Communication and Processing, pp. 291–298.

Flynn, J., Neulander, I., Philbin, J., Snavely, N., 2016. DeepStereo: learning to predict new views from the world's imagery. In: Proc. IEEE Conference on Computer Vision and Pattern Recognition, pp. 5515–5524.

Fuhrmann, S., Langguth, F., Goesele, M., 2015. Mve – a multi-view reconstruction environment. Comput. Graph. 53, 44–53.

Furukawa, Y., Curless, B., Seitz, S.M., Szeliski, R., 2010. Towards internet-scale multi-view stereo. In: Proc. IEEE Computer Society Conference on Computer Vision and Pattern Recognition, pp. 1434–1441.

Furukawa, Y., Ponce, J., 2009. Accurate, dense, and robust multiview stereopsis. IEEE Trans. Pattern Anal. Mach. Intell. 32, 1362–1376.

Furutani, T., Minami, M., 2021. Drones for disaster risk reduction and crisis response. In: Emerging Technologies for Disaster Resilience, pp. 51–62.

Galliani, S., Lasinger, K., Schindler, K., 2015. Massively parallel multiview stereopsis by surface normal diffusion. In: Proc. IEEE International Conference on Computer Vision, pp. 873–881.

Gallup, D., Frahm, J.M., Mordohai, P., Yang, Q., Pollefeys, M., 2007. Real-time plane-sweeping stereo with multiple sweeping directions. In: Proc. IEEE Conference on Computer Vision and Pattern Recognition, pp. 1–8.

Gallup, D., Frahm, J.M., Pollefeys, M., 2010. Piecewise planar and non-planar stereo for urban scene reconstruction. In: Proc. IEEE Conference on Computer Vision and Pattern Recognition, pp. 1418–1425.

Geiger, A., Roser, M., Urtasun, R., 2011. Efficient large-scale stereo matching. In: Proc. Asian Conference on Computer Vision, pp. 25–38.

Godard, C., Aodha, O.M., Brostow, G.J., 2017. Unsupervised monocular depth estimation with left-right consistency. In: Proc. IEEE Conference on Computer Vision and Pattern Recognition, pp. 270–279.

Godard, C., Aodha, O.M., Firman, M., Brostow, G.J., 2019. Digging into self-supervised monocular depth estimation. In: Proc. IEEE International Conference on Computer Vision, pp. 3828–3838.

Guizilini, V., Ambrus, R., Pillai, S., Raventos, A., Gaidon, A., 2020. 3D packing for self-supervised monocular depth estimation. In: Proc. IEEE Conference on Computer Vision and Pattern Recognition, pp. 2485–2494.

Haala, N., Rothermel, M., Cavegn, S., 2015. Extracting 3D urban models from oblique aerial images. In: Proc. IEEE Joint Urban Remote Sensing Event, pp. 1–4.

Häne, C., Heng, L., Lee, G.H., Sizov, A., Pollefeys, M., 2014. Real-time direct dense matching on fisheye images using plane-sweeping stereo. In: Proc. IEEE International Conference on 3D Vision, pp. 57–64.

Hartmann, W., Galliani, S., Havlena, M., Van Gool, L., Schindler, K., 2017. Learned multi-patch similarity. In: Proc. IEEE International Conference on Computer Vision, pp. 1586–1594.

He, K., Zhang, X., Ren, S., Sun, J., 2016. Deep residual learning for image recognition. In: Proc. IEEE Conference on Computer Vision and Pattern Recognition, pp. 770–778.

Hermann, M., Ruf, B., Weinmann, M., 2021. Real-time dense 3D reconstruction from monocular video data captured by low-cost UAVs. Int. Arch. Photogram. Rem. Sens. Spatial Inf. Sci. XLIII-B2–2021, 361–368.

Hermann, M., Ruf, B., Weinmann, M., Hinz, S., 2020. Self-supervised learning for monocular depth estimation from aerial imagery. ISPRS Annals of the Photogrammetry. Remote Sens. Spat. Inf. Sci. V-2–2020, 357–364.

Hernandez-Juarez, D., Chacón, A., Espinosa, A., Vázquez, D., Moure, J.C., López, A.M., 2016. Embedded real-time stereo estimation via semi-global matching on the gpu. Procedia Comput. Sci. 80, 143–153.

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

Hirschmueller, H., 2008. Stereo processing by semi-global matching and mutual information. IEEE Trans. Pattern Anal. Mach. Intell. 30, 328–341.
Huang, B., Yi, H., Huang, C., He, Y., Liu, J., Liu, X., 2021. M3VSNET: unsupervised multi-metric multi-view stereo network. In: Proc. IEEE International Conference on Image Processing, pp. 3163–3167.
Huang, P.H., Matzen, K., Kopf, J., Ahuja, N., Huang, J.B., 2018. DeepMVS: learning multi-view stereopsis. In: Proc. IEEE Conference on Computer Vision and Pattern Recognition, pp. 2821–2830.
ISPRS Scientific Initiative UseGeo, 2023a. UAV-based multi-sensor datasets for geospatial research – ISPRS Scientific Initiative 2021-2022. URL: https://usegeo.fbk.eu/. Aug 2023.
ISPRS Scientific Initiative UseGeo, 2023b. UseGeo – UAV-based multi-sensor dataset for geospatial research. URL: https://github.com/3DOM-FBK/usegeo. Aug 2023.
Jaderberg, M., Simonyan, K., Zisserman, A., Kavukcuoglu, K., 2015. Spatial transformer networks. In: Proc. Advances in Neural Information Processing Systems, pp. 2017–2025.
Kang, S.B., Szeliski, R., Chai, J., 2001. Handling occlusions in dense multi-view stereo. In: Proc. IEEE Computer Society Conference on Computer Vision and Pattern Recognition, pp. 103–110.
Kazhdan, M., Hoppe, H., 2013. Screened Poisson surface reconstruction. ACM Trans. Graph. 32, 1–13.
Kerle, N., Nex, F., Gerke, M., Duarte, D., Vetrivel, A., 2020. UAV-based structural damage mapping: a review. ISPRS Int. J. Geo-Inf. 9.
Kern, A., Bobbe, M., Khedar, Y., Bestmann, U., 2020. OpenREALM: real-time mapping for unmanned aerial vehicles. In: Proc. International Conference on Unmanned Aircraft Systems, pp. 902–911.
Khot, T., Agrawal, S., Tulsiani, S., Mertz, C., Lucey, S., Hebert, M., 2019. Learning unsupervised multi-view stereopsis via robust photometric consistency. In: Proc. IEEE Conference on Computer Vision and Pattern Recognition Workshops.
Knöbelreiter, P., Vogel, C., Pock, T., 2018. Self-supervised learning for stereo reconstruction on aerial images. In: Proc. IEEE International Geoscience and Remote Sensing Symposium, pp. 4379–4382.
Lowe, D.G., 2004. Distinctive image features from scale-invariant keypoints. Int. J. Comput. Vis. 60, 91–110.
Madhuanand, L., Nex, F., Yang, M.Y., 2021. Self-supervised monocular depth estimation from oblique uav videos. ISPRS J. Photogrammetry Remote Sens. 176, 1–14.
Mahjourian, R., Wicke, M., Angelova, A., 2018. Unsupervised learning of depth and ego-motion from monocular video using 3D geometric constraints. In: Proc. IEEE Conference on Computer Vision and Pattern Recognition, pp. 5667–5675.
Menze, M., Geiger, A., 2015. Object scene flow for autonomous vehicles. In: Proc. IEEE Conference on Computer Vision and Pattern Recognition, pp. 3061–3070.
Mildenhall, B., Srinivasan, P.P., Tancik, M., Barron, J.T., Ramamoorthi, R., Ng, R., 2021. Nerf: representing scenes as neural radiance fields for view synthesis. Commun. ACM 65, 99–106.
Moulon, P., Monasse, P., Perrot, R., Marlet, R., 2016. OpenMVG: open multiple view geometry. In: Proc. International Workshop on Reproducible Research in Pattern Recognition, pp. 60–74.
Nex, F., Stathopoulou, E.K., Remondino, F., Yang, M.Y., Madhuanand, L., Yogender, Y., Alsadik, B., Weinmann, M., Jutzi, B., Qin, R., 2024. UseGeo – a UAV-Based Multi-Sensor Dataset for Geospatial Research (submitted for publication).
Nex, F., Zhang, N., Remondino, F., Farella, E.M., Qin, R., Zhang, C., 2023. Benchmarking the extraction of 3d geometry from uav images with deep learning methods. Int. Arch. Photogrammet. Remote Sens. Spat. Inf. Sci. XLVIII-1/W3-2023 123–130. https://doi.org/10.5194/isprs-archives-XLVIII-1-W3-2023-123-2023. URL: https://isprs-archives.copernicus.org/articles/XLVIII-1-W3-2023/123/2023/.
Pepe, M., Alfio, V.S., Costantino, D., 2022. UAV platforms and the SfM-MVS approach in the 3d surveys and modelling: a review in the cultural heritage field. Appl. Sci. 12, 12886.
Pollefeys, M., Nistér, D., Frahm, J.M., Akbarzadeh, A., Mordohai, P., Clipp, B., Engels, C., Gallup, D., Kim, S.J., Merrell, P., Salmi, C., Sinha, S., Talton, B., Wang, L., Yang, Q., Stewénius, H., Yang, R., Welch, G., Towles, H., 2008. Detailed real-time urban 3D reconstruction from video. Int. J. Comput. Vis. 78, 143–167.
Ranftl, R., Bochkovskiy, A., Koltun, V., 2021. Vision transformers for dense prediction. In: Proc. IEEE International Conference on Computer Vision, pp. 12179–12188.
Ronneberger, O., Fischer, P., Brox, T., 2015. U-Net: convolutional networks for biomedical image segmentation. In: Proc. International Conference on Medical Image Computing and Computer-Assisted Intervention, pp. 234–241.
Rothermel, M., Wenzel, K., Fritsch, D., Haala, N., 2012. SURE: photogrammetric surface reconstruction from imagery. In: Proc. LowCost3D Workshop.
Ruf, B., 2022. Fast Dense Depth Estimation from UAV-Borne Aerial Imagery for the Assistance of Emergency Forces. Karlsruhe Institute of Technology. Ph.D. thesis.
Ruf, B., Weinmann, M., Hinz, S., 2021. FaSS-MVS – fast multi-view stereo with surface-aware semi-global matching from uav-borne monocular imagery. arXiv preprint arXiv:2112.00821v1.
Schönberger, J.L., Frahm, J.M., 2016. Structure-from-motion revisited. In: Proc. IEEE Conference on Computer Vision and Pattern Recognition, pp. 4104–4113.
Schönberger, J.L., Zheng, E., Pollefeys, M., Frahm, J.M., 2016. Pixelwise view selection for unstructured multi-view stereo. In: Proc. European Conference on Computer Vision, pp. 501–518.
Schöps, T., Schönberger, J., Galliani, S., Sattler, T., Schindler, K., Pollefeys, M., Geiger, A., 2017. A multi-view stereo benchmark with high-resolution images and multi-camera videos. In: Proc. IEEE Conference on Computer Vision and Pattern Recognition, pp. 3260–3269.
Seitz, S., Curless, B., Diebel, J., Scharstein, D., Szeliski, R., 2006. A comparison and evaluation of multi-view stereo reconstruction algorithms. In: Proc. IEEE Conference on Computer Vision and Pattern Recognition, pp. 519–528.
Shen, S., 2013. Accurate multiple view 3D reconstruction using patch-based stereo for large-scale scenes. IEEE Trans. Image Process. 22, 1901–1914.
Sinha, S.N., Scharstein, D., Szeliski, R., 2014. Efficient high-resolution stereo matching using local plane sweeps. In: Proc. IEEE Conference on Computer Vision and Pattern Recognition, pp. 1582–1589.
Sinha, S.N., Steedly, D., Szeliski, R., 2009. Piecewise planar stereo for image-based rendering. In: Proc. IEEE International Conference on Computer Vision, pp. 1881–1888.
Snavely, N., Seitz, S.M., Szeliski, R., 2006. Photo tourism: exploring photo collections in 3d. In: Proc. ACM Siggraph, pp. 835–846.
Spangenberg, R., Langner, T., Adfeldt, S., Rojas, R., 2014. Large scale semi-global matching on the CPU. In: Proc. IEEE Intelligent Vehicles Symposium, pp. 195–201.
Sun, C., Sun, M., Chen, H.T., 2022. Direct voxel grid optimization: super-fast convergence for radiance fields reconstruction. In: Proc. IEEE Conference on Computer Vision and Pattern Recognition, pp. 5459–5469.
Tancik, M., Casser, V., Yan, X., Pradhan, S., Mildenhall, B., Srinivasan, P.P., Barron, J.T., Kretzschmar, H., 2022. Block-nerf: scalable large scene neural view synthesis. In: Proc. IEEE Conference on Computer Vision and Pattern Recognition, pp. 8248–8258.
Tsoraeva, E.N., Gadzhiev, R.K., Kuchiev, S.E., Pekh, A.A., Mezhyan, S.A., 2021. Application of photogrammetric methods in architecture, construction and land management. IOP Conf. Ser. Mater. Sci. Eng. 1083, 012052.
Turki, H., Ramanan, D., Satyanarayanan, M., 2022. Mega-nerf: scalable construction of large-scale nerfs for virtual fly-throughs. In: Proc. IEEE Conference on Computer Vision and Pattern Recognition, pp. 12922–12931.
Wang, C., Buenaposada, J.M., Zhu, R., Lucey, S., 2018. Learning depth from monocular videos using direct methods. In: Proc. IEEE Conference on Computer Vision and Pattern Recognition, pp. 2022–2030.
Wu, C., 2011. Visualsfm: a visual structure from motion system. URL: http://ccwu.me/vsfm/. Aug 2023.
Wu, C., 2013. Towards linear-time incremental structure from motion. In: Proc. International Conference on 3D Vision, pp. 127–134. IEEE.
Xie, J., Girshick, R., Farhadi, A., 2016. Deep3D: fully automatic 2d-to-3d video conversion with deep convolutional neural networks. In: Proc. European Conference on Computer Vision, pp. 842–857.
Xu, Q., Kong, W., Tao, W., Pollefeys, M., 2022. Multi-scale geometric consistency guided and planar prior assisted multi-view stereo. IEEE Trans. Pattern Anal. Mach. Intell. 45, 4945–4963.
Xu, Q., Tao, W., 2019. Multi-scale geometric consistency guided multi-view stereo. In: Proc. IEEE Conference on Computer Vision and Pattern Recognition, pp. 5483–5492.
Xu, Q., Tao, W., 2020. Planar prior assisted patchmatch multi-view stereo. In: Proc. AAAI Conference on Artificial Intelligence, pp. 12516–12523.
Yao, Y., Luo, Z., Li, S., Fang, T., Quan, L., 2018. MVSNet: depth inference for unstructured multi-view stereo. In: Proceedings of the European Conference on Computer Vision, pp. 767–783.
Zbontar, J., LeCun, Y., 2016. Stereo matching by training a convolutional neural network to compare image patches. J. Mach. Learn. Res. 17, 2287–2318.
Zhang, J., Xu, S., Zhao, Y., Sun, J., Xu, S., Zhang, X., 2023a. Aerial orthoimage generation for UAV remote sensing: review. Inf. Fusion 89, 91–120.
Zhang, N., Nex, F., Vosselman, G., Kerle, N., 2023b. Lite-mono: a lightweight cnn and transformer architecture for self-supervised monocular depth estimation. In: Proc. IEEE Conference on Computer Vision and Pattern Recognition, pp. 18537–18546.
Zhao, H., Gallo, O., Frosio, I., Kautz, J., 2016. Loss functions for image restoration with neural networks. IEEE Trans. Comput. Imag. 3, 47–57.
Zhao, J., Liang, T., Feng, L., Ding, W., Sinha, S., Zhang, W., Shen, S., 2020. FP-Stereo: hardware-efficient stereo vision for embedded applications. In: Proc. IEEE International Conference on Field-Programmable Logic and Applications, pp. 269–276.
Zhao, Y., Chen, L., Zhang, X., Xu, S., Bu, S., Jiang, H., Han, P., Li, K., Wan, G., 2022. RTSfM: real-time structure from motion for mosaicing and DSM mapping of sequential aerial images with low overlap. IEEE Trans. Geosci. Rem. Sens. 60, 5607415.
Zheng, E., Dunn, E., Jojic, V., Frahm, J.M., 2014. PatchMatch based joint view selection and depthmap estimation. In: Proc. IEEE Conference on Computer Vision and Pattern Recognition, pp. 1510–1517.