Does it work outside this benchmark? Introducing the rigid depth constructor tool

Clément Pinard, Antoine Manzanera

Noname manuscript No.
(will be inserted by the editor)

Does it work outside this benchmark? Introducing the Rigid Depth Constructor tool

Depth validation dataset construction in rigid scenes for the masses

Clément Pinard · Antoine Manzanera

Received: date / Accepted: date

Abstract A new framework called Rigid Depth Constructor (RDC) is proposed, allowing a user to create his own dataset for the validation of depth map estimation algorithms in the context of autonomous navigation. Compared to the existing tools that rely on high quality fixed Lidar sensor, RDC is usable in low-cost setups requiring only a camera and any (e.g. handheld, or UAV-carried) Lidar sensor, which implies more flexible - and much faster - scene scan. Furthermore, unlike photogrammetry tools that use sparse RGB views, it can be applied to smooth videos while remaining computationally tractable. The framework includes a test suite to get insightful information from the evaluated algorithm. As examples, validation videos made from UAV footage are provided to evaluate two depth prediction algorithms initially tested on in-car driving video datasets, which shows that the drone context is dramatically different. This supports the need to benchmark depth estimation algorithms on a dataset that fits one's particular context, which often means creating a brand new one. An open source implementation accompanies the paper, designed to be as user-friendly as possible, to make depth dataset creation possible even for small teams. The key contributions are the following: (1) a complete, open-source and almost fully automatic software application for creating validation datasets with densely annotated depth, adaptable to a wide variety of image, video and range data; (2) selection tools to adapt the dataset to specific validation needs, and conversion tools to other dataset formats; (3) as use case examples, two new real datasets, outdoor and indoor, readily usable in UAV navigation context are provided, and used as

Clément Pinard
U2IS, ENSTA Paris, Institut Polytechnique de Paris
828, Boulevard des Maréchaux, 91762 Palaiseau Cedex
E-mail: clement.pinard@ensta-paris.fr

Antoine Manzanera
U2IS, ENSTA Paris, Institut Polytechnique de Paris
828, Boulevard des Maréchaux, 91762 Palaiseau Cedex
E-mail: antoine.manzanera@ensta-paris.fr

test sets in the evaluation of two depth prediction algorithms, using a collection of comprehensive (e.g. distribution based) metrics.

Keywords Monocular Depth Estimation · Validation Dataset Construction · Depth Evaluation Metrics

1 Introduction

Using computer vision for navigation has long been well established, as a camera sensor is very easy to set up, cheap and power efficient. The main uses are for odometry and 3D maps which are then used to control the navigation, especially find a path and avoid obstacles.

Estimating depth from a camera is not an easy task, and validation data is very hard to obtain. Indeed, knowing depth requires to know the 3-dimensional environment the camera is facing with respect to its position. This requires to explicitly measure depth with range sensors like Lidar or RGB-d cameras.

A major example of depth validation dataset is KITTI [1], where a set of cameras and a Lidar are mounted on a car. Following the acquisition, the Lidar and video signals are calibrated and synchronized in order to construct sparse depth maps for every camera at every moment. The main problem with this method is that you need to construct a rigid rig between a Lidar and a camera, which, in addition to being very costly, can become very heavy, and is not suitable to recreate the natural movement of a handheld camera or a consumer UAV camera. Other datasets have been built using RGB-d cameras based on different technologies like Structured light or Time-of-Flight [2], but their use is limited to indoor and/or short range applications, due to visibility constraints of the projected light pattern, or to phase ambiguities in the periodic light signals.

To address these problems, a new software tool is proposed to construct a depth dataset with a two-step method that first uses a Lidar to scan an environment, and then localizes images from a video camera with respect to the Lidar point cloud. Its goal is to be the most user friendly possible, and with maximal flexibility, both on the methods to construct the point cloud, and on the type of camera used for acquisition.

The flexibility regarding point clouds implies that the tool should work with the most generic point cloud: points are not colored, and they are not structured. This means that there is no information of outside or inside, or even from which point of view each point is visible. As such, it is very hard to estimate the local orientation of the surface (which is made by calculating a normal vector at each point of the cloud), and then to compute a mesh from the point cloud. This format, where only position of points is known is a possible output format when using proprietary scanning gear or measurements made by a team of professionals.

In addition to the paper, an open source package is provided that has been thoroughly tested with an industrial research team to ensure its usability. The tool is used in this paper to build datasets corresponding to two different UAV use cases. These datasets are used to show, with a benchmark on monocular depth estimation algorithms, that results can vary greatly depending on the context, and that the in-car environment - well referenced through the KITTI benchmarks - is far from being

representative of all navigation use cases. This means that for each new navigation scenario, a new dataset should be constructed, at least for validation, which is exactly what the proposed tool aims at making easier and cheaper.

The goal of the Rigid Depth Constructor (RDC) tool proposed in this paper is then to help answer the question asked in the title: "does it work outside this benchmark?", that is to say "will it work *as well* in my use case?". It allows a user to build, with minimal cost and effort, a validation dataset of image sequences with dense depth ground truth annotation, on the environment corresponding to his particular use case, so that he can actually evaluate a depth map estimation algorithm, whose performance metrics are generally provided on a standard benchmark, which can differ much from the user's need.

The contributions of the RDC framework are listed as follows:

- The tool itself: a complete, open-source and almost fully automatic software application for creating validation datasets with densely annotated depth, adaptable to a wide variety of image, video and range data
- Some additional tools, like: selection functions to adapt the dataset to specific validation needs, or conversion functions to other dataset formats
- As use case examples: two new real datasets, outdoor and indoor, readily usable in UAV navigation context
- A collection of comprehensive (e.g. distribution based) metrics, whose use is illustrated in the evaluation of two depth prediction algorithms on the two previous test sets

The remainder of the paper is organized as follows: Section 2 presents the existing softwares that can be used for building depth validation datasets, and points out their limitations and constraints. Section 3 details the construction pipeline of RDC, built on the combination and adaptation of existing tools. Section 4 presents the different quality metrics that can be used in the evaluation, discusses the existing ones, and proposes new ones. Section 5 presents results on two different use cases - one outdoor and one indoor - and for two aspects: first in the construction of the validation datasets, and second in the evaluation of two depthmap prediction algorithms on these datasets. Finally Section 6 concludes the paper and discusses the remaining limitations and possible future works.

2 Related works

2.1 Depth estimation for navigation

Depth estimation, and more generally 3D perception is a core task for autonomous navigation. In the context of very light vehicles such as UAV, that can't carry heavy hardware, depth is often deduced from one or multiple cameras. The stereo camera is often used to compute depth from disparity [3], that can also be estimated with a single camera, which has the advantage of being much cheaper to integrate. Depth can then be deduced from motion, by using methods based on epipolar geometry [4]. Structure from motion and SLAM techniques can then be used to deduce both

depth and camera movement using only optical flow and geometric equations. It must be noted that all structure from motion algorithms require rigid scenes, i.e. without moving objects.

More recently, depth inference networks have been shown to be able to estimate depth from a single image solely from perspective and context. Indeed, with only one image, they were able to get reasonable scale invariant (i.e. relative depth maps) quality measures [5, 6, 7, 8]. However, in a navigation context, the scale invariant quality is not really interesting without a way to link the estimation to the real world.

Instead of relative or normalized values, absolute depths are needed for a navigation algorithm such as Model Predictive Control [9] to be used. To this end, depth algorithms need to estimate both depth *and camera displacement* in order to work for navigation. Following Zhou *et al* [10], different authors have used photometric reconstruction error as a self-supervised training loss to predict the relative pose of the camera [11, 12, 13, 14].

It is important to note that depth from a single image does not use pixel displacements. It is deemed a more bio-inspired method which makes use of end-to-end training of convolutional neural networks and their capacity of generalizing implicit pixel structure (see also [15], that uses scale parameter from SIFT points to relate local appearance to distance) but it does not rely on explicit geometric constraints. As such, it is expected to be less robust for unusual scene, especially with an ambiguous 3d perspective.[1]

2.2 Validation sets and benchmarks: the specific case of consumer UAV cameras, and the need for a new dataset

This work was mostly motivated by the specific use case of depth estimation from UAV consumer drone, which lacks a proper validation dataset. Indeed, most depth algorithms are currently tested either for autonomous driving environments such as KITTI [16], or indoor environment such as NYUv2 [17]. In these evaluation frameworks, single image depth prediction algorithms heavily prevail within the leaderboard, which makes these techniques *de facto* state of the art.

However, those datasets are extremely distinctive in terms of appearance and context: KITTI images are always in-car viewed road scenes, with peculiar perspective and no pose changes; NYUv2 images are exclusively indoor room scenes viewed by a standing adult, with many straight lines and human artefacts. On the other hand, the context of UAV navigation is much more heterogeneous, both in terms of camera pose and surrounding environment. However, contrary to KITTI, the problem of moving objects is deemed (at least for now) a secondary problem when flying high above the ground. It is thus not certain that algorithms performing well on KITTI or NYUv2 will not perform poorly in this context. This is especially true when considering that moving objects in KITTI make depth from motion much harder, while depth from context is inherently sensitive to viewpoint variations [18]. As such, these

[1] i.e. when the 3d scene and/or the camera position produces an image where perception of distance or object sizes is ambiguous, or even deceptive (famous extreme examples are the Ame's room or the Corridor illusion).

	KITTI[16]	NYUv2[17]	Sintel Depth[19]	Still Box [21]	a good UAV dataset
Moving objects	✓	✗	✓	✗	✗
Outdoor	✓	✗	✓	N/A	✓
Camera orientation variation	✗	✓	✓	✓	✓
Camera position variation	✗	✗	✓	✓	✓
Real videos	✓	✓	✗	✗	✓

Table 1 Difficulties featured in existing datasets. Being non photo-realistic, the distinction between Indoor and Outdoor is irrelevant for Still Box.

datasets address issues for particular use cases with their own difficulties that are not reflected in the UAV use case and vice-versa. This idea is corroborated by the fact that the Sintel depth dataset [19] remains largely unsolved for the moment because in addition to moving objects, scene heterogeneity is much more prevalent.

Inspired by the Flying Chairs synthetic dataset for optical flow[20], which is founded on a surrealistic abstraction of the difficulties of optical flow, Still Box [21] has been proposed. It is a synthetic dataset aiming at simulating the difficulties of a UAV flight, focusing on heterogeneity of appearance, using random shapes and textures. As such, it is a good (fully supervised) training dataset, the same way Flying chairs is a good training dataset for optical flow, but obviously, it is not suited for evaluation. Table 1, which compares the difficulties of the datasets currently used as benchmarks for depth quality, shows that none of them is actually relevant for the UAV camera use case.

2.3 Constructing a depth enabled dataset

The basic principle of depth enabled dataset construction is to use a device with reliable depth estimation capability, that won't be available during evaluation, typically a rig with an RGB camera and a depth sensor like structured light [17], Time of Flight, embedded Lidar [16, 22], or light-field camera grids [23]. For evaluation, only the camera will be available, and the evaluation step will then measure the agreement between "reliable depth" measured by the dedicated sensor and estimated depth. It is important to note that an evaluation is only informative up to a certain point, where the quality of both methods are comparable. For dedicated depth sensors, one usually relies on the vendor's datasheet. However, this solution requires the simulation of camera movements typical of the targeted use case, *i.e.* without potentially heavy depth sensors. This is problematic for consumer UAV, because such movements are difficult to mimic. Not only the camera is well stabilized, with a very smooth trajectory, that is not reproducible by hand, but the size of these cameras make it easy to fly very close to obstacles, which is not reproducible by a heavier UAV that could carry an additional depth sensor. These kinds of UAVs are usually very dangerous and need to be operated far from obstacles.

An interesting method for rigid scenes has been presented with ETH3D [24] and *Tanks and Temples* [25] that usually serve to evaluate photogrammetry. Instead of having a depth sensor and a camera attached to the same rig, the data acquisition is done in two steps. They first get a point cloud measure from static tripod laser scanners such as the FARO Focus, and then take images from cameras in the same

scene after removing the laser scanner. This implies that no object is moving in the scene, but allows for any camera to be used. However, with this technique, the Lidar needs to be static and render colored point clouds. This makes the use of mobile scanners impossible, and thus the data acquisition very long. Gollob [26] compared static scanning and handheld mobile scanning in a forest case study, and established that static scanning was up to five times slower than mobile scannng with handheld lidar scanners. The difference would be even greater for mobile scanning with a dedicated UAV or ground vehicle. In the particular case of ETH3D, they added a photogrammetry step, using the taken pictures and some RGB-d rendering of the Lidar scanner with the COLMAP software [27]. This allowed them to get frame localization with respect to each other, but also with respect to the Lidar scanner position. With this technique, they were able to get depth map indirectly from the Lidar point cloud. They then used the generated depth map and camera position to get a stereo validation, with rectified frames. The ETH3D dataset construction method has been used as a foundation of our framework.

To summarize, the constraints / limitations of the existing works for building depth validation dataset, and that the RDC framework proposed in this paper aims at overcoming are the following:

- the need for a fixed, heavy, or potentially dangerous Lidar system
- the limitation to large / medium distances adapted to photogrammetry applications, preventing its use for medium to short distances. However, the whole range of distances is necessary for the navigation of light autonomous vehicles.
- the need for colored / rendered point clouds in the case of ETH3D, which are unavailable in many Lidar systems.
- the acquisition and computation costs, more adapted to sparse images than smooth videos.

The goal of the RDC framework is to generalize the ETH3D method not only for photogrammetry oriented footage, but for all kinds of videos, using anything available to acquire a 3D point cloud of a scene.

3 Dataset creation method

3.1 The foundations: COLMAP and ETH3D

ETH3D already offers to compute depth maps from a Lidar point cloud and a separate camera, but it has been used in a very particular context and some of its methods are not suited for a more general one. Since the proposed RDC framework uses the same tools as ETH3D, they are now analyzed to see what needs to be changed in order to increase flexibility.

3.1.1 Photogrammetry with COLMAP

COLMAP [28, 27] is a photogrammetry tool designed to be very robust, in order to reconstruct a 3D model from crowdsourced images taken from the internet. To

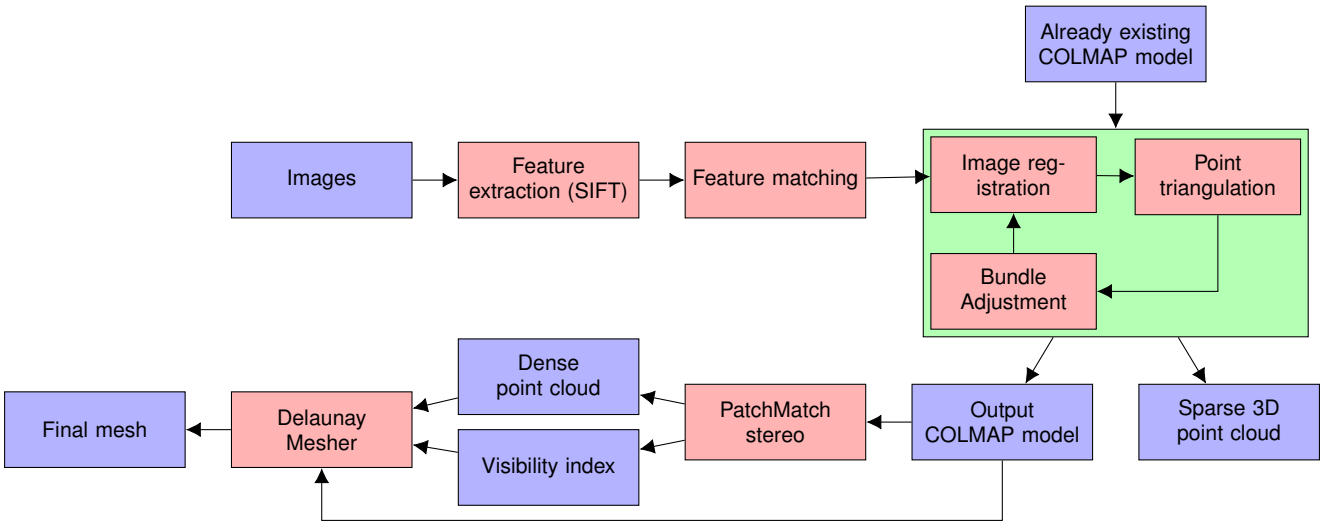


Fig. 1 Photogrammetry workflow used in COLMAP. Blue boxes represent data inputs and outputs, red boxes are operators. The green box stands for the mapping process which is a looped incremental procedure. Note that the mapping process can stop at "Image registration" and the Delaunay mesher accepts any point cloud as long as the visibility index is correct.

go from a set of images to a 3D model with images localization, the steps applied are presented in Fig. 1. The main problem of COLMAP is that even if the matching process can be dramatically accelerated with vocabulary tree matching [29], it is still in $\mathcal{O}(n^3)$ where n is the number of images. As such, the reconstruction process is practically intractable for videos.

3.1.2 Depth generation with ETH3D

ETH3D [24] uses COLMAP to localize calibrated images with respect to a Lidar point cloud taken with a FARO Focus. The FARO Focus is a fixed point Lidar that renders colored 3D points. Since it is fixed, every 3D point is measured from the same origin. This device allows then to synthesize high quality depth-valued images with known position of the scanner (*i.e.* the cloud origin) that can be integrated in the COLMAP reconstruction process. The position of each image is thus known with respect to the point cloud. Each image then gets its position refined by matching feature points between real images and rendered images from the colored point cloud of an equivalent camera at the estimated position. But this localization method is not usable in many use cases that have only access to basic point clouds, *i.e.* without colors. Nevertheless, as stated in their paper, the localization step can be done when simply constructing a 3D model with COLMAP and then register it with respect to the Lidar point cloud, with *e.g.* ICP [30].

Once the images are localized with respect to Lidar, the depth rendering part is detailed in Fig. 2. Mainly, in addition to image calibration and localization, an occlusion mesh needs to be computed from the point cloud, which is then used to

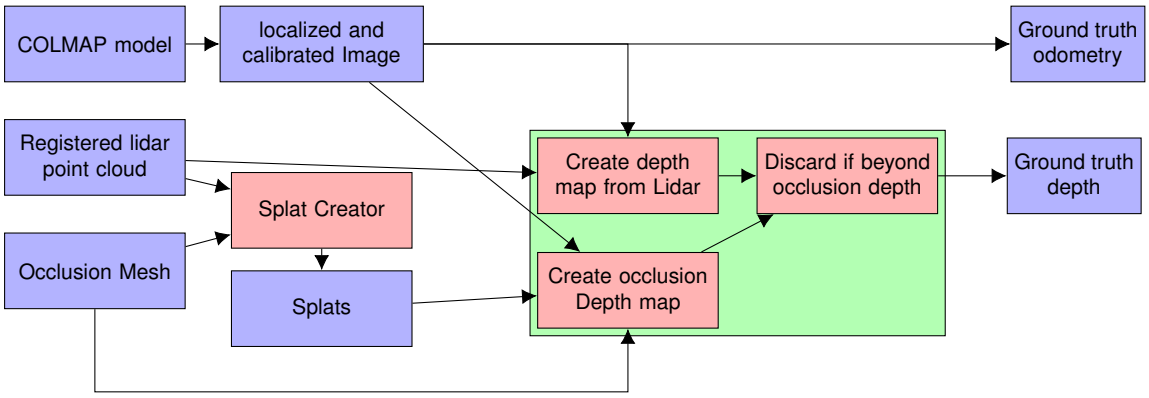


Fig. 2 Photogrammetry workflow used with ETH3D. As for figure 1, blue boxes represent data and red boxes represent operators. The green box here stands for the ground truth creation process. Note that this workflow does not include localization, which is much more complicated but ignored in the RDC framework, which assumes that the point cloud is not colored.

construct an occlusion aware depth map. This depth map is significantly worse than the one from the point cloud, but it is only used to determine occlusion (and thus point visibility) and then avoid the ghosting effect of seeing through a 3D object due to the sparsity of the point cloud. This occlusion mesh can be improved by the use of ”splats”, which are created from isolated points: assuming such points are representative of thin objects (such as the leg of a chair), they may not be represented by the occlusion mesh but still count as occluding point. The splat creator then puts an oriented square facet at each point position, in order to avoid the risk of rendering depth values of a background object for the rest of the thin object.

In ETH3D[24], the occlusion mesh is constructed with Poisson algorithm [31]. This supposes that normal vectors can be computed for all points. Again, this is possible with point clouds coming from a fixed laser scanner, where normals are always oriented toward the origin, but it is not always the case for generic point clouds, for which alternative solutions have to be found.

3.2 Changing the ETH3D workflow

The constraints from the ETH3D use case context can be summarized this way:

- The number of images for the typical use case is very small. In contrast, for autonomous navigation uses cases, that use video streams, thousands of frames must be localized, which represents a redhibitory computational cost.
- The Lidar is a very high quality color-enabled fixed scanner, which makes the registration and the meshing processes easier, but also implies acquisition protocol much heavier and longer. In contrast, the RDC framework must be able to use any collection of generic unordered point clouds, possibly acquired from handheld or UAV-carried scanners. devices allows for a much faster and cheaper acquisition, but the lower quality makes the computation of the mesh much harder.

So the purpose is to find a way to localize images with COLMAP in linear time while keeping a good 3D reconstruction, and a way to construct a good mesh from unordered point clouds.

3.2.1 Managing the COLMAP reconstruction complexity

The issue regarding the number of images can be solved by simply using a subset of images for reconstruction. The reconstruction needs to be good enough to be precisely registered with respect to the Lidar point cloud. For a small subset of frames, the full structure from motion mapping process can be applied even if it is very expensive. However, for remaining frames, the 3D point cloud will only be marginally better since their views are supposedly already covered by nearby frames. It then becomes manageable to only perform registration with respect to the already existing reconstruction. This process is much less expensive, as it does not need a global bundle adjustment at each frame.

The issue is now to choose a good subset of frames. This can be solved during the data acquisition step, by taking a set of pictures dedicated to photogrammetry following guidelines in [32]. The goal is, for a fixed number of pictures to be used in the mapping process, to have pictures with the most uniformly distributed position and orientation view points. This can be done for example with a UAV orbiting around a particular object or flying along a grid above the scene: pictures are sampled at a regular pace to ensure a good parallax between images.

3.2.2 Constructing the Occlusion mesh

If every image has been successfully localized with respect to the Lidar point cloud, a specific tool can be used to construct a mesh. After point cloud densification with multi view synthesis and depth maps fusion, COLMAP outputs a point cloud with normals and a visibility index indicating from which frame each 3D point is visible. These two features are used for mesh reconstruction. Indeed, as discussed earlier, Poisson reconstruction [31] can be used thanks to normals, and Delaunay meshing [33] can be used with visibility index. Under the (unrealistic) assumption that both dense reconstruction from COLMAP and Lidar point cloud are perfect, these features can be easily transferred from one cloud to another. Although COLMAP is known to have a low recall, by discarding many points in textureless areas, it has been observed in our experiments that transferring feature from the nearest neighbor of each Lidar point was sufficient.

3.3 RDC's final workflow

The final version of the workflow can be found in Figure 3, that represents the complete algorithm needed for ground truth creation from ETH3D. It is now developed in details.

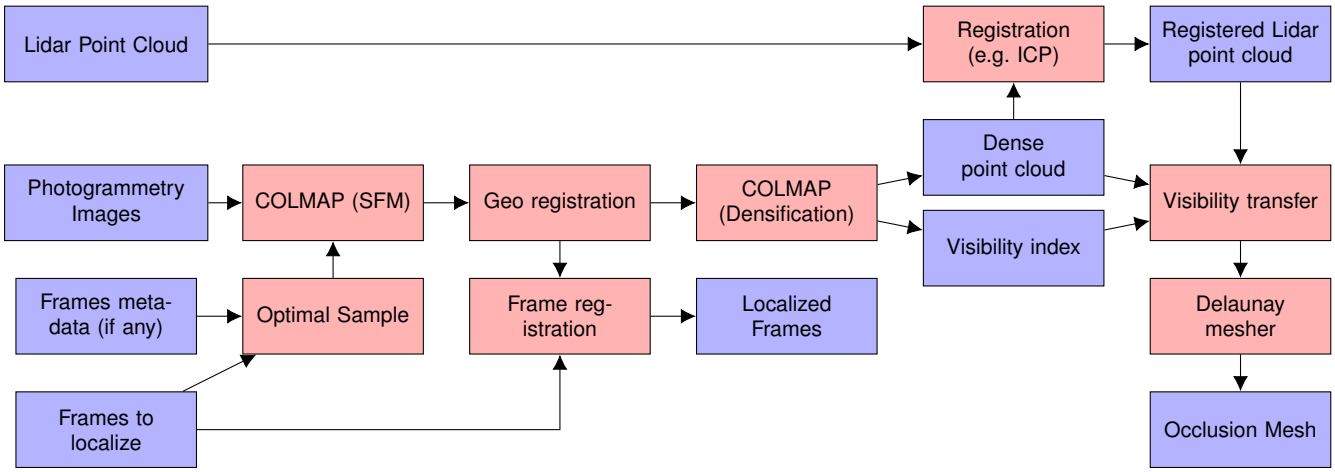


Fig. 3 Simplified representation of RDC's workflow before using ETH3D tools. As for figure 1, blue boxes represent data and red boxes represent operators. Video frames get registered with respect to the reconstruction point cloud, along with the Lidar point cloud. As such, ETH3D can be used with the COLMAP model, the Occlusion mesh and the registered Lidar point cloud.

3.3.1 Data acquisition and ground truth point cloud creation

In order to maximize the accessibility of the RDC framework, the acquisition protocol is made as unconstrained as possible, and only asks for a point cloud of the scene, acquired by any mean, and not necessarily colored. The point cloud will be used as a perfect ground truth, which means that subsequently constructed depth maps can only be as precise as this point cloud. As suggested by [24], a tripod fixed Lidar scanner can be used in order to have the maximum precision and density (precision at the millimeter level). However, other Lidar sensors can be used, such as mobile Lidars attached to a UAV, or a human handle. Although they lack precision (which is now at the centimeter level), their ease of use can be leveraged to have a much more complete point cloud, especially in cluttered environments or unreachable places like a building's roof. As a last resort, when no Lidar is available, even the result of a photogrammetry can be used, in order to compare a real-time depth algorithm to a thorough reconstruction algorithm that does not sacrifice quality for speed. Throughout this section, the point cloud that serves as a ground truth is referred to as Lidar point cloud, but everything applies the same for any other ground truth point cloud creation method.

3.3.2 First thorough photogrammetry

This first step uses COLMAP to construct a photogrammetry comprising a point cloud and the viewpoint position of every image that was successfully localized. The reconstructed point cloud will then be localized with respect to the ground truth point

cloud. Note that this step is different from ETH3D since the point cloud is not supposed to be colored, and then cannot be used to synthesize images from Lidar data.

As said above, the goal is to make model reconstruction as efficient as possible. If all the video frames had to be localized, the reconstruction would be comprehensive, but also extremely long. Instead, in addition to the "photogrammetry frames", only a subset of each video sequence is input to the photogrammetry process, so that the reconstruction is not too long while including sufficient number of different view points. The sampling can be based on frame rate (e.g. only take one frame per second), but it is not ideal when the camera velocity is not constant. An interesting solution could be to use a method similar to real-time SLAM systems like ORB-SLAM [34] where a subset of key frames is constructed progressively during the reconstruction. Frame that are too similar from already existing key frames won't be used for the reconstruction, and only for odometry. This solution was not considered for this tool as this would have needed heavy modifications in COLMAP's core algorithm.

However, for some camera devices like drones or IMU enabled cameras, video frames will include displacement or position metadata. A more efficient sub-sampling can thus be obtained by using K-means [35] on a 6D point cloud composed of frames positions and orientations obtained from metadata. Note that the importance of orientation in sampling can be parameterized by normalizing each angle coordinate α by $R\tan(\alpha)$, where R is the weight corresponding to the typical view range of the sequence.

3.3.3 Registration of ground truth point cloud with respect to the output of photogrammetry

This step requires to find the optimal rigid transformation (rotation, translation and scale) between the reconstructed point cloud and the Lidar point cloud. Assuming COLMAP's reconstruction is good enough, a simple ICP [30] or related algorithm like point-to-plane ICP [36] can be used to align the two point clouds. It is probably the most sensitive step of the process, that often requires a human supervision.

Indeed, ICP is a somewhat unstable process that needs assessment and a good initialization. As such, a human generally needs to thoroughly check the point cloud alignment and, sometimes, manually estimate a first rough transformation, with *e.g.* point pair picking. This can be done by *e.g.* meshlab [2] or Cloudcompare [3]. Although this actually prevents the process from being fully automatic, it is essential to emphasize that it is the only step of the whole process that can require human supervision, and that it is not expected to take more than a few minutes.

3.3.4 Video localization

This step only uses the image registration tool from COLMAP applied to the existing reconstruction. It does not contribute to the point cloud reconstruction, and only uses local bundle adjustment to localize the frame with respect to its neighbors. As

[2] https://www.meshlab.net/
[3] http://www.cloudcompare.org/

such, this operation is much faster than the whole reconstruction process, where point triangulation and global bundle adjustment was used.

3.3.5 Localization filtering

A significant advantage regarding COLMAP compared to other SLAM method is that everything is global. This way every frame is not only localized with respect to its neighbors, but also with respect to any other frame with which it shares a sufficient field of view. As such, if each video frame contains features matched in enough well globally localized photogrammetry frames, the localization does not drift with respect to the reconstruction. However, since odometry is not perfect either, it can generate a noisy trajectory that would not be possible for a real camera with bounded acceleration. This is especially true for consumer UAVs which usually focus on video smoothness using a gimbal stabilizer for aesthetic purposes.

To reflect this observation, on each localized video, a Savitzky-Golay filter [37] is applied to both trajectory and orientation. This filter not only smoothes the movement, but also helps detecting outlier that were badly localized. This way, we can discard the frames for which the distance between estimated and filtered 6D positions is above a certain threshold, and interpolate their position from neighboring frames. These frames will not be used for depth evaluation, but can be used *e.g.* for depth algorithms that rely on odometry.

3.3.6 Depth and pose ground truth generation

Finally, RDC uses the ground truth creator developed for ETH3D [24] to construct depth maps for all the video frames, so that a ground truth for odometry and depth is obtained for each successfully localized frame.

3.3.7 Dataset conversion and evaluation subset creation

Once obtained images with annotated odometry and dense maps, the dataset format can be converted in compliance with more popular datasets. For example, the same format as KITTI odometry can be used, in order to ease validation of depth algorithms on new datasets.

In addition, the same way as Eigen[8] proposed an evaluation split for depth, the subset of frames actually used for depth evaluation can be defined and adapted - based on odometry information - to different evaluation scenarios:

- Candidate frames can be selected according to the movement, *e.g.* only forward motion (like in the context of a car), or without rotation.
- For algorithms based on normalized (relative) depth with corresponding pose estimation with respect to previous frames (*e.g.* SFMLearner [10]), the scale factor can be solved with displacement magnitude, as suggested in [38].
- For algorithms that need odometry, such as multi view stereo, or algorithms that need frames with compensated rotation [38], it can be provided. This scenario is

realistic for navigation context in the case of a UAV, where velocity and orientation need to be known primarily for a stable flight and a smooth video, and thus are available for these algorithms.

3.4 Automation

All the above processes have been included into a script that makes extensive use of COLMAP, ETH3D, PCL, and Parrot's Anafi SDK. This script is intended to be as easy to use and as flexible as possible, in order to cover a wide range of use cases and budgets. It is open-sourced on Github with extensive usage documentation. [4]

3.5 How good a ground truth can we hope to reach?

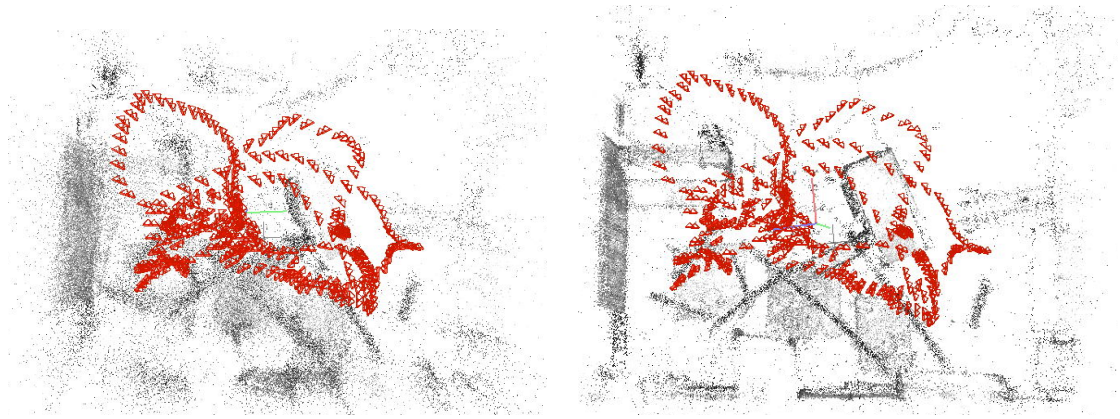
During the whole process, it has been assumed that the 3D point cloud was perfect, or at least the best quality possible, which implies trusting precision ranges given by scanner vendors. For example for a fixed Lidar scanner such as the Faro Focus, the precision is below the centimeter, while for handheld Lidars such as Velodyne VLP16 used either with a UAV or handheld, the precision is below 5 centimeters. This means that depending on the device used for mapping, one needs to pay attention to the depth ranges seen during videos to be localized. This is particularly true for videos very close to obstacles, where the precision of depth maps measured by COLMAP's patch match stereo step can be better than Lidar reference. Using fixed or mobile scanners is then a trade-off between mobility, scan time, precision and completeness.

Regarding odometry, the localization made by COLMAP is assumed to be perfect. It can be noted that this was not the case for ETH3D, where they applied a pose fine tuning for each frame. Unfortunately, their method is only available for colored point clouds, which is not available for most mobile Lidars. This problem can be mitigated by two factors:

- The colorless scanners are also the less precise ones. As such, solving this problem might bring negligible improvement since the point cloud quality will then become the limiting factor. Otherwise, ETH3D's pose refinement technique can simply be applied.
- COLMAP was tested on EuroC dataset, and surprisingly, odometry from the provided ground truth (measured with a Lidar scanner and an IMU) was not very good compared to odometry computed by COLMAP. This can be seen on Fig. 4, where the triangulated 3d points are visibly much less noisy from COLMAP odometry than from Lidar and IMU ground "truth". This is corroborated with the EuroC depth dataset proposed by [39], where synthesized depth maps from frame position and camera calibration were not exactly aligned with the camera, even in their illustrating figure (see Fig. 5).

In sum, there are good, albeit subjective, reasons to think that the odometry from dedicated sensors is not necessarily needed compared to the one computed by COLMAP.

[4] https://github.com/ClementPinard/depth-dataset-builder/



(a) (b)

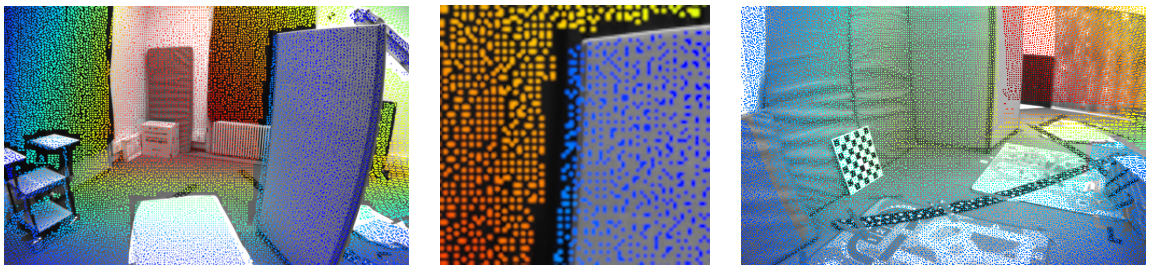
Fig. 4 Visual qualitative assessment from COLMAP mapping process, using COLMAP GUI. (a) localization from available ground truth odometry, measured from Lidar and IMU. (b) localization deduced by COLMAP during the mapping process with SLAM. The black points form the point cloud of the mapping, while the red polyhedra represent the estimated poses of the camera.

This is only based on visual assessment, but as mentioned in the first point, point cloud quality is often the limiting factor. However, it turns out that for some particular cases, such as a video that is isolated in a cluttered part of a scene and only connected to the rest of the photogrammetry by a few frames, the odometry can drift. This makes these localized frames misaligned with the Lidar point cloud, and thus with a poor ground truth depth, even if the Lidar point cloud registration step is optimal, because it's only a rigid transformation. A solution to this problem could be to apply a non-rigid registration [40] of the COLMAP point cloud, deforming the cloud and then also the frame localization to fit the Lidar point cloud more precisely.

This might be the occasion of a future work combining COLMAP's bundle adjustment and cloud-to-cloud distance between COLMAP and Lidar point cloud. However, a more direct way of limiting this problem is to ensure during data acquisition that all video frames can be localized with a large number of photogrammetry oriented pictures, so that a "loop closure" step is applied very often, with pictures designed to have a very precise localization with respect to the reconstruction cloud.

4 Measuring depth quality for navigation purposes

For an evaluation dataset, the most informative metrics are needed. In the present context of navigation, the methodology proposed in [38] is applied. Namely, contrary to the well used "Eigen-split" [8], using metrics from Garg *et al* [41], in a realistic navigation scenario, the estimated depth needs to be absolute, and not up to a factor computed with the ratio of medians. This makes depth from single view algorithms unable to compete, unless there is a way to compute the scale factor by comparing odometry with actual displacement, which is much more realistic for any kind of autonomous vehicle. Single frames algorithms such as Deep Ordinal Regression



(a) (b) (c)

Fig. 5 Depth maps proposed by [39] (Figures from their supplementary materials) showing alignment problems. (a) and (b, zoom detail): foreground/background delimitation error. (c): alignment error larger than 40 pixels.

Network (DORN) [5] or Big to Small (bts) [6] can thus not be used, but some other algorithms such as SFMLearner and its variants [10, 11, 12, 14, 13] are trained with a pose estimator and thus can be evaluated.

4.1 On the information a metric gives

Most of the time, for evaluation datasets, metrics are used for the only purpose of ranking different algorithms. This is useful for benchmarking and choosing an existing algorithm for one's usage, but it raises some problems from the end user's perspective:

- It does not reflect the context of the use case that might be different from the original validation set. For example, if someone wants to estimate depth from a single camera but only for close objects because the long range is already covered by another sensor, this user won't be interested in long range depth estimation quality, and the metric used for benchmarking will have to reflect this.
- Some characteristics of an algorithm are inherently antagonistic, and thus a trade-off must be decided between them. The most usual example is accuracy *vs* speed. By ranking algorithms with only one metric, the dataset makes the trade-off decision in place of the end user and thus takes the risk of not being informative. For example, because speed is only given as declared from the authors, KITTI depth benchmark [16] completely disregards it. It would be interesting to have data presented into a 2D chart, the same way as VOT [42] so that the end user can eventually set his own speed/accuracy trade-off.
- A metric can give information about the distribution of expected values given a particular estimation. As such, as shown in [43, p. 38], all metrics are not the same when it comes to trying to characterize possible real values of an estimation. The ideal solution would be to give for each estimated depth value the exact distribution of real depth. That would require an infinite set, but the validation set can give an approximation that gives more insight than just a single number.

To reflect those practices, both classic metrics and histograms are provided in the next section, in order to give as much information as possible for a given algorithm.

Metric Name	Acronym	Formula
Mean Absolute Error	MAE	$\mathbb{E}\lvert\widetilde{\theta} - \theta\rvert$
Mean Relative Error	MRE	$\mathbb{E}\frac{\lvert\widetilde{\theta}-\theta\rvert}{\theta}$
Mean Log Error	MLE	$\mathbb{E}\lvert\log(\widetilde{\theta}) - \log(\theta)\rvert$
Standard Absolute Error	SAE	$\sqrt{\mathbb{E}(\widetilde{\theta} - \theta)^2}$
Standard Log Error	SLE	$\sqrt{\mathbb{E}(\log(\widetilde{\theta}) - \log(\theta))^2}$
Precisions δ	P_δ	$P\left(\left\lvert\log\left(\frac{\widetilde{\theta}}{\theta}\right)\right\rvert \leq \log(\delta)\right)$

Table 2 Summary of the considered metrics

Again, the code for metric measurement is open sourced on Github [5]. It consists in getting all depth pixels and their estimation, within an unordered set V. Note that this evaluation set is not image-wise, all depth values are collected at first with corresponding metadata, and the metric computation is done at the end. This is useful for images with a very sparse depth, where no representative statistics can be computed.

4.2 Scalar metrics

A scalar metric is obtained by computing depth errors for all points and then averaging them. As mentioned above, the mean is global over the validation set V.

$$\mathcal{E}_f = \frac{1}{|V|} \sum_{(\widetilde{\theta},\theta)\in V} f(\widetilde{\theta},\theta) = \mathbb{E}_V(f(\widetilde{\theta},\theta)) \tag{1}$$

It is computed for any error function f as shown by Equation 1, where θ is the ground truth depth for a particular pixel, $\widetilde{\theta}$ its estimate, $|V|$ is the cardinality of set V, and $\mathbb{E}_V$ is the expectation calculated over V.

Table 2 shows the provided error functions and their corresponding names.

4.3 Histogram metrics

Histogram metrics cannot be used for ranking algorithms but they give much more information. Two different kinds of histograms are proposed as follows:

– Depth wise error: Given a scalar metric, the error can be computed as shown in Equation 2 for particular depth ranges $\Theta_i = [\theta_i, \theta_{i+1}[$.

$$H_D(\Theta_i) = \mathbb{E}_{\{\theta_i\leq\theta<\theta_{i+1}\}}(f(\theta,\widetilde{\theta})) \tag{2}$$

[5] https://github.com/ClementPinard/depth-dataset-builder/tree/master/evaluation_toolkit

– Difference distribution: This distribution is normally centered around 0, and its standard deviation is the standard error mentioned on Table 2. Having the whole distribution is especially interesting for distribution that are not symmetrical. A safety interval can then be deduced from this histogram for each side of estimated depth. For this part, it is assumed that the log of depth estimation is more likely to follow a symmetrical distribution than raw depth estimation. It is equivalent to get a distribution of the ratio between estimation and ground truth, usually centered around 1, see Equation 3.

$$H_{\Delta}(\delta) = P(\delta < \log(\widetilde{\theta}) - \log(\theta) < \delta + 1) = P\left(10^{\delta} < \frac{\widetilde{\theta}}{\theta} < 10^{\delta+1}\right) \quad (3)$$

4.4 Displacement wise metrics

In addition to those metrics, a displacement wise metric in the form of a histogram is particularly relevant for navigation. Knowing the displacement of the camera, it can be deduced what point in the image the camera is moving towards. This point, called the flight path vector (FPV) for aircraft, also corresponds to the epipole for multi-view geometry, and to the focus of expansion (FOE) for optical flow in the case of rotation-less movement. As a consequence, these points are deemed more important than the other ones.

$$\mathcal{E}_{FPV}(\alpha) = \mathbb{E}_{V(\alpha)}(f(\widetilde{\theta}, \theta)) \quad (4)$$

The corresponding histogram based metrics is defined in Equation 4, where $V(\alpha)$ is the set of points at a distance α from the FPV of each image (be it in pixels, or in radians).

This particular distribution can help discard an algorithm that fails to estimate depth around those points while having good metrics otherwise. This is the case for optical flow based method for a stabilized camera without rotation: optical flow is too small around the FPV and thus disparity based depth estimation becomes too noisy. [43, p 32]

5 Applications

5.1 Preamble

This section presents two use cases that have been covered using this tool. They both feature drone footage, and are used to test two algorithms: DepthNet [21] and SFM-Learner [10]. They are not state-of-the-art on popular benchmarks like KITTI[16], but they are relevant to demonstrate the interest of the proposed evaluation framework and data sets, for the following reasons:

– They are both compatible with an evaluation scenario focused on navigation, since their scale estimation can be scaled using only odometry

- They are supposed to be real time, contrary to MVS algorithms like the one used in COLMAP
- Their performance metrics are similar on KITTI depth [16]
- Contrary to SFMLearner, DepthNet has been specifically designed to be robust to variability of context and scene layout. As such, for heterogeneous data sets, DepthNet is expected to perform better than SFMLearner [28].

The evaluation protocol is the same for the two scenarios. Images and depth maps are rescaled to 416×234, and for both neural networks, the training schedule is the same as described in their original publication. As such, no hyperparameter search is performed and no validation set is needed. The dataset is thus divided into two parts, one for training and one for testing. These experiments are not intended to form an actual benchmark, but rather to show the possibility of constructing an evaluation where a reference benchmark, here, KITTI [16], is not longer relevant. DepthNet is pre-trained on the Still Box dataset [21], then trained on the learning data set with photometric depth auto-supervision, and supervision for odometry rotation. SFM-Learner is not pre-trained (no improvements were observed when pre-training with KITTI); it is entirely trained on the learning data set without any supervision.

5.2 First application: The drone Manoir dataset

5.2.1 Context

The first use case is a scene with a mansion (in French: manoir) in the countryside, on a $350 \times 100m^2$ terrain. The maximum altitude of obstacle is $20m$. 3D Lidar data was captured by a DJI Matrice 600 with a Velodyne VLP-16 on board, with RTK GPS system (see Figure 6). The flight altitude of this UAV was 30 meters at minimum for safety reasons. The UAV was used because it can cover an area much faster than any fixed scanner, and can easily scan building roofs. The whole scanning process took less than 10 minutes, meaning a very large area could have been covered in less than a day.

For photogrammetry oriented pictures, an Anafi drone was used with the free Pix4D app that allows to make one grid and two orbits above the scanned field. A personal DSLR (Sony alpha-6000) was also used for additional photo. Videos were acquired at two different quality settings for a total of 65k frames to localize:

- 3840×2160 (4K) at 30 fps, best quality setting.
- 1280×720 at 120 fps, bad quality but high frame rate.

See Figure 7.

5.2.2 Result

A subjective result of photogrammetry can be seen on Figure 8, for the optimal subset of 1000 frames and a full video taken by a drone. Finally, Figure 9 shows a sample of

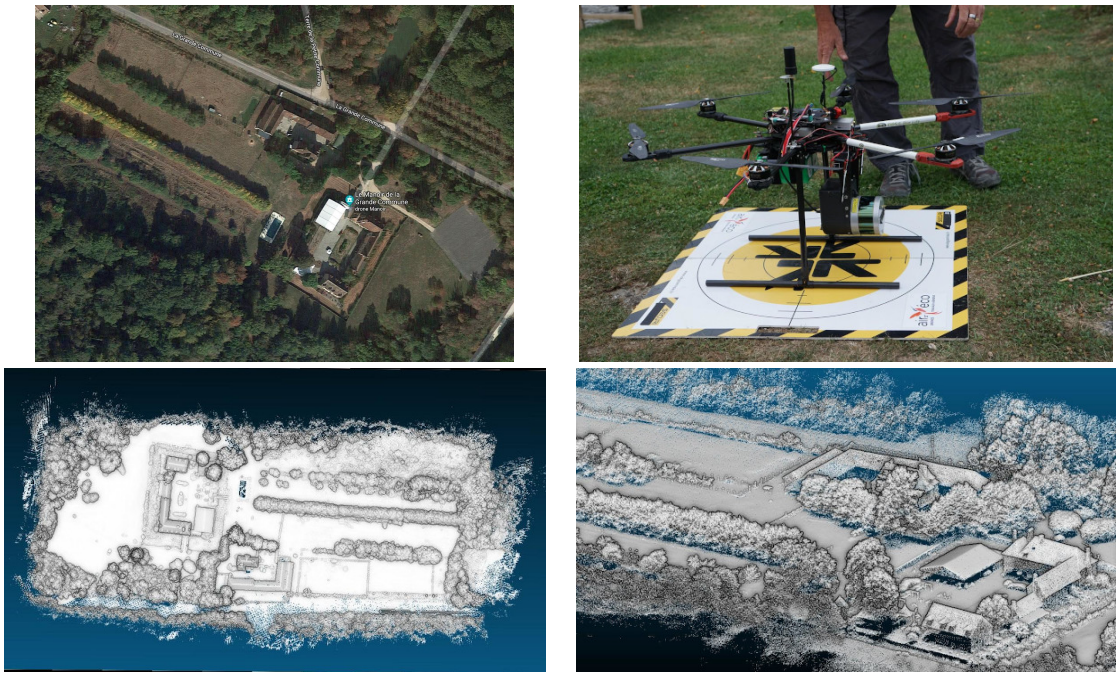


Fig. 6 3D Mapping process of the Manoir dataset. Top: map of the scanned area and scan device used for mapping. Bottom: two views of the resulting point cloud.

computed depth maps. A video playlist is available as a supplementary material for all the sequences [6]

5.2.3 Note on point cloud completeness

As it can be seen in Figure 6, the point cloud seriously lacks completeness for parts that are not easily seen from an altitude of 30 meters. This includes vertical sections like walls or poles, like seen in Figure 10, or cluttered sections, like under trees or inside a tunnel. From this remark, two observations can be made:

– Although it covers a large area very quickly, UAV Lidar scanning does not cover a wide range of viewpoints. The 30 meters altitude makes it difficult to see the same thing as within a few meters of altitude. As such, it would have been useful to add Lidar scans from a lower altitude, using other methods, for example a fixed or a handheld Lidar. Regarding this use case, a handheld Lidar is recommended because it's much faster than fixed Lidar (typically as fast as human gait, while fixed Lidar requires several minutes per viewpoint), and if a UAV is available, all the hardware for using Lidar with IMU technique like *e.g.* LIO-SAM [44] or a proprietary technique like GeoSLAM is already available.

[6] https://youtube.com/playlist?list=PLMeM2q87QjqhYA_LfJY925ZAGyD5cS6Q-

Fig. 7 Video acquisition process of the Manoir dataset. Left: grid (top) and orbit (bottom) flight plan used for photogrammetry. Right: Video samples from the Anafi drone, 4k (top) and 720p (bottom).

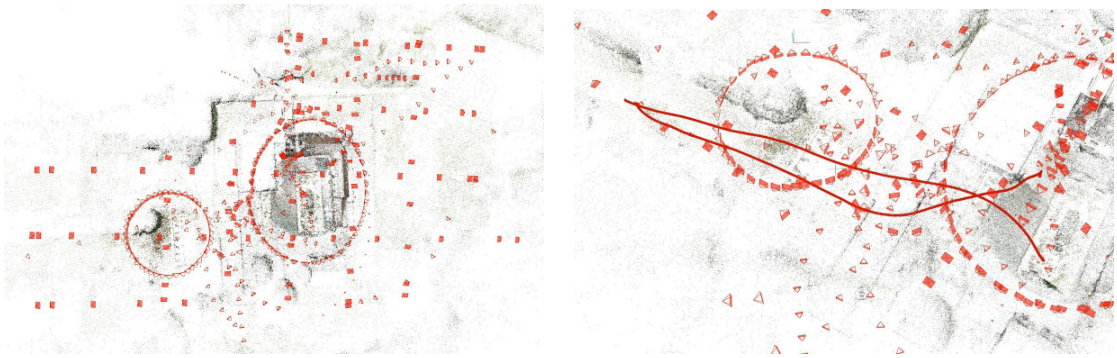


Fig. 8 Left: global photogrammetry result (point cloud in black, camera odometry in red). Right: camera odometry during a given video footage (red curve), w.r.t. the global photogrammetry.

– In case no other Lidar scan is possible (*e.g.* because the area is not reachable other than with a drone), the COLMAP cloud is locally good. An interesting compromise for future work would be to combine the Lidar cloud with COLMAP reconstruction for obstacles very close to the camera.

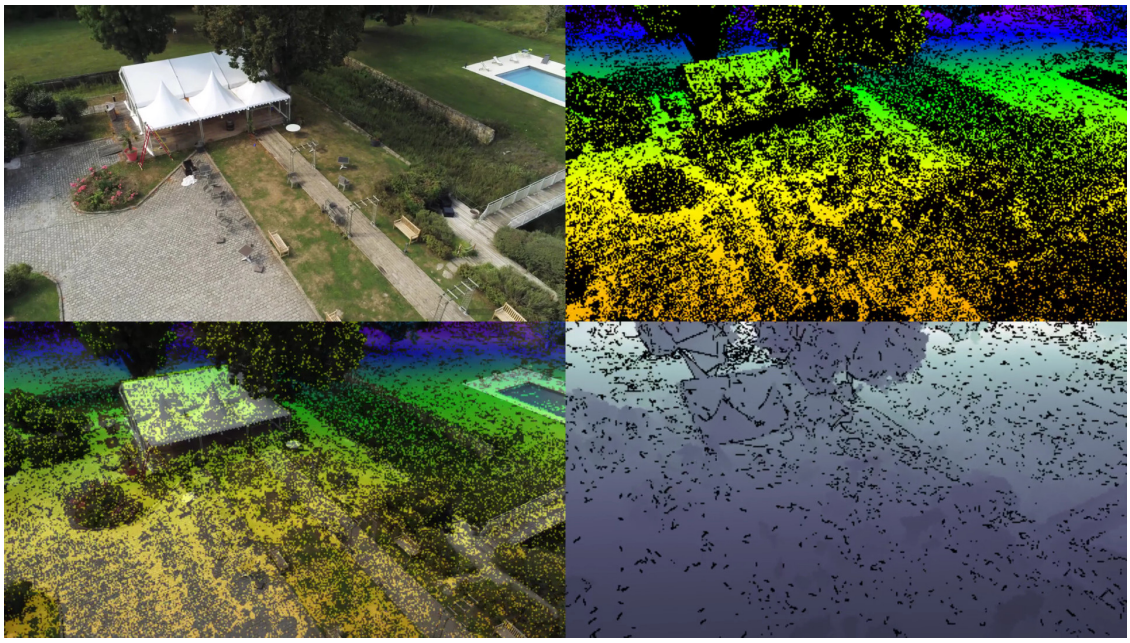


Fig. 9 Depth result from the Manoir dataset. From top to bottom, left to right: RGB image, depth map, depth map superposed to grayscale image, occlusion depth.

5.2.4 Benchmark

Now, regarding the evaluation of the two mentioned algorithms, scalar metrics are shown in Table 3. Figure 12 (left panel) displays, on the top the histogram of (estimation / ground truth) ratios (equivalent to log depth difference, see Equation 3), and on the bottom, the histogram of log-errors according to the estimated depth (See Equation 2), while samples can be seen Figure 13 (first three rows). This first evaluation shows that unsurprisingly, DepthNet largely outperforms SFMLearner in all metrics. This is also visible on the log difference distribution: although both distributions have their maximum at a null log difference (*i.e.* a ratio of 1), indicating an unbiased estimator, DepthNet's distribution is much more concentrated. Both distributions also exhibit a clear skewness toward negative difference. This is corroborated by the second plot where, for both algorithms, the log error is much lower above 20 meters. This is a good hint for a potential drone manufacturer to think of a dedicated system for low depth values. For example, a stereo camera setup would be complementary to these two systems, because it's more accurate for lower depth values.

For conciseness purpose, all the proposed histogram based metrics are not displayed here. However, they can be easily generated using the open source tool. [7]

[7] https://github.com/ClementPinard/depth-dataset-builder#depth-algorithm-evaluation

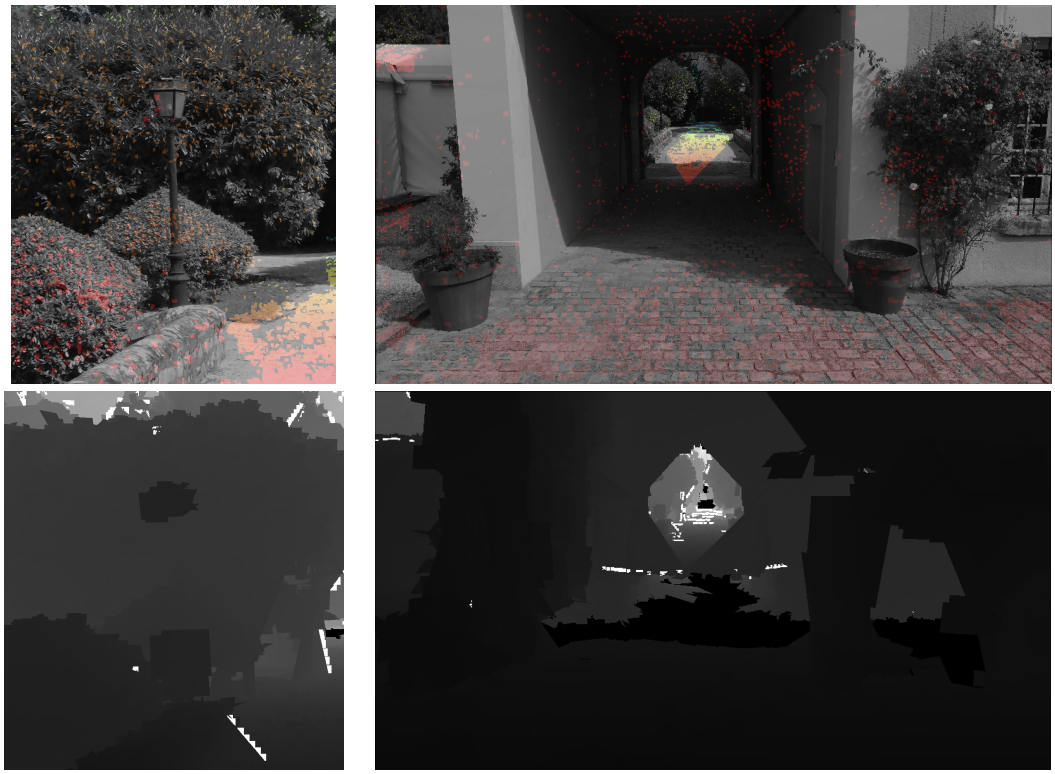


Fig. 10 Typical problems coming from point cloud sparsity. Top: depth map superposed to grayscale image. Bottom: corresponding occlusion depth map. Left: Thin object vanishment. Right: Tunnel shrinkage.

Method / Metric	MAE	MRE	MLE	SAE	SLE	$P_{1.25}$	$P_{1.25^2}$	$P_{1.25^3}$
SFMLearner [10]	18.40	0.5145	1.005	24.46	1.458	0.2395	0.4113	0.5385
DepthNet [38]	**11.99**	**0.3275**	**0.5290**	**18.51**	**0.9099**	**0.5241**	**0.7069**	**0.7717**

Table 3 Metric comparison between SFMLearner [10] and DepthNet [38] on the *Manoir* dataset. See metrics formulae in Table 2

5.3 Second application: University hall dataset

5.3.1 Context

The second use case is an indoor scene, shown on Figure 11, the hall of a University with a very high ceiling. For scan mapping, a handheld Zeb Horizon from GeoSLAM was used, with an announced precision of 4 cm. The videos were acquired using an Anafi drone, without using specific frames for photogrammetry. Like for Manoir dataset, the results of the small benchmark can be seen on Table 4 and Figure 12 (right panel). Some samples can be seen on Figure 13 (last two rows).

Although DepthNet still outperforms SFMLearner for most metrics, the estimation accuracy is much lower. This can be explained by the fact that the university

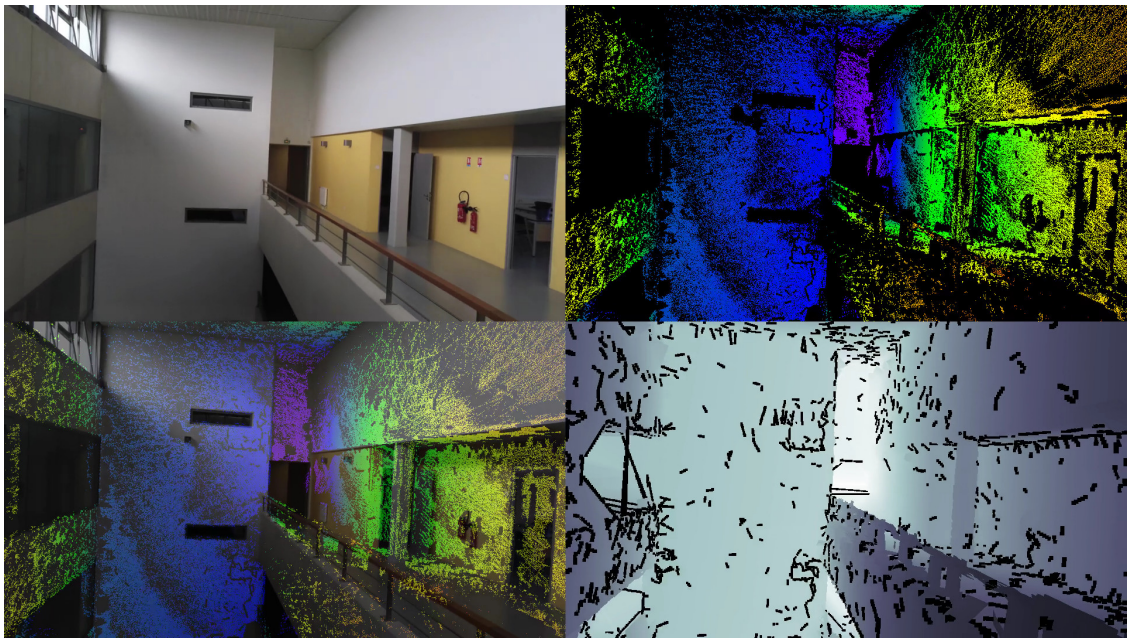


Fig. 11 University hall depth visualization, same structure as Figure 9

Method / Metric	MAE	MRE	MLE	SAE	SLE	$P_{1.25}$	$P_{1.25^2}$	$P_{1.25^3}$
SFMLearner [10]	10.25	1.0225	0.5682	18.29	0.7798	0.3180	0.5461	0.6912
DepthNet [38]	**5.890**	**0.5197**	**0.5056**	**10.612**	**0.726**	**0.3184**	**0.5982**	**0.7694**

Table 4 Metric comparison between SFMLearner [10] and DepthNet [38] on the *University hall* dataset. See metrics formulae in Table 2

hall is mostly composed of completely white walls, with many glass windows and a somewhat reflective ground (by the way, specular surfaces are also responsible for the ”holes” in the ground truth coming from the Lidar). This is a very challenging use case for SFM based training and inference methods. These results indicate that both SFMLearner and DepthNet are much more suited for outdoor scenarios, with irregular textured surfaces, and that active sensing solutions like structured light or time-of-flight are probably more effective for this dataset.

6 Discussion and conclusive remarks

This paper introduced Rigid Depth Constructor (RDC), a powerful tool for depth validation dataset ground truth generation. It is very flexible with respect to available budget and hardware. Its context of application can be as rudimentary as no-budget-at-all, only using a handheld camera (*e.g.* the user’s phone camera), and still it lets a user make the most of limited means, to have depth enabled videos with potentially infinite range. The open access of RDC will allow to improve it by gradually increasing the diversity of use cases.

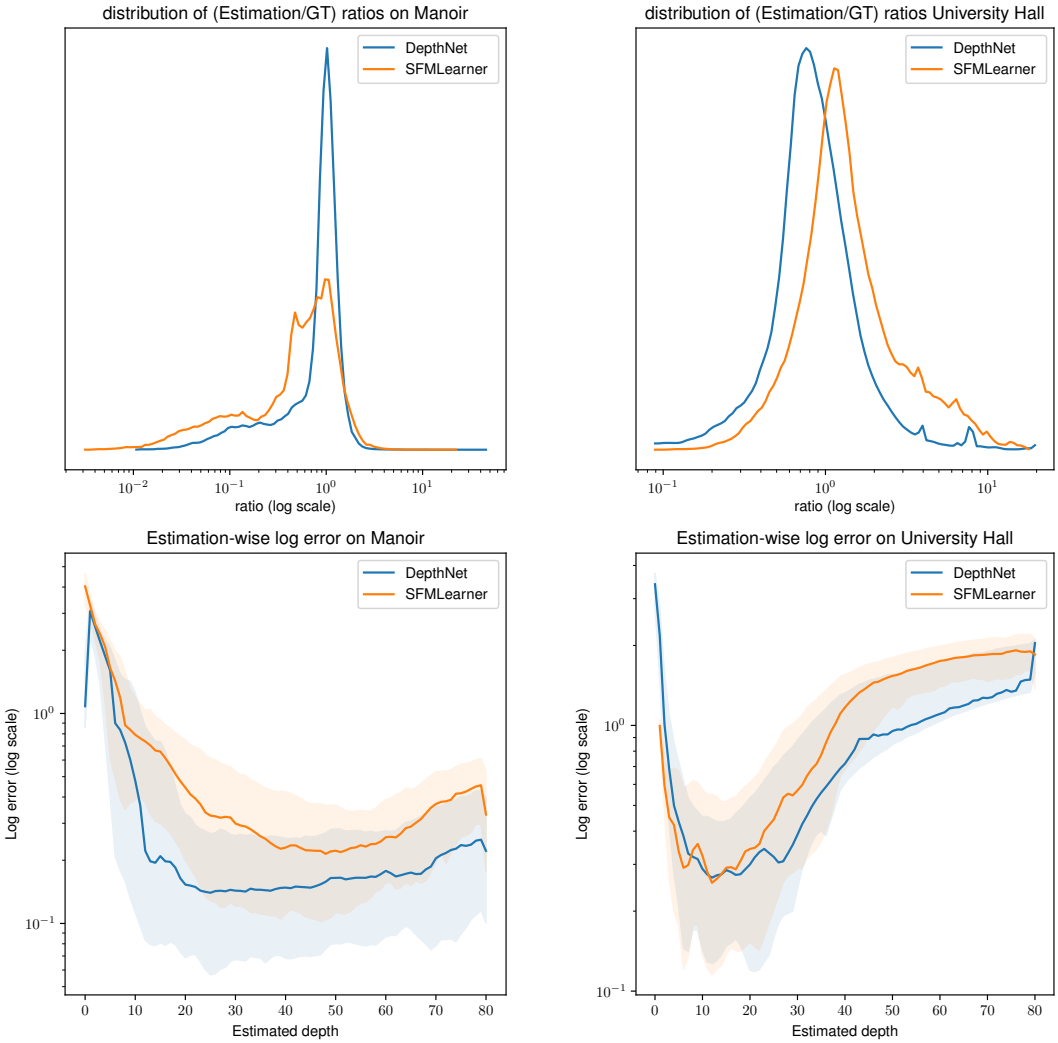


Fig. 12 Comparison of error distributions between SFMLearner [10] and DepthNet [38] on *Manoir* dataset (left) and on *University hall* dataset (right). Top: H_Δ (See Equation 3). Bottom: H_D (See Equation 2). For visualization purpose, the y scale is logarithmic. The curve shows the median value and the shaded area represents the 50% confidence interval.

6.1 Limitations

RDC still suffers from some already known limitations:

– The first limitation is obviously the need for a rigid scene. Although this problem can be mitigated in the case of a UAV flying, it can be particularly problematic for in-car environment in a urban area where many people and other vehicles are dynamic. It can also be a problem when Lidar scanning and video recording are

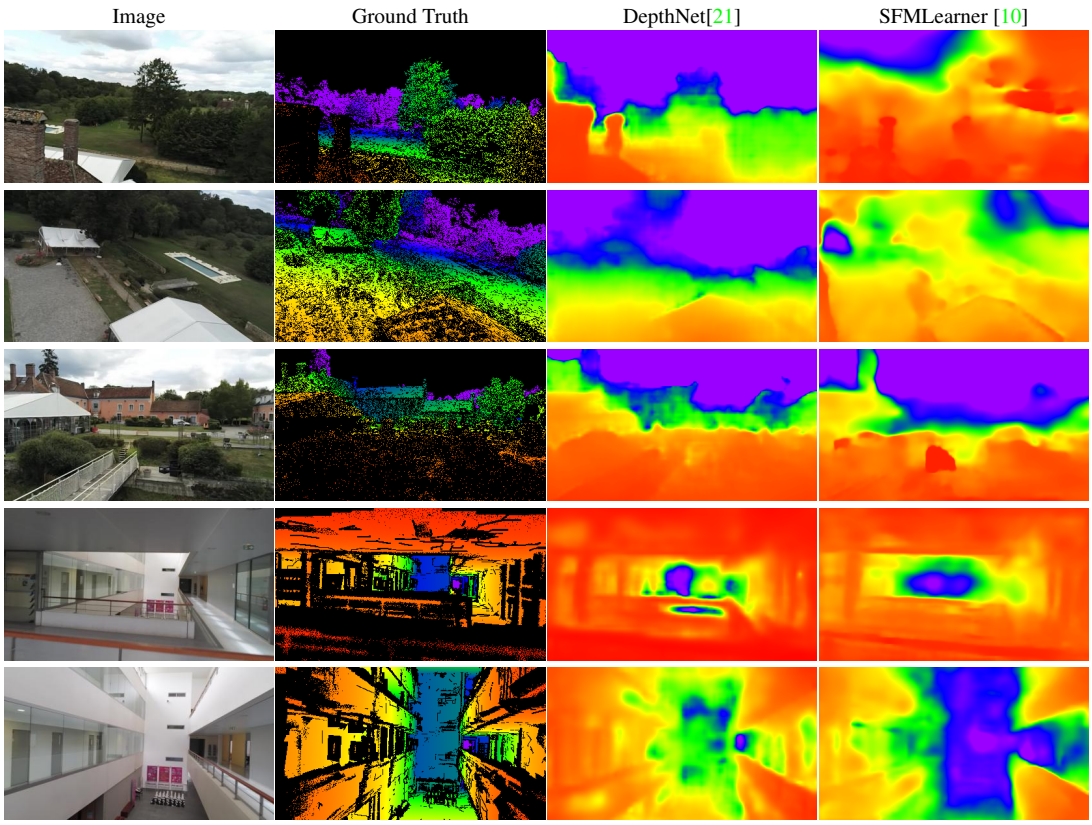


Fig. 13 Vizualisation of some depth map prediction samples on *Manoir* and *University hall* data sets. Colormap is OpenCV Rainbow, normalized to Ground truth.

not done at the same time, in a scene with rigid but movable objects. This was the case in the University hall dataset, where some seats had been moved, thus making some depth maps useless. A possible solution would be to combine a dense high quality point cloud of a scene, scanned beforehand, and sparse point cloud from a mobile Lidar. By characterizing possible dynamic elements, like humans and other vehicles in a urban area, it becomes possible to segment the movable objects from the rigid scene, in order to discard them for the depth map (See for example [45, 46]) and replace it with the depth from the sparse mobile Lidar.

– Although it does not require heavy hardware for data acquisition, the computer used by this tool must be powerful (as with any photogrammetry task), and have plenty of disk space, which mitigates the low budget argument. However, a regular gaming desktop computer is often enough, and is orders of magnitude cheaper than a full Lidar and cam rig solution.
– Although it can be improved by using better registration algorithms than a simple ICP, the registration part will often need a human eye to assess the registration

quality. This part of the process lies in the middle of it, which makes the tool not totally automatic. It is essential that it happens after the photogrammetry and before ground truth generation, so there is no obvious solution yet. However, if it is impossible for a human to evaluate the (absolute) quality of a depth map, it is very easy to check the registration between images and depth map, notably using the frontiers between foreground objects and the background scene.
- When the camera is located in a very sparse portion of the point cloud, the occlusion depth will be of poor quality. It is possible that the end user will discover after the whole dataset creation process that some scenes would need a better scanning process. It would then be interesting to enter a warning before the end of the process, as soon as the Lidar point cloud is registered, that some areas lack data.

6.2 Further work

The next logical step is now to construct a more comprehensive dataset with this tool, using different scenes and contexts in the same way ETH3D dataset was built, but with much more frames. Aside from the multiplicity of contexts, the variety of cameras and vision fields is also a challenge: panoramic cameras [47, 48], that are particularly interesting in mapping and navigation would also benefit much from more validation datasets.

Another natural outcome of this paper will be to perform a comprehensive benchmark of a large collection of state-of-the-art depth prediction models, including GeoNet [11], MonoDepth2 [12], SC-SFM [13], GLNet [14] and Struct2Depth and its descendants [49], to get quantitative assessment of their performance on environments possibly much different from those they were evaluated on. In this respect, it will be particularly interesting to evaluate techniques able to estimate uncertainty [50].

Additionally, a full evaluation suite would be very beneficial, as this initial version only provides basic examples of histogram based algorithm quality assessment.

Finally, using the localization of video frames in a point cloud, the ground truth generation can be extended to other tasks:

- From odometry and ground truth point cloud, ground truth optical flow can be deduced.
- Semantic annotations on the Lidar point cloud can facilitate the construction of ground truth semantic segmentation maps.

Acknowledgements Acquisitions for the *Manoir* dataset were made in collaboration with AIRD'ECO-Drone[8] company, thanks to the financial support of Parrot[9] company. Acquisitions for the *University hall* dataset were made entirely by Clément Pinard, thanks to the equipment and training provided by Geomesure[10] company.

[8] https://www.airdeco-drone.com
[9] https://www.parrot.com
[10] https://www.geomesure.fr/

Competing / Conflict of interest Declaration

The authors declare that they have no conflict of interest or competing interest related to this work.

References

1. Andreas Geiger, Philip Lenz, and Raquel Urtasun. Are we ready for autonomous driving? the kitti vision benchmark suite. In *2012 IEEE Conference on Computer Vision and Pattern Recognition*, pages 3354–3361. IEEE, 2012. 2
2. Cai, Z. and Han, J. and Liu, L. and Shao, L. RGB-D datasets using Microsoft Kinect or similar sensors: a survey. *Multimedia Tools and Applications*, 76:4313–4355, 2017. 2
3. Lazaros Nalpantidis, Ioannis Kostavelis, and Antonios Gasteratos. Stereovision-based algorithm for obstacle avoidance. In *International Conference on Intelligent Robotics and Applications*, pages 195–204. Springer, 2009. 3
4. Richard Hartley and Andrew Zisserman. *Multiple View Geometry in Computer Vision*. Cambridge University Press, 2 edition, 2004. 3
5. Huan Fu, Mingming Gong, Chaohui Wang, Kayhan Batmanghelich, and Dacheng Tao. Deep Ordinal Regression Network for Monocular Depth Estimation. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2018. 4, 15
6. Jin Han Lee, Myung-Kyu Han, Dong Wook Ko, and Il Hong Suh. From big to small: Multi-scale local planar guidance for monocular depth estimation. *arXiv preprint arXiv:1907.10326*, 2019. 4, 15
7. Ashutosh Saxena, Sung H Chung, and Andrew Y Ng. 3-d depth reconstruction from a single still image. *International journal of computer vision*, 76(1):53–69, 2008. 4
8. David Eigen, Christian Puhrsch, and Rob Fergus. Depth map prediction from a single image using a multi-scale deep network. *Advances in neural information processing systems*, 27:2366–2374, 2014. 4, 12, 14
9. Brett T Lopez and Jonathan P How. Aggressive 3-d collision avoidance for high-speed navigation. In *2017 IEEE International Conference on Robotics and Automation (ICRA)*, pages 5759–5765. IEEE, 2017. 4
10. Tinghui Zhou, Matthew Brown, Noah Snavely, and David G. Lowe. Unsupervised learning of depth and ego-motion from video. In *CVPR*, 2017. 4, 12, 15, 17, 22, 23, 24, 25
11. Zhichao Yin and Jianping Shi. GeoNet: Unsupervised learning of dense depth, optical flow and camera pose. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2018. 4, 15, 26
12. Clément Godard, Oisin Mac Aodha, Michael Firman, and Gabriel J. Brostow. Digging into self-supervised monocular depth prediction. In *The International Conference on Computer Vision (ICCV)*, October 2019. 4, 15, 26
13. Jiawang Bian, Zhichao Li, Naiyan Wang, Huangying Zhan, Chunhua Shen, Ming-Ming Cheng, and Ian Reid. Unsupervised scale-consistent depth and ego-motion learning from monocular video. In *Advances in Neural Information Processing Systems*, volume 32. Curran Associates, Inc., 2019. 4, 15, 26
14. Yuhua Chen, Cordelia Schmid, and Cristian Sminchisescu. Self-supervised learning with geometric constraints in monocular video: Connecting flow, depth, and camera. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, October 2019. 4, 15, 26
15. Guzel, Mehmet Serdar and Nattharith, Panus. New Technique for distance estimation using SIFT for mobile robots. In *2014 International Electrical Engineering Congress (iEECON)*, pages 1–4, 2014. 4
16. Jonas Uhrig, Nick Schneider, Lukas Schneider, Uwe Franke, Thomas Brox, and Andreas Geiger. Sparsity invariant cnns. In *International Conference on 3D Vision (3DV)*, 2017. 4, 5, 15, 17, 18
17. Pushmeet Kohli Nathan Silberman, Derek Hoiem and Rob Fergus. Indoor segmentation and support inference from rgbd images. In *ECCV*, 2012. 4, 5
18. Tom van Dijk and Guido de Croon. How do neural networks see depth in single images? In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, October 2019. 4

19. D. J. Butler, J. Wulff, G. B. Stanley, and M. J. Black. A naturalistic open source movie for optical flow evaluation. In A. Fitzgibbon et al. (Eds.), editor, *European Conf. on Computer Vision (ECCV)*, Part IV, LNCS 7577, pages 611–625. Springer-Verlag, October 2012. 5
20. A. Dosovitskiy, P. Fischer, E. Ilg, P. Häusser, C. Hazırbaş, V. Golkov, P. v.d. Smagt, D. Cremers, and T. Brox. Flownet: Learning optical flow with convolutional networks. In *IEEE International Conference on Computer Vision (ICCV)*, 2015. 5
21. C. Pinard, L. Chevalley, A. Manzanera, and D. Filliat. End-to-end depth from motion with stabilized monocular videos. *ISPRS Annals of Photogrammetry, Remote Sensing and Spatial Information Sciences*, IV-2/W3:67–74, 2017. 5, 17, 18, 25
22. Igor Vasiljevic, Nick Kolkin, Shanyi Zhang, Ruotian Luo, Haochen Wang, Falcon Z. Dai, Andrea F. Daniele, Mohammadreza Mostajabi, Steven Basart, Matthew R. Walter, and Gregory Shakhnarovich. DIODE: A Dense Indoor and Outdoor DEpth Dataset. *CoRR*, abs/1908.00463, 2019. 5
23. Hendrik Schilling, Marcel Gutsche, Alexander Brock, Dane Späth, Carsten Rother, and Karsten Krispin. Mind the gap - a benchmark for dense depth prediction beyond lidar. In *2020 IEEE Conference on Computer Vision and Pattern Recognition Workshops (CVPRW)*, volume in press, 2020. 5
24. Thomas Schöps, Johannes L. Schönberger, Silvano Galliani, Torsten Sattler, Konrad Schindler, Marc Pollefeys, and Andreas Geiger. A multi-view stereo benchmark with high-resolution images and multi-camera videos. In *Conference on Computer Vision and Pattern Recognition (CVPR)*, 2017. 5, 7, 8, 10, 12
25. Arno Knapitsch, Jaesik Park, Qian-Yi Zhou, and Vladlen Koltun. Tanks and temples: Benchmarking large-scale scene reconstruction. *ACM Transactions on Graphics*, 36(4), 2017. 5
26. Gollob, Christoph and Ritter, Tim and Nothdurft, Arne. Comparison of 3D Point Clouds Obtained by Terrestrial Laser Scanning and Personal Laser Scanning on Forest Inventory Sample Plots. *MDPI - Data*, 5(4), 2020. 6
27. Johannes Lutz Schönberger and Jan-Michael Frahm. Structure-from-motion revisited. In *Conference on Computer Vision and Pattern Recognition (CVPR)*, 2016. 6
28. Johannes Lutz Schönberger, Enliang Zheng, Marc Pollefeys, and Jan-Michael Frahm. Pixelwise view selection for unstructured multi-view stereo. In *European Conference on Computer Vision (ECCV)*, 2016. 6, 18
29. Johannes Lutz Schönberger, True Price, Torsten Sattler, Jan-Michael Frahm, and Marc Pollefeys. A vote-and-verify strategy for fast spatial verification in image retrieval. In *Asian Conference on Computer Vision (ACCV)*, 2016. 7
30. P. J. Besl and N. D. McKay. A method for registration of 3-d shapes. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 14(2):239–256, 1992. 7, 11
31. Michael Kazhdan, Matthew Bolitho, and Hugues Hoppe. Poisson surface reconstruction. In *Proceedings of the fourth Eurographics symposium on Geometry processing*, volume 7, 2006. 8, 9
32. Karl Kraus, Ian A. Harley, and Stephen Kyle. *Photogrammetry: Geometry from Images and Laser Scans*. De Gruyter, Berlin, Boston, 18 Oct. 2011. 9
33. P. Labatut, J.-P. Pons, and R. Keriven. Robust and Efficient Surface Reconstruction From Range Data. *Computer Graphics Forum*, 2009. 9
34. Mur-Artal, Raúl and Montiel, J. M. M. and Tardós, Juan D. ORB-SLAM: a Versatile and Accurate Monocular SLAM System. *IEEE Transactions on Robotics*, 31(5):1147–1163, 2015. 11
35. J. Macqueen. Some methods for classification and analysis of multivariate observations. In *In 5-th Berkeley Symposium on Mathematical Statistics and Probability*, pages 281–297, 1967. 11
36. Yang Chen and Gerard Medioni. Object modelling by registration of multiple range images. *Image and Vision Computing*, 10(3):145 – 155, 1992. Range Image Understanding. 11
37. Abraham. Savitzky and M. J. E. Golay. Smoothing and differentiation of data by simplified least squares procedures. *Analytical Chemistry*, 36(8):1627–1639, 1964. 12
38. Clément Pinard, Laure Chevalley, Antoine Manzanera, and David Filliat. Learning structure-from-motion from motion. In *Proceedings of the European Conference on Computer Vision (ECCV)*, 2018. 12, 14, 22, 23, 24
39. Ariel Gordon, Hanhan Li, Rico Jonschkowski, and Anelia Angelova. Depth from videos in the wild: Unsupervised monocular depth learning from unknown cameras. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, October 2019. 13, 15
40. G. K. L. Tam, Z. Cheng, Y. Lai, F. C. Langbein, Y. Liu, D. Marshall, R. R. Martin, X. Sun, and P. L. Rosin. Registration of 3d point clouds and meshes: A survey from rigid to nonrigid. *IEEE Transactions on Visualization and Computer Graphics*, 19(7):1199–1217, 2013. 14

41. Ravi Garg, BG Vijay Kumar, Gustavo Carneiro, and Ian Reid. Unsupervised cnn for single view depth estimation: Geometry to the rescue. In *European Conference on Computer Vision*, pages 740–756. Springer, 2016. 14
42. Matej Kristan, Jiri Matas, Aleš Leonardis, Tomas Vojir, Roman Pflugfelder, Gustavo Fernandez, Georg Nebehay, Fatih Porikli, and Luka Čehovin. A novel performance evaluation methodology for single-target trackers. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 38(11):2137–2155, Nov 2016. 15
43. Clément Pinard. *Robust Learning of a depth map for obstacle avoidance with a monocular stabilized flying camera*. Theses, Université Paris Saclay (COmUE), June 2019. 15, 17
44. Tixiao Shan, Brendan Englot, Drew Meyers, Wei Wang, Carlo Ratti, and Rus Daniela. Lio-sam: Tightly-coupled lidar inertial odometry via smoothing and mapping. In *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 5135–5142. IEEE, 2020. 19
45. Aikaterini Fragkiadaki, Bryan Seybold, Cordelia Schmid, Rahul Sukthankar, Sudheendra Vijayanarasimhan, and Susanna Ricco. Self-supervised learning of structure and motion from video. In *arxiv (2017)*, 2017. 25
46. Hanhan Li, Ariel Gordon, Hang Zhao, Vincent Casser, and Anelia Angelova. Unsupervised monocular depth learning in dynamic scenes. In *Conference on Robot Learning (CoRL)*, 2020. 25
47. Payen de La Garanderie, Grégoire and Atapour Abarghouei, Amir and Breckon, Toby P. Eliminating the Blind Spot: Adapting 3D Object Detection and Monocular Depth Estimation to 360° Panoramic Imagery. In Vittorio Ferrari, Martial Hebert, Cristian Sminchisescu, and Yair Weiss, editors, *Computer Vision – ECCV 2018*, pages 812–830, Cham, 2018. Springer International Publishing. 26
48. Zhou, Keyang and Yang, Kailun and Wang, Kaiwei. Panoramic Depth Estimation via Supervised and Unsupervised Learning in Indoor Scenes. *Applied optics*, 60 26:8188–8197, 2021. 26
49. Li, Hanhan and Gordon, Ariel and Zhao, Hang and Casser, Vincent and Angelova, Anelia. Unsupervised Monocular Depth Learning in Dynamic Scenes. In Jens Kober, Fabio Ramos, and Claire Tomlin, editors, *Proceedings of the 2020 Conference on Robot Learning*, volume 155 of *Proceedings of Machine Learning Research*, pages 1908–1917, 2021. 26
50. Poggi, Matteo and Aleotti, Filippo and Tosi, Fabio and Mattoccia, Stefano. On the uncertainty of self-supervised monocular depth estimation. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2020. 26