Dense 3D-Reconstruction from Monocular Image Sequences for Computationally Constrained UAS *

Matthias Domnik[1,2], Pedro Proenca[2], Jeff Delaune[2], Jörg Thiem[1], and Roland Brockers[2]

[1]University of Applied Sciences and Arts Dortmund, Dortmund, Germany
{matthias.domnik, joerg.thiem}@fh-dortmund.de
[2]Jet Propulsion Laboratory, California Institute of Technology, Pasadena, CA, USA
{pproenca, jeff.h.delaune, roland.brockers}@jpl.nasa.gov

Abstract

The ability to find safe landing sites over complex 3D terrain is an essential safety feature for fully autonomous small unmanned aerial systems (UAS), which requires on-board perception for 3D reconstruction and terrain analysis if the overflown terrain is unknown. This is a challenge for UAS that are limited in size, weight and computational power, such as small rotorcrafts executing autonomous missions on Earth, or in planetary applications such as the Mars Helicopter. For such a computationally constraint system, we propose a structure from motion approach that uses inputs from a single downward facing camera to produce dense point clouds of the overflown terrain in real time. In contrast to existing approaches, our method uses metric pose information from a visual-inertial odometry algorithm as camera pose priors, which allows deploying a fast pose refinement step to align camera frames such that a conventional stereo algorithm can be used for dense 3D reconstruction. We validate the performance of our approach with extensive evaluations in simulation, and demonstrate the feasibility with data from UAS flights.

1. Introduction

On July 30, 2020 NASA launched the rover *Perseverance* that is scheduled to land on Mars in February 2021. Traveling on-board the rover is the Mars Helicopter *Ingenuity* which was developed to prove that autonomous controlled flight is possible on Mars [3]. If this technology demonstration is successful, it could open the door for future Mars Science Helicopters enabling a whole new era of Mars exploration. Due to the signal delay between Mars and Earth, such a rotorcraft has to operate fully autonomously, which requires advanced navigation capabilities to fly over complex 3D terrain, including finding new safe landing places during flight. A feature that requires on-board 3D perception since maps derived from orbit cannot resolve small landing hazards (maximum HiRISE resolution is 25 cm/pixel [21]) and planned landing sites might not be reachable in cases of emergencies.

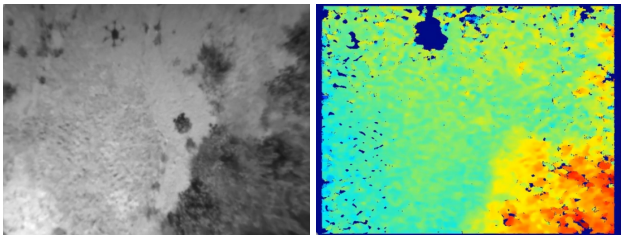

Figure 1. 3D reconstruction example: Left: rectified reference image from UAS flight; Right: reconstructed range image (warm colors are closer to the camera).

Detecting landing sites in unknown terrain requires a robust method for on-board 3D reconstruction before observations can be aggregated into a local map representation used for detecting suitable landing sites during flight. On-board 3D reconstruction on a computationally limited embedded computer is however challenging, since methods deployed need to be accurate and at the same time efficient enough to be executed in reasonable time.

In this paper, we introduce a 3D reconstruction method that takes advantage of existing autonomous navigation components on-board a small autonomous rotorcraft. Our method deploys a structure from motion approach with a single downward facing camera that receives metric pose priors from a visual-inertial odometry algorithm. This al-

*This research was carried out at the Jet Propulsion Laboratory, California Institute of Technology, under a contract with the National Aeronautics and Space Administration (80NM0018D0004).

lows us to efficiently perform a pose refinement step with subsequent deployment of a dense stereo algorithm on selected frames. The result is a dense 3D reconstruction of the observed surface (see Figure 1) as a basis for subsequent processes such as landing site detection that can be executed on small embedded compute modules as our target platform, a Qualcomm Snapdragon 820 system on a chip.

In the following chapters, we give a brief introduction on related work, and then explain our approach in detail, followed by experimental evaluation in simulation and with data from UAS flights.

2. Related Work

Obtaining accurate and reliable information about the environment is crucial for autonomous navigation. Decades of research evaluated a vast variety of options of sensor systems and algorithms.

Several sensors can be taken into account to approach this problem. Commonly used sensor types for this task are for example Lidar [24], ultrasonic sensors [17], or depth cameras for indoor applications [4]. While Lidars are too heavy and power hungry for our application, depth cameras and ultrasound sensors only work in close proximity. A stereo camera would be an ideal sensor, but small baselines increase depth errors quickly for flights at altitude. Monocular Structure-from-Motion (SfM) approaches for 3D reconstruction overcome this limitation by adapting the baseline between camera observations depending on the distance to the observed terrain. Of course, this comes at the price of a non rigid setup that cannot be pre-calibrated and thus requires accurate camera pose estimation. Conventional SfM couples the pose estimation problem with the 3D reconstruction problem, often requiring significant computational resources. Examples are commercial applications such as Pix4D [1] that deploy a full Bundle Adjustment (BA) over all camera poses and observed scene points to estimate highly-accurate terrain models. These methods are executed off-line on state-of-the art computation hardware, and thus not feasible for on-board 3D reconstruction in real time.

Speed up can by achieved by sparsifying the reconstructed environment [16], moving this method closer to Simultaneous Localization and Mapping (SLAM) methods that can be executed in real-time on computationally constraints systems. However, here the main focus is localization and not dense 3D reconstruction. SLAM approaches can be separated in terms of their processing of the image data in *indirect* and *direct* methods. Indirect methods use the geometric reprojection error in the optimization process [22, 26], while direct methods are based on the photometric error [11, 10, 23]. A hybrid-approach is given by [12, 13], where feature tracking is based on minimizing the photometric error for incremental camera poses. Indirect methods have advantages regarding robustness to outlier, large motion and lighting changes over time. In contrast, direct methods are more suitable for scenes with low texture. Since the major goal of approaches like ORB-SLAM [22] or VINS-Mono [26] are pose and state estimation in real-time, the density of reconstructed point clouds is low and too sparse for reconstructing the environment by a sequential mapping.

Engel *et al*. introduced Large-Scale Direct Monocular (LSD)-SLAM [11] and later Direct Sparse Odometry (DSO) [10], which provide semi-dense depth maps or point clouds with an adjustable density. These approaches come close to meeting our requirements, if the density is increased to full image reconstruction. However, since all 3D points are used for the combined optimization of camera poses and 3D observations, the process grows in computational complexity with increasing density, which is not appropriate for our target hardware.

Newcombe *et al*. [23] focuses on generating dense depth maps from monocular image sequences by either refining an already available 3D model or building initially a new model from sparse correspondences. The optimization is based on minimizing the photometric error by a global energy minimization framework extensively parallelizing calculations on a Graphics Processing Unit (GPU).

There are some approaches that also emphasize on dense reconstruction. An example is REMODE (REgularized MOnocular Depth Estimation) [25] which is a real-time temporal fusion approach based on Bayesian estimation that heavily uses GPU parallelization in order to reach acceptable execution times. This approach combines 3D reconstruction with a local mapping approach, which could be interesting for perception tasks if ample computational power exists, as shown previously by Daftry *et al*. [6] for autonomous landing.

Finally, if the overflown terrain is mostly flat, approaches that deploy a homography approach to estimate camera poses efficiently and then perform 3D reconstruction can have an advantage since pose estimation by homography decomposition does not require solving a computationally intensive optimization problem as shown by [5] and [9].

Methods that use on optimization back-end often deploy a standard optimization frame work. Prominent examples are g^2o [20], GTSAM [8] and Ceres [2].

g^2o, used in the first SVO release [12] and in ORB-SLAM [22], formulates the non-linear least square problem as a directed graph. With regard to efficiency, special structures, as they occur within BA or SLAM, as well as the sparseness of the graph can be explicitly taken into account.

Georgia Tech Smoothing and Mapping (GTSAM) is also a graph-based approach for solving non-linear least square problems, used *e.g*. by SVO in the second release [13]. This framework includes different approaches, such as iSAM

[19] and iSAM2 [18]. The approaches in GTSAM place a special focus on the incremental expansion of the problem, as is particularly the case with SLAM problems.

Ceres is used by VINS-MONO [26] during initialization. In contrast to the two previous approaches, Ceres is not graph-based. The focus in this framework is on efficient modeling and solving of large and complicated non-linear optimization problems.

3. Proposed Approach

In the following we present our approach to calculate dense depth maps from monocular image sequences. Figure 2 gives an overview of the processing pipeline for our Structure from Motion approach.

Camera pose priors are provided by an existing state estimator while we execute a separate feature tracker (see Section 3.1) to process the images from a downwards-facing camera and generate frame to frame feature tracks. We deploy a keyframe based approach which selects individual images based on a desired image overlap and a minimum parallax constraint (see Section 3.3), and collect keyframes in a sliding window buffer. Camera poses of keyframes are refined by an optimization step (Section 3.4). Inspired by related approaches [10, 13, 22], we focus on Bundle Adjustment (BA) as a less time-restricted optimization back-end in this work. For 3D reconstruction, a pair of keyframes is selected based on baseline constraints and a standard stereo algorithm is used for dense 3D reconstruction (see Section 3.5).

3.1. Feature Tracking

We extract FAST feature points in raw input images which are tracked with a KLT based algorithm that uses image binning to enforce an even distribution of features across the image. Lost feature tracks are replaced by newly detected features on a frame to frame basis up to a fixed maximum feature number.

3.2. Pose Priors from Visual Inertial Odometry

Our pose priors are obtained from the *xVIO* state estimator [7]. xVIO is based on Extended Kalman Filter (EKF) that tightly couples visual, and optionally range and solar measurements with inertial state propagation [7]. The concept is designed for use in space missions and therefore does not require GPS conditions. The advantage of this approach compared to other state-of-the-art visual-inertial frameworks is the use of a Laser-Range-Finder (LRF) to observe metric scale without requiring inertial excitation, which is usually not present on most real-world flight trajectories. Furthermore, the yaw drift is reduced through the use of the sun sensor.

3.3. Keyframe Management

Individual keyframes are selected out of the stream of individual images and the attached camera pose priors and collected in a sliding window keyframe buffer. For the reconstruction after the camera pose refinement, we further have to select two camera poses for stereo reconstruction. We split these tasks in two separate processes to select new camera frames as keyframes in one process and an image pair from the keyframe buffer for dense stereo reconstruction in a second process (see Section 3.5).

The camera frames in the keyframe buffer are selected by evaluating the rotation compensated, frame-to-frame parallax, which ensures a required amount of movement between the frames. Further, a new keyframe should preserve information that is already available in the buffer, while simultaneously adding new information to it. This means, a significant amount of feature tracks should be continued by the new keyframe and also new feature observations for starting new feature tracks are introduced.

3.4. Camera Pose Refinement

We deploy a windowed bundle adjustment (BA) algorithm to improve camera poses. In contrast to full bundle adjustment approaches which refine the complete history of camera poses and feature locations, the goal of our algorithm is to improve only camera poses in the recent history that may be used for 3D reconstruction - which requires images that observe corresponding terrain (image overlap).

We refer to the recent history of camera poses as a *window*, that consists of n keyframes. The keyframes are selected by the procedure as described in Section 3.3. Since the number of keyframes and also the number of features in a window are limited, the run time of the optimization is bounded.

Thus, the optimization run time depends on the window length and the number of feature observations. Sibley *et al*. [27] showed that the number of frames in the window has a minor influence on the accuracy of the optimization result, in contrast to the length of the feature tracks. Engel *et al*. [10] and Qin *et al*. [26] suggest a window length between 7 and 10 camera frames. Since this agrees with our own findings, which we present in Section 4.1, we select a keyframe window length of 7 frames and require features to be visible in all keyframes. The selection of an optimization framework is presented in Section 4.3.

Our objective function is composed of feature observations and camera pose priors, obtained from the state estimator. Thus, its based on the reprojection error as a geometric error measure:

$$\mathbf{r}_{pix}\left(\boldsymbol{\xi}, \mathbf{P}\right) = \hat{\mathbf{z}}_{pix} - \boldsymbol{\Pi}\left(\boldsymbol{\xi}, \mathbf{P}\right), \tag{1}$$

where the camera poses are given by $\boldsymbol{\xi}$, a 3D point by $\mathbf{P}$ and the reprojection function by $\boldsymbol{\Pi}(\dots)$. The feature observa-

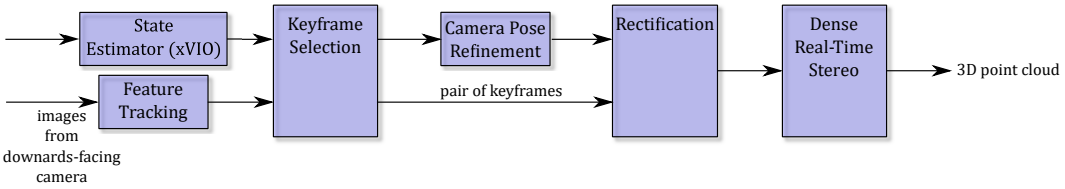

Figure 2. Proposed SfM pipeline for calculating dense dense depth maps/3D point clouds from monocular image sequences and pose priors for subsequent landing site detection.

tions are denoted by $\hat{\mathbf{z}}_{pix}$. Furthermore, we use the Huber norm as a robust loss function:

$$\rho(r) = r^2 \text{ for } |r| < \alpha \quad (2)$$
$$= 2\alpha|r| - \alpha^2 \text{ otherwise.} \quad (3)$$

One of the main advantages of the Huber norm is that, in contrast to the Cauchy norm, for example, it is convex and therefore does not introduce further local minima [14].

Since the residual distribution of the feature observations is widely considered to be a superposition of normal [27] and uniformly distributed [25] inlier and outlier, respectively. The residual distribution is sufficiently modeled by the Huber norm. For more details we refer to [14].

3.5. Dense 3D-Reconstruction

For 3D reconstruction, we deploy conventional stereo algorithm. This involves selecting an image pair from the keyframe buffer, that maximizes the baseline and also yields an image overlap in a range between 70% and 90%. The reference frame of the stereo pair is the most recent, *i.e.* current, frame.

After rectification, we deploy a fast standard block-matching stereo algorithm for correspondence search and 3D reconstruction.

4. Experimental Evaluation

The following section provides an overview of the impact of various parameters affecting the windowed BA (Section 4.1). Further, the three optimization back-ends from Section 2 are compared in terms of accuracy and run time, regarding our approach. Several experiments are conducted with the selected framework using simulated feature tracks (see Section 4.4), a full-image simulation (see Section 4.5) and a qualitative evaluation with UAS flight data (see Section 4.6).

4.1. Parameter Studies

This section is dedicated to the evaluation of different parameters and making design choices for our approach.

First, we analyze the influence of feature observation noise and camera pose perturbations on the windowed BA. Noise on feature observations has a major influence on the optimization problem, as experiments show. However, since the noise on the feature observation, *i.e.* our measurements, is not observable, it yields the lower bound of the achievable accuracy.

Rotational perturbations on the camera poses are found to be negligible, while translational errors in the direction of the motion of the rotorcraft affect the optimization, since scale in vision-only BA is not observable. However, with the assumption that the initialization of our BA optimization with pose priors from VIO is close to the global minimum of our BA problem, scale changes by the BA step can be neglected. We evaluated window lengths of 5, 7, 9, 11, 15 and 20 camera poses per window regarding improvements in accuracy and increasing run time. For this experiment, the feature tracks are continuous through the entire window and the number of features is constant.

Table 1. Comparison of the median Vertical Epipolar Error (VEE) (see Section 4.2) in a window for different sizes of the keyframe buffer, *i.e.* length of the window. For each keyframe buffer length 1500 camera pose refinements are processed.

	quantiles in [px]			
	25%	**50%**	**75%**	**99%**
5 frames	0.0285	0.0472	0.0744	0.1653
7 frames	0.0270	0.0431	0.0688	0.1526
9 frames	0.0244	0.0407	0.0650	0.1447
11 frames	0.0260	0.0402	0.0627	0.1331
15 frames	0.0241	0.0386	0.0620	0.1467
20 frames	0.0233	0.0364	0.0580	0.1274

Regarding the accuracy presented in Table 1, no significant improvements for an increasing window size are observable. In contrast, however, the run time per iteration is largely influenced by the window length, as presented in Figure 3.

The run time grows exponentially with an increasing window length. Thus, using a window length of 7 frames, as also suggested by [10], is appropriate.

4.2. Evaluation Metrics

To evaluate pose estimation methods, a common metric is to calculate the deviation of estimated trajectory to a ground truth trajectory, if available. Refinement approaches are usually compared and evaluated by the root mean square (RMS) of the reprojection error. However, the possibility of successful stereo reconstruction is not directly apparent

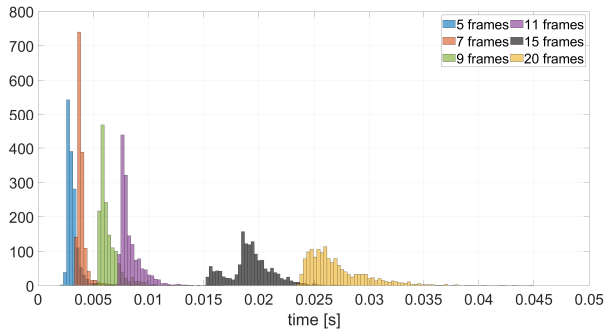

Figure 3. Run time evaluation for different window length (5, 7, 9, 11, 15 and 20) with a constant number of feature observations in the window.

from this. Therefore, we use the estimated *vertical epipolar error* (VEE) after image rectification as an error metric for the accuracy of our optimized camera poses to simulate the performance of a subsequent dense stereo algorithm that requires alignment of epipolar lines with image rows.

While a large VEE is indicating that stereo matching will fail due to the violated epipolar constraint, a horizontal epipolar error will affect the accuracy of depth reconstruction. Since both errors are coupled, we assume that the horizontal epipolar error is negligible, if the VEE is small.

Our calculations are based on projecting pixel coordinates onto a virtual ground plane, which altitude is derived from the actual altitude of the rotorcraft. Figure 4 visualizes the idea of the following described method. First, pixel coordinates are generated w.r.t. to the first camera view and are projected to the virtual ground plane using the camera matrix. These pixel coordinates are further rectified using the camera poses which are under examination, *e.g.* camera poses after the refinement step.

Next, the projected 3D points on the ground plane are reprojected into the view of the second camera and the pixel coordinates are also rectified. Subtracting the corresponding, rectified pixel coordinates from the first and second view, we obtain the vertical shift in pixels.

The maximum permissible VEE essentially depends on the matching algorithm used. In the following it is assumed, that with a VEE below $0.25\,\mathrm{px}$ the stereo reconstruction will not fail due to an incorrect rectification, and that depth errors caused by horizontal epipolar errors are sufficiently small to be absorbed by a subsequent mapping process.

4.3. Selection of Optimization Frameworks

We compare the three optimization frameworks, presented in Section 2, Ceres, g^{2}o and GTSAM.

First, the comparison focuses on the achievable accuracy of the frameworks regarding our approach. Therefore, we process the same data set for all frameworks. The data is retrieved from a non-visual simulation, which provides camera poses on a trajectory and ground truth feature observations via reprojection from a given 3D terrain mesh. The added noise on feature observations is Gaussian distributed with $0.1\,\mathrm{px}$. The camera poses are also perturbed by independent Gaussian noise with $0.01\,\mathrm{m}$ and $0.5°$, respectively.

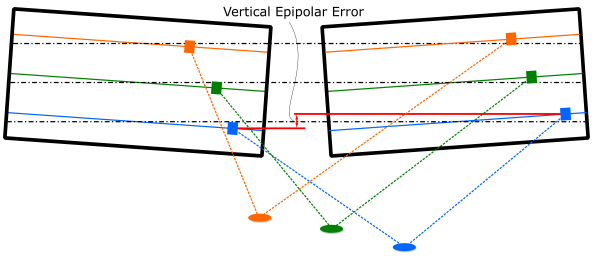

Figure 4. The vertical epipolar error (VEE) is calculated by projecting the features in the first image onto a virtual ground plane, and back-projection into the second camera view.

Table 2 presents the results of the comparison regarding the accuracy of the optimized solution of the three frameworks. For this experiment 574 subsequent camera poses refinements are performed by the optimization back-end on a circular trajectory. All frameworks are processed with exactly the same data and stopping criteria.

Table 2. Evaluation of the median VEE in a window calculated with Ceres, GTSAM and g^{2}o over a circular trajectory with 574 windows.

	quantiles in [px]			
Frameworks	**25%**	**50%**	**75%**	**99%**
Ceres	0.0105	0.0135	0.0172	0.0256
GTSAM	0.0104	0.0133	0.0173	0.0258
g2o	0.0107	0.0139	0.0176	0.0265

No significant difference in terms of accuracy can be observed between Ceres, GTSAM and g^{2}o.

Next, we evaluate the run time of each frame work. The data used for evaluation is derived from the previous test runs, that includes 574 measurements per framework. The test is carried out on an Intel i7 desktop computer.

Figure 5 shows the results as a histogram. Ceres has the shortest run times in this comparison; followed by g^{2}o, and GTSAM. Furthermore, Ceres has the smallest run-time standard deviation in our test.

Pose graph approaches, *i.e.* here g^{2}o and GTSAM, aim for problems on a larger scale, then we have to solve in our approach. Since we are only interested in refining camera poses, we select Ceres as framework for further work.

In the next step we prove the computational feasibility on the target platform, a Snapdragon 820. Since the noise on the feature observations majorly affects the optimization results, as discussed in Section 4.1, we examine the run time under different feature noise conditions. The simulated camera trajectory is a straight line with ground truth camera poses and 7 camera frames per window. Each noise

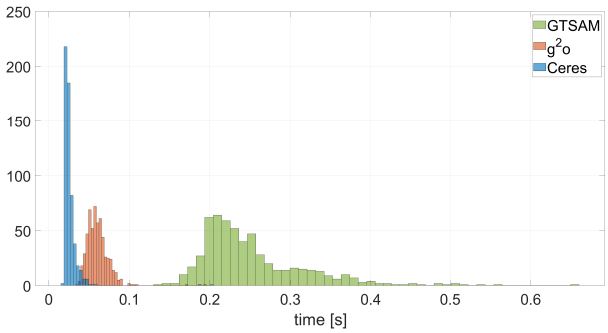

Figure 5. Run times of the examined frameworks measured on a desktop computer with an Intel i7.

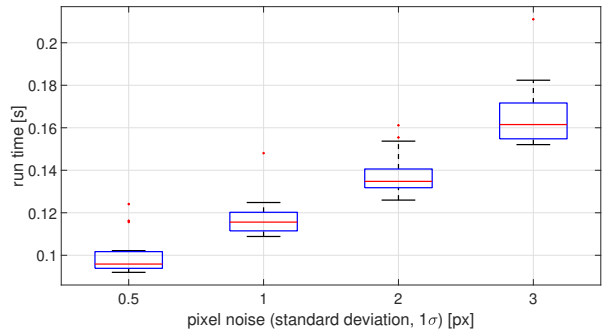

Figure 6. Run time of Ceres on the Snapdragon 820 under examination of different noise magnitudes on the feature observations (0.5 px, 1.0 px, 2.0 px and 3.0 px). The box plot presents the median as red line, the 25% and the 75% quantile by the blue box and the whiskers correspond to 1.5-times the inter quantile range. Outlier are marked separately.

level consists of 17 camera poses refinements, *i.e.* windows.

As expected, the run time increases with a higher noise magnitude on the feature observations. We aim for a frequency of 1 Hz for the entire pipeline. Thus, the optimization process should not take longer than 200 ms. Except for a single measurement, the run time stays well below this limit.

4.4. Optimization Back-End Evaluation

We evaluated the performance of our approach in a series of Monte Carlo tests in a simulation environment to analyze the accuracy of the approach when using pose priors from the xVIO state estimator [7]. We expect a worst-case uncertainty of 1% error of distance traveled for the translation. Further, a rotational 3σ-uncertainty of 2° is assumed and an average feature observation 1σ-noise of 0.2 px. All perturbations are distributed independently for all axes in a range around the ground truth values. The simulated terrain used for the Monte Carlo tests is derived from a 3D surface with elevation changes from -1 m to 3 m.

The virtual rotorcraft moves along straight lines in various directions over the terrain with a constant altitude of

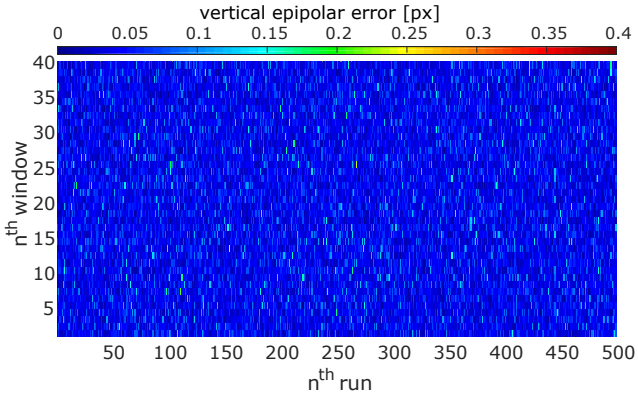

Figure 7. Qualitative overview of a Monte Carlo simulation with 500 runs (x-axis) and 40 windows (y-axis) each, with the y-axis showing the mean VEE for each refined window. The colorbar indicates the magnitude of the VEE from 0.0 px (blue) up to $\geq$0.4 px (red).

approx. 10 m. Figure 7 shows the results for the Monte Carlo simulation with 500 independent runs with 40 windows each. The color map provides a qualitative overview of the experiment, where each window is reduced to its mean VEE along the y-axis. The VEE on the colorbar increases from 0.0 px (blue) up to $\geq$0.4 px (red). As can be seen, the vast majority of the VEEs is distributed at lower end of the provided error scale.

Table 3 presents the quantitative analysis in terms of quantiles of the median VEE of the Monte Carlo experiment, while Table 4 summarizes the results as a histogram analysis.

Table 3. Quantitative analysis of the Monte Carlo simulation with 500 runs of various straight trajectories with 40 windows each.

	quantiles in [px]				
min [px]	25%	50%	75%	99%	**max [px]**
0.0046	0.0308	0.0416	0.0561	0.1211	0.2344

Table 4. Histogram analysis of the Monte Carlo simulation with 500 runs of various straight trajectories with 40 windows each.

bins in [%] (cumulative percentage)					
<0.0625 px	**<0.125 px**	**<0.25 px**	**<0.5 px**	**<0.75 px**	**≥ 0.75 px**
82.14 (82.14)	17.07 (99.21)	0.79 (100.0)	0.00 (100.0)	0.00 (100.0)	0.00 (100.0)

While the vast majority of the median VEE in a window stays below 0.1211 px (>99%), also the maximum error from all 20000 examined windows is below the 0.25 px limit. By analyzing Table 4 we can also observe the majority of the VEEs being mainly distributed in the smallest bin. This indicates, that a continuous dense depth reconstruction is feasible for all windows in the simulation.

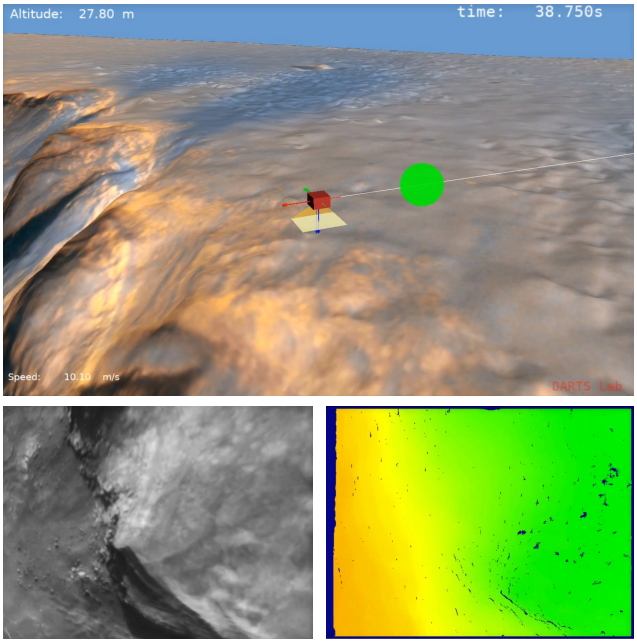

Figure 8. Flight in simulation environment. Top: Simulated UAS over Mars Victoria crater rim; Bottom left: Rectified image of down-facing camera; Bottom right: Associated disparity image (GT poses, no image noise; warmer colors are farther away).

4.5. Pipeline Evaluation in 3D Simulation

In this section we evaluate the proposed pipeline, including the feature tracker and the state estimator xVIO. Thus, a photo-realistic 3D simulator [15] is required for providing the test data (see Figure 8). Since the feature tracker may introduce large outliers due to mismatches, we also introduce a pre-optimization outlier rejection scheme, that removes gross outliers in the set of triangulated feature points. The removal is based on the reprojection error of the 3D feature point in every keyframe it is observed. Thus, if a reprojection error is larger than a threshold of $1\,\mathrm{px}$, this feature is not further considered in the optimization process.

The experiment was carried out with $114\,\mathrm{s}$-long simulated flight, that includes an overflight of a cliff at approximately $t = 18\,\mathrm{s}$ (see Figure 8, Bottom left), where the rotorcraft pitches to follow the terrain. The rotorcraft's altitude is around $20\,\mathrm{m}$ with a velocity of approx. $10\,\mathrm{m\,s^{-1}}$ and camera frame rate of $20\,\mathrm{Hz}$. For this experiment we obtain the initial guesses by ground truth data for simplicity and deploy the feature tracker on simulated images. The number of feature tracks is limited to 400 tracks per window.

The simulation is processed with the same settings with (black dots) and without (red dots) the proposed outlier rejection method in two different runs (see Figure 9).

We evaluate this experiment using the median VEE over each window. For the runs without any outlier rejection (red dots), we can observe VEEs beyond $0.25\,\mathrm{px}$ several times

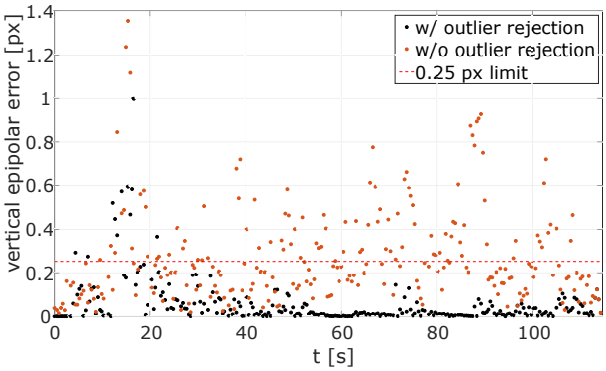

Figure 9. Improvement of median VEE per window by an outlier rejection step prior to the optimization process. Black dots: with outlier rejection; Red dots: without outlier rejection.

over the entire trajectory. In contrast, repeating the simulation run with outlier rejection (black dots) improved the rate of the median VEE over a window below $0.25\,\mathrm{px}$ significantly to over 95%, which is also presented in the last row of Table 6.

We further provide experiments with camera pose priors obtained from xVIO and also with image noise, corrupting the image by shot noise with a standard deviation of $0.179\,\mathrm{px}\,\sqrt{I}$, with I denoting the image intensity, and additional Gaussian blur with $0.25\,\mathrm{px}$ (1σ). All measurements of xVIO are modeled by the InvenSense MPU-9250 with a gyroscope noise spectral density of $1.3 \times 10^{-3}\,\mathrm{s^{-1}Hz^{-1/2}}$, a gyroscope bias random walk of $1.3 \times 10^{-4}\,\mathrm{s^{-2}Hz^{-1/2}}$, a accelerometer noise spectral density of $8.3 \times 10^{-3}\,\mathrm{s^{-2}Hz^{-1/2}}$ and a bias random walk of $8.3 \times 10^{-4}\,\mathrm{s^{-3}Hz^{-1/2}}$.

Table 5. Distribution of VEE for different camera pose refinement methods. Note, that IMU measurements for xVIO are simulated using noise parameters from the InvenSense MPU-9250 IMU, as described in the text.

	quantiles in [px]			
	25%	**50%**	**75%**	**99%**
xVIO	0.2451	0.5001	0.8430	1.5869
BA; w/ noise; w/ xVIO init	0.0805	0.1616	0.2548	0.7549
BA; w/ noise; w/ gt init	0.0045	0.0175	0.0405	0.3969
BA; w/o noise; w/ xVIO init	0.0323	0.0746	0.1711	1.0609
BA; w/o noise; w/ gt init	0.0059	0.0190	0.0438	0.2848

Table 5 presents the quantitative results of the pipeline evaluation using the 3D simulation with different initialization priors and various image noise settings in terms of quantiles, while the quantiles are calculated over the median VEE for each window. Table 6 shows the percentage separated in six groups of the median VEEs in the first line of each row and their cumulative percentage in the second. The first row yields results without the pose refinement step as a reference, using xVIO camera poses directly for stereo. Less than 30% of the camera poses lead to a median VEE per window below $0.25\,\mathrm{px}$ (see Table 6). This can also be

Table 6. Histogram analysis of VEE for different camera pose refinement methods. Note, that IMU measurements for xVIO are simulated using noise parameters from the InvenSense MPU-9250 IMU, as described in the text.

	bins in [%] (cumulative percentage)					
	<0.0625 px	<0.125 px	<0.25 px	<0.5 px	<0.75 px	≥ 0.75 px
xVIO	7.47 (7.47)	7.50 (14.98)	13.99 (28.97)	24.17 (53.14)	18.29 (71.44)	28.57 (100.0)
BA w/ noise w/ xVIO init	23.46 (23.46)	21.75 (45.22)	29.19 (74.41)	19.50 (93.90)	4.68 (98.56)	1.41 (100.0)
BA w/ noise w/ gt init	80.37 (80.37)	9.91 (90.28)	5.59 (95.87)	2.34 (98.21)	0.69 (98.91)	1.09 (100.0)
BA w/o noise w/ xVIO init	45.76 (45.76)	20.06 (65.82)	18.97 (84.79)	11.96 (96.75)	1.71 (98.48)	1.55 (100.0)
BA w/o noise w/ gt init	89.42 (89.42)	5.27 (94.69)	3.02 (97.70)	1.41 (99.12)	0.39 (99.52)	0.48 (100.0)

seen in Table 5: more than 71% of all pixels have a VEE larger than $0.25\,\mathrm{px}$. Thus, the required accuracy is achieved only in a minority of the cases tested.

The most realistic experiment is the initialization by xVIO and additional noise on the images, where approx. 94% of the median VEE are below $0.5\,\mathrm{px}$, while the vast majority of 74.41% stays below $0.25\,\mathrm{px}$. Using xVIO to estimate pose priors without additional noise (fourth row), the percentage of VEEs below $0.25\,\mathrm{px}$ increases significantly. As a reference, the last row shows the initialization with ground truth camera poses and no additional image noise, where over 97% of the median VEEs stay below $0.25\,\mathrm{px}$.

The experiments show, that the refinement step keeps over 90% of the median VEEs below $0.5\,\mathrm{px}$ and over 74% below $0.25\,\mathrm{px}$ in all scenarios, which is sufficient for the deployment of a dense stereo algorithm. The remaining horizontal epipolar error (see Section 4.2) is assumed to be small enough to be absorbed by a subsequent probabilistic mapping approach.

4.6. Evaluation with UAS Flight Data

This section is dedicated to experiments we carried out with UAS flight data over a desert area (see also Figure 1). Flights were executed at a constant height at low altitude ($6\,\mathrm{m}$ to $8\,\mathrm{m}$ above ground) with a constant velocity of $5\,\mathrm{m/s}$ over terrain that varied in height.

Figure 10 presents example frames for various time steps of a flight sequence with the current camera image, overlaid with features that are tracked from frame to frame (left), the rectified reference image (middle), and the corresponding range image (right). Stereo disparity maps were calculated with a SAD-based block matching algorithm using a 11x7 correlation window. The first row shows an increasing slope to the left of the image. In the middle row, some large obstacles are located at the bottom right, whereas the bottom row depicts some small bushes. Note, that colors in the range image are normalized per image.

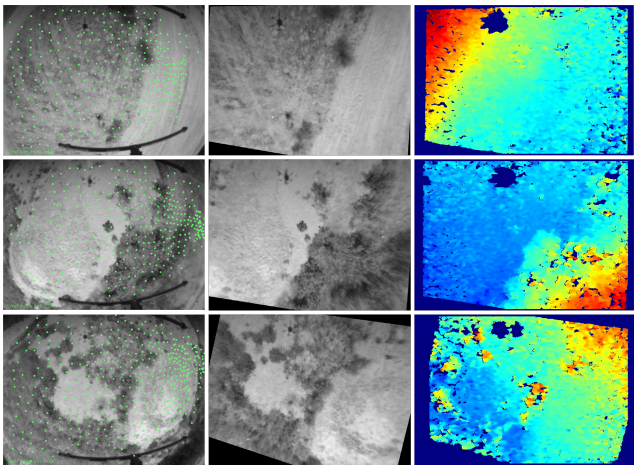

Figure 10. Sample results from UAS flight data over a dessert area. Left: Original camera image with overlaid tracked features; Middle: Rectified reference image; Right: Range image (color codes elevation; warmer colors are closer to the camera).

As one can see, the reconstruction is dense, with the exception of occluded regions and non-assignable areas caused by the moving shadow of the UAS.

5. Conclusion

We presented a real-time approach for generating dense depth maps from a sequence of monocular images taken by a downwards-facing camera. Further, we focused on a computationally efficient design, since our target hardware is constrained in size, weight and power and thus in computational power. First, we distinguished our approach from competitive state-of-the-art approaches. We compared the performance of well established BA optimization frameworks: Ceres, g^{2}o and GTSAM and concluded Ceres is significantly faster for the same accuracy. After proving the computational feasibility of using Ceres on our target hardware, we presented a Monte Carlo experiment with various trajectories over a 3D terrain. We evaluated our SfM approach in a simulation environment that generates realistic flight scenarios. Various combinations of priors and image noise were analyzed together with the influence of image noise. We showed a sufficient accuracy for aligning the stereo image pair for the vast majority of refined camera poses. Furthermore, we showed successful examples of dense 3D depth reconstruction with real-world UAS flight data, which ultimately verified the feasibility of our approach.

Future work includes improved outlier rejection to further increase the accuracy of the refined pose. Additionally, estimating the lower bound of achievable accuracy by the covariance matrix of the approximated Hessian is assumed to give an additional criterion for rejecting ill-posed problems.

References

[1] Pix4dmapper. https://www.pix4d.com/product/pix4dmapper-photogrammetry-software, visited Aug 2020.

[2] Sameer Agarwal, Keir Mierle, and Others. Ceres solver. http://ceres-solver.org.

[3] Bob Balaram, Timothy Canham, Courtney Duncan, Håvard F. Grip, Wayne Johnson, Justin Maki, Amelia Quon, Ryan Stern, and David Zhu. Mars helicopter technology demonstrator. In *2018 AIAA Atmospheric Flight Mechanics Conference*. American Institute of Aeronautics and Astronautics, Jan. 2018.

[4] Joydeep Biswas and Manuela Veloso. Depth camera based indoor mobile robot localization and navigation. *Proceedings - IEEE International Conference on Robotics and Automation*, pages 1697–1702, 05 2012.

[5] Roland Brockers, Sara Susca, David Zhu, and Larry Matthies. Fully self-contained vision-aided navigation and landing of a micro air vehicle independent from external sensor inputs. In *Unmanned Systems Technology XIV, Proc. SPIE 8387, 83870Q*, 2012.

[6] Shreyansh Daftry, Manash Das, Jeff Delaune, Cristina Sorice, Robert Hewitt, Shreetej Reddy, Daniel Lytle, Elvin Gu, and Larry Matthies. Robust Vision-based Autonomous Navigation, Mapping and Landing for MAVs at Night. In *Proc. of the International Symposium on Experimental Robotics, Buenos Aires, Argentina*, pages 232–242, 11 2018.

[7] Jeff Delaune, Roland Brockers, David S. Bayard, Harel Dor, Robert Hewitt, Jacek Sawoniewicz, Gerik Kubiak, Theodore Tzanetos, Larry Matthies, and J. Balaram. Extended navigation capabilities for a future mars science helicopter concept. In *2020 IEEE Aerospace Conference*, pages 1–10, 2020.

[8] Frank Dellaert. Factor graphs and GTSAM: A hands-on introduction. Technical Report GT-RIM-CP&R-2012-002, Institute for Robotics & Intelligent Machines, Georgia Institute of Technology, 2012.

[9] Vishnu Desaraju, Nathan Michael, Martin Humenberger, Roland Brockers, Stephan Weiss, Jeremy Nash, and Larry Matthies. Vision-based landing site evaluation and informed optimal trajectory generation toward autonomous rooftop landing. *Autonomous Robots*, 39(3), 2015.

[10] Jakob Engel, Vladlen Koltun, and Daniel Cremers. Direct Sparse Odometry. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 40(3):611–625, 2018.

[11] Jakob Engel, Thomas Schöps, and Daniel Cremers. LSD-SLAM: Large-scale direct monocular SLAM. In *Computer Vision – ECCV 2014*, pages 834–849. Springer International Publishing, 2014.

[12] Christian Forster, Matia Pizzoli, and Davide Scaramuzza. SVO: Fast semi-direct monocular visual odometry. In *2014 IEEE International Conference on Robotics and Automation (ICRA)*, pages 15–22, 2014.

[13] Christian Forster, Zichao Zhang, Michael Gassner, Manuel Werlberger, and Davide Scaramuzza. SVO: Semidirect Visual Odometry for Monocular and Multicamera Systems. *IEEE Transactions on Robotics*, 33(2):249–265, 2017.

[14] Richard Hartley and Andrew Zisserman. *Multiple View Geometry in Computer Vision*. Cambridge University Press, USA, 2000.

[15] Abhinandan Jain. DARTS - multibody modeling, simulation and analysis software. In *Multibody Dynamics 2019*, pages 433–441. Springer International Publishing, June 2019.

[16] Andrew Johnson, James F. Montgomery, and Larry Matthies. Vision guided landing of an autonomous helicopter in hazardous terrain. In *ICRA 2005*, 2005.

[17] Sungyoung Jung, Jungmin Kim, and Sungshin Kim. Simultaneous localization and mapping of a wheel-based autonomous vehicle with ultrasonic sensors. *Artificial Life and Robotics*, 14:186–190, 11 2009.

[18] Michael Kaess, Hordur Johannsson, Richard Roberts, Viorela Ila, John Leonard, and Frank Dellaert. iSAM2: Incremental smoothing and mapping with fluid relinearization and incremental variable reordering. In *2011 IEEE International Conference on Robotics and Automation*, pages 3281–3288, 2011.

[19] Michael Kaess, Ananth Ranganathan, and Frank Dellaert. iSAM: Incremental Smoothing and Mapping. *IEEE Transactions on Robotics*, 24(6):1365–1378, 2008.

[20] Rainer Kümmerle, Giorgio Grisetti, Hauke Strasdat, Kurt Konolige, and Wolfram Burgard. g2o: A general framework for graph optimization. In *Proc. of the IEEE Int. Conf. on Robotics and Automation (ICRA)*, Shanghai, China, 2011.

[21] Alfred S. McEwen, Eric M. Eliason, James W. Bergstrom, Nathan T. Bridges, Candice J. Hansen, W. Alan Delamere, John A. Grant, Virginia C. Gulick, Kenneth E. Herkenhoff, Laszlo Keszthelyi, Randolph L. Kirk, Michael T. Mellon, Steven W. Squyres, Nicolas Thomas, and Catherine M. Weitz. Mars Reconnaissance Orbiter's High Resolution Imaging Science Experiment (HiRISE). *Journal of Geophysical Research: Planets*, 112(E5), 2007.

[22] Raul Mur-Artal, J. M. M. Montiel, and Juan D. Tardos. ORB-SLAM: A Versatile and Accurate Monocular SLAM System. *IEEE Transactions on Robotics*, 31(5):1147–1163, 2015.

[23] Newcombe, Richard A. and Lovegrove, Steven J. and Davison, Andrew J. DTAM: Dense tracking and mapping in real-time. In *2011 International Conference on Computer Vision*, pages 2320–2327, 2011.

[24] Tomás Olvera, Ulises Orozco-Rosas, and Kenia Picos. Mapping and navigation in an unknown environment using LiDAR for mobile service robots. In Abdul A. S. Awwal, Khan M. Iftekharuddin, Victor H. Diaz-Ramirez, and Andrés Márquez, editors, *Optics and Photonics for Information Processing XIV*, volume 11509, pages 31–45. SPIE, 2020.

[25] Matia Pizzoli, Christian Forster, and Davide Scaramuzza. REMODE: Probabilistic, monocular dense reconstruction in real time. In *2014 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, May 2014.

[26] Tong Qin, Peiliang Li, and Shaojie Shen. VINS-Mono: A Robust and Versatile Monocular Visual-Inertial State Estimator. *IEEE Transactions on Robotics*, 34(4):1004–1020, 2018.

[27] Gabe Sibley, Larry Matthies, and Gaurav Sukhatme. Sliding window filter with application to planetary landing. *Journal of Field Robotics*, 27(5):587–608, 2010.