Real-time local 3D reconstruction for aerial inspection using superpixel expansion

Conference Paper**Author(s):**

Teixeira, Lucas ; Chli, Margarita

Publication date:

2017

Permanent link:

<https://doi.org/10.3929/ethz-a-010855647>

Rights / license:

[In Copyright - Non-Commercial Use Permitted](#)

Originally published in:

<https://doi.org/10.1109/ICRA.2017.7989530>

Funding acknowledgement:

644128 - Collaborative Aerial Robotic Workers (SBFI)

Real-time Local 3D Reconstruction for Aerial Inspection using Superpixel Expansion

Lucas Teixeira and Margarita Chli
Vision for Robotics Lab, ETH Zurich, Switzerland

Abstract—On the quest of automating the navigation of challenging and promising Robotics platforms such as small Unmanned Aerial Vehicles (UAVs), the community has been increasingly active in developing perception capabilities able to run onboard such platforms in real-time. Despite that vision-based techniques have been at the heart of recent advancements, the realistic employment onboard UAVs is still in its infancy.

Inspired by some of the most recent breakthroughs in online dense scene estimation and borrowing fundamental concepts from Computer Vision, in this work we propose a new pipeline for real-time, local scene reconstruction using a single camera for aerial navigation. Aiming for denser scene estimation than traditional feature-based maps with the ability to run onboard a small UAV in real-time, the proposed approach is demonstrated to achieve unprecedented performance producing rich maps of the camera’s workspace, timely enough to serve in obstacle avoidance and real-time interaction of a robot with its direct surroundings. Evaluation on benchmarking datasets and on challenging aerial footage captured with a UAV featuring a conventional camera, reveals dramatic speed-ups, as well as denser and more accurate local reconstructions with respect to the state of the art.

Video— <https://youtu.be/T5GbHs6H1tY>

I. INTRODUCTION

Over the past decade, we have witnessed some impressive advancements in Robotics technology. Right at the forefront are small Unmanned Aerial Vehicles (UAVs) equipped with onboard cameras, recently demonstrating that vision-based position control for UAVs [2] are possible without reliance on GPS, sparking great interest in a plethora of areas from entertainment to industrial inspection. But do we, today, have the technology to enable the autonomous navigation of a UAV that would help map a disaster area, for instance? The state of the art still lacks solutions ready to leave the controlled laboratory environment and perform in real missions, with onboard robotic perception constituting the biggest impediment.

With Simultaneous Localization And Mapping (SLAM) forming the most basic layer of robotic spatial awareness for autonomous navigation, the literature has reached certain maturity in this field by now. In particular, in monocular SLAM we have certainly come a long way since the first demonstration of the online, incremental egomotion estimation and mapping of MonoSLAM [3]. Following the realization that keyframe-based approaches offer better characteristics of performance versus computational cost [4], the community

This research was supported by the Swiss National Science Foundation (SNSF, Agreement no. PP00P2_157585) and EC’s Horizon 2020 Programme under grant agreement n. 644128 (AEROWORKS).

Fig. 1: Using visual and inertial data captured from flights with the UAVs visible in (b), (a) and (c) illustrate examples of the scene-depth estimates produced by the proposed algorithm, accumulated over a small time-window. The green overlay corresponds to scene ground truth captured using a Leica Total Station laser tracker – the drift observed reflects the SLAM algorithm’s pose drift. The aerial public dataset of [1] was used for this figure.

has been shifting towards keyframe-based SLAM, such as PTAM [5]. We have witnessed the employment of such systems onboard UAVs and fused with inertial measurements for aerial SLAM [2], while most recently, ORB-SLAM [6] aiming to exploit the power of newly emerging binary feature descriptors.

As SLAM techniques have started being successfully demonstrated onboard robots, a new horizon of applications and challenges emerges on taking robotic spatial perception a step further. With the prospect of employing robots in real tasks becoming more tangible, we have come to the realisation that the traditionally sparse feature-based SLAM maps are never going to be enough for navigation autonomy (e.g. generic obstacle avoidance and path planning) or any type of interaction of a robot with its environment (e.g. manipulation). In this work, we propose a novel pipeline for reconstructing the immediate surroundings of a single camera in real-time on a CPU onboard aerial UAV. Con-

ducting comparisons with the state of the art on existing benchmarking datasets, such as the handheld RGBD dataset of [7], the outdoor datasets of [1] captured by a professional UAV, as well as our own dataset captured by a consumer-grade UAV with particularly unstable dynamics. All involved UAVs are visible in Figure 1b, while Figures 1a and 1c illustrate an example of the scene’s map captured by the proposed pipeline.

The proposed approach follows a workflow similar to Structure from Motion (SfM). We employ a SLAM algorithm to estimate the poses of the camera frames. Following image rectification and dense matching across keyframes, we obtain a very fast, but crude depth estimation of the scene. The main contribution here is the clean-up and the densification of this initial scene estimation. Using very strict filtering, we first remove many noisy depth estimates, while also sacrificing the loss of some valid estimates. We mitigate this by propagating the surviving depth estimates temporally, while employing super-pixel expansion. Avoiding the computationally expensive steps in a typical SfM pipeline, this approach provides a local semi-dense 3D scene reconstruction in real time.

II. RELATED WORK

Following the emergence of high-quality real-time monocular SLAM systems, significant attention was triggered in denser monocular scene reconstruction. The unprecedented quality of the KinectFusion maps [8] built in real-time from not only visual, but also depth cues (also known as RGBD sensing) served as an aspiration for vision-only approaches. Shortly after, the emergence of KinectFusion, came DTAM [9] to pioneer online dense reconstruction using a monocular camera, achieving comparable reconstruction quality to KinectFusion. With the work by Stuehmer et al. [10] coming out at about the same time as DTAM, both of these works approached depth map estimation as a minimization problem of pixel-wise photometric error and employed Total-Variation (TV) regularization in a framework tailored for processing on Graphics Processing Units (GPUs). Aiming for GPU-enabled real-time denser scene reconstruction than traditional SLAM, Pizzoli et al. [11] proposed a GPU-based mapping using the tracker of Forster et al. [12]. The demanding computational and memory requirements of all these methods, however, not only necessitates the use of power-hungry GPUs, but also restricts their operation in smaller spaces (e.g. a desktop area), prohibiting their employment onboard robots with limited resources and capabilities, such as small UAVs.

Pradeep et al. [13] proposed dense reconstruction employing keyframe-based SLAM in the background instead of performing global optimization regularly. Building a depth map at each frame by stereo matching between the current frame and a relevant, past keyframe, this approach incorporates the depth maps into a discrete, implicit model of the scene. On one side, this technique eliminates the need for the expensive optimization process of DTAM, but this system still runs on a GPU. It was Engel et al. [14], who proposed possibly the first system, which uses only a multi-core CPU and runs in

real-time. In their LSD-SLAM approach, the camera pose is tracked simultaneously with the propagation of scene-depth information from one frame to the next, evaluating depth measurements only for pixels in the current frame based on their uncertainty. LSD-SLAM builds a “semi-dense” scene reconstruction, which is not as accurate and certainly not as dense as the type of map that DTAM builds, but is far less computationally complex.

Aiming for a similar computational footprint as LSD-SLAM, Semi-Dense Mapping (SDM) [15] employs the keyframe-based ORB-SLAM [6] system, as we do in this paper, and in contrast to LSD-SLAM, which incorporates tracking inside the loop of denser mapping. Running on an independent thread, the densification is performed on each keyframe, accumulating experiences over time, which are subjected to continuous local and global optimization resulting to a very accurate, albeit rather sparse final map. As a result, despite exhibiting better accuracy than LSD-SLAM, SDM comes at the cost of both sparsity and computational time. Finally, the most recent DPPTAM [16] presented direct tracking and mapping using a first suggestion of a semi-dense map to grow planar regions in 3D (which they call ‘superpixels’) and as a result, demonstrating good performance and denser reconstructions for very careful camera motions. While DPPTAM was shown to achieve accurate reconstructions, similarly to SDM, the associated computational cost results to a prohibitive lag for most tasks requiring prompt scene estimation (e.g. path planning). With the aim on developing an approach able to overcome these limitations, the proposed approach is designed to produce accurate and rich reconstructions of the camera’s vicinity that are achievable within real-time constraints in real, practical scenarios.

Very recently, another dense approach was proposed by Teixeira and Chli [1], proposing a novel methodology for meshing out SLAM landmarks for extremely fast scene estimation. While offering a practical alternative for tasks that do not require accuracy, this is still an interpolation method that circumvents the estimation of the real scene depth.

III. METHODOLOGY

Inspired by works in offline and online scene reconstruction, the proposed method comprises of a set of carefully selected steps appearing in the workflow of Figure 2. In this section, we explain how the proposed pipeline works all the way from the camera pose estimation to obtain the scene reconstruction of the camera’s surroundings, with reference to the blocks in this workflow(Figure 2).

A. Camera Pose Estimation

As demonstrated to be a rather powerful monocular SLAM system, for the camera pose estimation we employ the publicly available ORB-SLAM [6], which is a keyframe-based approach using binary descriptors. As this work is about local scene reconstruction rather than global map generation, loop-closure detection is no longer relevant and

Fig. 2: The workflow of the proposed pipeline. With SLAM running in the background, the proposed Densification runs on a different thread reading the latest keyframe (KF) and its neighbouring KFs. The neighbour with the most favourable viewpoint is rectified together with the latest KF to compute a semi-dense depth map. Following the strict filtering of each estimated point, the resulting map is integrated with the propagation of previous depth maps into the current local environment representation and published. If the camera pose has not changed enough to generate a new KF, the intensity-based depth propagation is performed.

so is disabled. Naturally, any estimates produced from this monocular set-up are subject to an arbitrary scale factor. However there are IMU available in any aerial platform that can be trivially used to estimate the scale in a loose couple manner using the public available sensor fusion described in [17]. Alternatively, the recently published ORB-SLAM with IMU [18]. We use the work of Furgale et al. [19] in order to find the transformation between the IMU and the camera.

B. Good KF Pair Selection

When the latest keyframe (KF) provided by ORB-SLAM is new with respect to the last latest KF, the set of Neighbours is examined to select an appropriate pairing to the latest KF. These neighbouring KFs ('Neighbours') are identified using the covisibility graph maintained within ORB-SLAM and they later serve as candidates for selection of a good stereo pair in our Densification pipeline. Following the ORB-SLAM Local BA step, the Densification thread is informed publishing the latest KF, the Neighbours and the average depth z_{avg} of the tracked features in the latest KF. The pair of images selected will be the basis of the depth map to be generated, so the baseline between these two KFs determines the precision of the depth estimates [20]. The disparity error ϵ_d and the depth error ϵ_z are related by $\epsilon_z = \epsilon_d z^2 / bf$, with a baseline b , focal length f (in pixels) and depth z . Evidently, the larger the baseline the smaller the depth error, however the two images need to have sufficient overlap to allow for meaningful disparity estimation. Limiting the maximum disparity to d_{max} pixels and considering the range of depth values to be estimated, we use the relationship $z = bf/d$ to identify a suitable baseline for the selected pair, such that z_{avg} causes a disparity d equal to $d_{max}/2$.

The disparity estimation in Section III-C will not permit search for the same scene-point in the second image if its disparity grows further than d_{max} pixels. As a result, the closest scene-point to the camera (i.e. the minimum scene depth allowed) should cause image disparity equal to or smaller than d_{max} . However, in order to not rely on the scale estimate that is noisy, we propose to find a KF pair exhibiting a baseline such that z_{avg} causes image disparity equal to $d_{max}/2$.

Ideally, the image pair selected should have parallel optical axes to minimize the impact of viewpoint change and maximize the overlap. In order to favour candidates i with suitable baseline and smaller optical axis deviation θ from the latest KF ref , we propose the following cost function:

$$cost(i) = \left| \frac{d_{max}}{2} - \frac{b^i * f}{z_{avg}^{ref}} \right| + \left| C \tanh^{-1} \left(\frac{4\theta^i}{\pi} \right) \right|. \quad (1)$$

The first term in Equation (1) penalizes deviations from the desired baseline, while the second term is chosen to be lenient on small deviations of θ from zero, but increases the cost exponentially with larger values. C is a constant factor that we set to $d_{max}/5$ as a rough indication of the intended overlap between the two images. The candidate with the smallest cost is selected to pair up to the latest KF, provided that its θ does not exceed $\pi/4$ rad. If no such candidate is present, the process is aborted and the Densification thread collects newer data that ORB-SLAM has published.

C. Rectification, Depth estimation and Filtering

The rectification by Fusiello et al. [21] is used to align the images along the epipolar lines. Following rectification, Dense Matching [22] is used to estimate the pixel-wise disparities and the subsequent depth map; for each pixel in the latest KF, a match in intensity in the vicinity of its pixel coordinates is searched for along the corresponding epipolar line in the paired image. In order to keep a low computational cost, Dense Matching only searches in a row d_{max} pixels on these pixel coordinates. In our experiments, we set d_{max} to 12.5% of the image width, which is an indication of the expected camera dynamics.

Finally, the depth map is filtered for local inconsistencies in a process typically known as Support Filtering. On this step we differ from most algorithm because we use a very strong criteria in order to eliminate almost all noise besides to have to sacrifice part of the good points. We group the pixel where all the pixel in the same group have difference of less then 4 disparities. We eliminate all blobs with less than 300 pixels.

D. Temporal Fusion

Given that the filtering stage eliminates several good 3D points, we use two strategies in order to improve the density of the depth map; temporal fusion and the intensity-based expansion discussed below. For temporal fusion, we accumulate depth information across previous depth maps. Each such depth map consumes a significant amount of memory and the sometime contain only sparse information (e.g. due to small parallax), so fusing them into a bigger map can really be much more computationally effective and practical. Taking into account the motion dynamics of a UAV, we construct a cylindrical map aligned with the horizon having 3 times the image’s resolution in width and 1.5 times in height. Each point in the map holds the depth estimate and the intensity of the corresponding pixel from the corresponding KF, the depth error and optionally the surface normal (or the normal to the image plane the pixel comes from).

The map is progressively populated as new depth maps become available and a data point in the map is only replaced when new measurements are received for this location. More sophisticated information fusion will not offer much advantage in this setup, as the accuracy of the camera pose estimation becomes the limiting factor.

E. Intensity-based Expansion

It is typical that the first densification of the latest KF runs very fast and sometimes faster than real-time. In such cases where time permits or when the camera has not moved far enough to visit a new ‘latest KF’, the Densification process proceeds to improve the latest computed depth map. While Dense Matching as described in Section III-C provides all the depth estimates possible on the textured parts of the scene, it is most often the case that extended areas of the image are spanned by areas of homogeneous intensities, resulting to holes in the scene reconstruction. As a result, depth-map dilation operators can be used to naively grow the depth-map by consulting a neighbourhood window around each pixel. This process is prone to introduce inaccuracies in the map, so to mitigate this, Mur-Artal et al. [15] restricts this window to only the 8 immediate neighbours, which results in sparse-looking reconstructions.

In this work, we employ image segmentation to define meaningful boundaries for depth-map dilation. In this way, we can grow the same area in several iterations (still restricting the consultation window to 8 neighbours) without the risk of overgrowing in erroneous areas. For this purpose, we found the SLIC Superpixel segmentation [23] to be suitable, as it is one of the fastest available and produces a fairly regular grid of superpixels with the boundaries adjusted on the intensity borders. Moreover, as it is an iterative algorithm, it is adjustable to the computational capabilities of the platform at hand. In the experiments presented in this paper, we use 5 iterations of the Energy Maximization step and a maximum limit of 255 superpixels. Figure 3 shows the effect of the superpixel in the depth map.

In the case that the superpixel boundaries align with the depth estimates, there is a small risk that background

depth-values are dilated on foreground objects and vice-versa. However, this potential damage can be contained given the regularity of the grid and as our results demonstrate, superpixel-based dilation offers greatly increased recall at negligible impact the accuracy of the depth-map.

Fig. 3: Demonstration of how the superpixels helps to improve depth map using the image intensities as a guideline. The image on the left shows an image example with the superpixel contours. The central image is the result immediately after the Support Filtering and on the right we display only the additional pixels added by the superpixel-dilation.

IV. RESULTS

As the most relevant works in the literature, here we present an evaluation of the proposed pipeline with respect to LSD-SLAM [14], DPPTAM [16] and the Semi-Dense Mapping of [15], which we dub ‘SDM’. Similarly to DPPTAM and in contrast to LSD-SLAM and SDM, our methodology is aimed for accurate, *local* scene estimation. The aim here in particular is the use in tasks with hard real-time demands onboard computationally constraint platforms. For all experiments we use the publicly available implementations of LSD-SLAM and DPPTAM provided by the authors, adjusting the settings for each sequence to achieve the best possible performance. The comparisons with SDM, however, can only be either with respect to timings and on the qualitative performance of the achieved reconstructions, as presented in their paper [15]. For this reason, we present a first, extensive set of experiments on the TUM dataset and benchmark of [7], which offers RGBD Kinect data on a variety of indoor scenes. All timings are recorded on an Intel-i7 4700MQ processor, which is exactly the same as in [15] and similar to the processors in our onboard computer (Intel i7 5557U).

As the TUM benchmarking datasets in [7] are exclusively captured indoors with very slow and careful (hand-held) camera motions (at times, turntable-like), we also present results on sequences that have been captured outdoors using a WVGA global shutter on an Ascending Technologies Neo UAV, resulting to more dynamic and jerky movements approaching more practical, out-of-the-lab scenarios. These sequences and part of our experimental analysis on the TUM datasets can be visualised in the supplemental video submitted.

A. Evaluation on the TUM Benchmarking Datasets

1) *Comparison with DPPTAM*: The authors in [16] report relatively high density and high-accuracy for DPPTAM with respect to state of the art, especially in scenes

with large planar structures, as their method occasionally attempts to fit planar ‘super-pixels’ in the estimated 3D scene reconstruction. With their best results reported on the *fr3_nostructure_texture_near_withloop* sequence (the scene comprises of a single plane), here we select this to compare DPPTAM with our method.

To generate the results shown in Table I, we record the overall performance in terms of the percentage of pixels in each frame (i.e. the *Recall*) and the mean and median of the *Depth Error*, defined to be the deviation from the Kinect depth map serving as ground truth. For DPPTAM, we record directly the values averaged over the whole sequence as reported in [16]. In order to estimate the Depth Error of our method, we estimate the scale factor of the reconstruction based on maximal support across the whole sequence. This is a standard procedure in monocular set-ups since scale is unobservable. Finally, we also report on the computational cost of the pipelines in full (including tracking and mapping) in terms of average time per frame with and without the superpixel estimation/augmentation step for each method, reflecting how often a scene estimation is received by an external application. The timings are recorded for both methods on the same hardware (details aforementioned). Note that this is a slower computer than the one used in [16], but it corresponds to typical computational power for a small-scale multi-copter UAV.

Algorithm	DPPTAM [16]	Ours
Depth Error mean	2.84 cm (no std given)	2.50 ± 6.74 cm
Depth Error median	2.49 cm	1.61 cm
Average Recall	50 %	28 %
Average time without SP	1061 ± 1439 ms	18 ± 7 ms
Average time with SP	2554 ± 2863 ms	217 ± 318 ms

TABLE I: Comparison between DPPTAM and the proposed method on the *fr3_nostructure_texture_near_withloop* sequence of [7]. The DPPTAM Depth Error measures as well as the average Recall per frame are extracted directly from [16] (no standard deviation from the mean is provided). The average computation time per frame has been recorded on the same hardware using the author’s original code and includes the time for tracking and mapping, while we show the difference of performing (or not) the superpixel (SP) estimation step for each method.

Evidently, the two methods are comparable in terms of accuracy, while DPPTAM achieves a denser scene reconstruction than the proposed method. This is to be attained to the planar structure of the scene, which is a perfect fit to the planar model of the superpixels used in DPPTAM, in contrast to our method. The big difference however, lies in the timeliness of the result of scene estimation, with DPPTAM consuming an order of magnitude more time (two orders of magnitude without the superpixel estimation) before producing a reconstruction of the camera’s surroundings. In very slow sequences such as this and more generally all the benchmark sequences of [7], the camera travels a comparatively short distance within the 2-3 seconds that DPPTAM needs to produce an estimate of the local scene, meaning that the camera can still be in roughly the same area by the time that the scene estimation is available. DPPTAM

first estimates a ‘Semi-dense’ map and subsequently fits superpixels to it to complete the local map before starting off a new map, but the estimation is so slow that there is a noticeable lag with respect to the current view even in this sequence ¹.

Fig. 4: Comparison of LSD-SLAM with our pipeline on the *freiburg3_nostructure_texture_near_withloop* sequence. Following an independent scale alignment of each method, the top graph depicts the average Depth Error per frame (solid lines) and the standard deviation (in dashed lines of corresponding color). The naive fusion of depth-maps in LSD-SLAM is the main source of increased error, while the increased recall in our method is to be attained to the superpixel-dilation method and the homogeneity of intensities in this scene.

Fig. 5: A visual comparison of the frontal (top) and side-views (bottom) of the reconstructions obtained by LSD-SLAM (left) and our method (right) on the *freiburg3_nostructure_texture_near_withloop* sequence. This is captured on the last frame in the sequence.

2) *Comparison with LSD-SLAM and Superpixel Analysis:* As LSD-SLAM incrementally builds a global map of the environment, it promises a more timely scene reconstruction than DPPTAM. As our proposed method is local, we cannot directly compare the time to complete building a local map in the same way as we have done with DPPTAM, but we can compare the quality of the final reconstruction using the author’s implementation. Using the same sequence

¹The comparative performance of the two methods can be observed in the supplemental video

as before (*freiburg3_nostructure_texture_near_withloop*), Figure 4 illustrates the evolution of the average Depth Error per frame for the two methods, which have their scales independently estimated as above. As evident, the depth error of the proposed method is consistently negligible apart from some spikes caused by the tracker that we use. On the other hand, LSD-SLAM suffers bigger deviations, sometimes of the order of 50cm . In this sequence, the camera is on average about $1.5 - 2\text{m}$ away from the posters strapped on the floor, so the depth error of 50cm is rather significant. Note that while our method provides a depth map per frame, LSD-SLAM provides this only at keyframes resulting to fewer data-points in these plots. Overall, LSD-SLAM produces visually appealing reconstructions from viewpoints close to keyframes, however the naive projection of all acquired depth maps in the current keyframe’s field of view results to poor error performance and still lower Recall rates. The lack of large patterned areas in this sequence is the reason behind the lower Recall performance of LSD-SLAM than the proposed framework, which can be visualized in Figure 5 together with the accuracy of depth estimation.

The use of superpixels in our method to ‘bleed’ the depth estimates in areas of homogeneous intensity provides a rather cheap way of producing denser maps. This pays off rather well in sequences of large areas without texture such as the sequence used in Figure 4 and Figure 5, but in order to evaluate its use on a more realistically complex scene we use the *freiburg3_long_office_household* captured in a typical office environment. As demonstrated in Figure 6, the Recall typically increases (even doubles in some areas) when the superpixel-dilation is switched on, while the impact on the depth error is negligible. As evident in the computational time of our method recorded on all sequences (Tables I, II and III), this improvement step takes substantially more time than the first densified map result, however, it should be highlighted that this is an optional step run only when the real-time constrains permit (e.g. when the camera position has not changed significantly, permitting improvement of the previously computed map).

3) *Comparison with SDM*: For a qualitative assessment of the proposed method with SDM [15] Figure 7 presents visual illustrations of the depth maps acquired using our method on the *freiburg3_structure_texture_near* (top row) and *freiburg2_desk* (bottom row) sequences. In comparison to the visual illustrations for the same sequences presented in [15], the additional local and global optimization and clean-up steps in SDM undeniably offer more crisp and accurate-looking reconstructions. However, this comes at the cost of significant sparsity in their reconstructions and timings as evident in Table II. It is worth pointing out that since the proposed method relies on the same SLAM method as in [15] for camera pose estimation, both pipelines exhibit the same tracking quality, which was demonstrated to be more accurate than the tracking in LSD-SLAM in [15]. The difference of SDM with respect to this work, however, is that the proposed method offers richer and dramatically faster local reconstructions, reaching up to $17\times$ faster performance

Fig. 6: Evaluation of the use of superpixels in our pipeline to densify depth-maps on the *freiburg3_long_office_household* sequence (solid line is the mean depth error per image and dashed lines correspond to the st. deviation). The use of superpixels clearly improves the depth maps acquired, since they boost recall at almost no impact on the error.

Pipeline / Sequence:	A	B	C	D
SDM [15]	425 ms	376 ms	451 ms	308 ms
Ours - first result	24 ms	23 ms	31 ms	27 ms
Ours - after SP-dilation	287 ms	250 ms	290 ms	271 ms

TABLE II: Average timings per keyframe with a break-down for the stages of proposed method (‘Ours’) to produce the first resulting map and the improved map (when time permits), after superpixel-dilation (‘SP-dilation’) – note that the latter includes the time for the first result. The tests are run on sequences A: *fr2_desk*, B: *fr3_nostructure_texture_near_withloop*, C: *fr3_long_office_household* and D: *fr3_structure_texture_near*.

for the first result (following the initial densification) and 1.6 times faster with the superpixel-dilation step.

B. Evaluation on outdoor UAV datasets

In order to assess the applicability of the proposed method in one of the most challenging and relevant scenarios that it has been developed for. On the first experiment we use the consumer-grade DIY Quadcopter with diameter of 250mm featuring with an onboard rolling shutter Mobius ActionCam camera. The camera captures images of 1280×720 resolution at 60Hz . This platform is far more unstable than small UAVs traditionally used in aerial navigation (e.g. in [2]) resulting to challenging camera dynamics to work with but is very common setup for First Person View (FPV) today. It is important to note that DPPTAM fails to run on both UAV sequences presented in this section, due to the fact that the camera motion exhibited proves to be challenging for the DPPTAM tracker (it was designed for slower dynamics).

Flying along the facades of a building for about 40m to capture the ‘UAV Building 1’ 13-second sequence, both the proposed method and LSD-SLAM are run for cross-evaluation on the visual reconstructions that they compute, shown in Figure 8. Note that LSD-SLAM was incapable of tracking this motion at the original frame-rate of 60Hz , so

Fig. 7: Visual illustration of the reconstructions obtained by the proposed pipeline on the *freiburg3_structure_texture_near* and *freiburg2_desk* sequences for qualitative comparison with the equivalent results of SDM in [15]. While the latter achieves seemingly cleaner reconstructions, these are significantly sparser than the ones obtained with the proposed method.

Stage / Sequence:	UAV Building	UAV Building 2
Ours - first result	18 <i>ms</i>	43 <i>ms</i>
Ours - after SP-dilation	175 <i>ms</i>	272 <i>ms</i>

TABLE III: Average timings per keyframe in the UAV sequences, with a break-down for the two stages of the proposed algorithm (SP: superpixel). Note that the last row includes the time to produce the first result.

for the sake of comparison we slowed down this sequence 6-fold only for LSD-SLAM, keeping all the frames in the sequence (now of 78-seconds at 10Hz). As evident in the visual illustrations, LSD-SLAM produces appealing views from viewpoints close to the ones visited with the UAV, but when compared to the satellite image from Google-maps it is evident that the proposed pipeline outperforms LSD-SLAM in quality of estimation. On the case of filter even further the LSD-SLAM map, this will create a very sparse map.

As a final test, we use the professional “Neo” hexacopter by Ascending Technologies featuring an Intel i7 processor onboard that is capable to process the whole pipeline on-board. This UAV is noticeably more stable than consumer-grade UAV used to capture ‘UAV Building 1’. The images for this new ‘UAV Building 2’ sequence are captured at 20Hz using one of the two global shutter cameras present in the Skybotix VI-Sensor mounted at 45 degrees of pitch with 752×480 resolution. This was a 6-minute flight, with the UAV travelling a total distance of 300m. This, together with the challenging camera dynamics the UAV render not only DPPTAM, but also LSD-SLAM unusable in this sequence. This last experiment, highlights the adaptability of

the proposed system in both a variety of scenes and camera dynamics. Example images and a qualitative evaluation of this sequence is visible in Figure 1.

V. CONCLUSIONS

Following a study of some of the most prominent works in the area of offline and online scene reconstruction, this paper presented a new framework to produce local, dense scene representations of the camera’s workspace in real-time. This system has the capability to provide real-time feedback on the coverage of a structure of interest as well as the density of images captured in a particular region, which is key in high quality 3D reconstruction for digitization of archaeological structures.

While the real-time constraints and limited computational power typical in tasks such as UAV navigation prohibit perfect recall and accuracy rates, our comprehensive analysis on benchmarking datasets and on new, challenging UAV sequences, reveals that the proposed framework achieves the best balance of performance and complexity. Consistently recording high accuracy and density of scene reconstruction at dramatically low computational cost (at times 17-fold) the proposed method surpasses unrealistic assumptions on camera motion and the often prohibitive lag of state-of-the-art methods, opening up new possibilities in time-critical tasks on computationally constraint platforms for the first time. Future work directions will address the interfacing of the proposed method to UAV obstacle avoidance as the well as the use of these local maps as a basis for a global reconstruction of the camera’s surroundings.

(a) Ours: front

(b) Ours: top

(c) LSD-SLAM: top

(d) LSD-SLAM: front

Fig. 8: Comparison of the reconstructions on the UAV Building dataset with LSD-SLAM. The sequence was slowed down by a factor of 6 only when processing it with LSD-SLAM, such that a successful run was possible. Our method computes reconstructions based only on the local views in contrast to LSD-SLAM, which fuses all past experience in the resulting map. The frontal views from both pipelines are visually appealing, but when compared to the satellite image from Google maps in Figure 8b, LSD-SLAM is substantially more inaccurate.

REFERENCES

- [1] L. Teixeira and M. Chli, "Real-Time Mesh-based Scene Estimation for Aerial Inspection," in *Proceedings of the IEEE/RSJ Conference on Intelligent Robots and Systems (IROS)*, 2016.
- [2] S. Weiss, M. W. Achtelik, S. Lynen, M. C. Achtelik, L. Kneip, M. Chli, and R. Siegwart, "Monocular Vision for Long-term MAV Navigation: A Compendium," *Journal of Field Robotics (JFR)*, 2013.
- [3] A. J. Davison, N. D. Molton, I. Reid, and O. Stasse, "MonoSLAM: Real-time single camera SLAM," *IEEE Transactions on Pattern Analysis and Machine Intelligence (PAMI)*, 2007.
- [4] H. Strasdat, J. M. M. Montiel, and A. J. Davison, "Real-time monocular SLAM: Why filter?" in *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)*, 2010.
- [5] G. Klein and D. W. Murray, "Parallel tracking and mapping for small AR workspaces," in *Proceedings of the International Symposium on Mixed and Augmented Reality (ISMAR)*, 2007.
- [6] R. Mur-Artal, J. M. M. Montiel, and J. D. Tardós, "ORB-SLAM: a versatile and accurate monocular SLAM system," *IEEE Transactions on Robotics (T-RO)*, vol. 31, no. 5, pp. 1147–1163, 2015.
- [7] J. Sturm, N. Engelhard, F. Endres, W. Burgard, and D. Cremers, "A Benchmark for the Evaluation of RGB-D SLAM Systems," in *Proceedings of the IEEE/RSJ Conference on Intelligent Robots and Systems (IROS)*, 2012.
- [8] R. Newcombe, S. Izardi, O. Hilgig, D. Molyneaux, D. Kim, A. J. Davison, P. Kohli, J. Shotton, S. Hodges, and A. Fitzgibbon, "KinectFusion: Real-time dense surface mapping and tracking," in *Proceedings of the International Symposium on Mixed and Augmented Reality (ISMAR)*, 2011.
- [9] R. A. Newcombe, S. L. Lovegrove, and A. J. Davison, "DTAM: Dense tracking and mapping in real-time," in *Proceedings of the International Conference on Computer Vision (ICCV)*, 2011.
- [10] J. Stuehmer, S. Gumhold, and D. Cremers, "Real-time dense geometry from a handheld camera," in *Proceedings of the DAGM Symposium on Pattern Recognition*, 2010.
- [11] M. Pizzoli, C. Forster, and D. Scaramuzza, "REMODE: Probabilistic, monocular dense reconstruction in real time," in *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)*, 2014.
- [12] C. Forster, M. Pizzoli, and D. Scaramuzza, "SVO: Fast semi-direct monocular visual odometry," in *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)*, 2014.
- [13] V. Pradeep, C. Rhemann, S. Izadi, C. Zach, M. Bleyer, and S. Bathiche, "MonoFusion: Real-time 3D reconstruction of small scenes with a single web camera," in *Proceedings of the International Symposium on Mixed and Augmented Reality (ISMAR)*, 2013.
- [14] J. Engel, T. Schöps, and D. Cremers, "LSD-SLAM: Large-Scale Direct Monocular SLAM," in *Proceedings of the European Conference on Computer Vision (ECCV)*, 2014.
- [15] R. Mur-Artal, J. M. M. Montiel, and J. D. Tardós, "Probabilistic semi-dense mapping from highly accurate feature-based monocular SLAM," in *Proceedings of Robotics: Science and Systems (RSS)*, 2015.
- [16] A. Concha and J. Civera, "DPPTAM: Dense piecewise planar tracking and mapping from a monocular sequence," in *Proceedings of the IEEE/RSJ Conference on Intelligent Robots and Systems (IROS)*, 2015.
- [17] S. Weiss and R. Siegwart, "Real-time metric state estimation for modular vision-inertial systems," in *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)*, 2011.
- [18] R. Mur-Artal and J. D. Tardós, "Visual-inertial monocular slam with map reuse," *IEEE Robotics and Automation Letters*, 2017.
- [19] P. Furgale, J. Rehder, and R. Siegwart, "Unified temporal and spatial calibration for multi-sensor systems," in *IEEE/RSJ International Conference on Intelligent Robots and Systems*, Nov 2013.
- [20] D. Gallup, J. Frahm, P. Mordohai, and M. Pollefeys, "Variable baseline/resolution stereo," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2008.
- [21] A. Fusiello, E. Trucco, and A. Verri, "A compact algorithm for rectification of stereo pairs," *Machine Vision Applications*, 2000.
- [22] K. Konolige, "Small vision systems: Hardware and implementation," in *Robotics Research*, 1998, pp. 203–212.
- [23] R. Achanta, A. Shaji, K. Smith, A. Lucchi, P. Fua, and S. Susstrunk, "SLIC superpixels compared to state-of-the-art superpixel methods," *IEEE Transactions on Pattern Analysis and Machine Intelligence (PAMI)*, vol. 34, no. 11, pp. 2274–2282, 2012.