Real-Time Mesh-based Scene Estimation for Aerial Inspection

Author(s):
Teixeira, Lucas; Chli, Margarita

Publication date:
2016

Permanent link:
https://doi.org/10.3929/ethz-a-010696554

Originally published in:
https://doi.org/10.1109/IROS.2016.7759714

Real-Time Mesh-based Scene Estimation for Aerial Inspection

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

***Abstract*— With society and industry pushing for robot-assisted systems to automate cumbersome tasks, such as inspection and maintenance, a vast amount of research effort has been dedicated to relevant technologies. Right at the forefront are small Unmanned Aerial Vehicles (UAVs) equipped with onboard cameras, recently demonstrating that vision-based autonomous flights without reliance on GPS are possible, sparking great interest in a plethora of areas. Current solutions, however, still lack in portability and generality struggling to perform outside the controlled laboratory environment, with onboard robotic perception constituting the biggest impediment.**

Driven by the need for real-time denser scene estimation, in this work we present a dramatically low-computation approach enabling estimation of the immediate surroundings of a UAV using the inertial and visual cues from a single onboard camera. Instead of following the recent trend towards dense scene reconstruction, we trade detail of reconstruction for efficiency of estimation, albeit without compromising accuracy. We present results against scene ground truth obtained by a millimetre-precise laser scanner which we make publicly available together with our code. The ETHZ CAB Building dataset contains the ground-truth and visual-inertial data captured from both handheld and flying setups.

CODE AND DATASET

The video, dataset and code related to this work are available at: `http://www.v4rl.ethz.ch/research/datasets-code.html`

I. INTRODUCTION

Dense scene estimation from vision-based cues is a highly active research area at present, with new sensors and algorithms emerging continuously. Promising unprecedented perception of the environment, such techniques offer exciting prospects to robotic navigation and subsequently, interaction of robots with their environment. With applications ranging from robotic scene manipulation to robot navigation and path planning for fast moving Unmanned Aerial Vehicles (UAVs) addressed in this work, the needs for accuracy and speed of computation vary greatly.

Following the success of real-time monocular systems for Simultaneous Localization And Mapping (SLAM), substantial interest was triggered in achieving 3D scene reconstruction with a monocular camera. Early attempts to create richer online maps resorted to meshing the sparse feature-based SLAM map and dressing the facades of the mesh with portions of images with similar viewpoints, such as in [1]. While the result is visually more pleasing than a sparse map, the crudeness of these maps prohibits them from realistic use in tasks such as navigation for inspection of large structures as envisioned, for example in [2].

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).

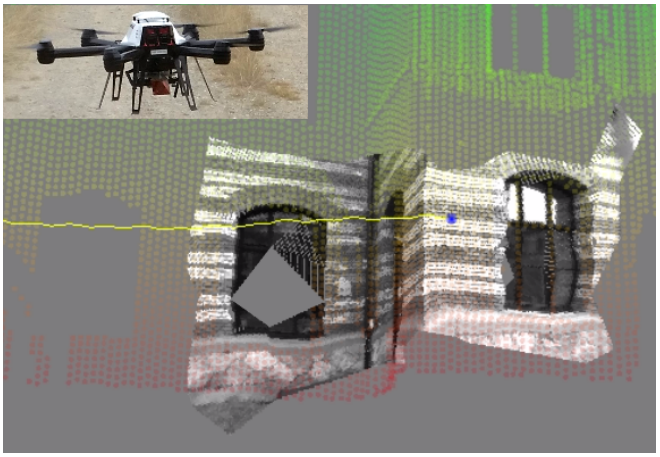

Fig. 1: The mesh created by the proposed algorithm in real-time providing a dense representation of the scene as seen from the current pose (blue point) of the UAV (visible in the inset). Each mesh-facade is dressed according to the image with the closest viewpoint to the facade's normal, considering all viewpoints experienced throughout the UAV's trajectory (yellow line). The laser point-cloud used as scene ground-truth is superimposed and colour-coded by height.

Addressing the need for real-time dense scene perception, research turned towards pixel-wise depth-estimation techniques, which in turn, meant processing of vast amounts of data necessitating the use of heavy-duty processors such as GPUs. In [3], Newcombe et al. proposed a real-time dense reconstruction technique running PTAM [4] on the background, while calculating a depth value for each pixel in a reference image using Total-Variation regularisation, obtaining dense surface models even in regions with low image texture. An important milestone in dense scene reconstruction has been the development of KinectFusion [5] demonstrating online scene reconstruction of impressive quality using a Kinect camera capturing both visual and depth information (aka RGB-D). However, it was with DTAM [6] that the Computer Vision community has been captivated, as it was the first to demonstrate that real-time dense reconstruction is possible using a visible-light camera achieving comparable quality to KinectFusion. The flip side, however, of these works lies with the amount of computation required, necessitating the use of power-hungry GPUs and restricting their operation to very small spaces and prohibiting them of employment onboard robots with restricted computational/power capabilities, such as small UAVs.

Circumventing the need for heavy GPU processing, Engel et al. in [7] proposed a novel SLAM method able of real-time dense scene estimation (i.e. denser than traditional feature-based SLAM) on a multi-threaded CPU. Performing depth measurements only for pixels in the current camera frame based on an estimate of their uncertainties, they simultaneously estimate the camera pose and propagate the depth information from frame to frame. Also targeting lower cost dense scene representation on a CPU, other works such as [8], [9] have recently emerged promising various levels of accuracy, density and computational complexity.

Motivated by the need for accurate dense scene representations for tasks, such as visual inspection from a UAV [2], here we propose a novel, simple approach to build on top of the basic SLAM functionality anyway running onboard a UAV on tasks requiring any autonomy. Via a mesh-based representation of the environment, the proposed method estimates the piecewise scene normals, the number of viewpoints a part of the scene has been observed from, as well as the distance of the camera from the scene. Our framework achieves an unprecedented balance between accuracy and computational complexity, especially suited to UAV navigation for inspection, where views perpendicular to the structure of interest are desired for effective visual inspection and potentially denser scene reconstruction.

Working with lower quality off-the-shelf webcams, Pradeep et al. [10] developed a 3D dense reconstruction technique without using global optimization and instead relying on PTAM-like SLAM running in the background. Their method creates a dense depth map at each frame by performing stereo matching between the current frame and previously selected keyframe and integrates the depth maps into a voxel-based implicit model. While the technique eliminates the need for the computationally expensive optimization process of DTAM, the system still needs to be parallelized on a GPU. Very recently, Engel et al. in [7] proposed a novel SLAM method, which runs in real-time on a multi-threaded CPU. Performing depth measurements only for pixels in the current camera frame based on an estimate of their uncertainties, they simultaneously estimate the camera pose and propagate the depth information from frame to frame. This results in a "semi-dense" reconstruction of the scene, not as dense and certainly not as accurate as a DTAM map, albeit in real-time on a CPU, bringing denser scene representations a step closer to realistic application onboard a robot.

Together with LSD-SLAM, the other most relevant pipeline to the one proposed in this paper is the most recent Densified ORB-SLAM in [8]. In contrast to LSD-SLAM, which puts tracking in the loop of denser mapping, in [8] as well as in the pipeline we propose here, the camera pose estimation is left to the keyframe-based ORB-SLAM [11]. Running on a separate thread, the densification is performed on every keyframe and in contrast to the proposed pipeline, [8] perform no image rectification and use a probabilistic measure in stereo matching for depth estimation, which together with their global optimization results to a very accurate, albeit rather sparse final map. As a result, despite demonstrating better accuracy than LSD-SLAM, in the framework in [8] this comes at the cost of both map-sparsity and quite importantly, computational time. With local and global optimization running continuously, the delay in acquiring a depth map of the current view prohibits its application to tasks such as obstacle avoidance and scene manipulation/grasping. With the focus on developing a methodology able to perform in these scenarios, the proposed approach, does not aim for acquiring a global map, but rather, accurate and denser views of the camera's vicinity, demonstrably circumventing the lack of accuracy of LSD-SLAM and the computational complexity and sparsity of ORB-SLAM.

Mesh-based approaches have long been explored for denser reconstruction. Using a stereo camera, Geiger et al. [12] create a mesh out of salient 3D points calculated on the stereo image pair and then use this mesh as prior for robust stereo matching. with runtime of about 800 ms per frame. Their method is not suitable for the hard real-time constrains and high frame-rate requirements during UAV navigation, while it also does not provide any pose information. The interactive-time approach using triangulated meshes of Daffry et al. [13] uses a bag of words approach to create matches between sequential frames, which is not very robust. These matches are then used to create a first triangulation subject to further optimization later. Our approach uses as input the sparse 3D point map coming from a SLAM system. Here, we present results using the visual inertial key-frame SLAM of[14]. This type of sparse map has high-quality and scale, but it is still too sparse to be used for many robotic tasks, such as path-planning for inspection. As a result, in this work we use a SLAM map that is anyway computed for navigation, as a basis for real-time mesh-based denser scene estimation.

II. Methodology

The proposed method extracts a dense representation of the scene including surface normals, based on the local neighbourhood of a sparse 3D feature map (as shown in Figure 1). As our method is coupled with a nominal keyframe-based SLAM algorithm that provides a 3D feature map, we use the knowledge of how the local point cloud is built in order to speed up the generation of the scene representation. Moreover, in order to achieve a computationally efficient algorithm, we also propose a scheme for fast mesh smoothing in 3D. The overall workflow of our method is as follows:

1) A keyframe-based SLAM system builds and maintains a sparse 3D feature map of the environment.
2) A probabilistic filter eliminates the worst points in the map, based on neighbourhood support.
3) The map points are projected onto the current image plane.
4) 2D Delaunay triangulation is performed on the image plane using the projected map points.
5) A 2D triangular mesh is obtained in the image plane.
6) Outlier detection is performed in 3D eliminating points that violate our specific constraints on the 2D mesh.

7) A smoothed out triangular mesh in obtained in 3D using all the points visible from the current viewpoint.
8) Attributes of the mesh are subject to interpolation, before the full scene representation is generated.

It is important to note that we relax the requirement for a complete mesh, i.e. a mesh with holes is often generated, as in autonomous motion planning, a whole surface representation of the scene is not a necessity. Supposing UAV motion parallel to the ground, a naive planner assuming that the size of the UAV corresponds to a single point, even a 1D profile of maximum depths and normals at each observed pixel is already enough to fly around any structure of interest. Here, a full representation of the scene in the neighbourhood of the camera is generated, while features violating the smoothness of the scene without enough neighbourhood support are penalised and areas around them corresponding to mesh facades can get eliminated. In effect, we trade a fully complete surface in turn for a high quality estimation of the surface normals wherever possible (i.e. in areas with stable, reliable 3D feature estimates).

A. UAV Pose and Scene Estimation using Monocular and Inertial Cues

The SLAM algorithm in the back-end of our algorithm is certainly the most important enabler of the scene representation estimation proposed here. SLAM using sophisticated laser range finders[15] promise some of the most accurate maps of the environment, however, they are not suitable for use onboard a small UAV due to their prohibitive weight and power consumption. Approaches using stereo vision or RGBD sensors are also more accurate than SLAM based on visual cues from a single camera, because they can allow instantaneous depth measurements of a point in the field of view, in contrast to the need for sufficient parallax in the monocular case. Despite this downside, monocular-based surface estimation not only is particularly suited to UAV navigation due to low weight, power and computation consumption, but also it eliminates the need for accurate extrinsic calibration in multi-camera setups, while it also not affected by direct sunlight as is the case with RGBD sensing. More importantly, the measurement range of a monocular setup is far more flexible than, say, a stereo one, as the desired baseline varies with the depth of the scene that needs to be estimated. This is particularly evident in aerial navigation, where the agility of UAVs is often the reason that the platforms are selected over other robots, while it is this agility that can vary the views of the scene of interest greatly, posing great challenges in accurate scene estimation. Integration of inertial cues is typical in UAV navigation [16] in order to provide more accurate instantaneous motion primitives than when using vision alone. Moreover, as Inertial Measurement Units (IMUs) are typically available onboard UAVs, it makes them a natural choice for use on the UAV sensor-suite.

While the proposed approach is largely agnostic to the SLAM algorithm used (it needs to be based on vision and use a keyframes-approach), here we choose to work with the monocular version of the visual-inertial SLAM implementation of [14], referred to as "OKVIS". This method performs local bundle adjustment over a window of keyframes close to the current time-stamp, resulting to constantly a constantly refined pose graph (and estimated point cloud) at the vicinity of the current keyframe. As input to our algorithm we use only the points in the current window that were found in the latest keyframe and we only generate mesh at keyframes (i.e. not on every frame). As most keyframe-based SLAM system, OKVIS provides information about the list of keyframes, where each feature was observed from, providing useful information on viewpoint estimation.

As input to our dense scene representation algorithm, we need the following from the SLAM approach used:

1) Camera intrinsics and distortion parameters.
2) The most recent camera frame and the 3D Points that were matched in it, in world coordinates.
3) If available, the quality q associated with each such 3D Point.
4) The transformation of the latest keyframe i to the world coordinate frame.

These serve as inputs to the mesh generation stage under the following assumptions:

- The projection of the features in the image plane is nearly perfect.
- The quality of each feature is inversely proportional to the uncertainty of the distance between camera origin and the feature in 3D, e.g. the depth value of the feature in relation to the camera.
- The depth of any feature within SLAM is calculated via triangulation of detections in 2 or more images.

B. Mesh Generation

Mesh Generation from point cloud is a ill-posed problem, because the same point cloud can generate many different solutions. This is particularly evident in typical, sparse feature-based SLAM maps, as some features can be completely disconnected from the rest. The assumption of world smoothness is commonly used to address this issue [3]. Other approaches include space carving and variational approaches [17], [18], [13]. Here, we aim for an efficient method and we bias meshing decisions towards grouping together locally co-planar regions.

To this end, we begin by creating 2D Delaunay Triangulation using the projections of the 3D points that the SLAM algorithm detects in the current frame. Then we apply our first condition over the triangulation in order to remove long edges (i.e. causing highly oblique triangles). Long edges are penalised since they expand the local-planarity assumption for each triangle over larger areas with little information. The second condition is the mesh cannot occlude a feature when observed from the two last keyframes, in which the feature was observed. In the next section we will explain how to apply this condition to the mesh. After the elimination of the problematic triangles and vertices, we apply an adaptation of the Laplacian smoothing algorithm.

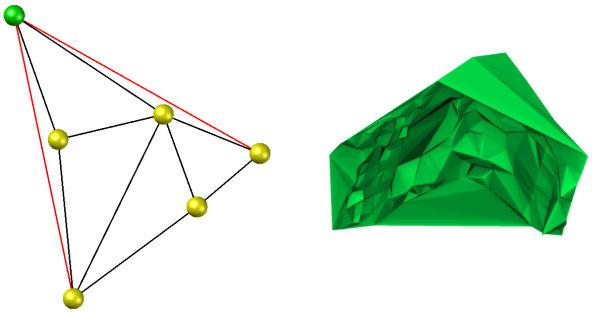

Fig. 2: A manifestation of unnecessary surface deformation due to the estimation of the convex hull during the Delaunay Triangulation (right), despite the often concave nature of the points. Instead, here long edges causing very oblique triangles in the mesh, such as the red ones on the left, are removed.

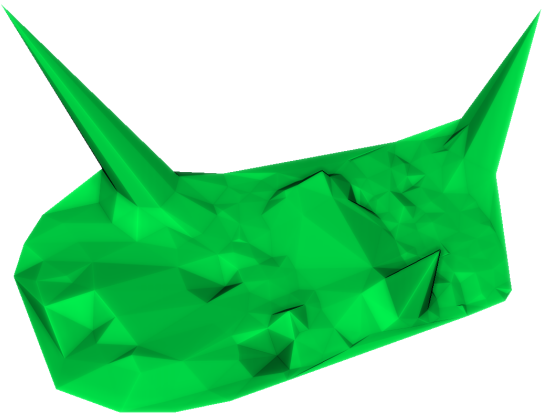

Fig. 3: The problem of large spikes in the meshing procedure. Such effects are generally associated with erroneous depth estimates of features in the SLAM map. This mesh was generated on images depicting a planar scene.

1) The Long-Edges Condition: There are two main cases, where long edges in Delaunay Triangulation are problematic. The first one is a consequence of the convex hull property; Delaney Triangulation produces the convex hull of the 2D points, while these point clouds often have a concave distribution. This is especially problematic in concave L-shaped corners, with the convex hull resulting to deformation of such corners creating virtual planes on the top and bottom of the corner as illustrated in Figure 2. In order to avoid such cases, here, we remove all triangles, in which the biggest edge is larger than the mean plus one standard deviation. Assuming a Gaussian distribution, 45% of the edges are eliminated with this step.

2) The Large Spikes Condition: Imposing this condition aims at removing big spikes in the mesh. These spikes are most often, a result of mismatches during the mapping process, usually due to similar features around each other. This problem is illustrated in Figure 3. This type of feature estimates are present in the map because we use the most recently updated map generated by the SLAM algorithm, often containing features that were observed just twice resulting to only one, often unreliable depth measurement. Instead of explicitly removing these vertices from the mesh, in order

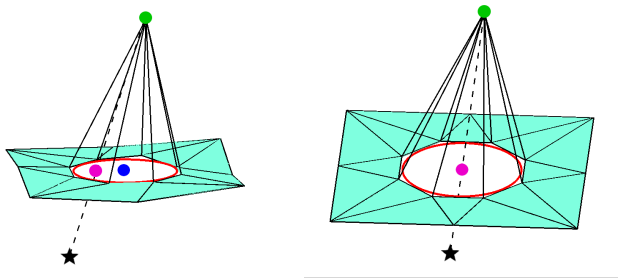

Fig. 4: An illustration of our approximation to the application of the Laplacian smoothing. On the left is a part of a mesh, where the local neighbourhood is not planar and the centroid of the neighbours of the green vertex, the blue point, is not aligned with the camera ray in direction to the origin of the camera (star). On the right is a depiction of the assumption we make in this case.

to avoid the cost of updating the Delaunay triangulation and all the book-keeping related with the graph, we just replace the depth of this vertex (coming from the corresponding feature in the SLAM map) with the average value of the neighbouring vertices in the mesh. We detect such spikes by comparing the distance between the 3D positions of each vertex and the average its neighbours in the mesh. Here, the threshold can be set either by consulting the covariance matrix or by setting an absolute value in the case that the 3D points are in metric scale (i.e. the user can set a suitable, stable absolute value). This procedure is strongly related to the Radius Outlier Removal Filter – in cases, where the facade under reconstruction is mostly planar, our approach will give the same result as this filter.

3) Approximated Laplacian Smoothing: The Laplacian smoothing operator is one of the most used in the field of mesh generation, because it provides a way of measuring the local curvature of the surface as represented in the mesh. The naive application of this operator, however, results to problematic cases and after discussing these cases below, we propose an approximation to this procedure more suitable to the nature of the vertices in this work. The main problem with the Laplacian operator is that it can deform areas of the mesh to convex structures. In our case, naive smoothing with the Laplacian operator causes rounding of the corners between surfaces. For this reason, here we perform a few smoothing iterations before obtaining the smoothed out mesh. Formally, Laplacian smoothing operator is,

$$\mu_k^i = \frac{1}{N} \sum_{r \in \Omega} x_r^i \tag{1}$$

$$x_k^{i+1} = (1 - \alpha) x_r^i + \alpha \mu_k^i \ , \tag{2}$$

where the x_k^{i+1} is the position of the feature k at the ith iteration. Ω is the set with size N, of the features directly connected with k in the mesh (neighbours). α is the damping term.

In order to avoid the violation of the assumption that the projection of the 3D map feature onto the current image plane is correct, we propose the following equation that varies only the depth of a feature without changing the projection onto

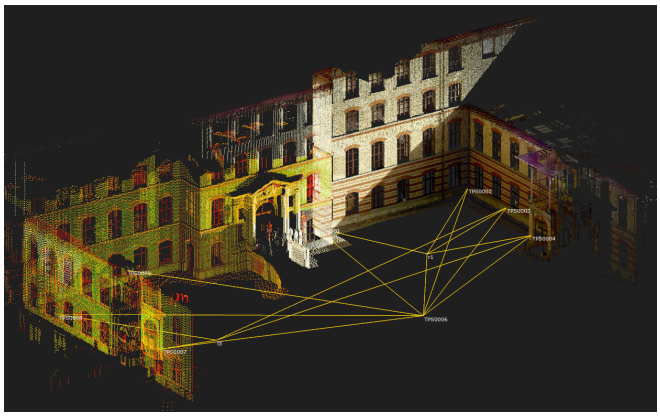

Fig. 5: A view of the scene ground truth captured with a high-precision laser scanner from different viewpoints.

the image plane.

$$x_k^{i+1} = \left(\alpha \frac{z_{\mu_k^i} - z_{x_r^i}}{z_{x_r^i}} + 1 \right) x_r^i \ , \tag{3}$$

where z_w if the depth component of the 3D point w. The effect of this approximation of the naive application of the Laplacian operator is illustrated in Figure 4.

The second approximation we make here, is to take account of the expectation of a regular grid of neighbours around the point of smoothing within the standard Laplacian approach, as this is not necessarily the case in the meshes obtained in this work. The traditional workaround to this problem is to resample the mesh to get a regular and sufficiently dense grid out of this mesh. Instead, here we rely on our pipeline to get a similar result more efficiently; firstly, a non-maximum suppression is applied over the features in the image plane, before the Delaunay Triangulation step – generally, a vision-based SLAM algorithm performs this step anyway internally and so there is no need to do it again. The Delaunay Triangulation builds a mesh that tends to be locally rather regular, since it maximizes the angles of the triangles. Moreover, with the elimination of the long edges both in 2D and the 3D using the large spike test, the meshing structure gets more regular.

III. The ETHZ CAB Building Dataset

As for scene ground truth the community typically resorts to simulated scenery and imagery [19], [20], in order to enable thorough evaluation of our work, we captured a dataset containing scene ground truth and real sensor data using a small rotorcraft UAV. To the best of our knowledge, there are no UAV datasets in real scenarios with scene ground truth in the literature at present. Both this dataset and the code for our mesh-based reconstruction are publicly available on the authors website

For scene ground truth, we used a high-definition laser scanner (Leica TS15 Total Station) offering millimetre precision, to scan the CAB Building of ETH Zurich. The resolution of the scans corresponds to one point per 0.15^o degree intervals, from 3 different positions in front yard of the building. A view of the scene ground truth can be seen in Figure 5. Note that every measurement point also holds the corresponding color data. Together with the laser point cloud, we also provide 3 aerial sequences captured at different distances from the building and heights (referred to as "Aerial 1-3"). A fourth one very close from the building (about $2m$ away from the facade of the building) is also available, but recorded using a hand-held setup, due to safety reasons (referred to as "Ground"). Each sequence was recorded with a VI-Sensor [21] that provides monochrome, global-shutter stereo images at $20Hz$ together with readings from a hardware-synchronised, high quality IMU.

IV. Results

Using the captured sequences posing different challenges as explained above, we generated the dense scene representation of the local vicinity of the camera with the proposed method, recording the deviation from the ground truth, as well as the computational time on an Intel-i7 4700MQ processor. For the mesh generation, we use the Fade2D Delaunay Triangulation software [22]. Our implementation was optimized in C++ avoid any expensive operations, such as trigonometric and power operators.

A. Qualitative Assessment of the Scene Estimation

In order to visually assess the quality of the scene representation obtained by the proposed approach, Figure 6 illustrated the obtained result on a view from the Aerial 2 sequence exhibiting interesting depth structure. As evident in the shape of the mesh and the surface normals visible through shading, the resulting representation, despite being an approximation to the underlying ground truth structure, it still follows closely the scene. Despite that the SLAM system used [14] is probably of the highest performing visual-inertial implementations at present, the feature points visible at any particular instant are still subject to erroneous estimates, especially in depth. As a result, reassessing all the candidate mesh vertices is absolutely necessary before a meaningful scene representation is obtained. Figure 8 is illustrates visually the effectiveness of the proposed method to diminish the effect of bad feature estimates, superimposing the obtained mesh with a mesh naively using all points visible from the current viewpoint – this is the same scene and view used in Figure 6, while Figure 7 shows the view from the camera with all the SLAM points being tracked.

B. Quantitative Evaluation of the normals against Scene Ground Truth

In order to assess the quality of the scene representation against our ground truth information on the scene, we evaluate the reconstruction accuracy of the system against a mostly planar scene. While the accuracy of the scene representation is directly affected by the quality of the SLAM estimated pose and points, since this varies with different SLAM implementations, this experiment is designed to analyse the estimation of normals by the proposed method, without the uncertainty of the alignment between of the ground truth and the camera. The ground truth curvature of this scene is visible in Figure 9. In red are the points with maximum curvature,

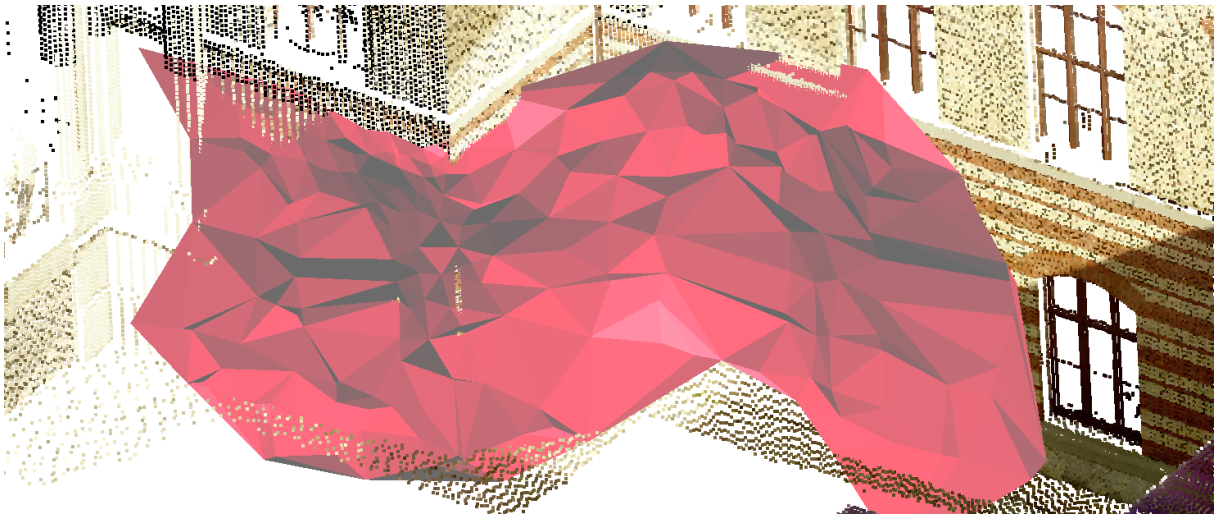

Fig. 6: In red is the computed mesh with shading providing visual feedback on the estimated normals calculated by our algorithm. This result, which was part of the Aerial 2 dataset obtained with $\alpha = 0.2$ and 5 iteration of smoothing, follows closely the scene ground truth shown in the background.

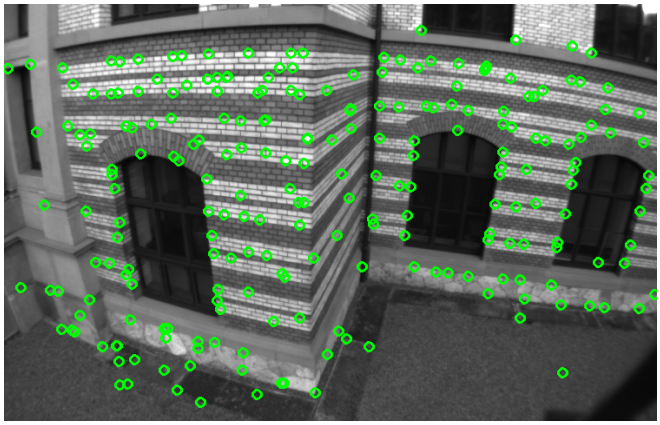

Fig. 7: The view from the camera superimposing the SLAM feature points successfully tracked in this frame and used for the mesh construction in both Figures 6 and 8.

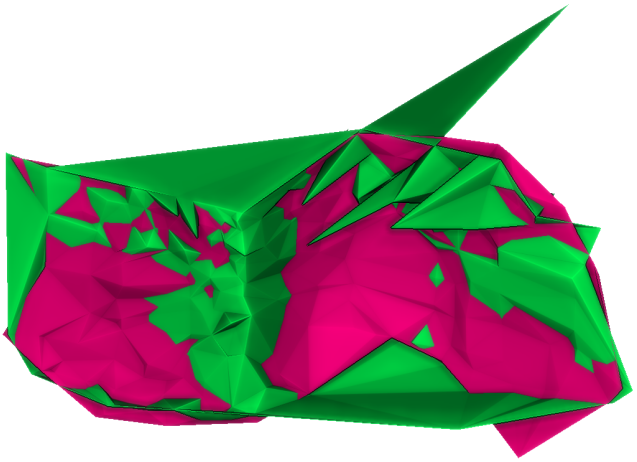

Fig. 8: The superposition of the initial mesh naively computed on the SLAM points visible from the current viewpoint in green, with the final mesh obtained using the proposed pipeline in red. This mesh is computed on the same scene as in Figure 6.

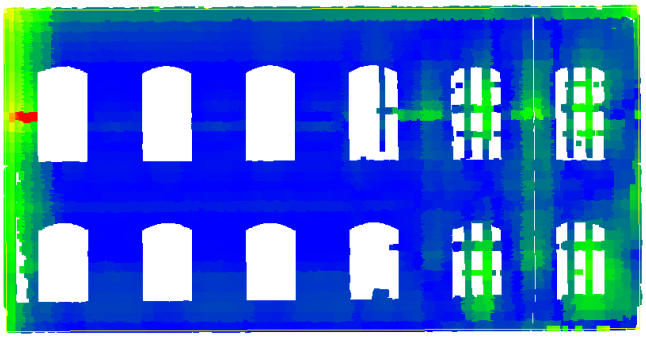

Fig. 9: The curvature of the scene extracted from the ground truth scan of the scene. The color map corresponds to blue for 0 curvature (i.e. planar area) up to red for maximum curvature.

No. iter.	α	mean	std	No. points
10	0.1	0.0789	0.0790	325k
5	0.2	0.12	0.129	515k
2	0.5	0.102	0.110	475k
Pure Delaunay		0.933	2.077	650k
Ground Truth		0.036	0.047	63k

TABLE I: The Mean Curvature values for different configurations captured on an $11s$-sequence as depicted in Figure 9. For comparison, we show the curvature values for pure Delaunay Triangulation and the ground truth. For each case, the total number of points participating in the estimation is shown. We use a spatial sampling of 5 cm over the triangulation. This is why configurations with more curvature tend to have more points. The number of points used for the ground truth is only related with the ground truth scan of the scene.

while blue corresponds to the minimum (i.e. planar areas). For this evaluation, we use the Ground sequence, because of the proximity it exhibits to the building's facade, allowing good views of largely planar scenes aims for this experiment. A sub-sequence of $11s$ from Ground, was used for this evaluation and the Mean Curvature value was recorded on point in each image at every frame. This is a standard method to analyse the smoothness of a surface and this value is direct proportional to the estimated normal at each point. Table I shows the Mean Curvature values for different configurations

No. iter.	Ground	Aerial 1	Aerial 2	Aerial 3
10	$7.1 \pm 3.7ms$	$5.7 \pm 1.8ms$	$6.9 \pm 2.3ms$	$< 1ms$
5	$6.8 \pm 1.9ms$	$5.0 \pm 1.7ms$	$5.2 \pm 1.5ms$	$< 1ms$
2	$6.6 \pm 1.8ms$	$5.3 \pm 1.7ms$	$4.9 \pm 1.7ms$	$< 1ms$

TABLE II: Average timings per keyframe on each sequence, using different number of iterations in the mesh generation. Evidently, the proposed approach is able to obtain a dense local scene representation at dramatically low computational cost.

of our method. It is important to highlight that the column-mean in this table is the mean of the Mean Curvature along all points that are originating from all keyframes in this test sequence. Evidently, the configuration with 10 iterations with a damping factor $\alpha = 0.1$ provides the best results, closely following the ground truth.

C. Timings

As the scene representation is only computed locally, the computational complexity of the proposed method is bounded and it depends on the number of candidate mesh vertices, which correspond to the features successfully tracked by the SLAM algorithm in the current view. With the emergence of binary features, such as BRISK [23] used in [14], current vision-based SLAM systems can handle numbers of the order of 100 features per frame, which was previously computationally infeasible (e.g. in [24]). Across all sequences that we test our approach on, the SLAM system tracks the scene with less than 200 features per frame (an example of the density of the features is visible in Figure 7). As evident in Table II, the proposed approach is extremely low cost achieving a dense, local scene representation even with 10 smoothing iterations of the mesh estimation. While there is no explicit records of the timings of the SLAM algorithm used in [14], in our implementation, the whole pipeline including SLAM estimation and dense scene estimation runs always well faster than real-time. Moreover, the dramatically low cost of computation of the proposed approach makes it applicable to other, potentially more expensive SLAM techniques.

V. CONCLUSIONS

In this paper we present a method achieving a dense, local scene representation using sensing cues from a single camera at a dramatically low computational cost. The proposed approach builds on top of a nominal monocular-inertial SLAM system to estimate a rough mesh-based representation of the scene, albeit of accuracy that permits high-fidelity navigation even for a highly agile and computationally constrained platform such as a UAV. We evaluate our system with respect to scene ground truth obtained using a laser scanner offering millimetre precision.

Further work, involves research into the use of the proposed method for more detailed scene reconstruction to speed up existing methods for both online and offline accurate, scene reconstruction.

REFERENCES

[1] S. Weiss, M. Achtelik, L. Kneip, D. Scaramuzza, and R. Siegwart, "Intuitive 3D Maps for MAV Terrain Exploration and Obstacle Avoidance," *Journal of Intelligent and Robotic Systems (JIRS)*.

[2] "AEROWORKS: Collaborative Aerial Workers," url http://www.aeroworks2020.eu, 2015.

[3] R. A. Newcombe and A. J. Davison, "Live dense reconstruction with a single moving camera," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2010.

[4] G. Klein and D. W. Murray, "Improving the agility of keyframe-based SLAM," in *Proceedings of the European Conference on Computer Vision (ECCV)*, 2008.

[5] R. Newcombe, S. Izardi, O. Hiliges, 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.

[6] 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.

[7] 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.

[8] 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.

[9] 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.

[10] 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.

[11] 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.

[12] A. Geiger, M. Roser, and R. Urtasun, "Efficient large-scale stereo matching," in *Asian Conference on Computer Vision (ACCV)*, 2010.

[13] S. Daftry, C. Hoppe, and H. Bischof, "Building with drones: Accurate 3D facade reconstruction using MAVs," in *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)*, 2015.

[14] S. Leutenegger, P. Furgale, V. Rabaud, M. Chli, and R. Siegwart, "Keyframe-based Visual-Inertial SLAM using Nonlinear Optimization," in *Proceedings of Robotics: Science and Systems (RSS)*, 2013.

[15] F. Moosmann and C. Stiller, "Velodyne SLAM," in *IEEE Intelligent Vehicles Symposium (IV), 2011 IEEE*, 2011.

[16] 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)*, vol. 30, pp. 803–831, 2013.

[17] D. Lovi, N. Birkbeck, D. Cobzas, and M. Jagersand, "Incremental free-space carving for real-time 3d reconstruction," in *Proceedings of the International Symposium on 3D Data Processing, Visualization and Transmission (3DPVT)*, 2010.

[18] C. Hoppe, M. Klopschitz, M. Donoser, and H. Bischof, "Incremental surface extraction from sparse structure-from-motion point clouds," in *Proceedings of the British Machine Vision Conference (BMVC)*, 2013.

[19] M. P. Martorell, A. Maki, S. Martull, Y. Ohkawa, and K. Fukui, "Towards a simulation driven stereo vision system," in *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)*, 2012.

[20] S. Martull, M. P. Martorell, and K. Fukui, "Realistic CG Stereo Image Dataset with Ground Truth Disparity Maps," in *In the TrakMark Workshop of the IEEE International Conference on Robotics and Automation (ICRA)*, 2012.

[21] J. Nikolic, J. Rehder, M. Burri, P. Gohl, S. Leutenegger, P. T. Furgale, and R. Siegwart, "A synchronized visual-inertial sensor system with fpga pre-processing for accurate real-time slam," in *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)*, 2014.

[22] "Fade2D Delayney Triangulation by Geom Software," url http://www.geom.at/fade2d/html/.

[23] S. Leutenegger, M. Chli, and R. Y. Siegwart, "BRISK: Binary Robust Invariant Scalable Keypoints," in *Proceedings of the IEEE International Conference on Computer Vision (ICCV)*, 2011.

[24] 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)*, vol. 29, no. 6, pp. 1052–1067, 2007.