COLMAP-SLAM: A FRAMEWORK FOR VISUAL ODOMETRY

L. Morelli [a,b], F. Ioli [c], R. Beber [a], F. Menna [a], F. Remondino [a], A. Vitti [b]

[a] 3D Optical Metrology (3DOM) unit, Bruno Kessler Foundation (FBK), Trento, Italy
Web: http://3dom.fbk.eu – Email: <lmorelli><rbeber><fmenna><remondino>@fbk.eu

[b] Dept. of Civil, Environmental and Mechanical Engineering (DICAM), University of Trento, Italy – Email: alfonso.vitti@unitn.it

[c] Dept. of Civil and Environmental Engineering (DICA), Politecnico di Milano, Milan, Italy – Email: francesco.ioli@polimi.it

Commission II

KEY WORDS: COLMAP, SLAM, Visual odometry, photogrammetry, navigation, mobile mapping, keyframe selection.

ABSTRACT:

SLAM technology is more and more integrated with other sensors for indoor and outdoor seamless navigation. This research topic is very active in particular on image matching with deep learning local features, keyframe selection approaches, or tests on new IMU and GNSS solutions. Integrating and testing new methodologies on other widely used SLAM implementations, such as ORB-SLAM, can be not a trivial task. Therefore, we propose an extension of COLMAP to be used in real-time as a feature-based Visual-SLAM that can be also coupled with other sensors. COLMAP has been chosen due to its modularity and the large community that assures the continuity of the repository. The paper presents a pipeline mainly thought for real-time evaluation of learning-based tie points and new SLAM features, that works with both monocular, stereo and multi-camera systems. It is also shown an example of keyframe selection algorithm based on deep learning local features, and a simple example of IMU integration. The code is available on the GitHub repository *https://github.com/3DOM-FBK/COLMAP_SLAM*.

1. INTRODUCTION

Accurate localization is a crucial task for robot navigation in applications such as autonomous driving and precise agriculture (Bai et al., 2023), where the required positioning accuracy can range from several decimeters to few centimeters. Over the years, the topic has received significant attention preferring the use of low-cost sensors and combining different technologies such as GNSS (Global Navigation Satellite System), LiDAR (Light Detection And Ranging), V-SLAM (Visual Simultaneous Localization And Mapping), VO (Visual Odometry), UWB (Ultra-Wide Band) and IMU (Inertial Measurement Unit). For integration of other sensors and further insight, see De Gaetani et al. (2019), Jia et al. (2019), Masiero et al. (2020), Gupta and Fernando (2022), Zhuang et al. (2023).

V-SLAM and VO are effective solutions typically applied in indoor applications and GNSS-denied environments, subjected to error drift in long trajectories. In particular, SLAM significantly reduces the drift error if loop-closure are detected (Singandhupe and La, 2019). In feature-based approaches (Scaramuzza and Fraundorfer, 2011) the detection and matching of reliable and repeatable interest points is of primary importance, but traditional approaches, like RootSIFT (Arandjelović and Zisserman, 2012), usually underperform in presence of wide-viewing angles, and strong illumination and radiometric changes (Jin et al., 2021; Morelli et al., 2022). These situations can be critical in real-time applications, looking for loop-closure under different viewing angles, or with strong illumination changes passing from brighter to darker aeras. In addition, accurate descriptors and keypoint repeatability are fundamental in presence of repetitive patterns or in poorly textured environments. Performance worsens if descriptors thought for lower computation resources instead of discriminative ones are used, such as ORB (Rublee et al., 2011) or SURF (Bay et al., 2006; Jin et al., 2021).

To overcome limitations of traditional hand-crafted approaches, learning-based methods are trained ad-hoc to be more reliable in challenging conditions, starting from TILDE (Verdie et al., 2015), to state-of-the-art SuperGlue (Sarlin et al., 2020) and LoFTR (Sun et al., 2021) methods. Nevertheless, in high accuracy scenarios, RootSIFT is still performing very well compared to learning-based approaches (Remondino et al., 2021; Bellavia et al., 2022a). In addition, most of new approaches are generally not invariant to rotation, except for few methods: Ono et al. (2018), Bökman and Kahl (2022) and Parihar et al. (2021) among end-to-end approaches, and Key.Net (Barroso-Laguna et al., 2019) + AffNet (Mishkin et al., 2018) + HardNet (Mishchuk et al., 2017) and Bellavia et al. (2022) among detect-then-describe approaches. For an extensive overview and evaluation of state-of-the-art in image matching please refer to Jin et al. (2021), Bellavia et al. (2022b), Morelli et al. (2022).

Despite the potential of these algorithms for SLAM and VO, they are usually tested for Structure-from-Motion applications. Few works tested them on SLAM: for instance, Mollica et al. (2023) extended the work of Campos et al. (2021), replacing ORB with SuperGlue.

To facilitate the integration of new image matching algorithms in SLAM, this paper proposes COLMAP-SLAM, an open-source framework in Python based on the COLMAP APIs (Schonberger and Frahm, 2016). It provides a modular software for developing new SLAM algorithms and in particular for a rapid integration and evaluation of new local features for feature-based SLAM/VO. Both monocular and stereo/multi-cameras cases are implemented, with support for hand-crafted as well as deep learning-based local features. The keyframe selection is based on the innovation of the optical-flow calculated with ORB or ALIKE (Zhao et al., 2022) features, but others can be easily integrated. Currently, loop-closure detection is supported only for the monocular scenario. If GNSS data is stored in the image EXIF tag, it is used to georeference the camera trajectory. The work is modular, allowing each single task to be studied independently. The main framework is in Python, and new features can be added using also other languages, and then called using Python as a wrapper.

The proposed pipeline is evaluated on the *EuRoC Machine Hall* dataset (Burri et al., 2016) and compared with OpenVSLAM (Sumikura et al., 2019), a VSLAM framework based on ORBSLAM2 (Mur-Artal and Tardós, 2017).

The aim of the paper is to understand the potential of COLMAP to be run in real-time, with a focus on the computational effort and the accuracy of the recovered camera poses. To the best of

This contribution has been peer-reviewed.

author's knowledge, this is the first open-source implementation of COLMAP to perform V-SLAM tasks, that includes multi-cameras support, support for different local features, and integration of GNSS data. Similarly, Nocerino et al. (2017) presented a 3D reconstruction pipeline to perform video acquisitions with a smartphone and geometric 3D reconstruction in the Cloud during multi-user concurrent or disjoint acquisition sessions. Another similar work is the project *An Offline Python SLAM using COLMAP*[1] that has not been released publicly. Finally, while we rely on COLMAP API, a similar work[22] based on PYCOLMAP[33] has been recently published on GitHub but not yet compared with our approach.

2. METHODOLOGY

2.1 Overview

The proposed pipeline for real-time navigation and mapping is built upon COLMAP. Specifically, its APIs are utilized for the extraction of RootSIFT local features, GPU matching and incremental bundle/reconstruction. The COLMAP engine was chosen due to its modularity and ease in integrating custom local features and matches.

A flowchart of the overall pipeline is reported in Figure 1 and its key points are:

- Frames from a single camera or a synchronized multi-camera system are saved locally, grouped per camera. Despite COLMAP can perform self-calibration, to limit computational time and improve accuracy, the calibration parameters of each camera should be provided.
- New frames are searched at fixed time intervals, and the keyframe-selection module (Section 2.2) selects only the frames that provide enough innovation in terms of tie points distribution. The keyframe selection is carried out only on the camera chosen as *master*. In future we plan to extend keyframe selection also to the other *slave* cameras. In fact, it is possible that on the *master* camera there is not enough innovation in terms of tie points distribution, while on the others the innovation can be significantly different, for instance because of a different viewing angle. To save computation time, an option can be to run keyframe selection on all frames on the *master* camera, and with a lower frame rate on the other cameras.
- Keyframes from the *master* camera are sequentially matched among different epochs, while the frames from the *slave* cameras are matched only with the synchronous master keyframes.

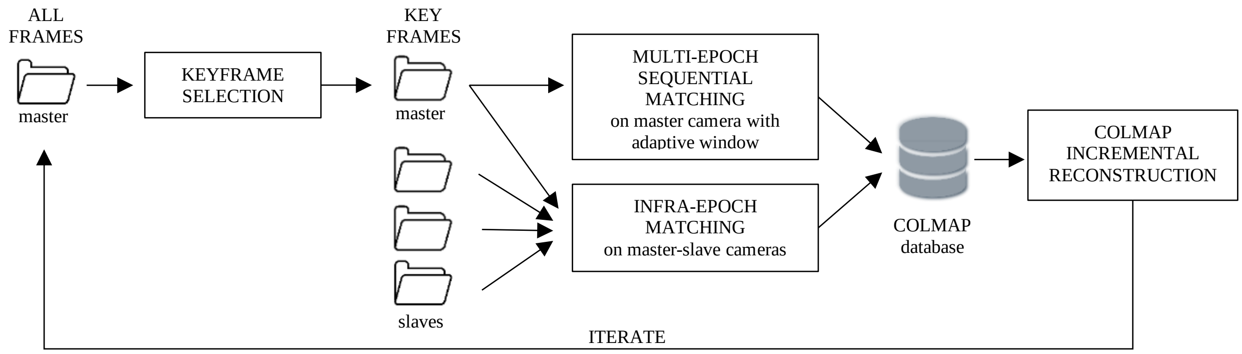

ALL FRAMES
master
KEYFRAME SELECTION
KEY FRAMES
master
slaves
MULTI-EPOCH SEQUENTIAL MATCHING on master camera with adaptive window
INFRA-EPOCH MATCHING on master-slave cameras
COLMAP database
COLMAP INCREMENTAL RECONSTRUCTION
ITERATE

Figure 1: Overview of the proposed COLMAP-SLAM pipeline.

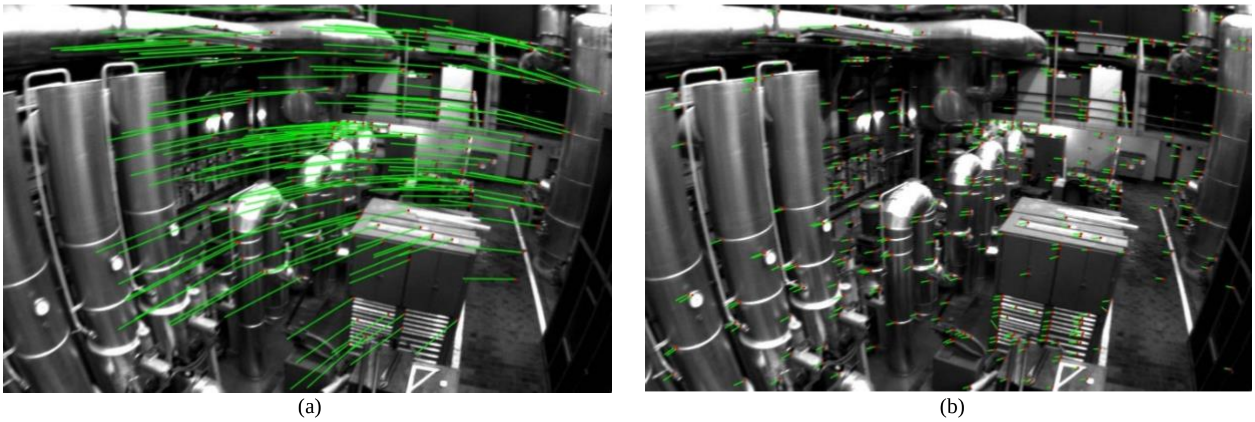


Figure 2: (a) Example of image from *EuRoC Machine Hall* (MH) 02 dataset selected as keyframe with a Median Matching Distance (MMD) of 105.74 px with respect to the last keyframe. Red dots represent the location of the features in the current frame, while green vectors denote the location of corresponding features in the last keyframe; (b) example of a frame that was not selected as a keyframe (MMD =11.05 px) and therefore rejected.

[1] https://tenhearts.github.io/3dv.html

[2] https://github.com/AlanSavio25/COLMAP_SLAM

[3] https://github.com/colmap/pycolmap

- All matches are saved in a SQLite database, then with the COLMAP *mapper* API new keyframes are registered, new tie points are triangulated, and finally 3D points and camera poses are adjusted. The process is repeated over again: new keyframes are searched among the new available frames, and the previous map is updated with new features and camera poses.
- The algorithm for the tie points extraction during the keyframe selection can be different from the one used for the incremental registration of new keyframes. This can be useful if the characteristics required by the keyframe selection (accuracy, computational resources, etc.) are different from those required during image registration.
- Different camera sensor configurations are supported such as monocular, stereo, and multi-camera. Currently, loop closure detection can be performed when the RootSIFT local feature is used, and it is supported only for the monocular case.
- The first batch of images (in our case 30 keyframes) defines the reference system. If GNSS data is available in the EXIF data, it is used to georeference the trajectory whereas IMU can be integrated as reported in Section 2.3. Currently the scale factor is calculated from the GNSS data or the stereo baseline using only the initial batch of images. If only one camera is used without additional information from other sensors, the camera trajectory is known up to a scale factor.
- For matching in real-time applications, RootSIFT, ORB, or ALIKE can be selected, while for offline simulations or post-processed image sequences additional local features can be used. Currently, SuperPoint (DeTone et al., 2018) and Key.Net + HardNet are supported. The offline option has been introduced to test local features that are too slow to be extracted in real-time or as an easier way to integrate and test local features. In fact, it is possible to previously extract the local features for all the frames in a target folder, then COLMAP-SLAM will look for keypoints when needed. Compatibility can be easily extended to a broader range of hand-crafted and deep learning-based local features.
- The *sequential_matcher* API of COLMAP is used for matching on a customized window of *k* (default *k*=1) images. For *n* non-oriented images, the matching window expands to *k*+*n* to increase the chance to ensure image tracking is not interrupted by image orientation failures. This approach adds re-localization capabilities within an adaptive matching, overcoming possible temporary obstacles in the scene or sudden change of illumination.

2.2 Keyframe selection

Keyframe selection is based on the optical flow innovation between the last keyframe and the current frame, based on corresponding local features. Therefore, optical flow for a tie point is defined as the 2D Euclidean distance in pixel between the coordinates of a tie point in the last keyframe and the coordinates of the same tie point in the current frame. At every iteration of SLAM loop, when a new frame is available, local features are extracted either with traditional ORB detector (Rublee et al., 2011) or with ALIKE (Zhao et al., 2022), a state-of-the-art differentiable key point detection algorithm, capable of sub-pixel accuracy and fast enough to run at 95 frames per second with 640×480 images on a commercial-grade GPU. Detected local features are then matched by using a cosine-similarity approach, that evaluates the similarity of n-dimensional vectors of the descriptors extracted. Corresponding matches are then filtered based on the epipolar constraint, by using Pydegensac (Mishkin et al., 2015; Jin et al., 2020). To decide whether the new frame brings enough innovation to the SLAM localization and thus should be selected as a keyframe the Median Matching Distance (MMD) is computed (Figure 2) and compared with a threshold. That is the median of the 2D Euclidean distance between matched keypoints in the current frame and last keyframe. Additionally, the algorithm checks if enough new matched features are found in the current frame to avoid rejecting it. ORB and ALIKE have similar performance: with both ORB and ALIKE, keyframe selection algorithm takes ca 0.05 s to evaluate a new frame and determine whether it should be designated as a keyframe or not (see Section 3 for hardware details).

2.3 IMU integration

The popularity of gyroscopes and accelerometers has increased significantly in the last years, and they are now commonly found in smartphones, cameras and robotic toys. The miniaturization of these sensors has been made possible through improvements in Microelectromechanical System (MEMS) technology, resulting in sensors with enhanced performance.

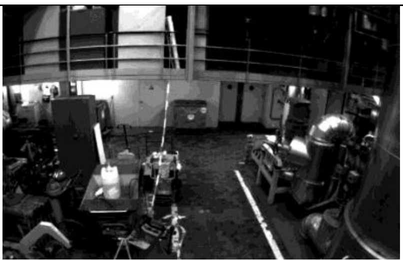
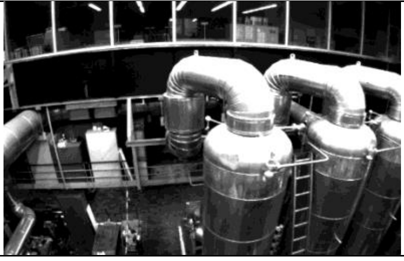
Machine Hall	Total # frames	Example frames	
MH_01_easy	3682		
MH_02_easy	3040		
MH_03_medium	2700		
MH_04_difficult	2033		
MH_05_difficult	2273		

Table 1. Example frames and total frame number for the five subsets of the *EuRoC Machine Hall* dataset.

An Inertial Measurement Unit (IMU) comprises 3 gyroscopes that measure angular velocity and 3 accelerometers that measure acceleration, along with gravity's direction. Although both the measurements of angular velocity and acceleration in the IMU reference system are valuable as separate observations, their integration through a sensor fusion algorithm increases the estimation accuracy of the IMU body frame orientation respect to the estimate you get integrating only gyroscope data.

Additionally, if a magnetometer is utilized in conjunction with the IMU, it is possible to establish the absolute measurement of orientation in relation to magnetic north, which facilitates the creation of an Attitude and Heading Reference System (AHRS). IMU raw data integration is performed via a revised AHRS algorithm[4] presented in chapter 7 of Madgwick (2014)'s PhD thesis. This is a different algorithm to the better-known initial AHRS algorithm presented in chapter 3, commonly referred to as the *Madgwick algorithm*. This is implemented in C and also made available by the authors through a Python package called *imufusion* available on PyPI. The algorithm, if magnetometer measurements are not available, can also be combined with an external source of heading measurements such as GNSS.

3. RESULTS AND DISCUSSION

COLMAP-SLAM's accuracy has been tested on the *EuRoC Machine Hall* dataset, that consists of five subsets classified by the authors according to an increasing difficulty from *easy*, *medium*, to *difficult* (Table 1). They present challenges due to variations in speed, high speed and slow movements as well as stationary sequences. An accurate ground truth is also provided to evaluate the performances of SLAM methods. Image sequences are fed as input in COLMAP-SLAM simulating a real-time acquisition. The accuracy of the computed camera poses is calculated as the Root Mean Square Error (RMSE) obtained from a *Helmert* transformation between COLMAP-SLAM trajectory and the ground truth trajectory. For each keyframe, 1024 keypoints have been extracted.

All tests described were run on an Ubuntu 20.04.3 LTS x86_64 machine, with Intel CPU i9-10900F (20) @ 5.200GHz and a NVIDIA GeForce GTX 1080 GPU.

3.1 Accuracy evaluation of monocular SLAM

The initial tests have been conducted for the monocular case on the *cam0* of the *EuRoC Machine Hall* 01 dataset. In this case, a simple keyframe selection approach based on subsampling the 20 Hz stream to 1 Hz has been used. For the evaluation of the approach described in Section 2.2, see Section 3.3 whereas results are reported in Table2.

When enabling loop-closure detection, COLMAP-SLAM achieved an RMSE of 3.3 cm, almost two times better than the 6.2 cm obtained using OpenVSLAM. This result can likely be attributed to RootSIFT of COLMAP-SLAM, which was reported in literature to be more accurate compared to ORB (Jin et al., 2021) used in OpenVSLAM. Without loop-closure detection, COLMAP-SLAM reached an RMSE two times worse respect the usage of loop-closures, an acceptable result considering the absence of loop-closures.

	OpenVSLAM	COLMAP-SLAM	
local feature	ORB	RootSIFT	RootSIFT
loop closure	yes	yes	no
RMSE [cm]	6.2	3.3	7.6

Table 2: Comparison of COLMAP-SLAM with OpenVSLAM on MH_01_easy monocular dataset.

3.2 Monocular vs stereo VO

In Table 3 the Machine Hall datasets 2 and 3 have been used to test the stereo case (*cam0* + *cam1*) against the monocular one (only *cam0*). In the stereo case, RMSE is calculated with a 6-parameters transformation between the estimated trajectory of *cam0* and the ground truth, since the estimate of the scale factor is known. The RMSE of the monocular case is calculated from the same trajectory of the stereo, without fixing the scale (7-parameters transformation). This test is useful because the stereo RMSE shows the accuracy of both trajectory and scale estimation, while the monocular RMSE shows the quality of the trajectory shape even if the scale factor estimation is not accurate.

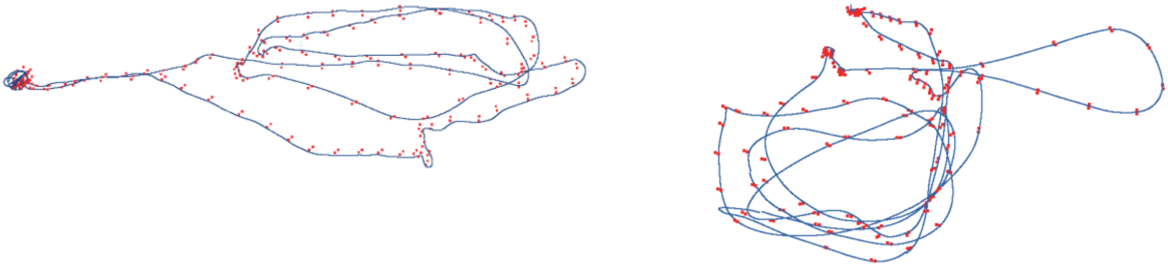
As in Section 3.1, frames at 1 Hz have been selected as keyframes. In *MH_02_easy* the COLMAP-SLAM stereo RMSE is 10.4 cm, almost double than the monocular RMSE (5.7 cm), while in *MH_03_medium* the stereo RMSE is 36.0 cm, more than three times the monocular one (9.0 cm). The higher error of the stereo is probably related to the scale factor that is calculated only on the first 30 keyframes and then kept fixed. Nevertheless, COLMAP-SLAM monocular errors are only 2-3 cm worse than OpenVSLAM (see Table 3), showing the good potential of COLMAP-SLAM in terms of trajectory shape. It also highlights that a more robust approach must be used for the calculation of the scale factor, for instance continuously updating it when new keyframes are added.

In addition, in this test COLMAP-SLAM results are without loop-closure detection (for both stereo and monocular), since such detection is not implemented yet for the stereo scenario. On the contrary, OpenVSLAM always performs loop-closures as feature generally contributes to achieve better results.

3.3 Keyframe selection

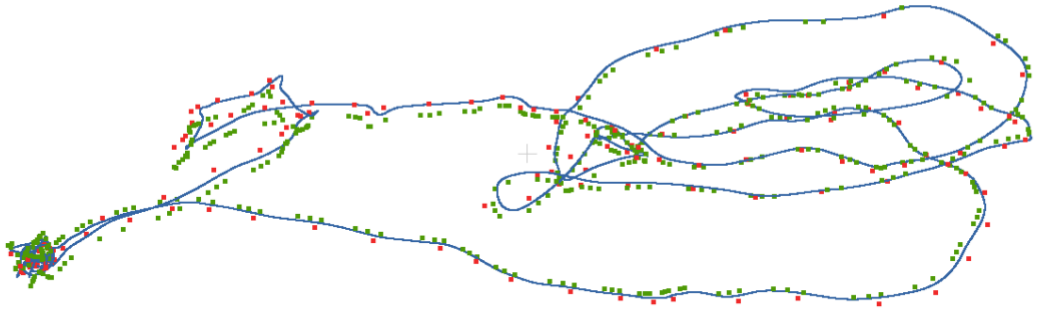
Table 4 presents a test for the role of keyframe selection, performed using only the left camera (*cam0*) of the *EuRoC Machine Hall* 01 (monocular case). For this initial test, a simple temporal selection approach was employed, comparing the results for selecting frames at 1Hz and 5Hz from the initial input stream of 20 Hz. To ensure statistical robustness, the experiment was run five times. No loop closure detection is used in this test. With one frame per second, an RMSE of less than a decimetre was achieved on a trajectory almost 80.6 meters long. However, using the timed criterium for frame selection, in the case of five frames per second, errors were more pronounced, likely due to the shorter baselines between the sequentially matched images. With shorter baselines the pose error accumulates more significantly due to sequential matching and windowed bundle adjustment (BA) being performed without any loop closure. Indeed, only two out of five runs achieved an accuracy better than one decimetre. The processing bottleneck is the global BA performed at each iteration, which significantly affects the computation time. Attempting to process more than 10 frames per second without keyframe selection leads to significant delays, which were therefore not included in the evaluation.

[4] https://github.com/xioTechnologies/Fusion



	OpenVSLAM with loop -closures	**COLMAP-SLAM** without loop-closures	
	Stereo RMSE [cm]	Monocular RMSE [cm]	Stereo RMSE [cm]
MH_02_easy	3.1	5.7	10.4
MH03_medium	5.7	9.0	36.0

Table 3: COLMAP-SLAM monocular vs stereo case. The ground truth trajectory is reported with a continuous blue line, while keyframes are reported with red dots.



colour	red	green
frame/sec	1	5
# keyframes	185	921
RMSE [cm]	10.3 / 7.9 / 7.3 / 4.9 / 7.4	9.9 / 121.1 / 15.4 / 110.9 / 11.2

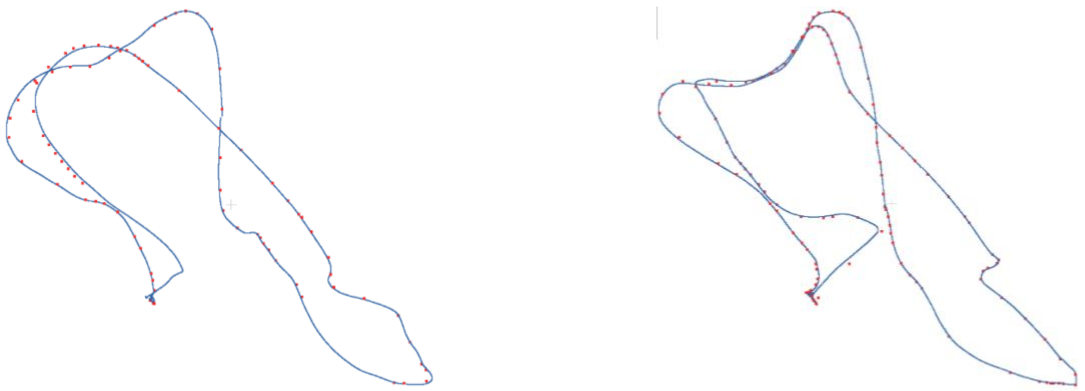
Table 4: Comparison on the impact of the number of processed keyframes with COLMAP-SLAM on MH_01_easy dataset. The ground truth trajectory is reported with a continuous blue line.

The keyframe selection algorithm presented in Section 2.2 was tested on full *EuRoC Machine Hall* 04 and 05 datasets (labelled as *difficult*), considering the left camera only *(cam0)*. Both ORB and ALIKE local features were used to compute MMD and assess the innovation of each frame with respect to the last keyframe. Keyframe selection was carried out by simulating a camera streaming images at 5Hz. This frame rate has been selected to be able to run in real-time both the keyframe selection and the incremental frame orientation. Innovation threshold (i.e., the value of MMD with respect to the last keyframe below which a frame is rejected) was set to 80 px, based on empirical tests. Selected keyframe were oriented by COLMAP sequential matching algorithm and the resulting camera trajectory was compared against the ground truth trajectory. Table 5 presents the results in terms of RMSE of the estimated trajectory and processing time, comparing both ORB and ALIKE local features on MH_04 (508 images, as the full dataset was subsampled at a frame rate of 5Hz) and MH_05 dataset (568 images). Keyframe selection took on average 0.05 s per frame evaluated both with ALIKE and ORB (with ALIKE that slightly reduces the total processing time compared to ORB). The total processing time including keyframe selection took less than 85 s for both MH_04 and MH_05 datasets, highlighting that COLMAP-SLAM can run in real-time with an input stream at 5Hz. For MH_5, the COLMAP-SLAM RMSE was 10.7 cm using ORB and 4.8 using ALIKE in keyframe selection, similar to OpenVSLAM with 7.3 cm. In the MH_04 dataset, on the other hand, the trajectory RMSE of OpenVSLAM diverged because a part of the trajectory was mis-estimated due to very low light conditions of a subset of images which affected ORB performances, while COLMAP-SLAM managed to fully reconstruct the whole trajectory.

3.2 Test for IMU integration

The *EuRoC Machine Hall 01* dataset has been chosen to test the *imufusion* Python package. In particular IMU ground truth position and orientation (in quaternions) are available in the Hexagon / Leica Geosystem (R) reference system (earth reference system). Meanwhile the accelerometers and gyroscope data from IMU are available in the body (B) reference system.

In the first two sub-plots of Figure 3 the gyroscopes and accelerometers raw data from IMU are reported with a frequency of 200 Hz. In the last sub-plot a comparison of the quaternions ground truth and the one predicted from IMU data is reported. There is a reasonable agreement in terms of data trend between the orientation predicted by the IMU (continuous line) and the ground truth (dotted line) with some divergencies with time, the bias in accelerometers and gyroscopes have to be properly corrected for. This test was made to understand the feasibility of using the *imufusion* Python package to set the initial orientation of the cameras and estimate the evolution of the orientation over time.

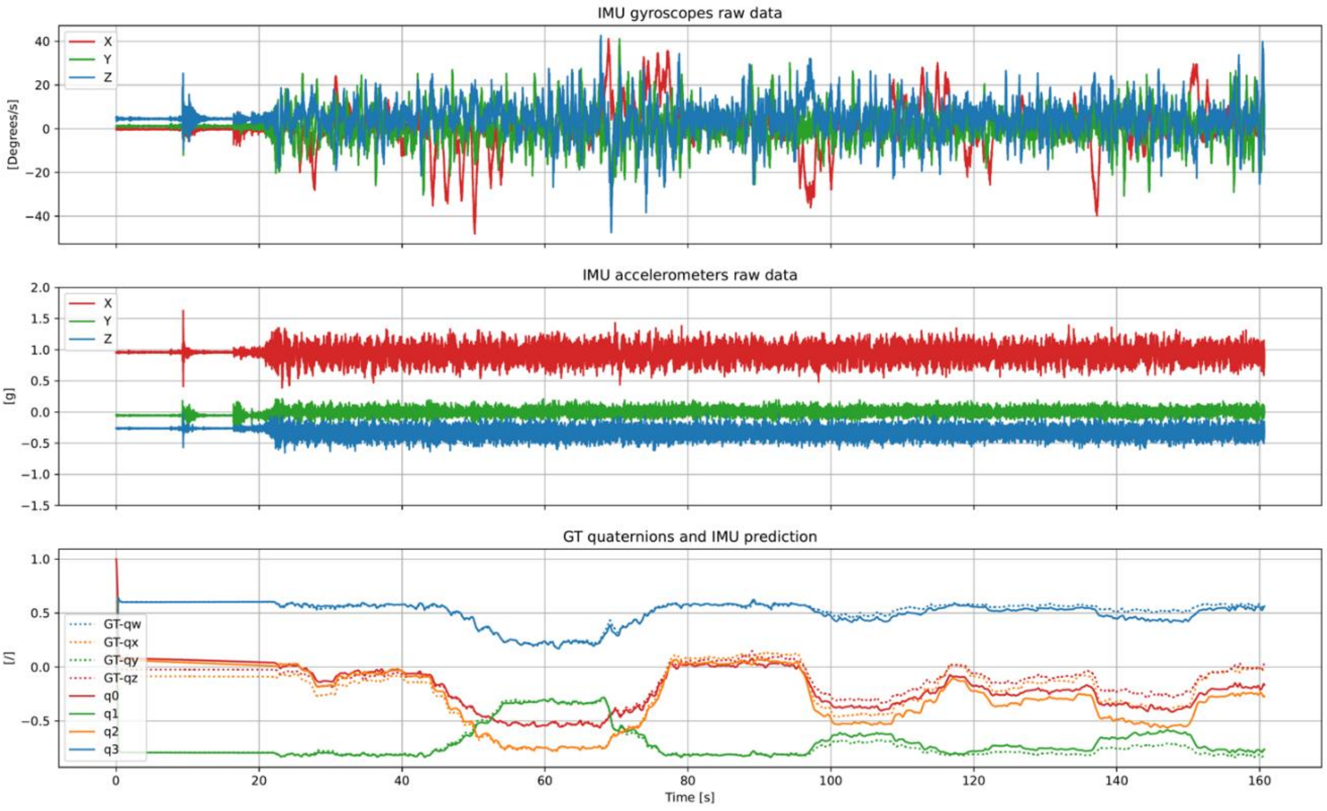


MH_04 difficult

MH_05 difficult

	OpenVSLAM with loop-closures	**COLMAP-SLAM** without loop-closures	
	Stereo RMSE [cm]	ORB for keyframe selection RMSE [cm] / # keyframes / Processing time [sec]	ALIKE for keyframe selection RMSE [cm] / # keyframes / Processing time [sec]
MH_04_difficult	989.9	6.5 / 98 / 72.59	11.5 / 93 / 72.53
MH_05_difficult	7.3	10.7 / 95 / 81.92	4.8 / 89 / 82.71

Table 5: Results of the keyframe selection algorithm test in the monocular scenario. For MH_04_difficult dataset the trajectory shown is obtained with the ALIKE keyframe selection, while for MH_05_difficult the keyframe selection method uses ORB. The ground truth trajectory is reported with a continuous blue line, while with red dots the keyframes are reported.



IMU gyroscopes raw data

IMU accelerometers raw data

GT quaternions and IMU prediction

Figure 3: IMU raw data and orientation comparison between quaternions ground truth and quaternion predicted by AHRS algorithm.

This is of particular interest during the acquisition process with the system because it enables the operator to re-initialize the orientation when the tracking is lost in an automatic fashion. Moreover, if the trajectory of the acquisition has been roughly defined a priori (i.e., acquisition on transept) the IMU orientation can be used, together with the last available velocity before losing tracking to estimate the next position of the system. If the orientation coincides within some degree with the prior orientation, then the system would have moved in the same direction meanwhile if the orientation is flipped of 180 degrees, then the operator have likely changed transept.

4. CONCLUSIONS AND FUTURE WORKS

The paper presented an open-source framework for the development of novel SLAM algorithms, with a particular focus on the inclusion and evaluation of learning-based detectors and descriptors. The framework is coded in Python and it is based on the COLMAP APIs for the extraction of RootSIFT local features, GPU-based matching and incremental bundle/reconstruction. It runs in real-time and it is modular in design to enable the targeted development of specific tasks. The platform supports both monocular and multi-camera systems. Results show that the proposed pipeline can achieve satisfactory results with an accuracy comparable to OpenVSLAM.

We plan to extend the framework under various aspects:

- Include cooperative SLAM (Poiesi et al., 2017), where mapping is performed by more than one moving platform.
- Extend loop-closure detection to other descriptors than RootSIFT and to the multi-camera systems. Currently, only loop-closure for RootSIFT under monocular scenario is supported.
- Improve and extend the integration of GNSS data to not be used only for scale definition on the initialization batch of images (default is the first 30 keyframes).
- Improve the scale estimation algorithm for multi-camera systems, that now utilize only the first batch of images for the estimation of baselines between the cameras.
- Add compatibility with Kornia (Riba et al., 2020) to use the wide range of local features already available and add further state-of-the-art local features.
- Currently, the keyframe selection is carried out only on the *master* camera while we plan to extend keyframe selection also to the other *slave* cameras.
- Integration of the IMU recovered orientation in the proposed pipeline.
- Include sensor fusion with Extended Kalman Filter.
- Improve efficiency and computational time.

REFERENCES

Afia, A.B., Escher, A.C., Macabiau, C. and Roche, S., 2015. A GNSS/IMU/WSS/VSLAM hybridization using an extended kalman filter. Proc. *ION 2015 Pacific PNT Meeting*, pp. 719-732.

Arandjelović, R. and Zisserman, A., 2012, June. Three things everyone should know to improve object retrieval. Proc. *IEEE CVPR*, pp. 2911-2918.

Azimi, A., Ahmadabadian, A.H. and Remondino, F., 2022. PKS: A photogrammetric key-frame selection method for visual-inertial systems built on ORB-SLAM3. *ISPRS Journal of Photogrammetry and Remote Sensing, 191*, pp.18-32.

Bai, Y., Zhang, B., Xu, N., Zhou, J., Shi, J. and Diao, Z., 2023. Vision-based navigation and guidance for agricultural autonomous vehicles and robots: A review. *Computers and Electronics in Agriculture, 205*, p.107584.

Bay, H., Tuytelaars, T. and Van Gool, L., 2006. Surf: Speeded up robust features. Proc. *ECCV*, LNCS, 3951, pp. 404-417.

Barroso-Laguna, A., Riba, E., Ponsa, D. and Mikolajczyk, K., 2019. Key.net: Keypoint detection by handcrafted and learned CNN filters. Proc. *ICCV*, pp. 5836-5844.

Bellavia, F., Morelli, L., Menna, F. and Remondino, F., 2022a. Image orientation with a hybrid pipeline robust to rotations and wide-baselines. *International Archives of the Photogrammetry, Remote Sensing & Spatial Information Sciences*, XLVI-2/W1-2022, 73-80.

Bellavia, F., Colombo, C., Morelli, L., Remondino, F., 2022b: Challenges in Image Matching for Cultural Heritage: An Overview and Perspective. Proc. *ICIAP*, LNCS, Vol 13373, Springer.

Bökman, G. and Kahl, F., 2022. A case for using rotation invariant features in state of the art feature matchers. Proc. *CVPR*, pp. 5110-5119.

Burri, M., Nikolic, J., Gohl, P., Schneider, T., Rehder, J., Omari, S., Achtelik, M.W. and Siegwart, R., 2016. The EuRoC micro aerial vehicle datasets. *The International Journal of Robotics Research*, *35*(10), pp. 1157-1163.

Campos, C., Elvira, R., Rodríguez, J.J.G., Montiel, J.M. and Tardós, J.D., 2021. Orb-slam3: An accurate open-source library for visual, visual–inertial, and multimap slam. *IEEE Transactions on Robotics*, *37*(6), pp. 1874-1890.

Cramariuc, A., Bernreiter, L., Tschopp, F., Fehr, M., Reijgwart, V., Nieto, J., Siegwart, R. and Cadena, C., 2022. maplab 2.0–A Modular and Multi-Modal Mapping Framework. *IEEE Robotics and Automation Letters*, *8*(2), pp. 520-527.

DeTone, D., Malisiewicz, T. and Rabinovich, A., 2018. Superpoint: Self-supervised interest point detection and description. Proc. *CVPR*, pp. 224-236.

De Gaetani, C.I., Pagliari, D., Realini, E., Reguzzoni, M., Rossi, L. and Pinto, L., 2019. Improving Low-Cost GNSS Navigation in Urban Areas by Integrating a Kinect Device. Proc. *IAG Scientific Assembly*, pp. 183-189.

Gupta, A. and Fernando, X., 2022. Simultaneous localization and mapping (slam) and data fusion in unmanned aerial vehicles: Recent advances and challenges. *Drones*, 6(4), p.85.

Jia, Y., Yan, X. and Xu, Y., 2019, December. A Survey of simultaneous localization and mapping for robot. *Proc. IEEE 4th Advanced Information Technology, Electronic and Automation Control Conference* (IAEAC), Vol. 1, pp. 857-861.

Jin, Y., Mishkin, D., Mishchuk, A., Matas, J., Fua, P., Yi, K.M. and Trulls, E., 2021. Image matching across wide baselines: From paper to practice. *International Journal of Computer Vision*, *129*(2), pp.517-547.

Madgwick, S. O., 2014. AHRS algorithms and calibration solutions to facilitate new applications using low-cost MEMS *Doctoral dissertation, University of Bristol, UK.*

Masiero, A., Perakis, H., Gabela, J., Toth, C., Gikas, V., Retscher, G., Goel, S., Kealy, A., Koppányi, Z., B¿aszczak-Bak, W., Li, Y. & Grejner-Brzezinska, D., 2020. Indoor navigation and mapping: Performance analysis of UWB-based platform positioning. *International Archives of the Photogrammetry, Remote Sensing and Spatial Information Sciences*, 43 (B1), 549-555.

Mishchuk, A., Mishkin, D., Radenovic, F. and Matas, J., 2017. Working hard to know your neighbor's margins: Local descriptor learning loss. Proc. *NIPS*, *30*.

Mishkin, D., Radenovic, F. and Matas, J., 2018. Repeatability is not enough: Learning affine regions via discriminability. Proc. *ECCV*, pp. 284-300.

Mollica, G., Legittimo, M., Dionigi, A., Costante, G. and Valigi, P., 2023. Integrating Sparse Learning-Based Feature Detectors into Simultaneous Localization and Mapping—A Benchmark Study. *Sensors*, *23*(4), p.2286.

Morelli, L., Bellavia, F., Menna, F. and Remondino, F., 2022. Photogrammetry now and then - from Hand-Crafted to Deep-Learning Tie Points. *International Archives of the Photogrammetry, Remote Sensing and Spatial Information Sciences*, *48*(2), pp.163-170.

Mur-Artal, R. and Tardós, J.D., 2017. Orb-slam2: An open-source slam system for monocular, stereo, and RGB-D cameras. *IEEE transactions on robotics*, *33*(5), pp.1255-1262.

Nguyen, A.Q., Ha, M.T., Tran, T.D., Ngo, D.A., Dao, N.P., Tran, D.T., Pestana, J., 2022. A Cloud-Based Visual Map Reconstruction for UAV Navigation Using Wireless Streaming. Proc. *IEEE ICCE,* pp. 319-324.

Nocerino, E., Poiesi, F., Locher, al, Y.T. Tefera, Remondino, F., Chippendale, P., Van Gool, L., 2017. 3D reconstruction with a collaborative approach based on smarthphones and a cloud-based server. *International Archives of the Photogrammetry, Remote Sensing and Spatial Information Sciences,* Vol. XLII-2/W8, pp. 187-194.

Ono, Y., Trulls, E., Fua, P. and Yi, K.M., 2018. LF-Net: Learning local features from images. *Advances in neural information processing systems, 31*.

Parihar, U.S., Gujarathi, A., Mehta, K., Tourani, S., Garg, S., Milford, M. and Krishna, K.M., 2021, September. RoRD: Rotation-robust descriptors and orthographic views for local feature matching. Proc. IEEE *IROS*, pp. 1593-1600.

Poiesi, F., Locher, A., Chippendale, P., Nocerino, E., Remondino, F., Van Gool, L., 2017. Cloud-based collaborative 3D reconstruction using smartphones. Proc. *CVMP*.

Remondino, F., Menna, F. and Morelli, L., 2021. Evaluating hand-crafted and learning-based features for photogrammetric applications. *The International Archives of Photogrammetry, Remote Sensing and Spatial Information Sciences*, *43*, pp.549-556.

Riba, E., Mishkin, D., Ponsa, D., Rublee, E. and Bradski, G., 2020. Kornia: an open source differentiable computer vision library for pytorch. Proc. *IEEE/CVF Winter Conference on Applications of Computer Vision*, pp. 3674-3683.

Rublee, E., Rabaud, V., Konolige, K. and Bradski, G., 2011, November. ORB: An efficient alternative to SIFT or SURF. Proc. *CVPR*, pp. 2564-2571.

Sarlin, P.E., DeTone, D., Malisiewicz, T. and Rabinovich, A., 2020. Superglue: Learning feature matching with graph neural networks. Proc. *CVPR*, pp. 4938-4947.

Scaramuzza, D. and Fraundorfer, F., 2011. Visual odometry [tutorial]. *IEEE robotics & automation magazine*, *18*(4), pp.80-92.

Schonberger, J.L. and Frahm, J.M., 2016. Structure-from-motion revisited. Proc. *CVPR*, pp. 4104-4113.

Singandhupe, A. and La, H.M., 2019, February. A review of slam techniques and security in autonomous driving. Proc. IEEE *IRC*, pp. 602-607.

Sumikura, S., Shibuya, M. and Sakurada, K., 2019, October. OpenVSLAM: A versatile visual SLAM framework. Proc. *27th ACM International Conference on Multimedia*, pp. 2292-2295.

Sun, J., Shen, Z., Wang, Y., Bao, H. and Zhou, X., 2021. LoFTR: Detector-free local feature matching with transformers. Proc. *CVPR*, pp. 8922-8931.

Verdie, Y., Yi, K., Fua, P. and Lepetit, V., 2015. Tilde: A temporally invariant learned detector. Proc. *CVPR*, pp. 5279-5288.

Younes, G., Asmar, D., Shammas, E. and Zelek, J., 2017. Keyframe-based monocular SLAM: design, survey, and future directions. *Robotics and Autonomous Systems*, *98*, pp.67-88.

Zhao, X., Wu, X., Miao, J., Chen, W., Chen, P.C. and Li, Z., 2022. Alike: Accurate and lightweight keypoint detection and descriptor extraction. *IEEE Transactions on Multimedia*.

Zhuang, Y., Sun, X., Li, Y., Huai, J., Hua, L., Yang, X., Cao, X., Zhang, P., Cao, Y., Qi, L. and Yang, J., 2023. Multi-sensor integrated navigation/positioning systems using data fusion: From analytics-based to learning-based approaches. *Information Fusion*, *95*, pp.62-90.