Work-in-Progress: Smart data reduction in SLAM methods for embedded systems

Quentin Picard, Stephane Chevobbe, Mehdi Darouich, Zoe Mandelli, Mathieu Carrier, Jean-Yves Didier

Work-in-Progress: Smart data reduction in SLAM methods for embedded systems

Quentin Picard
Université Paris-Saclay, CEA, List
F-91120, Palaiseau, France
quentin.picard@cea.fr

Stephane Chevobbe
Université Paris-Saclay, CEA, List
F-91120, Palaiseau, France
stephane.chevobbe@cea.fr

Mehdi Darouich
Université Paris-Saclay, CEA, List
F-91120, Palaiseau, France
mehdi.darouich@cea.fr

Zoe Mandelli
Université Paris-Saclay, CEA, List
F-91120, Palaiseau, France
zoe.mandelli@cea.fr

Mathieu Carrier
Université Paris-Saclay, CEA, List
F-91120, Palaiseau, France
mathieu.carrier@cea.fr

Jean-Yves Didier
IBISC, Univ Evry, Université Paris-Saclay
91025, Evry, France
jeanyves.didier@univ-evry.fr

***Abstract*—Visual-inertial simultaneous localization and mapping methods (SLAM) process and store large amounts of data based on image sequences to estimate accurate and robust real-time trajectories. Real-time performances, memory management and low power consumption are critical for embedded SLAM with restrictive hardware resources. We aim at reducing the amount of injected input data in SLAM algorithms and, thereby, the memory footprint while providing improved real-time performances. Two decimation approaches are used, constant filtering and adaptive filtering. The first one decimates input images to reduce frame rate (from 20 to 10, 7, 5 and 2 fps). The latter one uses inertial measurements to reduce the frame rate when no significant motion is detected. Applied to SLAM methods, it produces more accurate trajectories than the constant filtering approach, while further reducing the amount of injected data up to 85%. It also impacts the resource utilization by reducing up to an average of 91% the peak of memory consumption.**

***Index Terms*—SLAM, localization, embedded systems, real-time, data reduction, memory management.**

I. Introduction

Autonomous and mobile systems such as service robots, micro air vehicle and augmented/virtual reality devices require real-time performances, memory management and low power consumption for embedded systems. A large number of embedded platforms with restrictive resources [1] are available to implement advanced perception functions with simultaneous localization and mapping/visual-inertial odometry (SLAM/VIO) methods. Several embedded SLAM are implemented and aim to reduce the number of performed operations to minimize resource consumption while maintaining real-time processing. The implementation on resource-constrained systems includes trade-offs between resources usage and localization accuracy characterized by reducing the number of features per frame and the size of the optimization sliding window [2]. In [3], the co-design (between software and hardware) process of a VIO system on FPGA based on algorithmic, implementation and parameters choices is presented.

Our work aims to reduce the resource utilization of real-time localization methods based on input images and inertial measurements. Our main contribution is a smart data reduction relying on an adaptive filtering approach, based on embedded system motion estimation, to decrease the input frame rate. The goal is to show that 1/ the accuracy of SLAM methods is similar to the baseline with constant 20 fps and 2/ the memory consumption is drastically reduced.

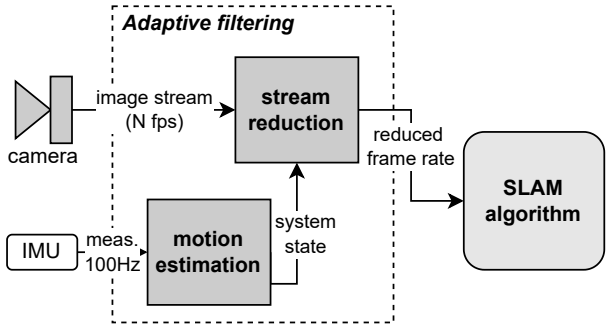

Fig. 1. Functional pipeline for the adaptive filtering approach

II. Adaptive filtering approach

The adaptive filtering (AF) approach is a decimation strategy to control the input frame rate for SLAM/VIO algorithms as shown in Figure 1. Motion is estimated in a fast and simple way from data provided by the inertial measurement unit (IMU) to modulate input image stream through the stream reduction: the lower the measured speed, the lower the input image bandwidth. The more movements there are, the more images are forwarded. When no motion is detected, a minimum frame rate is applied to capture other scene changes such as variations in luminosity. Here, we will apply a constant threshold for motion estimation.

III. Experiment and results

To measure the impact of our approach on the accuracy of the trajectory estimation and memory consumption, we identified and selected four SLAM methods, KimeraVIO [4],

TABLE I
PERFORMANCES OF AF ON EUROC* AND TUMVI† SEQUENCES.

Methods	Baseline (input 20 fps)		Adaptive filtering (AF)	
	ATE (m)	(MB)	ATE (m)	(MB)
KimeraVIO	*[0.05-3.49]	*1700	*[0.05-0.49]	*507.19
ORBSLAM3	*[0.01-0.10]	*730.12	*[0.01-0.10]	*724.25
	†[0.01-0.14]	†827.74	†[0.01-0.05]	†829.19
VINSFusion	*[0.12-0.54]	*1330	*[0.09-0.33]	*117.98
	†[0.06-0.87]	†989.55	†[0.06-1.96]	†228.69
OpenVINS	*[0.05-0.22]	*45.18	*[0.05->100.00]	*46.93
	†[0.06-0.83]	†N/A	†[0.06-0.76]	†N/A

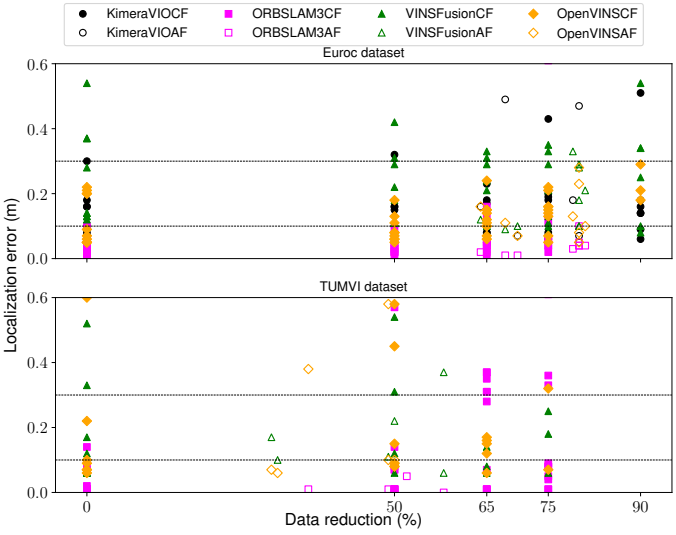

Fig. 2. Localization error of SLAM methods with input data reduction.

ORBSLAM3 [5], VINSFusion [6] and OpenVINS [7] for their localization accuracy, their computational complexities and their local and/or global optimization of 3D poses. To evaluate the performances of AF, we compare it to the constant filtering (CF) approach. It decimates the input data at a constant rate, regardless of the camera movement. For our comparisons, we reduced the input 20 fps baseline frame rate to 10, 7, 5 and 2 fps. The accuracy is defined by the absolute translation error (ATE) with the $SE(3)$ Umeyama alignment. We used the *heaptrack* heap memory profiler tool to evaluate the impact on memory consumption. Two datasets are used to assess the performances of the localization methods with the input data reduction: Euroc for drone sequences [8] and TUMVI for handheld sequences [9]. Each one has been pre-processed with CF and AF, and then injected to each SLAM/VIO methods. To overcome the variability due to the non-deterministic nature of SLAM methods, we use the median of the estimated trajectory ($\widehat{E}$) accuracy over 5 runs.

Figure 2 plots the localization error of $\widehat{E}$ with respect to the reduction of the input data for all tested SLAM algorithms and sequences. The first important conclusion is drawn from the results with CF: the data reduction by constant rate decimation at the input of the SLAM methods obtains a relatively good accuracy compared to the reference (0% reduction). On the Euroc dataset, for CF, the best accuracy is achieved with data down to 5-7 fps, which corresponds to a 65-75% data reduction. The results obtained on AF show that it is possible to drastically reduce the amount of injected images and, therefore, data (between 75% and 90%) while providing low localisation errors for most of the SLAM methods. Moreover, the graph shows that despite the higher difficulty of the TUMVI datasets, AF still reduces the amount of injected data while keeping the accuracy comparable to the reference, even if it is less important than the one obtained on the Euroc dataset (reduction of about 50% instead of 75% for Euroc).

Table I provides performances for baseline (input 20 fps) and AF in terms of localization error (ATE) and memory consumption (MB). The range of localization error is provided for both datasets, which highlights the impact of AF used with SLAM/VIO methods. The average memory consumption is given for Euroc and TUMVI. It shows a significant gain for KimeraVIO and VINSFusion that are using input frame buffers contrary to OpenVINS that do not have one. Therefore, the main impact of AF for this algorithm is the significant reduction of the number of operations due to the decimation of processed images.

IV. CONCLUSION

This work proposes to evaluate the robustness and the performances of SLAM/VIO algorithms in terms of localization error and memory consumption when applying our smart input data reduction. We proposed to modulate the input data stream using a simple motion estimation from inertial measurements. We show that the amount of injected data is drastically reduced down to 85% while the accuracy remains similar to the baseline input at 20 fps. Significant reduction of memory consumption is achieved with a decrease up to 91%.

REFERENCES

[1] A. N. Wilson, A. Kumar, A. Jha, and L. R. Cenkeramaddi, "Embedded sensors, communication technologies, computing platforms and machine learning for uavs: A review," *IEEE Sensors Journal*, 2022.

[2] J. Delmerico and D. Scaramuzza, "A benchmark comparison of monocular visual-inertial odometry algorithms for flying robots," in *IEEE International Conference on Robotics and Automation (ICRA)*, 2018.

[3] Z. Zhang, A. Suleiman, L. Carlone, V. Sze, and S. Karaman, "Visual-inertial odometry on chip: An algorithm-and-hardware co-design approach," in *Robotics: Science and Systems*, 2017.

[4] A. Rosinol, M. Abate, Y. Chang, and L. Carlone, "Kimera: an open-source library for real-time metric-semantic localization and mapping," in *IEEE Intl. Conf. on Robotics and Automation (ICRA)*, 2020.

[5] C. Campos, R. Elvira, J. J. G. Rodríguez, J. M. M. Montiel, and J. D. Tardós, "Orb-slam3: An accurate open-source library for visual, visual–inertial, and multimap slam," *IEEE Transactions on Robotics*, 2021.

[6] T. Qin, S. Cao, J. Pan, and S. Shen, "A general optimization-based framework for global pose estimation with multiple sensors," *arXiv preprint arXiv:1901.03642*, 2019.

[7] P. Geneva, K. Eckenhoff, W. Lee, Y. Yang, and G. Huang, "Openvins: A research platform for visual-inertial estimation," in *IEEE International Conference on Robotics and Automation (ICRA)*, 2020.

[8] M. Burri, J. Nikolic, P. Gohl, T. Schneider, J. Rehder, S. Omari, M. W. Achtelik, and R. Siegwart, "The euroc micro aerial vehicle datasets," *The International Journal of Robotics Research*, 2016.

[9] D. Schubert, T. Goll, N. Demmel, V. Usenko, J. Stückler, and D. Cremers, "The tum vi benchmark for evaluating visual-inertial odometry," in *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2018.