



The Role of Modularity in Multimodal Simultaneous Localization and Mapping Systems

Petros Kapsalas, Panasonic Automotive Systems Europe

Aris S. Lalos and Dimitrios Serpanos, Industrial Systems Institute/ATHENA

Konstantinos Moustakas, University of Patras

Simultaneous localization and mapping (SLAM) refers to the problem of mapping an environment using measurements from mobile sensors while simultaneously estimating the motion of those sensors relative to the map. Modular architectures are required to enable the commoditization and fast penetration of SLAMs in the emerging mobile computing systems.

Simultaneous localization and mapping (SLAM) is considered an abstract problem, but robust and efficient solutions are critical to enable emerging intelligent mobile devices.¹ From fully autonomous vehicles to markerless augmented reality (AR) and from gaming to household robotics, the ability to automatically estimate the 3D structure of the environment and track a device's position and orientation (pose) provides the necessary foundation to enable high-level interpretation, visualization, and interaction. Visual SLAM, where a camera is the primary sensor, has received significant attention in robotics, leading to dramatic progress over the past decade and effective market-ready SLAM solutions incorporated in a range of consumer and industrial products. Such solutions have attracted significant

attention in mobile and handheld computing because the incorporation of 3D visual SLAM capabilities into smartphones opens up a wide range of new applications, such as indoor positioning, AR, and mobile gaming. However, despite the deployment and

processing into a front end and a back end (see Figure 1). The front end processes sensor data, extracts feature measurements, performs data association between the features and the map, and computes the camera pose. Additionally, it is responsible for

sensor processing and map optimization, but standard architectures lead to systems that are restricted to a particular SLAM use case and do not provide flexibility, for example, in terms of freely interchanging landmark types and sensor modalities. We identify three types of modularity within a SLAM system: 1) a front-end component, 2) the front end, and 3) the back-end modularity.

Front-end component modularity refers to the ability to interchange specific components within the front end while maintaining the interface with the back end; an example is a visual odometry at the front end with support to switch among the following:

1. camera sensor types (for example, passive stereo, time of flight, and so forth)
2. multiple cameras (mono, stereo, panoramic, and heterogeneous)
3. feature detectors [for example, scale invariant feature transform (SIFT), features from accelerated segment test, and Harris]
4. feature descriptors [for example, SIFT, BRIEF (binary robust independent elementary features), and BRISK (binary robust invariant scalable keypoints)]
5. disparity estimators (for example, block matching CENSUS transform, and so forth)
6. pose estimators (such as perspective-n-point iterative closest point)
7. windowed bundle adjustment algorithms (such as Ceres, sparse bundle adjustment, incremental smoothing and mapping)
8. integration of dense optical flow (for example, enterprise security management)
9. integration or exclusion of an inertial measurement unit (IMU)
10. a magnetometer-based SLAM when the camera sensor doesn't have a view of the world.

Simultaneous localization and mapping is considered an abstract problem, but robust and efficient solutions are critical to enable emerging intelligent mobile devices.

market penetration, SLAM architectures are still nonscalable and application dependent, leading to systems with high processing requirements and high power consumption.

We argue that addressing SLAM architectural issues, which can be intrinsically supported by low-power, multicore, embedded systems, leads to scalable systems, where *scalability* refers to multimodal SLAMs with various sensors operating with common and reduced resources in terms of computation and power consumption, respectively. In the following, we introduce a modular SLAM architecture and provide an example for its application in a visual SLAM system that is flexible regarding the SLAM implementation approach. We demonstrate the effectiveness of the approach through specific use cases.

STANDARD ARCHITECTURE

SLAM systems typically follow a standard architecture that separates

computing global loop-closure constraints whenever sensors revisit previously mapped locations; typically, the estimation of global constraints is handled by a separate (place recognition) thread.

The back end receives the output of the front end as input, which is a set of camera poses and constraints among those poses. The back end constructs a pose graph and optimizes it to compute an optimal estimate of the camera poses. A map of the environment can be derived from this optimized graph by projecting the features from each pose into the reference frame of the corresponding optimized pose. This is a submapping approach, where the 3D structure is refined through a local SLAM optimization calculated by the windowed bundle adjustment.

MODULARITY IN SLAM SYSTEMS

Modularity between the front end and back end achieves decoupling between

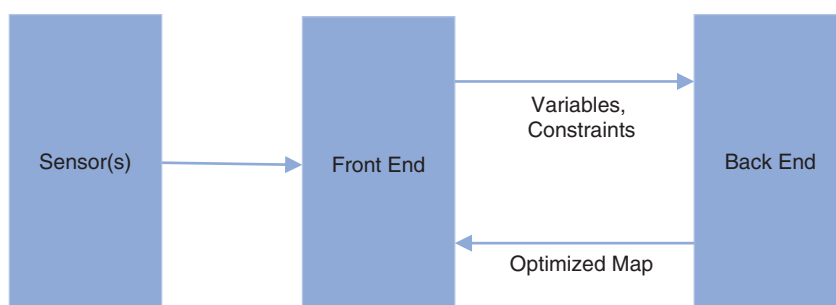


FIGURE 1. The standard SLAM architecture.

Front-end modularity enables switching among different front ends for a given back end. This type of modularity is supported by the Omnimapper framework,² providing greater flexibility than front-end component modularity since it supports switching among sensor modalities, for example, from depth-sensing cameras to line-scan lasers; feature types; and map types, for example, from a 2D laser scan to 3D semantic maps while deploying a single framework across multiple platforms.

Back-end modularity refers to the ability to switch among different back-end optimizers. This type of modularity requires a common abstraction layer between the back and front end(s) to ensure independence of the front end(s) from the back-end implementation.

A MODULAR, MINIMUM VIABLE SLAM

We describe a minimum viable SLAM (MVS) system (see Figure 2) that exploits modularity for scalability, as described previously. The front end consists of two modules: 1) a local mapping and motion estimator and 2) a

loop-closure optimizer. Furthermore, the local mapping and motion estimator is composed of two submodules: 1) tracking and 2) local mapping, which can be run as separate threads. Overall,

Considering a front-end similar to ORB-SLAM,⁴ feature matching is guided by input derived either by a camera motion model or from an IMU. The IMU measurement can be fused with the

Front-end component modularity refers to the ability to interchange specific components within the front end while maintaining the interface with the back end.

the front end is coupled with a back end that provides global map optimization.

Within the tracking and local mapping submodules, each feature is internally represented with parameters such as location and inverse depth, which enable superior performance in visual SLAMs due to its Gaussian error model,³ in contrast to the error model of depth measurements. The front-end/back-end interface needs to abstract the specifics of the related feature detectors and descriptors to allow any feature detection method that satisfies a given set of requirements to be used.

visual odometer pose from the previous frame to robustly predict the reprojection of the map landmarks into the current frame. In the modular SLAM context, the adoption of an IMU should not be treated as a separate input to the pose graph back end since this would result in double counting the IMU measurement and could lead to an overconfident (that is, inconsistent) estimate. Alternatively, IMU measurements can be fused either within the local mapping module—enhancing the 3D map quality—or directly in the back end.

In MVS, the front end maintains a local map and, through that, estimates

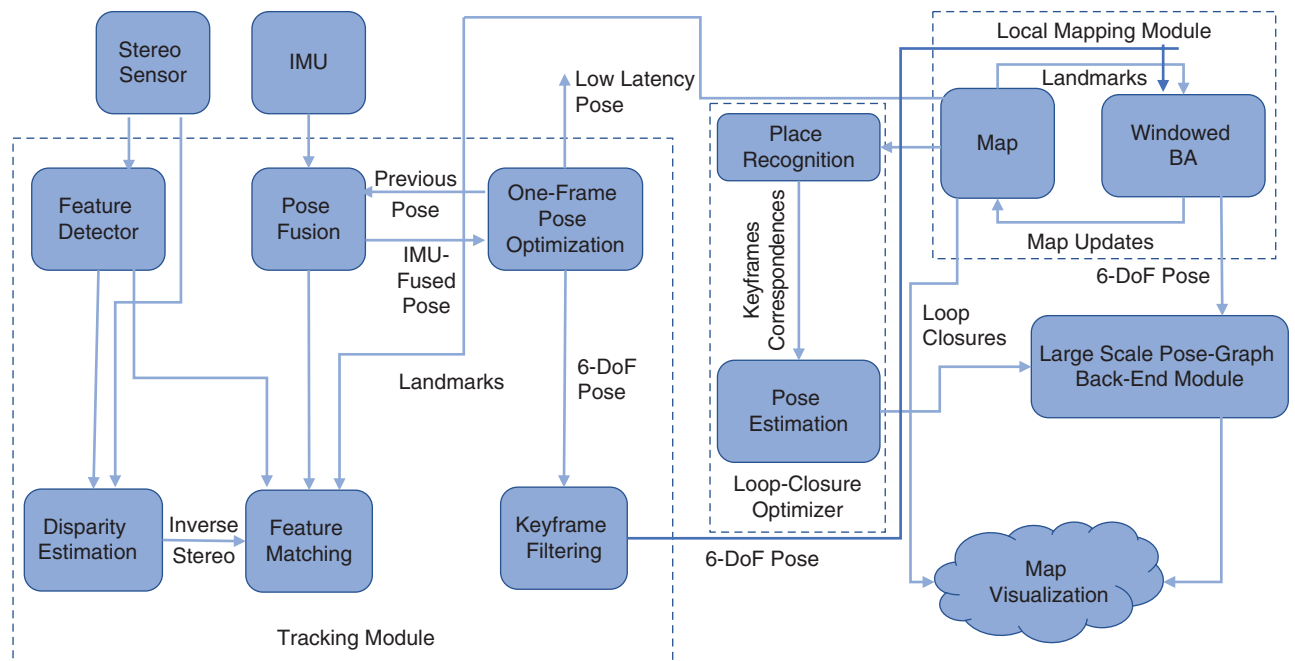


FIGURE 2. A minimum viable visual SLAM. DOF: degrees of freedom.

the motion constraints for the back end. To do this, the front end operates a bundle adjustment (BA) fixed-size window that slides forward as each new frame is input to the system. The BA window optimizes both the poses and landmark locations, taking longer range constraints into account. As each pose leaves the sliding window, a new constraint is passed to the back end. This introduces latency between

USE CASES

Common platforms that could deploy the proposed MVS solution are connected and autonomous vehicles, wheeled platforms equipped with various radar/visual sensors, and visually guided robot arms with end effectors equipped with visual sensors. The inclusion of the MVS solution to unmanned aerial vehicle (UAV) platforms and head-mounted displays puts tough

of the same challenges as UAVs due to their requirement for low-latency pose updates. A key difference here is that UAVs operate under closed-loop control, whereas HMDs do not.


The two primary requirements that UAVs have for visual processing are accurate and continuous low-latency pose estimates and locally accurate mapping for tasks.

the front and back ends. The modular design of the local mapping and motion estimator with its two submodules is effective for this purpose: applications that require a low-latency pose estimates use the output of the odometry module. Furthermore, parameters such as the BA window size and optimization parameters may be tuned to specific application requirements to minimize variation in the computation time of the windowed BA step.

As MVS separates local (small) and large-scale modules, in the back end, the global map is composed of the poses from the pose-graph optimizer and landmarks from the local mapping. The latter maintains each landmark's location relative to the frame of reference of the first pose initially observed. An additional optional functionality that can also be utilized at the back end for improving the pose estimation functionalities is the loop-closure optimizer (LCO). The LCO takes key input frames from the local mapping module, where the key frames contain the necessary feature descriptors for the back-end computation, for example, the (visual) bag-of-words representation. Importantly, any feature descriptor can be used for the recognition process considering alternative input feeds from the sensors.

requirements on the latency and robustness of the front end and requires approaches that mitigate against loss of visual tracking by fusing measurements across a heterogeneous set of sensors. In particular, the gold-standard approach, referred to as visual-inertial navigation, tightly couples the visual tracker with an onboard IMU. This is nowadays an open research direction in the field of robotics and extended reality with many impressive systems demonstrated over the past few years.

UAV platforms and head-mounted displays (HMDs) could also be used with the MVS solution. UAVs present a number of specific challenges due to their high level of agility and their low power requirements. Typical platforms include a variety of sensors, such as IMUs (accelerometer and gyro), downward-pointing sonar for active ranging and altitude control, pressure sensors for altitude control, a GPS for global position updates, and of course potentially multiple cameras. The two primary requirements that UAVs have for visual processing are accurate and continuous low-latency pose estimates and locally accurate mapping for tasks such as path planning, obstacle avoidance, and safe landing-zone identification. Although targeted at a different application domain, HMDs share many

SLAMs are increasingly important to mobile products that automatically reconstruct, interpret, and utilize 3D geometry of their surroundings; these products open up a wide range of new applications in indoor positioning, AR, mobile gaming, and robotics. The commoditization of visual SLAMs is strongly constrained by current nonmodular and non-scalable designs that are heavily dependent on specific sensors and back-end optimizers. Effective modular architectures, like MVS, achieve high flexibility and scalability, enabling high performance and low power consumption. Importantly, the proposed MVS architecture models camera projections, motions, and environments assumes multiview geometry and photometric consistency and thus enables the development of deep-learning SLAMs⁵ that employ powerful deep learning techniques in visual tasks. Visual SLAMs with learning or adaptive capabilities are quite promising since they were recently shown to successfully address such issues as imperfect sensor measurements, inaccurate system modeling, featureless areas, dynamic lighting, motion blurring and camera calibration. 

REFERENCES

1. H. Durrant-Whyte and T. Bailey, "Simultaneous localization and mapping: Part I," *IEEE Robot. Autom.*, vol. 13, no. 2, pp. 99–110, 2006. doi: 10.1109/MRA.2006.1638022.
2. T. Schöps, J. Engel, and D. Cremers, "Semi-dense visual odometry for AR on a smartphone," in *Proc. IEEE Int. Symp. Mixed Augmented Reality (ISMAR)*, 2014, pp. 145–150.
3. J. M. Montiel, J. Civera, and A. J. Davison, "Unified inverse depth parametrization for monocular SLAM," in *Proc. Robot. Sci. Syst.*, 2006, pp. 1–8.

4. R. Mur-Artal, J. M. M. Montiel, and J. D. Tardós, "Orb-SLAM: A versatile and accurate monocular SLAM system," *IEEE Trans. Robot.*, vol. 31, no. 5,

pp. 1147–1163, 2015. doi: 10.1109/TRO.2015.2463671.

5. R. Li, S. Wang, and D. Gu, "Ongoing evolution of visual SLAM from

geometry to deep learning: Challenges and opportunities," *Cogn. Comput.*, vol. 10, no. 6, pp. 875–889, 2018. doi: 10.1007/s12559-018-9591-8.

PETROS KAPSALAS is an advanced driver-assistance systems technical leader at Panasonic Automotive Systems Europe GmbH Langen, Hessen, 63225, Germany. Contact him at Petros.Kapsalas@eu.panasonic.com.

ARIS S. LALOS is a principal researcher at Industrial Systems Institute/ATHENA,

Platani, Patras, 26504, Greece. He is a Member of IEEE. Contact him at lalos@isi.gr.

DIMITRIOS SERPANOS is the director of Industrial Systems Institute/ATHENA and a professor at the University of Patras, Rio, Patras, 26504, Greece. He is a Member of IEEE, Association for Computing Machinery, American Association for the

Advancement of Science, and New York Academy of Sciences. Contact him at serpanos@computer.org.

KONSTANTINOS MOUSTAKAS is an associate professor at the University of Patras, Rio, Patras, 26504, Greece. He is a Member of IEEE. Contact him at moustakas@ece.upatras.gr.

Call for Articles

IEEE Pervasive Computing

seeks accessible, useful papers on the latest peer-reviewed developments in pervasive, mobile, and ubiquitous computing. Topics include hardware technology, software infrastructure, real-world sensing and interaction, human-computer interaction, and systems considerations, including deployment, scalability, security, and privacy.

Author guidelines:
www.computer.org/mcl/pervasive/author.htm

Further details:
pervasive@computer.org
www.computer.org/pervasive

IEEE pervasive COMPUTING
MOBILE AND UBIQUITOUS SYSTEMS