Good Feature Matching: Towards Accurate, Robust VO/VSLAM with Low Latency

Yipu Zhao, and Patricio A. Vela, *Member, IEEE*

***Abstract*—Analysis of state-of-the-art VO/VSLAM system exposes a gap in balancing performance (accuracy & robustness) and efficiency (latency). Feature-based systems exhibit good performance, yet have higher latency due to explicit data association; direct & semidirect systems have lower latency, but are inapplicable in some target scenarios or exhibit lower accuracy than feature-based ones. This paper aims to fill the performance-efficiency gap with an enhancement applied to feature-based VSLAM. We present good feature matching, an active map-to-frame feature matching method. Feature matching effort is tied to submatrix selection, which has combinatorial time complexity and requires choosing a scoring metric. Via simulation, the *Max-logDet* matrix revealing metric is shown to perform best. For real-time applicability, the combination of deterministic selection and randomized acceleration is studied. The proposed algorithm is integrated into monocular & stereo feature-based VSLAM systems. Extensive evaluations on multiple benchmarks and compute hardware quantify the latency reduction and the accuracy & robustness preservation.**

***Index Terms*—visual odometry (VO), visual simultaneous localization and mapping (VSLAM), feature selection, active matching**

I. Introduction

Pose tracking with vision sensors has application to Robotics and Augmented Reality (AR). Research over the past two decades has revealed a few key strategies for visual odometry (VO) and visual simultaneous localization and mapping (VSLAM). Efforts have focused on the accuracy and robustness of pose tracking [1]–[6] and mapping [7], [8], while meeting the real-time requirement (e.g. 30 fps) on desktops & laptops. However, the compute resources on practical robotics and AR platforms is more diverse, and somtimes more limiting. When targeting diverse platforms, VO/VSLAM should be accurate and robust while exhibiting low-latency, i.e., the time cost from capturing an image to estimating the corresponding pose should be low.

Dedicated hardware improves the runtime of VO/VSLAM on compute-constrained platforms. FPGA-based image processing speeds up feature extraction [9], [10], which is a dominant computation for feature-based methods (see Fig. 1, right). Exploring the co-design space between VO (with inertial) algorithm and hardware illuminates parametric settings that improve VO output [11]. Building more efficient VO/VSLAM algorithms, in parallel with better hardware integration, is important to realizing the goal of accurate, low-latency VSLAM.

Y. Zhao and P. A. Vela are with the School of Electrical and Computer Engineering, and Institute of Robotics and Intelligent Machines, Georgia Institute of Technology, Atlanta,GA, 30332 USA. e-mails: (yipu.zhao@gatech.edu, pvela@gatech.edu). This research was partially funded by the National Science Foundation (Award #1816138).

The focus of this paper is on algorithm design aspects of modern VSLAM. As an alternative sensing approach, low-latency visual sensors such as event camera have also been studied for VO/VSLAM tasks [12]–[14]. Application contexts, however, may require that more traditional visual cameras be used. Frame-based cameras are widely recognized as the primary vision sensor in a generic VO/VSLAM system (and downstream detection/recognition systems). The majority of VO/VSLAM systems are designed for frame-based cameras.

State-of-the-art VO/VSLAM systems on frame-based cameras break down into three groups: feature-based, direct, and semidirect systems. Feature-based VO/VSLAM typically consists of two modules: data association (i.e. feature extraction & matching) and state optimization. Due to robust and repeatable modern point-feature descriptors [15]–[18], feature-based systems (e.g., ORB-SLAM [3], OKVIS [19]) benefit from long-baseline feature matchings, and are accurate and robust in most scenarios with sufficient visual textures. In low-texture scenarios where point features fail, line features may be reliable alternative features for VO/VSLAM [20]–[22]. However, feature-based VO/VSLAM typically has high latency: data association is computationally expensive. Direct VO/VSLAM systems such as [4], [6], omit the explicit data association module and optimize a direct objective defined on the image measurements. In general, the computational load and latency of direct systems are lower than feature-based systems. However, the underlying direct objective is non-convex & non-smooth, and therefore harder to optimize versus the geometric objective used in feature-based systems. Furthermore, immediate recovery from track failure (i.e. re-localization) is a known issue for direct systems. Therefore, direct systems require certain conditions [6], [23], [24] for optimal performance, e.g., global shutter camera with precise calibration, minor or slow changes in lighting conditions, accurate motion prediction or smooth & slow camera motion. These conditions limit the applicability of direct systems to many robotics and AR applications, where VO/VSLAM is expected to operate with noisy sensory input under changing environments, for long duration. In addition, direct measurements rarely persist over long-baselines. For applications with frequent revisits, the percentage of long-baseline associations utilized by direct systems is lower than feature-based ones, impacting the performance of direct VO/VSLAM. Semidirect systems [5] also leverage direct measurements for pose tracking, thereby inheriting the reduced tracking performance property relative to feature-based methods. To summarize, there is a gap in the middle ground between performance (accuracy & robustness) and efficiency (low-latency) for state-

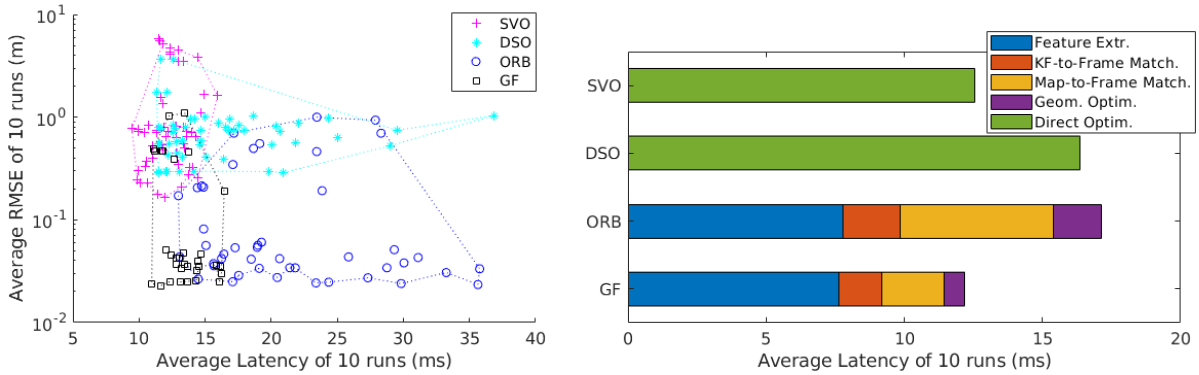


Fig. 1: Latency reduction and accuracy preservation of proposed approach on EuRoC MAV benchmark. Four monocular VO/VSLAM systems are assessed: semidirect SVO [5], direct DSO [6], feature-based ORB [3], and proposed GF-ORB. **Left**: Latency vs. accuracy of 4 systems. The working region of each system (dashed contour) is obtained by adjusting the maximum number of features/patches per frame. **Right**: Latency break down averages per module in pose tracking pipelines on the EuRoC benchmark. A configuration that yields good trade-off of latency and accuracy is set: 800 features/patches extracted per frame; for GF-ORB the good features threshold is 100.

of-the-art VO/VSLAM systems: feature-based systems have good performance yet the latency can be quite high due to the explicit data association; direct & semidirect systems have less latency than feature-based ones, however they are either inapplicable in many target scenarios, or exhibit relatively lower accuracy or robustness.

The objective of this research to balance latency and performance with a computational enhancement applied to feature-based VO/VSLAM. The enhancement reduces the latency to the level of direct systems while preserving the accuracy and robustness of feature-based ones. The key observation that renders the objective achievable being: not all the feature matchings contribute equally to the accurate & robust estimation of camera pose. If there is an efficient approach to identify a small subset of features that are most valuable towards pose estimation, a.k.a. *good features*, then both data association and state optimization should utilize the good features only. Latency of pose tracking will be significantly reduced while preserving accuracy & robustness.

The primary outcome of this work is illustrated in Fig. 1. The left column depicts the latency-accuracy trade-off of 4 monocular VO/VSLAM systems on a public benchmark (EuRoC MAV [25]) by plotting the operational domain of the systems. Each marker on the plot represents a successfully tracked sequence (zero track loss in 10 repeated trials) for the denoted VO/VSLAM system. To better understand the latency-accuracy trade-off in each VO/VSLAM system, we adjust the maximum number of features/patches extracted per frame (for GF-ORB, we also adjust the maximum number of good feature being matched per frame), to obtain the working region of each system in the latency-accuracy plot (in dashed contour). In Fig. 1, feature-based ORB-SLAM occupies the lower-right portion; it is accurate with high-latency. Direct DSO achieves lower latencies under some configurations, but it has an order of magnitude higher *absolute root-mean-square error* (RMSE) than ORB-SLAM; semidirect SVO has a tighter-bounded working region at the upper-left, meaning it is efficient yet inaccurate. The objectives of low-latency and high-accuracy are achieved with the proposed approach, GF-ORB-SLAM, whose markers are located in the lower-left region of the plot. Fig. 1 presents the break down of latency introduced by each module in pose tracking pipelines, under a typical configurations for each VO/VSLAM system. When GF-ORB-SLAM is compared with the baseline ORB-SLAM, the time cost of feature extraction is identical, but the feature matching and subsequent modules' costs are significantly reduced. The latency of GF-ORB is the lowest among all four systems.

This paper extends our previous work on good features [26]. Extensions include an in-depth study of randomized acceleration (Sec. V), and the addition of feature selection with active matching (Sec. VI). Further, the experiments (Sec. VII) are more comprehensive. Contributions of this work include:

1) Study of the **error model** of least squares pose optimization, to connect the performance of pose optimization to the spectral property of a weighted Jacobian matrix;
2) **Exploration of metrics** connected to the least squares conditioning of pose optimization, with quantification of *Max-logDet* as the optimal metric;
3) Introduction of an **efficient good feature selection algorithm** using the *Max-logDet* metric, which is an order of magnitude faster than state-of-the-art feature selection approaches;
4) Fusion of good feature selection and active matching into a **generic good feature matching algorithm**, which is efficient and applicable to feature-based VO/VSLAM; and
5) **Comprehensive evaluation** of the proposed good feature matching on a state-of-the-art feature-based VSLAM system, with multiple benchmarks, sensor setups, and compute platforms. Evaluation results demonstrate both latency reduction and accuracy & robustness preservation with the proposed method. Both monocular[1] and stereo[2] SLAM implementations are open-sourced.

II. Related Work

This work is closely connected to following three research topics in robotics and computer vision: feature selection,

[1]https://github.com/ivalab/GF_ORB_SLAM
[2]https://github.com/ivalab/gf_orb_slam2

submatrix selection, and active matching. In what follows, we review the literatures in each topic, and discuss the connection between existing works and this paper.

A. Feature Selection

Feature selection has been widely applied in VO/VSLAM for performance and efficiency purposes. Conventionally, fully data-driven methods such as random sample consensus (RANSAC) [1] and joint compatibility branch and bound (JCBB) [27] are used to reject outlier features. The computational efficiency of these methods are improved in extended works [28], [29]. These outlier rejection methods are utilized in many VO/VSLAM systems [1]–[3] to improve the robustness of state estimation.

Apart from outlier rejection, feature selection methods are also been utilized for inlier selection, which aims to identify valuable inlier matches from useless ones. One major benefit of inlier selection is the reduction of computation (and latency thereafter), since only a small set of selected inliers are processed by VO/VSLAM. In addition, it is possible to improve accuracy with inlier selection, as demonstrated in [26], [30]–[32]. The scope of this paper is on inlier selection, which reduces the latency of VO/VSLAM while preserving the accuracy and robustness.

Image appearance has been commonly used to guide inlier selection: feature points with distinct color/texture patterns are more likely to get matched correctly [33]–[35]. However, these works solely rely on quantifying distinct appearance, while the structural information of the 3D world and the camera motion are ignored. Appearance cues are important in feature selection, however, the focus of this paper is on the latter properties: identifying valuable features based on structural and motion information. The proposed structure-driven method can combine with a complementary appearance-based approach.

To exploit the structural and motion information, covariance-based inlier selection methods are studied [1], [36]–[40]. Most of these works are based on pose covariance matrix, which has two key characteristics: 1) it contains both structural and motion information implicitly, and 2) it approximately represents the uncertainty ellipsoid of pose estimation. Based on the pose covariance matrix, different metrics were introduced to guide the inlier selection, such as information gain [1], entropy [37], trace [38], covariance ratio [39], minimum eigenvalue and log determinant [40]. Covariance-based inlier selection methods are studied for both filtering-based VO/VSLAM [1], [36]–[39] and BA-based VO/VSLAM [40]–[42].

The observability matrix has been studied as an alternative to the covariance matrix for guiding feature selection [31], [32]. In these works, the connection between pose tracking accuracy and observability conditioning of SLAM as a dynamic system is studied. The insight of their work being: the better conditioned the SLAM system is, the more tolerant the pose estimator will be towards feature measurement error. To that end, the minimum singular value of the observability matrix is used as a metric to guide feature selection. However, observability matrix can only be constructed efficiently under piecewise linear assumption, which limits the applicability of observability-based feature selection. Furthermore, we argue that covariance matrix is better suited for the static or *instantaneous* bundle adjustment (BA) problem as formulated in pose tracking, as it can be constructed efficiently for non-linear optimizers under a sparsity assumption.

The study in [40] is most related to our work. In [40], feature selection is performed by maximizing the information gain of pose estimation within a prediction horizon. Two feature selection metrics were evaluated, minimal eigenvalue and log determinant (*Max-logDet*). Though the log determinant metric is utilized in our work, the algorithm for approximately selecting the feature subset maximizing *logDet* differs, as well as the matrix whose conditioning is optimized. Compared with [40], our work is more applicable for low-latency pose tracking from two key advantages. First, the lazier-greedy algorithm presented in our paper is efficient. It takes an order of magnitude less time than the lazy-greedy algorithm of [40], yet preserves the optimality bound. Second, we present the combination of efficient feature selection and active feature matching, which reduces the latency of both data association and state optimization. Meanwhile, [40] selects features after data association, therefore leaving the latency of data association unchanged. The experimental results in [40] support these claims: there are occasions when feature selection actually increases the latency of full pipeline, compared with the original all-feature approach.

B. Submatrix Selection

A key perspective of this work is connecting feature selection with submatrix selection under a spectral preservation objective, which has been extensively studied in the fields of computational theory and machine learning [43]–[47]. Submatrix selection is an NP-hard, combinatorial optimization problem. To make submatrix selection more amendable to optimization, one structural property, *submodularity*, has been explored [45]–[47]. If a set function (e.g. matrix-revealing metric in this paper) is submodular and monotone increasing, then the combinatorial optimization of the set function (e.g. submatrix selection in this paper) can be approximated by simple greedy method with approximation guarantee.

Compared with deterministic methods (e.g. classic greedy), randomized submatrix selection has been proven to be a faster alternative with probabilistic performance guarantees [48], [49]. Combining randomized selection with a deterministic method yields fast yet near-optimal submatrix selection, as demonstrated for specific matrix norms [44], [50] and general submodular functions [51], [52]. This paper uses the ideas from these works to design a good feature selection algorithm.

C. Active Matching

Another key perspective of this work is combining feature selection algorithm with active feature matching, which leads to latency-reduction in both data association and state optimization. Active matching refers to the guided feature matching methods that prioritize processing resource (e.g. CPU percentage, latency budget) on a subset of features.

Compared with brute force approaches that treat all features equally, active matching is potentially more efficient, especially under resource constraints.

Active matching has been primarily studied for filter-based VO/VSLAM, with representative works [53]–[55]. Traditional active matching methods require dense covariance matrices (i.e. majority of off-diagonal components are filled), and are less relevant to modern VO/VSLAM systems driven by non-linear sparse optimizers. Furthermore, the algorithms used by these active matching methods were computate-heavy, and provided little benefit when integrated into the real-time pose tracking thread of modern VO/VSLAM system. Therefore, the idea of active matching became less attractive. Quoting [56]: "the problem with this idea (active searching) was that ... too much computation is required to decide *where* to look." In this paper, we demonstrate the worth of revisiting the classic idea of active matching: the proposed good feature matching algorithm is extremely efficient and applicable, based upon specific matrices and selection algorithm tailored for non-linear optimization. To the best of our knowledge, this is the first work to demonstrate the applicability of latency-reduction and accuracy preservation in *real-time pose tracking* with active feature selection. The benefit of active matching is realized because the structure of modern VO/VSLAM methods permits first asking whether it is desirable to actively look, then to determine where. In effect, it decides *when* to look, *how much* to look, and *where* to look.

III. Least Squares Pose Optimization Uncertainty

This section examines pose covariance as a function of measurement and point estimation error, with reference to the least squares pose optimization objective commonly used in feature-based VO/VSLAM. The intent is to identify what matrices influence the pose covariance. Without loss of generality, write the least squares objective as,

$$\min \|h(x,p) - z\|^2, \tag{1}$$

where x is the pose of the camera, p are the 3D feature points and z are the corresponding 2D image measurements. The measurement function, $h(x,p)$, is a combination of the $SE(3)$ transformation (world-to-camera) and pin-hole projection. For simplification, we omit camera lens distortion in $h(x,p)$. Correcting for lens distortion involves undistorting the image measurements, z, based on the camera calibration parameters so that the model given by $h(x,p)$ is valid. We base the theory of good feature selection upon the objective of (1).

Solving the least squares objective often involves the first-order approximation to the non-linear measurement function $h(x,p)$:

$$\|h(x,p) - z\|^2 \approx \left\| h(x^{(s)},p) + H_x(x - x^{(s)}) - z \right\|^2, \tag{2}$$

where H_x is the measurement Jacobian linearized about the initial guess $x^{(s)}$. To minimize of the first-order approximation Eq (2) via Gauss-Newton, the pose estimate is iteratively updated via

$$x^{(s+1)} = x^{(s)} + H_x^+(z - h(x^{(s)},p)), \tag{3}$$

where H_x^+ is the left pseudoinverse of H_x.

The accuracy of Gauss-Newton depends on the residual error ϵ_r, which can be decomposed into two terms: measurement error ϵ_z and map error ϵ_p. Using the first-order approximation of $h(x,p)$ at the estimated pose $x^{(s)}$ and map point p to connect the pose optimization error with measurement and map errors leads to

$$\epsilon_x = H_x^+ \epsilon_r = H_x^+(\epsilon_z - H_p \epsilon_p). \tag{4}$$

The Jacobian of map-to-image projection, H_p, is a diagonal matrix with n diagonal blocks $H_p(i)$, where n is the number of matched features.

The Bundle Adjustment literature commonly assumes that the measurement error follows an independent and identically distributed Gaussian (i.e., there is an i.i.d. assumption). While keeping the independent Gaussian assumption, this paper relaxes the identical assumption. Instead, the distribution of measurement error is correlated with the image processing parameters, e.g. the scale-level of the extracted keypoint. Without loss of generality, the measurement error and the map error are modeled as $\epsilon_z(i) \sim N(0, \Sigma_z(i))$ and $\epsilon_p(i) \sim N(0, \Sigma_p(i))$. The combined residual error on image plane follows $\epsilon_r(i) \sim N(0, \Sigma_r(i))$, where

$$\Sigma_r(i) = \Sigma_z(i) + H_p(i)\Sigma_p(i)H_p(i)^T. \tag{5}$$

Applying a Cholesky decomposition to each 2×2 covariance matrix $\Sigma_r(i)$ leads to $\Sigma_r(i) = W_r(i)W_r(i)^T$. Assembling $W_r(i)$ from all n residual terms into a $2n \times 2n$ block diagonal weight matrix W_r and linking to the pose covariance matrix,

$$\Sigma_x = H_x^+ \Sigma_r (H_x^+)^T = H_x^+ W_r (H_x^+ W_r)^T. \tag{6}$$

We aim to simplify the right hand side. Moving both matrices on the right hand side of Eq (6) to the left hand side,

$$W_r^{-1} H_x \Sigma_x (W_r^{-1} H_x)^T = \mathbf{I}. \tag{7}$$

Note that W_r^{-1} is still a block diagonal matrix, consisting of 2×2 blocks denoted by $W_r^{-1}(i)$. Meanwhile, each row block of measurement Jacobian H_x can be written as $H_x(i)$. Following through on the block-wise multiplication results in the matrix H_c:

$$H_c = \begin{bmatrix} W_r^{-1}(0)H_x(0) \\ \dots \\ W_r^{-1}(n-1)H_x(n-1) \end{bmatrix}, \tag{8}$$

from which the simplified pose covariance matrix follows:

$$\Sigma_x = H_c^+ (H_c^+)^T = (H_c^T H_c)^{-1}, \tag{9}$$

assuming that there are sufficient tracked map points so that H_c is full rank. The conditioning of H_c determines the error propagation properties of the iteratively solved least-squares solution for the camera pose x.

IV. Good Feature Selection Using Max-LogDet

The pose covariance matrix Σ_x represents the uncertainty ellipsoid in pose configuration space. According to Eq (9), one should use all the features/measurements available to minimize the uncertainty (i.e. variance) of pose estimation: with more

measurements, the singular values of H_c should increase in magnitude. The worst case uncertainty would be proportional to the inverse of minimal singular value $\sigma_{min}(H_c)$, whereas in the best case it would be proportional to the inverse of maximal singular value $\sigma_{max}(H_c)$.

However, for the purpose of low-latency pose tracking, one should only utilize **sufficient** features. There is a tension between latency and error rejection. From the analysis, the uncertainty of least squares pose optimization problem is bounded by the extremal spectral properties of the matrix H_c. Hence, one possible metric to measure the sufficiency of a feature subset would be the factor of the worst case scenario $\sigma_{min}(H_c)$. Meanwhile, one may argue that the extremal spectral properties only decides the upper and lower bounds of pose optimization uncertainty. The true values would depend on what the overall spectral properties of the system are. It follows then, that another possible measurement of sufficiency would be the overall spectral properties of H_c.

Define the *good feature selection* problem to be: Given a set of 2D-3D feature matchings, find a constant-cardinality subset from them, such that the error of least squares pose optimization is minimized when using the subset only. Based on the previous discussion, the good feature selection problem is equivalent to submatrix selection: Given a matrix H_c, select a subset of row blocks so that the overall spectral properties of the selected submatrix are preserved as much as possible.

A Note Regarding Good Feature Selection & Matching: Good feature selection is slightly different from the final goal of this work, good feature matching. In good feature selection, all 2D-3D feature matchings are assumed to be available in the first place. In good feature matching, only the 3D features are known in the beginning, while the 2D-3D matchings are partially revealed during the guided matching process. Still, these two problems share the same core, which is how to prioritize a subset of features over the others for accuracy-preserving purposes. The section following this one will describe how to translate a good feature selection solution to a good feature matching solution.

A. Submodularity in Submatrix Selection

Submatrix selection with spectral preservation has been extensively studied in the numerical methods and machine learning fields [43], [44], for which several matrix-revealing metrics exist to score the subset selection process. They are listed in Table I. Subset selection with any of the listed matrix-revealing metrics is equivalent to a finite combinatorial optimization problem with a cardinality constraint:

$$\max_{S\subseteq\{1,2,\ldots,n\},|S|=k} f([H_c(S)]^T[H_c(S)]) \tag{10}$$

where S contains the index subsets of selected row blocks from the full matrix H_c, $[H_c(S)]$ is the corresponding row-wise concatenated submatrix, k is the cardinality of subset, and f the matrix-revealing metric.

While the combinatorial optimization can be solved by brute force, the exponentially-growing problem space quickly becomes impractical to search over for real-time VO/VSLAM applications. To employ more efficient subset selection strategies while limiting the loss in optimality, one structural property of the problem may be exploited, submodularity [40], [45]–[47]. If a set function (e.g. matrix-revealing metric) is submodular and monotone increasing, then the combinatorial optimization of the set function (e.g. subset selection) found via greedy methods has known approximation guarantees.

Max-Trace	Trace $Tr(Q) = \sum_1^m Q_{ii}$ is max.
Min-Cond	Condition $\kappa(Q) = \lambda_1(Q)/\lambda_m(Q)$ is min.
Max-MinEigenValue	Min. eigenvalue $\lambda_m(Q)$ is max.
Max-logDet	Log. of determinant $\log\det(Q)$ is max.

TABLE I: Commonly used matrix-revealing metrics, with input square matrix Q of rank m.

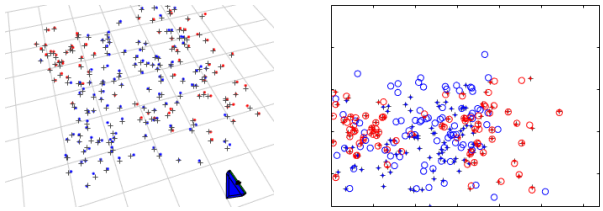


Fig. 2: Simulated pose optimization scenario. **Left**: map view. **Right**: camera view. Selected features are in red, while unselected ones are in blue.

Except for *Min-Cond*, the metrics listed in Table I are either submodular or approximately submodular, and monotone increasing. The *Max-logDet* metric is submodular [45], while the *Max-Trace* is modular (a stronger property) [47]. Lastly, *Max-MinEigenValue* is approximately submodular [46]. Therefore, selecting row blocks (as well as the corresponding features) with these metrics can be approximately solved with greedy methods. Using these known properties, the aim here is to arrive at an efficient algorithm for performing good feature selection or matching without significant loss in optimality.

B. Simulation of Good Feature Selection

To explore which matrix-revealing metrics might best guide good feature/row block selection for least squares pose optimization, this section evaluates the candidate metrics via simulation. The Matlab simulation environment [57], which assumes perfect data association, provides the testing framework. The evaluation scenario is illustrated in Fig. 2. The camera/robot is spawned at the origin of the world frame, and a fixed number of 3D feature points are randomly generated in front of the camera (200 in this simulation). After applying a small random pose transform to the robot/camera, the 2D projections of feature points are measured and perfectly matched with known 3D feature points. Then a Gauss-Newton optimizer uses the matchings to estimate the pose transform.

To simulate the residual error, both the 3D mapped features and the 2D measurements are perturbed with noise. A zero-mean Gaussian with the standard deviation of 0.02m are added to the 3D features stored as map. Three levels of measurement error are added to 2D measurements: zero-mean Gaussian with standard deviation of 0.5, 1.5 and 2.5 pixel. Subset sizes ranging from 80 to 200 are tested. To be statistically sound, 300 runs are repeated for each configuration.

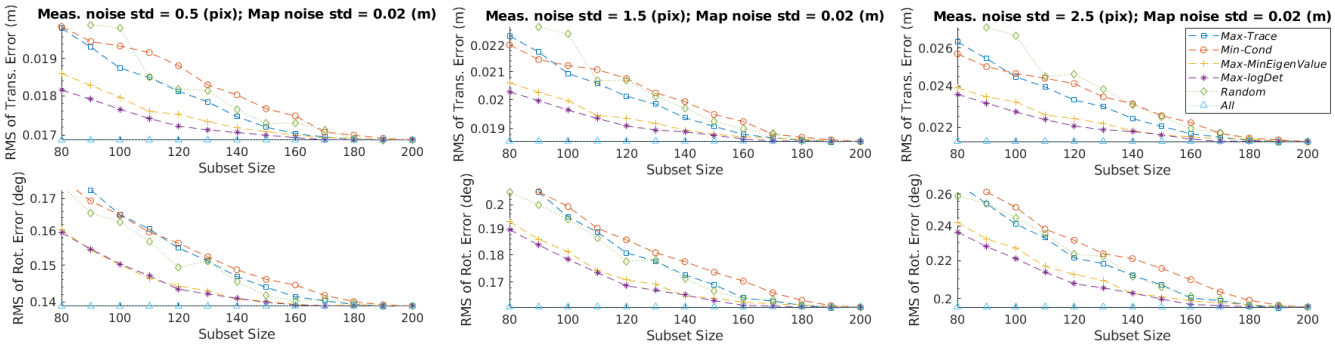


Fig. 3: Simulation results of least squares pose optimization. First row: RMS of translational error under 3 levels of residual error. Second row: RMS of rotational error under 3 levels of residual error.

A feature selection module is inserted prior to Gauss-Newton pose optimization, so that only a subset of selected features is sent into the optimizer. Feature selection is done in two steps: 1) compute the combined matrix H_c from measurement Jacobian H_x and noise weighting matrix W_r, 2) greedy selection of row block $H_c(i)$ based on the matrix-revealing metric, until reaching the chosen subset size. The simulation results are presented in Fig. 3, with the root-mean-square (RMS) of translational error (m) in the first row and rotational error (deg) in the second row. Each of the matrix-revealing metrics in Table I is tested. For reference, the plots include simulation results with randomized subset selection (*Random*) and with all features available (*All*).

From Fig. 3, two metrics stand out: *Max-MinEigenValue* and *Max-logDet*. Under all residual noise levels, their curves more quickly approach the baseline error (*All*) as a function of the subset size. Based on the outcomes, *Max-logDet* is chosen as the metric to guide good feature selection. The reasons being: (1) According to Fig. 3, the error curves of *Max-logDet* are always lower, if not at the same level, than those of *Max-MinEigenValue*. The subset selected with *Max-logDet* approximates the original full feature set better than the subset with *Max-MinEigenValue*. As discussed previously, greedy selection with *Max-logDet* has guaranteed approximation ratio due to submodularity. (2) The computational cost of *logDet* is lower than that of *MinEigenValue*. The main *logDet* computation is Cholesky factorization, with a complexity of $\mathcal{O}(0.33n^3)$, whereas for *MinEigenValue* the complexity is $\mathcal{O}(22n^3)$ [58]. Lastly, the error rate of *Random* selection is much higher than *logDet*-guided selection. To be effective randomized selection requires a larger subset size.

V. Efficient Good Feature Selection

Subset selection with *Max-logDet* metric has been studied for sensor selection [45] and feature selection [40], with reliance on a simple greedy algorithm commonly used to approximate the original NP-hard combinational optimization problem. Since *Max-logDet* is submodular and monotone increasing, the approximation ratio of a greedy approach is proven to be $1 - 1/e$ [47]. This approximation ratio is the best achievable by any polynomial time algorithm under the assumption that $P \neq NP$.

The classical greedy algorithm can be enhanced into an accelerated version, lazy greedy [59]. Instead of evaluating the actual margin gain of the chosen metric (e.g. *logDet*) at each iteration, a computationally-cheap, approximate upper bound is estimated and utilized to reject unwanted blocks/features. Speed enhancement of the lazy greedy hinges on the tightness of the upper bound. Consider an idealized case, where computing the upper bound takes zero-cost and a constant rejection ratio ρ is achieved with the upper bound. Hence the total complexity of selecting k features given n candidates using lazy greedy algorithm is $\mathcal{O}(k(1-\rho)n)$: the lazy greedy algorithm has to run k rounds, in each round it will go through $(1-\rho)n$ candidates to identify the current best feature.

Unfortunately, the commonly used upper bound of *logDet*, as derived from Hadamard's inequality [60], is quite loose [40] (i.e. $\rho \approx 0$):

$$\log\det(Q) \leq \sum_{i=1}^{m} \log(Q_{ii}),\ \operatorname{rank}(Q) = m. \qquad (11)$$

Therefore, *Max-logDet* feature selection does not appreciably benefit from a lazy greedy implementation. As reported in [40] and further confirmed in the simulation to be discussed shortly, the time cost of lazy greedy selection exceeds the real-time requirement (e.g. 30ms per frame), therefore lazy-greedy with *logDet* and *Trace* is impractical for good feature selection in real-time VO/VSLAM applications.

A. Lazier-than-lazy Greedy

To speed up the greedy feature selection, we explore the combination of deterministic selection (e.g. lazy greedy algorithm) and randomized acceleration (e.g. random sampling). One well-recognized method of combining these two, is lazier-than-lazy greedy [51] (referred as lazier greedy in the following). The idea of lazier greedy is simple: at each round of greedy selection, instead of going through all n candidates, only a random subset of candidates are evaluated to identify the current best feature. Furthermore, the size of the random subset s can be controlled by a decay factor ϵ: $s = \frac{n}{k}\log(\frac{1}{\epsilon})$. In this way, the total complexity is reduced from $\mathcal{O}(k(1-\rho)n)$ to $\mathcal{O}(\log(\frac{1}{\epsilon})n)$. Importantly, lazier greedy is near-optimal:

Theorem 1: [51] Let f be a non-negative monotone submoduar function. Let us also set $s = \frac{n}{k}\log(\frac{1}{\epsilon})$. Then lazier greedy achieves a $(1 - 1/e - \epsilon)$ approximation guarantee in expectation to the optimum solution of problem in Eq 10.

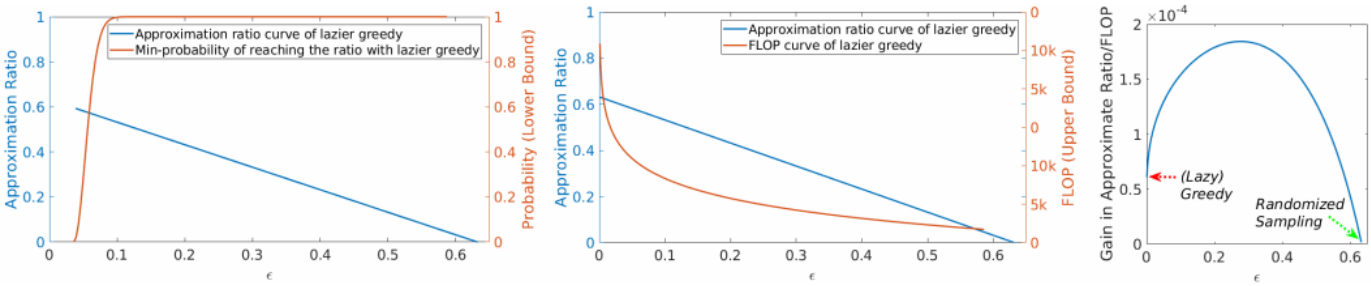


Fig. 4: Illustration of performance & efficiency of lazier greedy, when selecting 450 subset from 1500 rows with average approximation ratio $\mu = 0.8$ in maximizing margin gain (*logDet*). **Left**: Approximation ratio & probabilistic guarantee of lazier greedy, w.r.t. different decay factor ϵ. **Middle**: Approximation ratio & computation cost (FLOP) of lazier greedy, w.r.t. different decay factor ϵ. **Right**: Efficiency of lazier greedy, w.r.t. different decay factor ϵ.

Theorem 2: [52] The expectation of approximation guarantee of $(1 - 1/e - \epsilon)$ is reached with a minimum probability of $1 - e(-0.5k(\sqrt{\mu} + ln(\epsilon + e^{-1})/\sqrt{\mu})^2)$, when maximizing a monotone submodular function under cardinality constraint k with lazier-greedy. $\mu \in (0, 1]$ is the average approximation ratio in maximizing margin gain at each iteration of lazier greedy.

The symbols & formulations in Theorem 2 are adjusted from the original ones [52] to be consistent with Theorem 1. According to these two theorems: 1) lazier greedy introduce a linear loss ϵ to the approximation ratio *in expectation*; and 2) the expectation of linear-loss approximation ratio can be guaranteed with high probability, as illustrated in Fig. 4 (left). Compared to the theoretical upper bound of approximation ratio, $1-1/e$, which no polynomial time algorithm can exceed [47], lazier greedy only loses a small chunk from it (in expectation & in probability).

The approximation ratio and computational speed up of lazier greedy hinge on the decay factor ϵ. When the decay factor $\epsilon = 0$, the lazier greedy algorithm is the greedy algorithm. Meanwhile, when the decay factor is $e^{-\frac{k}{n}}$, lazier greedy becomes randomized sampling (i.e. $s = 1$), which has an approximation ratio of $1 - 1/e - e^{-\frac{k}{n}}$ in expectation. As illustrated in Fig. 4 (middle), the approximation ratio decays linearly with ϵ, while the computational cost (FLOP) decays logarithmically. As ϵ increases the resulting computational gain outpaces the loss in optimality, until hitting an inflection point after which the benefit reduces. By setting ϵ to a small positive value, e.g. 0.1-0.5 as indicated in Fig. 4 (right), lazier greedy will have a slightly degraded optimal bound but with a 3-4x higher efficiency than lazy greedy. Alg 1 describes an efficient algorithm for good feature selection based on the near-optimal lazier-greedy.

Algorithm 1: Lazier-greedy good feature selection algorithm.

Data: $H_c = \{H_c(1),\ H_c(2),\ \dots,\ H_c(n)\},\ k$
Result: $H_c^{sub} \subseteq H_c,\ |H_c^{sub}| = k$
$H_c^{sub} \leftarrow \emptyset$;
while $|H_c^{sub}| < k$ **do**
  $H_c^R \leftarrow$ a random subset obtained by sampling $s = \frac{n}{k}\log(\frac{1}{\epsilon})$ random elements from H_c;
  $H_c(i) \leftarrow \arg\max_{H_c(i) \in H_c^R} \log\det(H_c(i)^T H_c(i) + [H_c^{sub}]^T [H_c^{sub}])$;
  $H_c^{sub} \leftarrow H_c^{sub} \cup H_c(i)$;
  $H_c \leftarrow H_c \setminus H_c(i)$;
return H_c^{sub}.

B. Simulation of Lazier Greedy Feature Selection

To validate the benefits of lazier greedy, and to identify the proper value of decay factor ϵ, a simulation of good feature selection is conducted. A testing process similar to the Matlab one from previous pose optimization simulation was implemented C++ for speed assessment. The two feature selection algorithms tested are: lazy greedy [40] and lazier greedy (Alg 1). Like the simulation of pose optimization, a set of randomly-spawned 3D feature points, as well as the corresponding 2D measurements, are provided as input. Gaussian noise is added to both the 3D mapped features and the 2D measurements. The perturbed inputs are fed into a matrix building module, which estimates the combined matrix H_c for submatrix/feature selection.

To assess the performance and efficiency of good feature selection comprehensively, we sweep through the three parameters: the size of 3D feature set from 500 to 2500, the size of desired feature subset from 40 to 180, and the decay factor from 0.9 to 0.005. For each parameter combination, we randomly spawn 100 different worlds and evaluate each feature selection algorithm on each world. Due to the randomness of lazier greedy, we repeat it 20 times under each configuration.

Fig. 5 plots the simulation results for computational time and error ratio as a function of sizes of the desired subset and the full set. The error ratio uses the lazy-greedy outcome as the baseline, then computes the normalized RMS of the difference versus lazier greedy. The multiple surfaces for lazier-greedy correspond to different decay factors ϵ. Referring to the time cost graph, lazier greedy is 1-2 orders of magnitude lower than greedy, depending on ϵ. The plot includes a constant reference plane of $30ms$ time cost (in blue). The preference is to lie near to-or below-this reference plane, which lazier greedy can achieve over large regions of its parameter space while lazy greedy cannot. Moving to the error ratio graph, an error ratio of 0.01 indicates that the subset selected with lazier greedy is less than 1% different from the lazy greedy baseline. Though slow, lazy greedy performs well for good feature selection. According to Fig. 5, the average error ratio

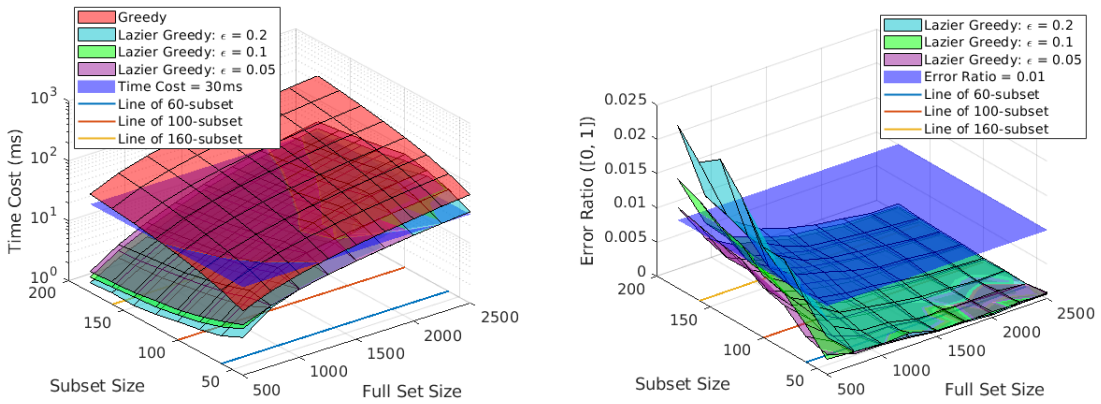


Fig. 5: Lazy greedy vs. lazier greedy in feature selection simulation. **Left**: average time cost of lazy greedy vs. lazier greedy under different decay factor ϵ. **Right**: average error ratio of lazier greedy (compared with lazy greedy baseline; the smaller the better) under different ϵ. Three exemplar working spaces of typical VSLAM problems are plotted (as lines) in both figures, with 60, 100 & 160 feature subset selected for pose tracking.

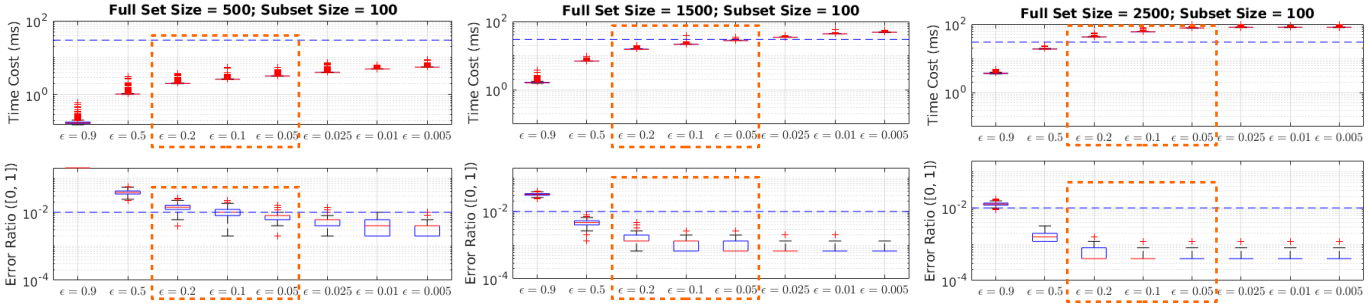


Fig. 6: Lazier greedy with different decay factor ϵ under 3 example configurations: selecting 100 features from full sets of 500, 1500, & 2500. **First row**: time cost of lazier greedy. **Second row**: error ratio of lazier greedy. For cross comparison, figures in both rows are with log-scale y-axis.

of lazier greedy is below 0.01 for the majority of configuration surfaces when $\epsilon \leq 0.1$. The graphs include three lines in the x-y visualization plane corresponding to the target subset sizes used in the VSLAM experiments. Lazier greedy with $\epsilon \leq 0.1$ consistently achieves a low error ratio, yet consumes a fraction of time cost compared to lazy greedy.

To further identify an acceptable decay factor ϵ, box-plots of time cost and error ratio are presented in Fig. 6 under three configurations, which vary the number of matched features. We consider $\epsilon = 0.1$ to be a favorable parameter choice for good feature selection: the lazier greedy time cost is minimized under the requirement of less-than-0.01 error ratio. In what follows, all experiments run lazier greedy with $\epsilon = 0.1$.

VI. Good Feature Matching in VO/VSLAM Pipeline

The prior discussion regarding the connection between tracked features and *Max-logDet* subset selection led to an efficient good feature selection algorithm. Selection is based on the assumption that all 2D-3D feature matchings are known, it only applies after data association (e.g. similar to the existing works [26], [31], [32], [40]). However, as shown in Figure 1, the time cost of data association occupies a significant portion of time (about 1/3) in the real-time pose tracking thread of feature-based VO/VSLAM. To reduce pose tracking time cost, consider translating the problem of good feature selection to data association: Given a set of 3D mapped features and a set of un-matched 2D measurements, only associate a constant-cardinality subset of 2D-3D matchings that will minimize the error of the least squares pose optimization. In what follows, we discuss an efficient solution to the translate problem, referred to as *good feature matching*.

A. Good Feature Matching in Monocular VO/VSLAM

While the good feature selection problem applies to already associated data, the good feature matching problem starts with a pool of 2D feature points and 3D map points whose correct associations are unknown. The aim is to establish which points should be targeted for matching and in what priority. Three modifications are made to transfer the previous solutions to good feature matching, which is described in Alg 2:

(1) Remove the dependency on 2D-3D matchings when constructing matrix H_c (lines 1-4 of Alg 2). Constructing H_c requires knowing the covariance matrix of 2D measurement $\Sigma_z(i)$ (as formulated in Eq (5)). To avoid this information, assume a constant prior (e.g. a 2×2 identity matrix) at the initial stage of good feature matching (line 3 of Alg 2);

(2) Add a feature matching condition check before updating the subsets in good feature selection (line 10 of Alg 2): only when the current best 3D feature (with highest *logDet* margin gain) is successfully matched with some 2D measurement, should the matrix (feature) subset get updated accordingly. For the current best 3D feature, a search of possible 2D

Algorithm 2: Good feature matching in mono VO/VSLAM.

Data: $P = \{p(1),\ p(2),\ \ldots,\ p(n)\}$,
$Z = \{z(1),\ z(2),\ \ldots,\ z(m)\}$, k, t_{max}
Result: $M = < p(i), z(j) >,\ |M| = k$

1 $M \leftarrow \emptyset$, $H_c^{sub} \leftarrow \emptyset$, $\mathcal{I} = \{1, \ldots, n\}$, $t_{accu} = 0$;
2 **foreach** *3D feature* $p(i)$ **do**
3 build Jacobians $H_x(i),\ H_p(i)$;
4 $W(i) = chol(\mathcal{I}_2 + H_p(i)\Sigma_p(i)H_p(i)^T)$;
5 $H_c(i) = W(i)^{-1}H_x(i)$;
6 **while** $|M| < k$ *and* $|\mathcal{I}| > 0$ *and* $(t_{accu} < t_{max})$ **do**
7 $H_c^R \leftarrow$ a random subset obtained by sampling $s = \frac{n}{k}\log(\frac{1}{\epsilon})$ non-repeated random elements from H_c;
8 **while** $|\mathcal{I}| > 0$ *and* $(t_{accu} < t_{max})$ **do**
9 $i \leftarrow \arg\max_{H_c(i) \in H_c^R} \log\det(H_c(i)^T H_c(i) + [H_c^{sub}]^T[H_c^{sub}])$;
10 **if** *found matched measurement* $z(j)$ *for* $p(i)$ **then**
11 $W(i) = chol(\Sigma_z(j) + H_p(i)\Sigma_p(i)H_p(i)^T)$;
12 $H_c(i) = W(i)^{-1}H_x(i)$;
13 $M \leftarrow M \cup < p(i), z(j) >$;
14 break;
15 **else**
16 $H_c^R \leftarrow H_c^R \setminus H_c(i)$;
17 $H_c^R \leftarrow H_c^R \cup$ a random sample from H_c;
18 $\mathcal{I} \leftarrow \mathcal{I} \setminus \{i\}$
19 $H_c^{sub} \leftarrow H_c^{sub} \cup H_c(i)$;
20 $H_c \leftarrow H_c \setminus H_c(i)$;
21 $Z \leftarrow Z \setminus z(j)$;
22 **return** M.

Algorithm 3: Good feature matching in stereo VO/VSLAM.

Data: $P = \{p(1),\ \ldots,\ p(n)\}$, $Z = \{z(1),\ \ldots,\ z(m)\}$,
$Z^r = \{z^r(1),\ \ldots,\ z^r(s)\}$, k
Result: $M = < p(i), z(j), z^r(r) >,\ |M| = k$

// line 1-9 identical with monocular version
10 **if** *found matched left measurement* $z(j)$ *for* $p(i)$ **then**
11 $W(i) = chol(\Sigma_z(j) + H_p(i)\Sigma_p(i)H_p(i)^T)$;
12 $H_c(i) = W(i)^{-1}H_x(i)$;
13 **if** *found matched right measurement* $z^r(d)$ *for* $p(i)$ **then**
14 $W^r(i) = chol(\Sigma_z^r(d) + H_p^r(i)\Sigma_p(i)H_p^r(i)^T)$;
15 $H_c(i) = [H_c(i); W^r(i)^{-1}H_x^r(i)]$;
16 $M \leftarrow M \cup < p(i), z(j), z^r(d) >$;
17 **else**
18 $M \leftarrow M \cup < p(i), z(j), \emptyset >$;
19 break;
// rest of lines identical with line 15-22 of monocular version

measurements is conducted on the image frame, with a size-fixed local search window (centered on the 2D projection of 3D feature). If no 2D measurement can be matched to the current best 3D feature, it moves on to match the next best 3D feature (lines 15-17 of Alg 2). By fusing good feature selection with feature matching, the selection strategy becomes an active matching algorithm: the feature matching effort prioritizes the subset with highest potential gain (in terms of *logDet*).

(3) Information from successful feature matchings assists follow-up good feature choice, by updating the measurement covariance $\Sigma_z(i)$ (and the associated block in $H_c(i)$) with measurement information (line 11 of Alg 2). The measurement covariance $\Sigma_z(i)$ is assumed to be quadratic with the scale/pyramid level of measurement extraction; it is updated once the pyramid level is known from 2D-3D feature matching. The corrected block $H_c(i)$ for the matched feature is concatenated into the selected submatrix, and next iteration of greedy matching starts (lines 18-20 of Alg 2).

A fourth modification is made for SLAM problems.

(4) Rather than exhaustively search the candidate matching pool for k matches, the loops in lines 6 and 8 of Alg. 2 include a time budget t_{max} condition. The time budget is sensible, as the submodularity property is associated with diminishing returns (i.e. the marginal value of the j^{th} match is less than earlier matches). Searching too long forfeits the task of balancing accuracy and latency. In experiments $t_{max} = 15$ms, and is rarely met.

B. Good Feature Matching in Stereo VO/VSLAM

Good feature matching applies to stereo cameras as well as to monocular cameras. Compared to monocular VO/VSLAM pipeline, stereo VO/VSLAM has an additional module in data association: stereo matching, which associates measurements between left and right frames. Since the stereo algorithm associates existing 3D mapped features to 2D measurements from both frames, each paired measurement provides twice the number of rows to the least squares objective (in pose-only and joint BA). Stereo methods also provide for instant initialization of new map points through triangulated 2D measurements from the left and right frames. However, optimization for the current pose (as pursued in pose tracking) only benefits from the stereo matchings associated with existing 3D mapped features! By exploiting this property, we can design a lazy-stereo VO/VSLAM pipeline that has lower latency than the original stereo pipeline. Stereo matching is postponed to after map-to-frame matching. Instead of searching for stereo matchings between all measurements, only those measurements associated with 3D map points are matched. After pose optimization, the remaining measurements are stereo-matched and triangulated as new 3D mapped features.

The lazy-stereo VO/VSLAM pipeline should have the same level of accuracy & robustness as the original pipeline, with reduced pose tracking latency. Implementing the stereo good feature matching algorithm with the lazy-stereo pipeline will further reduce latency while preserving accuracy & robustness. Compared with the monocular Alg 2, the stereo Alg 3 has additional steps of stereo matching at each successful iteration of map-to-frame feature matching (line 13 of Alg 3). Depending

on the matching outcome, the block $H_c(i)$ contains map-to-frame information only (no stereo matching found; line 11-12 of Alg 3), or both map-to-frame and left-to-right information (stereo matching found; line 14-15 of Alg 3).

C. Connection with Conventional Active Matching

Conventionally, active matching is iteratively conducted with 2 major steps: 1) a selecting step that chooses which 3D feature to match against, and 2) a matching step that searches for best 2D measurements within a local area/window of image frame. The local area is typically refined during the active matching process, as more feature matches are found and used to improve the current camera pose estimate. For filter-based VO/VSLAM where the covariance matrix is easily assessable, refining the local search area on-the-fly is possible (by partially updating the covariance matrix during the active matching process). Here we argue that for BA-based VO/VSLAM, refining local search area is not necessary (and not efficient). Instead, working with a fixed-size local searching window is sufficient; it also improves the robustness towards inconsistency and bias in state estimation. Compared with the conventional active matching, good feature matching also selects the best 3D feature to match at each iteration, but the local search area for feature matching remains fixed.

VII. Implementation and Evaluation

This section evaluates the performance of the proposed good feature matching algorithm on a state-of-the-art feature-based visual SLAM system, ORB-SLAM [3]. Applying the proposed algorithms to the real-time tracking thread of ORB-SLAM (Alg 2 for monocular ORB-SLAM[3] & Alg 3 for stereo ORB-SLAM2[4]), reduces the latency of pose tracking. Meanwhile, the tracking accuracy is either improved (on desktop) or the same as canonical ORB-SLAM (on low-power devices), and the robustness is preserved (i.e. avoiding tracking failure).

ORB-SLAM involves two data association steps, keyframe-to-frame and map-to-frame. Of the two, map-to-frame has the higher time cost (see Fig. 1) and will always consist of points with estimated 3D positions. Thus we elect to incorporate good feature matching into that module. Integrating the proposed good feature matching algorithm into the map-to-frame matching function of ORB-SLAM leads to several changes, which provide additional, valuable runtime properties. Since the keyframe-to-frame data association step will result in a set of matches, M_{K2F}, the good feature matching process does not need to identify a full set of n_{GF} good feature matches. Instead it will identify a smaller set of $k = \min(0, n_{GF} - |M_{K2F}|)$ good feature matches. The modification has an additional advantage. Figs. 5 and 6 indicate that the time cost of lazier greedy grows past a given low threshold as the map size grows (e.g., the full set size). Furthermore, the approximation error ratio grows as the subset size grows. By limiting k to a topping off functionality of M_{K2F} that relates to the target cardinality n_{GF}, we are able to move the good feature matching implementation towards the lower subset sizes, therefore bounding the time cost and the error ratio. Under Algs. 2 and 3, the map-to-frame module prioritizes map point association according to the *max-logDet* metric up to the target set size k, rather than attempt to match all visible local map points to the current measurements. This change establishes *when* to trigger active matching and *how much* effort to apply (per the value k). The follow-up pose tracking thread will utilize at most n_{GF} associations, which are faster to collect and to perform pose optimization with versus the original implementation.

Due to the latency-reduction of good feature matching, there is typically extra time between outputting the current pose estimate and receiving next image. House cleaning and anticipatory calculations occur during this time. House cleaning involves searching for additional map-to-frame feature matchings, when the current frame is selected as a keyframe. The additional matches permit the local BA process to still take advantage of the full set of feature matchings. Anticipatory calculations apply to the matrix preparation stage of good feature matching, i.e. line 1-4 of Alg 2. The steps are pre-computed to be immediately available for the next frame. The pre-computation further reduces the latency of good feature matching. The good-feature-matching enhanced ORB-SLAM is referred to as *GF-ORB-SLAM*, and *GF* for short.

For baseline comparison purposes, we integrate two reference methods into ORB-SLAM that modify Algs. 2 and 3 by prioritizing feature matching with simple heuristics. One heuristic is purely-randomized matching, i.e., *Rnd* (no prioritization). The other heuristic prioritizes map points with a long tracking history since they are more likely to be mapped accurately. We refer to the second heuristic method *Long*.

A. Benchmarks

The revised ORB-SLAM with good feature matching is evaluated against available, state-of-the-art VO/VSLAM systems on four public benchmarks:

1) *The EuRoC benchmark* [25], which contains 11 stereo-inertial sequences comprising 19 minutes of video, recorded in 3 different indoor environments. Ground-truth tracks are provided using motion capture systems (Vicon & Leica MS50). We evaluate monocular (e.g. left camera) and stereo versions.

2) *The TUM-VI benchmark* [61], which contains 28 stereo-inertial sequences of indoor and outdoor environments. Only the 6 sequences (i.e. room1-6, in total 14 minutes of video) with complete coverage by MoCap ground truth are selected. Compared with EuRoC, sequences in TUM-VI are recorded under much stronger camera motion, which is hard to track with monocular VO/VSLAM. Only stereo methods are tested.

3) *The TUM-RGBD benchmark* [62], which is recorded with a Microsoft Kinect RGBD camera. Three sequences that are relatively long (i.e. over 80 seconds each) and rich in camera motion are used in the evaluation. The total length of videos selected is 5.5 minutes. Compared with the previous two benchmarks, captured with global shutter cameras, the image quality of TUM-RGBD benchmark is lower, e.g. rolling shutter, motion blur [5]. This benchmark tests monocular

[3]https://github.com/raulmur/ORB_SLAM
[4]https://github.com/raulmur/ORB_SLAM2

VO/VSLAM on low-end sensors and slow motion, whereas the previous two test high-end sensors and fast motion.

4) *The KITTI benchmark* [63], which contains contains 11 stereo sequences recorded from a car in urban and highway environments. In total 40 minutes of video are recorded, with individual recording duration ranging from 30 seconds to 8 minutes. Ground truth tracks are provided by GPS/INS. Unlike the earlier three indoor benchmarks, KITTI is a large-scale outdoor benchmark that characterizes self-driving applications. Stereo VO/VSLAM methods are tested on KITTI.

B. Evaluation Metrics

Since the focus of this work is on real-time pose tracking, all evaluations are performed on the instantaneous output of pose tracking thread; key-frame poses after posterior bundle adjustment are not used. For fair comparison between VSLAM and VO methods, the loop closing modules are disabled in all ORB-SLAM variants. For the first three benchmarks that evaluate small-to-medium scale indoor scenarios, *absolute root-mean-square error* (RMSE) between ground truth track and SLAM estimated track is utilized as the accuracy metric (commonly used in SLAM evaluation [62], [64]–[66]). For the last benchmark (KITTI outdoor), two relative metrics Relative Position Error (RPE) and Relative Orientation Error (ROE) are reported, as recommended [63]. Full evaluation results for both RMSE and RPE/ROE from all benchmarks are provided externally[5]. Performance assessment involves a 10-run repeat for each configuration, i.e., the benchmark sequence, the VO/VSLAM approach and the parameter (number of features tracked per frame). Results are reported if the VO/VSLAM approach works reliably under the configuration; no tracking failure when running on a desktop, or at most 1 failure on a low-power device.

Additional values recorded include the latency of real-time pose tracking per frame, defined as the time span from receiving an image to publishing the state estimate. The latency of image capture and transmission are not included since they are typically lower than that of VO/VSLAM algorithm, and are outside of the scope of this investigation.

This section first evaluates the accuracy-latency trade-off of GF-ORB-SLAM against state-of-the-art monocular VO/VSLAM methods, then evaluates stereo version against stereo VO/VSLAM methods. In the process, we study the parameter-space of the *GF* modification in order to identify the operational domain of any free parameters to fix them at constant values in subsequent experiments. The experiments are conducted on 3 desktops with identical configuration: Intel i7-7700K CPU (passmark score of 2581 per thread), 16 GB RAM, Ubuntu 14.04 and ROS Indigo environment. Finally, this section evaluates monocular GF-ORB-SLAM on low-power devices, suited for light-weight platforms such as micro aerial and small ground vehicles.

C. Latency vs. Accuracy: Mono VO/VSLAM

In addition to the monocular ORB-SLAM baseline (*ORB*), two state-of-the-art monocular direct VO methods serve as

[5]https://github.com/ivalab/FullResults_GoodFeature

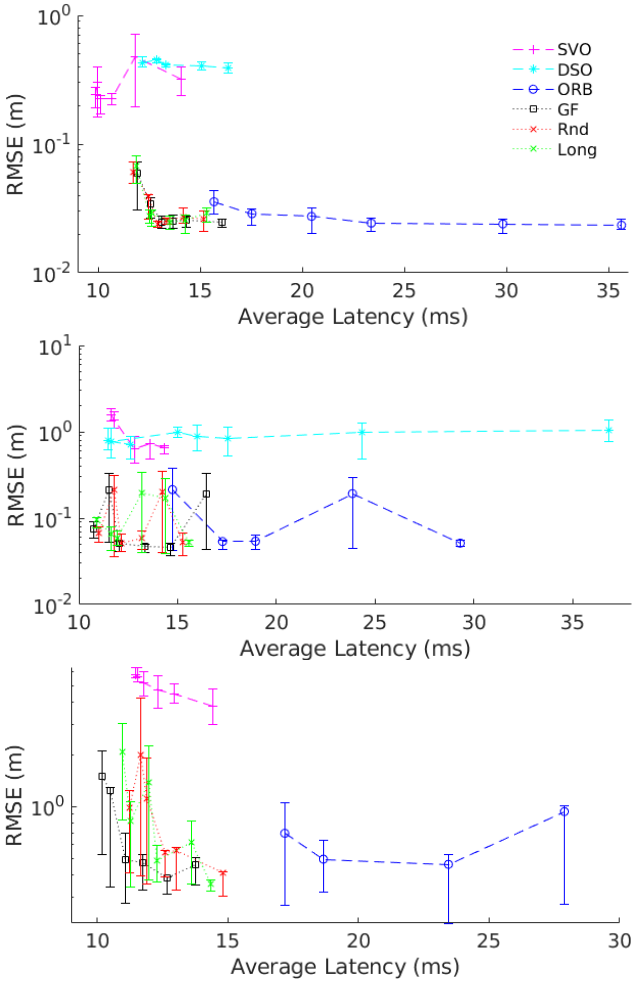


Fig. 7: Latency vs. accuracy on 3 EuRoC Monocular sequences: *MH 01 easy*, *V2 02 med*, and *MH 04 diff* (from top to bottom). Baseline systems are evaluated with *max feature number* ranging from 150 to 2000; ORB-SLAM variants are evaluated with *good feature number* ranging from 60 to 240, and *max feature number* fixed to 800. Only configurations with zero failure in a 10-run repeat are plotted (e.g. all configurations of DSO fail to track on *MH 04 diff*, hence it is omitted in row 3). The same rule applies subsequent latency vs. accuracy figures.

baselines: *SVO*[6] [5] and *DSO*[7] [6]. *SVO* is a light-weight direct VO system targeting low-latency pose tracking while sacrificing tracking accuracy. The multi-threaded option in *SVO* is enabled, so that the depth update/mapping runs on a separate thread from pose tracking. Compared with *SVO*, the direct objective in *DSO* is more comprehensive: it includes an explicit photometric model for image capturing. While *DSO* typically has better tracking performance than *SVO*, the latency of *DSO* can be much higher (up to 3x). Unlike *ORB* and *SVO*, *DSO* has a single-thread implementation only. Its latency varies dramatically between regular frames (e.g. 20 ms) and keyframes (e.g. 150 ms) [6]. A multi-threaded *DSO* would only have the latency of regular frames, as the keyframes

[6]http://rpg.ifi.uzh.ch/svo2.html
[7]https://github.com/JakobEngel/dso

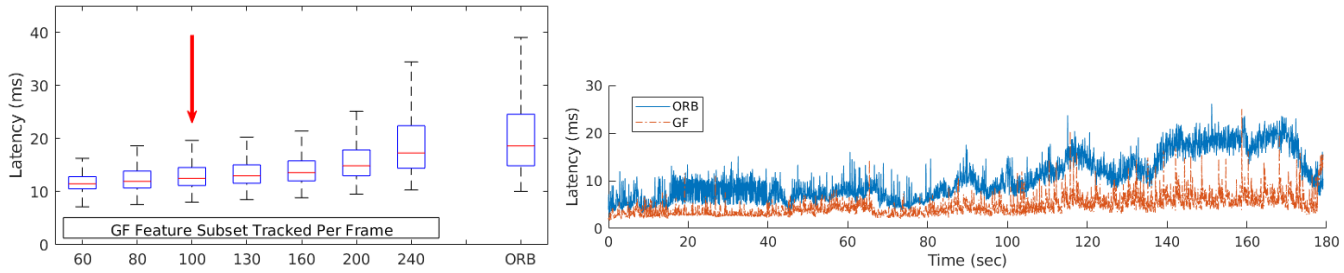


Fig. 8: Latency vs. *good feature number* on EuRoC sequence *MH 01 easy*. **Left**: box-plots for *GF* and baseline *ORB*. **Right**: the latency vs. time trend of *GF* under a *good feature number* of 100 (marked with red arrow on left) and *ORB* for 1 run.

can be processed in a separate thread. Such a multi-threaded *DSO* is expected to be slightly less accurate than the single-thread version, because the joint-optimized map points are no longer immediately accessible for real-time pose tracking. In our evaluation, an idealized multi-threaded *DSO* is assumed: latency is evaluated on regular frames only, while accuracy is evaluated on the single-thread *DSO*.

The latency and accuracy of VO/VSLAM systems can be adjusted through a few internal parameters. One key parameter that significantly impacts both latency and accuracy is the *max feature number*, i.e., the maximum number of features/patches tracked per frame. Running VO/VSLAM with high *max feature number* is beneficial for accuracy & robustness. Meanwhile, lowering the *max feature number* is preferred for latency reduction. To evaluate the trade-off between latency and accuracy for baseline systems (*ORB*, *SVO*, and *DSO*), all of them are configured to run 10-repeats for *max feature number* parameters ranging from 150 to 2000.

For a given *max feature number*, ORB-SLAM latency can be reduced via the proposed good feature matching algorithm. Adjusting the *good feature number*, i.e. the number of good features being matched in pose tracking, changes the latency. Tests with the three ORB-SLAM variants (*GF*, *Rnd* and *Long*) are configured to run 10-repeat under *good feature number* values ranging from 60 to 240. Meanwhile, the *max feature number* is fixed to 800, which yields a good balance of latency and accuracy for baseline *ORB*.

1) Parameter Exploration on EuRoC Monocular: Fig. 7 presents the latency-accuracy trade-off curves for monocular VO/VSLAM implementations on three example EuRoC sequences Amongst the baseline methods, *ORB* has the best accuracy while *SVO* has the lowest latency. Lowering the *max feature number* reduces the latency of *ORB* baseline, however, it comes with loss of tracking accuracy (e.g. the 1st blue marker in row 2), or even the risk of track failure (e.g. the first 2 blue markers are omitted in row 3). Meanwhile, a better latency-accuracy trade-off is achieved with the proposed *GF* method. According to Fig. 7, the latency of *GF* is in a similar range as *SVO*, but with the accuracy of *GF* being an order of magnitude better than both *SVO* and *DSO*. Furthermore, the accuracy-preserving property of *GF* is demonstrated when compared to the reference methods *Rnd* and *Long*. The latency-accuracy curves of *GF* are almost flat and lower than the other two, once a reasonable number of good features are set to be matched (e.g. starting from the 3rd black marker).

The latency-reduction of *GF* is further illustrated in Fig. 8, in which the *max feature number* is set to 800. Compared with *ORB*, the latency of *GF* has lower variance. A good setting for the *good feature number* is 100, marked by a red arrow in Fig. 8. The accuracy of *GF* with a *good feature number* of 100 is on par with *ORB*, as quantified by the 3rd black marker in each row of Fig. 7.

2) EuRoC Monocular: Here, we report the accuracy & latency of all monocular VO/VSLAM methods under fixed configurations: the RMSE values are in Table II (after a *Sim3* alignment to the ground truth), and the latency values in Table III. For the three VO/VSLAM baselines, the *max feature number* is 800. For the three *ORB* variants, the *max feature number* is 800 and the *good feature number* is 100. Results with any tracking failure are omitted from both tables. The *GF* subset selection does not impact the robustness of ORB-SLAM: it works on all eight sequences that ORB tracks. The average RMSE for all tracked sequences per method is given (i.e. All Avg.), as well as the average RMSE of the 5 sequences that all methods track successfully (i.e. Int. Avg.).

On each EuRoC sequence, the minimum RMSE is noted in bold. Interestingly, *GF* does not just preserve the accuracy & robustness of *ORB*; it further reduces the RMSE on several sequences. On average, *GF* has the lowest RMSE over all evaluated VO/VSLAM methods. Furthermore, *GF* also has better overall accuracy when compared with two reference selection methods. Though *Rnd* seems to have lowest RMSE on multiple sequences, the margin between *Rnd* and *GF* small for them. Meanwhile, both *Rnd* and *Long* lead to large accuracy loss on the difficult sequence *MH 04 diff*, while *GF* improves RMSE.

According to Table III, the average latency of *GF* is the lowest relative to all other methods: *GF* has an average latency 34% lower than *ORB*. Compared with the direct methods, the latency of *GF* has lower variance. The 1st quartile of *GF* latency is higher than direct methods, since feature extraction introduces a constant overhead. However, the 3rd quartile of *GF* latency is lower than direct methods, which might occasionally spend too much time on direct optimization.

3) TUM-RGBD Monocular: The RMSE values of all 6 methods (3 baseline VO/VSLAM, 3 *ORB* variants) evaluated on TUM-RGBD are summarized in Table IV. Due to the lower resolution, feature parameters used to obtain the results are roughly half of those configured in EuRoC: *max feature number* of 400, and *good feature number* of 60 (the lower limit recommended and tested in Fig. 7, based on the trends in Figs. 3 and 7). The average RMSE of *GF* is the 2nd lowest,

TABLE II: RMSE (m) on EuRoC Monocular Sequences

	VO/VSLAM					
Seq.	SVO	DSO	ORB	GF	Rnd	Long
MH 01 easy	0.227	0.407	0.027	0.025	**0.024**	0.029
MH 02 easy	0.761	-	**0.034**	0.043	0.038	0.040
MH 03 med	0.798	0.751	0.041	0.045	0.041	**0.040**
MH 04 diff	4.757	-	0.699	**0.492**	1.110	1.377
MH 05 diff	3.505	-	0.346	0.464	**0.216**	0.915
VR1 01 easy	0.726	0.950	0.057	0.037	**0.036**	0.037
VR1 02 med	0.808	**0.536**	-	-	-	-
VR1 03 diff	-	-	-	-	-	-
VR2 01 easy	0.277	0.297	0.025	0.024	0.025	**0.023**
VR2 02 med	0.722	0.880	0.053	**0.051**	**0.051**	0.059
VR2 03 diff	-	-	-	-	-	-
All Avg.	1.477	0.637	0.160	**0.147**	0.193	0.315
Int. Avg.	0.550	0.657	0.041	0.036	**0.035**	0.038

TABLE III: Latency (ms) on EuRoC Monocular Sequences

	VO/VSLAM					
	SVO	DSO	ORB	GF	Rnd	Long
$\boldsymbol{Q_1}$	7.4	**5.8**	13.9	10.3	10.0	10.0
Avg.	12.6	16.4	18.4	**12.2**	12.3	12.3
$\boldsymbol{Q_3}$	16.8	19.1	20.7	13.3	13.2	**13.0**

TABLE IV: RMSE (m) on TUM-RGBD Sequences

	VO/VSLAM					
Seq.	SVO	DSO	ORB	GF	Rnd	Long
f2 desk	0.407	0.975	**0.102**	0.103	0.106	0.109
f2 desk person	1.543	-	**0.042**	0.049	0.184	0.061
f3 long office	-	0.089	0.058	0.058	**0.057**	0.058
All Avg.	0.975	0.532	**0.067**	0.070	0.116	0.076

TABLE V: Latency (ms) on TUM-RGBD Sequences

	VO/VSLAM					
	SVO	DSO	ORB	GF	Rnd	Long
$\boldsymbol{Q_1}$	10.3	**5.8**	8.3	7.1	6.8	6.8
Avg.	12.7	11.5	10.3	8.3	8.1	**8.0**
$\boldsymbol{Q_3}$	15.0	12.0	10.8	8.5	8.6	**8.3**

next to the lowest RMSE from *ORB*. Not surprising, both the accuracy (e.g. average RMSE) and the robustness (e.g. track failure) of direct methods are bad due to rolling shutter effects.

Latency reduction of *GF* is less significant than the previous EuRoC results: it saves around 19% of average latency. Due to the lower image resolution and the relatively short duration of the TUM-RGBD sequences, it is less likely to accumulate enough measurements towards a large 3D feature map. *GF* is best suited to localizing with a relatively large-sized 3D map or domain; on a small map brute-force matching will suffix. This example demonstrates an example scenario with diminishing returns. However, for application scenarios that with improved image quality, a larger domain of operation, and long-term duration, the advantage of *GF* will be clearer.

D. Latency vs. Accuracy: Stereo VO/VSLAM

We also evaluate the latency-accuracy trade-off of stereo *GF* against state-of-the-art stereo VO/VSLAM systems. Compared to monocular VO/VSLAM, the amount of valid map points is much higher in stereo systems because of the extra stereo information. In the presence of a 3D map with high quality and quantity, the advantage of active map-to-frame matching is expected to be more significant than the monocular version. The proposed good feature matching (Alg 3) is integrated into the sped-up ORB-SLAM, *Lz-ORB*. In what follows, we again refer to the good feature enhanced ORB-SLAM as *GF*. As before, two heuristics are integrated into *Lz-ORB* as reference methods, i.e. *Rnd* and *Long*. Four baseline stereo systems are included in the evaluation as well: stereo *SVO*, stereo *DSO* (taken from published results [67] on KITTI since no open-source implementation is available), canonical stereo ORB-SLAM (*ORB*), and *Lz-ORB*, a sped-up version of stereo ORB-SLAM based on the lazy-stereo pipeline described earlier.

The *max feature number* is adjusted for stereo baseline systems to obtain the trade-off curve between accuracy and latency. All 3 baseline systems (*SVO*, *ORB*, and *Lz-ORB*) are configured to have 10-repeat runs under *max feature number* ranging from 150 to 2000. Meanwhile, the latency-accuracy trade-off of *GF* is obtained by adjusting the *good feature number*, which is the the total number of good features from both left and right frames that are matched to the local map. All 3 ORB-SLAM variants (*GF*, *Rnd* and *Long*) are configured for 10-repeat runs under *good feature number* ranging from 60 to 240 (while *max feature number* is fixed).

1) Parameter Exploration for EuRoC Stereo: The latency-accuracy trade-off of stereo VO/VSLAM on three example EuRoC sequences can be found in Fig. 9. Among all 3 baseline systems, *Lz-ORB* has the best accuracy, while *SVO* has the lowest latency. Simply lowering the *max feature number* leads to accuracy drop or even track failure in *Lz-ORB*. However, with *GF* the latency of pose tracking can be reduced to the same level as *SVO*, while the RMSE remains a magnitude lower than *SVO*. Two state-of-the-art stereo VINS systems, *OKVIS*[8] [19] and *MSCKF*[9] [68], are evaluated as well. Both VINS systems are assessed under the default parameters, therefore rather than having the full curve only one marker is presented in Fig. 9. The latency of *GF* is clearly lower than filter-based *MSCKF*, while the accuracy is even better than BA-based *OKVIS*. However, when comparing with two heuristics (*Rnd*, *Long*), the advantage of *GF* is harder to identify than monocular results.

The latency reduction of *GF* is further illustrated in Fig. 10. The *max feature number* being used in *Lz-ORB* and *ORB* is 800, which balances accuracy and latency. Compared with the two non-GF baselines, the latency of *GF* is has a lower upper bound. A reasonable *good feature number* is 160, since it yields low latency as well as high accuracy (the 3rd black mark from the right in Fig. 9).

2) EuRoC Stereo: The RMSEs and latencies of all 6 stereo VO/VSLAM methods under the example configurations (*max feature number* of 800 & *good feature number* of 160) are summarized in Table VI. The results of 2 stereo VINS systems under default parameters are reported as well. Different from monocular VO/VSLAM, it is expected for stereo systems to estimate scale correctly. Therefore, each cell of Table 9 reports the RMSE after *Sim3* alignment (as the 1st value) and the scale

[8]https://github.com/ethz-asl/okvis

[9]https://github.com/KumarRobotics/msckf_vio

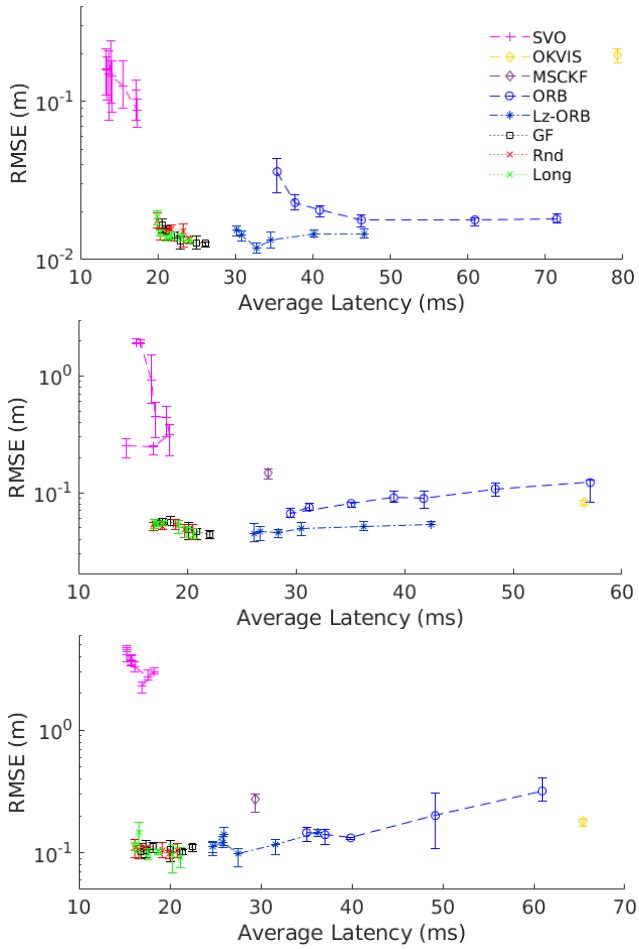


Fig. 9: Latency vs. accuracy on 3 EuRoC Stereo sequences: *MH 01 easy*, *V2 02 med*, and *MH 04 diff* (from top to bottom). Baseline systems are evaluated with *max feature number* ranging from 150 to 2000; ORB-SLAM variants are evaluated with *good feature number* ranging from 60 to 240, and *max feature number* fixed to 800.

error percentage (as the 2nd value). The lowest error within each category, i.e. VO/VSLAM or VINS, is highlighted in bold. Similar to the monocular experiment, *GF* is the lowest in terms of average RMSE and average scale error, compared with other stereo VO/VSLAM methods. Furthermore, the accuracy of *GF* is better than the two stereo VINS systems, while the robustness of *GF* is comparable to stereo VINS (each of them failed on 1 sequence). The advantage of *GF* over *Rnd* and *Long* can be verified as well: both *Rnd* and *Long* failed to track on *MH 02 easy* while *GF* succeed; the average RMSE and scale error of *GF* are lower than the other two as well.

The latency of all 8 stereo systems under the same configuration as Table XIII are summarized in Table VII. The lowest latency is achieved with *SVO*, though its accuracy is an order of magnitude higher than *GF*. Additionally, its third quartile latency is within 10% of the *GF* third quartile latency. The average latency reduction of *GF* is 27.4% when compared to *Lz-ORB*, and 46.2% when compared to *ORB*.

3) KITTI Stereo: The latency-accuracy trade-off of stereo systems on a KITTI sequence *04* are illustrated in Fig. 11. Two relative metrics, RPE and ROE, are estimated with a sliding window of 100m. As suggested in [63] and followed by [3], [67], relative metrics are suited to evaluate accuracy of VO/VSLAM on outdoor large-scale sequences. Stereo *DSO* [67] is not plotted, since there is no open-source implementation available to estimate tracking latency.

Similar to the previous results on indoor scenarios, *GF* is at the bottom-left of the latency-accuracy plane. The latency of *GF* is lower than *ORB*, while the relative error of *GF* is at the same level as *ORB*. *GF* also behaves more robustly than the two reference heuristics: both *Rnd* and *Long* tracks on one out of four configurations, while *GF* works on all four configurations (i.e. has four black-square markers).

Furthermore, we report the RPE & ROE of 7 stereo systems in Table VIII, and the latency in Table IX. Since the image resolution in KITTI is double that of previous benchmarks (captured by VGA/WVGA cameras), the numbers in Table VIII and IX are collected under *max feature number* of 1500. To be consistent with EuRoC stereo results, the *good feature number* is also fixed to 160. Stereo *DSO* results are obtained from the authors' online site;[10] from a single run of each sequence. All other methods are evaluated under 10-repeat runs.

According to Table VIII, *GF* and *ORB* track 10 out of 11 sequences, with *GF* having a lower RPE than *ORB*. The two reference methods, *Rnd* and *Long*, failed to track three sequences. The performance of direct systems varies: *SVO* has the worst accuracy on all 11 sequences, while stereo *DSO* works slightly better than *GF* in terms of accuracy and robustness. Several reasons contribute to the performance of *DSO*: the motion profile of a car is smoother than that of a MAV or hand-held camera; revisits happen at a lower rate than indoor scenarios; and the lighting condition is well-controlled with few low-light cases.

Latency reduction of *GF* is illustrated in Table IX. On average *GF* has 30% less latency than *ORB* and *Lz-ORB* to track a pair of stereo images. *GF* also has a much lower upper bound of pose tracking latency. The latency of *SVO* is higher than *GF*. The latency of *DSO* is not available.

4) TUM-VI Stereo: Under a *max feature number* of 600 and a *good feature number* of 160, we report the RMSE, scale error, and latency of all 8 stereo systems, in Tables X and XI. Compared to stereo VINS systems *OKVIS* and *MSCKF*, *GF* is less robust (i.e. failed to track on *room3*). We further argue that the drop in robustness of *GF* is not due to good feature matching: the original *ORB* tracks all 6 sequences, while our vanilla implementation of *Lz-ORB* fails on *room3* (and all 3 variants thereafter). The track failure for *room3* should be resolved with a better *Lz-ORB* implementation or with the incorporation of IMU measurements.

For the 5 sequences where *GF* succeeds, the RMSE of *GF* is lower than the vision-only baselines (*SVO*, *ORB* and *Lz-ORB*). The average RMSE of vision-only *GF* on 5 tracking sequences is close to that of visual-inertial *OKVIS*, while being lower than that of *MSCKF*. Furthermore, *GF* leads to a 40.2% reduction of average latency versus *Lz-ORB*, and 53.2% latency reduction versus *ORB*, according to Table XI.

[10]https://vision.in.tum.de/research/vslam/stereo-dso

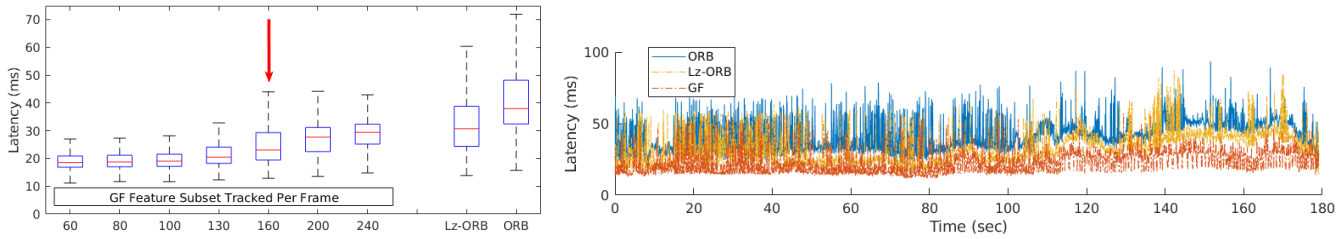


Fig. 10: Latency vs. *good feature number* on EuRoC sequence *MH 01 easy*. **Left**: latency for *GF* under different *good feature number*, and 2 baselines *Lz-ORB* and *ORB*. **Right**: the latency trend of *GF* under 160 *good feature number* (marked with red arrow at the left), *Lz-ORB* and *ORB* in 1 run.

TABLE VI: RMSE (m) and Scale Error (%) on EuRoC Stereo Sequences

	VO/VSLAM						VINS	
Seq.	SVO	ORB	Lz-ORB	GF	Rnd	Long	OKVIS	MSCKF
MH 01 easy	0.179 (0.6)	0.021 (0.7)	**0.012** (**0.5**)	0.013 (**0.5**)	0.016 (**0.5**)	0.014 (**0.5**)	**0.196** (**1.7**)	-
MH 02 easy	-	0.021 (0.3)	**0.018** (**0.1**)	0.021 (**0.1**)	-	-	**0.114** (**1.4**)	0.184 (2.0)
MH 03 med	0.514 (2.3)	0.029 (**0.3**)	**0.024** (0.4)	0.025 (0.4)	0.025 (0.4)	0.025 (0.4)	**0.146** (**0.4**)	0.260 (1.3)
MH 04 diff	3.753 (26.1)	0.140 (1.1)	0.120 (0.6)	0.106 (**0.5**)	0.104 (**0.5**)	**0.102** (0.6)	**0.179** (**0.9**)	0.273 (1.0)
MH 05 diff	1.665 (4.9)	0.096 (**0.2**)	**0.059** (**0.2**)	0.068 (0.3)	0.064 (**0.2**)	0.103 (**0.2**)	**0.266** (**1.2**)	0.356 (2.1)
VR1 01 easy	0.264 (2.3)	**0.033** (0.8)	**0.033** (0.8)	0.035 (**0.7**)	0.035 (0.8)	0.036 (**0.7**)	**0.046** (**0.4**)	0.090 (0.9)
VR1 02 med	0.629 (11.2)	0.064 (**0.4**)	0.047 (0.7)	0.038 (0.7)	**0.032** (0.7)	0.036 (0.7)	**0.068** (0.5)	0.123 (**0.3**)
VR1 03 diff	0.655 (17.4)	0.214 (**2.2**)	0.112 (2.9)	**0.075** (**2.0**)	0.080 (2.1)	0.080 (2.1)	**0.120** (**1.0**)	0.187 (1.1)
VR2 01 easy	0.074 (1.7)	**0.031** (1.1)	0.033 (0.9)	0.044 (**0.5**)	0.041 (0.6)	0.042 (0.6)	**0.053** (0.8)	0.071 (**0.3**)
VR2 02 med	0.447 (3.6)	0.091 (**0.2**)	**0.046** (0.8)	0.049 (0.9)	0.053 (0.9)	0.053 (0.9)	**0.083** (**0.7**)	0.149 (1.0)
VR2 03 diff	**1.618** (**58.7**)	-	-	-	-	-	-	**1.162** (**39.9**)
All Avg.	0.980 (12.9)	0.074 (0.7)	0.050 (0.8)	**0.047** (**0.6**)	0.050 (0.7)	0.054 (0.8)	**0.127** (**0.9**)	0.285 (5.0)
Int. Avg.	1.000 (8.7)	0.087 (0.8)	0.059 (0.9)	0.055 (**0.7**)	**0.054** (0.8)	0.060 (0.8)	**0.120** (**0.7**)	0.189 (1.0)

TABLE VII: Latency (ms) on EuRoC Stereo Sequences

	VO/VSLAM						VINS	
	SVO	ORB	Lz-ORB	GF	Rnd	Long	OKVIS	MSCKF
Q_1	**8.6**	30.0	21.5	14.5	14.2	14.2	50.5	**19.9**
Avg.	**16.4**	38.5	28.5	20.7	19.9	20.1	65.1	**28.3**
Q_3	23.3	44.2	32.1	24.2	**22.5**	22.9	80.3	**36.0**

TABLE VIII: RPE (m/s), and ROE (deg/s) on KITTI Stereo Sequences

	VO/VSLAM						
Seq.	SVO	DSO	ORB	Lz-ORB	GF	Rnd	Long
00	0.632, 0.898	**0.140, 0.163**	0.143, 0.169	0.144, 0.169	0.142, 0.167	0.143, 0.168	0.145, 0.170
01	4.418, 1.585	0.236, **0.041**	**0.234**, 0.068	-	-	-	-
02	0.553, 0.683	0.107, **0.053**	0.106, 0.061	-	**0.105**, 0.062	**0.105**, 0.062	**0.105**, 0.062
03	0.208, 0.170	0.061, **0.030**	0.057, 0.032	0.057, 0.038	**0.054**, 0.036	-	-
04	0.534, 0.337	**0.059, 0.024**	0.066, 0.024	0.061, 0.044	0.061, 0.025	-	-
05	0.312, 0.229	0.047, **0.042**	**0.045**, 0.048	0.046, 0.048	**0.045**, 0.047	**0.045**, 0.047	**0.045**, 0.047
06	0.879, 1.537	**0.061, 0.051**	0.067, 0.053	0.066, **0.051**	0.065, 0.055	-	-
07	0.244, 0.326	**0.048, 0.052**	0.049, 0.057	0.050, 0.059	**0.048**, 0.060	-	-
08	0.456, 0.304	**0.225, 0.055**	0.226, 0.061	-	**0.225**, 0.063	0.226, 0.063	0.226, 0.064
09	0.494, 0.491	0.070, **0.054**	0.065, 0.058	-	**0.062**, 0.058	-	-
10	0.668, 0.874	**0.062, 0.043**	**0.062**, 0.050	-	**0.062**, 0.052	-	-
All Avg.	0.854, 0.676	0.101, **0.055**	0.102, 0.062	**0.071**, 0.068	0.087, 0.062	0.130, 0.085	0.130, 0.086

TABLE IX: Latency (ms) on KITTI Stereo Sequences

	VO/VSLAM						
Seq.	SVO	DSO	ORB	Lz-ORB	GF	Rnd	Long
Q_1	26.0	-	33.9	26.4	21.6	**21.3**	21.4
Avg.	34.7	-	44.8	43.7	29.4	**29.1**	29.4
Q_3	40.7	-	54.0	60.6	31.6	**31.2**	31.4

Interestingly the average RMSE of the two heuristics (*Rnd* and *Long*) are slightly lower than for *GF* and they have a lower latency than *GF*. The advantage of *Rnd* and *Long* is largely due to the set-up of TUM-VI room sequences: these sequences are captured in a small room, with the camera performing repeated circular motion. In such a set-up, the 3D map of the entire room gets constructed after one to two circles, with high quality and a high quantity of features. The success rate of map-to-frame feature matching will be high for a small-scale world with frequent revisits. Under these conditions, simple heuristics such as *Rnd* and *Long* provide sufficient feature matching inliers for pose tracking with less

TABLE X: RMSE (m) and Scale Error (%) on TUM-VI Stereo Sequences

	VO/VSLAM						VINS	
Seq.	SVO	ORB	Lz-ORB	GF	Rnd	Long	OKVIS	MSCKF
room1	1.036 (95.9)	0.290 (8.0)	0.057 (**1.3**)	0.048 (1.6)	**0.040** (1.4)	0.044 (1.4)	**0.065 (0.6)**	0.152 (0.8)
room2	1.208 (97.9)	0.412 (11.4)	0.191 (2.7)	**0.141 (1.8)**	0.145 (1.9)	**0.141 (1.8)**	**0.101 (0.8)**	0.148 (1.5)
room3	1.204 (84.2)	0.160 (4.0)	-	-	-	-	**0.057 (0.4)**	0.201 (2.4)
room4	-	0.156 (4.0)	0.036 (**0.8**)	0.035 (1.0)	0.035 (1.0)	**0.034** (1.0)	**0.026 (0.3)**	0.130 (1.8)
room5	-	0.349 (11.7)	0.028 (**0.4**)	0.029 (**0.3**)	0.029 (**0.3**)	**0.028** (0.4)	**0.048 (0.3)**	0.137 (2.1)
room6	0.756 (55.6)	0.039 (3.3)	0.031 (**1.5**)	0.032 (**1.5**)	**0.030 (1.5)**	**0.030** (1.6)	**0.038 (0.7)**	0.116 (1.2)
All Avg.	1.051 (83.4)	0.234 (7.1)	0.069 (1.3)	0.057 (1.3)	**0.056 (1.2)**	**0.056** (1.3)	**0.056 (0.5)**	0.147 (1.7)
Int. Avg.	1.000 (83.1)	0.247 (7.7)	0.093 (1.8)	0.074 (**1.6**)	**0.072 (1.6)**	**0.072 (1.6)**	**0.068 (0.7)**	0.139 (1.2)

TABLE XI: Latency (ms) on TUM-VI Stereo Sequences

	VO/VSLAM						VINS	
	SVO	ORB	Lz-ORB	GF	Rnd	Long	OKVIS	MSCKF
Q_1	12.2	23.2	17.7	12.3	**11.6**	**11.6**	**5.8**	15.1
Avg.	18.1	29.3	22.9	15.1	14.4	**14.3**	**11.5**	22.2
Q_3	22.3	34.0	26.6	16.2	15.7	**15.4**	**17.0**	28.5

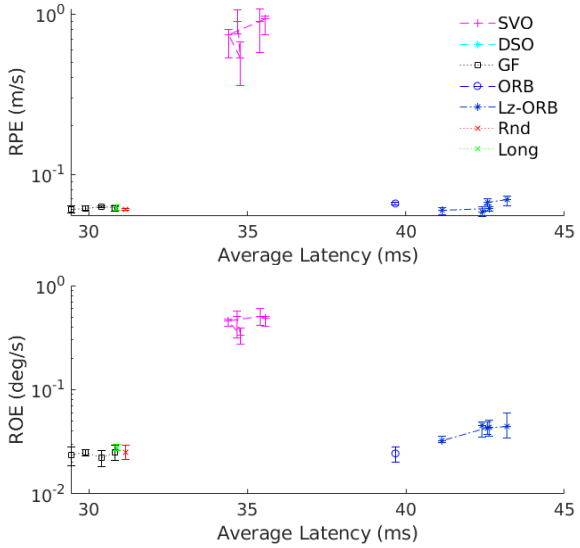


Fig. 11: Latency vs. accuracy on KITTI sequence *04*.

computation demands than search methods such as *GF*.

Based on the performance guarantees described in Section V, which are in expectation, there may exist situations where lazier greedy will operate with similar performance to randomized methods. The TUM scenarios highlight one such set of situations. However, outside of these situations, the two methods are anticipated to diverge. The poorer pose estimation of *Rnd* will then affect the long term pose tracking performance due to the recursive estimation nature of SLAM. This assertion is supported by the evaluation results in EuRoC (medium-scale SLAM) and KITTI (large-scale SLAM), where *GF* has better accuracy and robustness than *Rnd*.

E. Real-time Tracking on Low-Power Devices

Here, the proposed *GF* modification is deployed on three low-power devices with limited processing capabilities, which typically serve as on-board processing units for light weight platforms. The low-power devices tested include:
1) X200CA: a light-weight laptop with an Intel Pentium 2117U processor (Passmark score: 1662 per thread) and 4 GB of RAM. The processor has 2 cores and consumes 17W.
2) Jetson TX2: a 64-bit embedded single-board computer system, containing a hybrid processing unit (2 Denver2 + 4 ARM A57) and 8 GB of RAM. Power consumption is 7.5W.
3) Euclid: a 64-bit embedded single-board computer system, with a Intel Atom x7-Z8700 processor (Passmark score: 552 per thread) and 4 GB of RAM. The processor has 4 cores and consumes 4W.

GF, and three other monocular VO/VSLAM baselines, are deployed and evaluated with EuRoC monocular sequences. To run *ORB* variants near real-time, the pyramid levels for ORB feature extraction were reduced to 3 from 8, and the *max feature number* set to 400. As a consequence, the robustness performance of the *ORB* variants is worse than the previous EuRoC Mono results. In what follows, we relax the robustness condition slightly, and report results with 1 tracking failure in 10 runs as well (marked with underline).

The RMSEs on all three low-power devices are summarized in Table XII, while the latencies are summarized in Table XIII. The *good feature number* is set to 60 given the *max feature number* of 400 (similar to the TUM RGBD benchmark case).

1) When running on X200CA, *GF* has the 2nd lowest average RMSE (23% higher than *ORB*). However, the robustness of *GF* is slightly better than *ORB* and *SVO*: it tracks on 8 sequences without failure, while the other 2 baselines track 7 sequences and with failure. When comparing on the 7 sequences that *ORB* tracks, *GF* only introduces 14% to average RMSE. The strength of *SVO* is the low-latency; though the average latency of *GF* is 24% less than *ORB*, it is almost twice that of *SVO*.
2) The released binary of *SVO* does not support 64-bit Jetson TX2, therefore only 3 methods are assessed on Jetson. Similar to the X200CA results, *GF* is slightly worse than *ORB* in terms of average RMSE (by 8%). Notice *GF* is also less robust than *ORB*, as it introduces additional tracking failure on sequences *MH 02 easy* and *MH 04 diff*. The latency reduction of *GF* is also small: 11% less than *ORB*.
3) When running on Euclid, *GF* introduces 20% more error in terms of average RMSE. Again, notice that *GF* works on *MH 05 diff* while *ORB* cannot. If we only take the 6 sequences that

TABLE XII: RMSE (m) On EuRoC Monocular Systems, Running on Low-power Devices.

	X200CA				Jetson			Euclid					
Seq.	SVO	DSO	ORB	GF	DSO	ORB	GF	SVO	DSO	ORB	GF	SVOMSF	VIMono
MH 01 easy	0.327	-	0.041	**0.036**	-	**0.033**	0.037	0.244	-	0.044	**0.041**	0.29	0.20
MH 02 easy	-	-	0.053	**0.047**	-	**0.046**	0.135	-	-	**0.044**	0.045	0.31	0.18
MH 03 med	1.14	-	**0.050**	0.056	-	**0.055**	0.059	1.21	-	**0.050**	0.051	0.66	0.17
MH 04 diff	-	-	**0.281**	0.457	-	**0.231**	-	-	-	0.232	0.248	2.02	**0.12**
MH 05 diff	2.54	-	0.289	**0.233**	-	**0.258**	0.340	2.84	-	-	**0.158**	0.87	0.35
VR1 01 easy	0.552	-	**0.036**	**0.036**	0.826	**0.036**	**0.036**	0.645	-	**0.036**	0.040	0.36	0.05
VR1 02 med	**0.730**	-	-	-	-	-	-	**0.857**	-	-	-	0.78	**0.12**
VR1 03 diff	-	-	-	-	-	-	-	-	-	-	-	-	**0.10**
VR2 01 easy	0.397	0.295	0.032	**0.029**	0.288	0.029	**0.027**	0.402	0.300	**0.030**	0.032	0.33	0.08
VR2 02 med	0.634	0.832	-	**0.213**	**0.941**	-	-	0.688	-	-	-	0.59	**0.08**
VR2 03 diff	-	-	-	-	-	-	-	-	-	-	-	-	**0.17**
All Avg.	0.903	0.564	**0.112**	0.138	0.685	**0.098**	0.106	0.984	0.300	**0.073**	0.088	0.69	0.15
Int. Avg.	-	-	**0.112**	0.128	-	**0.076**	0.106	-	-	**0.073**	0.076	0.66	0.13

TABLE XIII: Latency (ms) On EuRoC Monocular Systems, Running on Low-power Devices.

	X200CA				Jetson			Euclid					
	SVO	DSO	ORB	GF	DSO	ORB	GF	SVO	DSO	ORB	GF	SVOMSF	VIMono
Q_1	**8.6**	12.4	19.3	14.5	**21.5**	30.2	25.7	**12.6**	21.5	28.7	24.8	29.8	88.9
Avg.	**9.8**	15.0	24.6	18.7	32.1	35.1	**31.1**	**13.4**	37.3	35.9	32.6	37.5	153.9
Q_3	**12.5**	16.4	28.5	21.0	37.5	38.8	**33.1**	**15.9**	51.1	41.4	39.3	42.1	209.5

ORB tracks into account, *GF* only introduces 4% to average RMSE. However, the latency reduction of *GF* for Euclid is smaller than the Jetson results: only 9% time savings. Apart from the 4 monocular VO/VSLAM systems, we also include the VINS results [69] evaluated on a UP Board, which has almost identical hardware specifications as Euclid. The RMSE of the VINS methods, labeled *SVOMSF* [69] and *VIMono* [70], are obtained by *Sim3* alignment to ground truth, which is identical with our evaluation. With additional input from inertial sensors, VINS are clearly more robust than vision-only systems. However, the accuracy of VINS is poorer than vision-only ones (when scale corrected). Furthermore, the latency of the VINS approaches is much higher than vision-only systems, which suggests the scalability of VINS is also poor for low-power devices. Therefore, for VO/VSLAM and VINS, a combination of algorithm improvements (e.g. Good Feature) and hardware improvements may be required to achieve low latency and good accuracy on embedded devices.

When the computate resources (e.g. processor speed, cache size) are highly limited, the latency reduction of *GF* is less significant. Preservation of accuracy & robustness, on the other hand, scales relatively well on different devices (only with slight drop). The limited scalability to devices such as Jetson & Euclid is mostly due to the sequential nature of the proposed *GF* algorithm. As embedded device hardware specifications improve, in terms of compute power and core quantity, we anticipate that improvements will favor the *GF* variant (as demonstrated on desktop and X200CA). Even on current embedded platforms, the small amount of latency reduced by *GF* could be important: it turns the near real-time *ORB* into a real-time applicable VSLAM system, as illustrated in Fig. 12.

Given the time cost of feature extraction (Fig. 12), efforts to move feature extraction onto FPGA devices [9], [10] are crucial. The times in Fig. 12 reflect a coarser pyramid and smaller feature extraction numbers. Off-loading the original ORB-SLAM pyramidal feature extraction block to an FPGA

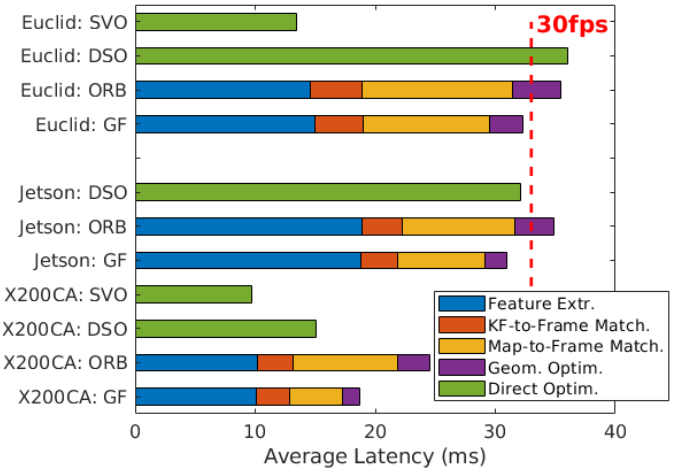


Fig. 12: Latency breakdown for all modules in pose tracking pipeline, running on low-power devices.

will have significant savings and would help preserve the accuracy properties of the original front end. When combining FPGA off-loading with the Good Feature matching method (and possibly also IMU integration), we expect the performance-efficiency of VSLAM on low-power devices be similar to the desktop outcomes (e.g. Table II).

VIII. Conclusion

This paper presents an active map-to-frame feature matching method, good feature matching, which reduces the computational cost (and therefore latency) of VO/VSLAM, while preserving the accuracy and robustness of pose tracking. Feature matching is connected to the submatrix selection problem. To that end, the *Max-logDet* matrix revealing metric was shown to perform best via simulated scenarios. For application to active feature matching, the combination of deterministic selection (greedy) and randomized acceleration (random sampling) is studied. The proposed good feature matching algorithm is

integrated into monocular and stereo feature-based VSLAM systems, followed by evaluation on multiple benchmarks and computate platforms. Good feature matching is shown to be an efficiency enhancement for low-latency VO/VSLAM, while preserving, if not improving, the accuracy and robustness of VO/VSLAM. Though the focus of this paper is reducing the latency of VO/VSLAM, the idea of active & logDet-guided feature matching is general: it can be extended to other feature modules (e.g. line features [22]) or localization tasks (e.g. image-based localization [71]).

REFERENCES

[1] A. J. Davison, I. D. Reid, N. D. Molton, and O. Stasse, "MonoSLAM: Real-time single camera SLAM," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 29, no. 6, pp. 1052–1067, 2007.

[2] G. Klein and D. Murray, "Parallel tracking and mapping for small AR workspaces," in *IEEE/ACM International Symposium on Mixed and Augmented Reality*, 2007, pp. 225–234.

[3] R. Mur-Artal, J. M. M. Montiel, and J. D. Tardos, "ORB-SLAM: a versatile and accurate monocular SLAM system," *IEEE Transactions on Robotics*, vol. 31, no. 5, pp. 1147–1163, 2015.

[4] J. Engel, T. Schöps, and D. Cremers, "LSD-SLAM: Large-Scale Direct monocular SLAM," in *European Conference on Computer Vision*. Springer, 2014, pp. 834–849.

[5] C. Forster, Z. Zhang, M. Gassner, M. Werlberger, and D. Scaramuzza, "SVO: Semidirect visual odometry for monocular and multicamera systems," *IEEE Transactions on Robotics*, vol. 33, no. 2, pp. 249–265, 2017.

[6] J. Engel, V. Koltun, and D. Cremers, "Direct sparse odometry," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 40, no. 3, pp. 611–625, 2017.

[7] R. A. Newcombe, S. Izadi, O. Hilliges, D. Molyneaux, D. Kim, A. J. Davison, P. Kohi, J. Shotton, S. Hodges, and A. Fitzgibbon, "KinectFusion: Real-time dense surface mapping and tracking," in *IEEE International Symposium on Mixed and Augmented Reality*, 2011, pp. 127–136.

[8] T. Whelan, R. F. Salas-Moreno, B. Glocker, A. J. Davison, and S. Leutenegger, "ElasticFusion: Real-time dense SLAM and light source estimation," *The International Journal of Robotics Research*, vol. 35, no. 14, pp. 1697–1716, 2016.

[9] W. Fang, Y. Zhang, B. Yu, and S. Liu, "FPGA-based ORB feature extraction for real-time visual SLAM," in *IEEE International Conference on Field Programmable Technology*, 2017, pp. 275–278.

[10] M. Quigley, K. Mohta, S. S. Shivakumar, M. Watterson, Y. Mulgaonkar, M. Arguedas, K. Sun, S. Liu, B. Pfrommer, V. Kumar *et al.*, "The open vision computer: An integrated sensing and compute system for mobile robots," *arXiv preprint arXiv:1809.07674*, 2018.

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

[12] B. Kueng, E. Mueggler, G. Gallego, and D. Scaramuzza, "Low-latency visual odometry using event-based feature tracks," in *IEEE/RSJ International Conference on Intelligent Robots and Systems*, 2016, pp. 16–23.

[13] H. Rebecq, T. Horstschaefer, and D. Scaramuzza, "Real-time visualinertial odometry for event cameras using keyframe-based nonlinear optimization," in *British Machine Vision Conference*, vol. 3, 2017.

[14] A. Z. Zhu, N. Atanasov, and K. Daniilidis, "Event-based visual inertial odometry." in *IEEE Conference on Computer Vision and Pattern Recognition*, 2017, pp. 5816–5824.

[15] M. Calonder, V. Lepetit, C. Strecha, and P. Fua, "BRIEF: Binary robust independent elementary features," in *European Conference on Computer Vision*. Springer, 2010, pp. 778–792.

[16] S. Leutenegger, M. Chli, and R. Y. Siegwart, "BRISK: Binary robust invariant scalable keypoints," in *IEEE International Conference on Computer Vision*, 2011, pp. 2548–2555.

[17] E. Rublee, V. Rabaud, K. Konolige, and G. Bradski, "ORB: An efficient alternative to SIFT or SURF," in *IEEE International Conference on Computer Vision*, 2011, pp. 2564–2571.

[18] P. Vandergheynst, R. Ortiz, and A. Alahi, "FREAK: Fast retina keypoint," in *IEEE Conference on Computer Vision and Pattern Recognition*, 2012, pp. 510–517.

[19] S. Leutenegger, S. Lynen, M. Bosse, R. Siegwart, and P. Furgale, "Keyframe-based visual–inertial odometry using nonlinear optimization," *The International Journal of Robotics Research*, vol. 34, no. 3, pp. 314–334, 2015.

[20] B. Přibyl, P. Zemčík, and M. Čadík, "Camera pose estimation from lines using Plücker coordinates," in *British Machine Vision Conference*, 2015, pp. 1–12.

[21] R. Gomez-Ojeda, F.-A. Moreno, D. Zuñiga-Noël, D. Scaramuzza, and J. Gonzalez-Jimenez, "PL-SLAM: a stereo SLAM system through the combination of points and line segments," *IEEE Transactions on Robotics*, 2019.

[22] Y. Zhao and P. A. Vela, "Good line cutting: towards accurate pose tracking of line-assisted VO/VSLAM," in *European Conference on Computer Vision*. Springer, 2018, pp. 516–531.

[23] N. Yang, R. Wang, X. Gao, and D. Cremers, "Challenges in monocular visual odometry: Photometric calibration, motion bias, and rolling shutter effect," *IEEE Robotics and Automation Letters*, vol. 3, no. 4, pp. 2878–2885, 2018.

[24] D. Schubert, N. Demmel, V. Usenko, J. Stuckler, and D. Cremers, "Direct sparse odometry with rolling shutter," in *European Conference on Computer Vision*. Springer, 2018, pp. 682–697.

[25] 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*, vol. 35, no. 10, pp. 1157–1163, 2016.

[26] Y. Zhao and P. Vela, "Good feature selection for least squares pose optimization in VO/VSLAM," in *IEEE/RSJ International Conference on Intelligent Robots and Systems*, 2018, pp. 3569–3574.

[27] J. Neira and J. D. Tardós, "Data association in stochastic mapping using the joint compatibility test," *IEEE Transactions on Robotics and Automation*, vol. 17, no. 6, pp. 890–897, 2001.

[28] A. Vedaldi, H. Jin, P. Favaro, and S. Soatto, "KalmanSAC: Robust filtering by consensus," in *IEEE International Conference on Computer Vision*, 2005, pp. 633–640.

[29] J. Civera, O. G. Grasa, A. J. Davison, and J. Montiel, "1-Point RANSAC for extended Kalman filtering: Application to real-time structure from motion and visual odometry," *Journal of Field Robotics*, vol. 27, no. 5, pp. 609–631, 2010.

[30] I. Cvišić and I. Petrović, "Stereo odometry based on careful feature selection and tracking," in *European Conference on Mobile Robots*, 2015, pp. 1–6.

[31] G. Zhang and P. A. Vela, "Optimally observable and minimal cardinality monocular SLAM," in *IEEE International Conference on Robotics and Automation*, 2015, pp. 5211–5218.

[32] G. Zhang and P. A. Vela, "Good features to track for visual SLAM," in *IEEE Conference on Computer Vision and Pattern Recognition*, 2015, pp. 1373–1382.

[33] J. Shi and C. Tomasi, "Good features to track," in *IEEE Conference on Computer Vision and Pattern Recognition*, 1994, pp. 593–600.

[34] P. Sala, R. Sim, A. Shokoufandeh, and S. Dickinson, "Landmark selection for vision-based navigation," *IEEE Transactions on Robotics*, vol. 22, no. 2, pp. 334–349, 2006.

[35] Z. Shi, Z. Liu, X. Wu, and W. Xu, "Feature selection for reliable data association in visual SLAM," *Machine Vision and Applications*, pp. 1–16, 2013.

[36] M. Kaess and F. Dellaert, "Covariance recovery from a square root information matrix for data association," *Robotics and Autonomous Systems*, vol. 57, no. 12, pp. 1198–1210, 2009.

[37] S. Zhang, L. Xie, and M. D. Adams, "Entropy based feature selection scheme for real time simultaneous localization and map building," in *IEEE/RSJ International Conference on Intelligent Robots and Systems*, 2005, pp. 1175–1180.

[38] R. Lerner, E. Rivlin, and I. Shimshoni, "Landmark selection for task-oriented navigation," *IEEE Transactions on Robotics*, vol. 23, no. 3, pp. 494–505, 2007.

[39] F. A. Cheein, G. Scaglia, F. di Sciasio, and R. Carelli, "Feature selection criteria for real time EKF-SLAM algorithm," *International Journal of Advanced Robotic Systems*, vol. 6, no. 3, p. 21, 2009.

[40] L. Carlone and S. Karaman, "Attention and anticipation in fast visual-inertial navigation," *IEEE Transactions on Robotics*, vol. 35, no. 1, pp. 1–20, 2019.

[41] V. Ila, L. Polok, M. Solony, and K. Istenic, "Fast incremental bundle adjustment with covariance recovery," in *IEEE International Conference on 3D Vision*, 2017, pp. 175–184.

[42] V. Ila, L. Polok, M. Solony, and P. Svoboda, "SLAM++-a highly efficient and temporally scalable incremental SLAM framework," *The*

International Journal of Robotics Research, vol. 36, no. 2, pp. 210–230, 2017.

[43] M. Gu and S. C. Eisenstat, "Efficient algorithms for computing a strong rank-revealing QR factorization," *SIAM Journal on Scientific Computing*, vol. 17, no. 4, pp. 848–869, 1996.

[44] C. Boutsidis, M. W. Mahoney, and P. Drineas, "An improved approximation algorithm for the column subset selection problem," in *ACM-SIAM Symposium on Discrete Algorithms*, 2009, pp. 968–977.

[45] M. Shamaiah, S. Banerjee, and H. Vikalo, "Greedy sensor selection: Leveraging submodularity," in *IEEE Conference on Decision and Control*, 2010, pp. 2572–2577.

[46] S. T. Jawaid and S. L. Smith, "Submodularity and greedy algorithms in sensor scheduling for linear dynamical systems," *Automatica*, vol. 61, pp. 282–288, 2015.

[47] T. H. Summers, F. L. Cortesi, and J. Lygeros, "On submodularity and controllability in complex dynamical networks," *IEEE Transactions on Control of Network Systems*, vol. 3, no. 1, pp. 91–101, 2016.

[48] P. Drineas, M. W. Mahoney, and S. Muthukrishnan, "Relative-error CUR matrix decompositions," *SIAM Journal on Matrix Analysis and Applications*, vol. 30, no. 2, pp. 844–881, 2008.

[49] P. Drineas, M. Magdon-Ismail, M. W. Mahoney, and D. P. Woodruff, "Fast approximation of matrix coherence and statistical leverage," *Journal of Machine Learning Research*, vol. 13, no. Dec, pp. 3475–3506, 2012.

[50] C. Boutsidis, P. Drineas, and M. Magdon-Ismail, "Near-optimal column-based matrix reconstruction," *SIAM Journal on Computing*, vol. 43, no. 2, pp. 687–717, 2014.

[51] B. Mirzasoleiman, A. Badanidiyuru, A. Karbasi, J. Vondrák, and A. Krause, "Lazier than lazy greedy." in *AAAI Conference on Artificial Intelligence*, 2015, pp. 1812–1818.

[52] A. Hassidim and Y. Singer, "Robust guarantees of stochastic greedy algorithms," in *International Conference on Machine Learning*, 2017, pp. 1424–1432.

[53] A. Davison, "Active search for real-time vision," in *IEEE International Conference on Computer Vision*, vol. 1, 2005, pp. 66–73.

[54] M. Chli and A. J. Davison, "Active matching," in *European Conference on Computer Vision*. Springer, 2008, pp. 72–85.

[55] A. Handa, M. Chli, H. Strasdat, and A. Davison, "Scalable active matching," in *IEEE Conference on Computer Vision and Pattern Recognition*, 2010, pp. 1546–1553.

[56] A. J. Davison, "FutureMapping: The computational structure of spatial AI systems," *arXiv preprint arXiv:1803.11288*, 2018.

[57] J. Sola, T. Vidal-Calleja, J. Civera, and J. M. M. Montiel, "Impact of landmark parametrization on monocular EKF-SLAM with points and lines," *International Journal of Computer Vision*, vol. 97, no. 3, pp. 339–368, 2012.

[58] G. H. Golub and C. F. Van Loan, *Matrix computations*. JHU Press, 2012, vol. 3.

[59] M. Minoux, "Accelerated greedy algorithms for maximizing submodular set functions," in *Optimization techniques*. Springer, 1978, pp. 234–243.

[60] R. A. Horn, R. A. Horn, and C. R. Johnson, *Matrix analysis*. Cambridge university press, 1990.

[61] D. Schubert, T. Goll, N. Demmel, V. Usenko, J. Stueckler, and D. Cremers, "The TUM VI benchmark for evaluating visual-inertial odometry," in *IEEE/RJS International Conference on Intelligent Robot Systems*, 2018.

[62] J. Sturm, W. Burgard, and D. Cremers, "Evaluating egomotion and structure-from-motion approaches using the TUM RGB-D benchmark," in *Workshop on Color-Depth Camera Fusion in Robotics at the IEEE/RJS International Conference on Intelligent Robot Systems*, 2012.

[63] A. Geiger, P. Lenz, C. Stiller, and R. Urtasun, "Vision meets robotics: The KITTI dataset," *The International Journal of Robotics Research*, vol. 32, no. 11, pp. 1231–1237, 2013.

[64] L. Nardi, B. Bodin, M. Z. Zia, J. Mawer, A. Nisbet, P. H. Kelly, A. J. Davison, M. Luján, M. F. O'Boyle, G. Riley *et al.*, "Introducing SLAMBench, a performance and accuracy benchmarking methodology for SLAM," in *IEEE International Conference on Robotics and Automation*, 2015, pp. 5783–5790.

[65] B. Bodin, H. Wagstaff, S. Saecdi, L. Nardi, E. Vespa, J. Mawer, A. Nisbet, M. Luján, S. Furber, A. J. Davison *et al.*, "SLAMBench2: Multi-objective head-to-head benchmarking for visual SLAM," in *IEEE International Conference on Robotics and Automation*, 2018, pp. 1–8.

[66] S. Saeedi, B. Bodin, H. Wagstaff, A. Nisbet, L. Nardi, J. Mawer, N. Melot, O. Palomar, E. Vespa, T. Spink *et al.*, "Navigating the landscape for real-time localization and mapping for robotics and virtual and augmented reality," *Proceedings of the IEEE*, no. 99, pp. 1–20, 2018.

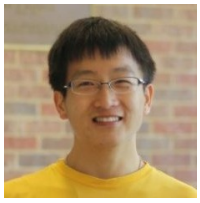
[67] R. Wang, M. Schworer, and D. Cremers, "Stereo DSO: Large-scale direct sparse visual odometry with stereo cameras," in *IEEE International Conference on Computer Vision*, 2017, pp. 3903–3911.

[68] K. Sun, K. Mohta, B. Pfrommer, M. Watterson, S. Liu, Y. Mulgaonkar, C. J. Taylor, and V. Kumar, "Robust stereo visual inertial odometry for fast autonomous flight," *IEEE Robotics and Automation Letters*, vol. 3, no. 2, pp. 965–972, 2018.

[69] J. Delmerico and D. Scaramuzza, "A benchmark comparison of monocular visual-inertial odometry algorithms for flying robots," *IEEE International Conference on Robotics and Automation*, vol. 10, p. 20, 2018.

[70] T. Qin, P. Li, and S. Shen, "VINS-Mono: A robust and versatile monocular visual-inertial state estimator," *IEEE Transactions on Robotics*, vol. 34, no. 4, pp. 1004–1020, 2018.

[71] Y. Zhao, W. Ye, and P. Vela, "Low-latency visual SLAM with appearance-enhanced local map building," in *IEEE International Conference on Robotics and Automation*, 2019, pp. 8213–8219.



Yipu Zhao obtained his Ph.D. in 2019, under the supervision of Patricio A. Vela, at the School of Electrical and Computer Engineering, Georgia Institute of Technology, USA. Previously he received his B.Sc. degree in 2010 and M.Sc. degree in 2013, at the Institute of Artificial Intelligence, Peking University, China. His research interests include visual odometry/SLAM, 3D reconstruction, and multi-object tracking.

Patricio A. Vela is an associate professor in the School of Electrical and Computer Engineering, and the Institute of Robotics and Intelligent Machines, at Georgia Institute of Technology, USA. His research interests lie in the geometric perspectives to control theory and computer vision. Recently, he has been interested in the role that computer vision can play for achieving control-theoretic objectives of (semi-)autonomous systems. His research also covers control of nonlinear systems, typically robotic systems.

Prof. Vela earned his B.Sc. degree in 1998 and his Ph.D. degree in control and dynamical systems in 2003, both from the California Institute of Technology, where he did his graduate research on geometric nonlinear control and robotics. In 2004, Dr. Vela was as a post-doctoral researcher on computer vision with School of ECE, Georgia Tech. He join the ECE faculty at Georgia Tech in 2005.

Prof. Vela is a member of IEEE.