Attention and Anticipation in Fast Visual-Inertial Navigation

Luca Carlone and Sertac Karaman

***Abstract*—We study a *Visual-Inertial Navigation* (VIN) problem in which a robot needs to estimate its state using an on-board camera and an inertial sensor, without any prior knowledge of the external environment. We consider the case in which the robot can allocate limited resources to VIN, due to tight computational constraints. Therefore, we answer the following question: under limited resources, what are the most relevant visual cues to maximize the performance of visual-inertial navigation? Our approach has four key ingredients. First, it is *task-driven*, in that the selection of the visual cues is guided by a metric quantifying the VIN performance. Second, it exploits the notion of *anticipation*, since it uses a simplified model for forward-simulation of robot dynamics, predicting the utility of a set of visual cues over a future time horizon. Third, it is *efficient and easy to implement*, since it leads to a greedy algorithm for the selection of the most relevant visual cues. Fourth, it provides *formal performance guarantees*: we leverage submodularity to prove that the greedy selection cannot be far from the optimal (combinatorial) selection. Simulations and real experiments on agile drones show that our approach leads to dramatic improvements in the VIN performance. In the easy scenarios, our approach outperforms the state-of-the-art in terms of localization errors. In the most challenging scenarios, it enables accurate visual-inertial navigation while the state of the art fails to track robot's motion during aggressive maneuvers.**

Supplementary Material

- Video: https://www.youtube.com/watch?v=uMLXNRiVuyU

I. Introduction

The human brain can extract conceptual information from an image in a time lapse as short as 13 ms [1]. One has proof of the human's capability to seamlessly process large amount of sensory data in everyday tasks, including driving a car on a highway, or walking on a crowded street. In the cognitive science literature, there is agreement on the fact that efficiency in processing the large amount of data we are confronted with is due to our ability to prioritize some aspects of the visual scene, while ignoring others [2]. One can imagine that sensory inputs compete to have access to the limited computational resources of our brain. These resource constraints are dictated by the fixed amount of energy available to the brain as well as time constraints imposed by time-critical tasks. *Visual attention* is the cognitive process that allows humans to parse a large amount of visual data by selecting relevant information and filtering out irrelevant stimuli, so to maximize performance[1] under limited resources.

L. Carlone and S. Karaman are with the Laboratory for Information & Decision Systems (LIDS), Massachusetts Institute of Technology, Cambridge, MA, USA, {lcarlone,sertac}@mit.edu

[1]This definition oversimplifies the attention mechanisms in humans. While the role of attention is to optimally allocate resources to maximize performance, it is known that some involuntary attention mechanisms can actually hinder the correct execution of a task [2].

Robots vs. humans. The astonishing progress in robotics and computer vision over the last three decades might induce us to ask: how far is robot perception from human performance? Let us approach this question by looking at the state of the art in visual processing for different tasks. Without any claim to be exhaustive, we consider few representative papers (sampled over the last 2 years) and we only look at timing performance. A state-of-the-art approach for object detection [3] detects objects in a scene in 22ms on a Titan X GPU. A high-performance approach for stereo reconstruction [4] builds a triangular mesh of a 3D scene in 10-100ms on a single CPU (at resolution 800×600). A state-of-the-art vision-based SLAM approach [5] requires around 400ms for local mapping and motion tracking and more than 1s for global map refinement (CPU, multiple cores). The reader may notice that for each task, in isolation, modern algorithms require more time than what a human needs to parse an entire scene. Arguably, while a merit of the robotics and computer vision communities has been to push performance in each task, we are quite far from a computational model in which all these tasks (pose estimation, geometry reconstruction, scene understanding) are concurrently executed in the blink of an eye.

Efficiency via general-purpose computing. One might argue that catching up with human efficiency is only a matter of time: according to Moore's law, the available computational power grows at exponential rate, hence we only need to wait for more powerful computers. An analogous argument would suggest that using GPU rather than CPU would boost performance in some of the tasks mentioned above. By comparison with human performance, we realize that this argument is not completely accurate. While it is true that we can keep increasing the computational resources to meet given time constraints (i.e., enable faster processing of sensory data), the increase in computation implies an increase in energy consumption; for instance, a Titan X GPU has a nominal power consumption of 250W [6] while a Core i7 CPU has a power consumption as low as 11W [7]. On the other hand, human processing constantly deals with limited time and energy constraints, and is parsimonious in allocating only the resources that are necessary to accomplish its goals.

Efficiency via specialized computing. Another potential alternative to enable high-rate low-power perception and bridge the gap between human and robot perception is to design specialized hardware for machine perception. As extensively discussed in our previous work [8], algorithms and hardware co-design allows minimizing resource utilization by exploiting a tight-integration of algorithms and specialized hardware, and leveraging opportunities (e.g., pipelining, low-cost arithmetic) provided by ASICs (Application-Specific Integrated Circuits) and FPGAs (Field-Programmable Gate Arrays). While we

have shown that using specialized hardware for VIN leads to a reduction of the power consumption of 1-2 orders of magnitude (with comparable performance), three main observations motivate the present work. First, the development of specialized hardware for perception is an expensive and time-consuming process and the resulting hardware is difficult to upgrade. Second, rather than designing optimized hardware that can meet given performance requirements, it may be desirable to develop a framework that can systematically trade-off performance for computation, hence more flexibly adjusting to the available, possibly time-varying, computational resources and performance requirements. Third, extensive biological evidence suggests that efficient perception requires both specialized circuitry (e.g., visual perception in humans is carried out by highly specialized areas of the brain [9]) and a mechanism to prioritize stimuli (i.e., visual attention [2]).

Contribution. In this paper, we investigate how to speed-up computation (or, equivalently, reduce the computational effort) in VIN by prioritizing sensor data in a task-dependent fashion. We consider the case in which, due to constraints on the on-board computation, the robot can only use a small number of visual features in the environment to support motion estimation. We then design a visual attention mechanism that selects a suitable set of visual features to maximize localization accuracy; our general framework is presented in Section III.

Our approach is task-driven: it selects features that maximize a task-dependent performance metric, that we present in Section III-A. Contrarily to the literature on visual feature selection, we believe that the utility of a feature is not an intrinsic property of the feature itself (e.g., appearance), but it rather stems from the intertwining of the environment and the observer state. Our approach seamlessly captures both visual saliency and the task-dependent utility of a set of features.

Our attention mechanism is predictive in nature: when deciding which feature is more useful, our approach performs fast forward-simulations of the state of the robot leading to a feature selection that is aware of the dynamics and the "intentions" of the robot. The forward simulation is based on a simplified model which we present in Section III-B.

Since the optimal allocation of the resources in a hard combinatorial problem, in Section IV we present a greedy algorithm for attention allocation. In the same section, we leverage recent results on submodularity to provide formal performance guarantees for the greedy algorithm. Section IV also reviews related techniques based on convex relaxations.

Section V provides an experimental evaluation of the proposed approach. The results confirm that our approach can boost performance in standard VIN pipelines and enables accurate navigation under agile motions and strict resource constraints. The proposed approach largely outperforms appearance-based feature selection methods, and drastically reduces the computational time required by the VIN back-end.

This paper extends the preliminary results presented in [10]. In particular, the discussion on convex relaxations for features selection (Section IV-A and Section IV-B), the performance guarantees of Proposition 11, the simulation results of Section V-A, and the experimental evaluation on the 11 *EuRoC* datasets [11] are novel and have not been previously published.

II. Related Work

This work intersects several lines of research across fields.

Attention and Saliency in Neuroscience and Psychology. Attention is a central topic in human and animal vision research with more than 2500 papers published since the 1980s [2]. While a complete coverage is outside the scope of this work, we review few basic concepts, using the surveys of Carrasco [2], Borji and Itti [12], Scholl [13], and the work of Caduff and Timpf [14] as main references. Scholl [13] defines attention as the discrimination of sensory stimuli, and the allocation of limited resources to competing attentional demands. Carrasco [2] identifies three types of attention: *spatial*, *feature-based*, and *object-based*. Spatial attention prioritizes different locations of the scene by moving the eyes towards a specific location (*overt* attention) or by focusing on relevant locations within the field of view (*covert* attention). Feature-based attention prioritizes the detection of a specific feature (color, motion direction, orientation) independently on its location. Object-based attention prioritizes specific objects. In this work, we are mainly interested in covert spatial attention: which locations in the field of view are the most informative for navigation? Covert attention in humans is a combination of voluntary and involuntary mechanisms that guide the processing of visual stimuli at given locations in the scene [2]. Empirical evidence shows that attention is task-dependent in both primates and humans [14], [15]. Borji and Itti [12] explicitly capture this aspect by distinguishing bottom-up and top-down attention models; in the former the attention is captured by visual cues (stimulus-driven), while in the latter the attention is guided by the goal of the observer. Caduff and Timpf [14] study landmark saliency in human navigation and conclude that saliency stems from the intertwining of intrinsic property of a landmark (e.g., appearance) and the state of the observer (e.g., prior knowledge, observation pose). Another important aspect, that traces back to the *guided search theory* of Wolfe [16] and Spekreijse [17], is the distinction between pre-attentive and attentive visual processes. Pre-attentive processes handle *all* incoming sensory data in parallel; then, attentive processes only work on a filtered-out-version of the data, which the brain deems more relevant. General computational models for attention are reviewed in [12], including Bayesian models, graph-theoretic, and information-theoretic formulations.

Feature Selection in Robotics and Computer Vision. The idea of enhancing performance in visual SLAM and visual odometry via active feature selection is not novel. Sim and Dudek [18] and Peretroukhin *et al.* [19] use training data to learn a model of the quality of the visual features. Each feature is mapped from a hand-crafted predictor space to a scalar weight that quantifies its usefulness for pose estimation; in [19] the weights are then used to rescale the measurement covariance of each observation. Ouerhani *et al.* [20] construct a topological map using attentional landmarks. Newman and Ho [21] consider a robot equipped with camera and laser range finder and perform feature selection using an appearance-based notion of visual saliency. Sala *et al.* [22] use a co-

visibility criterion to select good landmarks that are visible from multiple viewpoints. Siagian and Itti [23] investigate a bio-inpired attention model within Monte Carlo localization. Frintrop and Jensfelt [24] use an attention framework for landmark selection and active gaze control; feature selection is based on the VOCUS model [24], which includes a bottom-up attentional system (which computes saliency from the feature appearance), and can incorporate a top-down mechanism (which considers task performance). Active gaze control, instead, is obtained as the combination of three behaviors: landmark redetection, landmark tracking, and exploration of new areas. Hochdorfer and Schlegel [25] propose a landmark rating and selection mechanism based on area coverage to enable life-long mapping. Strasdat *et al.* [26] propose a reinforcement learning approach for landmark selection. Chli and Davison [27] and Handa *et al.* [28] use available priors to inform feature matching, hence reducing the computational cost. Jang *et al.* [29] propose an approach for landmark classification to improve accuracy in visual odometry; each feature class is used separately to estimate rotational and translational components of the ego-motion. Shi *et al.* [30] propose a feature selection technique to improve robustness of data association in SLAM. Very recent work in computer vision use attention to reduce the computational burden in neural networks. Mnih *et al.* [31] reduce the processing of object detection and tracking with a recurrent neural network by introducing the notion of *glimpse*, which provides higher resolution in areas of interest within the image. Xu *et al.* [32] use visual attention to improve image content description. Cvišić and Petrović [33] speed up computation in stereo odometry by feature selection; the selection procedure is based on bucketing (which uniformly distributes the features across the image), and appearance-based ranking.

Our approach is loosely related to techniques for graph sparsification in which features are pruned a-posteriori from the SLAM factor graph to reduce computation; we refer the reader to the survey [34] for a review of these techniques.

The contributions that are most relevant to our proposal are the one of Davison [35], Lerner *et al.* [36], Mu *et al.* [37], Wu *et al.* [38], and Zhang and Vela [39]. The pioneering work of Davison [35] is one of the first papers to use information theoretic constructs to reason about visual features, and shares many of the motivations discussed in the present paper. Contrarily to the present paper, Davison [35] considers a model-based tracking problem in which the state of a moving camera has to be estimated from observations of known features. In hindsight, we also provide a theoretical justification for the use of a greedy algorithm (similar to the one used in [35]) which we prove able to compute near-optimal solutions. Lerner *et al.* [36] study landmark selection in a localization problem with known landmarks; the robot has to choose a subset of landmarks to observe so to minimize the localization uncertainty. The optimal subset selection is formulated as a mixed-integer program and relaxed to an SDP. While the problem we solve in this paper is different (visual inertial odometry vs. localization with known map), an interesting aspect of [36] is the use of a *requirement matrix* that weights the state covariance and encodes task-dependent uncertainty constraints. Mu *et al.* [37] propose a two-stage approach to select a subset of landmarks to minimize the probability of collision and a subset of measurements to accurately localize those landmarks. Our approach shares the philosophy of task-driven measurement selection, but has three key differences. First, we use a simplified linear model for forward dynamics simulation: this is in the spirit of RANSAC, in that a simplified algebraic model is used to quickly filter out less relevant data. Second, we consider different performance metrics, going beyond the determinant criterion used in [37] and related work on graph sparsification. Third, we perform feature selection in a single stage and leverage submodularity to provide formal performance guarantees. Wu *et al.* [38] consider a multi camera system and split the feature selection process into a cascade of two resource-allocation problems: (i) how to allocate resources among the cameras, and (ii) how to select features in each camera, according to the allocated resources. The former problem is solved by taking simpliying assumptions on the distribution of the features, the latter is based on the heuristic feature selection scheme of [40]. Our paper attempts to formalize feature selection by leveraging the notion of submodularity. Zhang and Vela [39] perform feature selection using an observability score and provide sub-optimality guarantees using submodularity. Our proposal is similar in spirit to [39] with few important differences. First, our approach is based on *anticipation*: the feature selector is aware of the intention (future motion) of the robot and selects the features accordingly. Second, we do not require the presence of known points (i.e., the anchors in [39]). Third, we consider a metric quantifying the motion estimation error and we purposely disregard the map reconstruction quality. Fourth, from the theoretical standpoint, we provide multiplicative suboptimality bounds, which are stronger than the additive bound of [39], and we prove that those bounds are nontrivial.

Sensor Scheduling and Submodularity. Feature selection is deeply related to the problem of sensor scheduling in control theory. The most common setup for sensor scheduling is the case in which N sensors monitor a phenomenon of interest and one has to choose κ out of the N available sensors to maximize some information-collection metric; this setup is also known as *sensor selection* or *sensor placement*. The literature on sensor selection includes approaches based on convex relaxation [41], Bayesian optimal design [42], and submodular optimization [43]. The problem is shown to be NP-hard in [44]. Shamaiah *et al.* [45] leverage submodularity and provide performance guarantees when optimizing the log-determinant of the estimation error covariance. A setup which is closer to the one in this paper is the case in which the sensed phenomenon is dynamic; in such case the sensor scheduling can be formulated in terms of the optimal selection of κ out of N possible measurements to be used in the update phase of a Kalman filter (KF). Vitus *et al.* [46] use a tree-search approach for sensor scheduling. Zhang *et al.* [47] proves that sensor scheduling within Kalman filtering is NP-hard and

shows that the trace of the steady state prior and posterior KF covariances are not submodular, despite the fact that greedy algorithms are observed to work well in practice. Jawaid and Smith [48] provide counterexamples showing that in general the maximum eigenvalue and the trace of the covariance are not submodular. Tzoumas *et al.* [49] generalize the derivation of [48] to prove submodularity of the logdet over a fixed time horizon, under certain assumptions on the observation matrix. Summers *et al.* [50] show that several metrics based on the controllability and observability Gramians are submodular.

Visual-Inertial Navigation. As the combined use of the visual and vestibular system is key to human navigation, recent advances in visual-inertial navigation on mobile robots are enabling unprecedented performance in pose estimation in GPS-denied environments using commodity hardware. The literature on visual-inertial navigation is vast, with many contributions proposed over the last two years, including approaches based on filtering [51], [40], [52], [53], [54], [55], fixed-lag smoothing [56], [57], [58], [59], and full smoothing [60], [61], [62], [63], [64], [65], [66]. We refer the reader to [66] for a comprehensive review.

Notation. We use lowercase and uppercase bold letters to denote vectors (e.g. $\boldsymbol{v}$) and matrices (e.g. $\boldsymbol{M}$), respectively. Sets are denoted by sans script fonts (e.g. A). Non-bold face letters are used for scalars and indices (e.g. j) and function names (e.g. $f(\cdot)$). The symbol $|\mathsf{A}|$ denotes the cardinality of A. The identity matrix of size n is denoted with $\mathbf{I}_n$. An $m \times n$ zero matrix is denoted by $\mathbf{0}_{m\times n}$. The symbol $\|\cdot\|$ denotes the Euclidean norm for vectors and the spectral norm for matrices.

III. Attention in Visual-Inertial Navigation

We design an attention mechanism that selects κ relevant visual features (e.g., Harris corners) from the current frame in order to maximize the performance of visual-inertial motion estimation. The κ features have to be selected out of N available features present in the camera image; the approach can deal with both monocular and stereo cameras (a stereo camera is treated as a rigid pair of monocular cameras).

We call F the set of all available features (with $|\mathsf{F}|= N$). If we denote with $f(\cdot)$ our task-dependent performance metric (we formalize a suitable metric for VIN in Section III-A), we can state our feature selection problem as follows:

$$\max_{\mathsf{S}\subset\mathsf{F}} \quad f(\mathsf{S}) \quad \text{subject to} \quad |\mathsf{S}|\leq \kappa \tag{1}$$

The problem looks for a subset of features S, containing no more than κ features, which optimizes the task performance $f(\cdot)$. This is a standard feature selection problem and has been used across multiple fields, including machine learning [67], robotics [37], and sensor networks [41]. Problem (1) is NP-hard [44] in general. In the rest of this paper we are interested in designing a suitable performance metric $f(\mathsf{S})$ for our VIN task, and provide fast approximation algorithms to solve (1).

We would like to design a performance metric $f(\cdot)$ that captures task-dependent requirements: in our case the metric has to quantify the uncertainty in the VIN motion estimation. Moreover, the metric should capture aspects already deemed relevant in related work. First, the metric has to reward the selection of the most distinctive features (in terms of appearance) since these are more likely to be re-observed in consecutive frames. Second, the metric has to reward features that remain within the field of view for a longer time. Therefore, *anticipation* is a key aspect: the metric has to be aware that under certain motion some of the features are more likely to remain in the field of view of the camera. Third, the metric has to reward features that are more informative to reduce uncertainty. In the following section we propose two performance metrics that seamlessly capture all these aspects.

A. Task-dependent Performance Metrics for VIN

Here we propose two metrics that quantify the accumulation of estimation errors over an horizon H, under the selection of a set of visual features S. Assume that k is the time instant at which the features need to be selected. Let us call $\hat{\boldsymbol{x}}_k$ the (to-be-computed) state estimate of the robot at time k: we will be more precise about the variables included in $\hat{\boldsymbol{x}}_k$ in Section III-B1; for now the reader can think that $\hat{\boldsymbol{x}}_k$ contains the estimate for the pose and velocity of the robot at time k, as well as the IMU biases. We denote with $\hat{\boldsymbol{x}}_{k:k+H} \doteq [\hat{\boldsymbol{x}}_k \ \hat{\boldsymbol{x}}_{k+1} \ \dots \ \hat{\boldsymbol{x}}_{k+H}]$ the future state estimates within the horizon H. Moreover, we call $\boldsymbol{P}_{k:k+H}$ the covariance matrix of our estimate $\hat{\boldsymbol{x}}_{k:k+H}$, and $\boldsymbol{\Omega}_{k:k+H} \doteq \boldsymbol{P}^{-1}_{k:k+H}$ the corresponding information matrix. Two natural metrics to capture the accuracy of $\hat{\boldsymbol{x}}_{k:k+H}$ are described in the following.

Worst-case Estimation Error. The worst-case error variance is quantified by the largest eigenvalue $\lambda_{\max}(\boldsymbol{P}_{k:k+H})$ of the covariance matrix $\boldsymbol{P}_{k:k+H}$, see e.g., [41]. Calling $\lambda_{\min}(\boldsymbol{\Omega}_{k:k+H})$ the smallest eigenvalue of the information matrix $\boldsymbol{\Omega}_{k:k+H}$, if follows that $\lambda_{\max}(\boldsymbol{P}_{k:k+H}) = 1/\lambda_{\min}(\boldsymbol{\Omega}_{k:k+H})$, hence minimizing the worst-case error is the same as maximizing $\lambda_{\min}(\boldsymbol{\Omega}_{k:k+H})$. Note that the information matrix $\boldsymbol{\Omega}_{k:k+H}$ is function of the selected set of measurements S, hence we write $\lambda_{\min}(\boldsymbol{\Omega}_{k:k+H}(\mathsf{S}))$.

Therefore our first metric (to be maximized) is:

$$f_\lambda(\mathsf{S}) = \lambda_{\min}(\boldsymbol{\Omega}_{k:k+H}(\mathsf{S})) = \lambda_{\min}\left(\bar{\boldsymbol{\Omega}}_{k:k+H} + \sum_{l\in\mathsf{S}} \Delta_l\right) \tag{2}$$

where on the right-hand-side, we exploited the additive structure of the information matrix, where $\bar{\boldsymbol{\Omega}}_{k:k+H}$ is the information matrix of the estimate when no features are selected (intuitively, this is the inverse of the covariance resulting from the IMU integration), while Δ_l is the information matrix associated with the selection of the l-th feature. We will give an explicit expression to $\bar{\boldsymbol{\Omega}}_{k:k+H}$ and Δ_l in Section III-B.

Volume and Mean Radius of the Confidence Ellipsoid. The ε-confidence ellipsoid is the ellipsoid that contains the estimation error with probability ε. Both the volume and the mean radius of the ε-confidence ellipsoid are proportional to the determinant of the covariance matrix. In particular, the volume V and the mean radius $\bar{R}$ of an n-dimensional ellipsoid associated with the covariance $\boldsymbol{P}_{k:k+H}$ can be written as [41]:

$$V = \frac{(\alpha\pi)^{n/2}}{\Gamma(\frac{n}{2}+1)} \det(\boldsymbol{P}^{\frac{1}{2}}_{k:k+H}) \quad , \quad \bar{R} = \sqrt{\alpha}\det(\boldsymbol{P}_{k:k+H})^{\frac{1}{2n}} \tag{3}$$

where α is the quantile of the χ^2 distribution with n degrees of freedom and upper tail probability of ε, $\Gamma(\cdot)$ is the Gamma function, and $\det(\cdot)$ is the determinant of a square matrix.

From (3) we note that to minimize the volume and the mean radius of the confidence ellipsoid we can equivalently minimize the determinant of the covariance. Moreover, since

$$\log\det(\boldsymbol{P}_{k:k+H}) = \log\det(\boldsymbol{\Omega}_{k:k+H}^{-1}) = -\log\det(\boldsymbol{\Omega}_{k:k+H})$$

then minimizing the size of the confidence ellipsoid is the same as maximizing the log-determinant of the information matrix, leading to our second performance metric:

$$f_{\det}(\mathsf{S}) = \log\det(\boldsymbol{\Omega}_{k:k+H}(\mathsf{S})) = \log\det\left(\bar{\boldsymbol{\Omega}}_{k:k+H} + \sum_{l\in\mathsf{S}}\Delta_l\right) \tag{4}$$

where we again noted that the information matrix is function of the selected features and can be written in additive form.

Probabilistic Feature Tracks. The performance metrics described so far already capture some important aspects: they are task-dependent in that they both quantify the motion estimation performance; moreover, they are predictive, in the sense that they look at the result of selecting a set of features over a short (future) horizon. As we will see in Section III-B2, the model also captures the fact that longer feature tracks are more informative, therefore it implicitly rewards the selection of features that are co-visible across multiple frames.

The only aspect that is not yet modeled is the fact that, even when a feature is in the field of view of the camera, there is some chance that it will not be correctly tracked and the corresponding feature track will be lost. For instance, if the appearance of a feature is not distinctive enough, the feature track may be shorter than expected.

To model the probability that a feature track is lost, we introduce N Bernoulli random variables $b_1, \dots, b_N$. Each variable b_l represents the outcome of the tracking of feature l: if $b_l = 1$, then the feature is successfully tracked, otherwise, the feature track is lost. For each feature we assume $p_l = \text{Prob}(b_l = 1)$ to be given; in practice one can correlate the appearance of each feature to p_l, such that more distinctive features have higher probability of being tracked if they are in the field of view. Using the binary variables $\boldsymbol{b} \doteq \{b_1, \dots, b_N\}$, we write the information matrix at the end of the horizon as:

$$\boldsymbol{\Omega}_{k:k+H}(\mathsf{S}, \boldsymbol{b}) = \bar{\boldsymbol{\Omega}}_{k:k+H} + \textstyle\sum_{l\in\mathsf{S}} b_l\Delta_l \tag{5}$$

which has a clear interpretation: if the l-th feature is correctly tracked, then $b_l = 1$ and the corresponding information matrix Δ_l is added to $\bar{\boldsymbol{\Omega}}_{k:k+H}$; on the other hand, if the feature tracks is lost, then $b_l = 0$ and the corresponding information content simply disappears from the sum in (5).

Since $\boldsymbol{b}$ is a random vector, our information matrix is now a stochastic quantity $\boldsymbol{\Omega}_{k:k+H}(\mathsf{S}, \boldsymbol{b})$, hence we have to redefine our performance metrics to include the expectation over $\boldsymbol{b}$:

$$f(\mathsf{S}) = \mathbb{E}\left[f(\boldsymbol{\Omega}_{k:k+H}(\mathsf{S}, \boldsymbol{b}))\right] \tag{6}$$

where the function $f(\cdot)$ denotes either $f_\lambda(\cdot)$ or $f_{\det}(\cdot)$.

Computing the expectation (6) leads to a sum with a combinatorial number of terms, which is hard to even evaluate. To avoid the combinatorial explosion, we use Jensen's inequality:

$$\mathbb{E}\left[f(\boldsymbol{\Omega}_{k:k+H}(\mathsf{S}, \boldsymbol{b}))\right] \geq f(\mathbb{E}\left[\boldsymbol{\Omega}_{k:k+H}(\mathsf{S}, \boldsymbol{b})\right]) \tag{7}$$

which produces a lower bound for our expected cost. In the rest of this paper we maximize this lower bound, rather that the original cost. The advantage of doing so is that the right-hand-side of (7) can be efficiently computed as:

$$f(\mathbb{E}\left[\boldsymbol{\Omega}_{k:k+H}(\mathsf{S}, \boldsymbol{b})\right] = f(\bar{\boldsymbol{\Omega}}_{k:k+H} + \textstyle\sum_{l\in\mathsf{S}} p_l\Delta_l) \tag{8}$$

where we used the definition (5), the fact that the expectation is a linear operator, and that $\mathbb{E}[b_l] = p_l$. Therefore, our performance metrics can be written explicitly as:

$$f_\lambda(\mathsf{S}) = \lambda_{\min}\left(\bar{\boldsymbol{\Omega}}_{k:k+H} + \sum_{l\in\mathsf{S}} p_l\Delta_l\right) \tag{9}$$

$$f_{\det}(\mathsf{S}) = \log\det\left(\bar{\boldsymbol{\Omega}}_{k:k+H} + \sum_{l\in\mathsf{S}} p_l\Delta_l\right)$$

which coincide with the deterministic counterparts (2), (4) when $p_l = 1, \forall l$. Interestingly, in (9) the probability that a feature is not tracked simply discounts the corresponding information content. Therefore, the approach considers features that are more likely to get lost as less informative, which is a desired behavior. While the derivation so far is quite general and provides a feature selection mechanism for any feature-based SLAM system, in the following we focus on visual-inertial navigation and we provide explicit expressions for the matrices $\bar{\boldsymbol{\Omega}}_{k:k+H}$ and Δ_l appearing in eq. (9).

B. Forward-simulation Model

The feature selection model proposed in Section III and the metrics in Section III-A require to predict the evolution of the information matrix over the horizon H. In the following we show how to forward-simulate the IMU and the camera; we note that we do not require to simulate actual IMU measurements, but only need to predict the corresponding information matrix, which depends on the IMU noise statistics.

The forward-simulation model depends on the future motion of the robot (the IMU and vision models are function of the future poses of the robot); therefore, anticipation is a key element of our approach: the feature selection mechanism is aware of the immediate-future intentions of the robot and selects features accordingly. As we will see in the experiments, this enables a more clever selection of features during sharp turns and aggressive maneuvers. In practice, the future poses along the horizon can be computed from the current control actions; for instance, if the controller is planning over a receding horizon, one can get the future poses by integrating the dynamics of the vehicle. In this sense, our attention mechanism involves a tight integration of control and perception.

The algorithms for feature selection that we present in Section IV are generic and work for any positive definite $\bar{\boldsymbol{\Omega}}_{k:k+H}$ and any positive semidefinite Δ_l. Therefore, the non-interested reader can safely skip this section, which provides explicit expressions for $\bar{\boldsymbol{\Omega}}_{k:k+H}$ and Δ_l in the visual-inertial setup.

Before delving into the details of the IMU and vision model we remark a key design goal of our forward-simulation model: efficiency. The goal of an attention mechanism is to reduce the cognitive load later on in the processing pipeline; therefore, by design, it should not be computational demanding, as that would defeat its purpose. For this reasons, in this section we present a simplified VIN model which is designed to be efficient to compute, while capturing all the aspects of interest of a full visual-inertial estimation pipeline, e.g., [66].

1) IMU Model: Our simplified IMU model is based on a single assumption: the accumulation of the rotation error due to gyroscope integration over the time horizon is negligible. In other words, the relative rotation estimates predicted by the gyroscope are accurate. This assumption is realistic, even for inexpensive IMUs: the drift in rotation integration is typically small and negligible over the time horizon considered in our attention system (in our tests we consider an horizon of 3s).

Assuming that the rotations are accurately known allows restricting the state to the robot position, linear velocity, and the accelerometer bias. Therefore, in the rest of this paper, the (unknown) state of the robot at time k is $\boldsymbol{x}_k \doteq [\boldsymbol{t}_k \ \boldsymbol{v}_k \ \boldsymbol{b}_k]$, where $\boldsymbol{t}_k \in \mathbb{R}^3$ is the 3D position of the robot, $\boldsymbol{v}_k \in \mathbb{R}^3$ is its velocity, and $\boldsymbol{b}_k$ is the (time-varying) accelerometer bias. We also use the symbol $\boldsymbol{R}_k$ to denote the attitude of the robot at time k: this is assumed to be known from gyroscope integration over the horizon H, hence it is not part of the state.

As in most VIN pipelines, we want to estimate the state of the robot at each frame[2]. Therefore, the goal of this subsection, similarly to [66], is to reformulate a set of IMU measurements between two consecutive frames k and j as a single measurement that constrains $\boldsymbol{x}_k$ and $\boldsymbol{x}_j$. Differently from [66], we show how to get a *linear* measurement model.

The on-board accelerometer measures the acceleration $\boldsymbol{a}_k$ of the sensor with respect to an inertial frame, and is affected by additive white noise $\boldsymbol{\eta}_k$ and a slowly varying sensor bias $\boldsymbol{b}_k$. Therefore, the measurement $\tilde{\boldsymbol{a}}_k \in \mathbb{R}^3$ acquired by the accelorometer at time k is modeled as [66]:

$$\tilde{\boldsymbol{a}}_k \quad = \quad \boldsymbol{R}_k^{\mathsf{T}} \left(\boldsymbol{a}_k - \mathbf{g} \right) + \boldsymbol{b}_k + \boldsymbol{\eta}_k, \tag{10}$$

where $\mathbf{g}$ is the gravity vector, expressed in the inertial frame. To keep notation simple, we omit the reference frames in our notation, which follow closely the convention used in [66]: position $\boldsymbol{t}_k$ and velocity $\boldsymbol{v}_k$ are expressed in the global frame, while the bias $\boldsymbol{b}_k$ is expressed in the sensor frame.

Given position $\boldsymbol{t}_k$ and velocity $\boldsymbol{v}_k$ at time k, we can forward-integrate and obtain $\boldsymbol{t}_j$ and $\boldsymbol{v}_j$ at time $j > k$:

$$\begin{aligned}
\boldsymbol{v}_j \quad &= \quad \boldsymbol{v}_k + \textstyle\sum_{i=k}^{j-1} \boldsymbol{a}_i \delta \\
&\quad \text{(from (10) we know } \boldsymbol{a}_i = \mathbf{g} + \boldsymbol{R}_i(\tilde{\boldsymbol{a}}_i - \boldsymbol{b}_i - \boldsymbol{\eta}_i)\text{,} \\
&\quad \text{and assuming constant bias between frames, } \boldsymbol{b}_i = \boldsymbol{b}_k\text{)} \\
&= \quad \boldsymbol{v}_k + \mathbf{g}\delta_{kj} + \textstyle\sum_{i=k}^{j-1} \boldsymbol{R}_k \left(\tilde{\boldsymbol{a}}_i - \boldsymbol{b}_k - \boldsymbol{\eta}_i \right) \delta
\end{aligned} \tag{11}$$

[2] The derivation is identical for the case in which we associate a state to each keyframe, rather than each frame, as done in related work [66].

$$\begin{aligned}
\boldsymbol{t}_j \quad &= \quad \boldsymbol{t}_k + \textstyle\sum_{i=k}^{j-1} \left(\boldsymbol{v}_i \delta + \frac{1}{2} \boldsymbol{a}_i \delta^2 \right) \\
&\quad \text{(substituting } \boldsymbol{a}_i = \mathbf{g} + \boldsymbol{R}_i(\tilde{\boldsymbol{a}}_i - \boldsymbol{b}_k - \boldsymbol{\eta}_i)\text{)} \\
&= \quad \boldsymbol{t}_k + \textstyle\sum_{i=k}^{j-1} (\boldsymbol{v}_i \delta + \frac{1}{2} \mathbf{g} \delta^2 + \frac{1}{2} \boldsymbol{R}_k \left(\tilde{\boldsymbol{a}}_i - \boldsymbol{b}_k - \boldsymbol{\eta}_i \right) \delta^2) \\
&\quad \text{(substituting } \boldsymbol{v}_j \text{ from (11) with } j = i\text{)} \\
&= \quad \boldsymbol{t}_k + \tfrac{1}{2} \mathbf{g} \hat{\delta}_{kj}^2 + \textstyle\sum_{i=k}^{j-1} \frac{1}{2} \boldsymbol{R}_i \left(\tilde{\boldsymbol{a}}_i - \boldsymbol{b}_k - \boldsymbol{\eta}_i \right) \delta^2 \\
&+ \quad \textstyle\sum_{i=k}^{j-1} (\boldsymbol{v}_k + \mathbf{g}\delta_{ki} + \sum_{h=k}^{i-1} \boldsymbol{R}_h \left(\tilde{\boldsymbol{a}}_h - \boldsymbol{b}_k - \boldsymbol{\eta}_h \right) \delta)
\end{aligned} \tag{12}$$

where δ is the sampling time of the IMU, $\delta_{kj} \doteq \sum_{i=k}^{j-1} \delta$, and $\hat{\delta}_{kj}^2 \doteq \sum_{i=k}^{j-1} \delta^2$; as in [66], we assumed that the IMU bias remains constant between two frames. The evolution of the bias across frames can be modeled as a random walk:

$$\boldsymbol{b}_j \quad = \quad \boldsymbol{b}_k - \boldsymbol{\eta}_{kj}^{\mathbf{b}} \tag{13}$$

where $\boldsymbol{\eta}_{kj}^{\mathbf{b}}$ is a zero-mean random vector.

Noting that the state appears linearly in (11)-(13), it is easy to rewrite the three expressions together in matrix form:

$$\boldsymbol{z}_{kj}^{\text{IMU}} = \boldsymbol{A}_{kj} \boldsymbol{x}_{k:k+H} + \boldsymbol{\eta}_{kj}^{\text{IMU}} \tag{14}$$

where $\boldsymbol{z}_{kj}^{\text{IMU}} \in \mathbb{R}^9$ is a suitable vector, and $\boldsymbol{\epsilon}_{kj} \in \mathbb{R}^9$ is zero-mean random noise. We remark that while $\boldsymbol{z}_{kj}^{\text{IMU}}$ is function of the future IMU measurements, this vector is not actually used in our approach (what matters is $\boldsymbol{A}_{kj}$ and the information matrix of $\boldsymbol{\eta}_{kj}^{\text{IMU}}$), hence we do not need to simulate future measurements. An explicit expression for the matrix $\boldsymbol{A}_{kj} \in \mathbb{R}^{9 \times 9H}$, the vector $\boldsymbol{z}_{kj}^{\text{IMU}}$, and the covariance of $\boldsymbol{\eta}_{kj}^{\text{IMU}}$ is given in Appendix. The matrix $\boldsymbol{A}_{kj}$ is a sparse block matrix with 9×9 blocks, which is all zeros, except the blocks corresponding to the state at times k and j.

From linear estimation theory, we know that, given the IMU measurements (14) for all consecutive frames k, j in the horizon H, the information matrix of the optimal estimate of the state $\boldsymbol{x}_{k:k+H}$ is:

$$\bar{\boldsymbol{\Omega}}_{k:k+H} = \sum_{kj \in \mathsf{H}} (\boldsymbol{A}_{kj}^{\mathsf{T}} \boldsymbol{\Omega}_{kj}^{\text{IMU}} \boldsymbol{A}_{kj}) \tag{15}$$

where H is the set of consecutive frames within the time horizon H, and $\boldsymbol{\Omega}_{kj}^{\text{IMU}} \in \mathbb{R}^{9 \times 9}$ is the information matrix of the noise vector $\boldsymbol{\eta}_{kj}^{\text{IMU}}$. The matrix $\bar{\boldsymbol{\Omega}}_{k:k+H}$ is precisely the information matrix of the state estimate before any vision measurement is selected, that we already introduced in (5).

2) Vision Model: Also for the vision measurements, we are interested in designing a linear measurement model, which simplifies the actual (nonlinear) perspective projection model. To do so, we have to express a pixel measurement as a linear function of the unknown state that we want to estimate.

A (calibrated) pixel measurement of an external 3D point (or landmark) l identifies the 3D bearing of the landmark in the camera frame. Mathematically, if we call $\boldsymbol{u}_{kl}$ the unit vector, corresponding to the (calibrated) pixel observation of l from the robot pose at time k, $\boldsymbol{u}_{kl}$ satisfies the following relation:

$$\boldsymbol{u}_{kl} \times \left((\boldsymbol{R}_{\text{cam},k}^{\text{w}})^{\mathsf{T}} (\boldsymbol{p}_l - \boldsymbol{t}_{\text{cam},k}^{\text{w}}) \right) = \mathbf{0}_3 \tag{16}$$

where $\times$ is the cross product between two vectors, $\boldsymbol{p}_l$ is the 3D position of landmark l (in the world frame), $\boldsymbol{R}_{\text{cam},k}^{\text{w}}$ and $\boldsymbol{t}_{\text{cam},k}^{\text{w}}$ are the rotation and translation describing the camera pose

at time k (w.r.t. the world frame). In words, the model (16) requires the observed point (transformed to the camera frame) to be collinear to the measured direction $\boldsymbol{u}_{kl}$, since the cross product measures the deviation from collinearity [68].

Now we note that for two vectors $\boldsymbol{v}_1$ and $\boldsymbol{v}_2$, the cross product $\boldsymbol{v}_1 \times \boldsymbol{v}_2 = [\boldsymbol{v}_1]_\times \boldsymbol{v}_2$, where $[\boldsymbol{v}_1]_\times$ is the skew symmetric matrix built from $\boldsymbol{v}_1$. Moreover, we note that the camera pose w.r.t. the world frame, $(\boldsymbol{R}^{\text{w}}_{\text{cam},k}, \boldsymbol{t}^{\text{w}}_{\text{cam},k})$, can be written as the composition of the IMU pose w.r.t. the world frame, $(\boldsymbol{R}_k, \boldsymbol{t}_k)$, and the relative pose of the camera w.r.t. the IMU, $(\boldsymbol{R}^{\text{IMU}}_{\text{cam}}, \boldsymbol{t}^{\text{IMU}}_{\text{cam}})$ (known from calibration). Using these two considerations, we rewrite (16) equivalently as:

$$[\boldsymbol{u}_{kl}]_\times \left((\boldsymbol{R}_k \boldsymbol{R}^{\text{IMU}}_{\text{cam}})^\mathsf{T} (\boldsymbol{p}_l - (\boldsymbol{t}_k + \boldsymbol{R}_k \boldsymbol{t}^{\text{IMU}}_{\text{cam}})) \right) = \boldsymbol{0}_3 \tag{17}$$

In presence of measurement noise, (17) becomes:

$$[\boldsymbol{u}_{kl}]_\times \left((\boldsymbol{R}_k \boldsymbol{R}^{\text{IMU}}_{\text{cam}})^\mathsf{T} (\boldsymbol{p}_l - (\boldsymbol{t}_k + \boldsymbol{R}_k \boldsymbol{t}^{\text{IMU}}_{\text{cam}})) \right) = \boldsymbol{\eta}^{\text{cam}}_{kl} \tag{18}$$

where $\boldsymbol{\eta}^{\text{cam}}_{kl}$ is a zero-mean random noise with known covariance. Under the assumptions that rotations are known from gyroscope integration, the unknowns in model (18) are the robot position $\boldsymbol{t}_k$ (which is part of our state vector $\boldsymbol{x}_{k:k+H}$) and the position of the observed 3D landmark $\boldsymbol{p}_l$. The model is linear and can be written in matrix form as:

$$\boldsymbol{z}^{\text{cam}}_{kl} = \boldsymbol{F}_{kl} \boldsymbol{x}_{k:k+H} + \boldsymbol{E}_{kl} \boldsymbol{p}_l + \boldsymbol{\eta}^{\text{cam}}_{kl} \tag{19}$$

for a suitable vector $\boldsymbol{z}^{\text{cam}}_{kl}$, and matrices $\boldsymbol{F}_{kl}$ and $\boldsymbol{E}_{kl}$. In order to be triangulated, a point has to be observed across multiple frames. Stacking the linear system (19) for each observation pose from which l is visible, we get a single linear system:

$$\boldsymbol{z}^{\text{cam}}_{l} = \boldsymbol{F}_{l} \boldsymbol{x}_{k:k+H} + \boldsymbol{E}_{l} \boldsymbol{p}_l + \boldsymbol{\eta}^{\text{cam}}_{l} \tag{20}$$

where $\boldsymbol{z}^{\text{cam}}_{l}$, $\boldsymbol{F}_l$, and $\boldsymbol{E}_l$ are obtained by stacking (row-wise) $\boldsymbol{z}^{\text{cam}}_{kl}$, $\boldsymbol{F}_{kl}$, and $\boldsymbol{E}_{kl}$, respectively, for all frames $k : k+H$. As for the IMU model, the expression of $\boldsymbol{z}^{\text{cam}}_{l}$ is inconsequential for our derivation, as it does not influence the future state covariance. On the other hand $\boldsymbol{F}_l$ and $\boldsymbol{E}_l$ depend on the future measurements $\boldsymbol{u}_{kl}$: for this reason, computing these matrices requires simulating pixel projections of $\boldsymbol{p}_l$ for each frame in the horizon. When using a stereo camera, we have an estimate of $\boldsymbol{p}_l$ hence we can easily project it to the future frames. In a monocular setup, we can guess the depth of new features from the existing features in the VIN back-end.

Now we note that we cannot directly use the linear model (20) to estimate our state vector $\boldsymbol{x}_{k:k+H}$, since it contains the unknown position of landmark l. One way to circumvent this problem is to include the 3D point in the state vector. This is undesirable for two reasons; first, including the landmarks as part of the state would largely increases the dimension of the state space (and hence of the matrices in (9)). Second, it may create undesirable behaviors of our performance metrics; for instance, the metrics might induce to select features that minimize the uncertainty of a far 3D point rather than focusing on the variables we are actually interested in (i.e., the state of the robot).

To avoid this undesirable effects, we analytically eliminate the 3D point from the estimation using the Schur complement trick [69]. We first write the information matrix of the joint state $[\boldsymbol{x}_{k:k+H}\ \boldsymbol{p}_l]$ from the linear measurements (20):

$$\boldsymbol{\Omega}^{(l)}_{k:k+H} = \begin{bmatrix} \boldsymbol{F}_l^\mathsf{T} \boldsymbol{F}_l & \boldsymbol{F}_l^\mathsf{T} \boldsymbol{E}_l \\ \boldsymbol{E}_l^\mathsf{T} \boldsymbol{F}_l & \boldsymbol{E}_l^\mathsf{T} \boldsymbol{E}_l \end{bmatrix} \tag{21}$$

Using the Schur complement trick we marginalize out the landmark l and obtain the information matrix of our state $\boldsymbol{x}_{k:k+H}$ given the measurements (20):

$$\Delta_l = \boldsymbol{F}_l^\mathsf{T} \boldsymbol{F}_l - \boldsymbol{F}_l^\mathsf{T} \boldsymbol{E}_l (\boldsymbol{E}_l^\mathsf{T} \boldsymbol{E}_l)^{-1} \boldsymbol{E}_l^\mathsf{T} \boldsymbol{F}_l \tag{22}$$

Eq. (22) is the (additive) contribution to the information matrix of our state estimate due to the measurements of a single landmark l. This is the matrix that we already called Δ_l in (5). The matrix Δ_l is sparse, and its sparsity pattern is dictated by the co-visibility of landmark l across different frames [70]. It is worth noticing that $(\boldsymbol{E}_l^\mathsf{T} \boldsymbol{E}_l)^{-1}$ is the covariance of the estimate of the landmark position [70], and it is invertible as long as the landmark l can be triangulated.

Remark 1 (Linear measurement models). *Sections III-B1 and III-B2 provide linear measurement models for inertial and visual measurements. Within our framework, one might directly use linearized models of the nonlinear inertial and perspective models commonly used in VIN [66]. Our choice to design linear models has three motivations. First, we operate over a smaller state space (which does not include rotations and gyroscope biases), hence making matrix manipulations faster. Second, we avoid the actual computational cost of linearizing the nonlinear models. Third, thanks to the simplicity of the models, we enable a geometric understanding of our feature selection mechanisms (Section IV-D).*

IV. Attention Allocation: Algorithms and Performance Guarantees

In this section we discuss computational approaches to find a set of features that approximately solves the feature selection problem (1). It is known that finding the optimal subset $\mathsf{S}^\star$ which solves (1) exactly is NP-hard [44], hence we cannot hope to find efficient algorithms to compute $\mathsf{S}^\star$ in real-world problems.[3] The solution we adopt in this paper is to design *approximation algorithms*, which are computationally efficient and provide performance guarantees (roughly speaking, produce a set which is not far from the optimal subset $\mathsf{S}^\star$). We remark that we are designing a *covert attention mechanism*: our algorithms only select a set of features that have to be retained and used for state estimation, while we do not attempt to actively control the motion of the camera.

In the following we present two classes of algorithms. The former, discussed in Section IV-A, is based on a convex relaxation of the original combinatorial problem (1). The second, discussed in Section IV-C, is a simple greedy selection.

[3] In typical real-world problems, the set of available visual feature is larger than 200, and we are asked to select $10-100$ features, depending on on-board resources. In those instances, the cost of a brute force search is prohibitive.

A. Convex Relaxations

This section presents a convex-relaxation approach to compute an approximate solution for problem (1).

Using (9), we rewrite problem (1) explicitly as:

$$\max_{\mathsf{S}\subset\mathsf{F}} \quad f\left(\bar{\mathbf{\Omega}}_{k:k+H} + \textstyle\sum_{l\in\mathsf{S}} p_l \Delta_l\right) \quad \text{subject to } |\mathsf{S}| \leq \kappa \quad (23)$$

where $f(\cdot)$ denotes either $f_\lambda(\cdot)$ or $f_{\rm det}(\cdot)$ (for the moment there is not need to distinguish the two metrics).

Introducing binary variables s_l, for $l = 1, \ldots, N$, we rewrite (23) equivalently as:

$$\begin{aligned} \max_{s_1,\ldots,s_N} \quad & f\left(\bar{\mathbf{\Omega}}_{k:k+H} + \textstyle\sum_{l\in\mathsf{S}} s_l p_l \Delta_l\right) \quad (24) \\ \text{subject to} \quad & \textstyle\sum_{l=1}^N s_l \leq \kappa \;\;,\;\; s_l \in \{0,1\}\; \forall\, l \in \{1,\ldots,N\} \end{aligned}$$

Problem (24) is a binary optimization problem. While problem (24) would return the optimal subset $\mathsf{S}^\star$, it is still NP-hard to solve, due to the constraint that s_l have to be binary.

Problem (24) admits an simple convex relaxation:

$$\begin{aligned} f^\star_{\mathsf{cvx}} = \max_{s_1,\ldots,s_N} \quad & f\left(\bar{\mathbf{\Omega}}_{k:k+H} + \textstyle\sum_{l\in\mathsf{S}} s_l p_l \Delta_l\right) \quad (25) \\ \text{subject to} \quad & \textstyle\sum_{l=1}^N s_l \leq \kappa \;\;,\;\; s_l \in [0,1]\; \forall\, l \in \{1,\ldots,N\} \end{aligned}$$

where the binary constraint $s_l \in \{0,1\}$ is replaced by the convex constraint $s_l \in [0,1]$. Convexity of problem (25) follows from the fact that we maximize a concave cost under linear inequality constraints.[4]

This convex relaxation has been proposed multiple times in other contexts (see, e.g., [41]). The solution $s^\star_1, \ldots, s^\star_N$ of (25) is not binary in general and a rounding procedure is needed to distinguish the features that have to be discarded ($s_l = 0$) from the features that have to be selected ($s_l = 1$). A common rounding procedure is to simply select the κ features with the largest s_l, while randomized rounding procedures have also been considered [36]. We use the former, and we call S° the set including the indices of the κ features with the largest $s^\star_l$, where $s^\star_1, \ldots, s^\star_N$ is the optimal solution of (25).

B. Performance Guarantees for the Convex Relaxations

The convex relaxation (25) has been observed to work well in practice, although there is no clear (a-priori) performance guarantee on the quality of the set S°.

Let us call $f^\star_{\mathsf{cvx}}$ the optimal objective of the relaxed problem (25), $f(\mathsf{S}^\circ)$ the objective attained by the rounded solution, and $f(\mathsf{S}^\star)$ the optimal solution of the original NP-hard problem (24). Then, one can easily obtain a-posteriori performance bounds by observing that:

$$f(\mathsf{S}^\circ) \leq f(\mathsf{S}^\star) \leq f^\star_{\mathsf{cvx}} \quad (26)$$

where the first inequality follows from optimality of $\mathsf{S}^\star$ (any subset of κ features has cost at most $f(\mathsf{S}^\star)$), while the latter from the fact that (25) is a relaxation of the original problem.

[4]Both the smallest eigenvalue and the log-determinant of a positive definite matrix are concave functions [71] of the matrix entries.

The chain of inequality (26) suggests a simple (a-posteriori) performance bound for the quality of the set produced by the convex relaxation (25):

$$f(\mathsf{S}^\star) - f(\mathsf{S}^\circ) \leq f^\star_{\mathsf{cvx}} - f(\mathsf{S}^\circ) \quad (27)$$

i.e., the suboptimality gap $f(\mathsf{S}^\star) - f(\mathsf{S}^\circ)$ of the subset S° is bounded by the difference $f^\star_{\mathsf{cvx}} - f(\mathsf{S}^\circ)$, which can be computed (a posteriori) after solving (25).

C. Greedy Algorithms and Lazy Evaluation

This section presents a second approach to approximately solve problem (1). Contrarily to the convex relaxation of Section IV-A, here we consider a greedy algorithm that selects κ features that (approximately) maximize the cost $f(\cdot)$.

The algorithm starts with an empty set $\mathsf{S}^\#$ and performs κ iterations. At each iteration, it adds the feature that, if added to $\mathsf{S}^\#$, induces the largest increase in the cost function. The pseudocode of the algorithm is given in Algorithm 1.

Algorithm 1: Greedy algorithm with lazy evaluation

```
1  Input: Ω̄_{k:k+H}, Δ_l, for l = 1,...,N, and κ ;
2  Output: feature subset S# ;
3  S# = ∅ ;
4  for i = 1,...,κ do
5      % Compute upper bound for f(S# ∪ l), l = 1,...,N
6      [U_1,...,U_N] = upperBounds(Ω̄_{k:k+H}, Δ_1,...,Δ_N) ;
7      % Sort features using upper bound
8      F↓ = sort(U_1,...,U_N) ;
9      % Initialize best feature
10     f_max = −1  ;  l_max = −1 ;
11     for l ∈ F↓ do
12         if U_l < f_max then
13             break ;
14         end
15         if f(S# ∪ l) > f_max then
16             f_max = f(S# ∪ l)  ;  l_max = l ;
17         end
18     end
19     S# = S# ∪ l_max ;
20 end
```

In line 3 the algorithm starts with an empty set. The "for" loop in line 4 iterates κ times: at each time the best feature is added to the subset $\mathsf{S}^\#$ (line 19). The role of the "for" loop in line 11 is to compute the feature that induces the maximum increase in the cost (lines 15-17). The remaining lines provide a lazy evaluation mechanism. For each feature l we compute an upper bound on the cost $f(\mathsf{S}^\# \cup l)$ (line 6). The features are sorted (in descending order) according to this upper bound (line 8). The advantage of this is that by comparing the current best feature with this upper bound (line 12) we can avoid checking features that are guaranteed to attain a smaller cost.

Clearly, the lazy evaluation is advantageous if the upper bound is faster to compute than the actual cost. The following

propositions provide two useful (and computationally cheap) upper bounds for our cost functions.

Proposition 2 (Upper bounds for $\log\det$: Hadamard's inequality, Thm 7.8.1 [72]). *For a positive definite matrix $\boldsymbol{M} \in \mathbb{R}^{n\times n}$ with diagonal elements $\boldsymbol{M}_{ii}$, it holds:*

$$\det(\boldsymbol{M}) \leq \prod_{i=1}^{n} \boldsymbol{M}_{ii} \Leftrightarrow \log\det(\boldsymbol{M}) \leq \sum_{i=1}^{m} \log \boldsymbol{M}_{ii} \quad (28)$$

Proposition 3 (Eigenvalue Perturbation Bound [73]). *Given Hermitian matrices $\boldsymbol{M}, \Delta \in \mathbb{R}^{n\times n}$, and denoting with $\lambda_i(\boldsymbol{M})$ the i-th eigenvalue of $\boldsymbol{M}$, the following inequalities hold:*

$$|\lambda_i(\boldsymbol{M}+\Delta) - \lambda_i(\boldsymbol{M})| \leq \|\Delta\| \quad (29)$$
$$\min_j |\lambda_i(\boldsymbol{M}) - \lambda_j(\boldsymbol{M}+\Delta)| \leq \|\Delta \boldsymbol{v}_i\| \quad (30)$$

where $\boldsymbol{v}_i$ is the eigenvector of $\boldsymbol{M}$ associated to $\lambda_i(\boldsymbol{M})$.

Eq. (29) is a restatement of the classical Weyl inequality, while (30) is a tighter bound from Ipsen and Nadler [73]. To clarify how the bounds in Proposition 3 provide us with an upper bound for $\lambda_{\min}$, we prove the following result.

Corollary 4 (Upper bounds for $\lambda_{\min}$). *Given two symmetric and positive semidefinite matrices $\boldsymbol{M}, \Delta \in \mathbb{R}^{n\times n}$ the following inequality holds:*

$$\lambda_{\min}(\boldsymbol{M}+\Delta) \leq \lambda_{\min}(\boldsymbol{M}) + \|\Delta \boldsymbol{v}_{\min}\| \quad (31)$$

where $\boldsymbol{v}_{\min}$ is the eigenvector of $\boldsymbol{M}$ associated to the smallest eigenvalue $\lambda_{\min}(\boldsymbol{M})$.

Proof: The proof relies on the inequality (30) for i chosen to be the smallest eigenvalue. From the Weyl inequality [73], it follows $\lambda_j(\boldsymbol{M}+\Delta) \geq \lambda_{\min}(\boldsymbol{M})$, for all j. Using this fact, it follows that the minimum in (30) is attained by $\lambda_{\min}(\boldsymbol{M}+\Delta)$. Therefore, the inequality (30) becomes:

$$|\lambda_{\min}(\boldsymbol{M}) - \lambda_{\min}(\boldsymbol{M}+\Delta)| \leq \|\Delta \boldsymbol{v}_{\min}\| \quad (32)$$

From the positive definiteness of $\boldsymbol{M}$ and Δ (which implies $\lambda_{\min}(\boldsymbol{M}) \geq 0$ and $\lambda_{\min}(\boldsymbol{M}+\Delta) \geq 0$), and from the Weyl inequality, it follows $|\lambda_{\min}(\boldsymbol{M}) - \lambda_{\min}(\boldsymbol{M}+\Delta)| = \lambda_{\min}(\boldsymbol{M}+\Delta) - \lambda_{\min}(\boldsymbol{M})$, which substituted in (32) leads to (31). ∎

While Algorithm 1 highlights the simplicity of the greedy algorithm, it is unclear whether this algorithm produces good subsets of features. We tackle this question in the next section.

D. Performance Guarantees for the Greedy Algorithm

This section shows that the greedy algorithm (Algorithm 1) admits provable sub-optimality bounds. These bounds guarantee that the greedy selection cannot perform much worse than the optimal strategy. The section tackles separately the two metrics presented in Section III-A, since the corresponding performance guarantees are fundamentally different.

Our results are based on the recent literature on submodularity and submodular maximization. Before delving in the guarantees for each metric, we provide few preliminary definitions, which can be safely skipped by the expert reader.

Definition 5 (Normalized and Monotone Set Function [74]). *A set function $f : 2^{\mathcal{F}} \to \mathbb{R}$ is said to be* normalized *if $f(\emptyset) = 0$; $f(\mathcal{S})$ is said to be* monotone *(non-decreasing) if for any subsets $\mathcal{A} \subseteq \mathcal{B} \subseteq \mathcal{F}$, it holds $f(\mathcal{A}) \leq f(\mathcal{B})$.*

Definition 6 (Submodularity [74]). *A set function $f : 2^{\mathcal{F}} \to \mathbb{R}$ is* submodular *if, for any subsets $\mathcal{A} \subseteq \mathcal{B} \subseteq \mathcal{F}$, and for any element $e \in \mathcal{F} \setminus \mathcal{B}$, it holds that:*

$$f(\mathcal{A}\cup\{e\}) - f(\mathcal{A}) \geq f(\mathcal{B}\cup\{e\}) - f(\mathcal{B}) \quad (33)$$

Submodularity formalizes the notion of diminishing returns: adding a measurement to a small set of measurement is more advantageous than adding it to a large set. Our interest towards submodularity is motivated by the following result.

Proposition 7 (Near-optimal submodular maximization [74]). *Given a normalized, monotone, submodular set function $f : 2^{\mathcal{F}} \to \mathbb{R}$, and calling $\mathcal{S}^\star$ the optimal solution of the maximization problem* (1)*, then the set $\mathcal{S}^{\#}$, computed by the greedy Algorithm 1, is such that:*

$$f(\mathcal{S}^{\#}) \geq (1-1/\mathrm{e}) f(\mathcal{S}^\star) \approx 0.63 f(\mathcal{S}^\star) \quad (34)$$

This bound ensures us that the worst-case performance of the greedy algorithm cannot be far from the optimum. In the following we tailor this result to our feature selection problem.

1) Sub-optimality guarantees for $\log\det$: It is possible to show that $\log\det$ is submodular with respect to the set of measurements used for estimation. This result and the corresponding performance guarantees are formalized as follows.

Proposition 8 (Submodularity of $\log\det$ [45]). *The set function $f_{\det}(\mathcal{S})$ defined in* (4) *is monotone and submodular. Moreover, the greedy algorithm applied to* (1) *using $f_{\det}(\mathcal{S})$ as objective enjoys the following performance guarantees:*

$$f_{\det}(\mathcal{S}) \geq (1-1/\mathrm{e}) f_{\det}(\mathcal{S}^\star) + \frac{f_{\det}(\emptyset)}{\mathrm{e}} \quad (35)$$

The result is proven in [45] and has been later rectified to account for the need of normalized functions in [48]. The extra term $\frac{f_{\det}(\emptyset)}{e}$ in (35) indeed follows from the application of Proposition 7 to the normalized function $f_{\det}(\mathsf{S}) - f_{\det}(\emptyset)$.

2) Sub-optimality guarantees for $\lambda_{\min}$: Currently, no result is readily available to bound the suboptimality gap of the greedy algorithm applied to the maximization of the smallest eigenvalue of the information matrix (or equivalently minimizing the largest eigenvalue of the covariance). Indeed, related work provides counterexamples, showing that this metric is not submodular in general, while the greedy algorithm is observed to perform well in practice [48]. In this section we provide a first result showing that, despite the fact that $f_\lambda(\mathsf{S})$ fails to be submodular, it is not far from a submodular function. This notion is made more precise in the following.

Definition 9 (Submodularity ratio [67], [75]). *The* submodularity ratio *of a non-negative set function $f(\cdot)$ with respect to a set $\mathcal{S}$ and an integer $\kappa \geq 1$ is defined as:*

$$\gamma_{\mathcal{S}} \doteq \min_{\substack{\mathcal{L}\subseteq\mathcal{S},\\ \mathcal{E}:|\mathcal{E}|\leq\kappa, \mathcal{E}\cap\mathcal{L}=\emptyset}} \frac{\sum_{e\in\mathcal{E}} (f(\mathcal{L}\cup\{e\}) - f(\mathcal{L}))}{f(\mathcal{L}\cup\mathcal{E}) - f(\mathcal{L})} \quad (36)$$

It is possible to show that if $\gamma_{\mathsf{S}} \geq 1$, then the function $f(\cdot)$ is submodular. However, in this context we are interested in

the submodularity ratio, since it enables a less restrictive suboptimality bound, as described in the following proposition.

Proposition 10 (Approximate submodular maximization [67], [75]). *Let $f(\cdot)$ be a non-negative monotone set function and let $\mathcal{S}^\star$ be the optimal solution of the maximization problem* (1)*, then the set $\mathcal{S}^\#$, computed by the greedy Algorithm 1 is such that:*

$$f(\mathcal{S}^\#) \geq (1 - \mathrm{e}^{-\gamma_{\mathcal{S}^\#}}) f(\mathcal{S}^\star) \tag{37}$$

where $\gamma_{\mathcal{S}^\#}$ is the submodularity ratio of $f(\cdot)$ with respect to $\mathcal{S}^\#$ and $\kappa = |\mathcal{S}^\#|$.

Proposition 10 provides a multiplicative suboptimality bound whenever $\gamma_{\mathcal{S}^\#} > 0$. In the following we show that this is generally the case when maximizing the smallest eigenvalue.

Proposition 11 (Non-vanishing Submodularity ratio of $\lambda_{\min}$). *Call $\mathcal{S}^\#$ the set returned by the greedy algorithm maximizing $\lambda_{\min}$. For any set $\mathsf{L} \subseteq \mathcal{S}^\#$ call $\bar{\boldsymbol{\mu}}$ the eigenvector corresponding to the smallest eigenvalue of the matrix $\bar{\boldsymbol{\Omega}}_{k:k+H} + \sum_{l \in \mathsf{L}} \Delta_l$. Moreover call $\bar{\boldsymbol{\mu}}_0, \bar{\boldsymbol{\mu}}_2, \ldots, \bar{\boldsymbol{\mu}}_H \in \mathbb{R}^3$, the subvectors of $\bar{\boldsymbol{\mu}}$ corresponding the robot positions at time $k, \ldots, k+H$. Then the submodularity ratio of $\lambda_{\min}$ is bounded away from zero if $\bar{\boldsymbol{\mu}}_i \neq \bar{\boldsymbol{\mu}}_j$, for some i, j.*

Proof: In order to show that the submodularity ratio (36) does not vanish, we show that its numerator is bounded away from zero. To do so, we consider a single summand in (36):

$$(a) \doteq f(\mathsf{L} \cup \{e\}) - f(\mathsf{L}) = \lambda_{\min}(\boldsymbol{\Omega}_{\mathsf{L}} + \Delta_e) - \lambda_{\min}(\boldsymbol{\Omega}_{\mathsf{L}}) \tag{38}$$

where $\boldsymbol{\Omega}_{\mathsf{L}} \doteq \bar{\boldsymbol{\Omega}}_{k:k+H} + \sum_{l \in \mathsf{L}} \Delta_l$. Our task is to prove that (38) is different from zero. To do so, we substitute the eigenvalue with its definition through the Rayleigh quotient:

$$\begin{aligned}
&(a) = \min_{\|\boldsymbol{\mu}\|=1} \boldsymbol{\mu}^\mathsf{T}(\boldsymbol{\Omega}_{\mathsf{L}} + \Delta_e)\boldsymbol{\mu} - \min_{\|\boldsymbol{\nu}\|=1} \boldsymbol{\nu}^\mathsf{T}(\Delta_e)\boldsymbol{\nu} \\
&\text{(calling } \bar{\boldsymbol{\mu}} \text{ the minimizer of the first summand)} \\
&= \bar{\boldsymbol{\mu}}^\mathsf{T}(\boldsymbol{\Omega}_{\mathsf{L}} + \Delta_e)\bar{\boldsymbol{\mu}} - \min_{\|\boldsymbol{\nu}\|=1} \boldsymbol{\nu}^\mathsf{T}(\boldsymbol{\Omega}_{\mathsf{L}})\boldsymbol{\nu} \\
&\text{(using a suboptimal solution } \boldsymbol{\nu} = \bar{\boldsymbol{\mu}} \text{ in the second summand)} \\
&\geq \bar{\boldsymbol{\mu}}^\mathsf{T}(\boldsymbol{\Omega}_{\mathsf{L}} + \Delta_e)\bar{\boldsymbol{\mu}} - \bar{\boldsymbol{\mu}}^\mathsf{T}(\boldsymbol{\Omega}_{\mathsf{L}})\bar{\boldsymbol{\mu}} \\
&\text{(simplifying and substituting the expression of } \Delta_e \text{ from (22))} \\
&= \bar{\boldsymbol{\mu}}^\mathsf{T}\Delta_e\bar{\boldsymbol{\mu}} = \bar{\boldsymbol{\mu}}^\mathsf{T}\boldsymbol{F}_e^\mathsf{T}(\mathbf{I} - \boldsymbol{E}_e(\boldsymbol{E}_e^\mathsf{T}\boldsymbol{E}_e)^{-1}\boldsymbol{E}_e^\mathsf{T})\boldsymbol{F}_e\bar{\boldsymbol{\mu}} \\
&\text{(defining the idempotent matrix } \boldsymbol{Q}_e \doteq (\mathbf{I} - \boldsymbol{E}_e(\boldsymbol{E}_e^\mathsf{T}\boldsymbol{E}_e)^{-1}\boldsymbol{E}_e^\mathsf{T})) \\
&= \bar{\boldsymbol{\mu}}^\mathsf{T}\boldsymbol{F}_e^\mathsf{T}\boldsymbol{Q}_e\boldsymbol{F}_e\bar{\boldsymbol{\mu}} = \bar{\boldsymbol{\mu}}^\mathsf{T}\boldsymbol{F}_e^\mathsf{T}\boldsymbol{Q}_e\boldsymbol{Q}_e\boldsymbol{F}_e\bar{\boldsymbol{\mu}} = \|\boldsymbol{Q}_e\boldsymbol{F}_e\bar{\boldsymbol{\mu}}\|^2
\end{aligned}$$

Now we write $\boldsymbol{E}_e$ in terms of its 3×3 blocks, and we write the vector $\boldsymbol{F}_e\bar{\boldsymbol{\mu}}$ explicitly by noticing that the nonzero blocks in $\boldsymbol{F}_e$ are the same (up-to-sign) as the ones in $\boldsymbol{E}_e$ (*c.f.* the coefficient matrices in (18)):

$$\boldsymbol{E}_e = \begin{bmatrix} \boldsymbol{E}_{e0} \\ \boldsymbol{E}_{e1} \\ \vdots \\ \boldsymbol{E}_{eH} \end{bmatrix} \qquad \boldsymbol{F}_e\bar{\boldsymbol{\mu}} = \begin{bmatrix} -\boldsymbol{E}_{e0}\bar{\boldsymbol{\mu}}_0 \\ -\boldsymbol{E}_{e1}\bar{\boldsymbol{\mu}}_1 \\ \vdots \\ -\boldsymbol{E}_{eH}\bar{\boldsymbol{\mu}}_H \end{bmatrix} \tag{39}$$

Now we observe that $\boldsymbol{Q}_e$ is an orthogonal projector onto the null space of $\boldsymbol{E}_e$, and the null space of $\boldsymbol{Q}_e$ is spanned by the columns of $\boldsymbol{E}_e$. Therefore, any vector $\boldsymbol{v}$ that falls in the null space of $\boldsymbol{Q}_e$ can be written as a linear combination of the columns of $\boldsymbol{E}_e$:

$$\boldsymbol{Q}_e\boldsymbol{v} = \mathbf{0} \Leftrightarrow \boldsymbol{v} = \boldsymbol{E}_e\boldsymbol{w} \tag{40}$$

with $\boldsymbol{w} \in \mathbb{R}^3$. By comparison with (39), we note that $\boldsymbol{F}_e\bar{\boldsymbol{\mu}}$ can be written as $\boldsymbol{E}_e\boldsymbol{w}$ if and only if $\bar{\boldsymbol{\mu}}_1 = \bar{\boldsymbol{\mu}}_2 = \ldots = \bar{\boldsymbol{\mu}}_H$. Therefore, if $\bar{\boldsymbol{\mu}}_i \neq \bar{\boldsymbol{\mu}}_j$ for some i, j, then the vector $\boldsymbol{F}_e\bar{\boldsymbol{\mu}}$ cannot be in the null space of $\boldsymbol{Q}_e$, and the lower bound (39) must be greater than zero, concluding the proof. ∎

In words, Proposition 11 states that the submodularity ratio does not vanish as long as the directions of largest uncertainty change along the horizon. The following corollary is a straightforward consequence of Proposition 11.

Corollary 12 (Approximate submodularity of $\lambda_{\min}$). *The set function $f_\lambda(\mathsf{S})$ defined in* (9) *is monotone. Moreover, under the assumptions of Proposition 11, the greedy algorithm applied to* (1) *using $f_{\det}(\mathsf{S})$ as objective enjoys the guarantees of Proposition 10 for a nonzero $\gamma_{\mathcal{S}^\#}$.*

Proof: Monotonicity follows from the Weyl inequality [73]. The guarantees of the greedy algorithm follow from Proposition 10 and Proposition 11. ∎

Corollary 12 guarantees that the approximation bound of Proposition 10 does not vanish, hence the greedy algorithm always approximate the optimal solution up to a constant-factor. This is in contrast with [39], in which the additive bound can easily produce vacuous guarantees. Empirical evidence, shown in Section V, confirms that the greedy algorithm applied to the maximization of $f_\lambda(\mathsf{S})$ has excellent performance, producing near-optimal results in all test instances.

Remark 13 (Geometric Intuition Behind Greedy Selection with $\lambda_{\min}$). *Our linear model enables a deeper understanding of the geometry behind the greedy selection. The greedy selection rewards features l with large objective $\lambda_{\min}(\bar{\boldsymbol{\Omega}}_{k:k+H} + \Delta_l)$ or, equivalently, large marginal gain $\lambda_{\min}(\bar{\boldsymbol{\Omega}}_{k:k+H} + \Delta_l) - \lambda_{\min}(\bar{\boldsymbol{\Omega}}_{k:k+H})$. The following chain of relations provides a geometric understanding of which features induce a large marginal gain:*

$$\begin{aligned}
&\lambda_{\min}(\bar{\boldsymbol{\Omega}}_{k:k+H} + \Delta_l) - \lambda_{\min}(\bar{\boldsymbol{\Omega}}_{k:k+H}) \\
&\textit{(from Rayleigh quotient)} \\
&= \min_{\|\boldsymbol{\nu}\|=1} \boldsymbol{\nu}^\mathsf{T}(\bar{\boldsymbol{\Omega}}_{k:k+H} + \Delta_l)\boldsymbol{\nu} - \min_{\|\boldsymbol{\mu}\|=1} \boldsymbol{\mu}^\mathsf{T}(\bar{\boldsymbol{\Omega}}_{k:k+H})\boldsymbol{\mu} \\
&\textit{(calling } \bar{\mu} \textit{ the minimizer of the second summand)} \\
&= \min_{\|\boldsymbol{\nu}\|=1} \boldsymbol{\nu}^\mathsf{T}(\bar{\boldsymbol{\Omega}}_{k:k+H} + \Delta_l)\boldsymbol{\nu} - \bar{\boldsymbol{\mu}}^\mathsf{T}(\bar{\boldsymbol{\Omega}}_{k:k+H})\bar{\boldsymbol{\mu}} \\
&\textit{(substituting the suboptimal solution } \bar{\mu} \textit{ in the first summand)} \\
&\leq \bar{\boldsymbol{\mu}}^\mathsf{T}(\bar{\boldsymbol{\Omega}}_{k:k+H} + \Delta_l)\bar{\boldsymbol{\mu}} - \bar{\boldsymbol{\mu}}^\mathsf{T}(\bar{\boldsymbol{\Omega}}_{k:k+H})\bar{\boldsymbol{\mu}} \\
&\textit{(simplifying and substituting the expression of } \Delta_l) \\
&= \bar{\boldsymbol{\mu}}^\mathsf{T}\Delta_l\bar{\boldsymbol{\mu}} = \bar{\boldsymbol{\mu}}^\mathsf{T}\boldsymbol{F}_l^\mathsf{T}(\mathbf{I} - \boldsymbol{E}_l(\boldsymbol{E}_l^\mathsf{T}\boldsymbol{E}_l)^{-1}\boldsymbol{E}_l^\mathsf{T})\boldsymbol{F}_l\bar{\boldsymbol{\mu}} \\
&\textit{(defining the idempotent matrix } \boldsymbol{Q} \doteq (\mathbf{I} - \boldsymbol{E}_l(\boldsymbol{E}_l^\mathsf{T}\boldsymbol{E}_l)^{-1}\boldsymbol{E}_l^\mathsf{T})) \\
&= \bar{\boldsymbol{\mu}}^\mathsf{T}\boldsymbol{F}_l^\mathsf{T}\boldsymbol{Q}\boldsymbol{F}_l\bar{\boldsymbol{\mu}} = \bar{\boldsymbol{\mu}}^\mathsf{T}\boldsymbol{F}_l^\mathsf{T}\boldsymbol{Q}\boldsymbol{Q}\boldsymbol{F}_l\bar{\boldsymbol{\mu}} = \|\boldsymbol{Q}\boldsymbol{F}_l\bar{\boldsymbol{\mu}}\|^2 \\
&\textit{(using the triangle inequality and substituting } \boldsymbol{F}_l) \\
&\leq \|\boldsymbol{Q}\|^2\|\boldsymbol{F}_l\bar{\boldsymbol{\mu}}\|^2 = \|\boldsymbol{F}_l\bar{\boldsymbol{\mu}}\|^2 = \textstyle\sum_{k=0}^{H} \|[\boldsymbol{u}_{kl}]_\times(\boldsymbol{R}_{cam,k}^w)^\mathsf{T}\bar{\boldsymbol{\mu}}_k\|^2
\end{aligned}$$

where $\bar{\boldsymbol{\mu}}_k$ is the subvector of $\bar{\boldsymbol{\mu}}$ at the entries corresponding to the robot position at time k. Intuitively, the inequalities reveal that the marginal gain is small when $\|[\boldsymbol{u}_{kl}]_\times(\boldsymbol{R}_{cam,k}^w)^\mathsf{T}\bar{\boldsymbol{\mu}}_k\|$ is small, i.e., when we pick landmark observations where the

measured bearing $\boldsymbol{u}_{kl}$ is nearly parallel to the directions of large uncertainty $\bar{\boldsymbol{\mu}}_k$, transformed in the camera frame by the rotation $(\boldsymbol{R}^w_{cam,k})^\mathsf{T}$. For instance, if we have large uncertainty in the forward direction, it is not convenient to use features in front of the robot (i.e., with bearing parallel to the direction of largest uncertainty); accordingly, the greedy approach would select features in the periphery of the image, which intuitively provide a better way to reduce uncertainty.

V. Experiments

This section provides three sets of experimental results. The first set of tests, in Section V-A, shows that the greedy algorithm attains near-optimal solutions in solving problem (1), while being faster than convex relaxation techniques. The second set of tests, in Section V-B, evaluates our c++ pipeline in realistic simulations, showing that our feature selection techniques boost VIN performance; the same section also shows the advantage of using our lazy evaluation. The third set of tests, in Section V-C, evaluates our approach on real data collected by an agile micro aerial vehicle.

A. Assessment of the Greedy Algorithms for Feature Selection

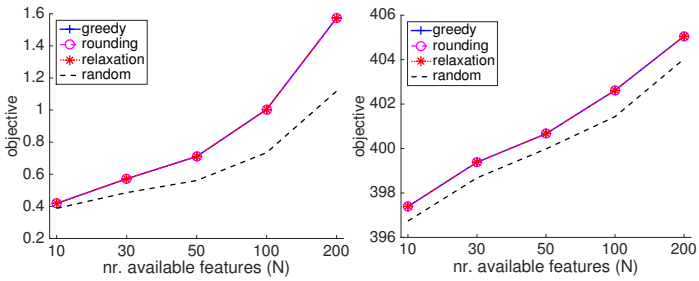
This section answers the following question: *how good is the greedy Algorithm 1 to (approximately) solve the combinatorial optimization problem* (1)*?* In particular, we show that the greedy algorithm finds a near-optimal solution of (1), for both choices of the cost function (9); we also show that the convex relaxation approach of Section IV-A finds near-optimal solutions, while being more computationally expensive.

Testing setup. To generate random instances of problem (1), we consider a robot moving along a straight line at a constant speed of $2\mathrm{m/s}$. The robot is equipped with an IMU with sampling period $\delta = 0.01\mathrm{s}$; we choose the accelerometer noise density equal to $0.02\mathrm{m}/(\mathrm{s}^2\sqrt{\mathrm{Hz}})$, and the accelerometer bias continuous-time noise density to be $0.03\mathrm{m}/(\mathrm{s}^3\sqrt{\mathrm{Hz}})$. We also simulate an on-board monocular camera, which measures 3D points randomly scattered in the environment, at a (key)frame rate of $0.5\mathrm{s}$. The robot has to select a set of κ features out of N available visual measurements. We assume that at the time of feature selection, the position covariance of the robot is $10^{-2}\cdot\mathbf{I}_3$, while its velocity and accelerometer bias covariances are $10^{-2}\cdot\mathbf{I}_3$ and $10^{-4}\cdot\mathbf{I}_3$, respectively. Using this information, we build the matrix $\bar{\boldsymbol{\Omega}}_{k:k+H}$, using a prediction horizon of $2.5\mathrm{s}$. Moreover, from the available feature measurements, we build the matrices Δ_l; in these tests we assume $p_l = 1$, i.e., we disregard appearance during feature selection.

Techniques and evaluation metrics. We compare two approaches to solve (1): the greedy algorithm of Algorithm 1 and the convex relaxation approach (25). We implemented the convex relaxation using CVX/MOSEK as parser/solver for (25), and then we computed the rounded solution as described in Section IV-A. For the evaluation in this section, we implemented both the greedy algorithm and the convex relaxation in Matlab. We evaluate these approaches for each choice of the objective functions f_λ and f_{det} defined in (9). Ideally, for each technique, we should compare the objective attained by the techniques, versus the optimal objective. Unfortunately, the optimal objective is hard to compute and a brute-force approach is prohibitively slow, even for relatively small problem instances.[5] Luckily, the convex relaxation (25) also produces an upper bound on the optimal cost of (1) (*c.f.* eq. (26)), hence we can use this upper bound to understand how far are the greedy and the rounded solution of (25) from optimality.

Results. We consider problems of increasing sizes in which we are given N features and we have to select half of them ($\kappa = N/2$) to maximize the objective in (1). For each N, we compute statistics over 50 Monte Carlo.

Fig. 1(a) shows the smallest eigenvalue objective f_λ attained by the different techniques for increasing number of features N. Besides the greedy, the rounded convex relaxation (label: rounded), and the relaxed objective (label: relaxed), we show the objective attained by picking a random subset of κ features (label: random). We are solving a maximization problem hence the larger the objective the better. Fig. 1(a) shows that in all tested instances, greedy and rounded match the upper bound relaxed (the three lines are practically indistinguishable), hence they both produce optimal solutions (*c.f.* eq. (27)). The resulting solution is far better than random. This result is somehow surprising, since the smallest eigenvalue is not submodular in general, and the greedy algorithm enjoys weaker performance guarantees (Corollary 12). However, this observation is in agreement with related work in other fields, e.g., [47]. While both greedy and rounded return good solutions, solving the convex problem (25) is usually more expensive than computing the greedy solution: the CPU time of our greedy algorithm in Matlab (without lazy evaluation) is around $0.4\mathrm{s}$ (for $N = 50$), while CVX requires around $0.8\mathrm{s}$.



(a) f_λ (b) f_{det}

Fig. 1. Techniques to approximately solve problem (1) for (a) the smallest eigenvalue objective f_λ, and (b) the log-determinant objective f_{det}. The figure reports the objective attained by the greedy algorithm (greedy), the rounded solution (rounded), and a random selection (random). The upper bound relaxed, attained by the convex problem (25) (before rounding), is shown for comparison.

Analogous considerations hold for the objective f_{det}. Fig. 1(b) shows the log-determinant attained by the different techniques, for increasing number of available features N; also in this case the algorithms have to select $\kappa = N/2$ features. As in the previous tests, greedy and rounded attain the optimal solution in all test instances, matching the upper bound relaxed, and performing remarkably better than a random

[5]Even in a small instance in which we are required to select 50 out of 100 available visual measurements, a brute-force approach would need to evaluate around 10^{29} possible sets.

choice. Regarding the CPU time, our Matlab implementation of the greedy algorithm to optimize f_λ takes around 0.1s (for $N = 50$), while CVX requires more than 1min to solve (25).[6] Since the greedy algorithms are as accurate as the convex relaxation technique, while being faster, in the following we focus on the former.

B. Importance of Feature Selection in VIN

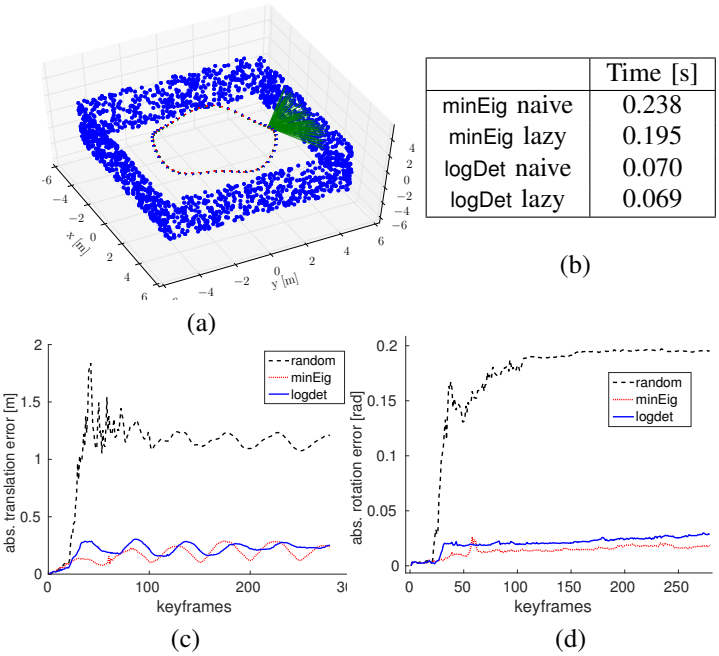
This section answers the following question: *does the feature selection resulting by solving* (1) *lead to performance improvements in VIN?* In the following we show that the proposed feature selection approach boosts VIN performance in realistic Monte Carlo simulations.

Testing setup. We adopt the benchmarking problem of [66] and pictured in Fig. 2(a) as testing setup. We simulate a robot that follows a circular trajectory with a sinusoidal vertical motion. The total length of the trajectory is 120m. The on-board camera has a focal length of 315 pixels and runs at a rate of 2.5Hz (simulating keyframes). Simulated acceleration and gyroscope measurements are obtained as in [66].

Implementation and evaluation metrics. In this section we focus on the greedy algorithms and we use those to select a subset of visual features. We implemented the greedy algorithms and the construction of the matrices required in the functions (9) in c++, using eigen for the computation of the log-determinant and the smallest eigenvalue. For numerical reasons, rather than computing the determinant and taking the logarithm, we directly compute the log-determinant from the Cholesky decomposition of the matrix. For the computation of the smallest eigenvalue we use eigen's svd function.

Our feature selection approach is used as an add-on to a visual-inertial pipeline similar to the one described in [66]. Our VIN pipeline estimates the navigation state (robot pose, velocity, and IMU biases) using the structureless visual model and the pre-integrated IMU model described in [66]. The entire implementation is based on the GTSAM optimization library [76]. Our implementation differs from [66] in three important ways. First, in this paper we use the iSAM2 algorithm within a fixed-lag smoothing approach; we marginalize out states outside a smoothing horizon of 6s, which helps bounding latency and memory requirements. Second, we do not adopt SVO as visual front-end; in this simulations we do not need a front-end as we simulate landmark observations, while in the following section we describe a simple real-world front-end. Finally, rather than feeding to the VIN estimator all available measurements, we use the feature selection algorithms described in this paper to select a small set of informative visual observations.

In this section we evaluate two main aspects of our approach. First, we show that a clever selection of the features does actually impact VIN accuracy. Second, we show that the lazy evaluation approach discussed in Section IV-C speeds up the computation of the greedy solution. We use two metrics for accuracy: the *absolute translation error*, which is the Euclidean distance between the estimated position and the actual position, and the *relative translation error*, which computes the Euclidean norm of the difference between the estimated translation between time k and time $k+1$ and the actual translation. Indeed the relative translation error quantifies how quickly the estimate drifts from the ground truth. Since absolute positions are not observable in visual-inertial odometry, the relative error is a more reliable performance metric. When useful, we report absolute and relative rotation errors (defined in analogy with the translation ones).



	Time [s]
minEig naive	0.238
minEig lazy	0.195
logDet naive	0.070
logDet lazy	0.069

Fig. 2. Simulation results: (a) simulated environment, (b) table with CPU times for different implementations of the greedy algorithms, (c) absolute translation errors, (d) absolute rotation errors.

Results. We simulate 50 Monte Carlo runs; in each run we add random noise to the acceleration, gyroscope, and camera measurements. To make the simulation realistic, the statistics about measurement noise are identical to the ones used in the real tests of Section V-C. In each run, the robot performs VIN and, at each camera frames, it selects $\kappa = 20$ visual features out of all the features in the field of view. We compare three feature selection strategies. The greedy selection resulting from Algorithm 1 with the eigenvalue objective f_λ (label: minEig), the greedy selection with the log-determinant cost $f_{\det}$ (label: logDet), and a random selection which randomly draws κ of the available features (label: random).

Fig. 2(c)-(d) show the absolute translation and absolute rotation errors, averaged over 50 Monte Carlo runs. From the figure is clear that a clever selection of the features, resulting from logDet or minEig, deeply impacts performance in VIN. Our techniques largely improve estimation errors, when compared against the random selection; both approaches result in similar performance. From the figure we note that the absolute errors have some oscillations: this is a consequence of the fact that the trajectory is circular; in general, this stresses the fact that

[6] CVX uses a successive approximation method to maximize the log-det objective, which is known to be fairly slow.

absolute metrics may be poor indicators of performance in visual-inertial odometry. In this case, the relative error metrics confirm the results of Fig. 2(c)-(d): the average translation and rotation errors are given in Table I; in parenthesis we report the error reduction percentage with respect to the random baseline.

Technique	Rel. Translation Error [m]	Rel. Rotation Error [rad]
random	0.0103	0.0049
minEig	0.0064 (-37%)	0.0025 (-48%)
logDet	0.0053 (-48%)	0.0018 (-63%)

TABLE I
RELATIVE TRANSLATION AND ROTATION ERRORS FOR THE SIMULATED TESTS OF SECTION V-B (AVERAGE OVER 50 MONTE CARLO RUNS)

Fig. 2(b) reports the CPU time required for feature selection. The figure considers both cost functions (logDet and minEig) and compares timing when using our lazy evaluation, as described in Algorithm 1, against a naive implementation of the greedy algorithm that always tests the marginal gain of every feature (i.e., for which the stopping condition in line 12 of Algorithm 1 is disabled). The naive greedy (without lazy evaluation) always results in κN objective evaluations. When using lazy evaluation, the number of objective evaluation depends on the tightness of the upper bounds used in Algorithm 1. From Fig. 2(b), we see that the advantage of using the lazy evaluation is marginal for the log-determinant cost; this is not surprising, since the Hadamard's inequality of Proposition 2 usually gives a fairly loose bound. On the other hard, the advantage of using the lazy evaluation is significant for the minEig, resulting in a reduction of the computational time of 20%. The average CPU time required by Algorithm 1 (with lazy evaluation) to select $\kappa = 20$ features is 0.069s for logDet and 0.195s for minEig. While these timing may be already acceptable for applications, there are large margins to speed up computation: we postpone these considerations to Section VI.

C. Real Tests: Agile Navigation on Micro Aerial Vehicles

In this section we show that our feature selection approach enhances VIN performance in real-world navigation problems with micro-aerial vehicles (MAVs).

Testing setup. We use the *EuRoC* benchmark [11] for our evaluation. The EuRoC datasets are collected with an AscTech Firefly hex-rotor helicopter equipped with a VI (stereo) visual-inertial sensor. The camera records stereo images at resolution 752×480 and framerate 20Hz; IMU data is collected at 200Hz. We refer to [11] for a technical description of the datasets. In this context we only remark that the datasets contain test instances at increasing levels of complexity, collected in a machine hall environment and in a smaller Vicon room. In our tests, the measurement variances, as well as the intrinsic and extrinsic calibration parameters match exactly the one specified in the dataset. The most relevant parameters used in our tests are given in Table II; in the front-end we used openCV's *goodFeaturesToTrack* for feature detection and the Lucas-Kanade method for feature tracking; as input to the detector we specify a minimum quality level for the features and a desired number of features to extract (N). From these N features our selector has to retain $\kappa = 10$ features that will be used by the back-end. In this sense, feature detection and tracking at the front-end are pre-attentive mechanisms: they work on a large set of features, which are later filtered out by our feature selector. The feature selector uses a predictive horizon of 3s; in practice, the future pose estimates along the horizon can be computed from the control inputs, by integrating the dynamics of the vehicle (Section III-B). Since the control inputs are not available in the EuRoC dataset, we compute the future poses by attaching ground truth motion increments to the current pose estimate. The only assumption in doing so is that the control loop and the estimation quality are good enough to track a desired set of future poses; this is the case in VIN in which the short-term drift is small.

	Parameter name	Value
Front-end	Nr. features to detect (N)	100
	Minimum quality level	0.001
	Time between keyframes	0.2s
Back-end	Smoothing window	6s
	iSAM2 iterations	1
Feature selector	Nr. features to select (κ)	10
	Horizon	3s

TABLE II
VIN AND FEATURE SELECTION PARAMETERS

Techniques. We compare four VIN approaches. The first two VIN approaches use the minEig and the logDet selectors proposed in this paper. The third approach uses a selector that picks the κ features with highest quality (i.e., highest score in *goodFeaturesToTrack*). This selector is commonly used in VIN and only accounts for the appearance of the visual features; we denote it with the label "quality", following openCV's terminology. The fourth technique is a VIN approach using 200 features (selected as the ones with largest score in *goodFeaturesToTrack*) and is used to have a reference performance for the case in which the VIN system has less stringent computational constraints (label: no-selection).

In order to compute the tracking probabilities p_l, we modified openCV's *goodFeaturesToTrack* in order to have access to the features' scores. Then, we mapped the scores to probabilities in $[0, 1]$, such that more distinguishable features have higher tracking probabilities p_l.

Results. Fig. 3 shows the performance of the compared techniques on all the 11 EuRoC datasets. The EuRoC benchmark includes datasets of different levels of complexity, with the difficult datasets being challenging for standard VIN pipeline due to the fast motion of the MAV. In this section we show that we can obtain accurate position estimation with as few as $\kappa = 10$ features; this budged is enforced for each frame; for instance, if we are tracking r features from the previous frame, then in the current frame we can only retain $\kappa - r$ features.

Fig. 3(a) compares the VIN performance using the relative translation errors as metric. The figure confirms that the difficult datasets tend to have larger translation errors. Moreover, it shows that the proposed techniques, minEig and logDet, lead to smallest errors compared to the baseline quality. Clearly, the technique no-selection, which uses 20x more features, leads to the smaller errors. To better appreciate the advantage of minEig and logDet with respect to quality, Fig. 3(b) shows the relative improvement, i.e., the relative translation error

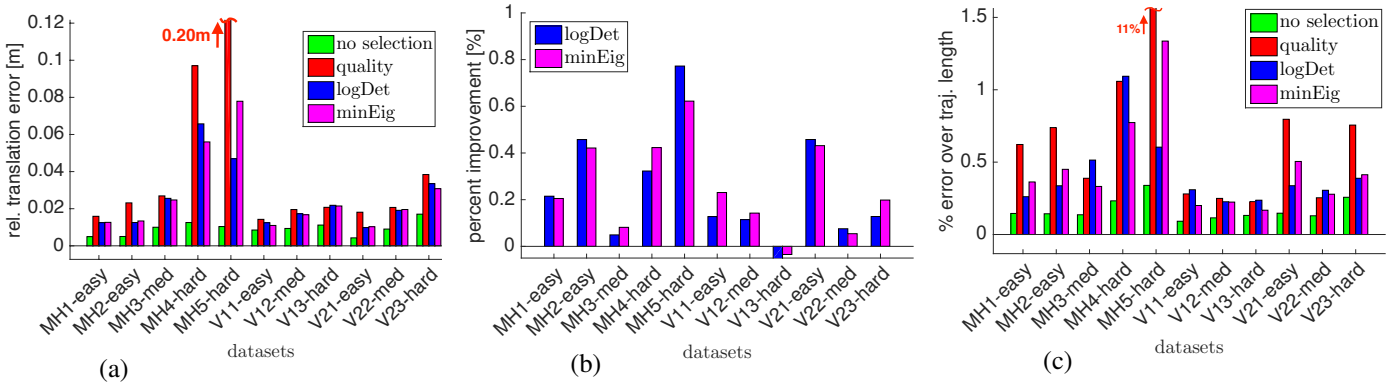
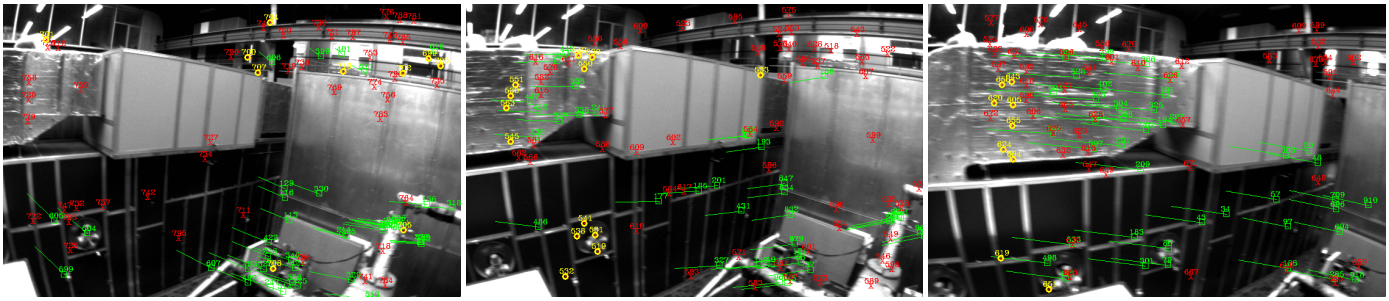


Fig. 3. Accuracy for the compared techniques on the 11 EuRoC MAV datasets. (a) Relative translation error; (b) Relative improvement (relative translation error reduction) of the proposed techniques with respect to the quality baseline; (c) Translation error as percentage of the overall trajectory length.



(a) quality (b) minEig (c) logDet

Fig. 4. Snapshots of the feature selection performed by the techniques quality, minEig, and logDet during a sharp left turn. Features tracked from previous frames are shown as green squares (with the corresponding optical flow vectors), the newly detected features are shown as red crosses, and the selected features are shown as yellow circles. We note that quality only selects the features from their appearance, and chooses many features on the right-hand side of the frames: these features will soon fall out of the field of view due to the sharp turn.

reduction, of the two techniques with respect to quality. The figure shows that the proposed feature selectors result in much smaller drift across all but one datasets. The average error reduction is larger than 20% and overcomes 40% in the datasets MH_02_easy, MH_05_difficult, and V2_01_easy. In particular, in the dataset MH_05_difficult the estimate resulting from the quality-based feature selection diverged after a sharp turn, while our techniques were able to ensure accurate pose estimation. The dataset V1_03_difficult is the only one in which the proposed techniques have slightly worse performance. We noticed that in datasets with severe motion blur the advantage of the proposed techniques may vary, and this is due to the fact that we are using a simplistic model for the tracking probabilities p_l. For completeness, Fig. 3(c) reports the absolute translation error as a percentage of the trajectory traveled; this is another common metric for VIN. We notice that no-selection has excellent performance, while using 200 features (average error accumulation is 0.17% of the trajectory length). Moreover, the proposed techniques, logDet and minEig, are able to ensure an average error accumulation of 0.42% and 0.46%, respectively, while using only 10 features!

To get a better intuition behind the large performance boost induced by the proposed techniques, we report few snapshots produced by our pipeline in Fig. 4. Each sub-figure shows, for the current frame, the tracked features (green squares with the optical flow vector), the available features (red crosses), and the features selected (yellow circles) by (a) quality, (b) logDet, and (c) minEig. The frames are captured during a sharp left turn from the MH_03_medium dataset. The quality selector simply picks the most distinguishable features, resulting in many features selected on the right-hand side of the image; these features are of scarce utility: they will soon disappear from the field of view due to the motion of the MAV. On the other hand, logDet and minEig are predictive and they leverage the knowledge of the immediate motion of the platform; therefore they tend to discard features that fall outside the field of view and select features on the left-hand side of the image.

Fig. 5 reports the average CPU time required by the VIN back-end for all techniques and datasets. The figure shows that logDet is able to cut the back-end time in half, with respect to no-selection. The CPU time of quality is even smaller, at the cost of degraded performance (Fig. 3). Consistently with the Monte Carlo analysis, in our current implementation logDet is faster than minEig. In the following section we discuss extensions that can make the selection time negligible.

VI. Conclusion and Future Work

This work provides an attention mechanism for visual-inertial navigation. This mechanism takes the form of a feature selector, which retains the most informative visual features detected by the VIN front-end (pre-attentive process) and feeds them to the estimation back-end. We proposed two algorithms

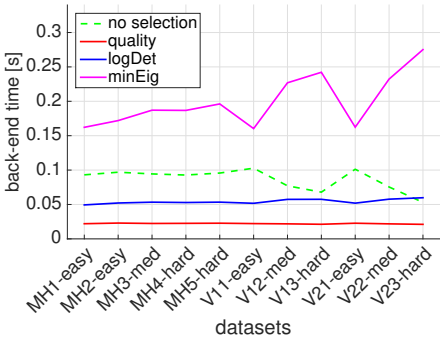


Fig. 5. CPU time for the back-end (including feature selection) for the compared techniques on the EuRoC datasets.

for feature selection. Both algorithms enjoy four desirable qualities: they are predictive in nature, in that they are aware of the motion of the robot in the immediate future; they are task-driven, since they select a set of features that minimize the VIN estimation error; they are greedy, hence efficient and easy to implement; they come with performance guarantees that bound their sub-optimality. We demonstrated our feature selection extensively on both realistic Monte Carlo simulations and real-world data collected by a micro aerial vehicle. The experiments suggest that the feature selection seriously impacts VIN performance; the use of the proposed techniques reduces the estimation error in easy datasets, and enables accurate estimation in difficult datasets in which standard approaches would fail on a limited budget of visual features. This work opens many avenues for future investigation.

Computational improvements. The first avenue for future work consists in reducing the computational time of feature selection. Two main ideas can make the feature selection time negligible. The first stems from the observation that the greedy algorithm is trivially parallelizable: the marginal gain of each feature can be computed independently; leveraging this fact alone would result in large computational savings. The second idea is to use sparse matrix manipulation to compute the determinant and the smallest eigenvalues; our current implementation uses dense matrices.

Task-driven perception. A second avenue for future work consists in extending our attention framework. We plan to explore two paths. First, while (1) minimizes the localization uncertainty subject to a feature budget, one may also consider a "dual" problem in which one minimizes the number of features to be used, while satisfying a desired localization performance. From the technical standpoint, this alternative formulation can be tackled in a similar manner and in both cases greedy algorithms have sub-optimality guarantees. This alternative formulation would provide a grounded answer to the question: *how much visual information is needed to navigate at a desired accuracy?* The second avenue consists in extending our attention framework to other tasks: for instance, *how many visual features does the robot need to sense in order to avoid crashing into nearby obstacles?* We believe these are necessary steps towards the development of a task-driven perception theory, that can enable autonomy on heavily resource-constrained robots with strict budget on sensing and computation.

ACKNOWLEDGMENTS

The authors gratefully acknowledge Christian Forster for the useful discussions and for providing the VIN simulator.

APPENDIX

In this appendix we provide explicit expressions for the matrices and vectors appearing in the IMU model (14).

Given the velocity $\boldsymbol{v}_k$ and the position $\boldsymbol{t}_k$ of the robot at time k, we can get $\boldsymbol{v}_{k+1}$ and $\boldsymbol{v}_{k+1}$ by simple Euler integration using the acceleration $\boldsymbol{a}_k$:

$$\boldsymbol{v}_{k+1} = \boldsymbol{v}_k + \boldsymbol{a}_k\delta \tag{41}$$
$$\boldsymbol{v}_{k+1} = \boldsymbol{t}_k + \boldsymbol{v}_k\delta + \tfrac{1}{2}\boldsymbol{a}_k\delta^2 \tag{42}$$

By induction, the velocity and position at time $j > k$ are:

$$\begin{aligned}
\boldsymbol{v}_j &= \boldsymbol{v}_k + \textstyle\sum_{i=k}^{j-1} \boldsymbol{a}_i\delta \\
\boldsymbol{t}_j &= \boldsymbol{t}_k + \textstyle\sum_{i=k}^{j-1} \boldsymbol{v}_i\delta + \frac{1}{2}\sum_{i=k}^{j-1} \boldsymbol{a}_i\delta^2 \\
&\quad \text{(substituting } \boldsymbol{v}_i\text{)} \\
&= \boldsymbol{t}_k + \textstyle\sum_{i=k}^{j-1}(\boldsymbol{v}_k + \sum_{h=k}^{i-1} \boldsymbol{a}_h\delta)\delta + \frac{1}{2}\sum_{i=k}^{j-1} \boldsymbol{a}_i\delta^2 \\
&\quad \text{(moving } \boldsymbol{v}_k \text{ outside the sum)} \\
&= \boldsymbol{t}_k + (j-k-1)\boldsymbol{v}_k\delta + \textstyle\sum_{i=k}^{j-1}\sum_{h=k}^{i-1} \boldsymbol{a}_h\delta^2 \\
&\quad + \frac{1}{2}\sum_{i=k}^{j-1} \boldsymbol{a}_i\delta^2 \\
&\quad \text{(noting that a term disappears from the sum)} \\
&= \boldsymbol{t}_k + (j-k-1)\boldsymbol{v}_k\delta + \textstyle\sum_{i=k}^{j-2}\sum_{h=k}^{i-1} \boldsymbol{a}_h\delta^2 \\
&\quad + \frac{1}{2}\sum_{i=k}^{j-1} \boldsymbol{a}_i\delta^2 \\
&\quad \text{(since the first term with } k \text{ appears } j-k-2 \text{ times)} \\
&= \boldsymbol{t}_k + (j-k-1)\boldsymbol{v}_k\delta + \textstyle\sum_{i=k}^{j-2}(j-i-1)\boldsymbol{a}_i\delta^2 \\
&\quad + \frac{1}{2}\sum_{i=k}^{j-1} \boldsymbol{a}_i\delta^2 \\
&\quad \text{(putting last two terms together)} \\
&= \boldsymbol{t}_k + (j-k-1)\boldsymbol{v}_k\delta + \tfrac{1}{2}\boldsymbol{a}_{j-1}\delta^2 \\
&\quad + \textstyle\sum_{i=k}^{j-2}(j-i-\frac{1}{2})\boldsymbol{a}_i\delta^2 \\
&\quad \text{(simplifying)} \\
&= \boldsymbol{t}_k + (j-k-1)\boldsymbol{v}_k\delta + \textstyle\sum_{i=k}^{j-1}(j-i-\frac{1}{2})\boldsymbol{a}_i\delta^2
\end{aligned}$$

Defining $\delta_{kj} \doteq (j-k-1)\delta$ and substituting $\boldsymbol{a}_k$ from (10):

$$\begin{aligned}
\boldsymbol{v}_j &= \boldsymbol{v}_k + \textstyle\sum_{i=k}^{j-1} \left(\boldsymbol{R}_i(\tilde{\boldsymbol{a}}_i - \boldsymbol{b}_k - \boldsymbol{\eta}_i) + \mathbf{g}\right)\delta \\
&= \boldsymbol{v}_k + \mathbf{g}\delta_{kj} - (\textstyle\sum_{i=k}^{j-1} \boldsymbol{R}_i\delta)\boldsymbol{b}_k + \sum_{i=k}^{j-1} \boldsymbol{R}_k\tilde{\boldsymbol{a}}_i\delta \\
&\quad - \textstyle\sum_{i=k}^{j-1} \boldsymbol{R}_k\boldsymbol{\eta}_i\delta \\
\boldsymbol{t}_j &= \boldsymbol{t}_k + \boldsymbol{v}_k\delta_{kj} + \textstyle\sum_{i=k}^{j-1}(j-i-\frac{1}{2})\left(\boldsymbol{R}_i(\tilde{\boldsymbol{a}}_i - \boldsymbol{b}_k - \boldsymbol{\eta}_i) + \mathbf{g}\right)\delta^2 \\
&= \boldsymbol{t}_k + \boldsymbol{v}_k\delta_{kj} + \textstyle\sum_{i=k}^{j-1}(j-i-\frac{1}{2})\mathbf{g}\delta^2 \\
&\quad - (\textstyle\sum_{i=k}^{j-1}(j-i-\frac{1}{2})\boldsymbol{R}_i\delta^2)\boldsymbol{b}_k + \sum_{i=k}^{j-1}(j-i-\frac{1}{2})\boldsymbol{R}_i(\tilde{\boldsymbol{a}}_i - \boldsymbol{\eta}_i)\delta^2
\end{aligned} \tag{43}$$

Let us now define the following vectors:

$$\begin{aligned}
\boldsymbol{z}^{\boldsymbol{v}}_{kj} &\doteq \mathbf{g}\delta_{kj} + \textstyle\sum_{i=k}^{j-1} \boldsymbol{R}_k\tilde{\boldsymbol{a}}_i\delta \\
\boldsymbol{\eta}^{\boldsymbol{v}}_{kj} &\doteq \textstyle\sum_{i=k}^{j-1} \boldsymbol{R}_k\boldsymbol{\eta}_i\delta \\
\boldsymbol{z}^{\boldsymbol{t}}_{kj} &\doteq \textstyle\sum_{i=k}^{j-1}(j-i-\frac{1}{2})\mathbf{g}\delta^2 \\
&\quad + \textstyle\sum_{i=k}^{j-1}(j-i-\frac{1}{2})\boldsymbol{R}_i\tilde{\boldsymbol{a}}_i\delta^2 \\
\boldsymbol{\eta}^{\boldsymbol{t}}_{kj} &\doteq + \textstyle\sum_{i=k}^{j-1}(j-i-\frac{1}{2})\boldsymbol{R}_i\boldsymbol{\eta}_i\delta^2
\end{aligned} \tag{44}$$

Using this notation we rewrite eq. (43) (putting position first) and adding the random walk random model on the bias:

$$\begin{aligned}\boldsymbol{t}_j &= \boldsymbol{t}_k + \boldsymbol{v}_k\delta_{kj} - \left(\textstyle\sum_{i=k}^{j-1}(j-i-\frac{1}{2})\boldsymbol{R}_i\delta^2\right)\boldsymbol{b}_k + \boldsymbol{z}^{\boldsymbol{t}}_{kj} - \boldsymbol{\eta}^{\boldsymbol{t}}_{kj}\\ \boldsymbol{v}_j &= \boldsymbol{v}_k - (\textstyle\sum_{i=k}^{j-1}\boldsymbol{R}_i\delta)\boldsymbol{b}_k + \boldsymbol{z}^{\boldsymbol{v}}_{kj} - \boldsymbol{\eta}^{\boldsymbol{v}}_{kj}\\ \boldsymbol{b}_j &= \boldsymbol{b}_k - \boldsymbol{\eta}^{\mathbf{b}}_{kj} \end{aligned} \quad (45)$$

In order to write (45) in compact matrix form, we define:

$$\boldsymbol{N}_{kj} \doteq \textstyle\sum_{i=k}^{j-1}(j-i-\frac{1}{2})\boldsymbol{R}_i\delta^2 \quad (46)$$

$$\boldsymbol{M}_{kj} \doteq \textstyle\sum_{i=k}^{j-1}\boldsymbol{R}_i\delta \quad (47)$$

which allows rewriting (45) succinctly as:

$$\begin{aligned}\boldsymbol{z}^{\boldsymbol{t}}_{kj} &= \boldsymbol{t}_j - \boldsymbol{t}_k - \boldsymbol{v}_k\delta_{kj} + \boldsymbol{N}_{kj}\boldsymbol{b}_k + \boldsymbol{\eta}^{\boldsymbol{t}}_{kj}\\ \boldsymbol{z}^{\boldsymbol{v}}_{kj} &= \boldsymbol{v}_j - \boldsymbol{v}_k + \boldsymbol{M}_{kj}\boldsymbol{b}_k + \boldsymbol{\eta}^{\boldsymbol{v}}_{kj}\\ \boldsymbol{z}^{\mathbf{b}}_{kj} &= \boldsymbol{b}_j - \boldsymbol{b}_k + \boldsymbol{\eta}^{\mathbf{b}}_{kj}\end{aligned} \quad (48)$$

where $\boldsymbol{z}^{\boldsymbol{v}}_{kj} = \mathbf{0}_3$ is the expected change in the bias.

Let us now define the following matrices and vectors:

$$\boldsymbol{A}_{kj} = \left[\begin{array}{c|c|ccc|c|c|c} \mathbf{0}_{9\times 9} & \dots & -\mathbf{I}_3 & -\mathbf{I}_3\delta_{kj} & \boldsymbol{N}_{kj} & \mathbf{I}_9 & \mathbf{0}_{9\times 9} & \dots \\ & & \mathbf{0} & -\mathbf{I}_3 & \boldsymbol{M}_{kj} & & & \\ & & \mathbf{0} & \mathbf{0} & -\mathbf{I}_3 & & & \end{array}\right]$$

$$\boldsymbol{z}^{\text{IMU}}_{kj} = \begin{bmatrix}\boldsymbol{z}^{\boldsymbol{t}}_{kj}\\ \boldsymbol{z}^{\boldsymbol{v}}_{kj}\\ \boldsymbol{z}^{\mathbf{b}}_{kj}\end{bmatrix} \qquad \boldsymbol{\eta}^{\text{IMU}}_{kj} = \begin{bmatrix}\boldsymbol{\eta}^{\boldsymbol{t}}_{kj}\\ \boldsymbol{\eta}^{\boldsymbol{v}}_{kj}\\ \boldsymbol{\eta}^{\mathbf{b}}_{kj}\end{bmatrix} \quad (49)$$

Using (49), we finally rewrite our model (48) as:

$$\boldsymbol{z}^{\text{IMU}}_{kj} = \boldsymbol{A}_{kj}\boldsymbol{x}_{k:k+H} + \boldsymbol{\eta}^{\text{IMU}}_{kj} \quad (50)$$

To fully characterize the linear measurement model (50) we only have to compute the covariance of the noise $\boldsymbol{\eta}^{\text{IMU}}_{kj}$, which is given by:

$$\text{cov}(\boldsymbol{\eta}^{\text{IMU}}_{kj}) = \begin{bmatrix}\sigma^2_{\text{IMU}}\boldsymbol{C}\boldsymbol{C}^\mathsf{T} & \mathbf{0}_{6\times 3}\\ \mathbf{0}_{3\times 6} & \text{cov}(\boldsymbol{\eta}^{\mathbf{b}}_{kj})\end{bmatrix} \quad (51)$$

where $\boldsymbol{C}$ includes the coefficient matrices of the noise in (45):

$$\boldsymbol{C} = \begin{bmatrix}(j-k-\frac{1}{2})\boldsymbol{R}_k\delta^2 & (j-k-\frac{3}{2})\boldsymbol{R}_{k+1}\delta^2 & \dots & \frac{1}{2}\boldsymbol{R}_{j-1}\delta^2\\ \boldsymbol{R}_k\delta & \boldsymbol{R}_{k+1}\delta & \dots & \boldsymbol{R}_{j-1}\delta\end{bmatrix}$$

Using the fact that any rotation matrix satisfies $\boldsymbol{R}^\mathsf{T}_k\boldsymbol{R}_k = \mathbf{I}_3$, the matrix $\boldsymbol{C}\boldsymbol{C}^\mathsf{T}$ can be computed simply as:

$$\boldsymbol{C}\boldsymbol{C}^\mathsf{T} = \begin{bmatrix}\left(\sum_{i=k}^{j-1}(j-i-\frac{1}{2})^2\right)\delta^4\mathbf{I}_3 & \left(\sum_{i=k}^{j-1}(j-i-\frac{1}{2})\right)\delta^3\mathbf{I}_3\\ \left(\sum_{i=k}^{j-1}(j-i-\frac{1}{2})\right)\delta^3\mathbf{I}_3 & (j-k-1)\delta^2\mathbf{I}_3\end{bmatrix}.$$

References

[1] M. Potter, B. Wyble, C. Hagmann, and E. McCourt, "Detecting meaning in RSVP at 13 ms per picture," *Attention, Perception, & Psychophysics*, vol. 76, no. 2, pp. 270–279, 2014.

[2] M. Carrasco, "Visual attention: The past 25 years," *Vision Research*, vol. 51, pp. 1484–1525, 2011.

[3] J. Redmon, S. Divvala, R. Girshick, and A. Farhadi, "You only look once: Unified, real-time object detection," in *IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*, 2016.

[4] S. Pillai, S. Ramalingam, and J. Leonard, "High-performance and tunable stereo reconstruction," in *IEEE Intl. Conf. on Robotics and Automation (ICRA)*, 2016.

[5] R. Mur-Artal, J. Montiel, and J. Tardós, "ORB-SLAM: A versatile and accurate monocular SLAM system," *IEEE Trans. Robotics*, vol. 31, no. 5, pp. 1147–1163, 2015.

[6] N. G. Website, "Geforce gtx titan x specifications." [Online]. Available: http://www.geforce.com/hardware/desktop-gpus/geforce-gtx-titan-x/specifications

[7] Wikipedia, "List of cpu power dissipation figures." [Online]. Available: https://en.wikipedia.org/wiki/List_of_CPU_power_dissipation_figures

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

[9] P. Cavanagh, "Visual cognition," *Vision Research*, vol. 51, no. 13, pp. 1538 – 1551, 2011, vision Research 50th Anniversary Issue: Part 2. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S0042698911000381

[10] L. Carlone and S. Karaman, "Attention and anticipation in fast visual-inertial navigation," in *IEEE Intl. Conf. on Robotics and Automation (ICRA)*, 2017, pp. 3886–3893, extended arxiv preprint: 1610.03344 (pdf).

[11] M. Burri, J. Nikolic, P. Gohl, T. Schneider, J. Rehder, S. Omari, M. Achtelik, and R. Siegwart, "The EuRoC micro aerial vehicle datasets," *Intl. J. of Robotics Research*, 2016.

[12] A. Borji and L. Itti, "State-of-the-art in visual attention modeling," *IEEE Trans. Pattern Anal. Machine Intell.*, vol. 35, no. 1, 2013.

[13] B. Scholl, "Objects and attention: the state of the art," *Cognition*, vol. 80, no. 1, pp. 1–46, 2001.

[14] D. Caduff and S. Timpf, "On the assessment of landmark salience for human navigation," *Cogn. Process*, vol. 9, pp. 249–267, 2008.

[15] J. Moran and J. Desimone, "Selective attention gates visual processing in the extrastriate cortex," *Science*, vol. 229, no. 4715, pp. 782–784, 1985.

[16] J. Wolfe, "Guided search 2.0 - a revised model of visual search," *Psychon Bull. Rev.*, vol. 1, no. 2, pp. 202–238, 1994.

[17] H. Spekreijse, "Pre-attentive and attentive mechanisms in vision. perceptual organization and dysfunction," *Vision Research*, vol. 40, no. 10-12, pp. 1179–1182, 2000.

[18] R. Sim and G. Dudek, "Learning and evaluating visual features for pose estimation," in *Intl. Conf. on Computer Vision (ICCV)*, 1999, pp. 1217–1222.

[19] V. Peretroukhin, L. Clement, M. Giamou, and J. Kelly, "PROBE: Predictive robust estimation for visual-inertial navigation," in *IEEE/RSJ Intl. Conf. on Intelligent Robots and Systems (IROS)*, 2015.

[20] N. Ouerhani, A. Bur, and H. Hügli, "Visual attention-based robot self-localization," in *ECMR*, 2005, pp. 8–13.

[21] P. Newman and K. Ho, "Slam-loop closing with visually salient features," in *IEEE Intl. Conf. on Robotics and Automation (ICRA)*, 2005, pp. 635–642.

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

[23] C. Siagian and L. Itti, "Biologically-inspired robotics vision monte-carlo localization in the outdoor environment," in *IEEE/RSJ Intl. Conf. on Intelligent Robots and Systems (IROS)*, 2007, pp. 1723–1730.

[24] S. Frintrop and P. Jensfelt, "Attentional landmarks and active gaze control for visual SLAM," *IEEE Trans. Robotics*, vol. 24, no. 5, pp. 1054–1065, 2008.

[25] S. Hochdorfer and C. Schlegel, "Landmark rating and selection according to localization coverage: Addressing the challenge of lifelong operation of slam in service robots," in *IEEE/RSJ Intl. Conf. on Intelligent Robots and Systems (IROS)*, 2009, pp. 382–387.

[26] H. Strasdat, C. Stachniss, and W. Burgard, "Which landmark is useful? learning selection policies for navigation in unknown environments," in *IEEE Intl. Conf. on Robotics and Automation (ICRA)*, 2009, pp. 1410–1415.

[27] M. Chli and A. Davison, "Active matching for visual tracking," *Robotics and Autonomous Systems*, vol. 57, no. 12, pp. 1173–1187, Dec. 2009.

[28] A. Handa, M. Chli, H. Strasdat, and A. J. Davison, "Scalable active matching," in *IEEE Conference on Computer Vision and Pattern Recognition*, 2010.

[29] J. Jang, K. Won, and S. Jung, "Geometric feature selection for vehicle pose estimation on dynamic road scenes," in *5th Intl. Conf. on Ubiquitous and Future Networks*, 2013.

[30] Z. Shi, Z. Liu, X. Wu, and W. Xu, "Feature selection for reliable data association in visual SLAM," *Machine Vision and Applications*, vol. 24, pp. 667–682, 2013.

[31] V. Mnih, N. Heess, A. Graves, and K. Kavukcuoglu, "Recurrent models of visual attention," in *Advances in Neural Information Processing Systems (NIPS)*, 2014, pp. 2204–2212.

[32] K. Xu, J. L. Ba, R. Kiros, K. Cho, A. Courville, R. Salakhutdinov, R. Zemel, and Y. Bengio, "Show, attend and tell: Neural image caption generation with visual attention," in *ArXiv preprint: 1502.03044*, 2016.

[33] I. Cvišić and I. Petrović, "Stereo odometry based on careful feature selection and tracking," in *Proc. of the European Conference on Mobile Robots (ECMR)*, 2015.

[34] C. Cadena, L. Carlone, H. Carrillo, Y. Latif, D. Scaramuzza, J. Neira, I. Reid, and J. J. Leonard, "Past, present, and future of simultaneous localization and mapping: Toward the robust-perception age," *IEEE Trans. Robotics*, vol. 32, no. 6, pp. 1309–1332, 2016, arxiv preprint: 1606.05830, (pdf).

[35] A. Davison, "Active search for real-time vision," in *Intl. Conf. on Computer Vision (ICCV)*, Oct 2005.

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

[37] B. Mu, A. Agha-mohammadi, L. Paull, M. Graham, J. How, and J. Leonard, "Two-stage focused inference for resource-constrained collision-free navigation," in *Robotics: Science and Systems (RSS)*, 2015.

[38] K. Wu, T. Do, L. C. Carrillo-Arce, and S. I. Roumeliotis, "On the VINS resource-allocation problem for a dual-camera, small-size quadrotor," in *Intl. Sym. on Experimental Robotics (ISER)*, 2016, pp. 538–549.

[39] G. Zhang and P. Vela, "Good features to track for visual slam," in *IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*, 2015.

[40] D. Kottas, R. DuToit, A. Ahmed, C. Guo, G. Georgiou, R. Li, and S. Roumeliotis, "A resource-aware vision-aided inertial navigation system for wearable and portable computers," in *IEEE Intl. Conf. on Robotics and Automation (ICRA)*, 2014, pp. 6336–6343.

[41] S. Joshi and S. Boyd, "Sensor selection via convex optimization," *IEEE Trans. Signal Processing*, vol. 57, pp. 451–462, 2009.

[42] C. Giraud and B. Jouvencel, "Sensor selection: A geometrical approach," in *IEEE/RSJ Intl. Conf. on Intelligent Robots and Systems (IROS)*, vol. 2, 1995, pp. 1410–1415.

[43] A. Krause, A. Singh, and C. Guestrin, "Near-optimal sensor placements in gaussian processes: Theory, efficient algorithms and empirical studies," *Journal of Machine Learning Research*, vol. 9, pp. 235–284, 2008.

[44] F. Bian, D. Kempe, and R. Govindan, "Utility based sensor selection," in *5th Int. Conf. Information Processing Sensor Networks*, 2006, pp. 11–18.

[45] M. Shamaiah, S. Banerjee, and H. Vikalo, "Greedy sensor selection: leveraging submodularity," in *IEEE Conf. on Decision and Control (CDC)*, 2010.

[46] M. Vitus, W. Zhang, A. Abate, J. Hu, and C. Tomlin, "On efficient sensor scheduling for linear dynamical systems," *Automatica*, vol. 48, pp. 2482–2493, 2012.

[47] H. Zhang, R. Ayoub, and S. Sundaram, "Sensor selection for optimal filtering of linear dynamical systems: Complexity and approximation," in *IEEE Conf. on Decision and Control (CDC)*, 2015.

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

[49] V. Tzoumas, A. Jadbabaie, and G. Pappas, "Sensor placement for optimal kalman filtering: Fundamental limits, submodularity, and algorithms," in *American Control Conference*, 2016.

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

[51] A. Mourikis and S. Roumeliotis, "A multi-state constraint Kalman filter for vision-aided inertial navigation," in *IEEE Intl. Conf. on Robotics and Automation (ICRA)*, April 2007, pp. 3565–3572.

[52] A. Davison, I. Reid, N. Molton, and O. Stasse, "MonoSLAM: Real-time single camera SLAM," *IEEE Trans. Pattern Anal. Machine Intell.*, vol. 29, no. 6, pp. 1052–1067, Jun 2007.

[53] M. Bloesch, S. Omari, M. Hutter, and R. Siegwart, "Robust visual inertial odometry using a direct EKF-based approach," in *IEEE/RSJ Intl. Conf. on Intelligent Robots and Systems (IROS)*. IEEE, 2015.

[54] E. Jones and S. Soatto, "Visual-inertial navigation, mapping and localization: A scalable real-time causal approach," *Intl. J. of Robotics Research*, vol. 30, no. 4, Apr 2011.

[55] J. Hesch, D. Kottas, S. Bowman, and S. Roumeliotis, "Camera-imu-based localization: Observability analysis and consistency improvement," *Intl. J. of Robotics Research*, vol. 33, no. 1, pp. 182–201, 2014.

[56] A. Mourikis and S. Roumeliotis, "A dual-layer estimator architecture for long-term localization," in *Proc. of the Workshop on Visual Localization for Mobile Platforms at CVPR*, Anchorage, Alaska, June 2008.

[57] G. Sibley, L. Matthies, and G. Sukhatme, "Sliding window filter with application to planetary landing," *J. of Field Robotics*, vol. 27, no. 5, pp. 587–608, 2010.

[58] T.-C. Dong-Si and A. Mourikis, "Motion tracking with fixed-lag smoothing: Algorithm consistency and analysis," in *IEEE Intl. Conf. on Robotics and Automation (ICRA)*, 2011.

[59] S. Leutenegger, S. Lynen, M. Bosse, R. Siegwart, and P. Furgale, "Keyframe-based visual-inertial slam using nonlinear optimization," *Intl. J. of Robotics Research*, 2015.

[60] M. Bryson, M. Johnson-Roberson, and S. Sukkarieh, "Airborne smoothing and mapping using vision and inertial sensors," in *IEEE Intl. Conf. on Robotics and Automation (ICRA)*, 2009, pp. 3143–3148.

[61] V. Indelman, S. Wiliams, M. Kaess, and F. Dellaert, "Information fusion in navigation systems via factor graph based incremental smoothing," *Robotics and Autonomous Systems*, vol. 61, no. 8, pp. 721–738, August 2013.

[62] S. Shen, *Autonomous Navigation in Complex Indoor and Outdoor Environments with Micro Aerial Vehicles*. PhD Thesis, University of Pennsylvania, 2014.

[63] N. Keivan, A. Patron-Perez, and G. Sibley, "Asynchronous adaptive conditioning for visual-inertial SLAM," in *Intl. Sym. on Experimental Robotics (ISER)*, 2014.

[64] A. Patron-Perez, S. Lovegrove, and G. Sibley, "A spline-based trajectory representation for sensor fusion and rolling shutter cameras," *Intl. J. of Computer Vision*, February 2015.

[65] T. Lupton and S. Sukkarieh, "Visual-inertial-aided navigation for high-dynamic motion in built environments without initial conditions," *IEEE Trans. Robotics*, vol. 28, no. 1, pp. 61–76, Feb 2012.

[66] C. Forster, L. Carlone, F. Dellaert, and D. Scaramuzza, "On-manifold preintegration theory for fast and accurate visual-inertial navigation," *IEEE Trans. Robotics*, vol. 33, no. 1, pp. 1–21, 2016, arxiv preprint: 1512.02363, (pdf), technical report GT-IRIM-CP&R-2015-001.

[67] A. Das and D. Kempe, "Submodular meets spectral: Greedy algorithms for subset selection, sparse approximation and dictionary selection," in *Intl. Conf. on Machine Learning (ICML)*, 2011, pp. 1057–1064.

[68] R. Tron, L. Carlone, F. Dellaert, and K. Daniilidis, "Rigid components identification and rigidity enforcement in bearing-only localization using the graph cycle basis," in *American Control Conference*, 2015, (pdf).

[69] L. Carlone, Z. Kira, C. Beall, V. Indelman, and F. Dellaert, "Eliminating conditionally independent sets in factor graphs: A unifying perspective based on smart factors," in *IEEE Intl. Conf. on Robotics and Automation (ICRA)*, 2014.

[70] L. Carlone, P. F. Alcantarilla, H. Chiu, K. Zsolt, and F. Dellaert, "Mining structure fragments for smart bundle adjustment," in *British Machine Vision Conf. (BMVC)*, 2014, accepted as oral presentation (acceptance rate 7.7%), (pdf) (supplemental material: (pdf)).

[71] S. Boyd and L. Vandenberghe, *Convex optimization*. Cambridge University Press, 2004.

[72] R. Horn and C. Johnson, *Matrix Analysis*. Cambridge University Press, 1985.

[73] I. C. F. Ipsen and B. Nadler, "Refined perturbation bounds for eigenvalues of hermitian and non-hermitian matrices," *SIAM J. Matrix Analysis*, vol. 31, no. 1, 2009.

[74] G. Nemhauser, L. Wolsey, and M. Fisher, "An analysis of approximations for maximizing submodular set functions," *Mathematical Programming*, vol. 14, no. 1, pp. 265–294, 1978.

[75] A. Das, A. Dasgupta, and R. Kumar, "Selecting diverse features via spectral regularization," in *Advances in Neural Information Processing Systems (NIPS)*, vol. 2, 2012, pp. 1583–1591.

[76] F. Dellaert, "Factor graphs and GTSAM: A hands-on introduction," Georgia Institute of Technology, Tech. Rep. GT-RIM-CP&R-2012-002, September 2012.