Anticipated Feature Selection in Visual-Inertial Odometry

Soumya Sudhakar and Parker Lusk

***Abstract*— The theory of an intelligent feature selection front end for visual-inertial navigation tasks is discussed. By using the attention and anticipation algorithms of Carlone [1], we are able to select features that minimize state estimation uncertainty. Features are selected according to their information content, allowing fewer features to be processed by an optimization back end. Using VINS-Mono, a C++ implementation of the attention and anticipation algorithm is given.**

I. Introduction

During aggressive maneuvers, vision-based perception techniques tend to fail because of the lack of tracked features. This is a well-known issue and is commonly reported in the literature (see for example, [2], [3]). The purpose of this project is to mitigate loss of feature tracks by being more clever in which features to use for graph-based state estimation. This adds robustness to the visual-inertial motion estimation pipeline by disregarding features that are predicted to soon be lost based on future robot motion. Further, we allow for more efficient optimization by using fewer features with high information content.

This project accomplishes these goals by incorporating the attention and anticipation formulation of Carlone [1] with the fixed-lag smoother of VINS-Mono [4], a recent implementation of visual-inertial odometry that has been shown to perform well under power and payload constraints of aerial platforms [5]. The resulting implementation is publicly available[1] and is referred to as `Anticipated VINS-Mono`. A high-level system architecture diagram is shown in Figure 4.

Related to the goal of robust vision-based state estimation during flight is the idea of active vision. An early example of being selective with which features to select was provided by Davison [6]. In his work, multual information is used to extract priors that inform feature tracking algorithms on where to look. Yu and Beard [7] formulate a vision-based collision avoidance technique for MAVs that simultaneously minimizes state estimation uncertainties while avoiding obstacle collisions. Falanga et al. [8] presented a model predictive control (MPC) framework that unifies both control and perception objectives—allowing a multirotor to optimally fulfill its mission and maximize visibility of a point of interest. The ideas of active vision can be summarized by psychologist Eleanor J. Gibson on the human learning behaviors: "we don't simply see, we look" [9].

The rest of this paper is organized as follows. In Section II we discuss concepts related to visual-inertial odometry and an overview of VINS-Mono. In Section III we provide details of our implementation of attention and anticipation. In Section IV we give results. Finally, we conclude in Section V.

[1] `https://github.com/plusk01/Anticipated-VINS-Mono`

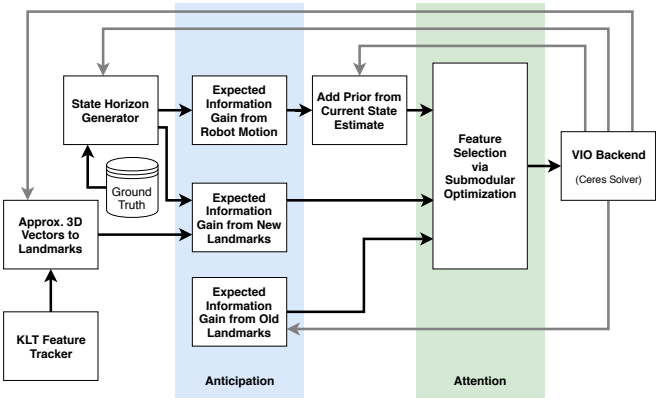

Fig. 1. System architecture of `Anticipated VINS-Mono`. All blocks are new except for the VIO back end, the core VINS-Mono component.

II. Overview of VINS-Mono

VINS-Mono is a state-of-the-art nonlinear optimization-based VIO system and includes a loop closure module for a SLAM system. It is a pose-graph-optimization-based VIO that uses preintegrated IMU factors for its sliding window estimator, and is generally shown to be accurate and efficient [5].

The VINS-Mono system includes modules for initialization, tracking and IMU preintegration, VIO back end, and an optional loop closure module. The entire system is real-time and has been run on-board multirotors.

For this project, the desired characteristics of the VIO system included that it be state-of-the-art, accurate, and potential for research use in the future. VINS-Mono fit this criteria, and is considered to be one of the best monocular VIO systems at this time. Delmerico and Scaramuzza found VINS-Mono to be the most accurate, robust, and consistent across hardware platforms, though also one of the more resource-intensive VIO systems [5]. We decided to use VINS-Mono for the VIO system while adding our components to the system architecture as seen in Figure 4.

III. Attention and Anticipation

The attention and anticipation algorithm [1] selects a subset S of the features F detected in the current frame to pass to the VIO back end. The subset of features should have at most κ features that are the most useful (as a set) for

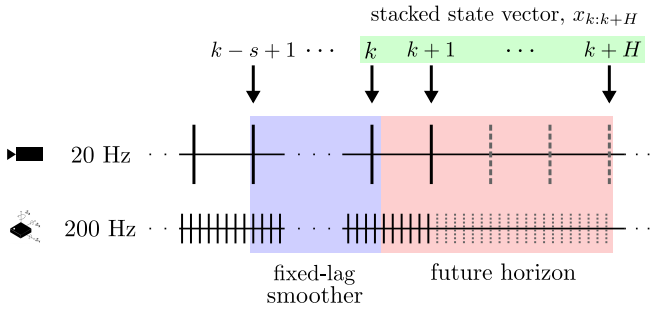

Fig. 2. Timing diagram. Note that between the previous frame k and the current frame $k+1$ at which we are selecting features, we have received IMU measurements. For future frames in the horizon, IMU measurements will be predicted by using the known rotation between frames and dividing by the expected number of IMU measurements between two frames. In our quaternion-based implementation, we use SLERP.

reducing uncertainty in vision-based state estimation. This feature selection problem can be stated as

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

The solution of this problem relies on the selection of a metric f that maps subsets of features to their usefulness.

Although sensor selection problems have been shown to be NP-hard because of the introduction of binary selection variables, recent results have leveraged submodular cost functions to allow greedy algorithms to efficiently find solutions to Problem (1) with guarantees on suboptimality [10]. The goal then is to identify a performance metric that exhibits submodularity and captures the accuracy of VIO.

Following [1], we use the logdet metric to measure the volume of the estimation uncertainty ellipsoid up to scaling. Let $k+1$ be the time at which we have obtained a new feature set to choose from. Then $\boldsymbol{x}_k$ is the optimized pose of the previous camera frame, which is the latest pose estimate from the fixed-lag smoother of the pose-graph optimization back end (see Figure 2). Let $\hat{\boldsymbol{x}}_{k:k+H} \triangleq \begin{bmatrix}\boldsymbol{x}_k & \hat{\boldsymbol{x}}_{k+1} & \cdots & \hat{\boldsymbol{x}}_{k+H}\end{bmatrix}$ denote the stacked state vector over a horizon H, where $\hat{\boldsymbol{x}}_{k+1:k+H}$ are predicted future states yet to be optimized over. Moreover, let $\boldsymbol{P}_{k:k+H}$ be the covariance of the estimation error corresponding to $\hat{\boldsymbol{x}}_{k:k+H}$ and its inverse is called the information matrix, denoted $\boldsymbol{\Omega}_{k:k+H} \triangleq \boldsymbol{P}_{k:k+H}^{-1}$. With these definitions, the logdet metric is written

$$f_{\text{det}}(\mathsf{S}) = \log\det(\boldsymbol{\Omega}_{k:k+H}(\mathsf{S})) \tag{2}$$

$$= \log\det\left(\bar{\boldsymbol{\Omega}}_{k:k+H} + \sum_{l\in\mathsf{S}} p_l \Delta_l\right), \tag{3}$$

where $\bar{\boldsymbol{\Omega}}_{k:k+H}$ is the information matrix corresponding to the predicted robot motion over the horizon and Δ_l is the information matrix of the l-th feature. The p_l is the probability that the l-th feature is tracked, which can come from normalizing the feature detection score. For more information on arriving at this probabilistic formulation, we refer the reader to [1]. The remainder of this section discusses the computation of $\bar{\boldsymbol{\Omega}}_{k:k+H}$ and Δ_l, and the implementation of the greedy selection algorithm.

A. Attention Allocation Algorithm for Feature Selection

The following greedy algorithm is used to approximately solve Problem (1) by selecting κ features that (approximately) maximize the logdet metric. As input, it takes the number of features to select (κ), the anticipated information from future robot motion ($\bar{\boldsymbol{\Omega}}_{k:k+H}$), the anticipated information from each new landmark ($\{\Delta_l\}_{l\in\mathsf{F}}$), and the current information of existing landmarks ($\{\Delta'_l\}_{l\in\mathsf{F}'}$). The following subsections will discuss how to obtain these information matrices.

The greedy algorithm with lazy evaluation involves the following steps. To maintain consistent tracking of features, we keep track of features we have previously determined were informative. When these features are detected in future frames, they are automatically passed through to VINS-Mono. Therefore, κ is chosen such that a certain number of features ($\bar{\kappa}$) is maintained at each time step k and can be calculated as $\min(0, \bar{\kappa} - |\mathsf{F}'|)$.

1) For each iteration over the number of features desired in the subset, upper bounds are computed that bound the information gained by adding each feature to the current subset.
2) After initializing the variables f_{max} and l_{max}, we check if the upper bound is less than the current f_{max} for each landmark in the sorted upper bound list; if so, we break our of the loop and and select the current best feature (lazy evaluation). If the upper bound is greater than the current f_{max}, we calculate the objective function of the new proposed subset by adding that particular feature to the current subset; if it is greater than the current f_{max}, we keep track of this value for the next iteration, ultimately keeping track of the maximum f_{max} for the feature that adds the most information to the current subset.
3) This feature is added to the subset, and the iteration continues until κ features are added to the subset.

Now we will look into how to calculate the information matrices.

B. Anticipated Information from Robot Motion

In a visual-inertial navigation system, IMU measurements are used as inertial constraints between camera frames. While this adds valuable information, it also adds uncertainty due to intrinsic IMU noise and drift. To quantify this uncertainty (also referred to as information) during the expected robot maneuver over the horizon H, we need a model of how the IMU noise is propagated. This information is denoted $\bar{\boldsymbol{\Omega}}^{\text{IMU}}_{k:k+H}$.

As in [1], we assume that the rotation between consecutive frames is known from gyro forward-simulation and integration. Thus, the state at each frame h in the horizon is $\hat{\boldsymbol{x}}_h \triangleq \begin{bmatrix}\boldsymbol{t}_h & \boldsymbol{v}_h & \boldsymbol{b}^a_h\end{bmatrix}$, where $\boldsymbol{t}_h \in \mathbb{R}^3$ and $\boldsymbol{v}_h \in \mathbb{R}^3$ are the inertial position and velocity of the robot (i.e., the IMU frame), and $\boldsymbol{b}^a_h \in \mathbb{R}^3$ is the time-varying accelerometer bias, expressed in the sensor frame. The effect of knowing the rotation is that we can formulate linear models which lessens

the required computation time. By utilizing IMU preintegration theory [11], we efficiently bundle the predicted IMU measurements between consecutive frames (see Figure 2).

Given that accelerometers measure specific force, a measurement $\boldsymbol{a}_i \in \mathbb{R}^3$ received at time i is modeled as

$$\tilde{\boldsymbol{a}}_i = (\boldsymbol{R}^{\mathrm{w}}_{\mathrm{imu},h})^\top (\boldsymbol{a}_i - \boldsymbol{g}) + \boldsymbol{b}^a_h + \boldsymbol{\eta}_h. \tag{4}$$

By integrating the m measurements between two consecutive frames h and h', we arrive at the following equations (c.f., [1, Appendix A])

$$\begin{aligned} \boldsymbol{z}^{\boldsymbol{t}}_{hh'} &= \boldsymbol{t}_{h'} - \boldsymbol{t}_h - \boldsymbol{v}_h m\delta + \boldsymbol{N}_{hh'}\boldsymbol{b}^a_h + \boldsymbol{\eta}^{\boldsymbol{t}}_{hh'} \\ \boldsymbol{z}^{\boldsymbol{v}}_{hh'} &= \boldsymbol{v}_{h'} - \boldsymbol{v}_h + \boldsymbol{M}_{hh'}\boldsymbol{b}^a_h + \boldsymbol{\eta}^{\boldsymbol{v}}_{hh'} \\ \boldsymbol{z}^{\boldsymbol{b}^a}_{hh'} &= \boldsymbol{b}^a_{h'} - \boldsymbol{b}^a_h + \boldsymbol{\eta}^{\boldsymbol{b}^a}_{hh'}, \end{aligned} \tag{5}$$

where $\boldsymbol{z}^{\boldsymbol{t}}_{hh'}, \boldsymbol{z}^{\boldsymbol{v}}_{hh'}, \boldsymbol{z}^{\boldsymbol{b}^a}_{hh'}$ are virtual measurements whose values are not necessary to know, δ is the sampling period of the accelerometer and

$$\begin{aligned} \boldsymbol{N}_{hh'} &\triangleq \textstyle\sum_{i=0}^{m-1}(m-i-\frac{1}{2})\boldsymbol{R}_i\delta^2 \\ \boldsymbol{M}_{hh'} &\triangleq \textstyle\sum_{i=0}^{m-1}\boldsymbol{R}_i\delta \\ \boldsymbol{\eta}^{\boldsymbol{t}}_{hh'} &\triangleq \textstyle\sum_{i=0}^{m-1}(m-i-\frac{1}{2})\boldsymbol{R}_i\boldsymbol{\eta}_i\delta^2 \\ \boldsymbol{\eta}^{\boldsymbol{v}}_{hh'} &\triangleq \textstyle\sum_{i=0}^{m-1}\boldsymbol{R}_i\boldsymbol{\eta}_i\delta. \end{aligned}$$

Note that this summations are IMU-rate: $\boldsymbol{R}_i$ is the incremental rotation from the $i-1$ IMU measurement to the i-th, with $\boldsymbol{R}_0 = \boldsymbol{I}$, and $\boldsymbol{\eta}_i \sim \mathcal{N}(0, \sigma^2_{\mathrm{acc}})$ is the discretized accelerometer noise. The random walk model on the bias is due to the noise $\boldsymbol{\eta}^{\boldsymbol{b}^a}_{hh'} \sim \mathcal{N}(0, \sigma^2_{\boldsymbol{b}^a})$, and we assume that the bias is constant between frames.

We can then write the system of linear equations (5) in the following compact form

$$\boldsymbol{z}^{\mathrm{IMU}}_{hh'} = \boldsymbol{A}_{hh'}\hat{\boldsymbol{x}}_{k:k+H} + \boldsymbol{\eta}^{\mathrm{IMU}}_{hh'}, \tag{6}$$

where

$$\boldsymbol{z}^{\mathrm{IMU}}_{hh'} = \begin{bmatrix} \boldsymbol{z}^{\boldsymbol{t}}_{hh'} \\ \boldsymbol{z}^{\boldsymbol{v}}_{hh'} \\ \boldsymbol{z}^{\boldsymbol{b}^a}_{hh'} \end{bmatrix} \qquad \boldsymbol{\eta}^{\mathrm{IMU}}_{hh'} = \begin{bmatrix} \boldsymbol{\eta}^{\boldsymbol{t}}_{hh'} \\ \boldsymbol{\eta}^{\boldsymbol{v}}_{hh'} \\ \boldsymbol{\eta}^{\boldsymbol{b}^a}_{hh'} \end{bmatrix},$$

and $\boldsymbol{A}_{hh'} \in \mathbb{R}^{9\times 9(H+1)}$ is a matrix of zeros except at the h and h' 9×9 sub-blocks, respectively:

$$\boldsymbol{A}_{hh'} = \left[\begin{array}{c|ccc|c|c} \cdots & -\boldsymbol{I}_3 & -\boldsymbol{I}_3 m\delta & \boldsymbol{N}_{hh'} & \boldsymbol{I}_9 & \cdots \\ & \boldsymbol{0} & -\boldsymbol{I}_3 & \boldsymbol{M}_{hh'} & & \\ & \boldsymbol{0} & \boldsymbol{0} & -\boldsymbol{I}_3 & & \end{array}\right]. \tag{7}$$

As we would like to calculate the information from preintegrating IMU measurements between consecutive frames h and h', we need to compute the covariance of the noise $\boldsymbol{\eta}^{\mathrm{IMU}}_{hh'}$, which is given by

$$(\boldsymbol{\Omega}^{\mathrm{IMU}}_{hh'})^{-1} = \mathrm{cov}(\boldsymbol{\eta}^{\mathrm{IMU}}_{hh'}) = \begin{bmatrix} \sigma^2_{\mathrm{IMU}}\boldsymbol{C}\boldsymbol{C}^\top & \boldsymbol{0}_{6\times 3} \\ \boldsymbol{0}_{3\times 6} & \sigma^2_{\boldsymbol{b}^a} \end{bmatrix}, \tag{8}$$

where

$$\boldsymbol{C}\boldsymbol{C}^\top = \begin{bmatrix} (\sum_{i=0}^{m-1}(m-i-\frac{1}{2})^2)\delta^4\boldsymbol{I} & (\sum_{i=0}^{m-1}(m-i-\frac{1}{2}))\delta^3\boldsymbol{I} \\ (\sum_{i=0}^{m-1}(m-i-\frac{1}{2}))\delta^3\boldsymbol{I} & m\delta^2\boldsymbol{I} \end{bmatrix}$$

Equipped with an expression for a model of the IMU preintegration between two consecutive frames, we can calculate the anticipated information from the IMU over the horizon as

$$\bar{\boldsymbol{\Omega}}^{\mathrm{IMU}}_{k:k+H} = \sum_{h,h'\in\mathsf{H}} (\boldsymbol{A}^\top_{hh'}\boldsymbol{\Omega}^{\mathrm{IMU}}_{hh'}\boldsymbol{A}_{hh'}), \tag{9}$$

where H is the set of consecutive frames over the horizon.

Note that this information matrix $\bar{\boldsymbol{\Omega}}^{\mathrm{IMU}}_{k:k+H}$ consists of only relative measurements (i.e., there is no constraint that "pins" down the information to a global reference). Therefore, $\bar{\boldsymbol{\Omega}}^{\mathrm{IMU}}_{k:k+H}$ is rank deficient. To rectify this, we need to use prior information from the VINS-Mono back end, denoted $\bar{\boldsymbol{\Omega}}^{\mathrm{PRIOR}}_{k:k+H}$, which consists of all zeros except for the upper-left 9×9 sub-block containing the prior from time k. Therefore, the anticipated information from robot motion over the horizon can be calculated as

$$\bar{\boldsymbol{\Omega}}_{k:k+H} = \bar{\boldsymbol{\Omega}}^{\mathrm{IMU}}_{k:k+H} + \bar{\boldsymbol{\Omega}}^{\mathrm{PRIOR}}_{k:k+H}. \tag{10}$$

Implementation Details: Ceres Solver does not have an elegant way to extract marginalized covariances. Because of this, VINS-Mono (in a fashion similar to OKVIS [12]) has created supporting code in the form of a `MarginalizationFactor` to keep track of the sliding optimization window and the necessary residuals and Jacobians. Therefore, it is possible to retrieve $\bar{\boldsymbol{\Omega}}^{\mathrm{PRIOR}}_{k:k+H}$. However, due to lack of time and VINS-Mono comments/documentation, it became difficult to reverse engineer the `MarginalizationFactor` to efficiently extract the properly-ordered residuals to construct the information matrix. In light of this, we choose the upper-left 9×9 block of $\bar{\boldsymbol{\Omega}}^{\mathrm{PRIOR}}_{k:k+H}$ to be $\boldsymbol{I}_9$, which removes the rank deficiency, but does not add any relevant information. To truly take advantage of attention and anticipation, it would be important to successfully (and efficiently) extract this prior.

In calculating $\bar{\boldsymbol{\Omega}}^{\mathrm{IMU}}_{k:k+H}$, we take advantage of the sparsity of each $\boldsymbol{A}_{hh'}$ matrix. For example, for $h=0$ (from frame k to $k+1$), $\boldsymbol{A}^\top_{hh'}\boldsymbol{\Omega}^{\mathrm{IMU}}_{hh'}\boldsymbol{A}_{hh'}$ has the following structure

$$\boldsymbol{A}^\top_{hh'}\boldsymbol{\Omega}^{\mathrm{IMU}}_{hh'}\boldsymbol{A}_{hh'} = \begin{bmatrix} \boldsymbol{A}^\top_{\mathrm{blk}}\boldsymbol{\Omega}^{\mathrm{IMU}}\boldsymbol{A}_{\mathrm{blk}} & \boldsymbol{A}^\top_{\mathrm{blk}}\boldsymbol{\Omega}^{\mathrm{IMU}} & \boldsymbol{0}_9 & \cdots \\ \boldsymbol{\Omega}^{\mathrm{IMU}}\boldsymbol{A}_{\mathrm{blk}} & \boldsymbol{\Omega}^{\mathrm{IMU}} & \boldsymbol{0}_9 & \cdots \\ \boldsymbol{0}_9 & \boldsymbol{0}_9 & \boldsymbol{0}_9 & \cdots \\ \vdots & \vdots & \vdots & \ddots \end{bmatrix},$$

where the subscripts are implied and $\boldsymbol{A}_{\mathrm{blk}} \in \mathbb{R}^{9\times 9}$ is the non-zero, non-identity block of the corresponding $\boldsymbol{A}_{hh'}$. With each successive summing loop of (9), the horizon counter h is incremented and the four 9×9 blocks slide along the main diagonal. After all of these terms are summed, the resulting $\bar{\boldsymbol{\Omega}}^{\mathrm{IMU}}_{k:k+H}$ is a block tri-diagonal matrix. As a result of this insight, we can simply add 9×9 blocks to the appropriate locations of $\bar{\boldsymbol{\Omega}}^{\mathrm{IMU}}_{k:k+H}$. This prevents the inefficiencies of large matrix multiplications and additions which may result in round-off error.

C. Anticipated Information from Visual Features

As before, we wish to use a linear measurement model which simplifies the nonlinear perspective projection model and allows for efficient computation. Let $\boldsymbol{u}_{hl} \in \mathbb{R}^3$ be the normalized bearing vector of the l-th landmark with respect to the camera pose at time h in the horizon (i.e., $k+1 \leq h \leq k+H$). Then a linear measurement model can be written using the colinearity of the landmark vector as

$$[\boldsymbol{u}_{hl}]_\times ((\boldsymbol{R}^{\text{w}}_{\text{cam},h})^\top (\boldsymbol{p}^{\text{w}}_l - \boldsymbol{t}^{\text{w}}_{\text{cam},h})) = \boldsymbol{0}_3, \tag{11}$$

where $\boldsymbol{p}^{\text{w}}_l$ is the position of the landmark with respect to the world frame and $(\boldsymbol{R}^{\text{w}}_{\text{cam},h}, \boldsymbol{t}^{\text{w}}_{\text{cam},h})$ is the predicted pose of the camera at time h in the horizon. Our objective is to form a linear system in state-space form that is a function of our state horizon $\hat{\boldsymbol{x}}_{k:k+H}$. This will allow us to think about the problem as a maximum likelihood estimation problem so that we can extract the anticipated information gain of the feature given $\hat{\boldsymbol{x}}_{k:k+H}$.

Since our state at each time step in the horizon is defined as $\hat{\boldsymbol{x}}_h = \begin{bmatrix} \boldsymbol{t}_h & \boldsymbol{v}_h & \boldsymbol{b}^a_h \end{bmatrix}$, where $\boldsymbol{t}_h$ and $\boldsymbol{v}_h$ are the position and velocity of the IMU frame with respect to the world, we need our linear system to be parameterized by those quantities. Luckily, the extrinsic transformation between the camera and IMU is known from calibration and we can write

$$[\boldsymbol{u}_{hl}]_\times ((\boldsymbol{R}^{\text{w}}_{\text{imu},h} \boldsymbol{R}^{\text{imu}}_{\text{cam}})^\top (\boldsymbol{p}^{\text{w}}_l - (\boldsymbol{t}^{\text{w}}_{\text{imu},h} + \boldsymbol{R}^{\text{w}}_{\text{imu},h} \boldsymbol{t}^{\text{imu}}_{\text{cam}}))) = \boldsymbol{\eta}^{\text{cam}}_{hl},$$

where $\boldsymbol{\eta}^{\text{cam}}_{hl} \sim \mathcal{N}(0, \Sigma_{\text{cam}})$ introduces noise. Rearranging terms, we can construct the following virtual measurement model for the l-th landmark viewed from the h-th frame

$$\boldsymbol{z}^{\text{cam}}_{hl} = [\boldsymbol{u}_{hl}]_\times (\boldsymbol{R}^{\text{w}}_{\text{imu},h} \boldsymbol{R}^{\text{imu}}_{\text{cam}})^\top (\boldsymbol{t}^{\text{w}}_{\text{imu},h} - \boldsymbol{p}^{\text{w}}_l) + \boldsymbol{\eta}^{\text{cam}}_{hl},$$

where $\boldsymbol{z}^{\text{cam}}_{hl} = [\boldsymbol{u}_{hl}]_\times (\boldsymbol{R}^{\text{imu}}_{\text{cam}})^\top \boldsymbol{t}^{\text{imu}}_{\text{cam}}$ is the irrelevant virtual measurement. Written compactly, we have

$$\boldsymbol{z}^{\text{cam}}_{hl} = \boldsymbol{F}_{hl} \hat{\boldsymbol{x}}_{k:k+H} + \boldsymbol{E}_{hl} \boldsymbol{p}^{\text{w}}_l + \boldsymbol{\eta}^{\text{cam}}_{hl}, \tag{12}$$

where $\boldsymbol{F}_{hl} \in \mathbb{R}^{3 \times 9(H+1)}$ is a matrix of zeros except for at the h-th 3×9 sub-block corresponding to feature visibility at camera frame h

$$\boldsymbol{F}_{hl} = \begin{bmatrix} \cdots \mid [\boldsymbol{u}_{hl}]_\times (\boldsymbol{R}^{\text{w}}_{\text{imu},h} \boldsymbol{R}^{\text{imu}}_{\text{cam}})^\top & \boldsymbol{0}_{3\times 6} \mid \cdots \end{bmatrix},$$

and $\boldsymbol{E}_{hl} \in \mathbb{R}^{3\times 3}$ is

$$\boldsymbol{E}_{hl} = -[\boldsymbol{u}_{hl}]_\times (\boldsymbol{R}^{\text{w}}_{\text{imu},h} \boldsymbol{R}^{\text{imu}}_{\text{cam}})^\top.$$

Now that we have a linear measurement model of the calibrated feature pixel as a function of the state, our goal is to understand where in the image each landmark detected at time $k+1$ can be found. The intuition is if the l-th feature is expected to be in view over the entire future horizon, then it will have a higher amount of information than a feature that is expected to be quickly lost. This requires us to project the l-th landmark onto the image plane of the camera of the robot at future times h over the horizon. As before, we assume that the rotations of the robot at time h is known from the gyros. We also require the predicted positions of the robot over the horizon, which are generated by the *State Horizon Generator*. In addition to forward simulation of the camera model at each predicted pose $\hat{\boldsymbol{x}}_h$, a visibility check is performed to determine if the landmark could be seen by this future pose. The forward simulation and visibility checking is illustrated in Figure 3.

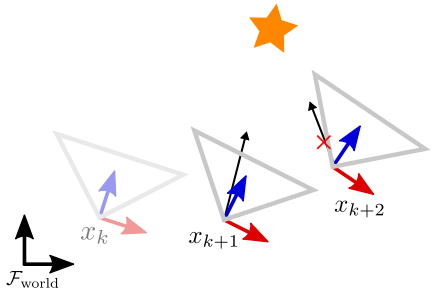

Fig. 3. Forward propagation of the bearing vector to landmark l, originally detected at $\hat{\boldsymbol{x}}_{k+1}$. Although $\hat{\boldsymbol{x}}_{k:k+H}$ contains the previous pose $\boldsymbol{x}_k$, the landmark was not detected there and so it is not considered. In this illustration, the predicted feature vector at $\hat{\boldsymbol{x}}_{k+2}$ lies outside the field of view and so fails the visibility check.

By stacking (12) row-wise for each time step in the horizon, we have the following compact model for how the l-th landmark detected at $k+1$ propagates throughout the horizon

$$\boldsymbol{z}^{\text{cam}}_l = \boldsymbol{F}_l \hat{\boldsymbol{x}}_{k:k+H} + \boldsymbol{E}_l \boldsymbol{p}^{\text{w}}_l + \boldsymbol{\eta}^{\text{cam}}_l, \tag{13}$$

where $\boldsymbol{z}^{\text{cam}}_l \in \mathbb{R}^{3H}$, $\boldsymbol{F}_l \in \mathbb{R}^{3H \times 9(H+1)}$, and $\boldsymbol{E}_l \in \mathbb{R}^{3H\times 3}$. However, in order for a point to be useful, it must be able to be triangulated (i.e., it must be seen from multiple time steps). Any rows that correspond to poses for which the landmark was not seen are removed to prevent rank deficiency. The final dimensions are then $\boldsymbol{z}^{\text{cam}}_l \in \mathbb{R}^{3n_\ell}$, $\boldsymbol{F}_l \in \mathbb{R}^{3n_\ell \times 9(H+1)}$, and $\boldsymbol{E}_l \in \mathbb{R}^{3n_\ell \times 3}$, where n_ℓ is the number of frames that landmark l is visible.

We wish to identify the amount of information that the landmark $\boldsymbol{p}^{\text{w}}_l$ adds to the state estimation task. Because $\boldsymbol{p}^{\text{w}}_l$ is not part of our state vector $\hat{\boldsymbol{x}}_{k:k+H}$, we first compute the information matrix of the joint state $\begin{bmatrix} \hat{\boldsymbol{x}}_{k:k+H} & \boldsymbol{p}^{\text{w}}_l \end{bmatrix}$ and then use the Schur complement to marginalize out the dependence on $\boldsymbol{p}^{\text{w}}_l$ and dispersing its information into the other states:

$$\boldsymbol{\Omega}^{(l)}_{k:k+H} = \begin{bmatrix} \boldsymbol{F}_l^\top \boldsymbol{F}_l & \boldsymbol{F}_l^\top \boldsymbol{E}_l \\ \boldsymbol{E}_l^\top \boldsymbol{F}_l & \boldsymbol{E}_l^\top \boldsymbol{E}_l \end{bmatrix} \in \mathbb{R}^{9(H+1)+3 \times 9(H+1)+3}.$$

The Schur complement of $\boldsymbol{\Omega}^{(l)}_{k:k+H}$ is computed as

$$\Delta_l = \boldsymbol{F}_l^\top \boldsymbol{F}_l - \boldsymbol{F}_l^\top \boldsymbol{E}_l (\boldsymbol{E}_l^\top \boldsymbol{E}_l)^{-1} \boldsymbol{E}_l^\top \boldsymbol{F}_l \in \mathbb{R}^{9(H+1)\times 9(H+1)}, \tag{14}$$

which gives the additive contribution of anticipated information from the l-th feature to our state estimate and is used in the greedy algorithm.

Implementation Details: Noting that the matrices in (13) are sparse, we can avoid large matrix multiplication by exploiting the sparsity patterns and reusing computation where possible. The stacked matrices $\boldsymbol{F}_l$ and $\boldsymbol{E}_l$ have the

following structure (before removing degenerate rows):

$$\boldsymbol{F}_l = \begin{bmatrix} \mathbf{0}_{3\times9} & \boldsymbol{A}_{k+1,l} & \mathbf{0}_{3\times9} & \cdots & \mathbf{0}_{3\times9} \\ \mathbf{0}_{3\times9} & \mathbf{0}_{3\times9} & \boldsymbol{A}_{h,l} & \cdots & \mathbf{0}_{3\times9} \\ \mathbf{0}_{3\times9} & \mathbf{0}_{3\times9} & \mathbf{0}_{3\times9} & \ddots & \mathbf{0}_{3\times9} \\ \mathbf{0}_{3\times9} & \mathbf{0}_{3\times9} & \mathbf{0}_{3\times9} & \cdots & \boldsymbol{A}_{k+H,l} \end{bmatrix} \tag{15}$$

$$\boldsymbol{E}_l = \begin{bmatrix} -\boldsymbol{B}_{k+1,l} \\ -\boldsymbol{B}_{h,l} \\ \vdots \\ -\boldsymbol{B}_{k+H,l} \end{bmatrix}, \tag{16}$$

where

$$\boldsymbol{A}_{h,l} = \begin{bmatrix} \boldsymbol{B}_{h,l} & \mathbf{0}_{3\times6} \end{bmatrix} \in \mathbb{R}^{3\times9} \tag{17}$$

$$\boldsymbol{B}_{h,l} = [\boldsymbol{u}_{hl}]_\times (\boldsymbol{R}^{\text{w}}_{\text{imu},h} \boldsymbol{R}^{\text{imu}}_{\text{cam}})^\top \in \mathbb{R}^{3\times3}. \tag{18}$$

Computing the information matrix of the joint state $\begin{bmatrix} \hat{\boldsymbol{x}}_{k:k+H} & \boldsymbol{p}^{\text{w}}_l \end{bmatrix}$ gives the following sub-blocks, where we have taken advantage of the sparsity of $\boldsymbol{A}_{h,l}$. For simplicity of notation, we drop the subscripts of $\boldsymbol{A}_{h,l}$ and $\boldsymbol{B}_{h,l}$ and simply use $1,\dots,H$ to denote $k+1,\dots,H$:

$$\boldsymbol{F}_l^\top \boldsymbol{F}_l = \begin{bmatrix} \mathbf{0}_{9\times9} & \mathbf{0}_{9\times9} & \cdots & \mathbf{0}_{9\times9} \\ \mathbf{0}_{9\times9} & \boldsymbol{A}_1^\top \boldsymbol{A}_1 & \cdots & \mathbf{0}_{9\times9} \\ \mathbf{0}_{9\times9} & \mathbf{0}_{9\times9} & \ddots & \mathbf{0}_{9\times9} \\ \mathbf{0}_{9\times9} & \mathbf{0}_{9\times9} & \mathbf{0}_{9\times9} & \boldsymbol{A}_H^\top \boldsymbol{A}_H \end{bmatrix}$$

$$= \begin{bmatrix} \mathbf{0}_{9\times9} & \mathbf{0}_{9\times3} & \mathbf{0}_{9\times6} & \cdots & \mathbf{0}_{9\times3} & \mathbf{0}_{9\times6} \\ \mathbf{0}_{3\times9} & \boldsymbol{B}_1^\top \boldsymbol{B}_1 & \mathbf{0}_{3\times6} & \cdots & \mathbf{0}_{3\times3} & \mathbf{0}_{3\times6} \\ \mathbf{0}_{6\times9} & \mathbf{0}_{6\times3} & \mathbf{0}_{6\times6} & \cdots & \mathbf{0}_{6\times3} & \mathbf{0}_{6\times6} \\ & & & \ddots & & \\ \mathbf{0}_{3\times9} & \mathbf{0}_{3\times3} & \mathbf{0}_{3\times6} & \cdots & \boldsymbol{B}_H^\top \boldsymbol{B}_H & \mathbf{0}_{3\times6} \\ \mathbf{0}_{6\times9} & \mathbf{0}_{6\times3} & \mathbf{0}_{6\times6} & \cdots & \mathbf{0}_{6\times3} & \mathbf{0}_{6\times6} \end{bmatrix}$$

$$\boldsymbol{F}_l^\top \boldsymbol{E}_l = \begin{bmatrix} \mathbf{0}_{9\times3} \\ -\boldsymbol{A}_1^\top \boldsymbol{B}_1 \\ \vdots \\ -\boldsymbol{A}_H^\top \boldsymbol{B}_H \end{bmatrix} = \begin{bmatrix} \mathbf{0}_{9\times3} \\ -\boldsymbol{B}_1^\top \boldsymbol{B}_1 \\ \mathbf{0}_{6\times3} \\ \vdots \\ -\boldsymbol{B}_H^\top \boldsymbol{B}_H \\ \mathbf{0}_{6\times3} \end{bmatrix} = (\boldsymbol{E}_l^\top \boldsymbol{F}_l)^\top$$

$$\boldsymbol{E}_l^\top \boldsymbol{E}_l = \begin{bmatrix} \boldsymbol{B}_1^\top \boldsymbol{B}_1 + \cdots + \boldsymbol{B}_H^\top \boldsymbol{B}_H \end{bmatrix}$$

With an understanding of the pattern that these matrices exhibit, we can then efficiently calculate Δ_l as follows. First, we note that the second addend in the Schur complement in (14) has the following form:

$$\boldsymbol{F}_l^\top \boldsymbol{E}_l (\boldsymbol{E}_l^\top \boldsymbol{E}_l)^{-1} \boldsymbol{E}_l^\top \boldsymbol{F}_l = \begin{bmatrix} \mathbf{0}_{9\times9} & \mathbf{0}_{9\times3} & \mathbf{0}_{9\times6} & \cdots & \mathbf{0}_{9\times3} & \mathbf{0}_{9\times6} \\ \mathbf{0}_{3\times9} & \boldsymbol{C}_1 \boldsymbol{W} \boldsymbol{C}_1^\top & \mathbf{0}_{3\times6} & \cdots & \boldsymbol{C}_1 \boldsymbol{W} \boldsymbol{C}_H^\top & \mathbf{0}_{3\times6} \\ \mathbf{0}_{6\times9} & \mathbf{0}_{6\times3} & \mathbf{0}_{6\times6} & \cdots & \mathbf{0}_{6\times3} & \mathbf{0}_{6\times6} \\ & \vdots & & \ddots & \vdots & \\ \mathbf{0}_{3\times9} & \boldsymbol{C}_H \boldsymbol{W} \boldsymbol{C}_1^\top & \mathbf{0}_{3\times6} & \cdots & \boldsymbol{C}_H \boldsymbol{W} \boldsymbol{C}_H^\top & \mathbf{0}_{3\times6} \\ \mathbf{0}_{6\times9} & \mathbf{0}_{6\times3} & \mathbf{0}_{6\times6} & \cdots & \mathbf{0}_{6\times3} & \mathbf{0}_{6\times6} \end{bmatrix},$$

where $\boldsymbol{C}_i \triangleq \boldsymbol{B}_i^\top \boldsymbol{B}_i$ and $\boldsymbol{W} \triangleq (\boldsymbol{E}_l^\top \boldsymbol{E}_l)^{-1}$ (which is only invertible if a landmark was seen by more than one future pose). Therefore, the final form of Δ_l is

$$\Delta_l = \tag{19}$$
$$\begin{bmatrix} \mathbf{0}_{9\times9} & \mathbf{0}_{9\times3} & \mathbf{0}_{9\times6} & \cdots & \mathbf{0}_{9\times3} & \mathbf{0}_{9\times6} \\ \mathbf{0}_{3\times9} & \boldsymbol{C}_1 - \boldsymbol{D}_{11} & \mathbf{0}_{3\times6} & \cdots & -\boldsymbol{D}_{1H} & \mathbf{0}_{3\times6} \\ \mathbf{0}_{6\times9} & \mathbf{0}_{6\times3} & \mathbf{0}_{6\times6} & \cdots & \mathbf{0}_{6\times3} & \mathbf{0}_{6\times6} \\ & \vdots & & \ddots & \vdots & \\ \mathbf{0}_{3\times9} & -\boldsymbol{D}_{H1} & \mathbf{0}_{3\times6} & \cdots & \boldsymbol{C}_H - \boldsymbol{D}_{HH} & \mathbf{0}_{3\times6} \\ \mathbf{0}_{6\times9} & \mathbf{0}_{6\times3} & \mathbf{0}_{6\times6} & \cdots & \mathbf{0}_{6\times3} & \mathbf{0}_{6\times6} \end{bmatrix},$$

where $\boldsymbol{D}_{ij} \triangleq \boldsymbol{C}_i \boldsymbol{W} \boldsymbol{C}_j^\top$. Finally, we note that when the camera at pose $\hat{\boldsymbol{x}}_h$ did not see the l-th landmark, the corresponding rows and columns are zeroed out because that landmark offers no information for those state estimates.

IV. Results

To test our implementation, we used the EuRoC datasets [13]. We selected an easy case and hard case of the EuRoC datasets to test the algorithm, MH_02 and MH_04. Timing data for the new modules in Anticipated VINS-Mono is given in Table III. Running this algorithm against VINS-Mono without anticipated feature selection would not be a good comparison since VINS-Mono can select up to 150 features and would be expected to do better than any method selecting fewer features. In order to have a benchmark to test the attention algorithm against, we made two test cases: *quality* and *random*. In *quality*, we run VINS-Mono without the anticipation algorithm, but limit the maximum features it can select to the size of κ we use for the attention algorithm. For *random*, we conduct feature selection randomly, choosing at random from the set of new features seen to satisfy the same κ constraint. The results are given in Figure I.

We can see that Anticipated VINS-Mono does better than both *quality* and *random* in certain cases. With the 10 features on the easier dataset, Anticipated VINS-Mono does better than *random* and better than *quality* as quality ended up being unstable. With 30 features, Anticipated VINS-Mono does better than *quality* and *random* again, with lower absolute translational error (ATE) and rotational translational error (RTE). However, Anticipated VINS-Mono does not perform well on MH_05, the more difficult dataset, whereas *quality* still produces a meaningful pose estimate.

There are a few reasons that could explain why Anticipated VINS-Mono does not work perfectly. Calculating $\boldsymbol{\Omega}^{\text{PRIOR}}_{k:k+H}$ was not implemented in the algorithm itself due to time constraints in retrieving it from VINS-Mono backend. Including p_l was also not included due to time constraints. At times, the processed poses showed that the pose (and error) was unstable and kept increasing (e.g., in Anticipated VINS-Mono run on MH_05 with 30 features). Moreover, we did not look at the VINS-Mono back end optimizer, and it is possible that Anticipated VINS-Mono could be improved there.

Parameters used are given in Table II with a timing summary in Table III[2].

[2]Video results can be seen at `https://www.youtube.com/watch?v=OOzhGPaxFd8`.

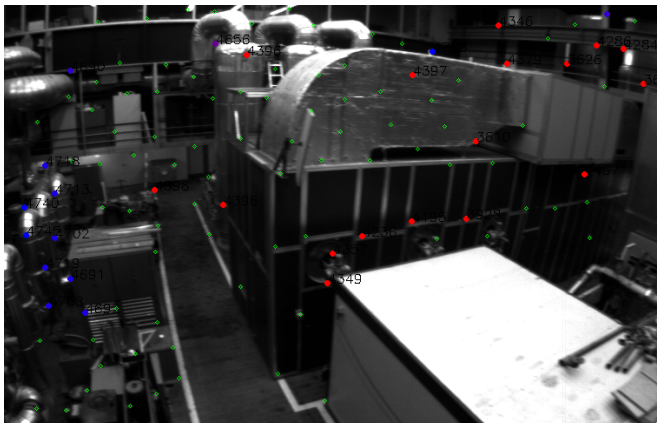

Fig. 4. Snapshot of EuRoC `MH_05_difficult` during a sharp left turn. Possible selections (new detections) are shown in green, selected features in blue, and tracked features in red. Notice how Anticipated VINS-Mono selects the majority of its features on the left side of the frame.

TABLE I

RESULTS OF ANTICIPATED VINS-MONO ON EUROC DATASETS AGAINST BENCHMARKS *quality* AND *random*

		RTE (m)	RTE - med. (m)	ATE (m)
MH_02 (10)	Anticipate	4.589	0.48	2.19
MH_02 (10)	Quality	0.887	0.653	15850
MH_02 (10)	Random	4.60	0.4801	1.934
MH_02 (30)	Anticipate	4.595	0.288	0.2021
MH_02 (30)	Quality	4.605	0.2925	0.2632
MH_02 (30)	Random	4.594	0.3058	0.3063
MH_05 (30)	Anticipate	2.23	1.097	10881
MH_05 (30)	Quality	2.2	1.07	7.874

V. CONCLUSION

Overall, our contributions are a C++ implementation of the attention and anticipation paper [1] and an integration of this algorithm with the state-of-the-art VIO system, VINS-Mono. The code is open-source and available on GitHub.

Future work involves adding $\mathbf{\Omega}^{\text{PRIOR}}_{k:k+H}$ to the algorithm and looking at the VINS-Mono to test if this improves performance. Moreover, future avenues for research include looking at how control can be incorporated rather than using the ground truth for future state horizon and looking at the dual problem of minimizing the features used to achieve performance requirements which would be very useful in the resource-constrained scenario.

TABLE II

PARAMETERS

Parameter	Value
Frame rate	10 Hz
Horizon	10 frames
Max features to detect	150
Features to maintain, $\bar{\kappa}$	30

TABLE III

TIMING STATISTICS ON EUROC MH_05_DIFFICULT

Thread	Component	Time (ms)
1	Feature Tracker	18
2	Feature Selector	9
	Windowed Optimization (1 sec)	30

REFERENCES

[1] L. Carlone and S. Karaman, "Attention and Anticipation in Fast Visual-Inertial Navigation," in *IEEE International Conference on Robotics and Automation (ICRA)*, Singapore, May 2017, pp. 3886–3893. [Online]. Available: http://ieeexplore.ieee.org/document/7989448/

[2] S. Shen, Y. Mulgaonkar, N. Michael, and V. Kumar, "Vision-based state estimation and trajectory control towards high-speed flight with a quadrotor," in *Proceedings of Robotics: Science and Systems*, Berlin, Germany, June 2013.

[3] D. Falanga, E. Mueggler, M. Faessler, and D. Scaramuzza, "Aggressive quadrotor flight through narrow gaps with onboard sensing and computing using active vision," in *IEEE International Conference on Robotics and Automation (ICRA)*, Singapore, May 2017, pp. 5774–5781.

[4] T. Qin, P. Li, and S. Shen, "VINS-Mono: A Robust and Versatile Monocular Visual-Inertial State Estimator," *IEEE Transactions on Robotics*, vol. 34, no. 4, pp. 1004–1020, Aug. 2018.

[5] J. Delmerico and D. Scaramuzza, "A benchmark comparison of monocular visual-inertial odometry algorithms for flying robots," in *IEEE International Conference on Robotics and Automation (ICRA)*, Brisbane, QLD, May 2018, pp. 2502–2509.

[6] A. J. Davison, "Active search for real-time vision," in *IEEE International Conference on Computer Vision (ICCV)*, Beijing, Oct 2005, pp. 66–73.

[7] H. Yu and R. W. Beard, "Vision-based local-level frame mapping and planning in spherical coordinates for miniature air vehicles," *IEEE Transactions on Control Systems Technology*, vol. 21, no. 3, pp. 695–703, 2013.

[8] D. Falanga, E. Mueggler, M. Faessler, and D. Scaramuzza, "Pampc: Perception-aware model predictive control for quadrotors," in *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, Madrid, Oct 2018.

[9] E. J. Gibson, "Exploratory behavior in the development of perceiving, acting, and the acquiring of knowledge," *Annual Review of Psychology*, vol. 39, no. 1, pp. 1–42, 1988.

[10] M. Shamaiah, S. Banerjee, and H. Vikalo, "Greedy sensor selection: Leveraging submodularity," in *IEEE Conference on Decision and Control (CDC)*, Atlanta, GA, Dec 2010, pp. 2572–2577.

[11] C. Forster, L. Carlone, F. Dellaert, and D. Scaramuzza, "On-manifold preintegration for real-time visual–inertial odometry," *Trans. Rob.*, vol. 33, no. 1, pp. 1–21, Feb. 2017. [Online]. Available: https://doi.org/10.1109/TRO.2016.2597321

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

[13] M. Burri, J. Nikolic, P. Gohl, T. Schneider, J. Rehder, S. Omari, M. W. Achtelik, and R. Siegwart, "The euroc micro aerial vehicle datasets," *The International Journal of Robotics Research*, 2016. [Online]. Available: http://ijr.sagepub.com/content/early/2016/01/21/0278364915620033.abstract