PROBE: Predictive Robust Estimation for Visual-Inertial Navigation

Valentin Peretroukhin, Lee Clement, Matthew Giamou, and Jonathan Kelly

Abstract— Localization and mapping within unknown, chaotic environments continue to present significant challenges for the robotics community. Lighting changes, self-similar textures, motion blur, and moving objects are all considerable stumbling blocks for state-of-the-art visual and visual-inertial navigation algorithms. In this work we present a novel technique to improve localization quality within an optimization-based visual-inertial navigation system (VINS). We make use of training data to learn a model for the quality of visual features with respect to localization error in a given environment. This model maps observations from a pre-defined *prediction space* of visual-inertial *predictors* onto a scalar weight, which is then applied to the covariance matrix of each observation, effectively predicting its influence on the final state estimate. We discuss our choice of predictor space and report substantial improvements in localization error on 4 km of data from the KITTI dataset, as well as on experimental datasets consisting of roughly 700 m of indoor and outdoor driving on a small ground rover equipped with a Skybotix VI-Sensor.

I. INTRODUCTION

Robotic navigation relies on an accurate quantification of sensor uncertainty in order to produce reliable state estimates. In practice, this uncertainty is often fixed for a given sensor and experiment, whether by automatic calibration or by manual tuning. Although a fixed measure of uncertainty may be reasonable in certain static environments, dynamic scenes often contain many effects that corrupt some of the available observations. For visual sensors, these effects include, for example, self-similar textures, variations in lighting, moving objects, and motion blur. In this paper we present PROBE, a Predictive ROBust Estimation technique that improves localization quality in the presence of such effects by building a model of the uncertainty in the affected visual observations. We learn the model in an offline training procedure and then use it online to predict the uncertainty of incoming observations as a function of their location in a pre-defined *prediction space*. Our model can be learned in completely unknown environments with frequent or infrequent ground truth data.

The primary contributions of this research are a flexible framework for learning the quality of visual features with respect to navigation estimates, and a straightforward way to incorporate this information into a navigation pipeline. On its own, PROBE can produce more accurate estimates than a binary outlier rejection scheme like Random Sample Consensus (RANSAC) [1] because it can simultaneously reduce the influence of outliers while intelligently weighting

[1]All authors are at the Institute for Aerospace Studies, University of Toronto, Canada. {v.peretroukhin, lee.clement, matthew.giamou}@mail.utoronto.ca, jkelly@utias.utoronto.ca.
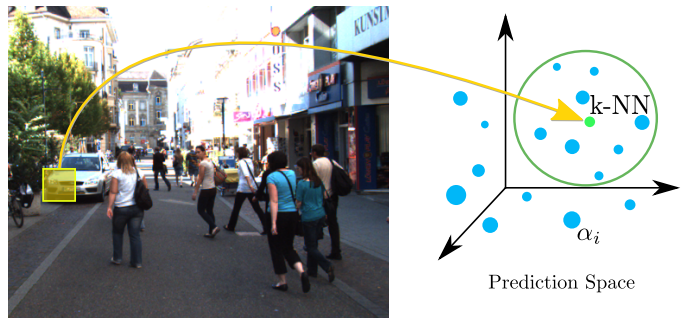
Fig. 1. PROBE maps image features into a prediction space to predict feature quality, reject outliers, and scale feature covariances.

inliers. PROBE reduces the need to develop finely-tuned uncertainty models for complex sensors such as cameras, and better accounts for the effects of complex, dynamic scenes than typical fixed-uncertainty models. While we present PROBE in the context of visual feature-based navigation, we stress that it is not limited to visual measurements and can also, in principle, be applied to other sensor modalities.

II. RELATED WORK

Two important tasks must be performed to obtain accurate estimates from navigation systems: sensor noise characterization and outlier rejection. Standard calibration techniques can provide some insight into sensor noise, but in practice, noise parameters are often manually tuned to optimize estimator performance for a given environment. Outlier rejection typically relies on front-end procedures such as Random Sample Consensus (RANSAC) [1], [2], which attempts to remove outliers before the estimator can process them, and back-end procedures such as M-estimation, which uses robust cost functions to limit the influence of large errors [3].

In visual systems, a large body of literature has also focused on compensating for specific effects such as moving objects [4], lighting variations [5], and motion blur [6] using carefully crafted models. Although such models can work well for specific phenomena, they are limited in their scope and often must rely on parallel computation to achieve online performance.

A more general approach for selecting an optimal set of features has also been investigated in the literature. In [7], visual features are parametrized in an attribute space that is used both to establish feature correspondences and to learn a model of feature reliability for pose estimation. Sufficiently reliable features are stored for use as landmarks during pose estimation while unreliable features are discarded. In a similar vein, our method maps features in a prediction space

to a scalar weight, but we use this weight in our estimator rather than to discard data. In more recent work [8], the authors forgo the concept of outliers in favour of finding the largest subset of coherent measurements. Once again, our method differs in that it does not discard any measurements.

Our work on predicting the general quality of visual features is inspired by recent research on predictive co-variance estimation [9], which estimates full measurement covariances by collecting empirical errors and storing them in a prediction space. For cameras, learning measurement covariances requires ground truth of the observed quantities, which is difficult to obtain for sparse visual landmarks. To relax this requirement, the authors of [10] describe an expectation-maximization technique that can serve as a proxy for ground truth, but it is unclear whether this technique is applicable to sparse vision-based navigation. In contrast, our method requires ground truth for only a subset of position states and can be straightforwardly applied to standard sparse visual navigation systems.

Finally, adaptive techniques exist that learn scalar weights to intelligently modify the influence of particular measure-ments within a Kalman filter estimator [11]. We adopt this approach for the optimization-based visual-inertial naviga-tion pipeline, but do so predictively rather than reactively to respond to new measurements with minimal time delay.

III. VISUAL-INERTIAL NAVIGATION SYSTEM

In this work, we evaluate PROBE as a way to improve navigation estimates from visual-inertial navigation systems (VINS). VINS can be implemented on many different robotic platforms including micro aerial vehicles (MAVs), ground vehicles, smartphones, and even wearable devices such as Google Glass [12]. Stereo cameras [13] and monocular cameras [14], [15] are common sensor choices for acquiring visual data in such systems.

In this work, we implement a stereo VINS adapted from a common stereo visual odometry (VO) framework used in numerous successful mobile robotics applications [16]–[18]. Unlike other VINS [13], [14], we choose not to use the linear accelerations reported by the IMU to drive a motion model. Instead, we opt to use only angular velocities to extract rotation estimates between successive poses.

Although this choice reduces the potential accuracy of the method, it also confers several advantages that make our pipeline a solid foundation from which to evaluate PROBE. Specifically, we do not need to keep track of a linear velocity state, acceleration biases, or gravity orientation. Although seemingly minor, this simplification greatly reduces the com-plexity of the pipeline, obviates the need for a special starting state that is typically required to make gravity observable, and eliminates several sensitive tuning parameters.

We use the relatively stable rotational rates reported by commercial IMUs to extract accurate relative rotation esti-mates over short time periods. Similar to [19], we bootstrap a crude transformation estimate with the rotation estimate before applying a full non-linear point cloud alignment.

Finally, because we assume a stereo camera, the metric scale of our solution is directly observable from the visual measurements, whereas monocular VINS must rely on noisy linear accelerations to determine metric scale [20].

A. Observation Model

To begin, we outline our point feature observation model. We define two coordinate frames, $\underrightarrow{\mathcal{F}}_a$ and $\underrightarrow{\mathcal{F}}_b$, which represent the stereo camera pose at time steps a and b. The coordinates $\mathbf{p}_a$ of a landmark observed in $\underrightarrow{\mathcal{F}}_a$ can be transformed into its coordinates $\mathbf{p}_b$ in $\underrightarrow{\mathcal{F}}_b$ as follows:

$$\mathbf{p}_b = \mathbf{C}_{ba}(\mathbf{p}_a - \mathbf{r}_a^{ba}). \tag{1}$$

$\mathbf{r}_a^{ba}$ is a vector from the origin of $\underrightarrow{\mathcal{F}}_a$ to the origin of $\underrightarrow{\mathcal{F}}_b$ expressed in $\underrightarrow{\mathcal{F}}_a$, and $\mathbf{C}_{ba}$ is the rotation from $\underrightarrow{\mathcal{F}}_a$ to $\underrightarrow{\mathcal{F}}_b$.

Assuming rectified stereo images, points $\mathbf{p}$ are projected from the camera frame (with its origin in the centre of the left camera) into image coordinates $\mathbf{y}$ in the left and right cameras according to the camera model

$$\mathbf{y} = \mathbf{f}(\mathbf{p}) = \begin{bmatrix} u_l \\ v_l \\ u_r \\ v_r \end{bmatrix} = \frac{f}{p_3} \begin{bmatrix} p_1 \\ p_2 \\ p_1 - b \\ p_2 \end{bmatrix} + \begin{bmatrix} c_u \\ c_v \\ c_u \\ c_v \end{bmatrix}. \tag{2}$$

Here, p_i are the components of $\mathbf{p}$, (u_l, u_r, v_l, v_r) are the horizontal and vertical pixel coordinates in each camera, b is the baseline of the stereo pair, (c_u, c_v) is the principal point of each camera, f is the focal length of each camera.

B. Direct Motion Solution

Assuming all landmarks are tracked correctly, our goal is to calculate the transformation between $\underrightarrow{\mathcal{F}}_a$ and $\underrightarrow{\mathcal{F}}_b$, parametrized by $\mathbf{C}_{ba}$ and $\mathbf{r}_a^{ba}$. We obtain an estimate for $\mathbf{C}_{ba}$ by integrating angular velocities, $\boldsymbol{\omega}$, from the IMU. Defining $\boldsymbol{\psi}_k := \boldsymbol{\omega}\Delta t$, $\psi_k := |\boldsymbol{\psi}_k|$, and $\hat{\boldsymbol{\psi}}_k := \boldsymbol{\psi}_k/\psi_k$, we have

$$\boldsymbol{\Psi}_i = \cos\psi_k \mathbf{1} + (1 - \cos\psi_k)\hat{\boldsymbol{\psi}}_k\hat{\boldsymbol{\psi}}_k^T - \sin\psi_k \hat{\boldsymbol{\psi}}_k^\times \tag{3}$$

$$\overline{\mathbf{C}}_{ba} = \mathbf{C}_{ci}\boldsymbol{\Psi}_k \cdots \boldsymbol{\Psi}_2\boldsymbol{\Psi}_1\mathbf{C}_{ci}^T \tag{4}$$

where k is the number of IMU measurements between two successive camera images, $\mathbf{C}_{ci}$ is the rotation from the IMU frame into the camera frame, and $\mathbf{1}$ is the identity matrix. Given $\overline{\mathbf{C}}_{ba}$, we can quickly calculate the translation

$$\overline{\mathbf{r}}_a^{ba} = -\overline{\mathbf{C}}_{ba}^T\mathbf{u}_b + \mathbf{u}_a \tag{5}$$

where $\mathbf{u}_a, \mathbf{u}_b$ are the point cloud centroids given by

$$\mathbf{u}_a = \frac{1}{N}\sum_{i=1}^{N}\mathbf{p}_a^i \qquad \text{and} \qquad \mathbf{u}_b = \frac{1}{N}\sum_{i=1}^{N}\mathbf{p}_b^i. \tag{6}$$

C. Non-linear Motion Solution

The direct motion solution forms an initial guess $\{\overline{\mathbf{C}}_{ba}, \overline{\mathbf{r}}_a^{ba}\}$ for a non-linear optimization procedure. The objective function we wish to minimize is

$$\mathcal{L} = \frac{1}{2}\sum_{i=1}^{N}\left(\mathbf{p}_b^i - \mathbf{C}_{ba}(\mathbf{p}_a^i - \mathbf{r}_a^{ba})\right)^T$$
$$\boldsymbol{\Gamma}^i\left(\mathbf{p}_b^i - \mathbf{C}_{ba}(\mathbf{p}_a^i - \mathbf{r}_a^{ba})\right), \tag{7}$$

with

$$\mathbf{\Gamma}^i = \left(\mathbf{G}_b^i \mathbf{R}_b^i \mathbf{G}_b^{i^T} + \mathbf{C}_{ba} \mathbf{G}_a^i \mathbf{R}_a^{i^T} \mathbf{G}_a^{i^T} \mathbf{C}_{ba}^T \right)^{-1}, \qquad (8)$$

where

$$\mathbf{G}_a^i = \left.\frac{\partial \mathbf{g}}{\partial \mathbf{y}}\right|_{\mathbf{f}(\mathbf{p}_a^i)} \qquad \text{and} \qquad \mathbf{G}_b^i = \left.\frac{\partial \mathbf{g}}{\partial \mathbf{y}}\right|_{\mathbf{f}(\mathbf{p}_b^i)} \qquad (9)$$

are the Jacobians of the inverse camera model $\mathbf{g}$, and $\mathbf{R}_a^i$, $\mathbf{R}_b^i$ are the covariance matrices of the jth point in image space.

We proceed by perturbing the initial guess by a vector $\boldsymbol{\xi} = \begin{bmatrix} \boldsymbol{\epsilon}^T & \boldsymbol{\phi}^T \end{bmatrix}^T$, where $\boldsymbol{\epsilon}$ is a translational perturbation and $\boldsymbol{\phi}$ is a rotational perturbation:

$$\mathbf{r}_a^{ba} = \bar{\mathbf{r}}_a^{ba} + \boldsymbol{\epsilon}, \qquad (10)$$

$$\mathbf{C}_{ba} = e^{-\boldsymbol{\phi}^\times} \overline{\mathbf{C}}_{ba} \approx (\mathbf{1} - \boldsymbol{\phi}^\times)\overline{\mathbf{C}}_{ba}, \qquad (11)$$

where $(\cdot)^\times$ is the skew-symmetric cross-product operator.

Inserting (10) and (11) into (7), we arrive at a cost function that is quadratic in the perturbations:

$$\mathcal{L} \approx \frac{1}{2} \sum_{j=1}^N \left(\bar{\mathbf{e}}^i + \overline{\mathbf{E}}^i \boldsymbol{\xi} \right)^T \overline{\mathbf{\Gamma}}^i \left(\bar{\mathbf{e}}^i + \overline{\mathbf{E}}^i \boldsymbol{\xi} \right), \qquad (12)$$

where

$$\bar{\mathbf{e}}^i = \mathbf{p}_b^i - \overline{\mathbf{C}}_{ba}(\mathbf{p}_a^i - \bar{\mathbf{r}}_a^{ba}),$$

$$\overline{\mathbf{E}}^i = \left[\overline{\mathbf{C}}_{ba} \quad -\left(\overline{\mathbf{C}}_{ba}(\mathbf{p}_a^i - \bar{\mathbf{r}}_a^{ba}) \right)^\times \right],$$

and $\overline{\mathbf{\Gamma}}^i$ indicates that $\overline{\mathbf{C}}_{ba}$ has replaced $\mathbf{C}_{ba}$ in (8). Setting the derivative of (12) with respect to $\boldsymbol{\xi}$ to zero yields a system of linear equations in the optimal update step $\boldsymbol{\xi}^*$:

$$\sum_{j=1}^N \left(\overline{\mathbf{E}}^{i^T} \overline{\mathbf{\Gamma}}^i \overline{\mathbf{E}}^i \right) \boldsymbol{\xi}^* = -\sum_{i=1}^N \overline{\mathbf{E}}^{i^T} \overline{\mathbf{\Gamma}}^i \bar{\mathbf{e}}^i. \qquad (13)$$

Once $\boldsymbol{\xi}^*$ is determined, the state estimate can be repeatedly updated using:

$$\overline{\mathbf{C}}_{ba} \leftarrow \exp((-\boldsymbol{\phi}^*)^\times)\overline{\mathbf{C}}_{ba}, \qquad (14)$$

$$\bar{\mathbf{r}}_a^{ba} \leftarrow \bar{\mathbf{r}}_a^{ba} + \boldsymbol{\epsilon}^*. \qquad (15)$$

In practice, (13) is also modified with Levenberg-Marquardt dampening to improve convergence.

IV. PROBE: PREDICTIVE ROBUST ESTIMATION

The aim of PROBE is to learn a general model for the quality of visual features, with the goal of reducing the impact of deleterious visual effects such as moving objects, motion blur, and shadows on navigation estimates. Feature quality is characterized by a scalar weight β_i for each visual feature in an environment. To compute β_i we define a prediction space (similar to [9]) that consists of a set of visual-inertial predictors computed from the local image region around the feature and the inertial state of the vehicle. We then scale the image covariance of each feature ($\mathbf{R}_a^i$, $\mathbf{R}_b^i$ in (8)) by β_i during the non-linear optimization.

Similarly to M-estimation, PROBE achieves robustness by varying the influence of certain measurements. However, in contrast to robust cost functions that weight measurements based purely on estimation error, PROBE weights measurements based on their assessed quality.

To learn the model, we require training data that consists of a traversal through a typical environment with some measure of ground truth for the path, but not for the visual features themselves. Like many machine learning techniques, we assume that the training data is representative of the test environments in which the learned model will be used.

We learn the quality of visual features *indirectly* through their effect on navigation estimates. We define high quality features as those that result in estimates that are close to ground truth. Our framework is flexible enough that we do not require ground truth at every image; we can learn the model based on displacement only (e.g., loop closure error).

A. Training

Training proceeds by traversing the training path, selecting a subset of visual features at each step, and using them to compute a position estimate. By comparing the estimated position to the ground truth position, we compute the translational Root Mean Squared Error (RMSE) and store it at each feature's position in the prediction space. The full algorithm is summarized in Figure 2. Note that $\alpha_{l,s}$ can be computed at each step, at intermittent steps, or for an entire path, depending on the availability of ground truth data.

```
1:  procedure TRAINPROBEMODEL
2:      for l ← 1, totalLearningIterations do
3:          for s ← 1, totalCameraImages do
4:              f_1, ..., f_J ← visualFeatureSubset(l)
5:              π_l^1, ..., π_l^J ← predictors(f_1, ..., f_J)
6:              C̄_ba, r̄_a^ba ← poseChange(f_1, ..., f_J)
7:              α_{l,s} ← computeRMSE(r̄_a^ba, r_a^ba_GT)
8:              Θ_{l,s} ← {π_{l,s}^1, ..., π_{l,s}^J, α_{l,s}}
9:          end for
10:     end for
11:     return Θ = {Θ_{l,s}}
12: end procedure
```

Fig. 2. The PROBE training procedure.

B. Evaluation

To use the PROBE model in a test environment, we compute the location of each observed visual feature in our prediction space, and then compute its relative weight β_i as a function of its K nearest neighbours in the training set. For efficiency, the K nearest neighbours are found using a k-d tree. The final scaling factor β_i is a function of the mean of the α values corresponding to the K nearest neighbours, normalized by $\bar{\alpha}$, the mean α value of the entire training set.

The value of K can be determined through cross-validation, and in practice depends on the size of the training set and the environment. The computation of β_i is designed to map small differences in learned α values to scalar weights that span several orders of magnitude. The optimal value of

```
 1: procedure USEPROBEMODEL(Θ)
 2:     for i ← 1, totalFeatures do
 3:         πᵢ ← predictors(fᵢ)
 4:         α₁, ..., α_K ← findKNN(πᵢ, K, Θ)
 5:         βᵢ ← (1/αK Σ_{k=1}^{K} α_k)^γ
 6:     end for
 7:     return β = {βᵢ}
 8: end procedure
```

Fig. 3.   The PROBE evaluation procedure.

γ can be found by searching through a set range of candidate γ values and choosing the value that minimizes the average RMSE (ARMSE) on the training set.

C. Prediction Space

A crucial component of our technique is the choice of prediction space. In practice, feature tracking quality is often degraded by a variety of effects such as motion blur, moving objects, and textureless or self-similar image regions. Features suffering from such effects are often poorly localized in the image, yet may contain useful information that we would like to retain in our estimator. Conversely, we would like to place more trust in features that do not suffer from such effects. The challenge is in determining predictors that account for such effects without requiring excessive computation. In our implementation, we use the following predictors, but stress that the choice of predictors can be tailored to suit particular applications and environments:

- Angular velocity and linear acceleration magnitudes
- Local image entropy
- Blur (quantified by the blur metric of [21])
- Optical flow variance score
- Image frequency composition

We discuss each of these predictors in turn.

1) Angular velocity and linear acceleration: While most of the predictors in our system are computed directly from image data, the magnitudes of the angular velocities and linear accelerations reported by the IMU are in themselves good predictors of image degradation (e.g. image blur) and hence poor feature tracking. We do not explicitly correct for bias in linear accelerations because we expect real motion-induced acceleration to trump bias at the timescales of our test trials. As a result, there is virtually no computational cost involved in incorporating these quantities as predictors.

2) Local image entropy: Entropy is a statistical measure of randomness that can be used to characterize the texture in an image or patch. Since the quality of feature detection is strongly influenced by the strength of the texture in the vicinity of the feature point, we expect the entropy of a patch centered on the feature to be a good predictor of its quality. We evaluate the entropy S in an image patch by sorting pixel intensities into N bins and computing

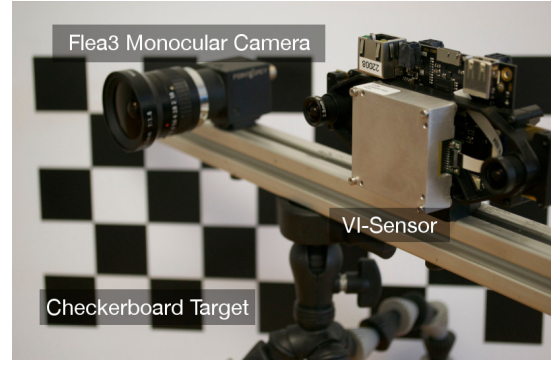$$S = -\sum_{i=1}^{N} c_i \log_2(c_i), \quad (16)$$

Fig. 4.   The Skybotix VI-Sensor, Point Grey Flea3, and checkerboard target used in our motion blur experiments.
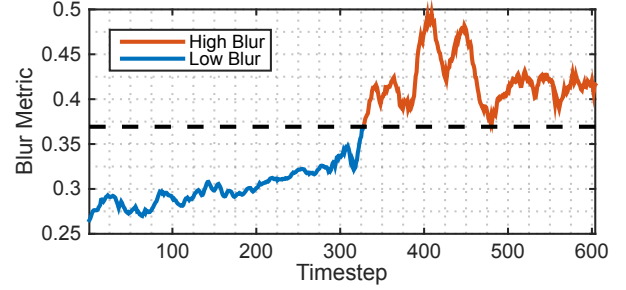

Fig. 5.   Blur metric [21] computed for the left camera of the VI-Sensor for the checkerboard dataset. We separate the dataset into regions of high and low blur corresponding to fast and slow motion, respectively.

where c_i is the number of pixels counted in the ith bin.

*3) Blur:* Blur can arise from a number of sources including motion, dirty lenses, and sensor defects. All of these have deleterious effects on feature tracking quality. To assess the effect of blur in detail, we performed a separate experiment. We recorded images of 32 interior corners of a standard checkerboard calibration target using a low frame-rate (20 FPS) Skybotix VI-Sensor stereo camera and a high frame-rate (125 FPS) Point Grey Flea3 monocular camera rigidly connected by a bar (Figure 4). Prior to the experiment, we determined the intrinsic and extrinsic calibration parameters of our rig using the Kalibr package [22]. The apparatus underwent both slow and fast translational and rotational motion, which induced different levels of motion blur as quantified by the blur metric proposed by [21] (Figure 5).

We detected checkerboard corners in each camera at synchronized timesteps, computed their 3D coordinates in the VI-Sensor frame, then reprojected these 3D coordinates into the Flea3 frame. We computed the reprojection error as the Euclidean distance between the reprojected image coordinates and the true image coordinates in the Flea3 frame. Since the Flea3 operated at a much higher frame rate than the VI-Sensor, it was less susceptible to motion blur and so we treated it as ground truth. We also computed a tracking error by comparing the image coordinates of checkerboard corners in the left camera of the VI-Sensor computed from both KLT tracking [23] and re-detection.

Figure 6 shows histograms and fitted normal distributions
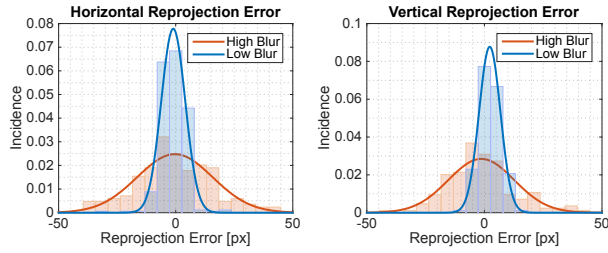
(a) Reprojection error of checkerboard corners triangulated from the VI-Sensor and reprojected into the Flea3.
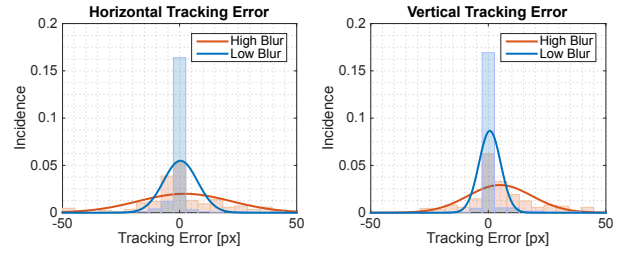
(b) Tracking error of KLT-tracked checkerboard corners in the left VI-Sensor camera compared to directly re-detected corners.

Fig. 6. Effect of blur on reprojection and tracking error for the slow-then-fast checkerboard dataset. We distinguish between high and low blur by thresholding the blur metric [21] as shown in Figure 5. The variance in both errors increases with blur.
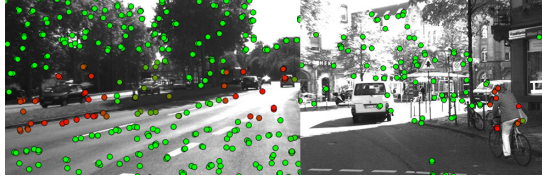

Fig. 7. The optical flow variance predictor can help in detecting moving objects. Red circles correspond to higher values of the optical flow variance score (i.e., features more likely to belong to a moving object).
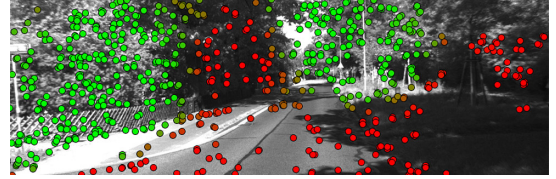

Fig. 8. A high-frequency predictor can distinguish between regions of high and low texture such as foliage and shadows. Green indicates higher values.
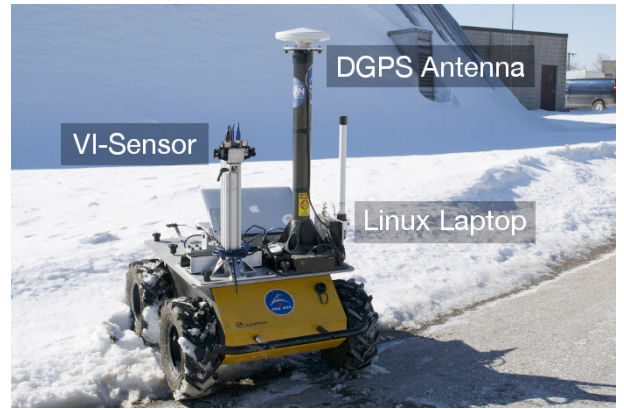

Fig. 10. Our four-wheeled skid-steered Clearpath Husky rover equipped with Skybotix VI-Sensor and Ashtech DGPS antenna used to collect the outdoor UTIAS dataset.

for both reprojection error and tracking error. From these distributions we can see that the errors remain approximately zero-mean, but that their variance increases with blur. This result is compelling evidence that the effect of blur on feature tracking quality can be accounted for by scaling the feature covariance matrix by a function of the blur metric.

4) Optical flow variance score: To detect moving objects, we compute a score for each feature based on the ratio of the variance in optical flow vectors in a small region around the feature to the variance in flow vectors of a larger region. Intuitively, if the flow variance in the small region differs significantly from that in the larger region, we might expect the feature in question to belong to a moving object, and we would therefore like to trust the feature less. Since we consider only the variance in optical flow vectors, we expect this predictor to be reasonably invariant to scene geometry.

We compute this optical flow variance score according to

$$\log\left(\frac{\bar{\sigma}_s^2}{\bar{\sigma}_l^2}\right), \tag{17}$$

where $\bar{\sigma}_s^2, \bar{\sigma}_l^2$ are the means of the variance of the vertical and horizontal optical flow vector components in the small and large regions respectively. Figure 7 shows sample results of this scoring procedure for two images in the KITTI dataset [24]. Our optical flow variance score generally picks out moving objects such as vehicles and cyclists in diverse scenes.

5) Image frequency composition: Reliable feature tracking is often difficult in textureless or self-similar environments due to low feature counts and false matches. We detect textureless and self-similar image regions by computing the Fast Fourier Transform (FFT) of each image and analyzing its frequency composition. For each feature, we compute a

coefficient for the low- and high-frequency regimes of the FFT. Figure 8 shows the result of the high-frequency version of this predictor on a sample image from the KITTI dataset [24]. Our high-frequency predictor effectively distinguishes between textureless regions (e.g., shadows and roads) and texture-rich regions (e.g., foliage).

V. EXPERIMENTAL RESULTS

A. Datasets

We trained and evaluated PROBE in two sets of experiments. The first set of experiments made use of 4.5 km of data from the City, Residential, and Road categories of the KITTI dataset [24]. In the second set of experiments, we collected indoor and outdoor datasets at the University of Toronto Institute for Aerospace Studies (UTIAS) using a Skybotix VI-Sensor mounted on an Adept MobileRobots

Trial	Type	Path Length	Nominal RANSAC (99% outlier rejection)		Aggressive RANSAC (99.99% outlier rejection)		PROBE	
			ARMSE	Final Error	ARMSE	Final Error	ARMSE	Final Error
`26_drive_0051`	City [1]	251.1 m	4.84 m	12.6 m	3.30 m	8.62 m	3.48 m	8.07 m
`26_drive_0104`	City [1]	245.1 m	0.977 m	4.43 m	0.850 m	3.46 m	1.19 m	3.61 m
`29_drive_0071`	City [1]	234.0 m	5.44 m	30.3 m	5.44 m	30.4 m	3.03 m	12.8 m
`26_drive_0117`	City [1]	322.5 m	2.29 m	9.07 m	2.29 m	9.07 m	2.76 m	9.08 m
`30_drive_0027`	Residential [1,†]	667.8 m	4.22 m	12.2 m	4.30 m	10.6 m	3.64 m	4.57 m
`26_drive_0022`	Residential [2]	515.3 m	2.21 m	3.99 m	2.66 m	6.09 m	3.06 m	4.99 m
`26_drive_0023`	Residential [2]	410.8 m	1.64 m	8.20 m	1.77 m	8.27 m	1.71 m	8.13 m
`26_drive_0027`	Road [3]	339.9 m	1.63 m	8.75 m	1.63 m	8.65 m	1.40 m	7.57 m
`26_drive_0028`	Road [3]	777.5 m	4.31 m	16.9 m	3.72 m	13.1 m	3.92 m	13.2 m
`30_drive_0016`	Road [3]	405.0 m	4.56 m	19.5 m	3.33 m	14.6 m	2.76 m	13.9 m
UTIAS Outdoor	Snowy parking lot	302.0 m	7.24 m	10.1 m	7.02 m	10.6 m	6.85 m	6.09 m
UTIAS Indoor	Lab interior	32.83 m	—	0.854 m	—	0.738 m	—	0.617 m

[1] Trained using sequence `09_26_drive_0005`. [2] Trained using sequence `09_26_drive_0046`. [3] Trained using sequence `09_26_drive_0015`.
[†] This residential trial was evaluated with a model trained on a sequence from the city category because of several moving vehicles that were better represented in that training dataset.

Fig. 9. Three types of environments in the KITTI dataset, as well as 2 types of environments at the University of Toronto. We use one trial from each category to train and then evaluate separate trials in the same category.

Pioneer 3-AT rover and a Clearpath Husky rover, respectively. In both cases, the camera recorded stereo images at 10 Hz while the IMU operated at 200 Hz. The outdoor dataset consisted of a 264 m training run followed by a 302 m evaluation run, with ground truth provided by RTK-corrected GPS. The indoor dataset consisted of a 32 m training run and a 33 m evaluation run through a room with varying light and shadows. For the indoor dataset, no ground truth was available, so we trained PROBE using only the knowledge that the training path should form a closed loop.

We compare PROBE to what we call the nominal VINS, as well as a VINS with an aggressive RANSAC routine. In the nominal pipeline, we use RANSAC with enough iterations to be 99% confident that we select only inliers when as many as 50% of the features are outliers. In the aggressive case, we increase the confidence level to 99.99%. When PROBE is applied, we apply a pre-processing step that makes use of the rotational estimate from the IMU to reject any egregious feature matches by thresholding the cosine distance between pairs of matched feature vectors. We assume small translations between frames and typically set the threshold to reject feature vectors that are separated by more than five degrees.

For feature extraction, matching, and sparse optical flow calculations, we use the open source vision library `LIBVISO2` [18]. `LIBVISO2` efficiently detects thousands of feature correspondences by finding stable features using blob and corner masks and matching them by comparing Sobel filter responses. For all prediction space calculations,

we use features in the left image of the stereo pair.

B. Training

Sample results of the training procedure described in Section IV are illustrated in Figure 11 for data from the Residential category. As ground truth is available for each image, we compute RMSE at every frame, and only iterate over the path 10 times. Note the large variance of training run error in the sharp turn in Figure 11, caused by a car that drives through the field-of-view. PROBE is able to distinguish features on the car and adequately reduce their influence with the final learned model.

C. Evaluation

To evaluate PROBE, we run the nominal VINS on a given test trial, tune the RANSAC threshold to achieve reasonable translation error ($< 5\%$ final drift), then repeat the trial with the aggressive RANSAC procedure. Finally, we disable RANSAC completely and apply our trained PROBE model (with pre-processing) to each observed feature. Table I compares the performance of each trained PROBE model to that of the nominal and aggressive-RANSAC VINS. In the best case, PROBE achieves a final translational error norm of less than half that of both reference VINS. Figures 12, 13, and 14 reveal problematic sections of the KITTI dataset where PROBE is able to significantly improve upon the performance of both reference VINS. Moving vehicles and pedestrians are the most obvious sources of error that PROBE is able to identify. Other more subtle effects include
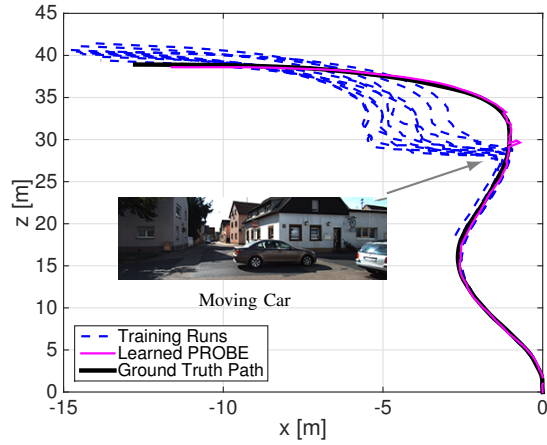
Fig. 11. Training iterations for the Residential category, sequence `09_26_drive_0046`. The left turn is particularly problematic due to a moving car that comes into view. Although none of the training runs completely remove all features from the car, the path differences are enough for the learned PROBE model to adequately reduce the influence of the car on the final motion estimate.
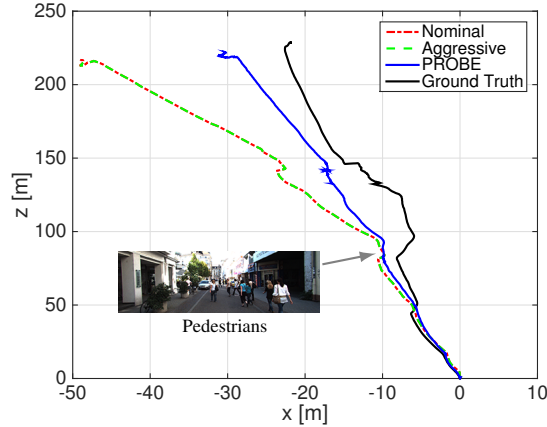

Fig. 12. A 234 m KITTI test run in the City category, sequence `09_29_drive_0071` containing numerous pedestrians that cross the camera's field of view with dramatic lighting changes. PROBE is able to produce more accurate navigation estimates than even an aggressive RANSAC routine.
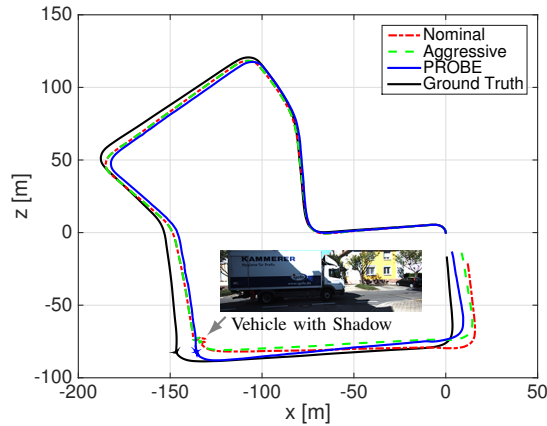

Fig. 13. A 667 m test run in the Residential category, sequence `09_30_drive_0027`. PROBE is better able to deal with a static portion when a large shadow and moving vehicle cross the field of view of the camera.
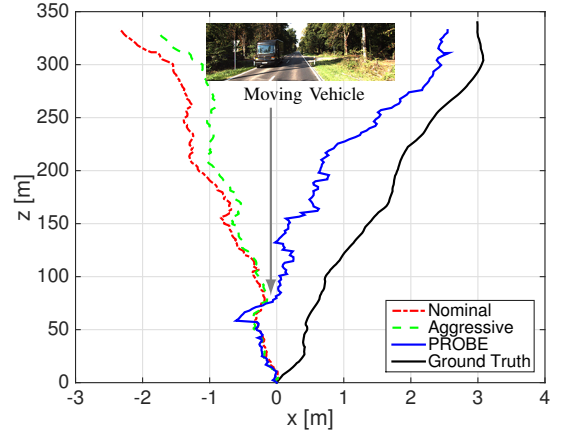

Fig. 14. A 440 m test run in the Road category, sequence `09_26_drive_0027`. PROBE is able to better predict a large moving vehicle, and extract higher quality features from dense foliage. Note the difference in scale between the two axes.
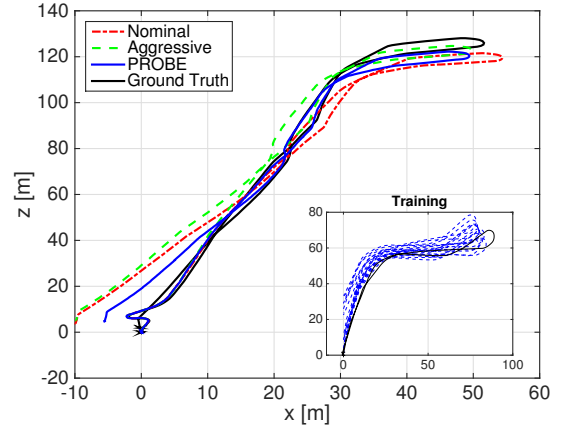

Fig. 15. Test and training trials in our outdoor UTIAS environment. The rover drives through a snowy landscape and people walk though the field of view.

motion blur (notable at the edge of images), slowly swaying vegetation, and shadows.

VI. DISCUSSION

In most of the datasets we evaluated, PROBE performed as well as a standard RANSAC routine in reducing the influence of outliers. In particular, PROBE has proven to be more robust than aggressive RANSAC to visual effects such as shadows, large moving objects, and dense foliage. Often, PROBE can produce more accurate navigation estimates by intelligently weighting measurements to reflect their quality while still exploiting the information contained in low-quality measurements. In this sense, PROBE can be thought of as a soft outlier classification scheme, while RANSAC rejects outliers based on binary classification.

Like with other machine learning approaches, a good training dataset is essential to having an accurate and generalizable model. PROBE is flexible enough that it can be taught with varying frequency of ground truth data. For instance, in the outdoor dataset collected at the UTIAS, GPS measurements were available at only 1Hz, and PROBE
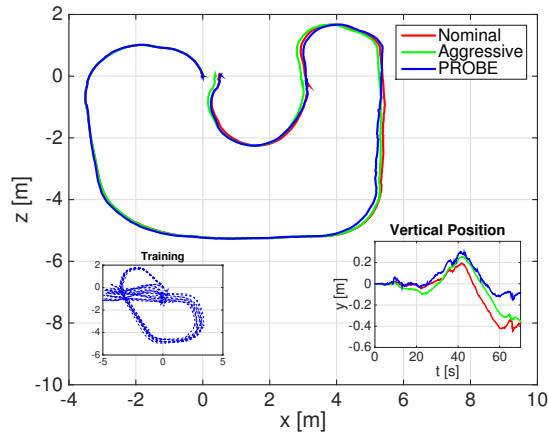
Fig. 16. A 32.8 m test run in our indoor UTIAS dataset. The subplot in the bottom-right corner of the figure shows that PROBE reduces drift in the vertical (y) direction more than the nominal VINS and aggressive RANSAC.

was trained by evaluating the ARMSE over the entire path. For the indoor dataset, no ground truth was available and the model was learned by computing the loop closure error between the start and end points of the training path.

VII. CONCLUSIONS

In this work, we presented PROBE, a novel method to predict the quality of visual features within complex, dynamic environments. By using training data to learn a mapping from a pre-defined prediction space of visual-inertial predictors to a scalar weight, we can adjust the influence of individual visual features on the final navigation estimate. PROBE can be used in place of traditional outlier rejection techniques such as RANSAC, or combined with them to more intelligently weight inlier measurements.

We explored a variety of potential predictors, and validated our technique using a visual-inertial navigation system on over 4 km of data from the KITTI dataset and 700 m of indoor and outdoor data collected at the University of Toronto Institute for Aerospace Studies. Our results show that PROBE outperforms RANSAC-based binary outlier rejection in many environments, even with sparsely available ground truth during the training step.

In future work, we plan to examine a broader set of predictors, and extend the training procedure to incorporate online learning using intermittent ground truth measurements or loop closures detected by a place recognition thread. Further, we are interested in analyzing the amount of training data required for a given improvement in navigation, and investigating PROBE's effect on estimator consistency.

ACKNOWLEDGEMENT

REFERENCES

[1] M. Fischler and R. Bolles, "Random Sample Consensus - a Paradigm for Model-Fitting with Applications to Image-Analysis and Automated Cartography," *CACM*, vol. 24, no. 6, pp. 381–395, 1981.

[2] D. Scaramuzza and F. Fraundorfer, "Visual Odometry [Tutorial],", *IEEE Robotics & Automation Mag.*, vol. 18, no. 4, pp. 80–92, 2011.

[3] Y. Latif, C. Cadena, and J. Neira, "Robust graph SLAM back-ends: A comparative analysis," in *Proc. IEEE/RSJ Int. Conf. Intelligent Robots and Systems (IROS)*, 2014, pp. 2683–2690.

[4] S. Wangsiripitak and D. W. Murray, "Avoiding moving outliers in visual SLAM by tracking moving objects," in *Proc. IEEE Int. Conf. Robotics and Automation (ICRA)*, 2009, pp. 375–380.

[5] C. McManus, W. Churchill, W. Maddern, A. D. Stewart, and P. Newman, "Shady dealings: Robust, long-term visual localisation using illumination invariance," in *Proc. IEEE Int. Conf. Robotics and Automation (ICRA)*, 2014, pp. 901–906.

[6] A. Pretto, E. Menegatti, M. Bennewitz, W. Burgard, and E. Pagello, "A visual odometry framework robust to motion blur," in *Robotics and Automation, 2009. ICRA '09. IEEE International Conference on*. IEEE, 2009, pp. 2250–2257.

[7] R. Sim and G. Dudek, "Learning and evaluating visual features for pose estimation," in *Proc. IEEE Int. Conf. Computer Vision*, 1999, pp. 1217–1222.

[8] L. Carlone, A. Censi, and F. Dellaert, "Selecting good measurements via $\ell 1$ relaxation: A convex approach for robust estimation over graphs," in *Proc. IEEE/RSJ Int. Conf. Intelligent Robots and Systems (IROS)*, 2014, pp. 2667–2674.

[9] W. Vega-Brown, A. Bachrach, A. Bry, J. Kelly, and N. Roy, "CELLO: A fast algorithm for Covariance Estimation," in *Proc. IEEE Int. Conf. Robotics and Automation (ICRA)*, 2013, pp. 3160–3167.

[10] W. Vega-Brown and N. Roy, "CELLO-EM: Adaptive sensor models without ground truth," in *Proc. IEEE/RSJ Int. Conf. Intelligent Robots and Systems (IROS)*, 2013, pp. 1907–1914.

[11] J.-A. Ting, E. Theodorou, and S. Schaal, "Learning an Outlier-Robust Kalman Filter," in *Machine Learning: ECML 2007*. Berlin, Heidelberg: Springer Berlin Heidelberg, 2007, pp. 748–756.

[12] A. I. Mourikis and D. G. Kottas, "An Iterative Kalman Smoother for Robust 3D Localization on Mobile and Wearable devices," University of Minnesota, Tech. Rep., Nov. 2014.

[13] S. Leutenegger, S. Lynen, and M. Bosse, "Keyframe-based visual–inertial odometry using nonlinear optimization," *Int. J. Robotics Research*, vol. 34, pp. 314–334, 2014.

[14] A. I. Mourikis and S. I. Roumeliotis, "A Multi-State Constraint Kalman Filter for Vision-aided Inertial Navigation," in *Proc. IEEE Int. Conf. Robotics and Automation*, 2007, pp. 3565–3572.

[15] V. Indelman, S. Williams, 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, Aug. 2013.

[16] Y. Cheng, M. W. Maimone, and L. Matthies, "Visual odometry on the Mars exploration rovers - a tool to ensure accurate driving and science imaging," *Robotics & Automation Magazine, IEEE*, vol. 13, no. 2, pp. 54–62, 2006.

[17] G. S. Sukhatme, J. Kelly, G. S. Sukhatme, S. Saripalli, and S. Saripalli, "Combined visual and inertial navigation for an unmanned aerial vehicle," *Field and Service Robotics*, 2008.

[18] A. Geiger, J. Ziegler, and C. Stiller, "StereoScan: Dense 3d reconstruction in real-time," *Proc. IEEE Intelligent Vehicles Symp. (IV)*, pp. 963–968, 2011.

[19] L. Kneip, M. Chli, and R. Siegwart, "Robust Real-Time Visual Odometry with a Single Camera and an IMU," in *Proc. British Machine Vision Conf. (BMVC)*. British Machine Vision Association, 2011, pp. 16.1–16.11.

[20] J. Kelly and G. S. Sukhatme, "Visual-Inertial Sensor Fusion: Localization, Mapping and Sensor-to-Sensor Self-calibration," *Int. J. Robotics Research*, vol. 30, no. 1, pp. 56–79, 2011.

[21] F. Crete, T. Dolmiere, P. Ladret, and M. Nicolas, "The blur effect: perception and estimation with a new no-reference perceptual blur metric," *Electronic Imaging*, vol. 6492, pp. 64 920I–64 920I–11, Feb. 2007.

[22] P. Furgale, J. Rehder, and R. Siegwart, "Unified temporal and spatial calibration for multi-sensor systems," in *Proc. IEEE/RSJ Int. Conf. Intelligent Robots and Systems (IROS)*. IEEE, 2013, pp. 1280–1286.

[23] B. D. Lucas and T. Kanade, "An iterative image registration technique with an application to stereo vision." *Proc. Int. Joint Conf. Artificial Intelligence (IJCAI)*, 1981.

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