2017 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)
September 24–28, 2017, Vancouver, BC, Canada

Edge-based Visual-Inertial Odometry

Hongsheng Yu, Anastasios I. Mourikis
Dept. of Electrical and Computer Engineering, University of California, Riverside
E-mail: {hyu|mourikis}@ece.ucr.edu

***Abstract*— In this paper we propose a method for monocular visual-inertial odometry that utilizes image edges as measurements. In contrast to previous feature-based approaches, the proposed method does not employ any assumption on the geometry of the scene (e.g., it does not assume straight lines). It can thus use measurements from all image areas with significant gradient, similarly to direct semi-dense methods. However, in contrast to direct semi-dense approaches, the proposed method's measurement model is invariant to linear changes in the image intensity. The novel edge parameterization and measurement model we propose explicitly account for the fact that edge points can only provide useful information in the direction of the image gradient. We present both Monte-Carlo simulations, as well as results from real-world experimental testing, which demonstrate that the proposed edge-based approach to visual-inertial odometry is consistent, and outperforms the point-based one.**

I. Introduction

This paper addresses the problem of motion estimation by fusing measurements from an inertial measurement unit (IMU) and a camera. Specifically, we focus on the task of using visual and inertial measurements to track the pose of a moving platform in an unknown environment, a task often termed visual-inertial odometry (VIO) [1]. The vast majority of algorithms developed for VIO rely on the use of *point features* (e.g., Shi-Tomasi corners [2], SIFT features [3], or FAST corners [4], among others), detected and tracked in consecutive images. Due to the fact that the estimation relies on these features, rather than the raw intensity values of the images, such approaches are broadly categorized as *indirect* ones. These algorithms greatly reduce the amount of processed data (going from hundreds of thousands of pixels to a few hundreds of point features), and therefore are suitable even for heavily resource-constrained systems.

However, point-feature extraction also leads to loss of information, as not all parts of the image are used. Therefore, there has been increased interest recently in *direct* methods, which directly use the image intensities as measurements for estimation [5]–[9]. These methods make it possible to use significantly more pixel locations (theoretically, even the entire image), and thus potentially exploit more information for motion estimation. However, direct approaches also face a number of shortcomings. First, they are very sensitive to errors in the projection geometry (caused, for instance, by inaccurate estimates of the camera intrinsics, or from the errors in the pose estimates themselves) [8]. Thus direct methods may fail in cases where low-cost hardware and/or low-texture environments lead to significant errors in the predicted location of the projection of scene points. Second, the process of image formation in a camera is complex, making it difficult to derive a precise photometric model that considers all possible factors. For example, the image intensity at a given pixel location is affected by the surface properties of scene objects, the viewing angle between the camera and the observed objects, global lighting conditions, camera exposure time, camera gains, and lens characteristics such as vignetting. These factors may be hard to model (e.g., the surface reflectance properties), and may change unpredictably (e.g., lighting conditions), thus leading to unmodeled errors.

To address this problem, direct methods may employ a photometric model in which the intensities of the projection of the same 3D point in two different images, $I_1(r, c)$ and $I_2(r', c')$, are related by a linear expression of the form $I_2(r', c') = \alpha I_1(r, c) + \beta$ [8]. The "gain" and "offset" parameters α, β are good representations of some of the factors mentioned above (e.g., global illumination changes and camera gain/exposure changes), but in practice are also necessary in order to *approximately* model additional effects, such as non-Lambertian surface properties. Estimating these parameters reliably presents challenges (e.g., using the same values for all pixels in an image may not work well), and also increases the computational cost, as extra states need to be estimated.

Motivated by the above, we here propose a different approach to motion estimation that lies, in some respects, between direct methods and feature-based ones. Specifically, in our approach we employ image *contours*, detected by applying edge-detection in an image. In contrast to point-feature methods, we are able to utilize information from *all* parts of an image where gradients with large magnitude exist. This is similar to semi-dense direct methods that use all image areas with significant gradient [6]. However, in contrast to such direct methods our measurement model relies on the geometry of contours in the image, rather than the raw image intensities (in that sense, our method is an indirect one). Importantly, the edge contours we employ (defined as the locations where the image gradient magnitude is maximum, along the gradient direction) are *invariant* to linear image intensity changes of the form $\alpha I(r, c) + \beta$. This provides robustness against scene illumination changes and camera gain/exposure changes, and potentially to additional unmodeled effects, if these can be locally well approximated by a linear model.

The use of measurements derived from edge contours provides a number of additional advantages. For instance, edges may be abundant in environments where point features are sparse (e.g., indoors). This observation has also been

978-1-5386-2682-5/17/$31.00 ©2017 IEEE

exploited in previous work, where straight-line features have been used for pose estimation [10], [11]. In contrast to such approaches, however, we do *not* employ any parametric model for the shape of the edges we are using. Instead, each curve in the world is represented as a collection of 3D points, each parameterized by a minimal, two-parameter representation. This allows us to utilize any edge detected in an image, not only those that conform to a certain model, making the approach applicable in any environment. It also allows for a simple measurement model to be derived, based on the reprojection errors of the 3D curve points.

In what follows, we present the details of our approach. The proposed contour-based formulation is employed for VIO within the multi-state constraint Kalman filter (MSCKF) [1], [12], which relies on maintaining in the state vector a sliding window of camera poses, while features are directly marginalized and never included in the state. This formulation results in computational complexity that is only linear in the number of edge points. In addition to our novel parameterization of the 3D curves, and the associated measurement model, we here describe a method for computing the accuracy of the edge pixels' location, based on a local bicubic approximation of the image. Our simulation and experimental results show that the resulting formulation outperforms the corresponding point-based approach.

II. Related Work

The vast majority of VIO methods to date have relied on the use of point features, which are detected and tracked in the images (see, e.g., [1], [13], [14] and references therein). In these methods, the measurements are formulated in geometric space (i.e., the coordinates of a feature's projection in the image). By contrast, direct methods for pose estimation formulate a measurement model in the image intensity space, computed for either all image pixels [5], or for image regions with significant gradient magnitude [6]–[8], [15], or only around a set of extracted feature points [9], [16], [17]. As discussed in the preceding section, the method we propose in this paper seeks to leverage the advantages of both approaches. By computing measurement residuals in geometric space, we obtain robustness to changes in the lighting and imaging conditions, similarly to point-feature based methods. Moreover, by utilizing image edges, we are able to exploit information from more parts of the image, similarly to semi-dense direct methods.

The use of edge information for *model-based* pose estimation has a long history in computer vision (see, e.g., [18]–[20]). In these approaches and their descendants, the 3D structure of the scene (or of an object whose pose is being tracked) is assumed to be known. The 3D camera pose is obtained by minimizing the "reprojection errors" between the observed edges and those predicted based on the known 3D model. In our work, however, we are focusing on localization in an unknown environment. Therefore, knowledge of a 3D model of the scene cannot be assumed, and this type of methods are not applicable.

For edge-based pose estimation in unknown environments, a number of approaches exist that employ a stereo pair of cameras [21], [22] or a depth camera [23]. These approaches are conceptually similar to the model-based ones. Specifically, in both cases, given either a stereo image pair or a depth image, one can create a 3D model for the locations of the edge points in the scene. Then, pose estimation can proceed by finding the transformation that optimally aligns a new image pair or depth scan to that 3D model. These methods rely crucially on having a depth estimate for each of the edge pixels detected in an image (obtained via stereo triangulation or via the depth camera). However, since in our work we employ a monocular camera, this requirement is not met, and the aforementioned approaches cannot be used.

For monocular localization in unknown environments, the VIO algorithms of [10], [11] rely on the use of *straight-line* edges. Since only straight lines can be used, the applicability of these methods is limited to environments where such features are abundant. Moreover, there is a risk of introducing unmodeled errors, when lines that are slightly curved are treated as straight. In a similar vein, the methods of [24]–[26] rely on the use of straight-line segments. By contrast, in our work we make no assumptions on the shape of the curves that result in the observed image edges. We also note that higher-order curve parameterizations have been employed in the literature, such as B-splines and NURBS models [27], [28]. While able to handle a larger class of curves than straight lines, these representations are also susceptible to modeling errors, and make the correspondence problem harder, as typically the entire curve must be visible in all images.

The work that is closest to ours is that of Tarrio and Pedre [29]. Similarly to our approach, [29] employs a monocular camera for localization, and a general, point-based parameterization for the edges. However, pose constraints are only computed between pairs of images. By contrast, in our work, *all* the constraints created when an edge is observed in multiple images are utilized, via the use of the MSCKF formulation. Additionally, the algorithm of [29] uses an overparameterization of edge points, representing each of them as a general 3D point. In our work, a minimal, 2-parameter representation of the edge points is used. Finally, we model the accuracy of the detection of each edge pixel, instead of treating it as a constant. In what follows, we describe the details of our approach.

III. Estimator formulation

We begin by describing the formulation of the estimator that we employ for VIO. This is based on the MSCKF algorithm [1], [12], which is an extended-Kalman-filter (EKF)-based method. The state vector of the estimator contains a sliding window of poses, corresponding to the time instants the latest m images were recorded. The IMU readings are used to propagate the IMU state, while the image observations are used to derive probabilistic constraints on the camera poses. We here briefly present the MSCKF estimator, to introduce the notation for the remainder of the paper. The interested reader is referred to [1], [12] for additional details.

Algorithm 1 Edge-based MSCKF Algorithm

Propagation: Propagate the state vector and the state covariance matrix using the IMU readings.

Update: When camera measurements become available:

- Augment the sliding window with a new state, and begin image processing.
- For each completed edge-point track
  - Obtain the maximum-likelihood estimate, $\hat{\mathbf{f}}_i$, for the edge point's parameters shown in (8), using all the observations of this edge point in a least-squares minimization.
  - Compute the residuals associated with all the edge point measurements, as shown in (9), and their Jacobians (equation (10)), and apply the method of [12] to remove the term involving $\tilde{\mathbf{f}}_i$.
  - Perform a Mahalanobis-distance gating test.
- Perform an EKF update using all the edge points that pass the Mahalanobis test

State Management:

- Remove all sliding-window states that have no active edge point tracks associated with them, or poses older than m.

Our goal is to estimate the pose of a moving platform with respect to a gravity-aligned global coordinate frame $\{G\}$, using IMU readings and images recorded by a global-shutter camera. To formulate the estimator equations, we affix a coordinate frame $\{I\}$ to the IMU, and a coordinate frame $\{C\}$ to the camera, respectively. We here assume that the intrinsic parameters of the camera, as well as the relative transformation between the camera and the IMU are known via prior calibration. This is done for ease of presentation, and also because these assumptions hold true for our experimental setup. However, these are not requirements for the method we present. If any of the sensor calibration parameters are not accurately known, they can be estimated online, as described in [30].

The state vector of the MSCKF at time-step k is given by:

$$\mathbf{x}_k = \begin{bmatrix} \mathbf{x}_{E_k}^T & \mathbf{x}_{I_{k-1}}^T & \cdots & \mathbf{x}_{I_{k-m}}^T \end{bmatrix}^T \tag{1}$$

where $\mathbf{x}_{E_k}$ is the "evolving state" of the IMU:

$$\mathbf{x}_{E_k} = \begin{bmatrix} {}^{I_k}_{G}\bar{\mathbf{q}}^T & {}^G\mathbf{p}_k^T & {}^G\mathbf{v}_k^T & \mathbf{b}_{\mathbf{g}_k}^T & \mathbf{b}_{\mathbf{a}_k}^T \end{bmatrix}^T \tag{2}$$

The IMU state comprises the unit quaternion ${}^{I_k}_{G}\bar{\mathbf{q}}$, representing the rotation from frame $\{G\}$ to the IMU frame $\{I\}$ at time-step k, the IMU position and velocity in the global frame, ${}^G\mathbf{p}_k$ and ${}^G\mathbf{v}_k$, respectively, as well as the gyroscope and accelerometer biases, $\mathbf{b}_{\mathbf{g}_k}$ and $\mathbf{b}_{\mathbf{a}_k}$, respectively, which are modeled as Gaussian random-walk processes.

The error-state for the evolving IMU state is defined as [1]:

$$\tilde{\mathbf{x}}_{E_k} = \begin{bmatrix} {}^G\tilde{\boldsymbol{\theta}}_k^T & {}^G\tilde{\mathbf{p}}_k^T & {}^G\tilde{\mathbf{v}}_k^T & \tilde{\mathbf{b}}_{\mathbf{g}_k}^T & \tilde{\mathbf{b}}_{\mathbf{a}_k}^T \end{bmatrix}^T \tag{3}$$

where the standard additive error definition is used for the position, velocity and biases (i.e., for a random variable $\mathbf{y}$, its estimate is denoted $\hat{\mathbf{y}}$, and the estimation error is defined as $\tilde{\mathbf{y}} = \mathbf{y} - \hat{\mathbf{y}}$), while for the orientation errors we use a minimal 3-dimensional representation, defined in [1].

Each of the states $\mathbf{x}_{I_j}$, $j = k-m, \ldots, k-1$ comprises the IMU position and orientation at the time instant the corresponding image was recorded:

$$\mathbf{x}_{I_j} = \begin{bmatrix} {}^{I_j}_{G}\bar{\mathbf{q}}^T & {}^G\mathbf{p}_j^T \end{bmatrix}^T \quad j = k-m, \ldots, k-1$$

and the errors in the estimates of these states are defined accordingly:

$$\tilde{\mathbf{x}}_{I_j} = \begin{bmatrix} {}^G\tilde{\boldsymbol{\theta}}_j^T & {}^G\tilde{\mathbf{p}}_j^T \end{bmatrix}^T \quad j = k-m, \ldots k-1$$

When an IMU measurement is received, it is used to propagate the evolving state and the filter covariance matrix, as described in [1]. On the other hand, when a new image is received, the sliding window of states is augmented with a new pose, and the image is processed to extract and match edge points. Each edge point is tracked for as long as possible (or until the maximum limit of m poses is reached), and all its measurements are processed together once the tracking is complete, to provide constraints involving the poses of the sliding window. For this purpose, the multi-state-constraint method of [12] is employed, which makes it possible to use the feature measurements without including the feature in the EKF state vector.

Prior to using an edge point's measurements for an EKF update, a Mahalanobis-distance gating test is performed, to remove outliers. All the edge points that pass the gating test are then employed for an EKF update. At the end of the update, edge points that are no longer visible and old sliding-window states with no active edge-point tracks associated with them are removed. Note that, to ensure the correct observability properties of the linearized system model, and thus improve the estimation accuracy and consistency, the estimator employs fixed linearization points for each state [1].

IV. EKF Update with Edges

In this section we present the key contributions of this paper, namely the parameterization of 3D curves that we employ for estimation, and the derivation of the accompanying measurement model for edge observations.

A. Edge Point Parameterization

In this work, we do not assume any parametric form for the edges in the images, or for the 3D curves whose projection forms these edges. Instead, a 3D curve $\mathcal{C}$ is represented by a set of points, $\mathbf{p}_{c_i}$, $i = 1, \ldots, N$ (the number of points on each curve will generally be different). To define these points, we start with the first observation of the curve. Let $\mathcal{E}$ denote the edge that results from the first observation of $\mathcal{C}$ in an image. We define a number of edge points $\mathbf{e}_i$, $i = 1, \ldots, N$, spaced at regular intervals along $\mathcal{E}$ (see Fig. 1). Each of the 3D curve points $\mathbf{p}_{c_i}$ is defined as the intersection of the 3D curve with the plane π_i that contains the edge point $\mathbf{e}_i$, the

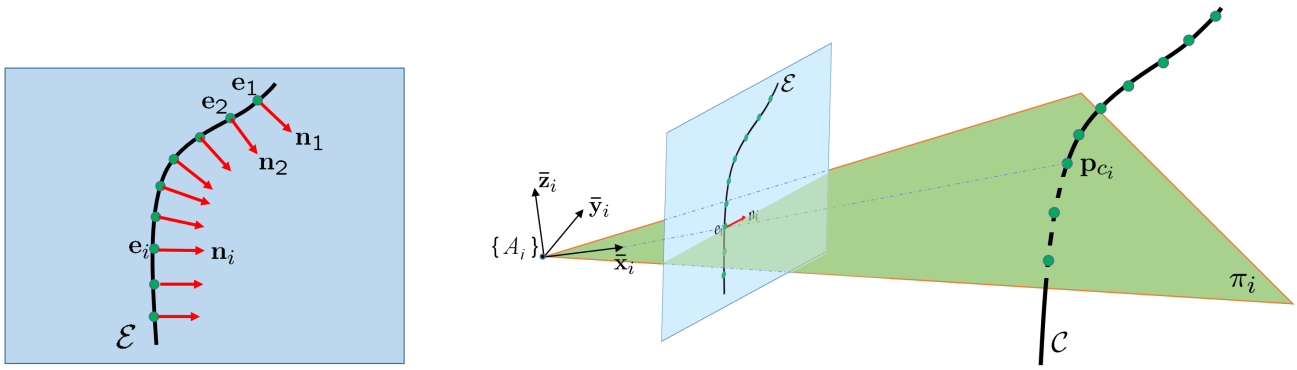

Fig. 1. Edge Parameterization. Left: Given the first observation of a new curve, we define a number of edge points $\mathbf{e}_i$, spaced at regular intervals along the edge. Right: Each of the edge points, $\mathbf{e}_i$, together with the corresponding edge normal, $\mathbf{n}_i$, and the camera optical center, define a plane π_i. We parameterize the intersection point of π_i with the 3D curve, $\mathbf{p}_{c_i}$, with a 2-parameter vector $\mathbf{f}_i$, representing the position of $\mathbf{p}_{c_i}$ in π_i.

camera optical center, and the edge normal vector $\mathbf{n}_i$ (see Fig. 1).

The motivation for defining the set of points $\mathbf{p}_{c_i}$ as described above, i.e., via the intersection of $\mathcal{C}$ with a set of planes π_i, is that this leads to a *two-dimensional parameterization* for each of the points $\mathbf{p}_{c_i}$. These two parameters describe the position of the intersection of the curve $\mathcal{C}$ with the *known* two-dimensional plane π_i. This is a desirable property: note that changes in the position of $\mathbf{p}_{c_i}$ along the curve are *unobservable*, since, in general, we cannot distinguish different points along an edge in an image. Therefore, if the point $\mathbf{p}_{c_i}$ was represented by three parameters (the standard representation for points in 3D), this would be an over-parameterization, in the context of the problem at hand.

We now describe the definition of the two parameters that we use to represent each of the points $\mathbf{p}_{c_i}$, $i = 1, \ldots, N$. Given the normalized image coordinates of an edge point, $\mathbf{e}_i$, and the edge-normal vector at this point (i.e., the image gradient vector), $\mathbf{n}_i$, we define a *known, constant* coordinate frame $\{A_i\}$, whose origin coincides with the estimate of the camera coordinate frame, and whose coordinate axes are defined in the global frame as:

$$^G\bar{\mathbf{x}}_i = \frac{1}{\eta_x} {}^G_C\hat{\mathbf{R}} \begin{bmatrix} \mathbf{e}_i \\ 1 \end{bmatrix} \tag{4}$$

$$^G\bar{\mathbf{z}}_i = \frac{1}{\eta_z} {}^G_C\hat{\mathbf{R}} \left(\begin{bmatrix} \mathbf{e}_i \\ 1 \end{bmatrix} \times \begin{bmatrix} \mathbf{n}_i \\ 0 \end{bmatrix} \right) \tag{5}$$

$$^G\bar{\mathbf{y}}_i = {}^G\bar{\mathbf{z}}_i \times {}^G\bar{\mathbf{x}}_i \tag{6}$$

where ${}^G_C\hat{\mathbf{R}}$ is the estimate for the rotation matrix between the camera frame and the global frame, and η_x and η_z are normalization constants to ensure unit length of the corresponding vectors. Intuitively, the above definitions mean that the x-axis of the frame $\{A_i\}$ is along the vector from the camera optical center to $\mathbf{e}_i$, and the z-axis is normal to the plane π_i. With this definition of $\{A_i\}$, the point $\mathbf{p}_{c_i}$ has a z-coordinate of zero, and we therefore parameterize it as:

$$^{A_i}\mathbf{p}_{c_i} = \frac{1}{\rho_i} \begin{bmatrix} \cos\theta_i \\ \sin\theta_i \\ 0 \end{bmatrix} \tag{7}$$

In the above parameterization, ρ_i has the role of inverse depth, which is known to lead to improved linearity properties in vision-based estimation [31]. Moreover, we note that with our frame definition, the initial estimate of the parameter θ_i is by definition zero.

To summarize, given the image edge corresponding to a new 3D curve, we define a set of points along the image edge, $\mathbf{e}_i$, $i = 1, \ldots, N$, and subsequently we define a set of *known, constant* coordinate frames $\{A_i\}$, whose origin coincides with the origin of the estimated camera frame, and their principal axes are defined as shown in (4)-(6). We subsequently parameterize the intersections of the 3D curve with the $x-y$ planes of these frames by the two-parameter vector:

$$\mathbf{f}_i = \begin{bmatrix} \rho_i & \theta_i \end{bmatrix}, \quad i = 1, \ldots, N \tag{8}$$

which defines the points according to (7).

B. Measurement model

To derive a measurement model based on the parameterization described in the preceding section, we rely on the fact that the projections of all points $\mathbf{p}_{c_i}$, $i = 1, \ldots, N$ of a 3D curve, should lie on the edges corresponding to the curve in all images. Therefore, if we denote by $\mathbf{h}(\hat{\mathbf{x}}_{I_j}, \hat{\mathbf{f}}_i)$ the predicted projection of $\mathbf{p}_{c_i}$ on the image at time step j, the following quantity can be viewed as a measurement residual:

$$r_{d_{ij}} = \text{dist}(\mathcal{E}_j, \mathbf{h}(\hat{\mathbf{x}}_{I_j}, \hat{\mathbf{f}}_i))$$

where $\mathcal{E}_j$ denotes the observed edge resulting from the projection of the curve on the image at time step j, while $\hat{\mathbf{x}}_{I_j}$ and $\hat{\mathbf{f}}_i$ are the estimates for the IMU pose and the curve-points' parameters, used to predict the projection of $\mathbf{p}_{c_i}$ on the image (the exact form of the projection function, $\mathbf{h}$, is shown in Section IV-C). While valid, the above expression is not suitable for use in an EKF estimator, as it is a nonnegative quantity. To address this issue, we instead define the following residual, which is a signed quantity:

$$r_{ij} = \mathbf{n}_{ij}^T \left(\mathbf{z}_{ij} - \mathbf{h}(\hat{\mathbf{x}}_{I_j}, \hat{\mathbf{f}}_i) \right) \tag{9}$$

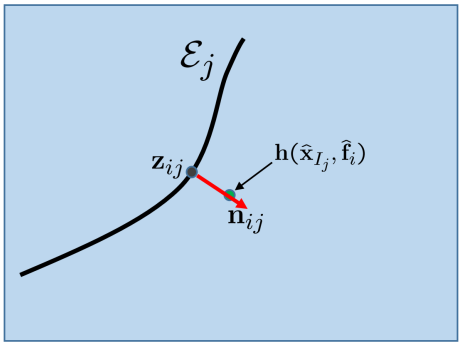

Fig. 2. Illustration of the measurement model.

where $\mathbf{z}_{ij}$ is the point on $\mathcal{E}_j$ that is closest to $\mathbf{h}(\hat{\mathbf{x}}_{I_j}, \hat{\mathbf{f}}_i)$, and $\mathbf{n}_{ij}^T$ is a constant unit vector. Note that, for any unit vector, the above quantity would be a valid residual, in the sense that if the estimates were perfect and the edge measurement noiseless, it would equal zero. However, to obtain a meaningful residual, we choose the vector $\mathbf{n}_{ij}$ as the gradient of the image at $\mathbf{z}_{ij}$. This is illustrated in Fig. 2. It is important to point out that the residual in (9) is a *scalar* quantity, which is desirable, since edge observations can only provide useful information in the direction normal to the gradient.

To obtain a linearized approximation of the residual, we begin by using first-order Taylor expansion of $\mathbf{h}(\mathbf{x}_{I_j}, \mathbf{f}_i)$, which yields:

$$\mathbf{h}(\mathbf{x}_{I_j}, \mathbf{f}_i) \simeq \mathbf{h}(\hat{\mathbf{x}}_{I_j}, \hat{\mathbf{f}}_i) + \mathbf{H}_{\mathbf{x}_{ij}} \tilde{\mathbf{x}}_{I_j} + \mathbf{H}_{\mathbf{f}_{ij}} \tilde{\mathbf{f}}_i$$

where $\tilde{\mathbf{x}}_{I_j}$ and $\tilde{\mathbf{f}}_i$ are the errors in the estimates for the IMU pose at time step j and for the point's parameters, respectively, and $\mathbf{H}_{\mathbf{x}_{ij}}$ and $\mathbf{H}_{\mathbf{f}_{ij}}$ are the corresponding Jacobians. Moreover, let $\mathbf{z}_{ij} = \check{\mathbf{z}}_{ij} + \boldsymbol{\eta}_{ij}$, where $\check{\mathbf{z}}_{ij}$ is the true location of the point on the edge, and $\boldsymbol{\eta}_{ij}$ the noise in its detection. Substituting in (9), and noting that $\check{\mathbf{z}}_{ij} = \mathbf{h}(\mathbf{x}_{I_j}, \mathbf{f}_i)$, we obtain the following linearized form of the residual:

$$r_{ij} \simeq \mathbf{H}_{\mathbf{x}_{ij}} \tilde{\mathbf{x}}_{I_j} + \mathbf{H}_{\mathbf{f}_{ij}} \tilde{\mathbf{f}}_i - \mathbf{n}_{ij}^T \boldsymbol{\eta}_{ij} \tag{10}$$

It is important to note that the term $\mathbf{n}_{ij}^T \boldsymbol{\eta}_{ij}$ represents the noise in the detection of the point that is closest to $\mathbf{h}(\hat{\mathbf{x}}_{I_j}, \hat{\mathbf{f}}_i)$, *along the direction of the gradient*. In other words, this noise term corresponds to the accuracy with which the edge can be detected in the image. In Section IV-D, we describe how the covariance of this noise term can be computed.

Once the residuals defined in (9), as well as their corresponding Jacobians in (10), have been computed for all the images in which the curve point can be tracked, the method described in [12] is employed in order to marginalize out the parameter error $\mathbf{f}_i$, and a Mahalabobis gating test is performed. If the test is successful, the measurements of this curve point are used for an EKF update, along with all other curve points available at this time, as detailed in [12].

C. Geometric Model

We now present the function $\mathbf{h}$, which describes the projection of a curve point $\mathbf{p}_{c_i}$ with parameter vector $\mathbf{f}_i$, on the image at time step j. We begin by computing the position of $\mathbf{p}_{c_i}$ with respect to the camera at time step j:

$${}^{C_j}\mathbf{p}_i = {}^{C}_{I}\mathbf{R}\, {}^{I_j}_{G}\mathbf{R} \left({}^{G}\mathbf{p}_{A_i} + {}^{G}_{A_i}\mathbf{R}\, {}^{A_i}\mathbf{p}_{c_i} - {}^{G}\mathbf{p}_{I_j} \right) + {}^{C}\mathbf{p}_I$$

where ${}^{C}_{I}\mathbf{R}$ and ${}^{C}\mathbf{p}_I$ are the known rotation and translation between the camera and IMU frames, ${}^{I_j}_{G}\mathbf{R}$ and ${}^{G}\mathbf{p}_{I_j}$ describe the IMU pose in the global frame, ${}^{G}_{A_i}\mathbf{R}$ and ${}^{G}\mathbf{p}_{A_i}$ are the known, constant parameters of the frame $\{A_i\}$ associated with the point $\mathbf{p}_{c_i}$, and ${}^{A_i}\mathbf{p}_{c_i}$ is given by (7).

Given the vector ${}^{C_j}\mathbf{p}_i = [{}^{C_j}x_i \;\; {}^{C_j}y_i \;\; {}^{C_j}z_i]^T$, the image coordinates of its projection on image j depend only on the imaging geometry of the camera. In our experiments, we employ a fisheye-lens camera, and we use the model of [32] to describe the projection geometry:

$$\mathbf{h}\left(\mathbf{x}_{I_j}, \mathbf{f}_i\right) = \frac{1}{r_u \omega} \arctan\left(2 r_u \tan\left(\frac{\omega}{2}\right)\right) \begin{bmatrix} a_u u \\ a_v v \end{bmatrix} + \mathbf{p}_p \tag{11}$$

where $\mathbf{p}_p$ is the pixel location of the principal point, (a_u, a_v) are the camera focal length measured in horizontal and vertical pixel units, ω is the distortion parameter, and

$$r_u = \sqrt{u^2 + v^2} \tag{12}$$

$$\begin{bmatrix} u \\ v \end{bmatrix} = \frac{1}{{}^{C_j}z_i} \begin{bmatrix} {}^{C_j}x_i \\ {}^{C_j}y_i \end{bmatrix} \tag{13}$$

All the camera intrinsic parameters, namely $\mathbf{p}_p, a_u, a_v$, and ω, are known through prior calibration.

D. Edge Detection And Uncertainty Model

We now describe our approach for detecting edges in the image with subpixel precision, and estimating the accuracy with which they are localized. Our approach is based on the method of [33], where a bicubic polynomial is employed to model the local image intensity. In particular, edges are first detected to pixel-level accuracy by applying a Canny-like edge detection method on the images [34]. Then, for those pixels where subpixel accuracy is required (i.e., those used to define the curve points as described in Section IV and their correspondences) we fit the following bicubic polynomial to model the intensity of the image in a local 7×7 pixel neighborhood:

$$\begin{aligned} f(c, r) = {} & k_1 + k_2 c + k_3 r + k_4 c^2 + k_5 cr + k_6 r^2 \\ & + k_7 c^3 + k_8 c^2 r + k_9 c r^2 + k_{10} r^3 \end{aligned} \tag{14}$$

where c and r are the column and row coordinates with respect to the center edge pixel. We then obtain the subpixel location of the edge, as well as its accuracy, based on the parameters of this polynomial. Specifically, note that the direction of the image gradient at the edge pixel under consideration is given by $[k_2 \;\; k_3]^T$. The location of the edge is the point, along this direction, where the second-order directional derivative along the gradient is zero. By solving the resulting set of equations, we obtain the following subpixel offset for the location for the edge, along the direction of the gradient:

$$o(\mathbf{k}) = \frac{(-k_2^2 k_4 - k_2 k_3 k_5 - k_3^2 k_6)\sqrt{k_2^2 + k_3^2}}{3(k_2^3 k_7 + k_2^2 k_3^2 + k_2 k_3^3 k_9 + k_3^3 k_{10})} \tag{15}$$

where we have made explicit the fact that this offset is a function of the parameter vector $\mathbf{k} = [k_1 \;\; k_2 \;\; \ldots \;\; k_{10}]^T$. As explained in the preceding section, the accuracy with which this offset can be determined represents the "measurement noise" in our measurement model (10). To compute the variance of this noise, we employ linearization to compute the error in $o(\mathbf{k})$ as:

$$\tilde{o}(\mathbf{k}) \simeq \frac{\partial o}{\partial \mathbf{k}} \frac{\partial \mathbf{k}}{\partial \mathbf{I}} \tilde{\mathbf{I}}$$

where $\mathbf{I}$ represents the values of the image intensity in the local 7×7 image neighborhood, and $\tilde{\mathbf{I}}$ the corresponding image-intensity noise. Therefore, the variance of the measurement noise is given by:

$$\sigma_o^2 = \frac{\partial o}{\partial \mathbf{k}} \mathbf{C}_\mathbf{k} \frac{\partial o}{\partial \mathbf{k}}^T \tag{16}$$

where $\mathbf{C}_\mathbf{k}$ is the covariance matrix of the errors in the estimate of $\mathbf{k}$ that we obtain from fitting the model (14). This matrix is a function of the image intensity noise variance, but, importantly, it does not depend on the value of $\mathbf{k}$ itself, and can thus be precomputed. By contrast, σ_o^2 will depend on the local image appearance, which affects the Jacobian $\frac{\partial o}{\partial \mathbf{k}}$. This allows us to model the fact that sharper edges can be localized more accurately than smoother ones.

We note that the process for computing the noise variance described above models the effects of image noise, but does not account for additional sources of error. For example, the true image intensity in the local region will, in general, not follow a bicubic-polynomial model. To account for these additional effects, in our implementation we use a threshold on the minimum allowable value of σ_o. Specifically, the value of σ_o that we use is the maximum of the value computed by (16), or 0.5 pixels.

V. Experimental Results

To test the performance of the proposed method, we conducted a set of Monte-Carlo simulations to evaluate its consistency, as well as a set of real-world experiments to evaluate its accuracy in real-world settings.

A. Monte-Carlo simulations

For the simulations, the trajectory of a hand-held device in a room-sized environment, recorded in a prior experiment, is used to generate a realistic ground-truth motion. Based on this ground truth trajectory, inertial measurements are subsequently generated, with random noise realizations used in each Monte-Carlo trial. To generate simulated camera images, we texture-map images recorded during real-world experiments onto the four simulated walls of the room, as well as the floor and ceiling. The simulated camera images are created with a different realization of Gaussian noise added to the image intensities in each trial. Two sample images generated in the simulation are shown in Fig. 3. The IMU measurements are available at 200 Hz, and the camera images at 10 Hz. The trajectory is approximately 181 m long, traversed in about 2.9 mins.

Fig. 3. Sample images generated by the simulator.

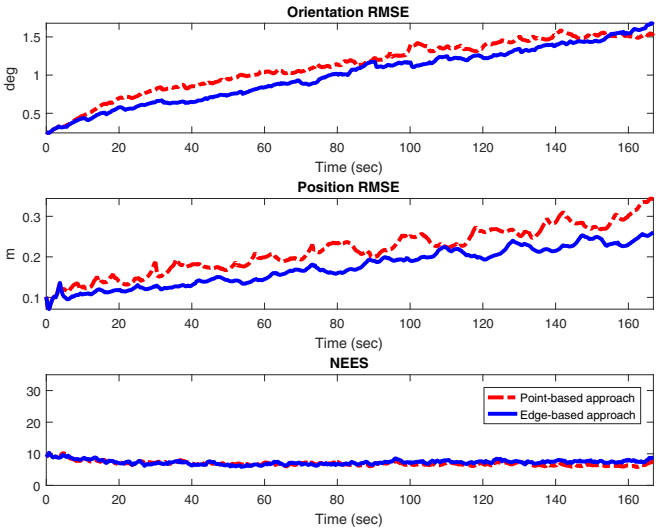

Fig. 4. Average RMS error and NEES over 50 Monte-Carlo trials.

For the edge-based VIO approach, we define the 3D curve points via edge points spaced every 3 pixels along the first observation of a curve, as described in Section IV. These points are detected at subpixel accuracy, using the method in Section IV-D. Matching from image j to image $j+1$ is performed using search along epipolar lines. For a given edge point in image j, if an edge point is found along the epipolar line in image $j+1$, and the directions of the gradient vectors are close (within a threshold of 30^o), then normalized cross-correlation is used as the matching criterion. Once matching for all points from image j is complete, new points are introduced in the edge segments of image $j+1$ where no matches have been detected.

To evaluate the accuracy and consistency of the edge-based VIO approach in comparison to the point-based one, we perform 50 Monte-Carlo simulations. For both formulations, we compute the RMS error for the IMU orientation and position, as well as the normalized estimation error squared (NEES) of the IMU pose at each time step. The RMS errors provide us with a measure of accuracy, while the NEES is a measure of consistency [1]. Specifically, if the estimator is consistent (i.e., if the estimation errors are zero mean, and have ensemble covariance equal to the covariance reported by the filter [35]) the average value of the NEES over all Monte Carlo trials should equal six (i.e., should equal the dimension of the pose-error vector). The average RMS and NEES over all 50 Monte Carlo trials are shown in

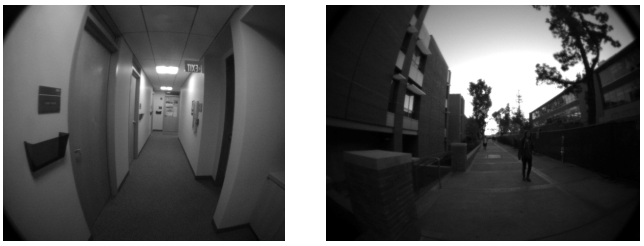

Fig. 5. Sample images recorded during the experiments.

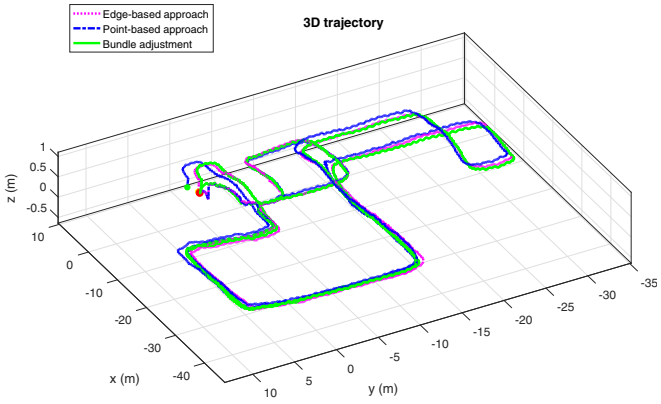

Fig. 6. Trajectory estimates computed using the edge-based and point-based MSCKF, as well as the bundle-adjustment ground truth, in one of the datasets.

Fig. 4, plotted over time. We observe that the proposed edge-based approach outperforms the point-based one in terms of estimation precision. In addition, we note that the NEES of the edge-based approach is 7.36, close to the ideal value of 6 for consistency.

B. Real-World Experiments

We next present results from real-world experiments that were conducted to compare the performance of the proposed edge-based approach to the original, point-based formulation of the MSCKF in real-world settings. We collected six datasets in both indoor and outdoor environments, with trajectory lengths ranging from 314 m to 465 m (sample images from the experiments are shown in Fig. 5). For data collection a Tango developer platform was used, which records IMU data at a sample rate of 200 Hz and images at 30 Hz. All datasets were processed offline, so that comparisons between the different approaches could be performed.

We evaluate the performance of the proposed edge-based MSCKF formulation, against the point-based one, using the performance metrics of [36]. Specifically, we compute the error in the position and orientation estimates of each method as a function of path length. For this to be feasible, the ground truth of the trajectory is required. Since an external ground-truth system is not available in the mixed indoor-outdoor settings where the datasets were collected, we here employ global visual-inertial bundle adjustment to obtain a trajectory estimate that is used as ground truth. This bundle adjustment utilizes point features as the image measurements, and makes use of loop-closure constraints.

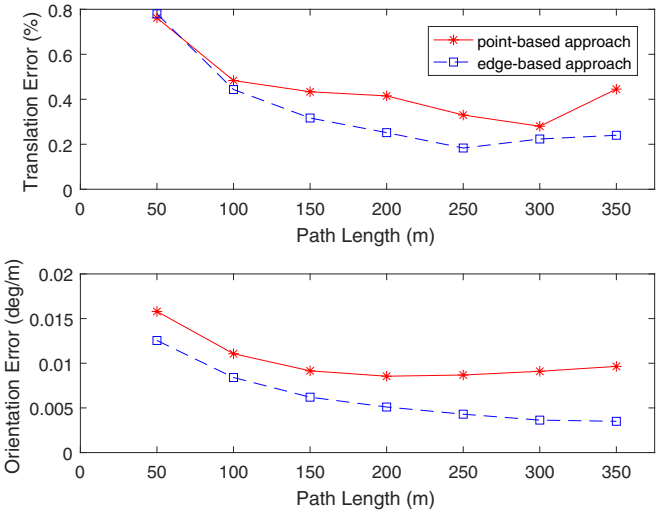

Fig. 7. Performance comparison of the edge-based MSCKF to the original point-based one. The apparent lower accuracy of the methods over small path lengths is attributed to errors in the estimate of the bundle adjustment that is used as ground truth.

While collecting data, care was taken so that a significant number of "loop closure events" occur along the trajectories. This ensures that the accuracy of the bundle adjustment is significantly higher than the accuracy of the point- and edge-based VIO, and using it as a ground truth is a reasonable approximation. The trajectory estimates produced by the three approaches in one of the datasets are shown in Fig. 6.

The evaluation of the accuracy of the point-based and edge-based approaches is shown in Fig. 7. To compute the error for a path length L, the errors in the relative-position and relative-orientation estimates of each method are computed and averaged over all trajectory segments of length L in all datasets. The errors are then divided by L, to obtain the position error as a percentage of the traveled distance, and the orientation-error growth in degrees per meter traveled:

$$\text{Error}_{\text{trans}}(L) = \frac{1}{L}\text{average}\left(\left|\left|{}^{I_{t_i}}\mathbf{p}_{I_{t_{i+L}}} - {}^{I_{t_i}}\hat{\mathbf{p}}_{I_{t_{i+L}}}\right|\right|_2\right)$$

$$\text{Error}_{\text{orient}}(L) = \frac{1}{L}\text{average}\left(\left|\left|{}^{I_{t_i}}\tilde{\boldsymbol{\theta}}_{I_{t_{i+L}}}\right|\right|_2\right)$$

where the average is taken over all time intervals $[t_i, t_{i+L}]$ corresponding to path segments of length L, and ${}^{I_{t_i}}\tilde{\boldsymbol{\theta}}_{I_{t_{i+L}}}$ represents the error of the estimate of the relative orientation ${}^{I_{t_i}}_{I_{t_{i+L}}}\mathbf{R}$. The results of Fig. 7 show that, due to the use of more information in the images, the proposed edge-based approach outperforms the point-based one, both in terms of position and orientation errors. The average reduction in error for the edge-based method, over all path lengths, is 41.6% for the orientation, and 26.1% for the position.

VI. Conclusion

In this paper, we have presented a novel, edge-based algorithm for monocular visual-inertial odometry. The proposed method relies on a point-based parameterization of 3D curves in which each point is parameterized by only two

parameters, defining the intersection of a known plane with the curve. This minimal parameterization, in conjunction with a measurement model that only employs measurement residuals along the direction of the edge gradient, results in an approach that can use all image edges, without any assumptions on scene geometry. Through extensive experimental validation, both in a simulated 3D environment, as well as in real-world experiments in both indoor and outdoor settings, we have shown that the proposed approach is consistent, and leads to improved estimation accuracy, compared to the "traditional" point-based one.

ACKNOWLEDGEMENTS

This work was supported by the National Science Foundation (grant no. IIS-1253314 and IIS-1316934), and Google, Inc. The authors would like to thank Mingyang Li, Zack Moratto, and Konstantine Tsotsos for the fruitful discussions and suggestions that helped improve this paper.

REFERENCES

[1] M. Li and A. I. Mourikis, "High-precision, consistent EKF-based visual-inertial odometry," *International Journal of Robotics Research*, vol. 32, no. 6, pp. 690–711, May 2013.

[2] J. Shi and C. Tomasi, "Good features to track," in *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, Seattle, WA, June 1994, pp. 593–600.

[3] D. G. Lowe, "Distinctive image features from scale-invariant keypoints," *International Journal of Computer Vision*, vol. 260, no. 2, pp. 91–110, Nov. 2004.

[4] E. Rosten, R. Porter, and T. Drummond, "Faster and better: a machine learning approach to corner detection," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 32, no. 1, pp. 105–119, 2010.

[5] R. A. Newcombe, S. J. Lovegrove, and A. J. Davison, "DTAM: Dense tracking and mapping in real-time," in *Proceedings of the International Conference on Computer Vision*, 6-13 Nov. 2011, pp. 2320–2327.

[6] J. Engel, J. Sturm, and D. Cremers, "Semi-dense visual odometry for a monocular camera," in *Proceedings of the IEEE International Conference on Computer Vision*, 1-8 Dec. 2013, pp. 1449–1456.

[7] J. Engel, T. Schöps, and D. Cremers, "LSD-SLAM: Large-scale direct monocular SLAM," in *Proceedings of the European Conference on Computer Vision (ECCV)*, 2014, pp. 834–849.

[8] J. Engel, V. Koltun, and D. Cremers, "Direct sparse odometry," *CoRR*, vol. abs/1607.02565, 2016. [Online]. Available: http://arxiv.org/abs/1607.02565

[9] C. Forster, M. Pizzoli, and D. Scaramuzza, "SVO: Fast semi-direct monocular visual odometry," in *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)*, May 31-June 7 2014, pp. 15–22.

[10] D. G. Kottas and S. I. Roumeliotis, "Efficient and consistent vision-aided inertial navigation using line observations," in *Proceedings of the IEEE International Conference on Robotics and Automation*, Karlsruhe, Germany, May 2013, pp. 1540 – 1547.

[11] H. Yu and A. I. Mourikis, "Vision-aided inertial navigation with line features and a rolling-shutter camera," in *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, Sept. 28 2015-Oct. 2 2015, pp. 892–899.

[12] A. I. Mourikis and S. I. Roumeliotis, "A multi-state constraint Kalman filter for vision-aided inertial navigation," in *Proceedings of the IEEE International Conference on Robotics and Automation*, Rome, Italy, Apr. 2007, pp. 3565–3572.

[13] E. Jones and S. Soatto, "Visual-inertial navigation, mapping and localization: A scalable real-time causal approach," *International Journal of Robotics Research*, vol. 30, no. 4, pp. 407–430, Apr. 2011.

[14] J. A. Hesch, D. G. Kottas, S. L. Bowman, and S. I. Roumeliotis, "Observability-constrained vision-aided inertial navigation," Dept. of Computer Science and Engineering, University of Minnesota, Tech. Rep., 2012.

[15] V. Usenko, J. Engel, J. Stckler, and D. Cremers, "Direct visual-inertial odometry with stereo cameras," in *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)*, Stockholm, Sweden, May 2016, pp. 1885–1892.

[16] H. Jin, P. Favaro, and S. Soatto, "A semi-direct approach to structure from motion," *The Visual Computer*, vol. 19, no. 6, pp. 377–394, 2003.

[17] P. Tanskanen, T. Naegeli, M. Pollefeys, and O. Hilliges, "Semi-direct EKF-based monocular visual-inertial odometry," in *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, Sept. 28 2015-Oct. 2 2015, pp. 6073–6078.

[18] D. Koller, K. Danilidis, and H.-H. Nagel, "Model-based object tracking in monocular image sequences of road traffic scenes," *International Journal of Computer Vision*, vol. 10, no. 3, pp. 257–281, June 1993.

[19] M. Armstrong and A. Zisserman, "Robust object tracking," in *Proceedings of the Asian Conference on Computer Vision*, 1995, pp. 58–61.

[20] D. Gennery, "Visual tracking of known three-dimensional objects," *International Journal of Computer Vision*, vol. 7, no. 3, pp. 243–270, 1992.

[21] A. Richardson and E. Olson, "PAS: visual odometry with perspective alignment search," in *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems*, Chicago, IL, Sept 2014, pp. 1053–1059.

[22] M. Tomono, "Robust 3D SLAM with a stereo camera based on an edge-point ICP algorithm," in *Proceedings of the IEEE International Conference on Robotics and Automation*, Kobe, Japan, May 2009, pp. 4306–4311.

[23] M. Kuse and S. Shen, "Robust camera motion estimation using direct edge alignment and sub-gradient method," in *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)*, 16-21 May 2016, pp. 573–579.

[24] E. Eade and T. Drummond, "Edge landmarks in monocular SLAM," in *Proceedings of the British Machine Vision Conference*, Edinburgh, Sep 2006, pp. 2.1–2.10.

[25] P. Smith, I. Reid, and A. Davison, "Real-time monocular SLAM with straight lines," in *Proceedings of the British Machine Vision Conference*, Edinburgh, Sep 2006, pp. 3.1–3.10.

[26] G. Klein and D. W. Murray, "Improving the agility of keyframe-based SLAM," in *Proceedings of the European Conference on Computer Vision (ECCV)*, Marseille, France, 2008.

[27] Y. J. Xiao and Y. F. Li, "Optimized stereo reconstruction of free-form space curves based on a nonuniform rational B-spline model," *Journal of the Optical Society of America A*, vol. 22, no. 9, pp. 1746–1762, Sep 2005.

[28] I. Nurutdinova and A. Fitzgibbon, "Towards pointless structure from motion: 3D reconstruction and camera parameters from general 3D curves," in *Proceedings of the IEEE International Conference on Computer Vision (ICCV)*, 7-13 Dec. 2015, pp. 2363–2371.

[29] J. J. Tarrio and S. Pedre, "Realtime edge-based visual odometry for a monocular camera," in *Proceedings of the IEEE International Conference on Computer Vision (ICCV)*, 7-13 Dec. 2015, pp. 702–710.

[30] M. Li, H. Yu, X. Zheng, and A. I. Mourikis, "High-fidelity sensor modeling and self-calibration in vision-aided inertial navigation," in *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)*, May 31-June 7 2014, pp. 409–416.

[31] J. Montiel, J. Civera, and A. Davison, "Unified inverse depth parametrization for monocular SLAM," in *Proceedings of Robotics: Science and Systems*, Philadelphia, PA, Aug. 2006, pp. 81–88.

[32] F. Devernay and O. Faugeras, "Straight lines have to be straight," *Machine Vision and Applications*, vol. 13, no. 1, pp. 14–24, 2001.

[33] R. M. Haralick, "Digital step edge from zero crossing of second directional derivatives," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 6, pp. 58–68, 1984.

[34] J. Canny, "A computational approach to edge detection," *IEEE Transacions on Pattern Analysis and Machine Intelligence*, vol. 8, no. 6, pp. 679–698, 1986.

[35] Y. Bar-Shalom, X. R. Li, and T. Kirubarajan, *Estimation with Applications to Tracking and Navigation*. John Wiley & Sons, 2001.

[36] A. Geiger, "Are we ready for autonomous driving? The KITTI vision benchmark suite," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, Washington, DC, USA, 2012, pp. 3354–3361.