

Sequential Pose Estimation Using Linearized Rotation Matrices

Timothy Michael Drews*, Paul G. Kry†, James Richard Forbes‡, and Clark Verbrugge§

McGill University

Montréal, Québec, Canada

*Email: timothy.drews@mail.mcgill.ca

†Email: kry@cs.mcgill.ca

‡Email: james.richard.forbes@mcgill.ca

§Email: clump@cs.mcgill.ca

Abstract—We present a new formulation for pose estimation using an extended Kalman filter that takes advantage of the Lie group structure of rotations. Using the exponential map along with linearized rotations for updates and errors permits a graceful filter formulation that avoids the awkward representation of Euler angles and the required norm constraint for quaternions. We demonstrate this approach with an implementation that uses sensors commonly found in consumer tablets and mobile phones: a camera and gyroscope, which we use to estimate attitude, position, and gyroscope bias. We use gyroscope measurements for prediction, and vision-based measurements for correction. We show results and discuss the performance of our pose estimation method using ground truth data obtained via a motion capture system.

Keywords—augmented reality; pose estimation; Kalman filter; linearized rotations; sensor fusion

I. INTRODUCTION

The recent flood of new tablets and mobile phones available to consumers is paving the way for mass adoption of augmented reality (AR) applications, and novel uses in mobile robotics. These consumer devices now come with significant processing power and a wide range of sensors that are useful for rigid motion tracking. Our primary interest is the problem of real-time pose estimation, that is, tracking rigid motion of the device relative to a fixed global reference frame.

Vision-based methods are the most straightforward solution for tracking objects in the real world. The use of fiducial markers or tags (small planar images with dots, patterns, or textures) have become popular as they can easily be localized in images and can store additional information. Markers can also be used as landmarks fixed in the environment to allow the camera's attitude and position to be estimated with reasonable accuracy. Alternatively, there has also been good progress in tracking rigid motion in natural and unstructured environments, with much of the attention on the simultaneous objectives of localization and mapping. Ultimately, vision techniques are possibly the best and most flexible mechanism for tracking attitude and position, and given enough computation time, very high quality results can be computed offline, for instance with *match moving* software in the production of visual effects.

While these vision-based methods work well, a camera can only be used to determine the device's pose if a sufficient

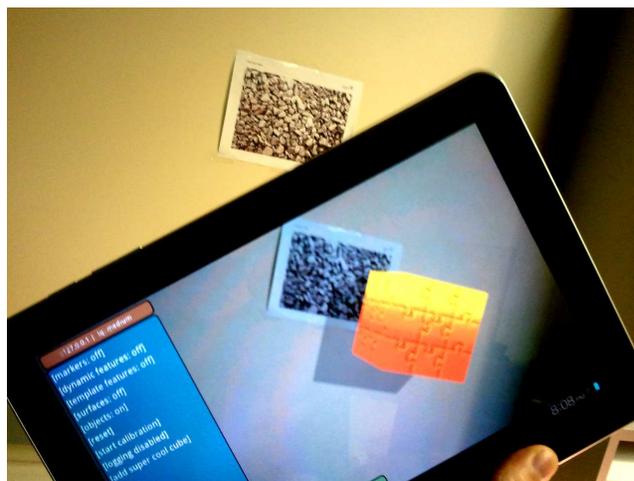


Fig. 1. Demonstration of a simple AR application running our tracking software on a tablet.

number of natural features are visible, or if a fiducial marker is in view. Features and markers can come in and out of view at any time, and motion blur or environmental changes can make it very tricky or impossible to get a good estimate of the device's pose in real-time. Measurements from inertial sensors can also determine the device's pose; however, these sensors have noise, and naive integration of gyroscope and accelerometer readings accumulate large amounts of drift — making the estimates increasingly unreliable over time. The ideal solution is to exploit the best capabilities of all available sensors in the computation of an estimate. This has been an active area for recent research, and is also the focus of our work.

Kalman filters have become a standard technique for exactly this kind of problem. The filter estimates a time-varying state while also modeling the uncertainty in the estimate due to process and measurement noise. A nice property is that the filter provides a minimum mean square error estimate in the case of linear systems with Gaussian white-noise. Since we are estimating attitude, we build upon the extended Kalman filter (EKF), which deals with nonlinear systems through linearization of the process and measurement models. For

the prediction step we use gyroscope measurements, while modeling changes in position as a random walk. The device’s camera provides a measurement of attitude and position for the correction step. We use Qualcomm’s fiducial (and image) tracking *Vuforia* SDK to obtain vision-based measurements.

The primary contribution of our work is the formulation and derivation of the EKF. We do not impose an awkward attitude representation such as Euler angles, or quaternions as they require a norm constraint. Instead we see rotations as elements of the Lie group $\mathbf{SO}(3)$. Using the group structure leads to a graceful formulation that is neither influenced nor encumbered by a particular representation of rotation. We treat error and update terms as small angular motions, elements of the Lie algebra $\mathfrak{so}(3)$; the small rotation corresponding to a small angular motion can be computed with the exponential map, and we can compose the result with a current attitude estimate using multiplication. Thus, we actually have a multiplicative EKF for the attitude part of the system state. The linearization necessary for the EKF is achieved using a first order Taylor series approximation of the exponential map $\mathfrak{so}(3) \rightarrow \mathbf{SO}(3)$.

We believe this formulation is ultimately a simpler cleaner approach to real-time pose estimation, and its unencumbered and computationally inexpensive nature makes it suitable for not only AR, but for mobile robotics as well. Moreover, our principled derivation may easily be applied to more complex process and measurement models. We evaluate the performance of our approach using ground truth data obtained via a motion capture system, and we demonstrate our technique in a simple AR application running on an Android based tablet (see a preview in Figure 1). We also discuss practical issues of our implementation, limitations, and future work.

II. RELATED WORK

There has been a great deal of success using purely vision-based methods to estimate a camera’s attitude and position. An especially prevalent approach in AR is fiducial marker tracking. *ARToolKit* [12] and *ARToolKitPlus* [22] are popular examples, while *CyberCode* [18] is another. Alternatively, arbitrary images can be used as markers provided they have sufficient texture, as is the case with Qualcomm’s *Vuforia* SDK, or Robocortex’s *Rox Tracking* SDK.

Similar in spirit to marker tracking is model-based tracking, in which *a priori* 3D objects are tracked. As with marker tracking, model-based tracking enables the recovery of the camera’s pose. For example, Reitmayr and Drummond [17] track a textured model using edges for an outdoor AR application.

In the mobile robotics community simultaneous localization and mapping (SLAM) techniques have been highly successful, and these methods — particularly the ones that run in real-time — have direct use in AR. Furthermore, mapping the user’s environment also enables interesting applications that would be difficult with marker tracking alone. Davison et al. [5] developed one of the first real-time single camera SLAM systems, *MonoSLAM*, which uses an EKF approach; Eade and Drummond [7] developed a monocular SLAM system

Contribution	Method
Koller 1997 [15]	Multiplicative (except innovation) EKF; rotation vectors; single camera
You 2001 [23]	Additive EKF; rotation vectors; single camera, gyro
Davison 2007 [5]	SLAM (EKF); quaternions; single camera
Eade 2007 [7]	SLAM (bundle adjustment); $\mathbf{SE}(3)$, single camera
Klein 2007 [13]	SLAM (bundle adjustment); $\mathbf{SE}(3)$, single camera
Strasdat 2010 [21]	SLAM (bundle adjustment); $\mathbf{SE}(3)$; single camera
Servant 2010 [19]	EKF; rotation vectors; single camera, IMU
Our formulation	Multiplicative EKF, rotation vectors, rotation matrices; single camera, gyro

Table 1. Overview of related work detailing the high-level pose estimation strategy of various contributions.

that runs in real-time using bundle adjustment; and Klein and Murray’s *Parallel Tracking and Mapping* (PTAM) [13] solution performs SLAM specifically for AR applications.

More recently Klein and Murray ported their PTAM system to a mobile phone [14], and Strasdat et al. [21] developed a SLAM system for large maps using bundle adjustment with a $\mathbf{SE}(3)$ formulation. Servant et al. [19] developed a plane based SLAM algorithm that also uses inertial measurements, while describing their attitude dynamics using an axis-angle representation. Moreover, 13th Lab’s *PointCloud* SDK brings commercially viable SLAM capabilities to modern mobile phones.

Our work is primarily concerned with sensor fusion based pose estimation for AR, and in particular EKF methods. Koller et al. [15] used an EKF with a single camera and an acceleration-level process model; You et al. [23] used an additive EKF with gyro measurements; Foxlin et al. [9] developed a wearable tracking device that fuses vision and IMU measurements using an EKF; and more recently, Bleser and Stricker [2] used a sensor fusion model-based tracker using a quaternion based EKF.

Attitude estimation using quaternion based techniques are very popular in the aeronautics community, for instance, see Crassidis [3]. However, using quaternions within an EKF requires either a normalization step or a norm constraint [24] within the filter to maintain a valid rotation. We avoid these complexities and the issues of representation by using the group structure of rotations in our EKF formulation.

In Table 1 we summarize the methods of various contributions, detailing the high-level pose estimation strategy, attitude representation, and sensors used. Our work differs in several aspects from those outlined. We are not performing SLAM, and we focus only on pose estimation. Thus, our solution is computationally inexpensive compared to SLAM, making it applicable for mobile applications that require computing resources for other tasks, such as rendering or physics simulation. Furthermore, our formulation’s linearization procedure differs from most other EKF based methods — we use rotation matrix perturbations as opposed to taking partial derivatives directly — enabling the integration of other sensors and more

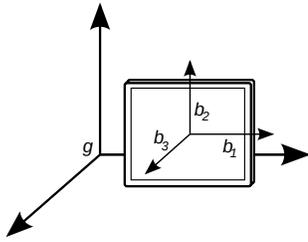


Fig. 2. The global and body reference frames.

complex dynamics easily.

III. POSE ESTIMATION

The pose estimation problem is to determine the device's attitude and position in the global reference frame at the current time while the user is free to move the device with some arbitrary motion.

To solve the pose estimation problem we use a Kalman filter. The Kalman filter is a linear state estimator with a predictor-corrector structure [11]. However, as the pose estimation problem is inherently nonlinear, the state dynamics (Section III-B) and measurement model (Section III-C) must be linearized [4], leading to an EKF. We take the approach of Farrenkopf [8] by using angular velocity measurements via the device's gyroscope within the EKF's prediction step (or propagation step), and vision-based measurements via fiducial tracking in the EKF's correction step.

The following subsections synthesize our EKF formulation by deriving the various components of the EKF, and describe the details of the prediction and correction steps. We assume the reader has some familiarity with the Kalman filter, so we avoid the preliminaries. The Kalman filter equations can be found in Crassidis and Junkins [4], and Simon [20].

Firstly, we use the following reference frames in the definition of the system state and synthesis of our EKF formulation (see Figure 2). The global frame g is an inertial frame defined during the initialization step (usually fixed to a fiducial marker). The body frame b is attached to the device with its origin at the center of the device. With respect to the device, the first axis b_1 points to the right, the second axis b_2 points out of the top, and the third axis b_3 points out of the screen.

A. System State

The system state is composed of the time-varying quantities that model the underlying physical process of rigid motion. Our state includes the device's attitude, position, and gyroscope bias. We do not include the device's linear velocity as we are not using an accelerometer, and in practice we found that using strictly vision-based measurements to estimate velocity resulted in poor performance. Hence, our state consists of three quantities,

$$\mathbf{C}_{bg}(t), \mathbf{p}_g(t), \text{ and } \mathbf{b}_b(t),$$

where $\mathbf{C}_{bg} \in \mathbf{SO}(3)$ rotates the global frame to the body frame, $\mathbf{p}_g \in \mathbb{R}^3$ is the origin of the body frame with respect to the global frame expressed in the global frame, and $\mathbf{b}_b \in \mathbb{R}^3$ is the gyroscope bias expressed in the body frame.

B. State Dynamics

The state dynamics describe how the system state evolves over time, in our case we have

$$\dot{\mathbf{C}}_{bg}(t) = -(\boldsymbol{\omega}_b^{bg}(t))^\times \mathbf{C}_{bg}(t), \quad (1)$$

$$\dot{\mathbf{p}}_g(t) = \boldsymbol{\nu}_p(t), \quad (2)$$

$$\dot{\mathbf{b}}_b(t) = \boldsymbol{\nu}_b(t), \quad (3)$$

where $\boldsymbol{\omega}_b^{bg}$ is the true angular velocity of the body frame relative to the global frame expressed in the body frame, $\boldsymbol{\nu}_p$ is the random noise affecting the position evolution, $\boldsymbol{\nu}_b$ is the random noise affecting the gyroscope bias evolution, and $(\cdot)^\times$ converts a 3×1 column matrix into a 3×3 skew-symmetric matrix; i.e., given $\mathbf{a} \in \mathbb{R}^3$,

$$\mathbf{a}^\times \triangleq \begin{bmatrix} 0 & -a_3 & a_2 \\ a_3 & 0 & -a_1 \\ -a_2 & a_1 & 0 \end{bmatrix}.$$

The attitude evolution (Equation 1) is described by Poisson's equation [10], and the position evolution (Equation 2) describes a random walk, as there is no simple method available to predict how the user will move the device.

The true angular velocity and gyroscope bias evolution (Equation 3) comes from the commonly used gyroscope sensor model presented by Farrenkopf [8], and Crassidis and Junkins [4], and is expressed as

$$\boldsymbol{\omega}_b^{bg}(t) = \boldsymbol{\omega}_{b,m}^{bg}(t) - \mathbf{b}_b(t) + \boldsymbol{\nu}_\omega(t), \quad (4)$$

where $\boldsymbol{\omega}_{b,m}^{bg}$ is the measured angular velocity of the body frame relative to the global frame expressed in the body frame, and $\boldsymbol{\nu}_\omega$ is the random noise (e.g., electrical) affecting the true angular velocity.

In the following two subsections, we discretize and linearize the continuous-time state dynamics given by Equations 1, 2, and 3 in order to synthesize our EKF formulation. Note, for the remainder of this paper, for the sake of brevity, all superscripts and subscripts pertaining to reference frames are omitted.

1) *Discrete-time State Dynamics*: The discrete-time state dynamics describe how the system state is evolved over time in a discrete setting. To obtain these dynamics we integrate the differentials in Equations 1, 2, and 3 by a first order integrator over a time interval $T = t_k - t_{k-1}$. As such, we have

$$\mathbf{C}_k = \boldsymbol{\Psi}_k \mathbf{C}_{k-1}, \quad (5)$$

$$\mathbf{p}_k = \mathbf{p}_{k-1} + T \boldsymbol{\nu}_{p,k-1}, \quad (6)$$

$$\mathbf{b}_k = \mathbf{b}_{k-1} + T \boldsymbol{\nu}_{b,k-1}, \quad (7)$$

where,

$$\boldsymbol{\Psi}_k = \exp(-\boldsymbol{\psi}_k^\times), \quad (8)$$

$$\boldsymbol{\psi}_k = T \boldsymbol{\omega}_k,$$

and $\exp(\cdot)$ is the exponential map $\mathfrak{so}(3) \rightarrow \mathbf{SO}(3)$ [16].

As we are in a Kalman filter setting we must of course make some assumptions about the noise affecting our state

dynamics. Given that our process noise \mathbf{w}_k is

$$\mathbf{w}_k = \begin{bmatrix} \boldsymbol{\nu}_{\omega,k} \\ \boldsymbol{\nu}_{p,k} \\ \boldsymbol{\nu}_{b,k} \end{bmatrix}, \quad (9)$$

we assume that \mathbf{w}_k is drawn from a zero-mean multivariate Gaussian distribution with covariance \mathbf{Q}_k . The covariance matrix \mathbf{Q}_k is simply

$$\begin{aligned} \mathbf{Q}_k &= E[\mathbf{w}_k \mathbf{w}_k^\top] \\ &= \begin{bmatrix} \mathbf{1} \sigma_\omega^2 & \mathbf{0} & \mathbf{0} \\ \mathbf{0} & \mathbf{1} \sigma_p^2 & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & \mathbf{1} \sigma_b^2 \end{bmatrix}, \end{aligned}$$

where σ_ω^2 is the variance of the true angular velocity noise, σ_p^2 is the variance of the position noise, and σ_b^2 is the variance of the gyroscope bias noise. In section IV we describe how we derive the aforementioned variances.

2) *Linear Discrete-time State Dynamics*: The discrete-time state dynamics are linearized by performing a first order Taylor series approximation about the state's nominal trajectory [4]. We perturb a rotation matrix via multiplication, and a vector quantity via addition. So, to perform the linearization we compute

$$\begin{aligned} \mathbf{C}_k &= \delta \mathbf{C}_k \bar{\mathbf{C}}_k, \\ \mathbf{p}_k &= \bar{\mathbf{p}}_k + \delta \mathbf{p}_k, \\ \mathbf{b}_k &= \bar{\mathbf{b}}_k + \delta \mathbf{b}_k, \end{aligned}$$

while neglecting all second order and greater perturbation terms (i.e., those terms adorned with “ δ ”). After linearizing we recover the process model \mathbf{F}_k and process noise model \mathbf{L}_k for the EKF.

Firstly, we must consider how to perturb some arbitrary rotation matrix $\mathbf{R} \in \mathbf{SO}(3)$. Barfoot et al. [1] show that a rotation matrix perturbation can be approximated by

$$\delta \mathbf{R} = \mathbf{1} - \delta \boldsymbol{\theta}^\times,$$

where $\delta \boldsymbol{\theta}$ is a rotation vector. Note, $\mathbf{R} = \exp(-\boldsymbol{\theta}^\times)$. Thus, we can write

$$\mathbf{R} = (\mathbf{1} - \delta \boldsymbol{\theta}^\times) \bar{\mathbf{R}}. \quad (10)$$

Secondly, we must consider how to perturb the true angular velocity $\boldsymbol{\omega}_k$. By recognizing that the nominal value of $\boldsymbol{\omega}_k$ is an unbiased and noiseless measurement of the true angular velocity, we can write

$$\begin{aligned} \boldsymbol{\omega}_k &= \bar{\boldsymbol{\omega}}_k + \delta \boldsymbol{\omega}_k, \\ \bar{\boldsymbol{\omega}}_k &= \boldsymbol{\omega}_{m,k} - \bar{\mathbf{b}}_{k-1}, \end{aligned} \quad (11)$$

$$\delta \boldsymbol{\omega}_k = -\delta \mathbf{b}_{b,k-1} + \delta \boldsymbol{\nu}_{\omega,k}. \quad (12)$$

Equations 11 and 12 are needed to linearize expressions involving $\bar{\Psi}_k$ (Equation 8).

With Equations 10, 11, and 12 we linearize Equation 5. First we consider that $\mathbf{C}_k = \exp(-\phi_k^\times)$, and we perturb each rotation matrix [1],

$$(\mathbf{1} - \delta \phi_k^\times) \bar{\mathbf{C}}_k = (\mathbf{1} - \delta \psi_k^\times) \bar{\Psi}_k (\mathbf{1} - \delta \phi_{k-1}^\times) \bar{\mathbf{C}}_{k-1}.$$

Expanding and neglecting all perturbation terms of degree two or greater gives

$$(\mathbf{1} - \delta \phi_k^\times) \bar{\mathbf{C}}_k = \bar{\Psi}_k \bar{\mathbf{C}}_{k-1} - \bar{\Psi}_k \delta \phi_{k-1}^\times \bar{\mathbf{C}}_{k-1} - \delta \psi_k^\times \bar{\Psi}_k \bar{\mathbf{C}}_{k-1}.$$

Subtracting the nominal solution $\bar{\mathbf{C}}_k = \bar{\Psi}_k \bar{\mathbf{C}}_{k-1}$ we have

$$\begin{aligned} \delta \phi_k^\times \bar{\mathbf{C}}_k &= \bar{\Psi}_k \delta \phi_{k-1}^\times \bar{\mathbf{C}}_{k-1} + \delta \psi_k^\times \bar{\Psi}_k \bar{\mathbf{C}}_{k-1}, \\ \delta \phi_k^\times \bar{\Psi}_k \bar{\mathbf{C}}_{k-1} &= \bar{\Psi}_k \delta \phi_{k-1}^\times \bar{\mathbf{C}}_{k-1} + \delta \psi_k^\times \bar{\Psi}_k \bar{\mathbf{C}}_{k-1}. \end{aligned}$$

Right multiplying by $\bar{\mathbf{C}}_{k-1}^\top \bar{\Psi}_k^\top$,

$$\delta \phi_k^\times = \bar{\Psi}_k \delta \phi_{k-1}^\times \bar{\Psi}_k^\top + \delta \psi_k^\times,$$

then using the identity $(\mathbf{C}\mathbf{a})^\times = \mathbf{C}\mathbf{a}^\times \mathbf{C}^\top$ for some $\mathbf{C} \in \mathbf{SO}(3)$ and $\mathbf{a} \in \mathbb{R}^3$ [10],

$$\delta \phi_k^\times = (\bar{\Psi}_k \delta \phi_{k-1}^\times)^\times + \delta \psi_k^\times.$$

Lastly, reversing the cross operator and substituting in $\delta \boldsymbol{\omega}_k$ yields

$$\begin{aligned} \delta \phi_k &= \bar{\Psi} \delta \phi_{k-1} + \delta \psi_k \\ &= \bar{\Psi} \delta \phi_{k-1} + T \delta \boldsymbol{\omega}_k \\ &= \bar{\Psi} \delta \phi_{k-1} + T (-\delta \mathbf{b}_{k-1} + \delta \boldsymbol{\nu}_{\omega,k}) \\ &= \bar{\Psi} \delta \phi_{k-1} - T \delta \mathbf{b}_{k-1} + T \delta \boldsymbol{\nu}_{\omega,k}, \end{aligned} \quad (13)$$

giving us an expression for evolving the attitude perturbation. Note, $\bar{\Psi}_k = \exp(-T(\boldsymbol{\omega}_{m,k} - \bar{\mathbf{b}}_{k-1})^\times)$.

As Equations 6 and 7 are already linear in \mathbf{p}_{k-1} and \mathbf{b}_{k-1} respectively, we can simply write

$$\delta \mathbf{p}_k = \delta \mathbf{p}_{k-1} + T \delta \boldsymbol{\nu}_{p,k-1}, \quad (14)$$

$$\delta \mathbf{b}_k = \delta \mathbf{b}_{k-1} + T \delta \boldsymbol{\nu}_{b,k-1}, \quad (15)$$

giving us expressions for evolving the position and gyroscope bias perturbations.

With Equations 13, 14, and 15 we write

$$\begin{aligned} \begin{bmatrix} \delta \phi_k \\ \delta \mathbf{p}_k \\ \delta \mathbf{b}_k \end{bmatrix} &= \begin{bmatrix} \bar{\Psi}_k & \mathbf{0} & -T \mathbf{1} \\ \mathbf{0} & \mathbf{1} & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & \mathbf{1} \end{bmatrix} \begin{bmatrix} \delta \phi_{k-1} \\ \delta \mathbf{p}_{k-1} \\ \delta \mathbf{b}_{k-1} \end{bmatrix} \\ &+ \begin{bmatrix} T \mathbf{1} & \mathbf{0} & \mathbf{0} \\ \mathbf{0} & T \mathbf{1} & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & T \mathbf{1} \end{bmatrix} \begin{bmatrix} \delta \boldsymbol{\nu}_{\omega,k-1} \\ \delta \boldsymbol{\nu}_{p,k-1} \\ \delta \boldsymbol{\nu}_{b,k-1} \end{bmatrix}, \end{aligned}$$

that is,

$$\delta \mathbf{x}_k = \mathbf{F}_k \delta \mathbf{x}_{k-1} + \mathbf{L}_k \delta \mathbf{w}_{k-1},$$

where \mathbf{x}_k is the state vector. Note, since the dynamics are linearized the expressions for evolving the perturbations are the same for evolving the state (due to superposition).

C. Measurement Model

The measurement model describes how exteroceptive measurements are made. As we use full pose measurements we have

$$\mathbf{C}_{m,k} = \mathbf{N}_{m,k} \mathbf{C}_k, \quad (16)$$

$$\mathbf{p}_{m,k} = \mathbf{p}_k + \boldsymbol{\nu}_{p_m,k}, \quad (17)$$

where $\mathbf{C}_{m,k}$ is the measured rotation from the global frame to the body frame, $\mathbf{p}_{m,k}$ is the measured origin of the body frame with respect to the global frame expressed in the global frame, $\mathbf{N}_{m,k} \in \mathbf{SO}(3)$ is the random noise affecting the attitude measurement, and $\boldsymbol{\nu}_{p_{m,k}}$ is the random noise affecting the position measurement. Note, the gyroscope bias cannot be directly measured on the device.

As with the process noise we must make some assumptions about the noise affecting our measurements. First we parameterize the attitude measurement noise as a rotation vector, that is, $\mathbf{N}_{m,k} = \exp(-\boldsymbol{\zeta}_k)$. Thus, our measurement noise \mathbf{v}_k is

$$\mathbf{v}_k = \begin{bmatrix} \boldsymbol{\zeta}_k \\ \boldsymbol{\nu}_{p_{m,k}} \end{bmatrix}, \quad (18)$$

and we assume that \mathbf{v}_k is drawn from a zero-mean multivariate Gaussian distribution with covariance \mathbf{R}_k . The covariance matrix \mathbf{R}_k is

$$\begin{aligned} \mathbf{R}_k &= E[\mathbf{v}_k \mathbf{v}_k^\top] \\ &= \begin{bmatrix} \mathbf{1} \sigma_\zeta^2 & \mathbf{0} \\ \mathbf{0} & \mathbf{1} \sigma_{p_m}^2 \end{bmatrix}, \end{aligned}$$

where σ_ζ^2 is the variance of the rotation vector parameterization of the measured attitude noise, and $\sigma_{p_m}^2$ is the variance of the measured position noise.

In the following subsection we linearize the measurement model en route to synthesizing our EKF formulation.

1) *Linear Measurement Model:* The measurement model is linearized the same way as the discrete-time state dynamics, and after linearizing we recover the measurement model \mathbf{H}_k for the EKF. With Equation 10 we linearize Equation 16. First we consider that $\mathbf{C}_{m,k} = \exp(-\boldsymbol{\phi}_{m,k})$, and we perturb each rotation matrix,

$$(\mathbf{1} - \delta\boldsymbol{\phi}_{m,k}^\times) \bar{\mathbf{C}}_{m,k} = (\mathbf{1} - \delta\boldsymbol{\zeta}_k^\times) \bar{\mathbf{N}}_{m,k} (\mathbf{1} - \delta\boldsymbol{\phi}_k^\times) \bar{\mathbf{C}}_k.$$

Now, since $\boldsymbol{\zeta}_k$ is drawn from a zero-mean distribution, $\bar{\mathbf{N}}_{m,k} = \mathbf{1}$, such that

$$(\mathbf{1} - \delta\boldsymbol{\phi}_{m,k}^\times) \bar{\mathbf{C}}_{m,k} = (\mathbf{1} - \delta\boldsymbol{\zeta}_k^\times) (\mathbf{1} - \delta\boldsymbol{\phi}_k^\times) \bar{\mathbf{C}}_k.$$

Expanding and neglecting all perturbation terms of degree two or greater, we have

$$(\mathbf{1} - \delta\boldsymbol{\phi}_{m,k}^\times) \bar{\mathbf{C}}_{m,k} = (\mathbf{1} - \delta\boldsymbol{\zeta}_k^\times - \delta\boldsymbol{\phi}_k^\times) \bar{\mathbf{C}}_k.$$

Recognizing that the nominal solution is simply $\bar{\mathbf{C}}_{m,k} = \bar{\mathbf{C}}_k$,

$$\begin{aligned} \mathbf{1} - \delta\boldsymbol{\phi}_{m,k}^\times &= \mathbf{1} - \delta\boldsymbol{\zeta}_k^\times - \delta\boldsymbol{\phi}_k^\times, \\ \delta\boldsymbol{\phi}_{m,k}^\times &= \delta\boldsymbol{\zeta}_k^\times + \delta\boldsymbol{\phi}_k^\times, \\ \delta\boldsymbol{\phi}_{m,k} &= \delta\boldsymbol{\phi}_k + \delta\boldsymbol{\zeta}_k, \end{aligned} \quad (19)$$

giving us an expression for mapping the attitude perturbation from state space to measurement space.

As Equation 17 is already linear in \mathbf{p}_k we can simply write

$$\delta\mathbf{p}_{m,k} = \delta\mathbf{p}_k + \delta\boldsymbol{\nu}_{p_{m,k}}, \quad (20)$$

giving us an expression for mapping the position perturbation from state space to measurement space.

With Equations 19 and 20 we write

$$\begin{bmatrix} \delta\boldsymbol{\phi}_{m,k} \\ \delta\mathbf{p}_{m,k} \end{bmatrix} = \begin{bmatrix} \mathbf{1} & \mathbf{0} & \mathbf{0} \\ \mathbf{0} & \mathbf{1} & \mathbf{0} \end{bmatrix} \begin{bmatrix} \delta\boldsymbol{\phi}_k \\ \delta\mathbf{p}_k \\ \delta\mathbf{b}_k \end{bmatrix} + \begin{bmatrix} \delta\boldsymbol{\zeta}_k \\ \delta\boldsymbol{\nu}_{p_{m,k}} \end{bmatrix},$$

that is,

$$\delta\mathbf{y}_k = \mathbf{H}_k \delta\mathbf{x}_k + \delta\mathbf{v}_k,$$

where \mathbf{y}_k is the measurement vector.

D. Extended Kalman Filter

Now that we have derived \mathbf{F}_k , \mathbf{L}_k , \mathbf{H}_k , and the covariance matrices \mathbf{Q}_k and \mathbf{R}_k , we can apply the EKF to the pose estimation problem.

1) *Initialization:* The initialization step sets the value of the initial state and error covariance estimates. The initial state estimate is simply set to the expected value of the state, that is,

$$\hat{\mathbf{C}}_0 = E[\mathbf{C}_0], \quad \hat{\mathbf{p}}_0 = E[\mathbf{p}_0], \quad \text{and} \quad \mathbf{b}_0 = E[\mathbf{b}_0],$$

and the initial error covariance estimate $\hat{\mathbf{P}}_0$ is set to the expected value of the initial residual error $\tilde{\mathbf{x}}_0$. The residual error is the difference between the actual state and the estimated state, that is,

$$\tilde{\mathbf{x}}_k = \begin{bmatrix} \tilde{\boldsymbol{\phi}}_k \\ \mathbf{p}_k - \hat{\mathbf{p}}_k \\ \mathbf{b}_k - \hat{\mathbf{b}}_k \end{bmatrix},$$

where $\tilde{\boldsymbol{\phi}}_k^\times = \log(\mathbf{C}_k \hat{\mathbf{C}}_k^\top)$, and $\log(\cdot)$ is the log map from $\mathbf{SO}(3) \rightarrow \mathfrak{so}(3)$ [16]. Thus,

$$\hat{\mathbf{P}}_0 = E[\tilde{\mathbf{x}}_0 \tilde{\mathbf{x}}_0^\top].$$

2) *Prediction:* The prediction step is performed to compute a new state estimate and error covariance estimate using a gyroscope measurement. The *a priori* state estimate is computed using the discrete-time state dynamics while ignoring the noise components (Equations 5, 6, and 7). The *a priori* error covariance estimate $\hat{\mathbf{P}}_k^-$ is computed by

$$\hat{\mathbf{P}}_k^- = \mathbf{F}_k \hat{\mathbf{P}}_{k-1} \mathbf{F}_k^\top + \mathbf{L}_k \mathbf{Q}_k \mathbf{L}_k^\top, \quad (21)$$

where \mathbf{F}_k and \mathbf{L}_k are evaluated using the current state estimate in place of the nominal state (i.e., linearization occurs about the current state estimate).

3) *Correction:* The correction step is performed to compute a new state estimate and error covariance estimate using a vision-based measurement. The innovation $\tilde{\mathbf{y}}_k$ is computed via

$$\tilde{\mathbf{y}}_k = \begin{bmatrix} \tilde{\boldsymbol{\phi}}_{m,k} \\ \mathbf{p}_{m,k} - \hat{\mathbf{p}}_k^- \end{bmatrix}, \quad (22)$$

where $\tilde{\boldsymbol{\phi}}_{m,k}^\times = \log(\mathbf{C}_{m,k} \hat{\mathbf{C}}_k^{-\top})$. The Kalman Gain \mathbf{K}_k is computed by

$$\mathbf{K}_k = \hat{\mathbf{P}}_k^- \mathbf{H}_k^\top \mathbf{W}_k^{-1}, \quad (23)$$

where

$$\mathbf{W}_k = \mathbf{H}_k \hat{\mathbf{P}}_k^- \mathbf{H}_k^\top + \mathbf{R}_k. \quad (24)$$

Now, using Equations 22 and 23 the correction terms are computed via

$$\begin{bmatrix} \Delta\phi_k \\ \Delta\mathbf{p}_k \\ \Delta\mathbf{b}_k \end{bmatrix} = \mathbf{K}_k \tilde{\mathbf{y}}_k, \quad (25)$$

and the *a posteriori* state estimate is given by

$$\hat{\mathbf{C}}_k = \exp(\Delta\phi_k^\times) \hat{\mathbf{C}}_k^-, \quad (26)$$

$$\hat{\mathbf{p}}_k = \hat{\mathbf{p}}_k^- + \Delta\mathbf{p}_k, \quad (27)$$

$$\hat{\mathbf{b}}_k = \hat{\mathbf{b}}_k^- + \Delta\mathbf{b}_k. \quad (28)$$

Equations 22 through 28 taken together are analogous to the single correction equation of the standard Kalman filter; however, as we have a multiplicative structure we treat the attitude parts differently. Lastly, the *a posteriori* error covariance estimate is computed via the Joseph Formula,

$$\hat{\mathbf{P}}_k = \hat{\mathbf{P}}_k^- - \mathbf{K}_k \mathbf{H}_k \hat{\mathbf{P}}_k^- - \hat{\mathbf{P}}_k^- \mathbf{H}_k^\top \mathbf{K}_k^\top + \mathbf{K}_k \mathbf{W}_k \mathbf{K}_k^\top.$$

Now that we have shown how to synthesize an EKF formulation for the pose estimation problem using linearized rotation matrices, we discuss some implementation details and results of our approach.

IV. IMPLEMENTATION DETAILS

Our pose estimation method has been implemented for Android devices and tested on the *Samsung Galaxy 10.1* tablet (with a 1 GHz dual-core *NVIDIA Tegra 2* processor). Sensor measurements are performed using Android’s standard API, and vision-based measurements are performed using Qualcomm’s *Vuforia* SDK. Implementation of our EKF formulation has been relatively straightforward. Our implementation has been written partly in *Java* and partly in *C++*, and has been integrated into our larger AR framework which will be used for future work (see Section VI).

Although the primary components of the EKF are straightforward to implement, there exist a number of challenges: handling the delay from when a camera image was taken to when the actual vision-based measurement is available, ensuring vision-based measurements have consistent noise characteristics, and tuning the EKF’s covariance matrices. We address these issues below.

Due to the exposure time of the camera and the image processing time required by the fiducial tracking system, the time when a vision-based measurement is available is delayed from the time when the actual camera image was taken. This causes difficulty when propagating the state because the gyro measurements must be synchronized with the vision-based measurements, i.e., gyro measurements that happen in the future with respect to a camera image should not be used when propagating the state. To mitigate this difficulty we buffer the gyro measurements, and when a camera image becomes available we propagate the state multiple times using the applicable buffered gyro measurements. After multiple propagations the state is corrected using the vision-based measurement. A similar scheme is used in Servant [19].

In our current implementation we use a constant measurement noise covariance. As such, it is crucial to use vision-based measurements that have consistent noise characteristics (simply using a larger than otherwise normal measurement noise covariance results in ineffective corrections). To satisfy this desire we use a temporally based infinite impulse response filter (a pose change filter) that allows only pose measurements that are within a certain range of an exponentially weighted average of past pose measurements to be used within a correction step. We use a filter coefficient of 0.5, a maximum attitude change of 7.5 degrees, and a maximum position change of 3.2 centimeters.

Tuning the covariance matrices of the EKF can be a somewhat delicate process, as the filter’s accuracy, consistency, stability, and qualitative performance must each be considered.

We tuned the covariance matrices by first trying to ensure the estimated error covariance captures the actual uncertainty of the state estimate. We did this by using a sensor test application on the device to get an estimate of the gyro noise, and by using a motion capture system to compute the attitude and position error variance of the fiducial tracking system. However, we further tuned the covariance matrices to ensure stability and good qualitative performance. Stability issues arise as \mathbf{W}_k (Equation 24) may become ill conditioned due to large differences in the various noise variances. This difficulty can be mitigated by tweaking the noise variances and by also using a truncated SVD inverse. Correcting the stability issues also causes smoother pose estimate trajectories as it weakens the correction step, and for AR this is somewhat beneficial as it reduces scene jitter.

V. RESULTS

To quantitatively evaluate the performance and effectiveness of our implementation, we compare the EKF with the fiducial tracking system alone (without the pose change filter) using ground truth data obtained via a motion capture system. Trials were performed by subjecting the device to a number of different types of motion with the camera having constant line of sight (LOS) to a tracked image fiducial at all times (note, even with constant LOS, vision-based measurements are not always possible due to motion blur). The EKF’s estimated state and the fiducial tracking system’s pose measurements were simultaneously acquired while also collecting motion capture data.

The motion capture system was calibrated by physically aligning the global frame of the motion capture system with an image marker fastened to the floor (which defines the global frame of our implementation). Finer grained calibration was then performed by capturing data with the device at rest, and adjusting for any bias — either caused by misalignment of the global frames, or slightly imprecise initial calibration of the motion capture system’s rigid body tracking system.

Below we discuss some data obtained from a representative trial. In this particular trial we maneuvered the device through various rotational motions while keeping the speed relatively constant. The speed was fast enough such that motion blur

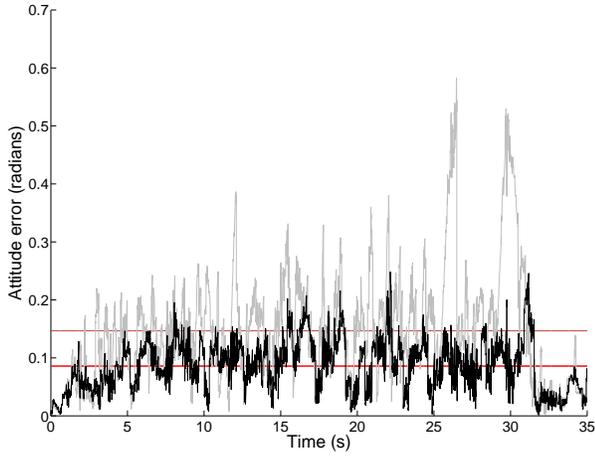


Fig. 3. Attitude error magnitude, EKF in thick black, fiducial tracking system in thin gray, EKF MAE bottom horizontal line, and fiducial tracking system MAE top horizontal line.

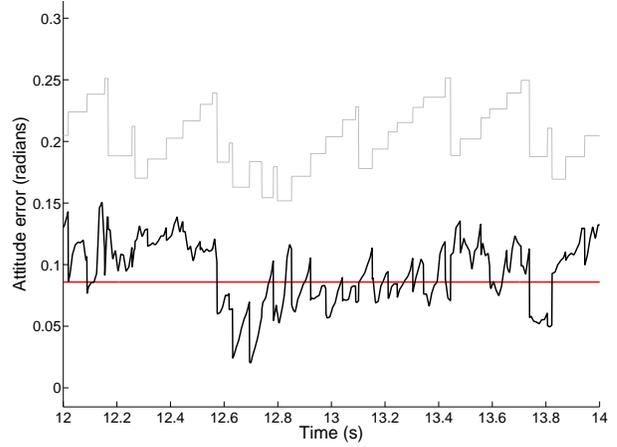


Fig. 4. EKF attitude error magnitude, with estimated standard deviation of the attitude error magnitude (3 sigma) in thin gray.

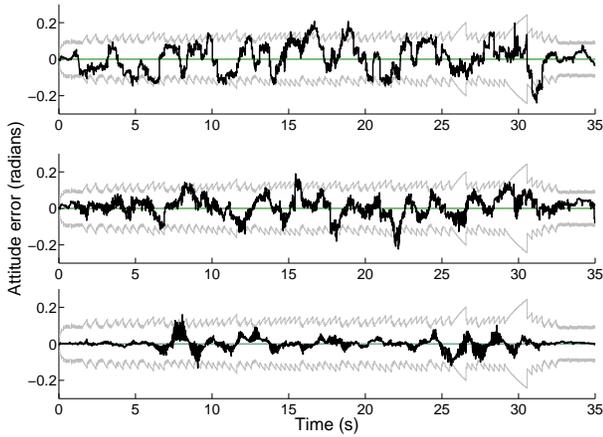


Fig. 5. EKF attitude error about body axes 1, 2, and 3 (from top to bottom), with estimated standard deviation of the attitude error (3 sigma) in thin gray.

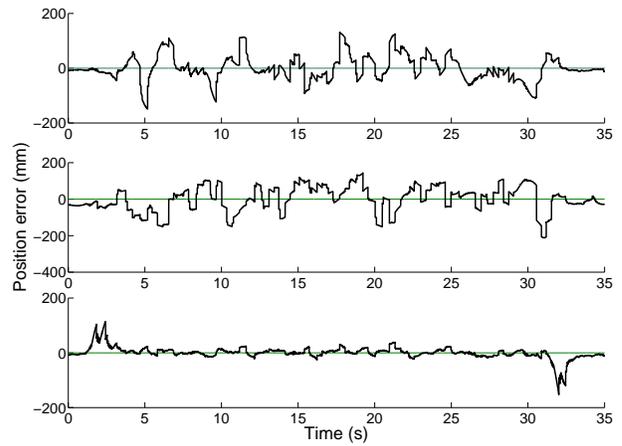


Fig. 6. EKF position error along global axes 1, 2, and 3 (from top to bottom).

would occur, but also reasonable in that a user could still perceive the images on the device’s display (i.e., we emulated a typical usage scenario).

Figure 3 shows the magnitude of the attitude error from the EKF (thick black line) and from the fiducial tracking system (thin gray line). The error is computed by taking the L2 norm of the rotation vector corresponding to the rotation that brings the estimated attitude to the ground truth attitude. The bottom horizontal line shows the mean absolute error (MAE) from the EKF, equal to 4.9 degrees, and the top horizontal line shows the MAE from the fiducial tracking system, equal to 8.4 degrees. The EKF has a maximum error of 14 degrees, and the fiducial tracking system has a maximum error of 33 degrees.

Figure 4 shows a short interval of the attitude error magnitude from the EKF, along with the estimated standard deviation of the attitude error magnitude. The horizontal line is the

MAE. This figure more clearly shows the correlations between the estimated uncertainty and actual error. Note, the estimated standard deviation is not smooth because it is updated discretely on the device.

Figure 5 shows the attitude error from the EKF (thick black line) about the 3 axes of the body frame, each with a 3 sigma envelope showing the estimated standard deviation of the attitude error (i.e., the first 3 diagonal entries of \hat{P}_k). The appearance of the 3 sigma envelope is typical of an EKF; it shows that as propagation occurs, the uncertainty in the state estimate grows, and when a correction step occurs, the uncertainty is instantaneously reduced.

Figure 6 shows the position error from the EKF. The position estimate is updated much less frequently than the attitude estimate as it can only be updated during a correction step.

As expected the EKF provides a more consistent attitude

estimate with a lower MAE compared to the fiducial tracking system alone. We found similar results over many different trials with different motion types. We also applied the normalized error square (NES) test [4], [6] to check the EKF's consistency for each trial, and found that in most cases the filter's error dynamics are zero-mean white noise processes within a 95% confidence interval.

VI. CONCLUSIONS & FUTURE WORK

Here, we develop a principled approach to linearizing the equations of motion and measurement model of an EKF formulation for rigid motion tracking. We perform linearization using a first order Taylor series approximation of the exponential map $so(3) \rightarrow SO(3)$, and use a multiplicative EKF structure. The technique we develop has the added advantage of having an elegant formulation, avoiding the complexities found in other methods based on Euler angles or quaternions.

We have shown quantitatively that our approach improves purely vision-based methods. Qualitatively the results are clear as well; using the filter significantly reduces jitter in augmented objects. Moreover, the inexpensive processing required does not preclude other computationally expensive tasks from being executed (e.g., rendering). Our implementation takes approximately 3.5 milliseconds on average to perform a series of prediction steps followed by a correction step (not including the time taken to perform fiducial marker tracking). We believe that our approach offers a good balance between performance and computational complexity, making it an applicable solution for any mobile applications requiring pose estimation.

Our approach extends naturally to accommodate other sources of information, and we plan on incorporating an accelerometer and magnetometer in future work — both of which are available on most devices. We also plan to explore EKF formulations based on $SE(3)$, as opposed to using decoupled attitude and position dynamics as we have done here. Furthermore, we would like to produce comparisons between multiple pose estimation filters, especially the norm constraint Kalman filter, using ground truth motion capture data. Our current work is aimed at demonstrating the practical utility of our approach in the context of application development, where we are using the method here as part of a larger AR framework.

ACKNOWLEDGEMENTS

This work was supported by funding from NSERC and GRAND NCE. We thank the anonymous reviewers for their suggestions for improving the paper. The third author thanks Dr. Timothy Barfoot for exposing him to the idea of using linearized rotations within an EKF structure, which is also discussed in the course AER 1513 - State Estimation for Aerospace Vehicles (and the accompanying course notes), taught by Dr. Barfoot at the University of Toronto, Institute for Aerospace Studies.

REFERENCES

- [1] T. Barfoot, J. R. Forbes, and P. T. Furgale. Pose estimation using linearized rotations and quaternion algebra. *Acta Astronautica*, 68:101–112, 2010.
- [2] G. Bleser and D. Stricker. Advanced tracking through efficient image processing and visual-inertial sensor fusion. *Computers & Graphics*, 33(1):59 – 72, 2009.
- [3] J. Crassidis, F. Markley, and Y. Cheng. Survey of nonlinear attitude estimation methods. *Journal of guidance control and dynamics*, 30(1):12, 2007.
- [4] J. L. Crassidis and J. L. Junkins. *Optimal Estimation of Dynamic Systems*. Chapman & Hall/CRC applied mathematics and nonlinear science series, 2004.
- [5] A. Davison, I. Reid, N. Molton, and O. Stasse. MonoSLAM: Real-time single camera SLAM. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 29(6):1052–1067, 2007.
- [6] H. Durant-Whyte. Introduction to estimation and the Kalman filter. Technical report, Australian Centre for Field Robotics, 2001.
- [7] E. Eade and T. Drummond. Monocular slam as a graph of coalesced observations. In *Computer Vision, 2007. ICCV 2007. IEEE 11th International Conference on*, pages 1–8. IEEE, 2007.
- [8] R. Farrenkopf. Analytic steady-state accuracy solutions for two common spacecraft attitude estimators. *Guidance and Control*, 1(4):282–284, 1978.
- [9] E. Foxlin, L. Naimark, et al. Vis-tracker: A wearable vision-inertial self-tracker. In *Proceedings of the IEEE Virtual Reality*, page 199, 2003.
- [10] P. Hughes. *Spacecraft Attitude Dynamics*. Wiley, 2004.
- [11] R. E. Kalman. A new approach to linear filtering and prediction problems. *Transactions of the ASME – Journal of Basic Engineering*, 82(Series D):35–45, 1960.
- [12] H. Kato and M. Billinghurst. Marker tracking and hmd calibration for a video-based augmented reality conferencing system. In *Augmented Reality, 1999. (IWAR '99) Proceedings. 2nd IEEE and ACM International Workshop on*, pages 85 –94, 1999.
- [13] G. Klein and D. Murray. Parallel tracking and mapping for small ar workspaces. In *Mixed and Augmented Reality, 2007. ISMAR 2007. 6th IEEE and ACM International Symposium on*, pages 225–234. IEEE, 2007.
- [14] G. Klein and D. Murray. Parallel tracking and mapping on a camera phone. In *Mixed and Augmented Reality, 2009. ISMAR 2009. 8th IEEE International Symposium on*, pages 83–86. IEEE, 2009.
- [15] D. Koller, G. Klinker, E. Rose, D. Breen, R. Whitaker, and M. Tuceryan. Real-time vision-based camera tracking for augmented reality applications. In *Proceedings of the ACM symposium on Virtual reality software and technology*, pages 87–94. ACM, 1997.
- [16] R. M. Murray, Z. Li, and S. S. Sastry. *A Mathematical Introduction to Robotic Manipulation*. CRC Press, 1994.
- [17] G. Reitmayr and T. Drummond. Going out: robust model-based tracking for outdoor augmented reality. In *Mixed and Augmented Reality, 2006. ISMAR 2006. IEEE/ACM International Symposium on*, pages 109–118. IEEE, 2006.
- [18] J. Rekimoto and Y. Ayatsuka. Cybercode: designing augmented reality environments with visual tags. In *Proceedings of DARE 2000 on Designing augmented reality environments*, DARE '00, pages 1–10, 2000.
- [19] F. Servant, P. Houlier, E. Marchand, et al. Improving monocular plane-based slam with inertial measures. In *IEEE/RSJ Int. Conf. on Intelligent Robots and Systems, IROS'10*, pages 3810–3815, 2010.
- [20] D. Simon. *State Estimation*. John Wiley & Sons, Inc., 2006.
- [21] H. Strasdat, J. Montiel, and A. Davison. Scale drift-aware large scale monocular slam. In *Proceedings of Robotics: Science and Systems (RSS)*, volume 2, page 5, 2010.
- [22] D. Wagner and D. Schmalstieg. Artoolkitplus for pose tracking on mobile devices. In *Proceedings of 12th Computer Vision Winter Workshop (CVWW'07)*, pages 139–146, 2007.
- [23] S. You and U. Neumann. Fusion of vision and gyro tracking for robust augmented reality registration. In *Proceedings of the Virtual Reality 2001 Conference (VR'01)*, pages 71–78, 2001.
- [24] R. Zanetti, M. Majji, R. H. Bishop, and D. Mortari. Norm-constrained Kalman filtering. *Journal of guidance, control, and dynamics*, 32(5):1458–1465, 2009.