Special Issue Article

Advances in Mechanical Engineering

Advances in Mechanical Engineering
2016, Vol. 8(3) 1–11

DOI: 10.1177/1687814016640996
aime.sagepub.com
SAGE

Computationally efficient visual–inertial sensor fusion for Global Positioning System–denied navigation on a small quadrotor

Chang Liu, Stephen D Prior, WT Luke Teacy and Martin Warner

Abstract

Because of the complementary nature of visual and inertial sensors, the combination of both is able to provide fast and accurate 6 degree-of-freedom state estimation, which is the fundamental requirement for robotic (especially, unmanned aerial vehicle) navigation tasks in Global Positioning System–denied environments. This article presents a computationally efficient visual–inertial fusion algorithm, by separating orientation fusion from the position fusion process. The algorithm is designed to perform 6 degree-of-freedom state estimation, based on a gyroscope, an accelerometer and a monocular visual-based simultaneous localisation and mapping algorithm measurement. It also recovers the visual scale for the monocular visual-based simultaneous localisation and mapping. In particular, the fusion algorithm treats the orientation fusion and position fusion as two separate processes, where the orientation fusion is based on a very efficient gradient descent algorithm, whereas the position fusion is based on a 13-state linear Kalman filter. The elimination of the magnetometer sensor avoids the problem of magnetic distortion, which makes it a power-on-and-go system once the accelerometer is factory calibrated. The resulting algorithm shows a significant computational reduction over the conventional extended Kalman filter, with competitive accuracy. Moreover, the separation between orientation and position fusion processes enables the algorithm to be easily implemented onto two individual hardware elements and thus allows the two fusion processes to be executed concurrently.

Keywords

Visual inertial, sensor fusion, Kalman filter, gradient descent algorithm, unmanned aerial vehicle, quadrotor, Global Positioning System denied, navigation

Date received: 2 November 2015; accepted: 1 March 2016

Academic Editor: Teen-Hang Meen

Introduction

The combination of visual and inertial sensors has been shown to be viable, and the significant performance improvement over a single sensor system has attracted many researchers into the field after the success of SFly project,[1] which enabled the world's first autonomous unmanned aerial vehicle (UAV) in Global Positioning System–denied environments.[2]

In the past 5 years, many prominent research institutions began to develop advanced monocular visual-based simultaneous localisation and mapping (mSLAM) algorithms based on structure from motion

Aeronautics, Astronautics and Computational Engineering Group, Engineering Centre of Excellence, University of Southampton, Southampton, UK

Corresponding author:
Chang Liu, Aeronautics, Astronautics and Computational Engineering Group, Engineering Centre of Excellence, University of Southampton, Boldrewood Campus, Southampton SO16 7QF, UK.
Email: cl21g11@soton.ac.uk

(SFM) theory,[3–11] which are suitable to modern onboard embedded computers. Moreover, the visual scale problem, which was the main challenge of involving monocular vision into the control loop, has been addressed by fusing onboard inertial measurements (accelerometer and gyroscope), called the visual–inertial navigation system (VINS).[12–20]

Almost all of the visual–inertial fusion algorithms, to our knowledge, rely on nonlinear Kalman filter (KF) techniques (extended KF, unscented KF, etc.) to process both the orientation and the position measurement in the same process, this results in a large state vector (usually more than 20 states) and a complex nonlinear system model. However, recent advances in computationally efficient inertial measurement unit (IMU) orientation estimation[21] show a competitive accuracy against Kalman-based algorithms by utilising optimisation-based methods. Thus, in this article, a computationally efficient visual–inertial fusion algorithm is proposed by separating the orientation and the position fusion processes, this maintains the same level of accuracy with nonlinear KF approach. The algorithm is designed to perform a 6 degree-of-freedom (DOF) state estimation, based on a gyroscope, an accelerometer and an mSLAM measurement. It also recovers the visual scale for the mSLAM.

The rest of this article is organised as follows: section 'Algorithm preliminaries' gives an overview of the visual–inertial fusion algorithm; section 'Orientation fusion process' presents the mathematical expression of the orientation filter; and section 'Position fusion process' presents the mathematical expression of the position filter. The implementation, test result and conclusion are shown in the last three sections.

Algorithm preliminaries

Coordinate system

The coordinate system used is shown in Figure 1. All the coordinate frames are defined following the right-hand rule. The earth frame $\{E\}$ is fixed to the world, and z_E-axis is defined to be parallel to gravity vector. The sensor frame $\{S\}$ is shared by the gyroscope, the accelerometer and the camera, where x_S-axis points to sensor front and z_S-axis points to sensor top. The vision frame $\{V\}$ is the world frame assumed in the mSLAM algorithm, in which the projection of x_v-axis on the $x_E - y_E$ plane is parallel to x_E, and the orientation of z_V-axis is arbitrary depending on how the mSLAM is initialised. Note that there are two assumptions under this coordinate system: first, the origin of $\{E\}$ is assumed to be very close to the origin of $\{V\}$ (here, we separate the two frames in Figure 1 for the sake of clearance); second, the x_V-axis is assumed to not be perpendicular to the $x_E - y_E$ plane in $\{E\}$.

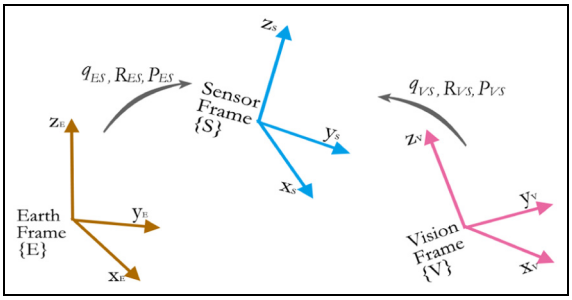

Figure 1. Coordinate system.

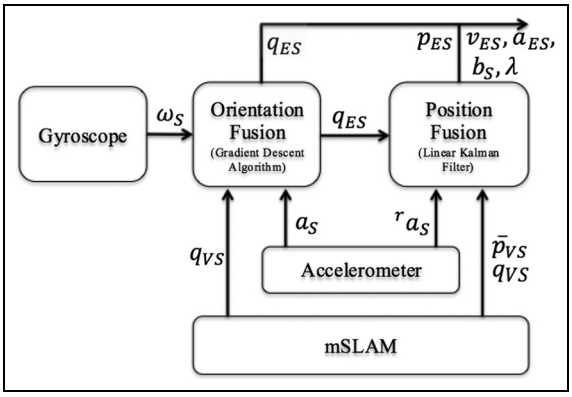

Figure 2. Algorithm overview.

The orientation of $\{S\}$ with respect to $\{E\}$ can be expressed as $q_{ES} \in \mathbb{R}^4$ in quaternion or $R_{ES} \in SO(3)$ in rotation matrix and the position as $p_{ES} \in \mathbb{R}^4$. Similarly, the orientation of $\{S\}$ with respect to $\{V\}$ can be expressed as $q_{VS} \in \mathbb{R}^4$ or $R_{VS} \in SO(3)$ and the position as $p_{VS} \in \mathbb{R}^4$.

Algorithm overview

As shown in Figure 2, the visual–inertial fusion algorithm assumes that rotation q_{VS}, as well as the unscaled position $\bar{p}_{VS}$ is provided by an mSLAM algorithm, which is treated as a black box. Moreover, it receives angular rates measurement $\omega_S \in \mathbb{R}^4$ from gyroscope and acceleration measurement $a_S \in \mathbb{R}^4$ from accelerometer. The output of the fusion process is to estimate rotation q_{ES} and position $p_{ES} \in \mathbb{R}^3$ of $\{S\}$ in $\{E\}$. Furthermore, the position filter also estimates the linear velocity $v_{ES} \in \mathbb{R}^3$, linear acceleration $a_{ES} \in \mathbb{R}^3$ and accelerometer bias $b_{ES} \in \mathbb{R}^3$, as well as the metric scale of the mSLAM position measurement λ.

The fusion is separated into two fusion processes: orientation fusion process and position fusion process. The orientation fusion is based on very efficient gradient descent algorithm,[21] and position fusion is based on a 13-state linear KF. The following two sections will

present the mathematical expressions of the two algorithms, respectively.

Orientation fusion process

The orientation fusion algorithm is based on the gradient descent algorithm in quaternion representation. The origin of the algorithm comes from[21] where the detailed mathematical derivation and proof are presented. However, different from the original algorithm, the following fusion derivation eliminates the magnetometer sensor, while, instead, the rotation correction about gravity vector is compensated by fusing the vision (mSLAM) measurement. Therefore, it avoids the problem of magnetic field distortion, thus only factory calibration is required once for accelerometer.

Moreover, given that all the mSLAM orientation measurements tend to drift over time and distance due to the accumulated error, fusing the vision measurement in the same way as the magnetometer will result in the estimation error on gravity direction. This can be very sensitive for the quadrotor stabilisation and control. Thus, the following algorithm decouples vision measurement with the gravity direction estimation, while maintaining the effective fusion about the gravity vector.

In order to perform orientation estimation, as shown in Figure 3, three coordination frames are used: sensor frame $\{S\}$ represents the orientation of all the coincide sensors (gyroscope, accelerometer and camera); earth frame $\{E\}$ represents the reference frame of the inertial sensors (gyroscope and accelerometer) and vision frame $\{V\}$ represents the reference frame of the mSLAM algorithm based on the camera. Additionally, g_E is the unit vector representing the true gravity direction, which is also called gravity field vector in the rest of the article, and $proj[x_V]$ is the unit projection vector of the x_V-axis onto the $x_E - y_E$ plane, which is also called vision field vector in the rest of the article.

The purpose of the orientation fusion is to estimate optimal quaternion transformation q_{ES} from $\{E\}$ to $\{S\}$ so that (1) the z_E-axis aligns with the gravity field and (2) the x_v-axis aligns with the vision field.

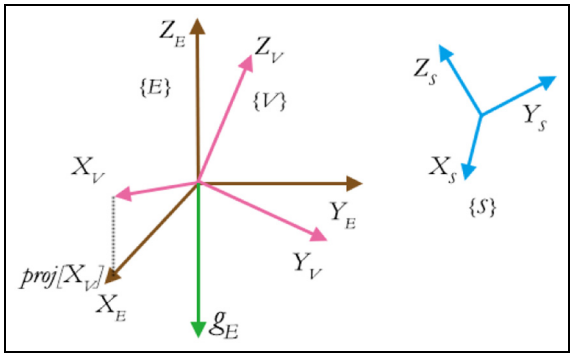

Figure 3. Gravity field and vision field.

The essential mathematical expression of one iteration at time t is shown as follows. Note that the orientation estimation from last iteration $q_{ES,t-1}$ is assumed to be known, and the sampling period is denoted as Δt.

Orientation derivative estimated by gyroscope

The three-axis gyroscope measures the angular velocity (rate) in *rads* about the three axes of $\{S\}$ frame, which we denote as $\omega_S = [0, \omega_x, \omega_y, \omega_z]^{\mathrm{T}}$. The quaternion derivative of the gyroscope estimation $\dot{q}_{\omega,t}$ at time t can be computed by equation (1), given $q_{ES,t-1}$ as the previous orientation estimation from all sensors

$$\dot{q}_{\omega,t} = \frac{1}{2} q_{ES,t-1} \otimes \omega_S \tag{1}$$

where $\otimes$ denotes a quaternion product. Note that all the quaternions in this article follow $q = [q_w, q_x, q_y, q_z]^{\mathrm{T}}$ and they are all unit quaternions $(q_w^2 + q_x^2 + q_y^2 + q_z^2 = 1)$.

Orientation optimisation from homogeneous field

In order to obtain the optimal orientation estimation $q_{ES} = [q_0, q_1, q_2, q_3]^{\mathrm{T}}$, the field measured from the sensor has to be aligned with the predefined reference field as close as possible. Thus, this can be formularised as an optimisation problem, where, for any homogeneous field $b_E \in \mathbb{R}^4$ in $\{E\}$, equation (2) is solved to minimise an objective error function

$$\min_{\forall q_{ES} \in \mathbb{R}^4} f(q_{ES}, b_E, s_S) \tag{2}$$

where $s_S \in \mathbb{R}^4$ is the field vector measured by the corresponding sensor in $\{S\}$. The gradient descent algorithm is one of the most computationally efficient optimisation algorithms to solve the above problem. Equation (3) describes it for n iterations, which starts from initial orientation estimation $q_{ES,0}$ to final estimation $q_{ES,n+1}$

$$q_{ES,k+1} = q_{ES,k} - \mu \frac{\Delta f(q_{ES,k}, b_E, s)}{\|\Delta f(q_{ES,k}, b_E, s)\|}, \quad k = 0, 1, 2, \ldots, n \tag{3}$$

where μ is the a non-negative scalar, named as step-size, and the error direction $\Delta f(q_{ES,k}, b_E, s)$ is computed by the objective error function f and its Jacobian matrix

$$\Delta f(q_{ES,k}, b_E, s) = J^{\mathrm{T}}(q_{ES,k}) f(q_{ES,k}, b_E, s) \tag{4}$$

Gravity field objective error function. Gravity field objective error function represents the error between gravity

vector in principle and the sensed gravity acceleration vector, expressed in $\{S\}$. Thus, given the gravity field vector g_E, the gravity field objective error function f_g is defined as

$$f_g(q_{ES}, g_E, a_S) = q_{ES}^{\star} \otimes g_E \otimes q_{ES} - a_S \tag{5}$$

where $q_{ES}^{\star}$ is the conjugate of q_{ES}, and $a_S = [0, a_x, a_y, a_z]^{\mathrm{T}}$ is the normalised accelerometer measurement. Since $g_E = [0, 0, 0, -1]^{\mathrm{T}}$, then it can be further simplified as equation (6)

$$f_g(q_{ES}, a_S) = \begin{bmatrix} -2(q_1 q_3 - q_0 q_2) - a_x \\ -2(q_0 q_1 + q_2 q_3) - a_y \\ -2\left(\frac{1}{2} - q_1^2 - q_2^2\right) - a_z \end{bmatrix} \tag{6}$$

Therefore, its Jacobian matrix is

$$J_g(q_{ES}) = \begin{bmatrix} 2q_2 & -2q_3 & 2q_0 & -2q_1 \\ -2q_1 & -2q_0 & -2q_3 & -2q_2 \\ 0 & 4q_1 & 4q_2 & 0 \end{bmatrix} \tag{7}$$

Vision field objective error function. Vision field objective error function represents the difference between the vision field vector $proj[x_V]$ and the x_E-axis of $\{E\}$ represented in $\{S\}$, thus the vision field objective error function f_v is defined as equation (8)

$$f_V(q_{ES}, x_E, x_{VS}) = q_{ES}^{\star} \otimes x_E \otimes q_{ES} - x_{VS} \tag{8}$$

where x_{VS} is $proj[x_V]$ represented in $\{S\}$ as shown in Figure 3. It is treated as a constant vector once been pre-computed by (equations (9)–(11))

$$x_V = q_{VS}^{\star} \otimes (q_{ES} \otimes x_E \otimes q_{ES}^{\star}) \otimes q_{VS} \tag{9}$$

$$proj[x_V] = \left[0, \frac{x_{V1}}{x_{V1}^2 + x_{V2}^2}, \frac{x_{V2}}{x_{V1}^2 + x_{V2}^2}, 0\right] \tag{10}$$

$$x_{VS} = q_{ES}^{\star} \otimes proj[x_V] \otimes q_{ES} \tag{11}$$

where $x_V = [0, x_{V1}, x_{V2}, x_{V3}]^{\mathrm{T}}$ is the unit x_V-axis vector in $\{E\}$, and $proj[x_V]$ is the normalised projection of x_V onto the $x_E - y_E$ plane measured by mSLAM algorithm, which can be computed by equation (10). Since $x_E = [0, 1, 0, 0]^{\mathrm{T}}$, it can then be simplified as

$$f_V(q_{ES}, x_{VS}) = \begin{bmatrix} 2\left(\frac{1}{2} - q_2^2 - q_3^2\right) - x_{VS1} \\ 2(q_1 q_2 - q_0 q_3) - x_{VS2} \\ 2(q_0 q_2 + q_1 q_3) - x_{VS3} \end{bmatrix} \tag{12}$$

where $x_{VS} = [0, x_{VS1}, x_{VS2}, x_{VS3}]$. Thus, its Jacobian matrix is

$$J_V(q_{ES}) = \begin{bmatrix} 0 & 0 & -4q_2 & -4q_3 \\ -2q_3 & 2q_2 & 2q_1 & -2q_0 \\ 2q_2 & 2q_3 & 2q_0 & 2q_1 \end{bmatrix} \tag{13}$$

Fusion of three sensors

As stated in the gradient descent algorithm,[21] given that the convergence rate of the estimated orientation is equal or greater than the angular rate of the physical orientation, only one iteration is required to be computed per sample time, Δt. Therefore, an unconventional gradient descent algorithm is derived to fuse all the three sensor measurements. To compute the orientation in next time stamp $q_{ES,t+1}$, the process is summarised as

$$q_{ES,t+1} = q_{ES,t} + \dot{q}_{ES,t+1} \Delta t \tag{14}$$

$$\dot{q}_{ES,t+1} = \dot{q}_{\omega,t+1} - \beta \frac{\Delta f}{\|\Delta f\|} \tag{15}$$

where $\Delta f / \|\Delta f\|$ can be assigned a physical meaning as the normalised direction of the error of $\dot{q}_{ES,t+1}$, and it can be expressed as $\dot{q}_{\varepsilon,t+1}$

$$\dot{q}_{\varepsilon,t+1} = \frac{\Delta f}{\|\Delta f\|} \tag{16}$$

Moreover, β is the only adjustable parameter of this filter. It represents the magnitude of the gyroscope measurement error, which is removed in the direction according to the accelerometer and vision sensor. The higher the β, the faster that the estimated orientation converge to the accelerometer estimation. The theoretically optimal value of β is

$$\beta = \sqrt{\frac{3}{4}} \tilde{\omega}_{\max} \tag{17}$$

where $\tilde{\omega}_{\max}$ is the maximum gyroscope measurement error for each axis. Moreover, since IMU and the vision sensor operate in different speed, depending on which sensor measurement is available, Δf can then be expressed as

$$\Delta f = \begin{cases} J_g^{\mathrm{T}}(q_{ES}) f_g(q_{ES}, a_S) \\ J_{g_V}^{\mathrm{T}}(q_{ES}) f_{g_V}(q_{ES}, a_S, x_{VS}) \end{cases} \tag{18}$$

where $f_{g_V}(q_{ES}, a_S, x_{VS})$ and $J_{g_V}(q_{ES})$ are the combined measurements of both field from the sensors, which can be expressed as

$$f_{g_V}(q_{ES}, a_S, x_{VS}) = \begin{bmatrix} f_g(q_{ES}, a_S) \\ f_V(q_{ES}, x_{VS}) \end{bmatrix} \tag{19}$$

$$J_{g_V}(q_{ES}) = \begin{bmatrix} J_g(q_{ES}) \\ J_V(q_{ES}) \end{bmatrix} \tag{20}$$

Gyroscope bias online estimation

Given the fact that the gyroscope bias drifts with temperature and motion, in practice, any high-accuracy fusion algorithm must be able to estimate the varying gyroscope bias online. Kalman-based methods generally cope with this by introducing the bias variables into the state vector. However, a much more computationally efficient estimation method is used by DC component of the angular error feedback, similar with Madgwick.[22]

The normalised direction of the error in the rate of change of orientation, $\dot{q}_\varepsilon$, which is defined by equation (16), can be converted to the angular error ω_ε in $\{S\}$ frame by inverting equation (1). This yields

$$\omega_{\varepsilon,t} = 2q^{\star}_{ES,t-1} \otimes \dot{q}_{\varepsilon,t} \tag{21}$$

The gyroscope bias, ω_b, can then be represented as the DC component of ω_ε and thus can be removed from the gyroscope measurement, ω_S, as the integral of ω_ε, weighted by a gain, ζ

$$\omega_{b,t} = \zeta \int \omega_{\varepsilon,t} dt \tag{22}$$

$$\omega_{c,t} = \omega_{s,t} - \omega_{b,t} \tag{23}$$

where ω_c is the corrected gyroscope reading, thus it can be used to replace the raw gyroscope reading, ω_s, in equation (1).

Here, the weighting factor, ζ, decides the convergence rate of the gyroscope bias estimation, where the higher the ζ, the faster and noisier the convergence. While the theoretical optimal value of ζ is defined as

$$\zeta = \sqrt{\frac{3}{4}}\dot{\tilde{\omega}}_{\max} \tag{24}$$

where $\dot{\tilde{\omega}}_{\max}$ is the maximum rate of change of the gyroscope measurement error of each axis.

Position fusion process

This position fusion algorithm assumes that the orientation of the sensors is known and only estimates the translational state of the system. Being able to avoid estimating the orientation not only reduces the length of the state vector significantly but also keeps the system model almost linear. It takes three inputs: (1) the orientation estimation q_{ES} in $\{E\}$ from the result of the orientation fusion process; (2) the raw sensor acceleration measurement ${}^{r}a_S \in \mathbb{R}^4$ (m/s^2) from accelerometer and (3) the unscaled position $\bar{p}_{VS} \in \mathbb{R}^4$ and orientation $q_{VS} \in \mathbb{R}^4$ in $\{V\}$ from the mSLAM. It outputs its state vector, which contains position estimation $p_{ES} \in \mathbb{R}^3$, velocity estimation $v_{ES} \in \mathbb{R}^3$ and acceleration estimation $a_{ES} \in \mathbb{R}^3$, in $\{E\}$, and accelerometer bias $b_S \in \mathbb{R}^3$, as well as the metric scale $\lambda > 0$ of the mSLAM position estimation, which is defined as $\bar{p}_{VS} = \lambda p_{VS}$. The position fusion algorithm is formed of a coordinate frame management process and a 13-state linear KF. The KF conducts in the earth frame $\{E\}$, thus all the sensor measurement values have to be converted to $\{E\}$ in the coordinate frame management process.

Coordinate frame management process

Dynamic acceleration in earth frame. Different from the orientation fusion process, in the position fusion process, we consider that the accelerometer not only measures the gravity but also measures the pure dynamic acceleration caused by the movement of the body frame. And since the orientation estimation q_{ES} obtained from orientation fusion process recovers the gravity vector $g_E = [0, 0, 0, -1]$ in $\{E\}$, therefore the dynamic acceleration ${}^{s}a_{ES} \in \mathbb{R}^3$ in $\{E\}$ can be easily obtained by

$$\begin{bmatrix} 0 \\ {}^{s}a_{ES} \end{bmatrix} = \|{}^{r}a_S\| \left(q_{ES} \otimes a_S \otimes q^{\star}_{ES} - g_E \right) \tag{25}$$

recalling the normalised accelerometer measurement $a_S = {}^{r}a_S / \|{}^{r}a_S\|$.

Unscaled vision position in earth frame. The unscaled position estimation from mSLAM algorithm ${}^{s}\bar{p}_{ES} \in \mathbb{R}^3$ in $\{E\}$ can be obtained by

$$\begin{bmatrix} 0 \\ {}^{s}\bar{p}_{ES} \end{bmatrix} = q_{ES} \otimes (q^{\star}_{VS} \otimes \bar{p}_{VS} \otimes q_{VS}) \otimes q^{\star}_{ES} \tag{26}$$

Therefore, the resulting measurements in $\{E\}$ frame (${}^{s}a_{ES}$ and ${}^{s}\bar{p}_{ES}$) can be passed to the position KF as two individual sensor measurements, which will be described as follows.

Linear Kalman filter

The conventional KF framework consists of a prediction step, which performs the state vector time update in constant time interval; and a measurement update step, which performs the correction of the state vector based on the new sensor measurement. Here, in order to encounter the asynchronous measurements from both accelerometer and mSLAM algorithm, two different measurement update models are constructed and will be executed depending on which sensor measurement is available.

State representation and prediction model. The state of the KF is represented as a state vector $x \in \mathbb{R}^{13}$

$$x = \left[p^T_{ES}, v^T_{ES}, a^T_{ES}, b^T_S, \lambda \right]^{\mathrm{T}} \tag{27}$$

where position estimation p_{ES}, velocity estimation v_{ES} and acceleration estimation a_{ES} are in $\{E\}$, and accelerometer bias b_S is in $\{S\}$, as well as the metric scale $\lambda>0$ of the mSLAM position estimation, which is defined as $\bar{p}_{VS} = \lambda p_{VS}$.

The state vector is updated once every time interval, following the rule defined by the prediction model, which defines the physics of the inertial system. It is summarised as

$$\dot{p}_{ES} = v_{ES} \tag{28}$$

$$\dot{v}_{ES} = a_{ES} \tag{29}$$

$$\dot{a}_{ES} = n_a, \quad \dot{b}_S = n_b, \quad \dot{\lambda} = n_\lambda \tag{30}$$

The linear acceleration a_{ES}, accelerometer bias b_S and mSLAM metric scale factor λ are modelled as Gaussian random walk. Thus, $n_a \in \mathbb{R}^3$, $n_b \in \mathbb{R}^3$ and n_λ are independent zero-mean normal distribution Gaussian process noise

$$p(n_a) \sim N(0, Q_a) \tag{31}$$

$$p(n_b) \sim N(0, Q_b) \tag{32}$$

$$p(n_\lambda) \sim N(0, \sigma_\lambda^2) \tag{33}$$

where $Q_a = \sigma_a^2 I_3$ and $Q_b = \sigma_b^2 I_3$ are the process noise covariance of acceleration and accelerometer bias, respectively. σ_a, σ_b and σ_λ are the standard deviations of n_a, n_b and n_λ, respectively, and I_3 is three-by-three identity matrix.

Following Welch and Bishop,[23] the discrete KF time update includes two steps:

1. State vector propagation to predict the state vector in next time step

$$\hat{x}_k = A x_{k-1} \tag{34}$$

where A is the state transition matrix, which is the Jacobian matrix of partial derivatives of the prediction model with respect to x.

2. State covariance matrix $P \in \mathbb{R}^{13\times 13}$ propagation to predict the state noise in next time step

$$\hat{P}_k = A P_{k-1} A^{\mathrm{T}} + W Q W^{\mathrm{T}} \tag{35}$$

where W is the Jacobian matrix of partial derivatives of the prediction model with respect to the noise vector, P is the state covariance matrix and $Q = diag(0_{6\times 6}, Q_a, Q_b, \sigma_\lambda^2)$ is the process noise covariance matrix.

Measurement model. The measurement model is derived in the form of

$$z_\star = H_\star x + e_\star \tag{36}$$

where $z_\star$ is the measurement from the mSLAM vision sensor or the IMU, $H_\star$ is the measurement model matrix and $e_\star$ denotes the measurement error from the sensor, where $\star$ can be *as* or *vs* depending on which sensor measurement is available between acceleration sensor measurement and vision sensor measurement. Here, $e_\star$ is also modelled as independent zero-mean normal distribution Gaussian process noise

$$p(e_\star) \sim N(0, R_\star) \tag{37}$$

where $R_\star$ is the measurement noise covariance of $e_\star$.

When the accelerometer measurement ${}^s a_{ES}$ is available, the KF performs accelerometer measurement update, to adjust state vector and state covariance matrix according to the accelerometer measurement model. Here, the accelerometer measurement model is defined by matrix $H_{as} \in \mathbb{R}^{3\times 13}$

$$H_{as} = [0_{3\times 6} \quad I_3 \quad R_{ES} \quad 0_{3\times 1}] \tag{38}$$

where $R_{ES} \in SO(3)$ is the corresponding rotation matrix to q_{ES} and the accelerometer measurement noise covariance $R_{as} = \sigma_{as}^2 I_3$, where σ_{as} is the standard deviations of e_{as}.

When the unscaled position measurement ${}^s \bar{p}_{ES}$ is available from mSLAM algorithm, the KF performs vision measurement update to adjust state vector and state covariance matrix according to the vision measurement model. Here, the vision measurement model is defined by matrix $H_{vs} \in \mathbb{R}^{3\times 13}$

$$H_{vs} = [\lambda I_3 \quad 0_{3\times 9} \quad p_{ES}] \tag{39}$$

And the vision measurement noise covariance $R_{vs} = \sigma_{vs}^2 I_3$, where σ_{vs} is the standard deviations of e_{vs}.

Following Welch and Bishop,[23] the measurement update steps are summarised as follows:

1. Compute Kalman gain $K_k \in \mathbb{R}^{13\times 6}$

$$K_k = \hat{P}_k H_\star^{\mathrm{T}} (H_\star \hat{P}_k H_\star^{\mathrm{T}} + R_\star)^{-1} \tag{40}$$

2. State vector update

$$x_k = \hat{x}_k + K_t (z_k - H_\star \hat{x}_k) \tag{41}$$

3. State covariance update

$$P_k = (I_{16} - K_k H_\star) \hat{P}_k \tag{42}$$

Measurement update process handles different sampling rate between mSLAM and IMU estimation, by only updating state with the corresponding

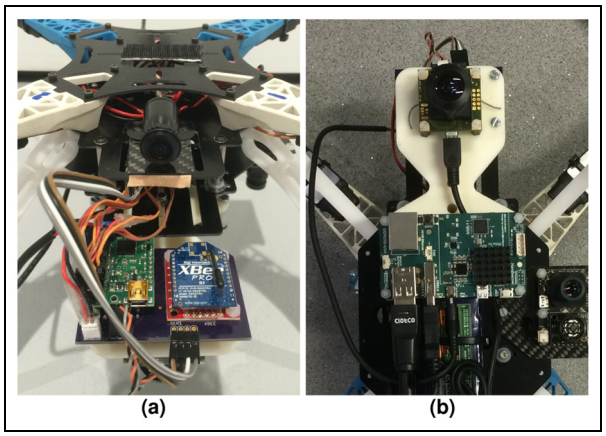

Figure 4. Hardware set-up: (a) freeIMU sensor and Teensy 3.1 processor and (b) uEye camera and Odroid-U3 computer.

measurement, which becomes available. Thus, by assuming that the orientation fusion reaches steady state, the state vector x can be effectively estimated over time.

Implementation

This section describes the details on the algorithm implementation on an embedded platform. FreeIMU v0.4.3 (http://www.varesano.net/projects/hardware/FreeIMU) hardware was used as the IMU, which includes an MPU6050 gyroscope–accelerometer combo chip, an HMC5883 magnetometer and MS5611 high-resolution pressure sensor. However, only the MPU6050 was used in the algorithm. We performed orientation estimation in Teensy 3.1 (https://www.pjrc.com/teensy), which features an ARM Cortex-M4 processor with 96 MHz clock cycle. Besides, the real-time video frame was captured by an onboard uEye global shutter monocular camera in maximum 80 frames/s. Then, both the video frame and the orientation estimation are processed by the semi-direct monocular visual odometry (SVO) mSLAM framework with the extended Kalman filter (EKF) position fusion algorithm operating in parallel on Odroid-U3 (http://www.hardkernel.com/) single board-embedded Linux computer, which features an 1.7 GHz Quad-Core processor with 2 GB RAM. The communication between software packages is realised by Robot Operating System (ROS) (http://www.ros.org).

The entire system was installed on a quadrotor platform,[24] as shown in Figure 4. The autopilot board including the FreeIMU, Teensy processor, servo controller and XBee radio is shown in Figure 4(a). Since the components are overlaid to minimise the size, Figure 5 shows the top and bottom layers of the disassembled autopilot board, indicating the position of the FreeIMU and Teensy processor. The uEye camera and Odroid-U3 computer is installed underneath the quadrotor as shown in Figure 4(b).

Figure 5. FreeIMU and Teensy open up view.

Table 1. Parameter set up.

Parameter	Value
β	0.5
σ_a	0.5
σ_b	$1\times e^{-6}$
σ_λ	$1\times e^{-6}$
σ_{as}	0.013
σ_{vs}	0.005

The Teensy processor is capable of executing the orientation fusion alongside with autopilot control algorithm at 300 Hz while communicating with Odroid-U3 computer with ROS protocol, including publishing orientation estimation and acceleration measurement at 100 Hz and subscribing the pose estimation from SVO mSLAM framework in Odroid-U3. Moreover, in the Odroid-U3 computer, the SVO mSLAM is executed at 40 FPS with the KF position fusion algorithm running at 100 Hz in parallel.

We measured the simultaneous localisation and mapping (SLAM) processing delay and set the message buffer manually without hard synchronisation. The parameters set-up for both orientation and KF position fusion algorithm is summarised in Table 1. β was left as default value, 0.5, by assuming $\tilde{\omega}_{\max}$ was approximately

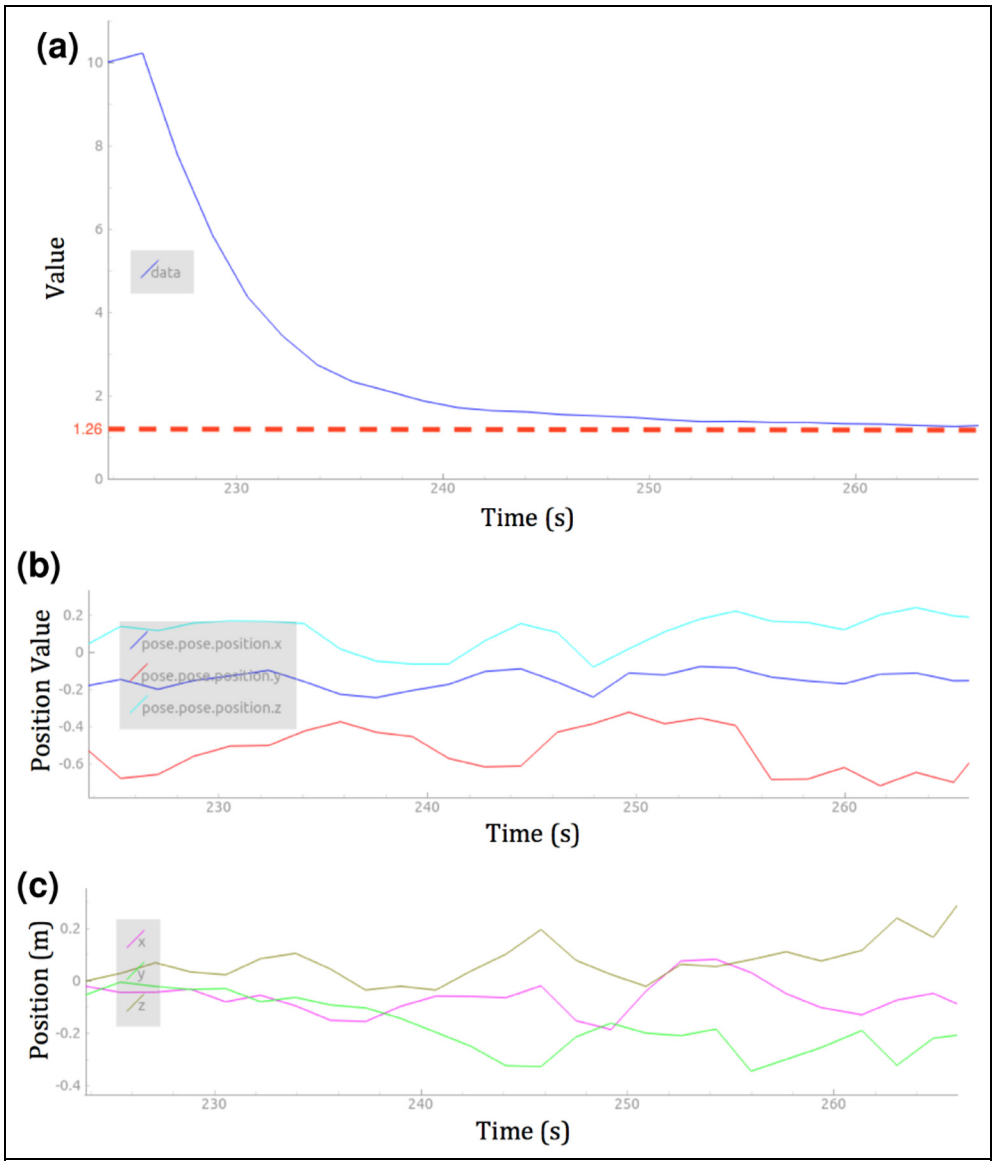

Figure 6. KF position fusion result: (a) scale factor λ, (b) the unscaled position measurement from SVO SLAM and (c) the scaled position measurement from KF position estimator.

0.58 rad/s. The value of σ_{as} was selected according to the accelerometer standard noise from the data sheet of the MPU6050 and σ_{vs} was set to be 0.005 in the map scale. σ_a, σ_b and σ_λ were manually tuned through experiments. Here, we assumed accelerometer bias and visual scale change very slowly, thus σ_b and σ_λ were set very small. σ_a determines the confidence level for the prediction model, which means the higher was the σ_a, the less confidence is the estimator about its prediction model, and thus the easier the sensor measurements affect the estimation.

Test results

Two test trials were performed to shown the effectiveness of the algorithm.

The first trial focuses on evaluating the scale factor (λ) estimation from an arbitrary initial value. The trial was conducted in real time under general indoor condition and handholding the quadrotor with gentle moving within $0.3 \times 0.3 \times 0.3\ \mathrm{m}^3$ space. Note that the position fusion assumes that the orientation fusion reaches the steady state before initialisation. The KF position fusion algorithm is initialised with the state vector $x_0 = [0_{1\times 12}, 10]^{\mathrm{T}}$; note that we initialise the scale factor λ to 10 as a arbitrary positive value to show how it converges to the true value. As shown in Figure 6, the initialisation occurs at 227 s and the record shows a 39-s trial, indicating how the true scale factor is recovered, and how the output position estimation relates to the raw input position measurement over the same period of time. It is clear that the scale factor λ is

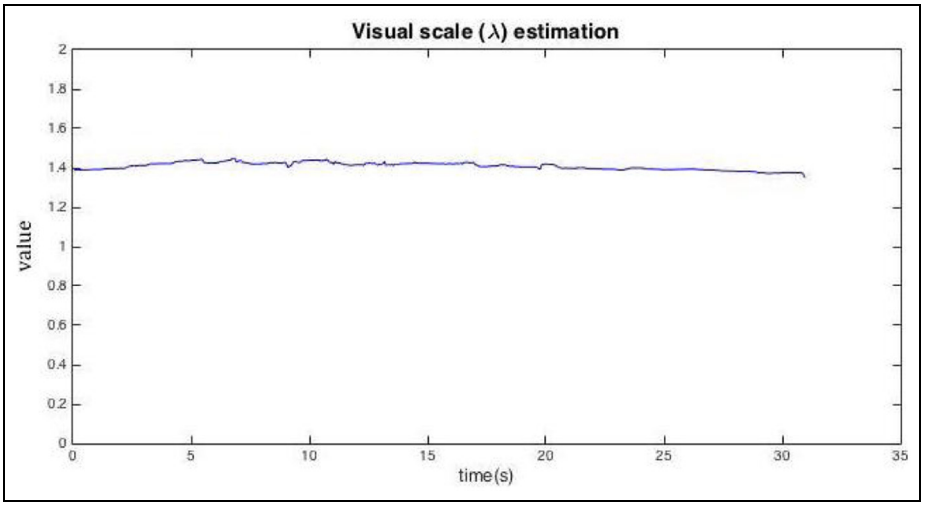

Figure 7. Scale factor estimation.

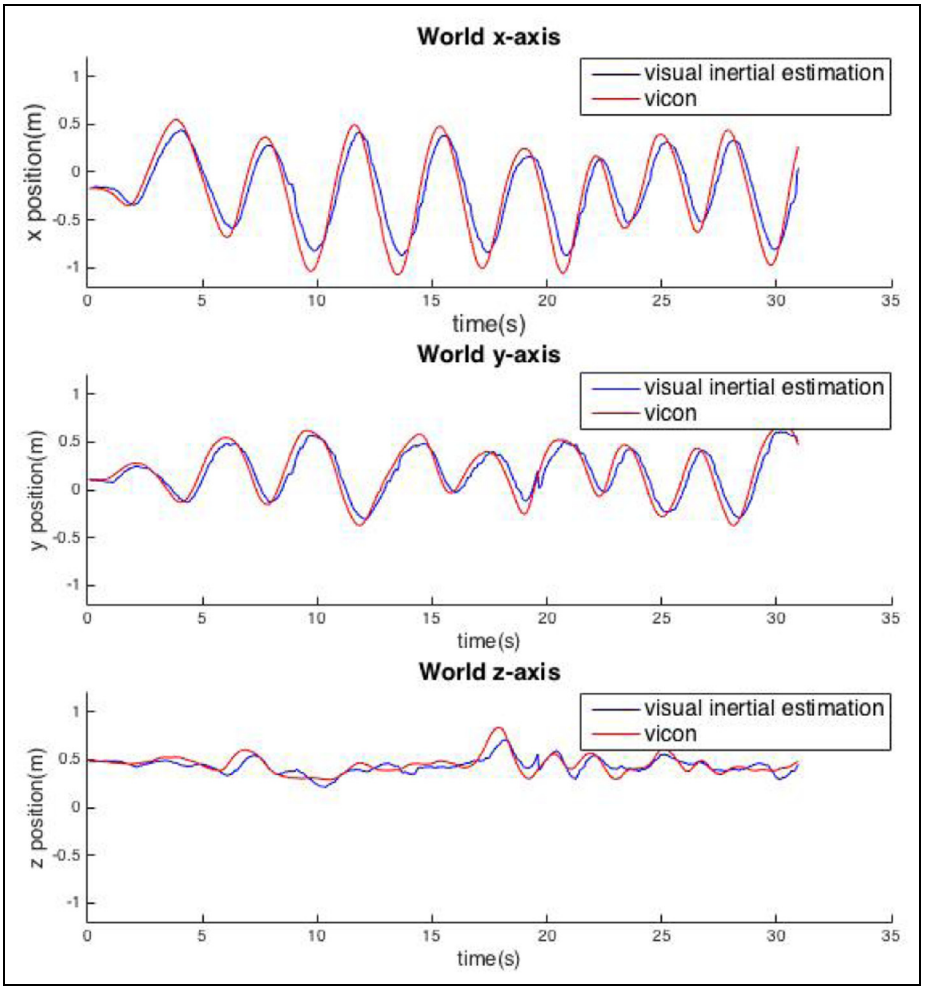

Figure 8. Position estimation ground truth comparison.

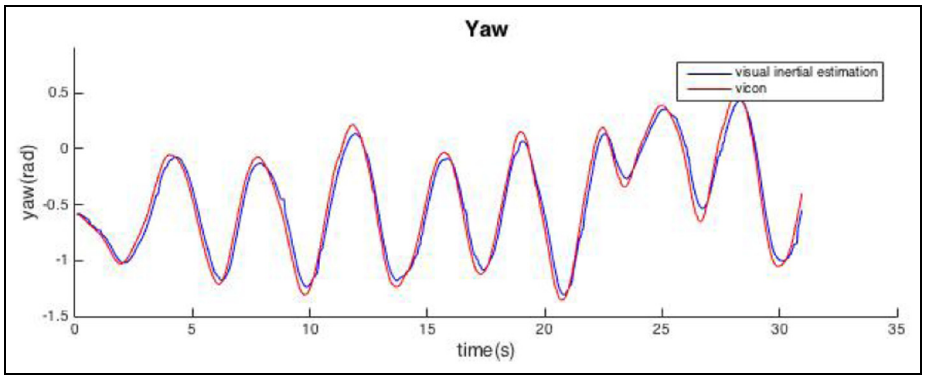

Figure 9. Yaw angle estimation ground truth comparison.

effectively discovered as 1.26, despite that its initial value is set to 10, as shown in Figure 6(a). And during the converging period, the position estimation output from KF position estimator is scaled accordingly with λ over time. Since KF position estimator fuses the acceleration with SVO position measurement, the output position estimation performs better in dynamic operation.

The second trial evaluates the position and yaw angle estimation performance by comparing the estimation against the ground truth. The 6-DOF ground truth was provided by VICON (https://www.vicon.com) motion capturing system. Again, the movement was generated by handheld motion in about $2 \times 2 \times 2$ m^3 space for 31-s duration. The estimated scale factor of the SVO is shown in Figure 7. The position estimation for each axis is shown in Figure 8, and the yaw estimation is shown in Figure 9. It is shown that the system has a good orientation and position tracking, with 0.1 radians orientation error and 0.1 m position error. However, the position error is believed to be mainly introduced by the small error in visual scale estimation, which could be the result of small timing error from manual synchronisation or slow converging rate close to the true value. Moreover, the latency shown in the two graphs are from the wireless communication from the two source systems. The three-dimensional (3D) position can be illustrated as shown in Figure 10.

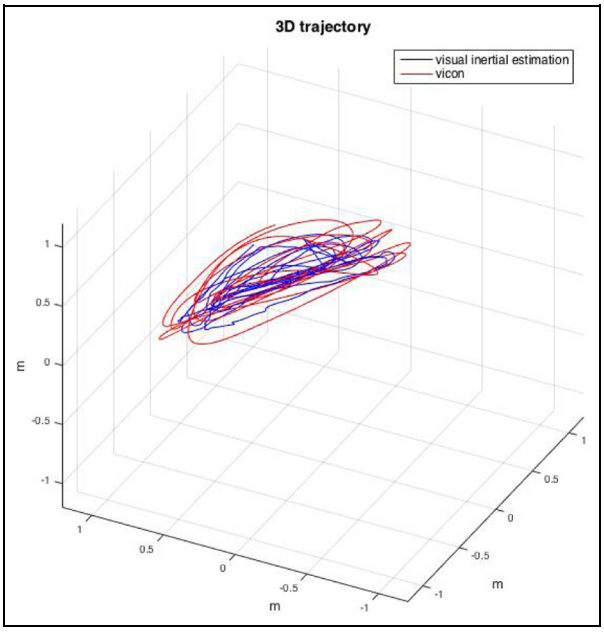

Figure 10. Three-dimensional trajectory illustration against ground truth comparison.

Conclusion and future work

This article has shown the design and implementation work of a sensor fusion framework, which is capable of performing the 6-DOF sensor state estimation, by the fusion of a three-axis gyroscope, a three-axis accelerometer and an mSLAM algorithm.

Two test trials were carried out to show the effectiveness of the system. The first trial shows that scale factor of the mSLAM can be sufficiently estimated from an arbitrary value, thus the position output is scaled accordingly. Whereas in the second trial, the comparison against ground truth shows that the 3D position can be effectively estimated, and the drift-free yaw rotation can be estimated accurately without magnetometer.

The future work includes estimation of error comparison and computational performance evaluation by benchmarking with other existing fusion algorithms.

Declaration of conflicting interests

The author(s) declared no potential conflicts of interest with respect to the research, authorship, and/or publication of this article.

Funding

The author(s) received no financial support for the research, authorship, and/or publication of this article.

References

1. Weiss S and Siegwart R. Real-time metric state estimation for modular vision-inertial systems. In: *Proceedings of the IEEE international conference on robotics and automation*, Shanghai, China, 9–13 May 2011, vol. 231855, pp.4531–4537. New York: IEEE.
2. Blösch M, Weiss S, Scaramuzza D, et al. Vision based MAV navigation in unknown and unstructured environments. In: *Proceedings of the IEEE international conference on robotics and automation*, Anchorage, AK, 3–7 May 2010, pp.21–28. New York: IEEE.
3. Klein G and Murray D. Parallel tracking and mapping for small AR workspaces. In: *6th IEEE and ACM international symposium on mixed and augmented reality, ISMAR*, Nara, Japan, 13–16 November 2007, pp.1–10. New York: IEEE.
4. Engel J, Schöps T and Cremers D. LSD-SLAM: large-scale direct monocular SLAM. In: *Eccv*, 2014, pp.834–849, http://vision.in.tum.de/research/vslam/lsdslam
5. Forster C, Pizzoli M and Scaramuzza D. SVO: fast semi-direct monocular visual odometry. In: *IEEE international conference on robotics and automation (ICRA)*, Hong Kong, 31 May–7 June 2014. New York: IEEE.
6. Pizzoli M, Forster C and Scaramuzza D. REMODE: probabilistic, monocular dense reconstruction in real time. In: *IEEE international conference on robotics and automation (ICRA)*, Hong Kong, 31 May–7 June 2014. New York: IEEE.
7. Vogiatzis G and Hernández C. Video-based, real-time multi-view stereo. *Image Vision Comput* 2011; 29: 434–441.
8. Newcombe RA, Lovegrove SJ and Davison AJ. DTAM: dense tracking and mapping in real-time. In: *Proceedings of the IEEE international conference on computer vision*, Barcelona, 6–13 November 2011, pp.2320–2327. New York: IEEE.
9. Roussillon C, Gonzalez A, Solà J, et al. RT-SLAM: a generic and real-time visual SLAM implementation. In: *Computer vision systems* (Lecture notes in computer science), vol. LNCS 6962. Berlin: Springer, 2011, pp.31–40.
10. Montemerlo M, Thrun S, Roller D, et al. FastSLAM 2.0: an improved particle filtering algorithm for simultaneous localization and mapping that provably converges. In: *IJCAI international joint conference on artificial intelligence*, Beijing, China, 9–15 October 2003, vol. 18, pp.1151–1156. New York: IEEE.
11. Engel J, Sturm J and Cremers D. Semi-dense visual odometry for a monocular camera. In: *Proceedings of the IEEE international conference on computer vision*, Sydney, NSW, Australia, 1–8 December 2013, pp.1449–1456. New York: IEEE.
12. Kelly J and Sukhatme GS. Visual-inertial simultaneous localization, mapping and sensor-to-sensor self-calibration. In: *Proceedings of IEEE international symposium on computational intelligence in robotics and automation, CIRA*, Daejeon, South Korea, 15–18 December 2009, pp.360–368. New York: IEEE.
13. Kelly J and Sukhatme GS. Visual-inertial sensor fusion: localization, mapping and sensor-to-sensor self-calibration. *Int J Robot Res* 2010; 30: 56–79.
14. Lynen S, Achtelik MW, Weiss S, et al. A robust and modular multi-sensor fusion approach applied to MAV navigation. In: *IEEE international conference on intelligent robots and systems*, Tokyo, Japan, 3–7 November 2013, pp.3923–3929. New York: IEEE.
15. Lobo J and Dias J. Vision and inertial sensor cooperation using gravity as a vertical reference. *IEEE T Pattern Anal* 2003; 25: 1597–1608.
16. Dunkley O, Engel J, Sturm J, et al. Visual-inertial navigation for a camera-equipped 25 g nano-quadrotor. In: *IROS2014 aerial open source robotics workshop*, Chicago, IL, 14 September 2014, no. 2.
17. Li M and Mourikis AI. High-precision, consistent EKF-based visual-inertial odometry. *Int J Robot Research* 2013; 32: 690–711.
18. Jones ES and Soatto S. Visual-inertial navigation, mapping and localization: a scalable real-time causal approach. *Int J Robot Research* 2011; 30: 1–38.
19. Weiss S, Achtelik MW, Chli M, et al. Versatile distributed pose estimation and sensor self-calibration for an autonomous MAV. In: *Proceedings of the IEEE international conference on robotics and automation*, Saint Paul, MN, 14–18 May 2012, pp.31–38. New York: IEEE.
20. Shen S, Mulgaonkar Y, Michael N, et al. Vision-based state estimation for autonomous rotorcraft MAVs in complex environments. In: *Proceedings of the IEEE international conference on robotics and automation*, Karlsruhe, 6–10 May 2013, pp.1758–1764. New York: IEEE.
21. Madgwick SOH, Harrison AJL and Vaidyanathan R. Estimation of IMU and MARG orientation using a gradient descent algorithm. *IEEE Int Conf Rehabil Robot* 2011; 2011: 5975346.
22. Madgwick S. An efficient orientation filter for inertial and inertial/magnetic sensor arrays. Report x-io and University of Bristol, 2010, no. 32, https://www.samba.org/tridge/UAV/madgwick_internal_report.pdf
23. Welch G and Bishop G. An introduction to the Kalman filter. *In Practice* 2006; 7: 1–16.
24. Liu C and Prior SD. Design and implementation of a mini quadrotor control system in GPS denied environments. In: *International conference on unmanned aircraft systems, ICUAS'15*, Denver, CO, 9–12 June 2015, pp.462–469. New York: IEEE.