Abstract—In this paper, we investigate the problem of estimating the instantaneous position, orientation, velocity, and angular velocity of a rigid body during its free flight. Precisely speaking, the flight is under the influences of gravity, air drag, and the Magnus force only. Two cameras are synchronized to take images of the flying object. Feature points are extracted from each frame as observables from a governing system which integrates quaternion-based dynamics with a camera model that deals with radial distortion. An extended Kalman filter (EKF) is employed to carry out the online estimation. Experimental results have demonstrated a close match between the two image sequences cast by the object’s real motion and the estimated one.

I. INTRODUCTION

Real-time tracking of the orientation and position of a rigid body moving in the space is an important and challenging problem with applications in space exploration, air traffic control, missile guidance, underwater vehicles, robotics, virtual reality, etc. Techniques for estimating the velocity and angular velocity have relied on either accelerometers or cameras mounted inside the body, or cameras stationed outside.

Accelerometer-based estimation measured the accelerations at the mounting points of several linear accelerometers along orthogonal axes, and then solved a system of kinematic equations for the body’s angular acceleration [12], or determined it through optimization [3], [4]. In [10], a magnetometer, accelerometer, and angular rate sensor were employed to obtain measurements from which a quaternion was calculated and supplied to a Kalman filter for polishing. However, none of the above methods addressed how to estimate the object’s linear velocity.

The work [15] showed that the rotation motion of an asteroid about a fixed axis could be recovered on a spaceship from images taken of outside landmarks by combining particle filtering with optimization. The constant angular velocity, if constant, could be estimated based on line correspondences in the images of the object by combining a least-squares method with interpolation [9].

Vision-based estimation often integrated dual quaternions into an extended Kalman filter (EKF), taking linear features from images generated by a single camera [6], or by a network of cameras followed by optimization to achieve consensus [18]. Since rigid body dynamics were not fully integrated into the system, simulation and experiments were conducted to estimate only constant velocities and angular velocities.

This paper will investigate estimation of the full state of a free-flying rigid body, which includes its position, rotation, velocity, and angular velocity. We will consider the gravitational, drag, and Magnus forces only. Two cameras will be used to take images from which feature points are extracted to feed an EKF, which will continuously reduce the errors of its estimate.

Throughout the paper, \( I_k \) denotes the \( k \times k \) identity matrix. We also let \( \mathbf{i} = (1, 0, 0)^T \), \( \mathbf{j} = (0, 1, 0) \), and \( \mathbf{k} = (0, 0, 1) \) be the unit vectors along the \( x\)-, \( y\)-, \( z\)-axis of any frame.

II. SYSTEM DYNAMICS

Consider a flying object as shown in Figure 1. The object, with known geometry and physical properties, has a body frame \( F_b \) located at its center of mass \( o_b \), and defined by its principal axes. Under the frame, the angular inertia matrix \( Q \) is diagonalized. The frame \( F_b \) has a translation \( \mathbf{\delta} \) from the world frame \( F_w \). The rotation of \( F_b \) from \( F_w \) is described by a unit quaternion [8]:

\[
\mathbf{r} = \begin{pmatrix}
\mathbf{r}_0 \\
\mathbf{r}
\end{pmatrix} = \begin{pmatrix}
r_0 \\
r_1 \\
r_2 \\
r_3
\end{pmatrix},
\]
which is a 4-vector with some special multiplication rule.

Let $\mathbf{v}$ be the object’s velocity in $\mathcal{F}_w$, and $\mathbf{\omega}$ be its angular velocity in terms of $\mathcal{F}_b$, that is, relative to a fixed frame instantaneously coinciding with $\mathcal{F}_b$. The angular velocity is $\mathbf{r}_r$ when expressed in $\mathcal{F}_w$.

In addition to the gravitational force, the object is subject to two other forces. The first one is air drag acting opposite to its velocity:

$$f_d = -\frac{1}{2} \rho A C_d \parallel \mathbf{v} \parallel \mathbf{v},$$

where $\rho$ is the air density, $A$ the area of the object’s cross section normal to $\mathbf{v}$, and $C_d$ the drag coefficient. The second one is the Magnus force [5, pp. 16-33] due to uneven air pressure created by the air flow passing the object’s top and bottom parts as it rotates in the air. This force, traverse to the air flow, is given as

$$f_m = \frac{1}{2} \rho A C_m (r \mathbf{\omega}^*) \times \mathbf{v},$$

where $C_m$ is the lift coefficient. Based on the works [1], [16], [13], [11], we approximate the two forces as

$$f_d = -e_d \parallel \mathbf{v} \parallel \mathbf{v} \quad \text{and} \quad f_m = e_m (r \mathbf{\omega}^*) \times \mathbf{v},$$

where $e_d$ and $e_m$ are two constant coefficients, after assigning to $A$ the average cross section area during the flight.

Newton’s equation then takes the form:

$$\dot{\mathbf{v}} = \mathbf{g} - e_d \parallel \mathbf{v} \parallel \mathbf{v} + e_m \mathbf{\omega} \times \mathbf{v}, \quad (1)$$

where $\mathbf{g}$ is the gravitational acceleration vector. It is reasonable to assume that the forces $f_d$ and $f_m$ act through the object’s center of mass, just like the gravitational force. Then Euler’s equation assumes the form

$$Q \dot{\mathbf{\omega}} + \mathbf{\omega} \times Q \mathbf{\omega} = 0,$$

from which we immediately obtain

$$\dot{\mathbf{\omega}} = -Q^{-1}(\mathbf{\omega} \times Q \mathbf{\omega}). \quad (2)$$

The 13-vector $s = (\mathbf{\delta}, \mathbf{r}, \mathbf{v}^T, \mathbf{\omega}^T)^T$ describes the state of the flying object. The quaternion $\mathbf{r}$ describing the object’s orientation has the following derivative given in Appendix C of [8]:

$$\dot{\mathbf{r}} = \frac{1}{2} \mathbf{r} \mathbf{\omega}. \quad (3)$$

This, together with $\dot{\mathbf{\delta}} = \mathbf{v}$, (1), (2), forms a system of nonlinear differential equations below:

$$\frac{ds}{dt} = \frac{d}{dt} \begin{pmatrix} \mathbf{\delta} \\ \mathbf{r} \\ \mathbf{v} \\ \mathbf{\omega} \end{pmatrix} = \mathbf{a}(s), \quad (4)$$

where

$$\mathbf{a}(s) = \begin{pmatrix} \mathbf{v} \\ \frac{1}{2} \mathbf{r} \mathbf{\omega} \\ \mathbf{g} - e_d \parallel \mathbf{v} \parallel \mathbf{v} + e_m \mathbf{\omega} \times \mathbf{v} \\ -Q^{-1}(\mathbf{\omega} \times Q \mathbf{\omega}) \end{pmatrix}. \quad (5)$$

III. MEASUREMENT MODEL

The object’s state is estimated based on measurements extracted from its images taken simultaneously by two cameras. Here, we focus on one camera to understand the measurement model. As shown in Fig. 2, the camera’s focal point is located at $c$, where a frame $\mathcal{F}_c$ is set up to have the $z$-axis perpendicular to the image plane. The image plane gets flipped from its real location inside the camera to appear in front of the focal point by a distance equal to the focal length $f$. The rotation of $\mathcal{F}_c$ from the world frame $\mathcal{F}_w$ is described by a quaternion $r_c$. The image plane has a local coordinate system with
the axes of camera such that the edges of its box are aligned with world frame \( A \). Calibration diagram: (marker), while \( p \) for some constants \( k \) describing radial distortions as follows:

\[
p^{(w)} = \delta + rp^{(b)}r^*, \quad (6)
\]

\[
p^{(c)} = r^* (p^{(w)} - c) r_c. \quad (7)
\]

The pinhole camera model would generate the following image coordinates:

\[
p^{(p)} = \frac{f}{k^*} (i, j)^T p^{(c)}, \quad (8)
\]

where \( f \) is the camera’s focal length. However, for a wide-angle lens, with which our camera is equipped, significant distortions are present in the image, and would severely affect the accuracy of pixel locations on the object under the pinhole model. Hence, Brown’s distortion model \([2]\) is used to map \( p^{(p)} \) generated under the pinhole model to the following radially distorted point predicted in the actual image:

\[
p^{(i)} = \frac{1}{\mu} \alpha \left( p^{(p)} \right) p^{(p)} + \left( u_c, v_c \right), \quad (9)
\]

where \( \mu \) is the width per pixel, and \( \alpha \) is a scalar function describing radial distortions as follows:

\[
\alpha \left( p^{(p)} \right) = 1 + k_1 \| p^{(p)} \|^2 + k_2 \| p^{(p)} \|^4 + k_3 \| p^{(p)} \|^6; \quad (10)
\]

for some constants \( k_1, k_2, \) and \( k_3 \). Also in (9), \( u_c \) and \( v_c \) are the image coordinates of the center of projection. Due to manufacturing inaccuracy of the lens, the center of projection is not exactly at the geometric center of the image, and the focal length \( f \) as specified is also not exact.

Note that \( p^{(b)}, p^{(w)} \) and \( p^{(c)} \) are 3-vectors with \( p^{(b)} \) being determined beforehand (as the position of a marker), while \( p^{(p)} \) and \( p^{(i)} \) are 2-vectors. The sequence of transformations is best summarized in the following diagram:

\[
p^{(b)} \rightarrow p^{(w)} \rightarrow p^{(c)} \rightarrow p^{(p)} \rightarrow p^{(i)}, \quad (11)
\]

A. Calibration

We first determine the quaternion \( r_c \) that rotates the world frame \( F_w \) to the camera frame \( F_c \). Place the camera such that the edges of its box are aligned with the axes of \( F_w \). Measure several vertices of the box. Then measure the same set of vertices with the camera at its mounting position. This sets up the correspondences between the two sets of measure points. Apply Horn’s method \([7]\) to calculate \( r_c \).

There are six intrinsic parameters for the camera: \( u_c, v_c, f, k_1, k_2, k_3 \). They are determined via calibration in a least-squares manner. We use \( l \) points \( p_1^{(w)}, \ldots, p_l^{(w)} \) in the world coordinates, and obtain their image coordinates \( u_1, \ldots, u_l \). This is done in the least-squares manner by minimizing the following function:

\[
g(u_c, v_c, f, k_1, k_2, k_3) = \sum_{j=1}^{l} \left( p^{(i)} \left( p_j^{(w)} \right) - u_j \right)^2, \quad (12)
\]

where the image coordinates \( p^{(i)} \) is a function of the world coordinates \( p^{(w)} \) by concatenating the three steps described by equations (7), (8), and (9).

The gradient of \( g \) is

\[
\nabla g = 2 \sum_{j=1}^{l} \left( p^{(i)} \left( p_j^{(w)} \right) - u_j \right)^T \quad (13)
\]

\[
\left( \frac{\partial p^{(i)}}{\partial u_c}, \frac{\partial p^{(i)}}{\partial v_c}, \frac{\partial p^{(i)}}{\partial f}, \frac{\partial p^{(i)}}{\partial k_1}, \frac{\partial p^{(i)}}{\partial k_2}, \frac{\partial p^{(i)}}{\partial k_3} \right) \bigg|_{p^{(p)}} (13),
\]

where the evaluation point \( p^{(p)} \) is obtained by applying (7) and (8) sequentially. The partial derivatives of \( p^{(i)} \) with respect to the six parameters are given below:

\[
\begin{align*}
\frac{\partial p^{(i)}}{\partial u_c} &= \left( \begin{array}{c} 1 \\ 0 \end{array} \right), \\
\frac{\partial p^{(i)}}{\partial v_c} &= \left( \begin{array}{c} 0 \\ 1 \end{array} \right), \\
\frac{\partial p^{(i)}}{\partial f} &= \frac{1}{k^*} \frac{\partial p^{(p)}}{\partial p^{(c)}} \cdot (\hat{i}, j)^T p^{(c)}, \\
\frac{\partial p^{(i)}}{\partial k_i} &= \frac{1}{\mu} \| p^{(p)} \|^2 \cdot p^{(p)}, \quad i = 1, 2, 3,
\end{align*}
\]

where the partial derivative \( \partial p^{(i)}/\partial p^{(p)} \) is given in (17) in Appendix A.

The function \( g \) is minimized using the steepest descent method starting with initial estimates:

\[
u_c^{(0)}, v_c^{(0)}, f^{(0)}, k_1^{(0)}, k_2^{(0)}, k_3^{(0)},
\]

where \( u_c^{(0)} \) and \( v_c^{(0)} \) are halves of the image’s width and height, respectively (namely, they give the position of the center in the image frame with the origin at the image’s upper left corner), and \( f^{(0)} \) is provided by the manufacturer. Initial values \( k_1^{(0)}, k_2^{(0)}, k_3^{(0)} \) are obtained from employing a widely used calibration method \([17]\).
IV. EXTENDED KALMAN FILTER

In this section, we set up an extended Kalman filter (EKF) [14, p. 405] to observe the state \( s = (s^T, r^T, \omega^T, v^T) \) of the flying object from the images of \( m \) marks on the object at discrete time instants. The centers \( p_1^{(i)}, \ldots, p_m^{(i)} \) of these marks are extracted as feature points. Write

\[
y = \begin{pmatrix} p_1^{(i)} \\ \vdots \\ p_m^{(i)} \end{pmatrix} = h(s, \epsilon), \tag{14}
\]

where the measurement error \( \epsilon \) follows the normal distribution with zero mean and some covariance matrix \( R \), i.e., \( \epsilon \sim (0, R) \).

We make use of the Jacobians of \( a \) and \( h \):

\[
A = \frac{\partial a}{\partial s} \quad \text{and} \quad H = \frac{\partial h}{\partial s}, \tag{15}
\]

which are derived in Appendix A. Denote by \( \bar{s} \) the state estimate. The covariance of the error in \( \bar{s} \) is the following \( 13 \times 13 \) matrix:

\[
P = E ((s - \bar{s})(\bar{s} - s)^T),
\]

where \( E \) is the mean function. The EKF algorithm is carried out at time instants \( k = 1, 2, \ldots \). At time instant \( k \), denote the value of a term just before the measurement by the superscript ‘−’ while its updated value just after the measurement by the superscript ‘+’. The a priori state estimate is denoted by \( \bar{s}_k^- \), the measurement by \( y_k \), and the a posteriori state estimate by \( \bar{s}_k^+ \). Also, denote \( H_k = H(\bar{s}_k^-) \).

The EKF algorithm performs the following steps:

1) Initialize the filter:

\[
\bar{s}_0^- = E(s_0), \quad P_0^- = E((s_0 - \bar{s}_0^-)(s_0 - \bar{s}_0^-)^T).
\]

2) For \( k = 1, 2, \ldots \), iterate the following.

a) Obtain \( \bar{s}_k^- \) and \( P_k^- \) from \( \bar{s}_{k-1}^- \) and \( P_{k-1}^- \) through integration of the state estimate and its covariance over the period between times \((k-1)^- \) and \( k^- \) as follows:

\[
\bar{s}^- = a(\bar{s}), \quad \bar{P}^- = AP + PA^T.
\]

b) At time \( k \), carry out the following sequentially:

\[
K_k = P_k^- H_k^T (H_k P_k^- H_k^T + R)^{-1}, \quad \bar{s}_k^+ = \bar{s}_k^- + K_k (y_k - h(\bar{s}_k^-, 0)), \quad P_k^+ = (I_{13} - K_k H_k) P_k^- (I_{13} - K_k H_k)^T + K_k R K_k^T.
\]

V. EXPERIMENT

Experiments were performed using two Ximea MQ022CG-CM high-speed, color cameras equipped with Navitar NMV-6 lenses with a 102 degree field of view. The cameras were carefully calibrated using the procedure from Section III-A with data points obtained from images of a large 27×71 checkerboard pattern. The resulting calibration parameters are provided in Table I.

<table>
<thead>
<tr>
<th>( u_c ) (px)</th>
<th>( v_c ) (px)</th>
<th>( f ) (mm)</th>
<th>( k_1 ) (10^3)</th>
<th>( k_2 ) (10^3)</th>
<th>( k_3 ) (10^11)</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>1009.76</td>
<td>596.29</td>
<td>6.318</td>
<td>-7.375</td>
<td>3.036</td>
</tr>
<tr>
<td>2</td>
<td>1026.07</td>
<td>590.85</td>
<td>6.324</td>
<td>-8.422</td>
<td>9.612</td>
</tr>
</tbody>
</table>

TABLE I

CALIBRATION PARAMETERS

A hollow, plastic cuboid marked with different colors on each of its six faces was thrown by hand across the cameras view. The colored marks form six unique features that were placed at the center of each face such that they align with the axes of the body frame. Images of the object’s flight were captured and then processed using color information to extract the features. The cameras were positioned such that any pose of the cuboid would have at least three features visible to the two cameras. These feature points were then input into the EKF formulated in Section IV.

Fig. 3 shows eight frames from video taken of the flying object. Each pair of images are from the two cameras triggered at the same time. The estimated pose of the object is drawn in black as three axes which form a body frame. Each line extends out to a colored point that represents the estimated position of the corresponding feature of that color. Additionally, the detected centroid of each visible feature is drawn as a point in the same color. The images in (a) show the initial guess of the object’s position and orientation, which were set in (c) and (d).
Fig. 3. Images of the cuboid at the (a) 0th, (b) 5th, (c) 20th, and (d) 45th iterations of the EKF. Each iteration shows an image from both cameras with the estimated pose drawn as black lines forming a body frame. Colored points at the end of each line locate the estimated position of the feature in the same color.

The trace of the error covariance matrix $P_k$, which the EKF is known to minimize [14], was reduced from the initial value of 24.08 to 0.031 over 70 iterations. Although, noticeable error is visible in images of the object and its estimated pose. Fig. 4 plots the error normalized over the object’s size, denoted $\delta$. That is, at each time instant $k$, the error was calculated as $\delta_j = \|p_j^{(i)} - \tilde{p}_j^{(i)}\|/\sqrt{A}$, for $j = 1, \ldots, m$, where $A$ is the objects pixel area, and the estimated feature point $\tilde{p}_j^{(i)}$ was obtained from the current state $\tilde{s}_k$. As shown in the plot, this error starts high due to the initial estimate of the state, and rapidly shrinks as the EKF iterates. With the EKF converging early, later measurements have less effect in correcting the estimate, allowing a slight error accumulation.

Fig. 4. Error between the expected and observed feature points relative to the object’s size in pixels. Solid lines show the errors for camera 1, and dashed lines for camera 2. Each line’s color matches the color of the mark on the object in Fig. 3. These colors are blue, red, green, orange, and purple (drawn here as magenta for better contrast). Gaps between the plotted lines are formed from features going in and out of view of the cameras. Refer to the end of Section V for details on $\delta$.

VI. DISCUSSION

Accurate, high-speed motion estimation of an object flying and tumbling in space can be a daunting task for two fixed cameras. Modeling of flight dynamics can be inaccurate due to air drag and Magnus effects. Cameras can have significant nonlinear distortion to prevent sources of feature points from being detected with high accuracy. Feature correspondences across consecutive frames can be difficult to keep track of since features appear and disappear as the object is rotating in the air. Real-time image processing at a high frame rate puts heavy load on computation, of which a certain portion has to be allocated to EKF-based estimation.

We have presented an approach that demonstrates reasonable accuracy by formulating an EKF that considers a full model of system dynamics and camera projection. Shortcomings of the preliminary results presented here are expected to be addressed with two future efforts. The first will come from vision. Further investigation is required to improve modeling of camera distortion. More thorough calibration needs to be conducted to achieve small error on locating a point in a large volume of space. More cameras placed around the flight can generate images of different sides of object to improve the orientation estimation.

The second effort will be on tuning of the EKF parameters. Further refinement of the error covariances and initial estimates will be performed specific to our experiment to ensure proper convergence. The EKF
may also be adapted to more modern variations for comparison, such as the Unscented Kalman Filter (UKF) and Invariant EKF (IEKF).

**APPENDIX A**

**PARTIAL DERIVATIVES USED BY THE EKF**

The EKF described in Section IV requires linearization of both the system dynamics (4) and the observation function (14). This is done through obtaining the partial derivatives of the functions $a$ and $h$ with respect to the state $s$. These partial derivatives all depend on those of the feature points $p_k^{(i)}$, $1 \leq k \leq m$, in the image with respect to $s$. For such a point $p$,

$$ \frac{\partial p^{(i)}}{\partial s} = \frac{\partial p^{(i)}}{\partial s} \cdot \frac{\partial p^{(w)}}{\partial s}. $$

We will obtain the two partial derivatives on the right hand side. Then the left hand side can be evaluated at $p_k^{(i)}$, $1 \leq k \leq m$.

From (7)–(9) we have

$$ \frac{\partial p^{(i)}}{\partial p^{(w)}} = \frac{\partial p^{(i)}}{\partial p^{(p)}} \cdot \frac{\partial p^{(p)}}{\partial p^{(c)}} \cdot \frac{\partial p^{(c)}}{\partial p^{(w)}}, \quad (16) $$

where, denoting $r_c = r_{c0} + r_c$,

$$ \frac{\partial p^{(i)}}{\partial p^{(p)}} = \frac{1}{\mu} \left( p^{(p)} \frac{\partial}{\partial p^{(p)}} (\alpha I_2) \right), \quad (17) $$

$$ \frac{\partial p^{(p)}}{\partial p^{(c)}} = -\frac{f}{(kT p^{(c)})^2} (\hat{i}, \hat{j})^T (\hat{i}, \hat{j})^T $$

$$ \frac{\partial p^{(c)}}{\partial p^{(w)}} = \left( r_{c0}^2 - ||r_c^2||^2 \right) I_3 + 2 r_c r_v^T - 2 r_{c0} r_v \times . \quad (19) $$

The last derivative is from Appendix C in [8], with $r_v \times$ denoting the $3 \times 3$ anti-symmetric matrix whose product with a vector yields the cross product of $r_v$ with that vector.

From (16) we have $\partial p^{(i)}/\partial p^{(w)}$ so we need only obtain $\partial p^{(w)}/\partial s$. It is clear from (6) that the velocity and angular velocity have no part in $p^{(w)}$; hence

$$ \frac{\partial p^{(w)}}{\partial v} = 0 \quad \text{and} \quad \frac{\partial p^{(w)}}{\partial \omega} = 0. $$

Differentiate (6) with respect to $\delta$:

$$ \frac{\partial p^{(w)}}{\partial \delta} = I_3, \quad (20) $$

and then with respect to $r$ [8]:

$$ \frac{\partial p^{(w)}}{\partial r} = 2 \left( r_0 p^{(b)} + r \times p^{(b)}, -p^{(b)} r^T + r^T p^{(b)} I_3 + r p^{(b)} T - r_0 p^{(b)} \times \right) \quad (21) $$

We easily write out the matrix $H$ defined in (15) from differentiating the function $h$ given in (14):

$$ H = \begin{pmatrix} \frac{\partial p^{(i)}}{\partial s} \end{pmatrix}. $$

Continue to the Jacobian $A$ of the function $a$. For convenience, we let $a_1, a_2, a_3, a_4$ be its four vector components, respectively. Then

$$ \frac{\partial a}{\partial s} = \begin{pmatrix} 0 & 0 & I_3 & 0 \\ 0 & \frac{\partial a_2}{\partial r} & 0 & \frac{\partial a_2}{\partial \omega} \\ 0 & 0 & \frac{\partial a_3}{\partial v} & \frac{\partial a_3}{\partial \omega} \\ 0 & 0 & \frac{\partial a_4}{\partial v} & \frac{\partial a_4}{\partial \omega} \end{pmatrix}. \quad (22) $$

The two partial derivatives of $a_2$ in the above follow from Appendix C in [8]:

$$ \frac{\partial a_2}{\partial r} = \frac{\partial}{\partial r} \begin{pmatrix} 1 & \frac{1}{2} r \omega \end{pmatrix}, $$

$$ \frac{\partial a_2}{\partial \omega} = \frac{1}{2} \begin{pmatrix} r_0 I_3 + r \times \end{pmatrix}. $$

The other three partial derivatives are given below:

$$ \frac{\partial a_3}{\partial v} = \frac{\partial}{\partial v} \begin{pmatrix} g - e_d ||v|| + e_m \omega \times v \end{pmatrix}, $$

$$ = -e_d \begin{pmatrix} v \frac{\partial ||v||}{\partial v} + ||v|| I_3 \end{pmatrix} + e_m \omega \times, $$

$$ \frac{\partial a_3}{\partial \omega} = -e_m v \times, $$

$$ \frac{\partial a_4}{\partial v} = \frac{\partial}{\partial v} \begin{pmatrix} -Q^{-1} (\omega \times Q \omega) \end{pmatrix}, $$

$$ = -Q^{-1} \left( -(Q \omega) \times + \omega \times Q \right). $$

**REFERENCES**


