Unmanned Systems, Vol. 0, No. 0 (2015) 1–11

Design and Implementation of a Low Cost Mini Quadrotor for Vision based Manoeuvres in GPS Denied Environments

Chang Liu[a,*], Stephen D. Prior[a] and James P. Scanlan[a]

[a] *Faculty of Engineering and the Environment, University of Southampton, Southampton, SO16 7QF, UK*
E-mail: cl21g11r@soton.ac.uk

This paper presents the design and implementation detail of an advanced mini quadrotor system, including the low cost commercial-off-the-shelf (COTS) electronics and advanced control algorithm. The proposed quadrotor has a gross takeoff weight of 758 g and 360 mm frame diagonal size. It is capable of semi-autonomous manoeuvre in GPS denied environments, solely relying on onboard sensors and computers. A globally defined quadrotor model is formularised, and a nonlinear velocity tracking controller is implemented on the special Euclidean group SE(3). An optical flow and ultrasonic based onboard downward-facing camera is used as the primary sensor to provide velocity and altitude measurement feedback for the controller. The control and sensor fusion algorithm is developed under Arduino compatible open source electronics.

Keywords: GPS denied; optical flow; robust control; manoeuvres.

1. Introduction

Unmanned aerial vehicles (UAVs) are being considered in an increasing number of defence-related applications, for the purpose of reducing downside risk and rising confidence in mission success. Morever, the civilian market is predicted to rapidly expand over the next decade.[1] The quadrotor is one of the most popular subset of UAVs. Because of its agile manoeuvrability, as well as its ability of vertical take-off and landing (VTOL) and stable hovering, it is commonly agreed to be an ideal candidate for search and rescue, surveillance, exploration, agriculture, monitoring and military applications in both indoor and outdoor environments.

Over the last decade, the Global Positioning System (GPS) has been the key to enabling the autonomy of UAVs. It provides global localization service with the best accuracy of 1-2 metres. However, recently, due to the proven weakness of GPS signal and rapid development of onboard sensing and computation capability, there has been growing interest in developing and researching alternative navigation methods for UAVs in GPS denied environments.[2–7] The successful implementations will not only improve system robustness under GPS failure, but also enable a new range of applications out of GPS coverage.

A mini quadrotor is defined to carry under 2 kg payload,[1] which is sufficient for light weight perception sensors (such as cameras, laser scanner, radar and ultrasonic sensor) and embedded computer, which are essential for an autonomous navigation. Additionally, because they are low cost, easy to maintain, and safe to operate, these make them very good test-beds for research and development.[8]

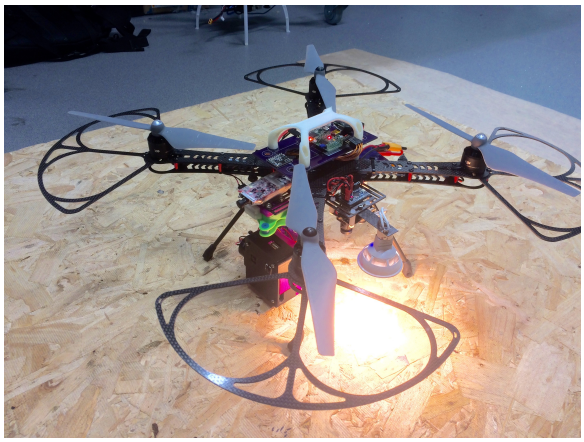


Fig. 1: The developed quadrotor.

In this paper, the vision-based method is believed to be the optimal sensor for navigation. The reason is that a camera has significant advantages over other sensors, such

*Aeronautics, Astronautics and Computational Engineering Design Group, Faculty of Engineering and the Environment, Engineering Centre of Excellence, Building 176, Boldrewood Campus, University of Southampton, Southampton, SO16 7QF, UK.

as low mass, low power consumption, low price, adjustable field of view (FOV), high accuracy, additional colour information and long range. During the past five years, world top research institutes had paid high attention on developing advanced visual-based simultaneous localization and mapping (vSLAM) algorithms based on structure from motion (SFM) theory.[9–15] Those algorithms are efficient enough to execute in near real time on the modern onboard embedded computer, which makes it possible for a mini quadrotor to perform complete autonomous tasks in GPS-denied environments solely relying on onboard sensors and computers by utilising similar technology. Moreover, the visual scale problem, which was the main challenge of involving vision in control loop, is addressed to various extent by fusing onboard inertial measurements (accelerometer and gyroscope), which is named visual inertial navigation system (VINS).[6,16–23]

Therefore, in order to integrate the similar technologies into mini quadrotor platforms and for future improvements and developments, a suitable platform is required as the fundamental testbed. The most popular research platform in this category is Hummingbird quadrotor sold by Ascending Technologies,[24] with the state-of-the-art quadrotor autonomous control theory.[8] Thus, extensive research has been conducted on mini quadrotor development.[25–27] However, none of the above considers vision feedback in the quadrotor control loop. On the contrary, there are a few of commercial platforms, which are capable of vision based navigation, such as AR-drone and Bebop by Parrot, Phantom-3 and Inspire-1 by DJI. However, due to the nature of their consumer level application and the intellectual properties, they are all lack of accessibility and extendibility for the usage as a research platform with onboard processing. Given this situation, as a continuous work based previous publication,[28] this mini quadrotor, as shown in Fig. 1, is designed and implemented aiming to provide a test-bed for developing similar algorithms in the near future.

The rest of this paper is formed as follows: in Section 2, it explains the modelling of the quadrotor dynamics, and Section 3 describes the control architecture design based on the dynamic model. Then, Section 4 summarizes quadrotor implementation details, and then, Section 5 shows the test data to demonstrate the system performance. Lastly, Section 6 concludes and proposes future work.

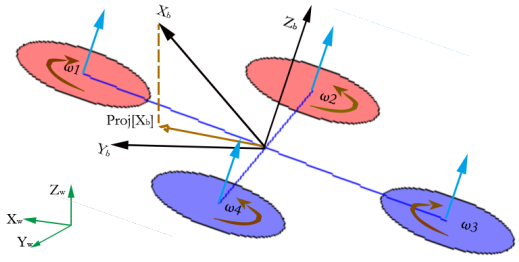


Fig. 2: Coordinate system and quadrotor setup.

2. Quadrotor Dynamics Modelling

This section presents the nonlinear dynamic model of the mini quadrotor, which forms the basis for the controller synthesis in Section 3.

The coordinate frames and system setup is indicated in Fig. 2. Quadrotor body frame is fixed to the quadrotor body following right hand rule with X_b–axis pointing forward, Y_b–axis pointing left, Z_b–axis pointing up. World frame is fixed to the world, and is defined to have the same origin with body frame at the moment when the quadrotor connects to a battery. Z_w–axis points to the opposite direction of gravity and X_w–axis points to the same direction as quadrotor heading when connects to battery. The angles defined in the system follow the right hand rule. Fig. 2 also shows that the quadrotor has the cross configuration and four motors are numbered 1–4, with spinning directions as indicated.

The rest of the paper uses $\mathbf{x_w}$, $\mathbf{y_w}$, $\mathbf{z_w} \in \mathbb{R}^3$ to denote unit vectors of the three world coordinates, thus $\mathbf{x_w} = (1,0,0)^\top, \mathbf{y_w} = (0,1,0)^\top, \mathbf{z_w} = (0,0,1)^\top$. And the unit vectors of the body frame are expressed in world frame as $\mathbf{x_b}, \mathbf{y_b}, \mathbf{z_b} \in \mathbb{R}^3$. The 3D rotation of the quadrotor body is represented by rotation matrix $\mathbf{R} \in SO(3)$. Therefore, we have $\mathbf{x_b} = \mathbf{R}\mathbf{x_w}$, $\mathbf{y_b} = \mathbf{R}\mathbf{y_w}$, $\mathbf{z_b} = \mathbf{R}\mathbf{z_w}$, thus $\mathbf{R}$ can also be expressed as:

$$\mathbf{R} = [\mathbf{x_b}\ \mathbf{y_b}\ \mathbf{z_b}]\,. \tag{1}$$

Note that we express the heading of the quadrotor as the unit vector parallel to the projection of X_b–axis onto the X_w–Y_w plane in world frame, denoted as $proj[\mathbf{x_b}] \in \mathbb{R}^3$, in Fig. 2.

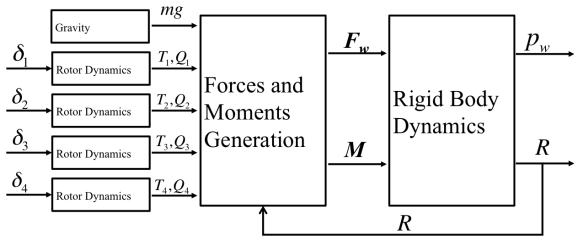


Fig. 3: Overview of quadrotor dynamics model.

The overview of the nonlinear model is shown in Fig. 3, where inputs of the model, δ_n, are the normalized pulse width modulation (PWM) command signal to the electronic speed controller (ESC) of motors, and outputs of the model are 3 dimensional (3D) position vector $\mathbf{p_w}$ $(= (x, y, z)^\top)$ in world frame and body rotation matrix $\mathbf{R}$. The following subsections will explain the included components individually.

2.1. *Rotor dynamics*

The propulsion system of the quadrotor includes two pairs of counter-rotating ESC-motor-propeller systems. The dy-

namics of the four systems are identical and are approximated by the rotor dynamics model. The model receives the normalized PWM command δ and outputs thrust, T, in g and torque, Q, in Nm. If a propeller, with diameter D, rotates at n angular velocity in free air, whose density is ρ, assuming that the propeller drag is equal to the torque applied to spin, the thrust T and torque Q that it produces can be modelled as:

$$T = \rho n^2 D^4 C_T \tag{2}$$

$$Q = \rho n^2 D^5 C_Q, \tag{3}$$

where C_T and C_Q are thrust and torque coefficients respectively, which depend on propeller geometry, profile and Reynolds number. Furthermore, by assuming an ideal closed-loop ESC-motor system, which spins the propeller at the angular velocity that is linear to the normalised pulse width modulation (PWM) command, δ, ranging from 0 to 1, without mechanical delay:

$$n = k\delta - c, \tag{4}$$

where k and c are constants. Therefore, to model the propulsion system, simply substitute (4) into (2) and (3). However, in practice, to model the propulsion system with given parameters (ρ, D, C_T and C_Q), the derived equations can be simplified as:

$$T = c_T(\delta - c_o)^2, \tag{5}$$

$$Q = c_Q T, \tag{6}$$

where c_T, c_Q and c_o are constants determined by the given parameters, and c_T and c_o can be easily obtained from static thrust tests.

2.2. *Force and moment generation*

All the forces and moments applied to the quadrotor result in movement. They are generally generated by four different sources,[29] i.e., the gravitational force, the rotor thrust and moment, rotor reaction torques, and their gyroscopic effects. However, the last two have insignificant effect on overall forces and moments, thus we only consider the former two. Therefore, this module converts all the forces and moments into a force vector, $\mathbf{F_w} \in \mathbb{R}^3$ in world frame, and a moment vector, $\mathbf{M} \in \mathbb{R}^3$ about each body axis.

The gravitational force in world frame only applies to negative Z_w axis, which yields:

$$\mathbf{F_{gravity}} = \begin{pmatrix} 0 \\ 0 \\ -mg \end{pmatrix}, \tag{7}$$

where m is quadrotor mass and g is standard gravitational acceleration.

Besides, each of the four rotors on the quadrotor generates thrust, T_n, and torque, Q_n, where $n = 1, 2, 3, 4$ (the numbering order is indicated in Fig. 2). The force generated by the four rotors applies to the positive Z_b axis in body frame, thus by rotating it into the world frame, we get:

$$\mathbf{F_{thrust}} = \mathbf{R}\begin{pmatrix} 0 \\ 0 \\ T_{total} \end{pmatrix} = \mathbf{z_b} T_{total}, \tag{8}$$

$$T_{total} = T_1 + T_2 + T_3 + T_4, \tag{9}$$

where T_{total} is the total thrust provided by the four rotors. $\mathbf{R}$ is the rotation matrix of the body frame. And $\mathbf{z_b}$ is the unit vector of Z_b–axis, as defined in (1).

Therefore, the total force applied onto the quadrotor in world frame can be derived as:

$$\mathbf{F_w} = \mathbf{F_{gravity}} + \mathbf{F_{thrust}}. \tag{10}$$

The other output from the module is the moment vector in body frame, which is approximated in this paper to be generate by the thrusts and torques of the four rotors. Roll moment is contributed by the thrust difference between rotors 1, 4 and 2, 3. Pitch moment is contributed by the thrust difference between rotors 1, 2 and 3, 4. Yaw moment is contributed by the torque difference between rotors 1, 3 and 2, 4. Thus, by also substituting (6), it then can be formulated as:

$$\mathbf{M} = \mathbf{BCT}, \tag{11}$$

where:

$$\mathbf{B} = \begin{bmatrix} \frac{\sqrt{2}}{2}l & 0 & 0 \\ 0 & \frac{\sqrt{2}}{2}l & 0 \\ 0 & 0 & c_Q \end{bmatrix}, \tag{12}$$

$$\mathbf{C} = \begin{bmatrix} 1 & -1 & -1 & 1 \\ -1 & -1 & 1 & 1 \\ 1 & -1 & 1 & -1 \end{bmatrix}, \tag{13}$$

$$\mathbf{T} = (T_1, T_2, T_3, T_4)^\top. \tag{14}$$

The thrust vector, $\mathbf{T}$, represents the thrusts generated by the four rotors, and l is the distance between rotors and quadrotor centre of mass.

2.3. *Rigid-body dynamics*

Rigid-body dynamics formularises the translational and rotational dynamics of the quadrotor, by utilising the simplified Newton-Euler formalism. Therefore, the resulting position vector, $\mathbf{p_w} \in \mathbb{R}^3$ in world frame, and angular speed vector $\mathbf{\Omega} \in \mathbb{R}^3$ about each body axis, can be obtained as:

$$m\ddot{\mathbf{p}}_\mathbf{w} = \mathbf{F_w}, \tag{15}$$

$$\mathbf{J}\dot{\mathbf{\Omega}} = \mathbf{M}, \tag{16}$$

where $\mathbf{J}$ is the inertia matrix of the quadrotor, and since our quadrotor is approximately four way symmetrical, $\mathbf{J}$ is assumed to be a diagonal matrix, as:

$$\mathbf{J} = \begin{bmatrix} J_X & 0 & 0 \\ 0 & J_Y & 0 \\ 0 & 0 & J_Z \end{bmatrix}, \tag{17}$$

where J_X, J_Y, J_Z are the moment of inertia values around each axis of body frame.

3. Controller Design

Based on the dynamics model developed in the previous section, a nonlinear robust controller is designed to ultimately control the quadrotor 3D position $\mathbf{p_w}$ and heading $proj[\mathbf{x_b}]$ to match user input command $\mathbf{p_w^\star}$ and $proj[\mathbf{x_b^\star}]$. As shown in Fig. 4, three sub-controllers: attitude controller, acceleration controller and position controller, are developed. The quadrotor is controlled accordingly by taking the feedback measurement from inertial measurement unit (IMU) and vision based position sensor. Note that the detail of the position sensor design is not the focus of this section, thus here we assume the position is obtained from the position sensor.

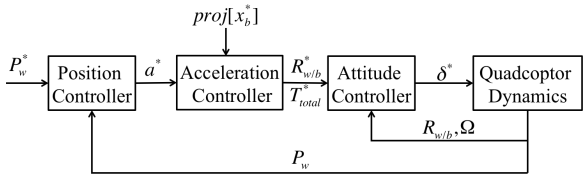


Fig. 4: Overview of the nonlinear controller.

3.1. *Attitude controller*

The attitude controller receives the desired body orientation represented as rotation matrix $\mathbf{R}^\star$, and desired total thrust provided by the four rotors, $T^\star_{total}$, from acceleration controller output. With the help of attitude feedback measurement, $\mathbf{R}$, from IMU attitude fusion, and angular velocity, $\mathbf{\Omega}$, directly measured from gyroscope, then it commands normalised PWM signals, $\delta^\star_n$, to the four ESCs of motors. It is designed to minimise both the attitude tracking error, $\mathbf{e_R} \in \mathbb{R}^3$, and angular velocity error, $\mathbf{e_\Omega} \in \mathbb{R}^3$, while maintaining the total thrust, T_{total}, as commanded.

Given the desired orientation, $\mathbf{R}^\star$, the attitude error, $\mathbf{e_R}$, is defined to be the sine of the angle of rotation about each body axis to go from $\mathbf{R}$ to $\mathbf{R}^\star$. It can be formularised as:[30]

$$\mathbf{e}_{\mathbf{R}\times} = \frac{1}{2}(\mathbf{R}^{\star\top}\mathbf{R} - \mathbf{R}^\top\mathbf{R}^\star), \tag{18}$$

which yields a skew-symmetric matrix in the form of:

$$\mathbf{e}_{\mathbf{R}\times} = \begin{bmatrix} 0 & -e_{Rz} & e_{Ry} \\ e_{Rz} & 0 & -e_{Rx} \\ -e_{Ry} & e_{Rx} & 0 \end{bmatrix}, \tag{19}$$

where the cross map $\times : \mathbb{R}^3 \to \mathfrak{so}(3)$, thus we get $\mathbf{e_R} = (e_{Rx}, e_{Ry}, e_{Rz})^\top$. Moreover, the angular velocity error, $\mathbf{e_\Omega}$, is defined as:[30]

$$\mathbf{e_\Omega} = \mathbf{\Omega} - \mathbf{R}^\top\mathbf{R}^\star\mathbf{\Omega}^\star, \tag{20}$$

where $\mathbf{\Omega}^\star \in \mathbb{R}^3$ is the angular velocity of the desired rotation, $\mathbf{R}^\star$, about each axis of the desired body frame. It can be obtained by the tangent operator equation of the desired rotation matrix, as:

$$\mathbf{\Omega}^\star = (\mathbf{R}^{\star\top}\dot{\mathbf{R}}^\star)^\vee. \tag{21}$$

Then, we can apply the proportional–derivative (PD) control law to compute the desired angular acceleration vector, $\boldsymbol{\alpha}^\star \in \mathbb{R}^3$, about each body axis to be applied to quadrotor body in order to minimise the difference between $\mathbf{R}$ and $\mathbf{R}^\star$, thus:

$$\boldsymbol{\alpha}^\star = -\mathbf{k_p}\mathbf{e_R} - \mathbf{k_d}\mathbf{e_\Omega}, \tag{22}$$

where $\mathbf{k_p}, \mathbf{k_d} \in \mathbb{R}^3$ are non-negative gain vectors, can be tuned, depending on the aggressiveness of the required manoeuvre.

Therefore, based on (16), the desired moment, $\mathbf{M}^\star \in \mathbb{R}^3$, to be generated onto quadrotor body can be computed by:

$$\mathbf{M}^\star = \mathbf{J}^{-1}\boldsymbol{\alpha}^\star, \tag{23}$$

And then we add total thrust control. Thus, by combining (9) and (11) we can say:

$$\begin{bmatrix} \mathbf{M}^\star \\ \frac{1}{4}T^\star_{total} \end{bmatrix} = \begin{bmatrix} \mathbf{B} & \mathbf{0}_{3\times1} \\ \mathbf{0}_{1\times3} & 1 \end{bmatrix} \begin{bmatrix} \mathbf{C} \\ \mathbf{1}_{1\times4} \end{bmatrix} \mathbf{T}^\star, \tag{24}$$

where matrix $\mathbf{B}$ and $\mathbf{C}$ are defined in (12) and (13) respectively, and $\mathbf{T}^\star = (T^\star_1, T^\star_2, T^\star_3, T^\star_4)^\top$ is desired thrust vector, which represents the desired thrust command to each rotor. Therefore, the thrust command for individual rotors can be computed by reversing (24):

$$\mathbf{T}^\star = \begin{bmatrix} \mathbf{C} \\ \mathbf{1}_{1\times4} \end{bmatrix}^{-1} \begin{bmatrix} \mathbf{B} & \mathbf{0}_{3\times1} \\ \mathbf{0}_{1\times3} & 1 \end{bmatrix}^{-1} \begin{bmatrix} \mathbf{M}^\star \\ \frac{1}{4}T^\star_{total} \end{bmatrix}. \tag{25}$$

This expression of thrust commands not only applies the desired moment to the quadrotor, but also ensures the total thrust provided by the four rotors is equal to $T^\star_{total}$.

Finally, to generate the normalised PWM signal command, $\delta^\star_n$, for individual rotor, simply apply inverted (5) on $T^\star_n$. Then we get:

$$\delta^\star_n = \sqrt{\frac{T^\star_n}{c_T}} + c_o. \tag{26}$$

This nonlinear attitude tracking controller is demonstrated to recover from any initial orientation in simulation,[30] and it is proved to have exponential stability when the initial attitude error is less than 90°, and it yields almost global exponentially attractiveness when the initial attitude error is less than 180°.

3.2. *Acceleration controller*

The Acceleration controller receives the desired acceleration command vector, $\mathbf{a}^\star = (a_x^\star, a_y^\star, a_z^\star)^\top \in \mathbb{R}^3$, from the position controller output, and quadrotor heading command, $proj[\mathbf{x_b}]$, from user. It then converts the commands into the desired orientation, $\mathbf{R}^\star$, and desired total thrust, $T_{total}^\star$, commands for attitude controller.

For a given acceleration command, $\mathbf{a}^\star$, based on (15) and (10), the desired thrust force acts on the quadrotor in world frame, $\mathbf{F_{thrust}^\star}$, can be computed as:

$$\mathbf{F_{thrust}^\star} = m\mathbf{a}^\star - \mathbf{F_{gravity}}. \tag{27}$$

Based on (8), $\mathbf{F_{thrust}^\star}$ is shown to have the same direction with the desire body Z_b–axis, $\mathbf{z_b^\star}$, with the magnitude equals to desired total thrust, $T_{total}^\star$. Thus:

$$T_{total}^\star = \|\mathbf{F_{thrust}^\star}\|, \tag{28}$$

$$\mathbf{z_b^\star} = \frac{\mathbf{F_{thrust}^\star}}{T_{total}^\star}. \tag{29}$$

Then by assuming the desire heading command, $proj[\mathbf{x_b}]$, is not parallel to $\mathbf{z_b^\star}$, we can obtain the unit axis vectors $\mathbf{y_b^\star}$ and $\mathbf{x_b^\star}$ by:

$$\mathbf{y_b^\star} = \frac{\mathbf{z_b^\star} \times proj[\mathbf{x_b}]}{\|\mathbf{z_b^\star} \times proj[\mathbf{x_b}]\|}, \tag{30}$$

$$\mathbf{x_b^\star} = \mathbf{y_b^\star} \times \mathbf{z_b^\star}. \tag{31}$$

Therefore, the output desired quadrotor rotation matrix will be:

$$\mathbf{R}^\star = [\mathbf{x_b^\star}\ \mathbf{y_b^\star}\ \mathbf{z_b^\star}]. \tag{32}$$

However, when the acceleration command $\mathbf{a}^\star$ given to the acceleration controller results in a desired total thrust, $T_{total}^\star$, which is higher than 80% of the maximum total thrust, $T_{80\%}$, that the four rotors can provide, the system will encounter the tracking instability.

Therefore, an acceleration limit must be conducted in this case. Here, we scale down $\mathbf{a}^\star$ to $\widetilde{\mathbf{a}}^\star = \beta\mathbf{a}^\star$, with $0 < \beta < 1$, so that the quadrotor body acceleration maintain the same direction, while the resulting total thrust is equal to $T_{80\%}$. Thus, in other words:

$$\mathbf{F_{thrust}^\star} = m\widetilde{\mathbf{a}}^\star - \mathbf{F_{gravity}}, \tag{33}$$

$$T_{80\%} = \|\mathbf{F_{thrust}^\star}\|, \tag{34}$$

$$\widetilde{\mathbf{a}}^\star = \beta\mathbf{a}^\star, \tag{35}$$

which results in a quadratic equation of β, in the form of:

$$a\beta^2 + b\beta + c = 0, \tag{36}$$

where:

$$a = \|m\mathbf{a}^\star\|^2, \tag{37}$$

$$b = 2m^2 g a_z^\star, \tag{38}$$

$$c = m^2 g^2 - T_{80\%}^2. \tag{39}$$

Thus β can be obtained from the quadratic formula $\beta = \frac{-b+\sqrt{b^2-4ac}}{2a}$. Then $\mathbf{a}^\star$ can be replaced by $\widetilde{\mathbf{a}}^\star$, $T_{total}^\star = T_{80\%}$, and $\mathbf{R}^\star$ can be computed in the same form as (29), (30), (31), (32).

3.3. *Velocity Controller*

The velocity controller outputs the desired acceleration vector, $\mathbf{a}^\star$, to the acceleration controller, in order to minimises the error between desired velocity, $\dot{\mathbf{p}}_\mathbf{w}^\star$, commanded from the user, and quadrotor velocity measurement, $\dot{\mathbf{p}}_\mathbf{w}$, in the world frame.

The position error vector, $\mathbf{e_p}$, and velocity error vector, $\mathbf{e_v}$, are defined as:

$$\mathbf{e_p} = \mathbf{p_w} - \mathbf{p_w^\star}, \tag{40}$$

$$\mathbf{e_v} = \dot{\mathbf{p}}_\mathbf{w} - \dot{\mathbf{p}}_\mathbf{w}^\star. \tag{41}$$

Then proportional-integral-acceleration (PI-A) control law is applied, which yields an expression as:

$$\mathbf{a}^\star = -\mathbf{k_i'}\mathbf{e_p} - \mathbf{k_p'}\mathbf{e_v} - \mathbf{k_a'}\ddot{\mathbf{p}}_\mathbf{w}, \tag{42}$$

where $\mathbf{k_i'}$, $\mathbf{k_p'}$ and $\mathbf{k_a'}$ are non-negative controller gain vectors, which can be tuned according to the require aggressiveness of the position tracking performance. $\dot{\mathbf{p}}_\mathbf{w}^\star$ is from user command and $\dot{\mathbf{p}}_\mathbf{w}$ is measured from velocity estimator described in next section. $\mathbf{p_w^\star}$ and $\mathbf{p_w}$ are obtained by integrating $\dot{\mathbf{p}}_\mathbf{w}^\star$ and $\dot{\mathbf{p}}_\mathbf{w}$. And $\ddot{\mathbf{p}}_\mathbf{w}$ is obtained directly from accelerometer measurement.

Additionally we add a position-error-integral term with gain value k_h' for z-axis position controller to remove altitude control steady-state error.

4. Implementation

This section summarises the implementation details for the working UAS system, including mechanical setup, and autopilot electronics and software description. The quadrotor basic details are summarised in Table. 1.

Table 1: Platform details.

Quantity name	**Value**	**Unit**
Takoff mass(m)	758	g
Arm Length (l)	180	mm
Propeller size	9.4×5.0	$inch$
Motor Kv	960	RPM/V

4.1. *Chassis and Propulsion System*

We selected a GF360 carbon fibre quadrotor frame[a] for industrial standard design suitable for fast prototyping applications. The 360 mm motor span optimised for 9.4 $inch$ propeller, here we use DJI E310 propulsion system[b] for robustness and simplicity. This results in a 600 mm tip-to-tip span, which is almost the maximum safe size to manoeuvre through standard UK doorways (762 mm width).

We then conducted bench static thrust test of the E310 propulsion system and obtained c_T and c_o in (5) by applying linear regression on square root of T versus δ. The obtained c_o is 48.1385 and c_T is 0.09376, which gives 95% accuracy, as shown in Fig. 5, where the blue curve is the measured data from thrust test, and the red curve is generated from (5) with c_T and c_o equal to above values.

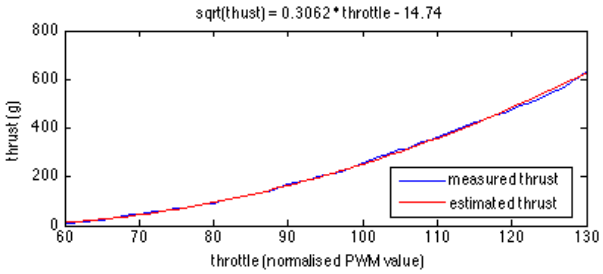


Fig. 5: Propulsion system model.

4.2. *Moment of Inertia*

The moment of inertia of the quadrotor is obtained by averaging between analytical method and direct measurement.

In order to obtain the analytical measurement of the moment of inertia, an approximated CAD model is constructed in SolidWorks®, with the mass assigned to all components individually. The constructed model is shown in Fig. 6.

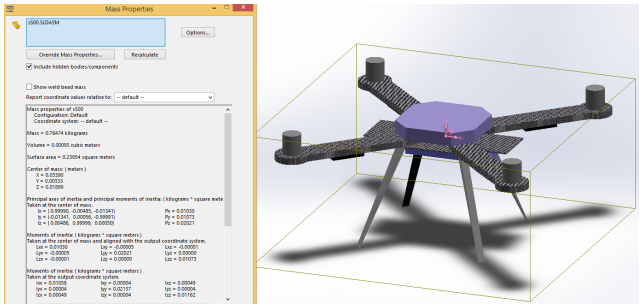


Fig. 6: Quadrotor CAD model.

The direct measurement of moment of inertia is obtained by applying the bifilar pendulum theory, where for each axis, the moment of inertia, J, can be computed by measuring the twist oscillation period with the setup as shown in Fig. 7, the equation used to compute the moment of inertia is:

$$J = \frac{mgT^2b^2}{4\pi^2 L}, \tag{43}$$

where T is the period measured over one oscillation. Here, to improve the accuracy, the averaged period from 40 oscillations is obtained with manual stopwatch. And L, b is indicated in Fig. 7.

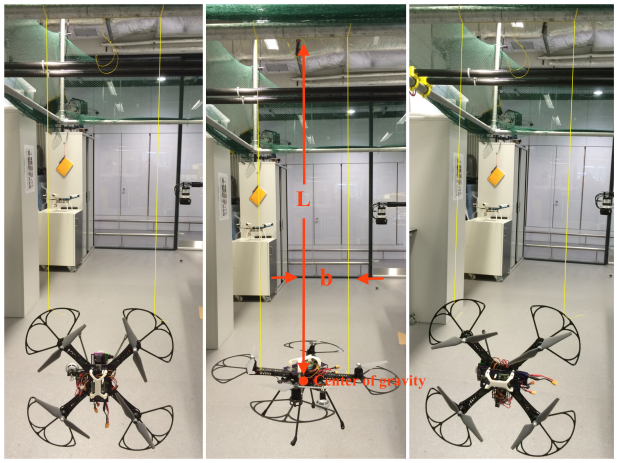


Fig. 7: The bifilar pendulum experiment setup for the three body axis.

The results obtained from the CAD simulation, bifilar pendulum experiment and the average of both are summarised in Table 2. Note that the averaged value is used as the final measurement result.

Table 2: Moment of inertia experiment results.

Quantity name	**Value from CAD**	**Value from Bifilar Pendulum**	**Averaged Value**	**Unit**
J_X	0.01030	0.00875	0.00953	$kg \times m^2$
J_Y	0.02021	0.01863	0.01942	$kg \times m^2$
J_Z	0.01073	0.00921	0.00997	$kg \times m^2$

[a] http://www.dhgate.com/product/gf-360-carbon-fiber-folding-four-axis-quadcopter/187566640.html

[b] http://www.dji.com/product/e310

4.3. *Flight Controller Implementation*

Thanks to the high speed Teensy 3.1 processor and a dedicated servo controller, the control loop implementing Section 3 executes within 3 ms. The physical layout is shown in Fig. 9a and Fig. 9b, and the block diagram in Fig. 8 shows the interactions between components.

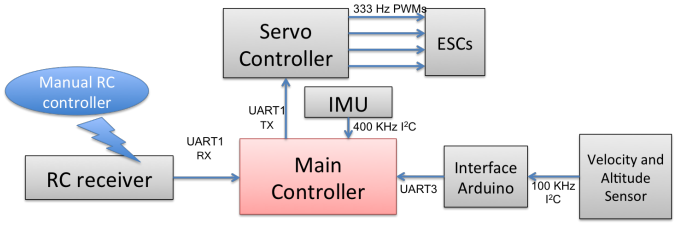
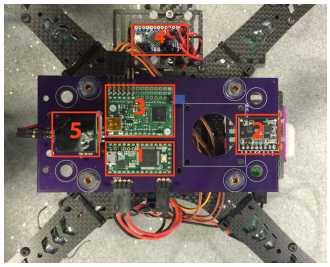
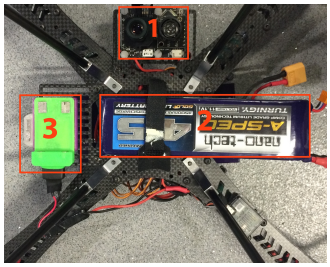


Fig. 8: Block diagram.



(a) Top view.
1. Main Teensy controller;
2. FreeIMU;
3. Servo controller;
4. Interface Arduino;
5. Voltage regulator.



(b) Bottom view.
1. PX4Flow camera;
2. Li-Po battery.
3. Interface to additional camera payload.

Fig. 9: Mechanical system layout.

(1) **Main Controller Board** is based on Teensy 3.1 MCU board[c]. It is an ARM based Arduino compatible development board, which features very small form factor (35×18 mm) and fast processor (ARM Cortex–M4 with up to 96 MHz clock speed). It is ideal for a flight controller.
(2) **IMU** is based on FreeIMU sensor suite[d],[31] including a MPU6050 gyroscope-accelerometer combo-chip, a HMC5883L magnetometer and MS5611-01BA high resolution pressure sensor. However, only the MPU6050 chip is used in this implementation. The orientation fusion estimation uses the library provided with the sensor.
(3) **Servo Controller** is based on Pololu Mini Maestro Servo Controller board[e]. It is a dedicated servo controller board, which features high resolution (0.25 μs) servo PWM output to 12 channels, with update rate up to 333 Hz, and the fast UART Serial protocol makes it easy to receive command from the main controller board.

4.4. *Velocity and Altitude Estimator Implementation*

The horizontal position is obtained by integrating the horizontal velocity. The horizontal velocity of the vehicle is obtained by fusing the measurements from PX4Flow[32] camera[f] and IMU. The vertical position is directly measured by the ultrasonic sensor on PX4Flow camera. On the PX4Flow, the CMOS high speed vision sensor with 21 degree field of view, measures the optical flow at 100 Hz. Then it obtains the ground velocity relative to quadrotor by scaling the average optical flow by the ground distance. Moreover, by subtracting the scaled gyroscope rate, thus it compensates for the optical flow caused by roll and pitch rotation.

In particular, the PX4Flow camera measures the horizontal velocity, $\mathbf{v_c} = [v_{cx}, v_{cy}, 0]^\top$ and vertical position, h. Note that $\mathbf{v_c}$ is always measured with respect to vehicle heading, $proj[\mathbf{x_b}]$, thus the measured vehicle horizontal velocity $\mathbf{v'_w}$ is:

$$\mathbf{v'_w} = proj[\mathbf{x_b}] \odot \mathbf{v_c}, \tag{44}$$

where $\odot$ represents vector element-wise multiplication.

Then, given the IMU sampling time, ΔT, and measured body acceleration, $\mathbf{a}$, in body frame from accelerometer, we apply multiple rate complementary filter to compute the estimated vehicle horizontal velocity ${\mathbf{v_w}}^t = [v_{wx}, v_{wy}, v_{wr}]^\top$ at time t

$${\mathbf{v_w}}^t = n({\mathbf{v_w}}^{t-1} + \Delta T \mathbf{Ra}) + (1-n)\mathbf{v'_w}, \tag{45}$$

where the coefficient n can be computed by

$$n = \frac{\tau}{\tau + T}. \tag{46}$$

Thus, τ is the time constant for the complementary filter. Therefore the vehicle position $\mathbf{p_w}$ can be computed by

$$\dot{\mathbf{p_w}} = [v_{wx}, v_{wy}, \dot{h}]^\top. \tag{47}$$

The filter executes at IMU sampling frequency, and $\mathbf{v'_w}$ remains the latest measurement value before new velocity is updated from PX4Flow camera.

[c] `https://www.pjrc.com/teensy/teensy31.html`
[d] `http://www.varesano.net/projects/hardware/FreeIMU`
[e] `http://www.pololu.com/product/1352`
[f] `https://pixhawk.org/modules/px4flow`

4.5. *Components Cost Summary*

Table 3 summarises the cost of individual onboard components. The total cost of the entire system is indicated in the last row. Note that the cost will be significantly reduced with higher quantity production.

Table 3: Components cost summary with retail prices.

Item	Price ($)
Teensy 3.1 Main Controller	19.8
GF360 Frame	85.8
Turnigy 4500mah 3S Battery	49.9
DJI E310 Propulsion System	223.0
FreeIMU 4.0.3	59.6
Pololu Maestro Servo Controller	51.8
Optical Flow Camera (PX4FLOW)	159.7
Voltage Regulator	15.6
Prototype PCB Manufacture	34.0
Total	**699.2**

5. Test Results

An indoor flight test was conducted as indicated in Fig. 10. The manual tuning was conducted in advance of this trial and the following tuning parameters in Table. 4 were used.

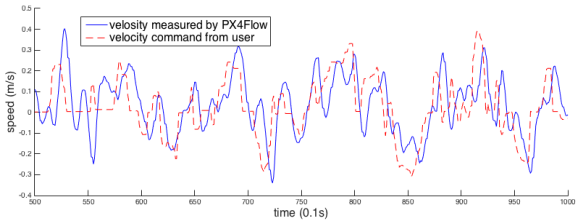
Table 4: PID parameters.

Controller	x-axis	y-axis	z-axis
$\mathbf{k_p}$	2200	2200	15
$\mathbf{k_d}$	460	460	10
$\mathbf{k_i'}$	0.7	0.7	1.4
$\mathbf{k_p'}$	3.2	3.2	2.3
$\mathbf{k_a'}$	0.45	0.45	0.5
k_h'	–	–	0.7

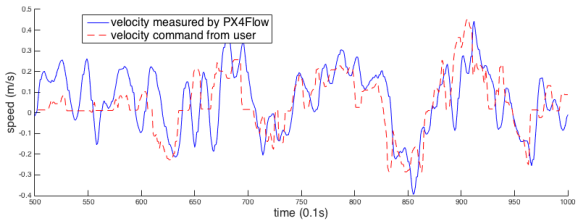
Flight data was recorded to demonstrate the control performance and validate the theory. Fig. 11 and Fig. 12 show the command-response graphs of terms directly controlled by the user, including horizontal velocity, altitude and heading angle, over 50 seconds duration (from 50s to 100s). Fig. 11a and Fig. 11b shows the horizontal velocity can be effectively controlled within ± 0.2 m/s accuracy and 0.5 s response time. Due to the position feedback error introduced in velocity controller in (40), the steady state velocity error was completely removed, since the position feedback error acts as the integral of the velocity feedback error. Moreover, it acts as the position hold effect when encounter an external turbulence. Fig. 12a shows that the altitude is controlled within ± 0.15 m accuracy. Fig. 12b shows the heading angle is controlled within ± 0.1 rad with a small steady-state error as a result of the PD attitude controller.

Fig. 10: Indoor flight test scene.

Moreover, to verify the cascaded control architecture, Two 28-second command-response graphs are shown in Fig. 13 (122s to 150s), indicating the velocity controller performance and the intermediate control signal between acceleration controller and attitude controller at the corresponding time. Note that we have rotated the velocity in the world frame to match the quadrotor heading so that the pitch angle of the quadrotor will result in the change in x-axis velocity change in match-heading frame.

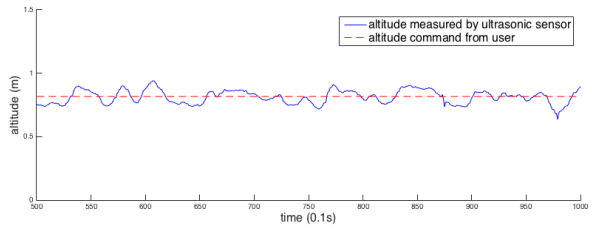


(a) Command-response graph of velocity in x-axis in the world frame.

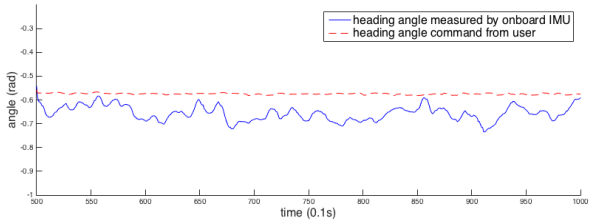


(b) Command-response graph of velocity in y-axis in the world frame.

Fig. 11: Velocity control performance evaluation graphs.

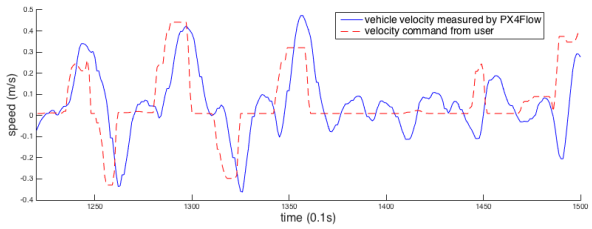


(a) Command-response graph of altitude.

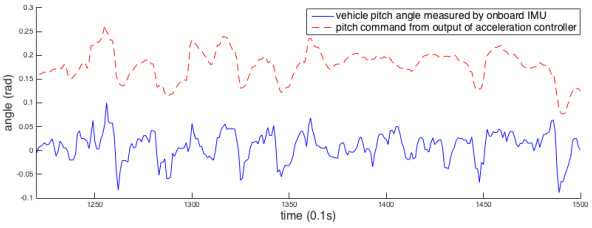


(b) Command-response graph of the heading angle.

Fig. 12: Altitude and heading control performance evaluation graphs.



(a) Short-term command-response graph of velocity in x-axis in match-heading frame.



(b) Short-term command-response graph of pitch angle.

Fig. 13: Cascaded control validation graphs.

It is clearly shown that the output from the acceleration controller (dashed in Fig. 13b reacts to the velocity error (indicated as the difference between dashed and solid in Fig. 13a), and acts as the command input to the attitude controller, although the output from acceleration controller also reacts to the acceleration measured directly by the accelerometer, which is not shown in the graph. Moreover, Fig. 13b also shows that there is significant offset (steady-state error) as expected from the PD attitude controller design with an imperfect mass balance of quadrotor body, which has been sufficiently compensated by the higher level velocity controller. It is indicated by the resulting pitch angle (solid in Fig. 13b centred at 0 *rad*.

6. Conclusion and Future Work

This paper has shown the quadrotor modelling and controller design principle, as well as implementation details. The flight test result showed a good attitude and altitude hold and an acceptable velocity control performance. The cascaded control architecture of the developed quadrotor is suitable for testing vision based localization algorithms, and testing new control strategies. The fully customised design makes it easy to integrate new sensors and manipulating controller.

Future work includes: implement more sophisticated error vector expression for attitude controller; design and manufacture customised printed circuit board (PCB) to interface all the onboard components; develop and test vision based localization solutions.

Acknowledgement

I would like to thank Mantas Brazinskas and Mehmet Ali Erbil for their continued technical support and encouragement. I offer my sincere appreciation for the learning opportunities provided by them.

References

[1] F. Kendoul, Survey of advances in guidance, navigation, and control of unmanned rotorcraft systems, *Journal of Field Robotics* **29** (March 2012) 315–378.

[2] A. Bachrach, A. De Winter, R. He, G. Hemann, S. Prentice and N. Roy, RANGE - Robust autonomous navigation in GPS-denied environments, *Proceedings - IEEE International Conference on Robotics and Automation* (2010) 1096–1097.

[3] A. Bry, A. Bachrach and N. Roy, State estimation for aggressive flight in GPS-denied environments using onboard sensing, *Proceedings - IEEE International Conference on Robotics and Automation* (May 2012) 1–8.

[4] J. Engel, J. Sturm and D. Cremers, Camera-based navigation of a low-cost quadrocopter, *IEEE International Conference on Intelligent Robots and Systems* (October 2012) 2815–2821.

[5] S. Shen, Y. Mulgaonkar, N. Michael and V. Kumar, Vision-based state estimation and trajectory control towards high-speed flight with a quadrotor, *Robotics: Science and Systems* (2013).

[6] E. S. Jones and S. Soatto, Visual-inertial navigation, mapping and localization: A scalable real-time causal approach, *The International Journal of Robotics Research* (2011) 1–38.

[7] M. W. Achtelik, S. Lynen, S. Weiss, L. Kneip, M. Chli and R. Siegwart, Visual-inertial SLAM for a small helicopter in large outdoor environments, *IEEE International Conference on Intelligent Robots and Systems* (October 2012) 2651–2652.

[8] R. Mahony, V. Kumar and P. Corke, Multirotor Aerial Vehicles: Modeling, Estimation, and Control of Quadrotor, *IEEE Robotics & Automation Magazine* **19**(3) (2012) 20–32.

[9] G. Klein and D. Murray, Parallel tracking and mapping for small AR workspaces, *2007 6th IEEE and ACM International Symposium on Mixed and Augmented Reality, ISMAR* (November 2007) 1–10.

[10] J. Engel, T. Sch and D. Cremers, LSD-SLAM: Large-Scale Direct Monocular SLAM, *Eccv* (2014) 1–16.

[11] C. Forster, M. Pizzoli and D. Scaramuzza, SVO : Fast Semi-Direct Monocular Visual Odometry, *IEEE International Conference on Robotics and Automation (ICRA)* (2014).

[12] M. Pizzoli, C. Forster and D. Scaramuzza, REMODE : Probabilistic , Monocular Dense Reconstruction in Real Time, *Proc. IEEE International Conference on Robotics and Automation (ICRA)* (2014).

[13] R. A. Newcombe, S. J. Lovegrove and A. J. Davison, DTAM: Dense tracking and mapping in real-time, *Proceedings of the IEEE International Conference on Computer Vision* (November 2011) 2320–2327.

[14] M. Montemerlo, S. Thrun, D. Roller and B. Wegbreit, FastSLAM 2.0: An improved particle filtering algorithm for simultaneous localization and mapping that provably converges, *IJCAI International Joint Conference on Artificial Intelligence* **18** (2003) 1151–1156.

[15] J. Engel, J. Sturm and D. Cremers, Semi-dense visual odometry for a monocular camera, *Proceedings of the IEEE International Conference on Computer Vision* (2013) 1449–1456.

[16] J. Kelly and G. S. Sukhatme, Visual-inertial simultaneous localization, mapping and sensor-to-sensor self-calibration, *Proceedings of IEEE International Symposium on Computational Intelligence in Robotics and Automation, CIRA* (December 2009) 360–368.

[17] J. Kelly and G. S. Sukhatme, Visual-Inertial Sensor Fusion: Localiation, Mapping and Sensor-to-Sensor Self-Calibration, *The International Journal of Robotics Research* (2010).

[18] S. Lynen, M. W. Achtelik, S. Weiss, M. Chli and R. Siegwart, A robust and modular multi-sensor fusion approach applied to MAV navigation, *IEEE International Conference on Intelligent Robots and Systems* (November 2013) 3923–3929.

[19] J. Lobo and J. Dias, Vision and Inertial Sensor Cooperation Using Gravity as a Vertical Reference, *IEEE Transactions on Pattern Analysis and Machine Intelligence* **25**(12) (2003) 1597–1608.

[20] O. Dunkley, J. Engel, J. Sturm and D. Cremers, Visual-Inertial Navigation for a Camera-Equipped 25g Nano-Quadrotor, *IROS2014 Aerial Open Source Robotics Workshop* (2014) p. 2.

[21] M. Li and a. I. Mourikis, High-precision, consistent EKF-based visual-inertial odometry, *The International Journal of Robotics Research* **32** (June 2013) 690–711.

[22] S. Weiss, M. W. Achtelik, M. Chli and R. Siegwart, Versatile distributed pose estimation and sensor self-calibration for an autonomous MAV, *Proceedings - IEEE International Conference on Robotics and Automation* (May 2012) 31–38.

[23] S. Shen, Y. Mulgaonkar, N. Michael and V. Kumar, Vision-based state estimation for autonomous rotorcraft MAVs in complex environments, *Proceedings - IEEE International Conference on Robotics and Automation* (May 2013) 1758–1764.

[24] A. Kushleyev, D. Mellinger, C. Powers and V. Kumar, Towards a swarm of agile micro quadrotors, *Autonomous Robots* **35** (July 2013) 287–300.

[25] S. Jeong and S. Jung, Design, Control, and Implementation of Small Quad-Rotor System Under Practical Limitation of Cost Effectiveness, *International Journal of Fuzzy Logic and Intelligent Systems* **13** (December 2013) 324–335.

[26] H. Fernando, A. De Silva, M. De Zoysa, K. Dilshan and S. R. Munasinghe, Modelling, simulation and implementation of a quadrotor UAV, *2013 IEEE 8th International Conference on Industrial and Information Systems, ICIIS 2013 - Conference Proceedings* (2013) 207–212.

[27] M. Elsamanty, a. Khalifa, M. Fanni, a. Ramadan and a. Abo-Ismail, Methodology for identifying quadrotor parameters, attitude estimation and control, *2013 IEEE/ASME International Conference on Advanced Intelligent Mechatronics: Mechatronics for Human Wellbeing, AIM 2013* (July 2013) 1343–1348.

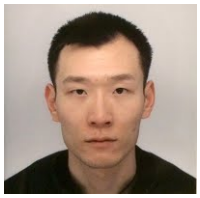
[28] C. Liu and S. D. Prior, Design and Implementation of a Mini Quadrotor Control System in GPS Denied Environments, *2015 International Conference on Unmanned Aircraft Systems, ICUAS 2015 - Conference Proceedings* (2015) 462–469.

[29] S. K. Phang, K. Li, K. H. Yu, B. M. Chen and T. H. Lee, Systematic Design and Implementation of a Micro Unmanned Quadrotor System, *Unmanned Systems* **02** (April 2014) 121–141.

[30] T. Lee, M. Leok and N. H. McClamroch, Control of Complex Maneuvers for a Quadrotor UAV using Geometric Methods on SE(3), *arXiv preprint arXiv:1003.2005* (March 2010) p. 8.

[31] F. Varesano, FreeIMU: An Open Hardware Framework for Orientation and Motion Sensing, *arXiv preprint* (2013).

[32] D. Honegger, L. Meier, P. Tanskanen and M. Pollefeys, An open source and open hardware embedded metric optical flow CMOS camera for indoor and outdoor applications, *Proceedings - IEEE International Conference on Robotics and Automation* (2013) 1736–1741.



Chang Liu started his PhD investigation into 'Visual control strategies for small remotely piloted aircraft (RPA) operations in GPS denied environments' at the University of Southampton in October 2013. Before that, he obtained a Masters Degree in Artificial Intelligence from the same University in 2012, and a Bachelor's Degree in Electronic Engineering from Newcastle University in 2011.

In the one-year gap between his Masters and the PhD, he undertook a 6-month placement at Kite Power Solutions as an embedded software engineer, helping with the research on autonomous flight of the surf kite to generate electric power.

Dr Stephen D. Prior has been working in the area of Field Robotics for the past 25 years. His research interest in autonomous systems relates to a shortlisted entry to the MoD Grand Challenge event in August 2008, where he led a team to design, make and test a novel unmanned aerial vehicle, which consisted of a patented Y6 arrangement. On the basis of this, he founded the Autonomous Systems Lab and has been researching with a small team of staff/students working on defence-related robotic technologies. He is on the editorial board for the International Journal of Micro Air Vehicles and has published widely on the subject. Recent work involved the design and development of a series of Nanotechnology platforms, which were demonstrated and flown at the DSEI exhibition at the Excel Centre in London (September 2011), as well as developing the winning entry to the DARPA UAVForge challenge 2012. During the last two years he has been developing a novel tethered UAS solution for persistent stare capability.

Prof Jim P. Scanlan received a materials science degree from Manchester University in 1977. He then spent over 12 years in the aerospace industry in a variety of roles, his final post being Head of Manufacturing Research at BAe regional aircraft. Whilst at BAe, Jim studied for an MSc at Salford University in aerospace design, and was sponsored by British Aerospace to study for a year at Cranfield University on a manufacturing management fellowship programme.

Jim joined the University of the West of England in 1990 and completed a PhD in 1995 in computer modelling of the Aerospace Design Process. Jim was appointed to Reader in 2004. In September 2004 Jim accepted a post at Southampton University as director of design within the Computational Engineering Design Centre (CEDC).

Jim manages a number of research programmes sponsored by BAE systems, Airbus, Rolls-Royce and the EPSRC. He has a particular interest in Design, Logistics, Simulation and Optimisation of organisations. He runs a number of MSc courses in these disciplines as part of the Aerospace IGDS MSc programme. Jim has recently created a spin-off business aimed at exploiting research into design process modelling.