Flexible Flight Navigation for Quadcopters Based on Geometric Formation Using Motion Sensing

Sonomi Nagata† Isao Miyagawa†† Kazuhito Murakami†
†Aichi Prefectural University, Aichi, Japan
E-mail: is141053@cis.aichi-pu.ac.jp, murakami@ist.aichi-pu.ac.jp
†† NTT Media Intelligence Laboratories, NTT Corporation, Kanagawa, Japan
E-mail: miyagawa.isao@lab.ntt.co.jp

Abstract - **We propose a flexible flight navigation method for quadcopters; it offers stable and safe control based on the geometric constraint formed between markers on the quadcopter and target markers. The 3D motions of the target markers simulate another quadcopter. Our method uses a motion capture system to extract the 3D positions of the markers. According to the detected 3D positions, it calculates translation motion and rotation motion to navigate the quadcopter via feedback control so that these markers form a rectangle in 3D space. We demonstrate flight trials by a prototype based on an AR.Drone and a motion capture system. Our experiments confirm that our method offers good flight navigation for quadcopter.**

Keywords - Flight navigation, quadcopter, feedback control, motion capture system, geometric formation.

I. INTRODUCTION

Unmanned aerial vehicles provide bird's-eye overviews, infrastructure inspections, distribution of goods, and disaster relief due to their many flight capabilities. Quadcopters are the dominant type of the unmanned aerial vehicle, and their four motor/propeller assemblies generate lift force and torque. By controlling the lift forces produced by the four propellers, they can fly forward/backward, horizontally, and vertically. Previous works [1][2][3] used Newton's laws subjected to mass and inertial moment to create dynamic equations that described the 3D motion of quadcopter. In order to fly a quadcopter with complete control to suit the many applications envisioned, we need to accurately estimate the self localization in sequence and adaptively control the flight motion. Since the dynamic model is an under-actuated system, backstepping control techniques are introduced in the dynamic model to fly quadcopter according to desired motion trajectory [2][3].

Nowadays, quadcopters are very common as drones. Most commercially available drones are operated by a smartphone or specialized controller. Some software platforms for drone control are available. For instance, AR.Drone provided by Parrot is one of the more popular quadcopters because AR.Drone 2.0 SDK makes it easy to develop various augmented reality displays or game applications based on AR.Drone [4]. Although the inputs and outputs of AR.Drone are open to the public for flight commands, the internal software implemented in the quadrotor is the black-box type and the parameters that refer to motion control, motors, and sensors are undocumented. Apparently, the input data do not follow the dynamic model used in previous works [1][2][3]. To analyze the internal behaviors, the input/output responses are proposed to be non-linear transfer functions [5]. Instead of the dynamic model, an extended Kalman filter with simple kinematic model is introduced to obtain state estimation in the non-linear system [6]. Assuming that the transfer functions designed in the system can be regarded as linear time invariant, a particle swarm optimization algorithm has been applied to flight control based on stochastic optimization [7].

It is well known that AR.Drone has accelerometers, gyroscopes, ultrasonic sensor, pressure sensor built into the inertial measurement unit, and front/bottom cameras. Some flight navigation methods combine the inertial sensors with computer vision approaches [8][9][10]. Even though AR.Drone outputs the velocities on longitudinal and transversal axes, altitude, and orientation, their accuracies are poor due to the use of low-cost devices. Although most quadcopters support flight motion by using internal sensing, it is not evident that the intrinsic sensors can achieve stable and robust autonomous flight to match a given 3D trajectory. The navigation approaches that depend on the inertial sensors focus on solo autonomous flight. If we assume that there is need for collision avoidance in response to multiple quadcopters, existing methods cannot provide safe formation flight.

Our motivation is to avoid collisions with other quadcopters or aircrafts and realize safe flight navigation for quadcopters. In this study, we assume that the 3D coordinates and orientations given by a pointer simulate the flight state of another quadcopter. We attach 3D sensing markers to the quadcopter and the pointer, and then locate these markers by employing a motion capture system. In order to control the quadcopter flight to respond to the pointer's movement, our approach ensures that the markers mounted on the pointer and quadcopter form a geometric shape, e.g. rectangle, in 3D space. We demonstrate flight trials by a prototype based on an AR.Drone and a motion capture system. Our experiments confirm that our proposal offers stable and safe control based on the geometric constraint formed by the markers.

This paper is organized as follows: Section II describes the flight motion by satisfying geometric formation with 3D motion sensing. Based on the flight motion, we introduce our flight navigation for AR.Drone control. Section III conducts flight trials that demonstrate our approach. Finally, Section IV concludes this study and mentions future work.

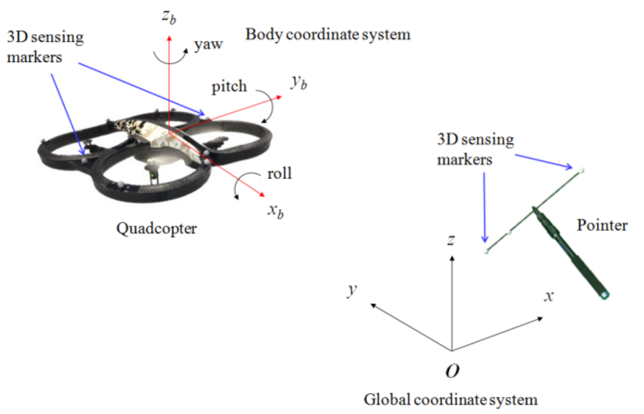


Fig. 1 Global and body coordinate systems.

II. Proposed Method

Fig. 1 shows the global coordinate system used in motion sensing and the body coordinate system adopted by the quadcopter. We assume that pointer movement replicates the 3D motion of the master quadcopter. We locate the world coordinate system (xyz) on the floor as determined by the motion capture system. Quadcopters generally have a body coordinate system ($x_b y_b z_b$) designed for internal sensing and flight command. We assume that the x_b-axis corresponds to forward direction. We attach 3D sensing markers, indicated by the blue arrows, to the quadcopter and obtain 3D coordinates of the markers by handling the motion capture system. In similar way, we extract the 3D coordinates of the sensing markers mounted on the pointer.

Fig. 2 illustrates the detected markers in the xy-plane. We use xy-translation for setup shown in Fig. 1, and take one point on the pointer as origin $\boldsymbol{O}$. Fig. 2 illustrates two target markers: $\boldsymbol{O}$ and $\mathbf{p}$ on pointer (small red circles) and two markers: $\mathbf{q}_1$ and $\mathbf{q}_2$ mounted on quadcopter (small blue circles) in the xy-coordinate system. We consider that the x_b-axis defined in the body faces the pointer. We assume that the z-coordinates of points $\boldsymbol{O}$ and $\mathbf{p}$ are given by p_z, which differ from the z-coordinates of points $\mathbf{q}_1$ and $\mathbf{q}_2$ at the initial state.

As this study uses a rectangle as the geometric constraint, we describe our flight navigation method using $L_p = L_q$ and $L_p \neq L$. We determine the translation motion and rotation motion that the quadcopter must perform to establish the desired rectangle. Given 3D coordinates $\mathbf{p} = [\, p_x, p_y, p_z \,]^T$, we obtain orientation angle ω by calculating $\tan^{-1}(p_y \,/\, p_x)$, $-\pi/2 < \omega < \pi/2$. We then estimate the desired points $\mathbf{p}_1$ and $\mathbf{p}_2$ as follows,

$$\mathbf{p}_1 = [\, -L\sin(\omega),\, L\cos(\omega),\, p_z \,]^T, \tag{1}$$

$$\mathbf{p}_2 = [\, L_p\cos(\omega) - L\sin(\omega),\, L_p\sin(\omega)+L\cos(\omega),\, p_z \,]^T. \tag{2}$$

These are the goals for points $\mathbf{q}_1$ and $\mathbf{q}_2$. In the xy-coordinate system, the four points: $\boldsymbol{O}$, $\mathbf{p}$, $\mathbf{p}_2$, and $\mathbf{p}_1$ form a rectangle. Using current midpoint $\mathbf{q}_m = (\mathbf{q}_1 + \mathbf{q}_2) / 2$ and the goal's midpoint $\mathbf{p}_m = (\mathbf{p}_1 + \mathbf{p}_2) / 2$, we obtain translation vector $\mathbf{T}$ by

$$\mathbf{T} = \mathbf{p}_m - \mathbf{q}_m, \tag{3}$$

which makes the midpoint $\mathbf{q}_m$ arrives at the goal $\mathbf{p}_m$. Next, we calculate rotational angle θ, i.e. the desired yaw rotation around the z_b-axis. Fig. 2(a) shows that $\mathbf{q}'_1 = \mathbf{q}_1 + \mathbf{T}$ and $\mathbf{q}'_2 = \mathbf{q}_2 + \mathbf{T}$. The inner-product produced by two unit vectors: $\mathbf{d}_p = (\mathbf{p}_2 - \mathbf{p}_m) \,/\, \|\mathbf{p}_2 - \mathbf{p}_m\|$ and $\mathbf{d}_q = (\mathbf{q}'_2 - \mathbf{p}_m) \,/\, \|\mathbf{q}'_2 - \mathbf{p}_m\|$ yields the yaw angle,

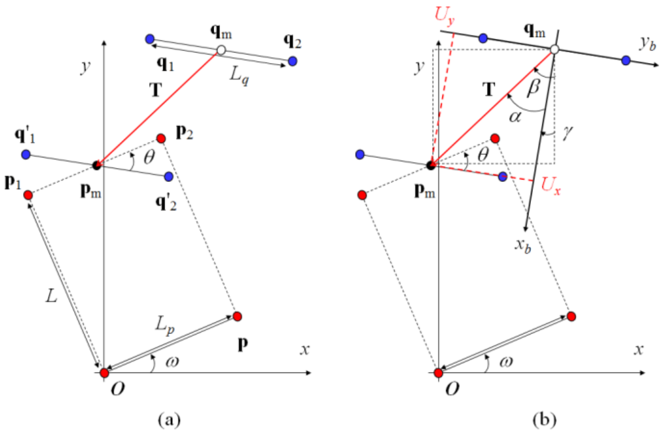


Fig. 2 Illustration of geometric formation.
(a) xy-coordinates. (b) Proper translation in $x_b y_b$-coordinates.

$$\theta = \cos^{-1}(\, \mathbf{d}^T_p \, \mathbf{d}_q \,), \tag{4}$$

within the range $-\pi/2 < \theta < \pi/2$. We consider positive or negative sign of rotational angle θ in Eq. (4) by introducing the outer-product of vectors $\mathbf{d}_p$ and $\mathbf{d}_q$.

Given translation vector $\mathbf{T} = [\, T_x, T_y, T_z \,]^T$ and rotation angle θ, we estimate the proper translation components, U_x and U_y, in the body coordinate system. As shown in Fig. 2(b), we assume that vector $\mathbf{T}$ on the xy-plane and x_b-axis direction vector form the angle of α. We calculate the angle by determining

$$\alpha = \beta - \gamma = \tan^{-1}(T_x \,/\, T_y) - \theta + \omega, \quad -\pi < \alpha \leq \pi\,, \tag{5}$$

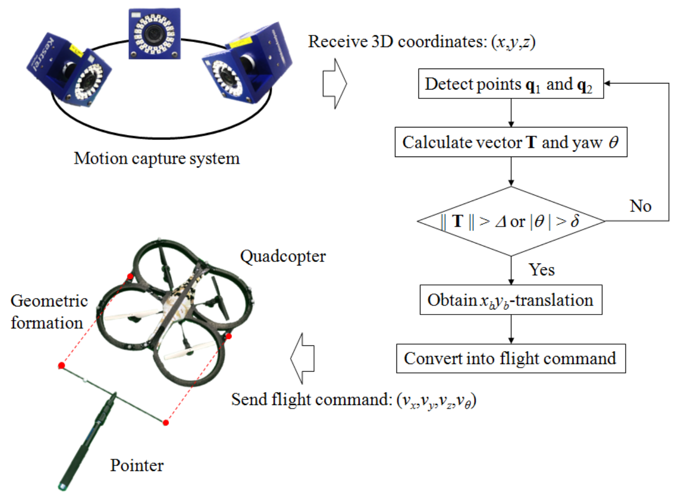
in the geometric constraint. We have proper translation components U_x and U_y along x_b-axis and y_b-axis as follows,

$$U_x = (T^2_x + T^2_y)^{1/2} \cos(\alpha\,), \quad \mathrm{U_y} = -\,(T^2_x + T^2_y)^{1/2} \sin(\alpha\,). \tag{6}$$

As our prototype uses an AR.Drone, so we convert the proper translation components into command data in accordance with AR.Drone SDK specifications. The flight commands that are sent to AR.Drone are encoded under a specific protocol via Wi-Fi link [4]. In this protocol, the command signals are normalized and arranged as elements of control vector $\mathbf{u}$ as follows,

$$\mathbf{u} = [\, v_x, v_y, v_z, v_\theta \,]^T \in \{\, -\, 1.0, +1.0\}, \tag{7}$$

where v_x represents inclination command (pitch) related to the y-axis; it indirectly indicates linear velocity command along x_b-axis, v_y represents inclination command (roll) related to the x-axis; it indirectly indicates linear velocity command along y_b-axis, v_z represents linear velocity command that yields translation along z_b-axis, and v_θ represents angular velocity command that yields rotation (yaw) around z-axis. Based on Eq. (7), we convert the $x_b y_b$-translation, attitude, and yaw rota-



Receive 3D coordinates: (x,y,z)
Detect points $\mathbf{q}_1$ and $\mathbf{q}_2$
Calculate vector $\mathbf{T}$ and yaw θ
$\|\mathbf{T}\| > \Delta$ or $|\theta| > \delta$
No
Yes
Obtain $x_b y_b$-translation
Convert into flight command
Send flight command: $(v_x, v_y, v_z, v_\theta)$
Motion capture system
Quadcopter
Geometric formation
Pointer

Fig. 3 Processing flow for flight navigation.

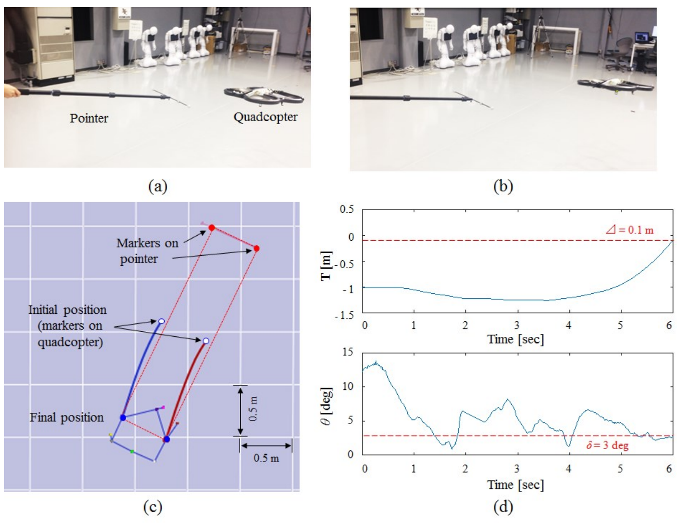


Fig. 4 Flight navigation when moving the pointer closer. (a) Initial. (b) Final. (c) Trajectory. (d) Variation of motion.

tion into the elements of the flight command vector $\mathbf{u}$ by

$$v_x = \kappa_x U_x, \quad v_y = \kappa_y U_y, \quad v_z = \kappa_z T_z, \quad v_\theta = \kappa_\theta \theta, \tag{8}$$

where κ_x, κ_y, κ_z, and κ_θ are normalized coefficients used in our system. After our method obtains the flight command vector $\mathbf{u}$, it is immediately sent to the AR.Drone via Wi-Fi.

The quadcopter is not likely to reach the desired position with just one flight command due to the non-linear input/output responses of the AR.Drone in flight [5]. In order to approach the goal step-by-step, we introduce feedback control for flight navigation. Fig. 3 shows a diagram of the processing flow. Given that points $\mathbf{q}_1$ and $\mathbf{q}_2$, which represent the 3D position and orientation of the quadcopter, are sequentially detected by the motion capture system, our method updates the translation and rotation motions by using Eqs. (3) and (4). Next, it calculates the norm of vector $\mathbf{T}$ which indicates the distance between the current position and the goal, and absolute value of yaw rotation θ. Given permissible errors Δ and δ, our approach sends the flight command data yielded by Eq. (8) to the quadcopter when $\|\mathbf{T}\| > \Delta$ or $|\theta| > \delta$. Until both $\|\mathbf{T}\|$ and $|\theta|$ are less than or equal to the permissible errors, feedback control is iterated. In the final state, the two points on the quadcopter approximately match the vertexes of the rectangle formed in 3D space. Thus, our proposed method provides flexible flight navigation where one quadcopter can be slaved to another.

Our approach is not restricted to flight navigation based on rectangle formation in 3D space because it is easy to extend the computation of translation vector $\mathbf{T}$ and rotation angle θ to other geometric constraints. For instance, when $L_p = L_q = L$, the four points form a square. If length L_p differs from length L_q, our navigation method forms an isosceles trapezoid on the xy-plane.

III. Experiments and Results

We employed a Parrot AR.Drone 2.0 as the quadcopter and Cortex 6.2 produced by Motion Analysis as the motion capture system. We installed ten motion capture cameras in our laboratory. The 3D sensing markers are made of the same shape and same material. The motion capture system accurately detected 3D coordinates: (x, y, z) of the four points at intervals of 1/60 seconds. Distances L_p and L_q were 0.5 meters and the distance L was 2.0 meters. In Fig. 1, there are other markers that are not indicated by blue arrows. They were asymmetrically set on the quadcopter and used to identify the four points. In our experimental system, we used permissible errors $\Delta = 0.1$ meters and $\delta = 3.0$ degrees and feedback control was performed at intervals of 1/6 seconds in order to achieve stable navigation taking account of input/output response times. After takeoff, the quadcopter hovered at the initial state ($t = 0$). According to AR.Drone 2.0 specifications, hovering maintains the current 3D position and orientation by optical flow processing. We placed the pointer for flight navigation in front of the quadcopter, as shown in Fig. 4(a) and Fig. 5(a). Given all 3D coordinates in the system, our method promptly computed translation and rotation motions so the quadcopter moved in real time.

We conducted two trials to examine the performance of our proposed method. The first test demonstrated flight navigation by bringing the pointer toward the quadcopter; i.e. the master quadcopter approached the slave. Fig. 4(a) shows the initial position of the quadcopter with $\|\mathbf{T}\| = 1.0$ meters. Fig. 4(b) shows the final formation achieved by our method. Our method smoothly navigated the quadcopter as shown in Fig. 4(c). We confirmed that the four markers, two on the quadcopter and two on the pointer, formed almost the desired rectangle in 3D space. Fig. 4(d) plots the variations of translation $\mathbf{T}$ and yaw rotation θ, and it shows that our method took 6 seconds to form the desired rectangle. According to the AR.Drone specifications, the quadcopter hovered for a short while after takeoff. In order to separately demonstrate the effects of estimated translation and rotation motions, we controlled yaw rotation from initial state for 4 seconds, and then adjusted, simultaneously, translation and yaw rotation. Although our method yielded unstable behavior in the control of yaw rotation, the variation of translation was stable and smooth.

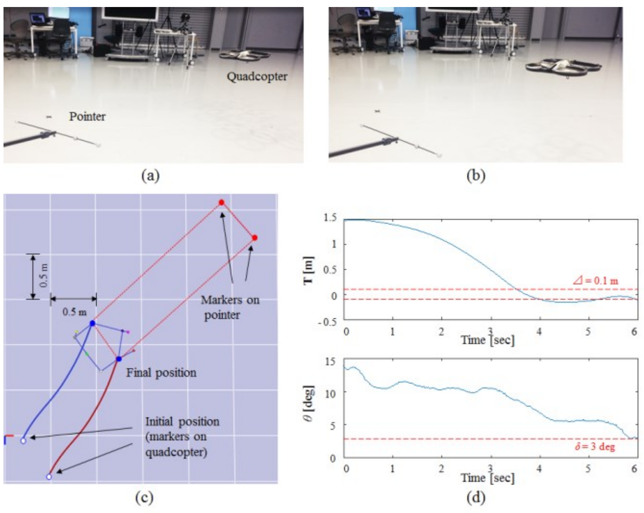


Fig. 5 Flight navigation when moving the pointer away.
(a) Initial. (b) Final. (c) Trajectory. (d) Variation of motion.

The second test demonstrated flight navigation when we moved the pointer away from quadcopter. Namely, the master quadcopter left the slave. Fig. 5(a) shows the initial position of the quadcopter with $\| \mathbf{T} \| = 1.5$ meters. In this experiment, we controlled translation and yaw rotation, simultaneously. Fig. 5(b) plots the final position yielded by our method. We show the flight trajectory from initial to final positions in Fig. 5(c); it also demonstrates that the four markers accurately formed the desired rectangle in the final state. Fig. 5(d) provides sequential variations of translation $\mathbf{T}$ and yaw rotation θ. After takeoff, the proposed method took 4 seconds to achieve the goal position. It took about 6 seconds to control the yaw rotation to under permissible error δ.

Since the above results show that the quadcopter was slaved to follow the pointer's movements, i.e. approach and separation, with perfect flight response, we confirmed that the geometric formation based on motion sensing works effectively for realizing flexible flight navigation for quadcopters. However, we found that the AR.Drone often drift when the elements of input flight command vector $\mathbf{u} = [\ v_x,\ v_y,\ v_z,\ v_\theta\ ]^{\mathrm{T}}$ are small. These results make it necessary for the feedback system to improve for achieving the goal as fast as possible. For instance, we consider that the normalized coefficients κ_{x}, κ_{y}, κ_{z}, and κ_θ are optimized in our system in order to adaptively complete the flight navigation given by Eq. (8). Instead of Eq. (8), it is also possible to design a suitable transformation scheme based on non-linear transfer functions between input and output responses for the quadcopter system [5]. Given various flight commands, we need to investigate the dynamic behaviors of quadcopter as detected by the motion capture system in detail.

IV. Conclusion

We assumed that the 3D movements of the pointer simulated the flight motion of another quadcopter. Our solution drove the quadcopter to respond to the pointer's movement. By employing a motion capture system, we located 3D sensing markers attached to the quadcopter and the target pointer. Given the 3D coordinates, our approach controlled the flight of the quadcopter by using translation and rotation motions estimated from a given geometric constraint. The method ensures that these markers form the desired geometric shape by using motion sensing. The proposal offers stable and safe flight navigation for quadcopter by maintaining the geometric formation in 3D space.

We demonstrated flight trials of a prototype based on an AR.Drone and a motion capture system. Since our results showed that the quadcopter followed the pointer's movements with perfect flight response, we conclude that the geometric formation based on motion sensing is an effective way of achieving flexible flight navigation for quadcopters. In the future, we will investigate the dynamic behaviors of quadcopter with greater accuracy by using the motion capture system and design a suitable flight command scheme for real time control. Its applications include the linked flight of multiple quadcopters in an orderly manner.

References

[1] P. Castillo, A. Dzul, and R. Lozano: "Real-Time Stabilization and Tracking of a Four-Rotor Mini Rotorcraft", IEEE Transactions on Control and Systems Technology, Vol.12, No.4, pp.510-516, 2004.

[2] S. Bouabdallah and R. Siegwart: "Backstepping and Sliding-mode Techniques Applied to an Indoor Micro Quadrotor", Proc. IEEE International Conference on Robotics and Automation, pp.2247-2252, 2005.

[3] T. Madani and A. Benallegue: "Backstepping Control for a Quadrotor Helicopter", Proc. IEEE/RSJ International Conference on Intelligent Robots and Systems, 2006.

[4] T. Krajník, V. Vonásek, D. Fišer, and J. Faigl: "AR-Drone as a Platform for Robotic Research and Education", Research and Education in Robotics (EUROBOT 2011), pp.172-186, 2011.

[5] A. Hernandez, C. Copot, R. De Keyser, T. Vlas, and I. Nascu: "Identification and Path Following Control of an AR.Drone Quadrotor", Proc. International Conference on System Theory, Control and Computing, 2013.

[6] L. V. Santana, A. S. Brandão, M. S-Filho, and R. Carelli: "A Trajectory Tracking and 3D Positioning Controller for the AR.Drone Quadrotor", Proc. International Conference on Unmanned Aircraft Systems, pp.27-30, 2014.

[7] T. T. Mac, C. Copot, T. T. Duc, and R. De Keyser: "AR.Drone UAV control parameters tuning based on particle swarm optimization algorithm", Proc. IEEE International Conference on Automation, Quality and Testing, Robotics, 2016.

[8] P-J. Bristeau, F. Callou, D. Vissière, and N. Petit: "The Navigation and Control technology inside the AR.Drone micro UAV", Proc. IFAC World Congress, p.1477-1484, 2011.

[9] J. Engel, J. Sturm, and D. Cremers: "Accurate Figure Flying with a Quadrocopter Using Onboard Visual and Inertial Sensing", Proc. International Workshop on Visual Control of Mobile Robots, 2012.

[10] A. Garcia, E. Mattison, and K. Ghose: "High-speed Vision-based Autonomous Indoor Navigation of a Quadcopter", Proc. International Conference on Unmanned Aircraft Systems, 2015.