Control Engineering Practice 88 (2019) 128–140

Contents lists available at ScienceDirect

Control Engineering Practice

journal homepage: www.elsevier.com/locate/conengprac

Outdoor cooperative flight using decentralized consensus algorithm and a guaranteed real-time communication protocol☆

Apurva Joshi [a], Ankit Wala [b], Mohit Ludhiyani [c], Debraj Chakraborty [d,*], Hoam Chung [e], D. Manjunath [d]

[a] IITB-Monash Research Academy, Mumbai, India
[b] Agrahyah Technologies, Mumbai, India
[c] Tata Innovation Lab, Kolkata, India
[d] Dept. of Electrical Engineering, IIT Bombay, Mumbai, India
[e] Dept. of Mechanical and Aerospace Engineering, Monash University, Melbourne, Australia

ARTICLE INFO

Keywords:
Decentralized control
Unmanned aerial vehicles
Autonomous robots
Cooperative systems
Wireless networks

ABSTRACT

This paper describes the implementation of a decentralized consensus law with theoretically provable convergence properties on a multi-agent testbed comprising of quadrotors. It is shown that for small roll and pitch angles and well-tuned control loops, the quadrotor dynamics can be approximated as a pair of double integrators. Several experiments are carried out in an outdoor environment for validation of the consensus law which is based on double integrator dynamics. For any arbitrary initial positions of the quadrotors, the consensus law is able to drive them to an autonomously decided common point, given that the communication graph is connected at each instant of time. The resulting experimental trajectories and the consensus point matches with theoretical predictions. For guaranteeing real-time reliability required for such coordinated motion, a novel synchronized, time-slotted, scalable and fully airborne communication protocol is proposed. The protocol avoids data collisions and ensures real-time, reliable communication between agents. It can also address changing communication graph topologies, temporary link-breaks and additions. Using this underlying protocol, the quadrotors attain consensus for static and dynamic communication graphs. Experiments to observe the effect of communication rate on consensus performance are also conducted.

1. Introduction

Cooperative control of multi-agent systems has significant impact on several civilian and military applications (Wang & Zhang, 2017). Applications of cooperative control in the domain of flying robots include localization and rescue during calamities (Casbeer, Kingston, Beard, & McLain, 2006), estimation of traffic density (Puri, 2005), surveillance and intelligence gathering (Acevedo, Arrue, Maza, & Ollero, 2013), mapping (Michael et al., 2012), construction (Lindsey, Mellinger, & Kumar, 2011), etc. To achieve cooperation, agents take control actions based on exchange of information over a communication network (Abdessameud & Tayebi, 2013). Wireless communication between flying robots poses challenges like limited range, interference, packet drop, etc. (Guerrero, Garcia, & Challal, 2013). Hence it is critical to design appropriate distributed algorithms and communication protocols so that a group of flying robots can reach consensus. In this paper, for the first time, a theoretically provable decentralized consensus law with a scalable, fully airborne communication protocol is designed and experimentally verified in an outdoor environment.

Availability of cheap miniature embedded systems for computation, communication and sensing have made it possible to easily construct quadrotor helicopters (or quadrotors) (Lim, Park, Lee, & Kim, 2012). The quadrotor is a good benchmark for testing advanced control algorithms due to its nonlinear under-actuated dynamics and manoeuvrability in six degrees of freedom (Mahony, Kumar, & Corke, 2012). Experiments have been carried out with quadrotors to test various cooperative motion control experiments in both indoor (Lupashin et al., 2014; Michael, Mellinger, Lindsey, & Kumar, 2010) and outdoor (Kang, Park, & Ahn, 2017; Vásárhelyi et al., 2014) environments. In an indoor setup, motion capture systems have been used for localization which provide millimetre-range accuracy and update rates of around 200 Hz (Lupashin et al., 2014). Due to the controlled nature of the environment, the agents are not subjected to external effects such as wind gusts and interference over communication channels. On the

☆ This work was supported by the Science and Engineering Research Board, Dept. of Science and Technology, Govt. of India.
* Corresponding author.
E-mail addresses: apurvajoshi@iitb.ac.in (A. Joshi), ankit@agrahyah.com (A. Wala), mohit.ludhiyani@tcs.com (M. Ludhiyani), dc@ee.iitb.ac.in (D. Chakraborty), hoam.chung@monash.edu (H. Chung), dmanju@ee.iitb.ac.in (D. Manjunath).

https://doi.org/10.1016/j.conengprac.2019.05.002
Received 10 August 2018; Received in revised form 20 March 2019; Accepted 1 May 2019
Available online 20 May 2019

other hand, outdoor experiments rely on the use of GPS for localization which provides metre-range accuracy and update rates close to 5–10 Hz (Vásárhelyi et al., 2014). The agents in this setup are frequently exposed to external effects common to outdoors such as adverse weather conditions (Hoffmann, Huang, Waslander, & Tomlin, 2007) and interference over communication frequencies, that make exchange of information difficult.

On the theoretical front, substantial literature is available on the consensus of single and double integrator agents. Several algorithms have been developed in this framework to achieve consensus among agents autonomously (e.g. Olfati-Saber & Murray, 2004; Ren & Atkins, 2007; Ren & Beard, 2008 and references therein). However, literature for complex nonlinear agents is rather limited (Moreau, 2005). A brief survey of multi-quadrotor testbeds used for physical implementation of cooperative flight is as follows: In Kushleyev, Mellinger, Powers, and Kumar (2013), the authors describe a centralized control algorithm to demonstrate tight flight formations, while a decentralized control algorithm is implemented in Turpin, Michael, and Kumar (2012) to achieve linear and rectangular-shaped formation flight. In both cases, the actual execution of the algorithm takes place on a central computer. Formations are achieved in Preiss, Honig, Sukhatme, and Ayanian (2017) by generating piecewise polynomials which satisfy given waypoint and continuity constraints, but are computed and uploaded from the base station. Decentralized, consensus-based algorithms for double-integrator dynamics have been used to achieve formation control among quadrotors in Saif, Fantoni, and Zavala-Río (2015) and Dong, Yu, Shi, and Zhong (2015). However, all these experiments have been performed in a controlled indoor environment. Formation flight in outdoor conditions with two physical industrial grade unmanned helicopters, and seven simulated helicopters is reported in Shaw, Chung, Hedrick, and Sastry (2007). However, the control laws used are not decentralized in nature. Successful execution of decentralized multi-quadrotor flock which performs stable autonomous outdoor flight is given in Kang et al. (2017) and Vásárhelyi et al. (2014). While Vásárhelyi et al. (2014) uses an empirical control law, Kang et al. (2017) proves persistence of formation with a single integrator model. In the absence of low level access to the autopilot control loops in either of these papers, it is not clear how the proposed theoretical models are applicable to quadrotors.

The thrust of research in recent times has been towards development of control algorithms, wherein inter-agent communication is assumed to be perfect. A summary of the state of the art in practical multi-UAV communication is available in Mammadov and Gueaieb (2014). For communication in indoor environments, the agents keep sending data packets repeatedly to make up for unreliability in data reception (Lupashin et al., 2014). In Preiss et al. (2017), to achieve low latency, the authors aim for high probability of communication by repeating request–response commands until acknowledged or a timeout occurs without a mechanism to avoid data collisions. In outdoor experiments performed in Shaw et al. (2007), a token ring communication protocol is used for collision-free inter-agent communication. Packet loss during wireless communication was one of the contributing factors to the experimental results differing from the ideal theoretical results (Shaw et al., 2007). In Vásárhelyi et al. (2014), to achieve consensus among quadrotors, XBee modules working in broadcast mode, which use standard MAC protocols with randomized collision resolution, are used for information exchange. The unpredictable behaviour of such protocols lead to delays in data transmission and deterioration in convergence accuracy/rates which are unacceptable in real-time applications (Guerrero et al., 2013). A study of the effect of control gain, communication update rate and graph topologies performed in Michael, Schwager, Kumar, and Rus (2014) shows that timely dissemination of information and its reception are vital for consensus to occur among agents.

The contributions in this work are as follows:

(1) *Theoretically provable, experimentally verified consensus law*: In this paper, it is shown that using the low level access to quadrotor control loops, quadrotors can be approximated as double integrators. Thus, available theory for consensus of double integrator agents can be modified and applied to quadrotors. The proposed consensus law for double integrator dynamics, which is a variation of Ren and Atkins (2007) requires the exchange of only the position information among agents. It guarantees that the agents will asymptotically reach an autonomously decided consensus point, assuming the communication graph remains connected at each instant of time. The trajectory generation is done on-board and there is no requirement of a ground station except for telemetry purposes. In order to implement the consensus law, the quadrotor motion is decoupled along two orthogonal axes in the horizontal plane. The quadrotors are kept at different altitudes in order to avoid collisions. Experiments are performed to provide a comparison between the trajectories generated by double integrator agents and physical units, and to demonstrate successful consensus with six quadrotors. Preliminary experiments to test such coordinated motion among quadrotors have been reported in Joshi et al. (2016).

(2) *Implementation of a scalable airborne communication protocol*: A novel communication protocol is proposed for facilitating the real-time exchange of data between multiple agents. This protocol avoids data packet collisions using time-slotting. During initialization, the clocks of all the agents are synchronized and each agent transmits data in an allotted time slot. The protocol has the ability to maintain synchronization and handle changes in the communication topology due to: obstacles, or the motion of the agents entering and leaving each others' communication range. This protocol is implemented on-board the quadrotors and is fully airborne, thus eliminating th requirement for a ground station. Average transmission and reception efficiencies of 98.04% and 98.52% respectively are achieved for complete information exchange between six agents for 18 runs of 300 s in indoor and outdoor environments. Preliminary experiments for validation of the communication protocol have been reported in Joshi et al. (2017).

(3) *Effect of communication on control and convergence*: Experiments are performed wherein the rate at which data is exchanged between the agents using the proposed law to reach consensus, is altered. It is observed that slower communication rates progressively lead to degraded performance in reaching consensus.

The consensus experiments are performed on a testbed of three quadrotors which are physically constructed in-house with cheap off-the-shelf components. Three virtual quadrotor emulators are also added to the testbed to test the efficacy of the communication protocol for more number of agents. The three emulators each consist of a laptop computer which simulates dynamics of a single quadrotor and a communication module. The communication module which runs the communication protocol is the same for the physical and virtual units. An overview of the setup is as shown in Fig. 1.

The outline of the paper is as follows: The preliminaries are presented in Section 2, in Section 3 it is shown that quadrotors can be approximated as a pair of double integrators and the consensus law to be implemented is proposed. Details regarding the proposed communication protocol are provided in Section 4 and the experimental results are given in Section 5.

2. Preliminaries

Some frames of reference are defined in Section 2.1 to describe the motion of the multi-agent system comprising of quadrotors as agents. Sections 2.2 and 2.3 contain a brief description of the physical and virtual quadrotor units. The conversions between different reference frames is presented in Section 2.5.

2.1. Frames of reference

To analyse the motion of the multi-agent system comprising of quadrotors as agents, standard definitions (Noureldin, Karamat, & Georgy, 2012) for the frames of reference are used in the analysis (see Figs. 2 and 3).

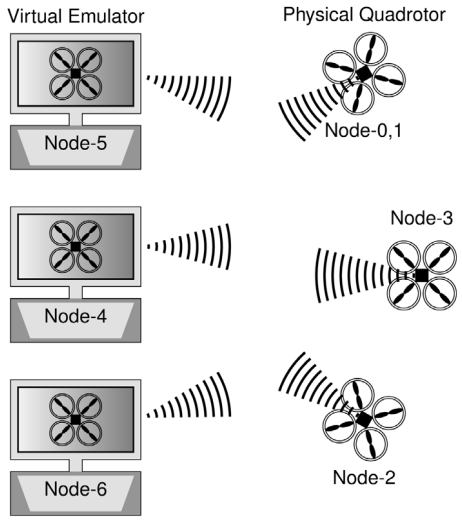

Fig. 1. The testbed comprising of three physical and three virtual units interacting with each other using a common communication protocol.

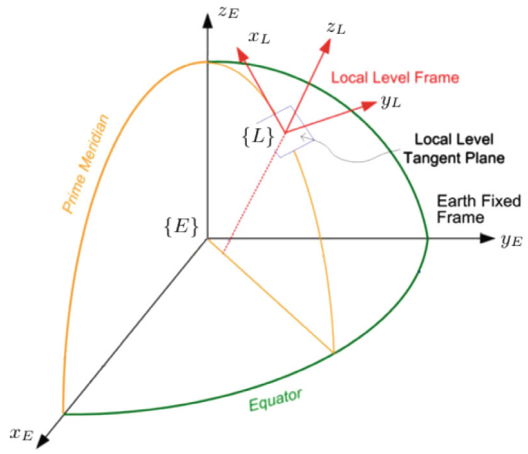

Fig. 2. A depiction of the $\{E\}$ and $\{L\}$ frames of reference.

2.1.1. Earth-centred earth-fixed frame, $\{E\}$

This is considered to be an inertial frame of reference and is defined as follows: (a) The origin is at the centre of mass of the Earth (b) The z_E-axis passes through the geographical north pole (c) The x_E-axis passes through the intersection of the equatorial plane and the Greenwich meridian (d) The y_E-axis completes the right-hand coordinate system.

2.1.2. Local-level frame, $\{L\}$

The local-level frame or the navigation frame is set up as follows: (a) The origin is at a fixed point on Earth (b) The x_L-axis points towards the north (c) The y_L-axis points towards the east (d) The z_L-axis points upwards.

2.1.3. Body frame, $\{B\}$

The body frame $\{B\}$ is oriented as follows: (a) The origin is at the centre of mass of the quadrotor (b) The x_B-axis points towards the front end of the quadrotor (c) The y_B-axis points towards the right end of the quadrotor (d) The z_B-axis points in the downward direction.

2.1.4. Body-fixed frame, $\{V\}$

A body-fixed coordinate frame $\{V\}$ is attached to the vehicle (Corke, 2011) such that (a) It has the same origin as $\{B\}$ (b) The z_V-axis is parallel to z_L-axis (c) x_V, y_V- axes are projections of x_B, y_B onto a plane parallel to the $x_L y_L$ plane in $\{E\}$ and passing through the origin of $\{V\}$.

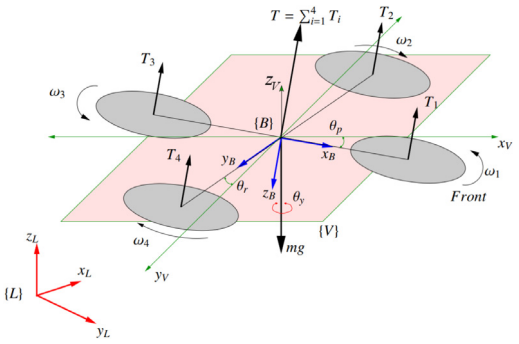

Fig. 3. The quadrotor system setup along with the $\{L\}, \{B\}$ and $\{V\}$ frames of reference.

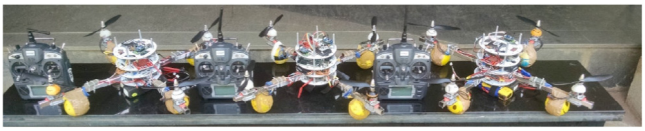

Fig. 4. A picture of the three physical quadrotor units.

The angles θ_p, θ_r and θ_y correspond to pitch, roll and yaw of the quadrotor respectively. The $x_V y_V$ plane is constrained to always remain parallel to the $x_L y_L$ plane as shown in Fig. 3. However, it can change its orientation by some θ_y about z_V. This change in orientation between $\{V\}$ and $\{L\}$ is given by a rotation matrix

$$\mathbf{R}_V^L = \begin{bmatrix} c\theta_y & s\theta_y & 0 \\ -s\theta_y & c\theta_y & 0 \\ 0 & 0 & 1 \end{bmatrix}, \tag{1}$$

where $c(\cdot) := \cos(\cdot)$ and $s(\cdot) := \sin(\cdot)$.

2.2. The physical unit

The physical quadrotors are constructed in-house using commercial off the shelf components as seen in Fig. 4. The system architecture is as shown in Fig. 5. Microcontroller-1 is the ATmega 2560 based APM 2.6 board (ArduPilot, 2015) which is used for flight control. Microcontroller-2 is an ATmega 2560 microcontroller which is used to orchestrate the communication between the multiple quadrotors. A u-blox LEA-6H GPS receiver provides localization information of the quadrotors in terms of latitudes and longitudes.

Each quadrotor is capable of manual and autonomous flight. The auto-pilot running onboard Microcontroller-1 is a modified version of the Aeroquad (2015) open-source autopilot. Details regarding the control loops are given in Section 3.

2.3. The virtual unit

Due to unavailability of resources to physically construct more quadrotors, three real-time virtual quadrotor emulators are created in addition to the physical units. Each emulator consists of a laptop computer running (ROS, 2017) and Gazebo (2017) platforms, used to simulate the dynamics of a single quadrotor. A pre-existing stack (Meyer, Sendobry, Kohlbrecher, Klingauf, & von Stryk, 2012) which provides ROS packages related to modelling, control and simulation of a quadrotor, is modified to generate real-time quadrotor trajectories. A communication module with the same hardware as on-board the physical unit, is interfaced with the laptop computer as shown in Fig. 6.

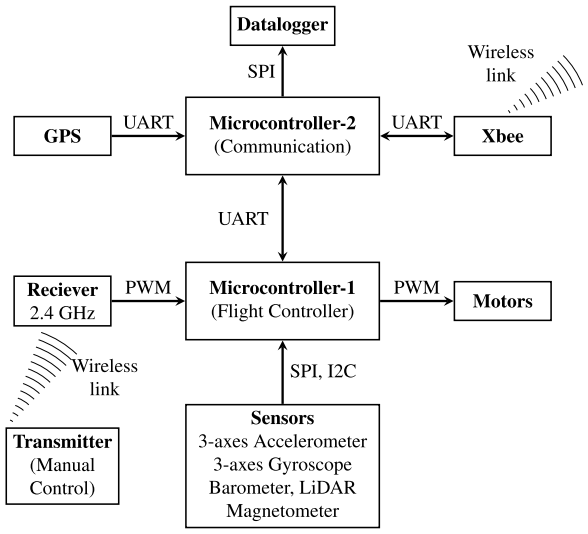

Fig. 5. A block diagram of the physical quadrotor unit.

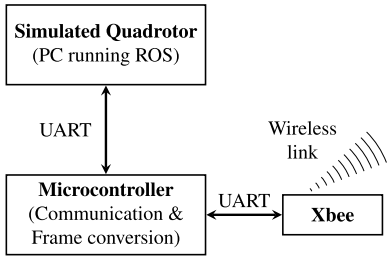

Fig. 6. A block diagram of the virtual quadrotor unit.

2.4. The communication module

The communication module facilitates the information exchange between all the agents, physical and virtual. The hardware comprises of a Xbee modem which is interfaced with an ATMega 2560 microcontroller as shown in Figs. 5 and 6. The microcontroller runs the communication protocol which is described in Section 4 and thus a wireless link is set up between the agents.

2.5. Combining the physical and virtual units

The motions of the physical and virtual units are inherently described in different coordinate systems. The physical units use geodetic coordinates (latitude, φ and longitude, λ) obtained from the GPS for localization and are usually converted to the navigation frame $\{L\}$. The motion of the virtual units in the simulator is described in a virtual reference frame, say $\{S\}$. To facilitate interaction between the physical and virtual units, equirectangular projections and the WGS84 system are used for transformation between these two coordinate systems, $\{L\}$ and $\{S\}$. A derivation of this transformation is given in Appendix A.

3. Quadrotors as double integrators

It is desired that the position, $\begin{bmatrix} p_x^E & p_y^E & p_z^E \end{bmatrix}^T \in \mathbb{R}^3$ of the quadrotor in $\{E\}$ and its heading angle, θ_y be controlled. This can be achieved using three independent control loops: (A) heading hold, (B) altitude hold and (C) waypoint navigation, described below. The control loops take desired values of state variables as inputs and generate values of the thrust and torques necessary to achieve the required pose. These thrust and torque values are converted to required rotor speeds (Corke, 2011). The desired rotor speeds are achieved by providing the on-board motors with the necessary pulse-width-modulated input using the on-board processor. The overall control architecture is shown in Fig. 7.

3.1. Heading hold

The heading of the quadrotor is the angle between x_B and x_L as seen in Fig. 3. This angle is represented by the yaw angle, θ_y. The heading of the quadrotor is kept constant at $\theta_y = 0$, in order to decouple the motion of the quadrotor along the roll and pitch axis. Hence, each quadrotor is aligned with respect to true north and manoeuvres in $x_L y_L$ plane by only changing its roll and pitch angles. The required yaw torque is generated using the following proportional-derivative (PD) control law

$$\tau_z = K_{p_y} e_H + K_{d_y} \dot{e}_H, \tag{2}$$

where $e_H = (\theta_y^* - \theta_y)$ is the error, θ_y is the measured yaw angle, θ_y^* is the desired yaw angle and K_{p_y}, K_{d_y} are the control gains.

3.2. Altitude hold

The altitude of the quadrotor is held constant using PD gain scheduling. Let p_z be the actual altitude, p_z^* the desired altitude and $\bar{T}$ the thrust value corresponding to p_z^*. Error bands $B_k := [b_k, b_{k+1})$, where $b_k \in \mathbb{R}$ and $b_0 < \cdots < b_k < \cdots < b_m$ are defined. The error $e_A = (p_z^* - p_z) \in B_k$ for some k. Different set of gains $K_p^{B_k}, K_d^{B_k}$ corresponding to each error band B_k are assigned. Then the required cumulative thrust correction is generated using

$$\Delta T = K_p^{B_k} e_A + K_d^{B_k} \dot{e}_A + \omega_0, \tag{3}$$

where ω_0 is the rotor speed bias required to produce thrust to counter the weight of the vehicle.

3.3. Waypoint navigation

Waypoint navigation is the process by which the quadrotor navigates to different points $\mathbf{p}^E = \begin{bmatrix} p_x^E & p_y^E \end{bmatrix}^T \in \mathbb{R}^2$ in $\{E\}$. The desired position is denoted by $\mathbf{p}^{E*} = \begin{bmatrix} p_x^{E*} & p_y^{E*} \end{bmatrix}^T$. This is achieved by varying θ_p and θ_r while θ_y is held constant by the heading control loop.

3.3.1. Approximation as double integrators

Motion of the quadrotor in the $x_V y_V$ plane is achieved by generating required rolling and pitching torques. The total force $\mathbf{f}^V = \begin{bmatrix} f_x^V & f_y^V & f_z^V \end{bmatrix}^T \in \mathbb{R}^3$ acting on the quadrotor is given by

$$\mathbf{f}^V = \mathbf{R}_x(\theta_r)\mathbf{R}_y(\theta_p) \begin{bmatrix} 0 & 0 & T \end{bmatrix}^T, \text{ where} \tag{4}$$

$$\mathbf{R}_x(\theta_r) = \begin{bmatrix} 1 & 0 & 0 \\ 0 & c\theta_r & s\theta_r \\ 0 & -s\theta_r & c\theta_r \end{bmatrix}, \mathbf{R}_y(\theta_p) = \begin{bmatrix} c\theta_p & 0 & s\theta_p \\ 0 & 1 & 0 \\ -s\theta_p & 0 & c\theta_p \end{bmatrix}.$$

For small θ_r and θ_p, (4) can be simplified to get

$$f_x^V = T \sin\theta_p \approx T\theta_p \tag{5}$$

$$f_y^V = T \sin\theta_r \cos\theta_p \approx T\theta_r. \tag{6}$$

In $\{E\}$, the total force acting on the quadrotor is given by

$$\mathbf{f}^E = \mathbf{R}_V^E \mathbf{f}^V. \tag{7}$$

Now, the dynamics of the quadrotor in the $x_E y_E$ plane is given by

$$\dot{\mathbf{p}}^E = \mathbf{v}^E, \qquad \dot{\mathbf{v}}^E = \mathbf{f}^E, \tag{8}$$

where $\mathbf{f}^E$ can be controlled through (7). To change $\mathbf{f}^E$ arbitrarily, θ_p and θ_r is adjusted according to (7). If θ_p and θ_r can be varied independently and instantaneously, then due to (7) and (8), the effective motion of the quadrotor in the $x_E y_E$ plane can be modelled as a double

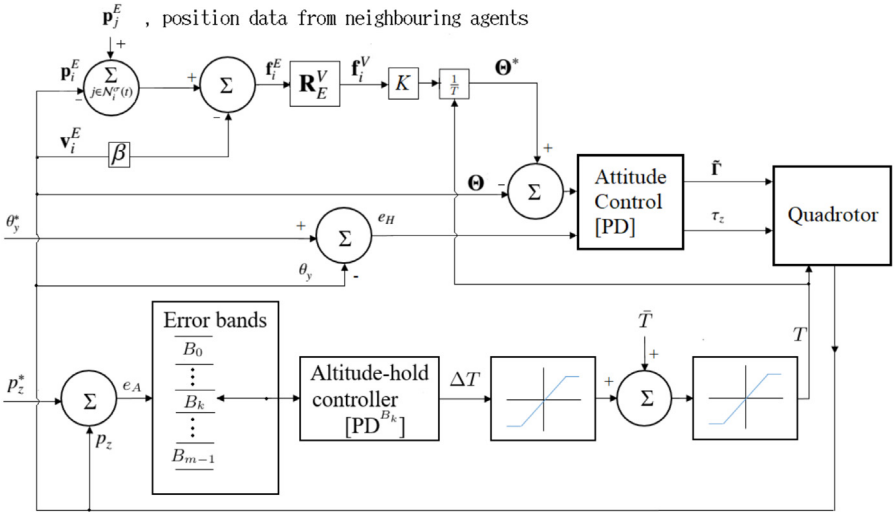

Fig. 7. A block diagram of the quadrotor control loops.

integrator. The attitude control loop is designed such that $\theta_p \rightarrow \theta_p^*$ and $\theta_r \rightarrow \theta_r^*$ in milliseconds. Due to low moment of inertia, a change in roll and pitch angles is attained much faster (within 50 ms) than translational motion, which is in the range of 3 to 5 m/s.

Remark 1. The heading control loop given by (2) and shown in Fig. 7 is designed such that $\theta_y \rightarrow \theta_y^*$ within around 500 ms and is held constant. Hence, the matrix given by (1) can be assumed to remain constant for translational motion considerations.

Remark 2. As θ_p and θ_r of the quadrotor change, the vertical component of the total thrust vector, T reduces by a factor of the cosine of θ_p and θ_r. The angles being small, the reduction is not severe. The altitude control loop given by (3) and shown in Fig. 7 is designed to be fast and corrects the experienced drop at a rate of 25 cm/s.

Thus the quadrotor can be approximated as two independent double integrator systems. This is verified (refer Section 5) by comparing the theoretically predicted trajectories of double integrator agents with the actual trajectories of physical quadrotors.

3.3.2. Control law for waypoint navigation

For the sake of analysis, $\{E\}$ is selected as reference to describe quadrotor motion using $\mathbf{p}^E$ and $\mathbf{v}^E$. For successful waypoint navigation, $\mathbf{f}^E$ should be varied according to the following law (Corke, 2011):

$$\mathbf{f}^E = mK_f[K_p(\mathbf{p}^{E*} - \mathbf{p}^E) - \mathbf{v}^E], \tag{9}$$

where K_f and K_p are constants. The corresponding desired values of angles, $\mathbf{\Theta}^* = \begin{bmatrix}\theta_p^* & \theta_r^*\end{bmatrix} \in \mathbb{R}^2$ from the control law (9) navigate the quadrotor to the desired location $\mathbf{p}^{E*}$. Manipulating (5), (6), (7) and (9), the expression to be

$$\mathbf{\Theta}^* = \mathbf{R}_E^V \frac{mK_f}{T}[K_p(\mathbf{p}^{E*} - \mathbf{p}^E) - \mathbf{v}^E]. \tag{10}$$

is obtained. In order to attain $\mathbf{\Theta}^*$, torques $\tilde{\mathbf{\Gamma}} = \begin{bmatrix}\tau_x & \tau_y\end{bmatrix}^T \in \mathbb{R}^2$ are generated using a proportional-derivative controller given by

$$\tilde{\mathbf{\Gamma}} = K_{p_{r,p}}(\mathbf{\Theta}^* - \mathbf{\Theta}) + K_{d_{r,p}}(\dot{\mathbf{\Theta}}^* - \dot{\mathbf{\Theta}}), \tag{11}$$

where $K_{p_{r,p}} = \begin{bmatrix}K_{p_r} & K_{p_p}\end{bmatrix}^T \in \mathbb{R}^2$ and $K_{d_{r,p}} = \begin{bmatrix}K_{d_r} & K_{d_p}\end{bmatrix}^T \in \mathbb{R}^2$ are the control gains.

3.4. Proposed consensus law

In this section a control law that drives n quadrotors to consensus is proposed. The team of n quadrotors is said to achieve consensus if, for all $\mathbf{p}_i^E(0)$ and $\mathbf{v}_i^E(0)$ and all $i, j = 1, \ldots, n$, $\|\mathbf{p}_i^E(t) - \mathbf{p}_j^E(t)\| \rightarrow 0$ and $\mathbf{v}_i^E(t) \rightarrow 0$ as $t \rightarrow \infty$.

Let each quadrotor represent a node in the set $\mathcal{V} = \{1, \ldots, n\}$ and a communication link between two quadrotors at time t, an edge $e_{ij}^\sigma(t) \in \mathcal{E}^\sigma(t) \subseteq (\mathcal{V} \times \mathcal{V})$. A communication link between nodes is formed whenever they are within each others' communication range. If a communication link exists between nodes i and j, then they are said to be neighbours. Let a set S index all possible undirected interaction topologies between the n nodes. Thus, a time-varying undirected graph, $\mathcal{G}^\sigma(t) := (\mathcal{V}, \mathcal{E}^\sigma(t))$, where a piecewise constant switching signal $\sigma(t) : [0, \infty) \rightarrow S$ with switching times $t_0, t_1, \ldots$, is used to model the switching interaction topology. Let τ be the minimum dwell time between consecutive switches such that $t_{k+1} - t_k \geq \tau$ for all non-negative integers, k. Let $\mathcal{N}_i^\sigma(t) := \{j \in \mathcal{V} : (i, j) \in \mathcal{E}^\sigma\}$ denote the set of neighbours for a node i at time t. The adjacency matrix $\mathcal{A}^\sigma(t)$ of the undirected graph $\mathcal{G}^\sigma(t)$ represents communication network between nodes at time instant t and is defined as $\mathcal{A}^\sigma(t) := [a_{ij}^\sigma(t)] \in \mathbb{R}^{n \times n}; a_{ij}^\sigma(t) = 1$, if communication link exists between agents i and j at time t ($e_{ij}^\sigma(t) \in \mathcal{E}^\sigma(t)$), else $a_{ij}^\sigma(t) = 0$. The Laplacian matrix $\mathcal{L}^\sigma(t)$ of the undirected graph $\mathcal{G}^\sigma(t)$ is defined as $\mathcal{L}^\sigma(t) := [l_{ij}^\sigma(t)] \in \mathbb{R}^{n \times n}; l_{ij}^\sigma(t) = -a_{ij}^\sigma(t), i \neq j, l_{ii}^\sigma(t) = \sum_{j=1, j \neq i}^n a_{ij}^\sigma(t)$.

As shown in Section 3, the forces acting on the quadrotor can be controlled by manipulating θ_r and θ_p. The waypoint navigation law for a single quadrotor given in (9) is extended to formulate a consensus law to drive the n nodes to consensus. The proposed consensus law is given by

$$\mathbf{f}_i^E(t) = \sum_{j \in \mathcal{N}_i^\sigma(t)} a_{ij}^\sigma(t)(\mathbf{p}_j^E(t) - \mathbf{p}_i^E(t)) - \beta \mathbf{v}_i^E(t), \quad i = 1, \ldots, n \tag{12}$$

where β is a positive constant to be tuned by the designer. Under this consensus law, the closed-loop dynamics of the system in matrix form becomes

$$\begin{bmatrix}\dot{\mathbf{p}} \\ \dot{\mathbf{v}}\end{bmatrix} = (\boldsymbol{\alpha}^\sigma(t) \otimes I_2)\begin{bmatrix}\mathbf{p} \\ \mathbf{v}\end{bmatrix} \text{ where } \boldsymbol{\alpha}^\sigma(t) = \begin{bmatrix}0_{n \times n} & I_n \\ -\mathcal{L}^\sigma(t) & -\beta I_n\end{bmatrix}, \tag{13}$$

where $I_n \in \mathbb{R}^{n \times n}$ is an identity matrix, $\mathbf{p} := \begin{bmatrix}\mathbf{p}_1^E & \ldots & \mathbf{p}_n^E\end{bmatrix}^T$ and $\mathbf{v} := \begin{bmatrix}\mathbf{v}_1^E & \ldots & \mathbf{v}_n^E\end{bmatrix}^T$.

If the communication graph remains static, that is $\sigma(t) =$ a constant (say s), for all t, then the following theorem predicts the final consensus value that the agents reach.

Theorem 1. *Suppose a system is governed by dynamics given by (13) with constant $\sigma(t) = s$. Then, the control law in (12) with $\beta > 0$ achieves consensus asymptotically if the communication graph $\mathcal{G}^s$ is connected.*

Proof. Following the analysis given in Ren and Atkins (2007, Lemma 4.1), it is possible to show that $\mathbf{p}(t) \to (\beta \mathbf{1}_n \mathbf{1}_n^T \otimes I_2)\mathbf{p}(0) + (\mathbf{1}_n \mathbf{1}_n^T \otimes I_2)\mathbf{v}(0)$ and $\mathbf{v}(t) \to 0$ as $t \to \infty$.

For details of proof refer Appendix B.1 □

Let us define the consensus error vector as $\tilde{\mathbf{p}} = \begin{bmatrix}\mathbf{p}_{12}^E & \dots & \mathbf{p}_{1n}^E\end{bmatrix}^T$ and $\tilde{\mathbf{v}} = \begin{bmatrix}\mathbf{v}_{12}^E & \dots & \mathbf{v}_{1n}^E\end{bmatrix}^T$, where $\mathbf{p}_{ij}^E = \mathbf{p}_i^E - \mathbf{p}_j^E$ and $\mathbf{v}_{ij}^E = \mathbf{v}_i^E - \mathbf{v}_j^E$. Then (13) can be rewritten in relative dynamics form as

$$\begin{bmatrix}\dot{\tilde{\mathbf{p}}} \\ \dot{\tilde{\mathbf{v}}}\end{bmatrix} = (\tilde{\boldsymbol{\alpha}}^\sigma \otimes I_2) \begin{bmatrix}\tilde{\mathbf{p}} \\ \tilde{\mathbf{v}}\end{bmatrix}, \tag{14}$$

where $\tilde{\boldsymbol{\alpha}}^\sigma \in \mathbb{R}^{2(n-1)\times 2(n-1)}$. If, for fixed $\sigma \in S$, $\tilde{\boldsymbol{\alpha}}^\sigma$ is stable, then there exist $p^\sigma \geq 0$ and $q^\sigma \geq 0$ such that $\|e^{\tilde{\boldsymbol{\alpha}}^\sigma t}\| \leq e^{(p^\sigma - q^\sigma)t}, t \geq 0$ (Ren & Atkins, 2007).

Theorem 2. *If $\beta > 0$, the communication graph is connected for all $\sigma \in S$ and the dwell time, τ satisfies $\tau > \sup_{\sigma \in S} \frac{p^\sigma}{q^\sigma}$, then consensus is asymptotically achieved using (12).*

Proof. Refer Appendix B.2 for the proof. □

This consensus law is implemented on the multi-quadrotor testbed in an outdoor environment.

4. Communication protocol

For successful execution of cooperative control strategies, inter-agent communication plays a vital role. Carrier sense multiple access with collision avoidance protocols (CSMA/CA) are commonly used for communication in multi-agent systems (Guerrero et al., 2013; Vásárhelyi et al., 2014). However, communication outages due to data packet collisions for up to 2 s have been observed in preliminary experiments with multiple quadrotors (Joshi et al., 2016). With additional computational overheads, Vásárhelyi et al. (2014) reports outages in the range of seconds. Thus, the standard CSMA-CA protocol is not suitable for real-time communication in this context as there is no guaranteed upper-bound on the delay in transmission (Guerrero et al., 2013).

In this work a communication protocol based on time division multiplexing (TDM) is developed wherein agents with limited communication range can reliably transmit data without collisions. The protocol allows synchronized real-time communication between agents without the need for a ground station. Synchronization is achieved using an additional reference node mounted on-board one of the quadrotors, thus making the communications system capable of being fully airborne. It is also capable of handling dynamic communication topologies and robust to link breakages with the reference node. It provides guaranteed throughput (of up to 3 data packets per second per node for data packets of 15 bytes) and maintains connectivity between agents.

4.1. Steps involved

Broadly, the protocol comprises of the following steps:

1. Time synchronization
2. Slot allotment
3. Data transfer
4. Re-synchronization

Suppose that the communication network needs to be set up between n agents. Then $n+1$ nodes are required for the implementation of this protocol. A reference node, say Node 0, is used for synchronization. Nodes $1, \dots, n$ correspond to the n agents. Node 0 and Node 1 are physically set up adjacent to each other on one agent. Fig. 8 depicts the various steps from left to right in chronological order. Each step is discussed in detail below.

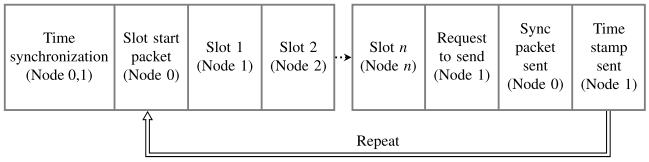

Fig. 8. Workflow of the communication protocol.

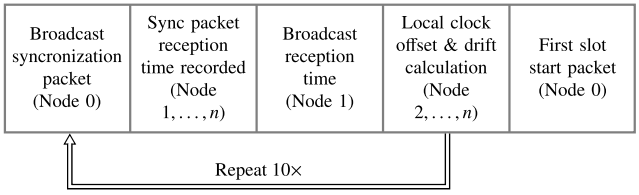

Fig. 9. Steps involved in the initial time synchronization.

4.1.1. Time synchronization

To achieve time-based slot allotment, the local clocks of all nodes need to be synchronized during the initialization phase. All the nodes are synchronized using reference broadcast synchronization (RBS) (Elson, Girod, & Estrin, 2002). Fig. 9 explains the time synchronization process.

Example

Consider time synchronization between Node 1 and Node 2. Initially, Node 0 broadcasts a synchronization packet which is received by Node 1 at time t_1 and Node 2 at t_2. Node 1 now sends its packet reception time, t_1. This process is repeated ten times. Node 2 uses these ten time-stamps to calculate its local clock offset (t_2-t_1) with respect to local clock of Node 1 and correct for any clock drift.

Experiments show that the worst case synchronization error between the clocks of Node 1 and any other node is less than 2 ms.

4.1.2. Slot allotment

Now that local clocks of all the nodes are synchronized, the time slots during which each node can transmit its data, can be allotted. All nodes are assigned a number based on which the time slots are allotted in an ascending order. It is observed that given the worst case of synchronization error, a node is successfully able to transmit a data packet of 15 bytes in a time duration of less than 20 ms. Hence, each node is given 20 ms for data transmission. The end of slot allotment is marked by a broadcast packet by Node 0 which signals that the nodes can now start data transfer. Fig. 8 depicts the manner in which Node i transmits data in Slot i.

4.1.3. Data transfer

Data transfer commences once Node 0 broadcasts a slot start packet. Node 1 is the first to transmit data. The other nodes follow based on local clock times which are synchronized with respect to Node 1. The Xbee API mode is used for data transfer. Fig. 10 shows the data packet structure used for transmission. The packet consists the standard start delimiter (0x7E), which marks start of new packet and length, the position coordinates (15 bytes) of the node and checksum which is used to validate the received data.

4.1.4. Re-synchronization

This step helps maintain the synchronization achieved during the initialization phase. At the end of the last slot of every frame, Node 1 requests Node 0 to transmit a synchronization packet followed by the time-stamp sent by Node 1 as shown in Fig. 8. New coefficients are

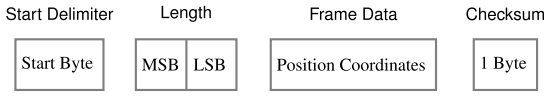

Fig. 10. Data packet structure of Xbee in API mode.

calculated using a sliding window average of the last ten consecutive time differences for drift compensation. This process is similar to the time synchronization step. Node 0 broadcasts a special data packet that marks the end of time synchronization and the whole data transfer process is repeated.

4.2. Addressing contingencies

The main application of this protocol is in the case where the agents are in motion. As the agents move in and out each others' communication range, links are dynamically created and dropped. In cluttered environments, there can be situations where obstacles affect communication between agents.

4.2.1. Link break with Node 0, 1

An obstacle appearing in between Node 0, 1 and any other nodes, can result in a loss of reception of the synchronizing and time stamped data packets. In such a case, the node receives data packets from other nodes in its vicinity and records their time of arrival. As each node knows its assigned slot number, it can calculate its relative slot start time based on the recorded time of arrival. This process is explained below with the help of an example.

Example

Consider the situation presented in Fig. 11. Observe that Node 3 has lost connection with Node 0, 1 due to the presence of an obstacle. However, it can still form a communication link with nodes 2 and/or 4. Suppose the total frame cycle has a duration of T milliseconds. If Node 3 receives a data packet from Node 2 at time t, then it calculates its own slot start time to be $(t+20)$ milliseconds. If Node 3 receives a data packet from Node 4 at t, then it calculates its own slot start time to be $(t+T-20)$ milliseconds. This process continues till the broken link with the Node 0, 1 is re-established and synchronization takes place.

4.2.2. Complete loss of reception

There can be a situation wherein a node is completely isolated and does not receive any data from the other nodes. In such a case, the node keeps sending its data packets in the allotted time slot for up to 15 s after complete loss of reception and then switches to only receiving data from the other nodes. A time duration of 15 s is chosen as it is experimentally observed that nodes maintain acceptable synchronization of up to 20 ms, after which data packet collisions tend to occur due to time slots overlapping. When a node returns to the network, it gets synchronized again and starts participating in the normal data transfer process.

5. Results

This section contains the results of the experiments performed to test the performance of the communication protocol, the consensus algorithm and the effect of communication rate on consensus performance.

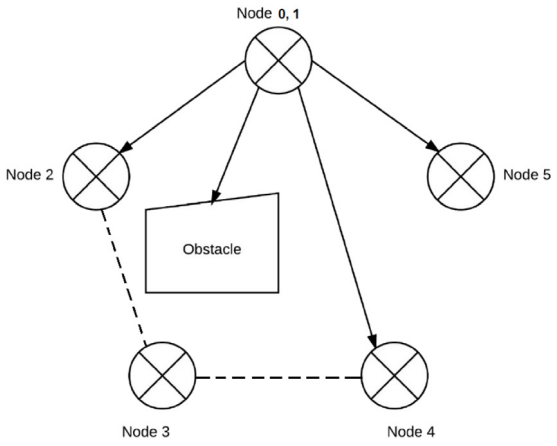

Fig. 11. An instance of case of link break between Nodes 1 and 3 due to the presence of an obstacle.

Table 1
Sample packet transmission times of Node 0 during initialization phase.

Packet type	Packet no.	Time (s)
Broadcast sync packet	1	0.000000
	2	0.228948
	3	0.462112
	4	0.686640
	5	0.916764
	6	1.146924
	7	1.379964
	8	1.607384
	9	1.837680
	10	2.068088
Slot start packet	1	3.098908

5.1. Communication protocol experiments

Experiment 1 : Verification of communication protocol

(a) Setup time: Setup time is the time required for the completion of the initialization phase which comprises of the time synchronization and slot allotment stages. Table 1 contains sample data regarding the times at which Node 0 broadcasts synchronization and slot start packets. It broadcasts ten synchronization packets between 0–2.07 s. This is followed by the slot allotment phase after which the slot start packet is broadcast at 3.1 s.

(b) Communication efficiency: The efficiency of communication can be inferred from the transmission efficiency and reception efficiency. The transmission efficiency is characterized by the number of slots a node participates in. The number of slots a node participates in is equivalent to the number of data packets it transmits. As stated earlier, local clocks of all nodes are synchronized with respect to the clock of Node 1. Node 1 always receives the slot start packet sent by Node 0 as they are physically placed next to each other. Thus, it is assumed that Node 1 participates in all the slots. Based on this assumption, the efficiency of transmission of Node i is calculated as

$$\eta_{t,i} = \frac{D_{t,i}}{D_{t,1}}, \tag{15}$$

where $D_{t,i}$ is the total number of packets transmitted by Node i over the entire duration of the experiment. Reception efficiency is defined as

$$\eta_{r,i} = \frac{1}{n-1}\sum_{\substack{j=1 \\ j\neq i}}^{n} \frac{D_{r,i}}{D_{t,j}}, \tag{16}$$

where $D_{r,i}$ is the total number of packets received by Node i over the entire duration of the experiment. Then, the average transmission and

Table 2
A comparison of number of nodes in the network and communication efficiency.

No. of nodes	No. of test runs	η_t^{avg}	η_r^{avg}
3	6	99.09%	99.06%
4	4	98.29%	99.60%
5	3	98.82%	98.96%

Table 3
A comparison of communication efficiency in indoor and outdoor environment.

Environment	No. of test runs	η_t^{avg}	η_r^{avg}
Indoor	2	97.38%	97.88%
Outdoor	3	95.04%	95.58%

Table 4
Number of data packets transmitted and received by three nodes in an indoor environment.

Node	1	2	3
1	**886**	824	804
2	866	**855**	873
3	880	851	**877**

Table 5
Communication efficiency of three nodes in an indoor environment.

Node	1	2	3
$\eta_{t,i}$	100	98.73	98.98
$\eta_{r,i}$	94.03	98.64	99.43

reception efficiency over R test runs is calculated as

$$\eta_t^{avg} = \frac{1}{R}\frac{\sum_{i=1}^{n}\eta_{t,i}}{n} \tag{17}$$

and

$$\eta_r^{avg} = \frac{1}{R}\frac{\sum_{i=1}^{n}\eta_{r,i}}{n} \tag{18}$$

respectively.

A total of 18 test runs of duration 300 s each are carried out in indoor and outdoor environments with three to six nodes in the communication network. Using (17), the average transmission efficiency is found to be 98.04%. The reception efficiency of the protocol is found using (18) to be 98.52%. Experiments are performed to observe the effect of addition of nodes to the network, on the communication transmission and reception efficiencies. These experiments are repeated for different number of nodes. From Table 2, we can infer that the addition of communication nodes to the network does not decrease the transmission and reception efficiencies significantly.

Table 3 provides a comparison of communication efficiency in indoor and outdoor environment. The data in this table is for communication between 6 nodes. The indoor test runs are carried out in an area of 10 m × 10 m and outdoor area has dimensions 55 m × 46 m. It can be seen that there is a reduction (approximately 2%) in transmission and reception communication efficiency. This loss can be attributed to environmental factors like interference and attenuation.

Sample data from the test runs are given in Tables 4, 6 and 8. In these tables, the (i,i)th entry denotes the number of data packets transmitted by Node i and the (i,j)th entry denotes the number of data packets of Node j received by Node i.

Table 4 shows the data of a sample run performed using three nodes. This run is performed in an indoor area with dimensions 10 m × 10 m and the nodes are at a height of 1 m from the ground. The communication efficiency for this run is shown in Table 5.

Similarly, Table 6 shows the data of a sample run performed using six nodes. This run is also performed in an indoor area with dimensions 10 m × 10 m and the nodes are at a height of 1 m from the ground. The communication efficiency for this run is shown in Table 7.

Table 6
Number of data packets transmitted and received by six nodes in an indoor environment.

Node	1	2	3	4	5	6
1	**881**	871	857	864	872	837
2	878	**875**	851	867	872	837
3	877	871	**868**	854	872	837
4	878	871	852	**875**	872	837
5	878	870	857	861	**873**	810
6	840	834	830	837	836	**837**

Table 7
Communication efficiency of six nodes in an indoor environment.

Node	1	2	3	4	5	6
$\eta_{t,i}$	100	99.32	98.52	99.32	99.09	95.01
$\eta_{r,i}$	99.38	99.33	99.31	99.45	98.60	95.54

Table 8
Number of data packets transmitted and received by six nodes in an outdoor environment.

Node	1	2	3	4	5	6
1	**831**	770	710	782	742	778
2	791	**798**	715	774	751	775
3	725	770	**738**	771	741	780
4	757	789	709	**788**	769	778
5	792	782	721	779	**796**	786
6	770	769	704	776	726	**804**

Table 9
Communication efficiency of six nodes in an outdoor environment.

Node	1	2	3	4	5	6
$\eta_{t,i}$	100	96.03	88.81	94.83	95.79	96.75
$\eta_{r,i}$	96.38	96.21	94.34	95.88	97.52	94.82

Comparing Tables 5 and 7, it can be inferred that adding nodes to the communication protocol does not hamper the transmission and reception efficiency.

In Table 8, the data of a sample run performed using six nodes in an outdoor environment is presented. This run is performed in an outdoor area with dimensions 55 m × 46 m. The communication efficiency for this run is shown in Table 9.

Comparing Tables 7 and 9, it is observed that the average transmission efficiency has dropped by 3.17% and the average reception efficiency has dropped by 2.74%. This loss can be attributed to environmental factors.

(c) Link break: In this experiment, the performance of the protocol in case a node is unable to receive synchronization packets from Node 0, is tested. The experiment is performed with six nodes. Fig. 12 depicts, along the y-axis, the data packets that Node 3 transmits (in blue) and receives from other nodes (in grey, maroon, green, red and black). The different slots in which communication occurs are plot along the x-axis. It can be seen that Node 3 is unable to receive synchronization packets from Node 0 starting from slot number 260. However, it is still able to receive data packets from Node 2. Using the time of arrival of this data, Node 3 calculates its own time slot and successfully transmits data as seen in the figure.

(d) Sample communication cycle: A snapshot of nine complete time-frames of the communication protocol can be seen in Fig. 13. Each circle in the figure depicts the time at which the corresponding node transmits data. The first four packets (marked along the y-axis) correspond to the re-synchronization phase which is followed by the data exchange between the six nodes. A framework for communication between ten nodes is laid out. It can be verified that each node transmits data in the allotted slot of 20 ms and that there are no data collisions during this time period. Further, the re-synchronization stage is observed to take 130 ms. Hence, the duration of one time-frame is 20×10+130 = 330 ms. This can be verified in Fig. 13. Thus, a guaranteed throughput (of up to 3 data packets per second per node) is obtained.

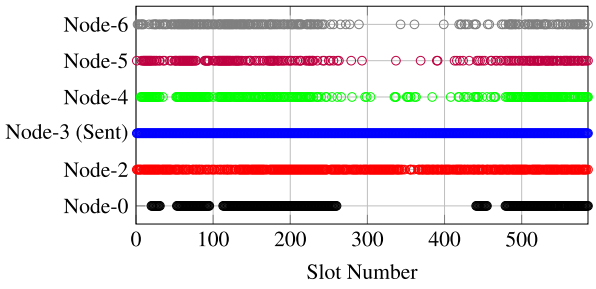

Fig. 12. Data transfer during the link break mode of operation. (For interpretation of the references to colour in this figure legend, the reader is referred to the web version of this article.)

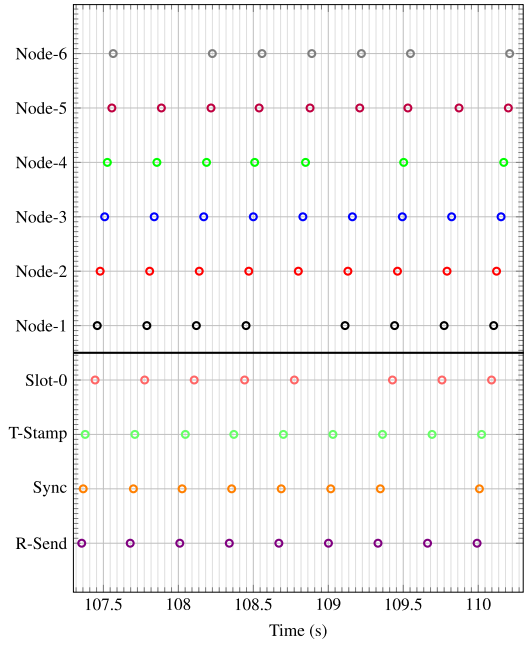

Fig. 13. A snapshot of nine time-frames of the communication protocol. Each circle corresponds to the time at which the node transmits data.

Table 10
Position data of the quadrotors for Experiment 2(a).

Node	Type	Initial position (m)	Final position (m)
1	Physical	(14.39, 0.57)	(20.62, 15.17)
2	Physical	(26.21, 49.11)	(20.25, 20.88)
3	Physical	(52.44, 5.18)	(22.66, 21.31)
Time to consensus = 4 s			

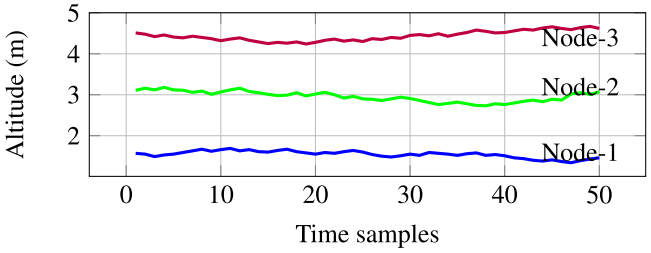

Fig. 14. Plot of altitudes of three physical quadrotors.

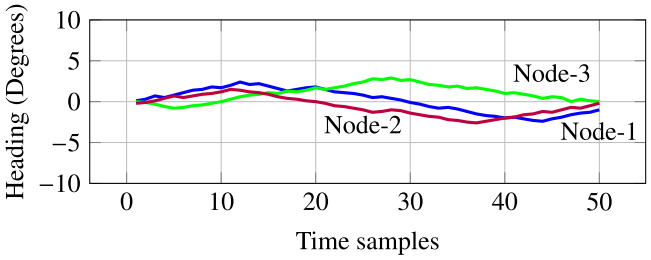

Fig. 15. Heading angle plot of three physical quadrotors.

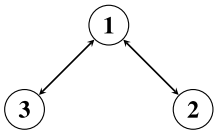

Fig. 16. Static communication graph for consensus experiment with three physical nodes.

5.2. Cooperative control experiments

In this section the results of the cooperative control experiments performed with the physical and virtual quadrotors are presented. The experiments are performed on an outdoor field with dimensions 90 m × 50 m. The ublox LEA-6H GPS receiver used for localizaton by the quadrotors has an accuracy of 2.5 m CEP (ublox, 2017) (the measured position is within a circle of radius 2.5 m of the true position 50% of the time Nelson, 1988). Hence, for practical purposes the quadrotors are said to achieve consensus when $\|\mathbf{p}_i^E(t) - \mathbf{p}_j^E(t)\| \leq 5$ m for all $i, j = 1, \ldots, 6$.

Experiment 2: Static graph

(a) Path graph: This experiment is performed with the communication graph between the three physical quadrotors as shown in Fig. 16. Table 10 provides details about the initial and final conditions of the three physical quadrotors with respect to a reference frame $\{R\}$ shown in Fig. 18. A GPS plot of the trajectories that the quadrotors follow to reach consensus is given in Fig. 18. A plot of the altitudes and the heading angles of the three quadrotors is given in Figs. 14 and 15 respectively. It can be seen in Fig. 14 that the altitude hold loop (see Section 3.2) is able to maintain the altitude of the three quadrotors at 1.5 m, 3 m and 4.5 m respectively. Also, the heading hold control loop (see Section 3.1) is able to maintain the heading angle of the three quadrotors at 0° as seen in Fig. 15.

A simulation for consensus of three double integrator agents with the same communication graph and initial conditions as the physical quadrotors is also performed. The resulting trajectories are depicted by the solid lines in Fig. 17. The corresponding trajectories from the physical implementation, depicted by the circles have been overlaid on the simulated trajectories.

In both cases the time taken by the agents to reach consensus is 4 s. The average absolute tracking error along the x-axis and y-axis is 6.16 m and 6.54 m respectively. The cumulative RMS error

$$e_{RMS} = \frac{1}{n}\sum_{i=1}^{n}\sum_{k=1}^{T}\frac{\sqrt{(p_{i,x}^E(k) - p_{i,x}^s(k))^2 + (p_{i,y}^E(k) - p_{i,y}^s(k))^2}}{T},$$

where $p_{i,x}^E$ and $p_{i,y}^E$ are the coordinates of the physical quadrotor and $p_{i,x}^s$ and $p_{i,y}^s$ are the coordinates of the simulated quadrotor i along the x and y axis respectively, and T is the time duration of the experiment, is found to be 9.83 m.

(b) Complete graph: This experiment is performed with a complete communication graph between the three physical and three virtual quadrotors, i.e. every quadrotor can communicate with every other quadrotor. Table 11 provides details about the initial and final conditions of the three physical quadrotors with respect to a reference frame

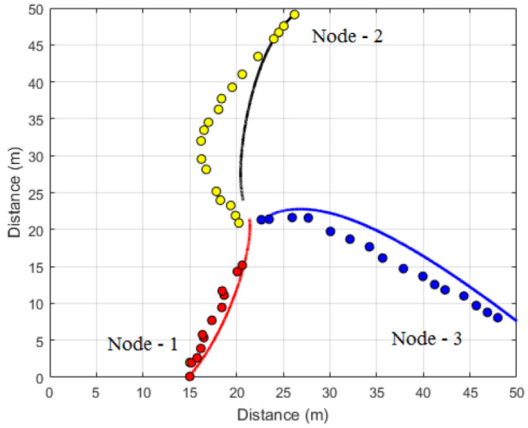

Fig. 17. Comparison between simulated and experimental trajectories of three agents under the consensus law. Trajectories from the experiment are depicted by the circles and simulated trajectories are by solid lines.

Fig. 18. GPS plot of consensus between three physical quadrotors.

Table 11
Position data of the quadrotors for Experiment 2(b).

Node	Type	Initial position (m)	Final position (m)
1	Physical	(0.45, 22.78)	(27.29, 28.07)
2	Physical	(15.93, 10.98)	(27.67, 28.56)
3	Physical	(1.66, 10.50)	(29.17, 30.28)
4	Virtual	(49.97, 10.05)	(27.48, 28.49)
5	Virtual	(49.78, 52.74)	(28.26, 29.72)
6	Virtual	(23.65, 51.01)	(27.33, 27.96)
Time to consensus = 4 s			

{R} shown in Fig. 19. From the GPS plot of trajectories and the data in Table 11, it can be seen that the six nodes reach consensus, inspite of wind disturbances. The time taken to reach consensus in this case is also 4 s.

Experiment 3: Dynamic graphs

In these experiments, the communication graph between the quadrotors changes based on the distance between them.

(a) Simulation results: This experiment is performed with three virtual quadrotors. The communication range of each node is 50 m. Initially, the three quadrotors are placed such that the communication graph between them is as shown in Fig. 20 a. Each node moves towards the centroid of its neighbours as per the consensus law (12). Consider the motion of Node 1. As Node 1 is connected to only Node 2, it starts moving towards the centroid of Nodes 1 and 2, $\frac{(\mathbf{p}_2+\mathbf{p}_1)}{2}$. As soon as the distance between Nodes 1 and 3 reduces to below 50 m (as depicted

Fig. 19. GPS plot of consensus between six nodes: three physical and three virtual units.

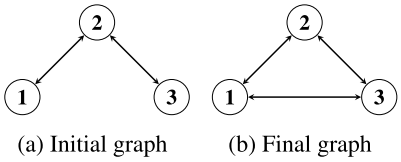

Fig. 20. A case of switching communication topology.

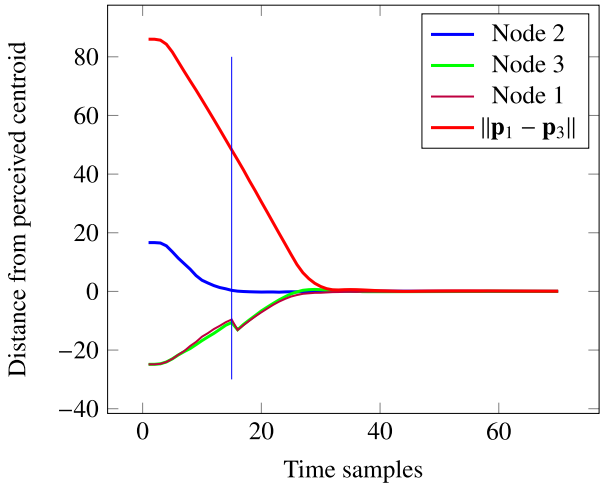

Fig. 21. Centroid tracking in the case of switching communication topology. The blue vertical line denotes the time at which a link between Nodes 1 and 3 is formed.

by the vertical blue line in Fig. 21), a communication link between them is formed. The new communication graph is now as shown in Fig. 20b. Thus, Node 1 now starts moving towards the updated centroid, $\frac{(\mathbf{p}_1+\mathbf{p}_2+\mathbf{p}_3)}{3}$, of all its neighbours i.e. Nodes 1, 2 and 3. Inspite of the shift in consensus point due to the change in the communication graph, as depicted by the kink in Fig. 21, the nodes finally reach consensus.

(b) On-field implementation: The same experiment is performed using physical quadrotors. Initially, the communication graph between the nodes is as shown in Fig. 20a and Nodes 1 and 3 move towards Node 2. As soon as Nodes 1 and 3 are within range, switching occurs and the communication graph changes to Fig. 20b. Table 12 provides details about the initial and final conditions of the three physical quadrotors with respect to a reference frame {R} shown in Fig. 22. The quadrotors

Table 12
Position data of the quadrotors for Experiment 3(b).

Node	Type	Initial position (m)	Final position (m)
1	Physical	(29.61, −29.39)	(6.84, −5.87)
2	Physical	(−1.87, 3.91)	(7.52, −10.22)
3	Physical	(27.05, 34.58)	(7.22, −10.33)
Time to consensus = 8 s			

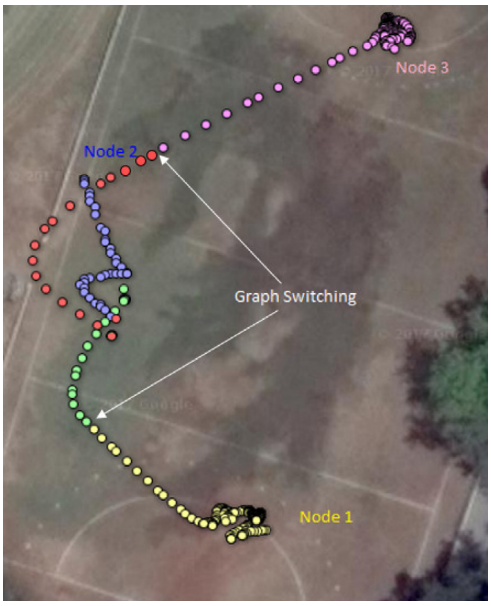

Fig. 22. GPS plot of consensus between three physical quadrotors with a dynamic communication graph.

follow trajectories as shown in Fig. 22 and reach consensus. The time taken to reach consensus in this case is 8 s.

Experiment 5: Consensus with different data exchange rates

In this experiment, the rate at which data is exchanged between the nodes is altered to study effect of communication on consensus law. The consensus law is executed on the virtual units for various data exchange rates ranging from 3 packets per second to one packet every ten seconds. The speed of the quadrotors is fixed to 5 m/s.

The consensus law guarantees that the agents will asymptotically reach consensus. It is observed that the quadrotors reach the same consensus point at approximately the same time in Fig. 23, (a) and (b). However, the performance is hampered as the frequency is reduced below 1 data packet per 1 s. For lower data rates, the quadrotors oscillate about a consensus point as seen in Fig. 23(c)–(f).

It should be noted that in Vásárhelyi et al. (2014), the authors report communication outages even in the range of seconds. From these experiments, it is evident that in close range operations, such as consensus, performance degrades in such situations.

6. Conclusion

In this work, a decentralized consensus algorithm on an outdoor testbed of quadrotors is implemented and a real-time communication protocol for inter-agent data transfer is developed. It is shown that the quadrotor, for small roll and pitch angles, with heading (yaw) held constant, can be approximated as a pair of double integrators. This allows us to modify and implement a consensus algorithm for double integrator dynamics with provable convergence properties on a testbed of six quadrotors. From the consensus experiments, a match between theoretical results and practise is observed. Consensus between quadrotors is achieved using minimum possible information exchange between agents (position only). Experiments are performed with the proposed communication protocol to verify its efficacy. It is verified that the protocol is capable of transmitting data without collisions in real-time and handling link breakages and additions. Consensus experiments with different data exchange rates are performed to show the degradation of performance as communication rate is reduced. A theory and quantitative study to characterize the effect of communication delay on consensus performance is left for future investigation.

Conflict of interests

None declared

Appendix A. Transformation between local-level frame ($\{L\}$) and virtual reference frame ($\{S\}$)

The fact that over very small distances, the latitude and longitude can be assumed to vary linearly in the $\{L\}$ frame (Clynch, 2002) is used.

Let the coordinates of a quadrotor be $\begin{bmatrix} p_x^L(t) & p_y^L(t) \end{bmatrix}^T$ in the $\{L\}$ frame and $\begin{bmatrix} p_x^S(t) & p_y^S(t) \end{bmatrix}^T$ in the $\{S\}$ frame. The reference latitude and longitude, $\begin{bmatrix} \varphi^* & \lambda^* \end{bmatrix}^T$ are set to be the actual GPS coordinates of a point in the outdoor arena where the experiments are being carried out. Then, $\begin{bmatrix} p_x^S(t) & p_y^S(t) \end{bmatrix}^T = (0,0)$ at $\begin{bmatrix} p_x^L(t) & p_y^L(t) \end{bmatrix}^T = \begin{bmatrix} \varphi^* & \lambda^* \end{bmatrix}^T$.

The following constants as per the WGS84 system (Clynch, 2002) are used for the transformation

- Equatorial Radius, $R_{eq} = 6378137.00$ m
- Polar Radius, $R_p = 6356752.31$ m
- Flattening, $f = \dfrac{R_{eq} - R_p}{R_{eq}} = \dfrac{1}{298.26}$
- Eccentricity, e is defined as $e^2 = 1 - \dfrac{R_p^2}{R_{eq}^2} = 2f - f^2$
- Radius of Curvature in Prime Vertical, $R_{PV} = R_N = R_{eq} \dfrac{1}{\sqrt{1 - e^2 sin^2(\varphi^*)}}$
- Radius of Curvature in Meridian $R_M = \dfrac{R_{PV}(1 - e^2)}{1 - e^2 sin^2(\varphi^*)}$

Conversion from $\{L\}$ to $\{S\}$ frame

The $\begin{bmatrix} p_x^S(t) & p_y^S(t) \end{bmatrix}^T$ coordinates in $\{S\}$ can be calculated using the following two equations:

$$p_x^S = R_M \times (p_x^L - \varphi^*) \times \frac{\pi}{180} \tag{A.1}$$

$$p_y^S = R_{PV} cos(\varphi^*) \times (p_y^L - \lambda^*) \times \frac{\pi}{180} \tag{A.2}$$

Conversion from $\{S\}$ to $\{E\}$ frame

The above two equations can be rearranged to give the GPS coordinates in $\{E\}$ frame as follows:

$$p_x^L = \varphi^* + \frac{p_x^S}{R_M} \times \frac{180}{\pi} \tag{A.3}$$

$$p_y^L = \lambda^* - \frac{p_y^S}{R_{PV} cos(\varphi^*)} \times \frac{180}{\pi} \tag{A.4}$$

Appendix B. Proofs of results in Section 3.4

Lemma 3. *If $\beta > 0$ and the undirected communication graph is connected at each time t, then $\boldsymbol{\alpha}^\sigma$ has $2n - 1$ negative real eigenvalues and one zero eigenvalue for each $\sigma \in S$.*

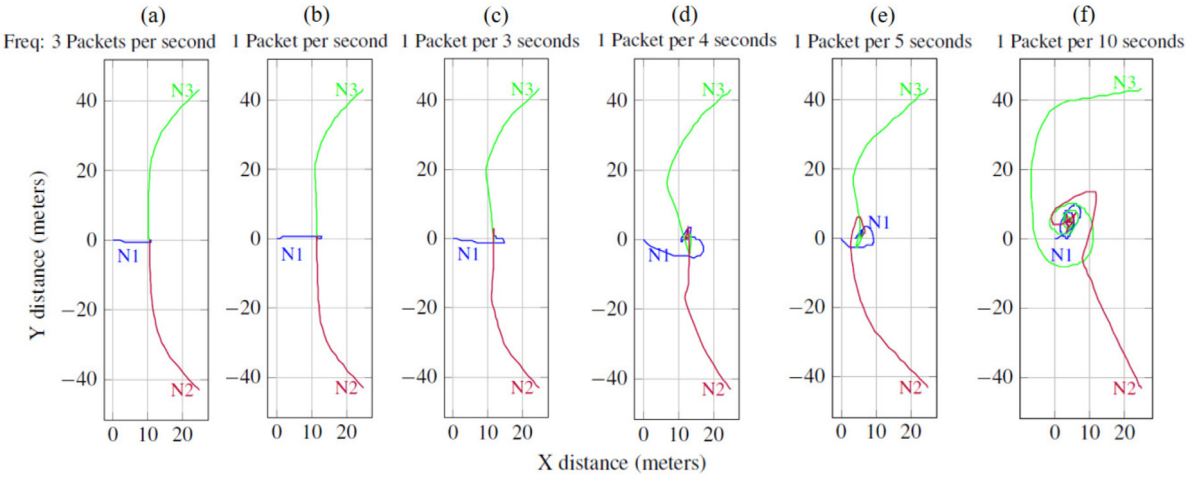

Fig. 23. Plots of quadrotor trajectories for different data exchange rates.

Proof. The eigenvalues, $\lambda_{i\pm}^{\sigma}$ of $\boldsymbol{\alpha}^{\sigma}$ are given by Ren and Atkins (2007)

$$\lambda_{i\pm}^{\sigma} = \frac{-\beta \pm \sqrt{\beta^2 + 4\mu_i^{\sigma}}}{2}, \tag{B.1}$$

where μ_i^{σ} is the ith eigenvalue of $-\mathcal{L}^{\sigma}$. For an undirected graph, all non-zero eigenvalues of $\mathcal{L}^{\sigma}$ are real, positive and zero is a simple eigenvalue if and only if the underlying graph is connected (Ren & Beard, 2008). Then, without loss of generality, say $\mu_1^{\sigma} = 0$. Hence $\lambda_{1+}^{\sigma} = 0$. Choosing $\beta > 0$ ensures that the remaining $2n-1$ eigenvalues of $\boldsymbol{\alpha}^{\sigma}$ have negative real parts as $\sqrt{\beta^2 + 4\mu_i^{\sigma}} \leq \beta$ for all μ_i^{σ} □

B.1. Proof for Theorem 1

Proof (*Sufficiency*). Given that the communication graph is connected, from Lemma 3, it is known that $\boldsymbol{\alpha}^s$ has a simple zero eigenvalue. Let $\mathbf{x}_1 = \begin{bmatrix}\mathbf{x}_a^T & \mathbf{x}_b^T\end{bmatrix}^T$ be the eigenvector of $\boldsymbol{\alpha}^s$ associated with the zero eigenvalue. Then,

$$\boldsymbol{\alpha}^s \mathbf{x}_1 = \begin{bmatrix} 0_{n\times n} & I_n \\ -\mathcal{L}^s & -\beta I_n \end{bmatrix} \begin{bmatrix}\mathbf{x}_a \\ \mathbf{x}_b\end{bmatrix} = \begin{bmatrix}\mathbf{0}_n \\ \mathbf{0}_n\end{bmatrix},$$

where $\mathbf{0}_n$ is the $n\times 1$ vector of all zeros. From the above equation, $\mathbf{x}_b = \mathbf{0}_n$ and $-\mathcal{L}^s\mathbf{x}_a = \mathbf{0}_n$. Then $\mathbf{x}_a$, the eigenvector of $\mathcal{L}^s$ associated with the zero eigenvalue equals $\mathbf{1}_n$, the $n \times 1$ vector of all ones.

Now $\boldsymbol{\alpha}^s$ can be written in the Jordan canonical form as

$$\boldsymbol{\alpha}^s = TJT^{-1} = \begin{bmatrix}\mathbf{x}_1 & \dots & \mathbf{x}_{2n}\end{bmatrix} \begin{bmatrix} 0_{1\times 1} & 0_{1\times 2n-1} \\ 0_{2n-1\times 1} & J' \end{bmatrix} \begin{bmatrix}\mathbf{y}_1^T \\ \vdots \\ \mathbf{y}_{2n}^T\end{bmatrix},$$

where $\mathbf{x}_i \in \mathbb{R}^{2n}$, $i = 1, \dots, 2n$ can be chosen to be the right eigenvectors or the generalized right eigenvectors of $\boldsymbol{\alpha}^s$, $\mathbf{y}_i \in \mathbb{R}^{2n}, i = 1, \dots, 2n$ can be chosen to be the left eigenvectors or the generalized left eigenvectors of $\boldsymbol{\alpha}^s$, and J' is the Jordan canonical matrix corresponding to the non-zero eigenvalues of $\boldsymbol{\alpha}^s$.

To get the above form, $\mathbf{x}_1 = \begin{bmatrix}\mathbf{1}_n^T & \mathbf{0}_n^T\end{bmatrix}^T$ is chosen as the right eigenvector and $\mathbf{y}_1 = \begin{bmatrix}\beta\mathbf{1}_n^T & \mathbf{1}_n^T\end{bmatrix}^T$ as the left eigenvector of $\boldsymbol{\alpha}^s$ corresponding to the zero eigenvalue. Following the analysis given in Ren and Atkins (2007, Lemma 4.1), it is possible to show that $\mathbf{p}(t) \to (\beta\mathbf{1}_n\mathbf{1}_n^T \otimes I_2)\mathbf{p}(0) + (\mathbf{1}_n\mathbf{1}_n^T \otimes I_2)\mathbf{v}(0)$ and $\mathbf{v}(t) \to 0$ as $t \to \infty$. Hence $\|\mathbf{p}_i^E(t) - \mathbf{p}_j^E(t)\| \to 0$ and $\mathbf{v}_i^E \to 0$ as $t \to \infty$ for all $i, j = 1, \dots, n$

The proof for necessity is similar to the necessity proof of Ren and Atkins (2007, Lemma 4.1). □

B.2. Proof for Theorem 2

Proof. Given $\beta > 0$ and that the communication graph is connected for each $\sigma \in \mathcal{S}$, Theorem 1 holds and ensures that consensus is achieved asymptotically for each $\sigma \in \mathcal{S}$. Hence, $\tilde{\mathbf{p}} \to 0$ and $\tilde{\mathbf{v}} \to 0$ asymptotically and the switched system (14) is stable for each $\sigma \in \mathcal{S}$. If the dwell time τ satisfies $\tau > \sup_{\sigma\in\mathcal{S}} \frac{p^{\sigma}}{q^{\sigma}}$, then using (Morse, 1996), it follows that the switched system (14) is globally exponentially stable. This implies that consensus can be achieved asymptotically. □

References

Abdessameud, A., & Tayebi, A. (2013). *Motion coordination for VTOL unmanned aerial vehicles: Attitude synchronisation and formation control*. Springer Science & Business Media.

Acevedo, J. J., Arrue, B. C., Maza, I., & Ollero, A. (2013). Cooperative large area surveillance with a team of aerial mobile robots for long endurance missions. *Journal of Intelligent and Robotic Systems*, 1–17.

Aeroquad - An Arduino based autopilot, https://code.google.com/p/aeroquad/downloads/list, (last accessed 22nd Oct, 2015).

ArduPilot Mega 2.6 Hardware Documenation, http://copter.ardupilot.com/wiki/common-apm25-and-26-\overview/, (last accessed 22nd Oct, 2015).

Casbeer, D. W., Kingston, D. B., Beard, R. W., & McLain, T. W. (2006). Cooperative forest fire surveillance using a team of small unmanned air vehicles. *International Journal of Systems Science*, *37*(6), 351–360.

Clynch, J. R. (2002). Radius of the earth-radii used in geodesy. *Naval Postgraduate School*.

Corke, P. (2011). *Robotics, vision and control: fundamental algorithms in MATLAB, vol. 73*. Springer Science & Business Media.

Dong, X., Yu, B., Shi, Z., & Zhong, Y. (2015). Time-varying formation control for unmanned aerial vehicles: Theories and applications. *IEEE Transactions on Control Systems Technology*, *23*(1), 340–348.

Elson, J., Girod, L., & Estrin, D. (2002). Fine-grained network time synchronization using reference broadcasts. *Operating Systems Review*, *36*(SI), 147–163.

Gazebo: Robot simulation made easy, http://gazebosim.org/, (last accessed 10th March, 2017).

Guerrero, J. A., Garcia, P. C., & Challal, Y. (2013). Quadrotors formation control. *Journal of Intelligent and Robotic Systems*, *70*(1–4), 221–231.

Hoffmann, G. M., Huang, H., Waslander, S. L., & Tomlin, C. J. (2007). Quadrotor helicopter flight dynamics and control: Theory and experiment. In *Proc. of the AIAA guidance, navigation, and control conference, vol. 2* (p. 4).

Joshi, A., Limbu, N., Ahuja, I., Mulla, A. K., Chung, H., & Chakraborty, D. (2016). Implementation of distributed consensus on an outdoor testbed. In *European control conference (ECC), 2016* (pp. 2146–2151). IEEE.

Joshi, A., Wala, A., Ludhiyani, M., Singh, S., Gagrani, M., Hazra, S., et al. (2017). Implementation of distributed consensus with guaranteed real-time communication on an outdoor quadrotor testbed. In *Conference on decision and control (CDC), 2017* (pp. 2158–2163). IEEE.

Kang, S.-M., Park, M.-C., & Ahn, H.-S. (2017). Distance-based cycle-free persistent formation: Global convergence and experimental test with a group of quadcopters. *IEEE Transactions on Industrial Electronics*, *64*(1), 380–389.

Kushleyev, A., Mellinger, D., Powers, C., & Kumar, V. (2013). Towards a swarm of agile micro quadrotors. *Autonomous Robots*, *35*(4), 287–300.

Lim, H., Park, J., Lee, D., & Kim, H. J. (2012). Build your own quadrotor: Open-source projects on unmanned aerial vehicles. *IEEE Robotics & Automation Magazine*, *19*(3), 33–45.

Lindsey, Q., Mellinger, D., & Kumar, V. (2011). Construction of cubic structures with quadrotor teams. In *Proc. Robotics: Science & Systems VII*.

Lupashin, S., Hehn, M., Mueller, M. W., Schoellig, A. P., Sherback, M., & D'Andrea, R. (2014). A platform for aerial robotics research and demonstration: The flying machine arena. *Mechatronics*, *24*(1), 41–54.

Mahony, R., Kumar, V., & Corke, P. (2012). Multirotor aerial vehicles: Modeling, estimation, and control of quadrotor. *IEEE Robotics & Automation Magazine*, *19*(3), 20–32. http://dx.doi.org/10.1109/MRA.2012.2206474.

Mammadov, E., & Gueaieb, W. (2014). Long-range communication framework for multi-agent autonomous UAVs. In *Unmanned aircraft systems (ICUAS), 2014 international conference on* (pp. 395–403). IEEE.

Meyer, J., Sendobry, A., Kohlbrecher, S., Klingauf, U., & von Stryk, O. (2012). Comprehensive simulation of quadrotor UAVs using ROS and gazebo. In *3rd Int. Conf. on simulation, modeling and programming for autonomous robots (SIMPAR)* (pp. 400–411).

Michael, N., Mellinger, D., Lindsey, Q., & Kumar, V. (2010). The grasp multiple micro-uav testbed. *IEEE Robotics & Automation Magazine*, *17*(3), 56–65.

Michael, N., Schwager, M., Kumar, V., & Rus, D. (2014). An experimental study of time scales and stability in networked multi-robot systems. In *Experimental Robotics* (pp. 631–643). Springer.

Michael, N., Shen, S., Mohta, K., Mulgaonkar, Y., Kumar, V., Nagatani, K., et al. (2012). Collaborative mapping of an earthquake-damaged building via ground and aerial robots. *Journal of Field Robotics*, *29*(5), 832–841.

Moreau, L. (2005). Stability of multiagent systems with time-dependent communication links. *IEEE Transactions on Automatic Control*, *50*(2), 169–182.

Morse, A. S. (1996). Supervisory control of families of linear set-point controllers-part i. exact matching. *IEEE Transactions on Automatic Control*, *41*(10), 1413–1431.

Nelson, W. (1988). Use of circular error probability in target detection, Technical Report, MITRE Corporation, Bedford, MA.

Noureldin, A., Karamat, T. B., & Georgy, J. (2012). *Fundamentals of inertial navigation, satellite-based positioning and their integration*. Springer Science & Business Media.

Olfati-Saber, R., & Murray, R. M. (2004). Consensus problems in networks of agents with switching topology and time-delays. *IEEE Transactions on Automatic Control*, *49*(9), 1520–1533.

Preiss, J. A., Honig, W., Sukhatme, G. S., & Ayanian, N. (2017). Crazyswarm: A large nano-quadcopter swarm. In *Robotics and automation (ICRA), 2017 IEEE international conference on* (pp. 3299–3304). IEEE.

Puri, A. (2005). A survey of unmanned aerial vehicles (UAV) for traffic surveillance. *Department of computer science and engineering, University of South Florida*, 1–29.

Ren, W., & Atkins, E. (2007). Distributed multi-vehicle coordinated control via local information exchange. *International Journal of Robust and Nonlinear Control*, *17*(10–11), 1002–1033.

Ren, W., & Beard, R. W. (2008). *Distributed consensus in multi-vehicle cooperative control*. Springer.

Robot Operating System (ROS), About, http://www.ros.org/about-ros/, (last accessed 10th March, 2017).

Saif, O., Fantoni, I., & Zavala-Río, A. (2015). Real-time flocking of multiple-quadrotor system of systems. In *System of systems engineering conference (SoSE), 2015 10th* (pp. 286–291). IEEE.

Shaw, E., Chung, H., Hedrick, J. K., & Sastry, S. (2007). Unmanned helicopter formation flight experiment for the study of mesh stability. In *Cooperative Systems, vol. 588* (pp. 37–56). Springer.

Turpin, M., Michael, N., & Kumar, V. (2012). Decentralized formation control with variable shapes for aerial robots. In *Robotics and automation (ICRA), 2012 ieee international conference on* (pp. 23–30). IEEE.

ublox LEA-4 series GPS modules, https://www.u-blox.com/en/product/lea-6-series#product-\information, (last accessed 10th March, 2017).

Vásárhelyi, G., Virágh, C., Somorjai, G., Tarcai, N., Szorenyi, T., Nepusz, T., et al. (2014). Outdoor flocking and formation flight with autonomous aerial robots. In *2014 IEEE/RSJ international conference on intelligent robots and systems (IROS 2014)* (pp. 3866–3873).

Wang, Y., & Zhang, F. (2017). *Cooperative control of multi-agent systems: Theory and applications*. John Wiley & Sons.