Self-Triggered Cooperative Path Following Control of Fixed Wing Unmanned Aerial Vehicles

R. Praveen Jain, A. Pedro Aguiar, João Sousa
Department of Electrical and Computer Engineering
Faculty of Engineering, University of Porto, Portugal
{praveenjain, pedro.aguiar, jtasso}@fe.up.pt

***Abstract*— Formation control of multi-robot system may involve extensive inter-robot information exchange. This paper proposes a methodology to reduce the frequency of information exchange in a formation control problem through the use of a self-triggered control strategy for cooperative path following (CPF) problems. In particular, a decentralized, self-triggered CPF controller is developed for fixed-wing Unmanned Aerial Vehicles, where the vehicles are tasked to follow desired geometric paths and keeping a desired formation pattern. Using the Input-to-State Stability framework, we provide guarantees of stability and convergence in the presence of event-based communication. Simulation results are provided to illustrate the efficacy of the developed strategy. It is shown that the self-triggered approach results in significant reduction of information exchange when compared to the conventional time-triggered (periodic) implementation.**

I. Introduction

Multi-Unmanned Aerial Vehicle (UAV) systems find many interesting and challenging applications ranging from civilian to military applications. In particular, formation control is a cooperative control problem of major importance, owing to its applicability in remote sensing, coastal monitoring[1], platooning and cooperative transport [2]. Furthermore, formation control problem of multi-UAV systems is challenging, especially when the underlying system is an under-actuated, nonlinear system such as a fixed wing UAV. One possible strategy to solve the formation control problem is to employ a cascaded, two layered control structure. The lower layer is responsible for the motion control of the individual aerial vehicle, called the *Path Following* (PF) controller [3]. The higher layer is responsible for cooperation among the UAVs called the *Cooperative* controller. This paper adopts the same strategy, termed Cooperative Path Following (CPF) control described in [4] but applied to UAVs.

Interesting applications where the CPF control strategy for UAVs can be used are scenarios of coastal monitoring with particular emphasis in situations where the terrain is dynamically changing making it necessary to have multiple vehicles observing larger areas simultaneously. For example, UAVs carrying an imaging sensor as payload could fly in parallel to acquire data related to a terrain while increasing the area that is being observed/mapped. Fig 1 depicts such scenario where multiple UAVs need to fly in tight formation

This project has received funding from the European Unions Horizon 2020 research and innovation programme under the Marie Sklodowska-Curie grant agreement No 642153.

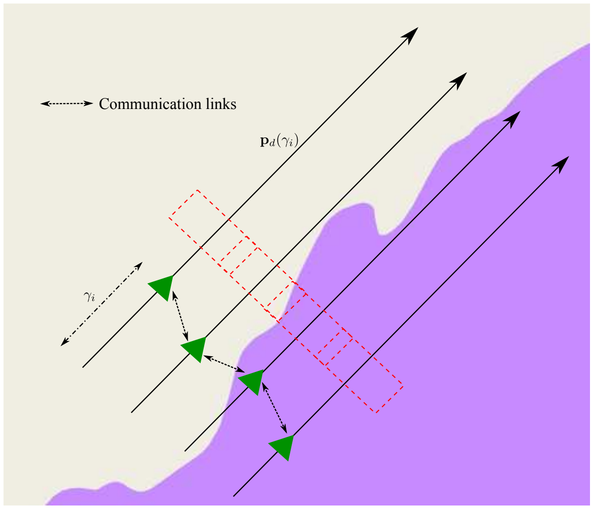


Fig. 1: A typical coastal monitoring scenario where UAVs fly in a formation with overlapping field of view.

to acquire data that could be stitched together for further processing. Additionally such a system can augment the satellite imagery by providing high resolution data of area of interest. Hence it is interesting to investigate the use of multiple UAVs for air borne remote sensing and coastal monitoring applications. In this paper, we address this and other similar problems where the UAVs are required to follow a given path with a desired geometric formation pattern.

Formation control of multi-robot systems involves extensive inter-vehicle information exchange. Most approaches in the literature assume that the information can be exchanged over the network without limitations. Such an assumption is usually not practical. Moreover, the vehicles have an on-board computer (usually the autopilot) which have limited computational power and resources. It is therefore, necessary to develop cooperative motion control methods which take the limitations of the underlying hardware into the consideration. One possible strategy to address the issue of continuous communication is to deviate from the traditional periodic sampling strategies [5], and employ event-based sampling techniques namely, the event-triggered control [6] and self-triggered control [7]. In this paper, we aim to reduce the frequency of information exchange between the UAVs and also the number of controller updates through the use of self-triggered control technique. The self-triggered control strategy is employed at the cooperative control level, whereas the path following controller is implemented using the traditional periodic sampling strategy. It is assumed that the vehicles are connected to one another through bidirectional communication links and modeled as a fixed, undirected communication graph.

Event-based sampling strategies have received great attention owing to its applicability from the viewpoint of the embedded control systems. It is mainly aimed at reducing the frequency of control computations and in the context of networked control systems, the frequency of transmission over the network. In the event-triggered case, this is achieved through the use of a triggering condition which usually is a function of the state of the system. The controller is executed when the triggering condition is satisfied. Clearly, the triggering condition has to be monitored continuously, which implies continuous sensing/communication. This limitation can be mitigated through the use of self-triggered control methods [7]. Significant research has been done and results are available for standalone systems. See [8] and the references therein for an extensive survey. However, decentralized event-based control in the context of multi-agent systems is still an open research topic. Most of the results have been limited to the first order consensus problem. One such pioneering work is presented in [9], [10]. Other works include [11], [12], where a decentralized, time dependent triggering condition is employed. Clearly, the problem of formation control, more specifically, CPF control of UAVs, in an event-based control framework provides plenty of scope for research. In this paper, a self-triggered, decentralized, CPF controller is developed for multiple UAVs. Our work builds on the results presented in [3], [4], [9], [13]. Simulation results are provided to illustrate the efficacy of the proposed approach. It is shown that the proposed self-triggered CPF control strategy results in significant reduction of control updates and frequency of information exchange over network when compared to the conventional time-triggered implementation.

Additionally, we also provide formal guarantees of stability and convergence. More precisely, using the Input-to-State Stability (ISS) framework [14], we show that the Path following and Cooperative control subsystems can be viewed as cascade of two ISS systems and hence the overall system is ISS.

The rest of the paper is organized as follows. Section II describes the model used for control design and a Lyapunov based Path Following controller for a fixed wing UAV in the presence of state estimation errors. Section III proposes the event-triggered Cooperative controller and shows ISS with respect to the measurement errors arising out of event-based communication between the UAV. Section IV, combines the results of previous two subsections and show that the overall CPF is ISS in presence of UAV internal dynamics, estimation errors and measurement errors arising out of the triggering condition of the event-triggered controller. Simulation results are provided and discussed in Section V followed by conclusions in Section VI.

Notations and Definitions: The euclidean norm and the induced matrix norm is represented as $\| \, . \, \|$. Set of non-negative integers is represented as $\mathbb{Z}_{\geq 0}$. The Kronecker product is denoted by $\otimes$. A continuous function $\alpha : [0, a) \to [0, \infty)$ is said to belong to class $\mathcal{K}$ if it is strictly increasing and $\alpha(0) = 0$. Additionally it is said to belong to class $\mathcal{K}_\infty$ if $a = \infty$ and $\alpha(r) \to \infty$ as $r \to \infty$. A continuous function $\beta : [0, a) \times [0, \infty) \to [0, \infty)$ is said to belong to class $\mathcal{KL}$, if for each fixed s, the mapping $\beta(r, s)$ belongs to class $\mathcal{K}$ with respect to r and, for each fixed r, the mapping $\beta(r, s)$ is decreasing with respect to s and $\beta(r, s) \to 0$ as $s \to \infty$. The system $\dot{x} = f(t, x, u)$ where $f : [0, \infty) \times \mathbb{R}^n \times \mathbb{R}^m \to \mathbb{R}^n$ is piecewise continuous in t and locally Lipschitz in x and u, is said to be Input-to-State Stable (ISS) if there exist a class $\mathcal{KL}$ function β and a class $\mathcal{K}$ function γ such that for any initial time t_0, initial state $x(t_0)$ and any bounded input $u(t)$, the solution $x(t)$ exists for all $t \geq t_0$ and satisfies

$$\|x(t)\| \leq \beta(\|x(t_0)\|, t - t_0) + \gamma \left(\sup_{t_0 \leq \tau \leq t} \|u(\tau)\| \right) \tag{1}$$

II. UAV Model and Path Following Control

In this section, we formulate and present a nonlinear controller which solves the Path Following (PF) for a fixed wing UAV. In the following we briefly describe the UAV model used for controller design, followed by problem formulation and controller design.

A. UAV Model

For control design purpose, a kinematic model of the fixed-wing UAV is used and it is assumed that the autopilot on-board the aircraft is able to track the reference control commands generated by the PF controller. We use the planar UAV kinematic model presented in [15] and inspired by [3], [16] develop a path following controller for 2D case assuming that the UAV maintains a constant altitude during flight. Consider two right handed frame of references, namely, the Inertial frame $\{I\}$ which is a fixed reference frame and a Body frame $\{B\}$ which is attached to the UAV, with its x-axis pointing towards the direction of travel (longitudinal direction). The UAV kinematic model is then given by

$$\begin{aligned} \dot{\mathbf{p}}(t) &= R(\psi)\mathbf{v}(t) \\ \dot{R}(\psi) &= R(\psi)S(\omega) \end{aligned} \tag{2}$$

$$\omega = \frac{g \tan \phi_r}{v_f} \tag{3}$$

where $\mathbf{p}(t) \in \mathbb{R}^2$ is the 2D position of the UAV i.e., the origin of $\{B\}$ with respect to $\{I\}$, $\mathbf{v}(t) = [v_f(t), 0]^T$ is the input velocity vector expressed in the $\{B\}$ frame. It can be noted that the longitudinal velocity is denoted by v_f, whereas the lateral velocity is 0. The matrix $R(\psi) \in SO(2)$ is the rotation matrix parameterized with the yaw angle ψ. $S(\omega(t)) \in so(2)$ is a skew symmetric matrix with input angular velocity $\omega \in \mathbb{R}$. The control inputs for the vehicle are $\mathbf{u}(t) = [v_f, \phi_r]^T$. The yaw rate is provided by the reference roll angle ϕ_r through the static map given by (3) and g is acceleration due to gravity. Considering ω to be the intermediate control input to be designed, the references for the roll command can be obtained by inverting the static map (3). The static map (3) is invertible assuming that the airspeed command of the aircraft v_f is non-zero. Such an assumption is valid for a fixed-wing aircraft in flight.

Remark 1. The planar UAV kinematics (2) is reduced to the well known unicycle kinematics by considering the intermediate control input to be ω and using (3) to obtain roll reference command ϕ_r. The control design for an UAV can then proceed as though a controller is being designed for an unicycle robot.

B. Problem Formulation

The Self-triggered CPF problem can be decomposed into two specific sub-problems, namely, the Path Following (PF) problem (solved in the current section) and the Self-triggered Cooperative Control (CC) problem (described in the next section).

Problem 1 (Path Following). Consider a given reference geometric path $\mathbf{p}_d(\gamma) : \mathbb{R} \to \mathbb{R}^2$ parameterized by the path variable $\gamma \in \mathbb{R}$, with a desired speed assignment $v_d(\gamma) \in \mathbb{R}$. The path following problem is to design a feedback control law $\mathbf{u}(t)$ such that the path following error, $\|\mathbf{p} - \mathbf{p}_d(\gamma)\|$ converges to an arbitrary small neighborhood of the origin as $t \to \infty$. Furthermore, the UAV has to satisfy the desired speed assignment, $\|\dot{\gamma} - v_d(\gamma)\| \to 0$ as $t \to \infty$.

Remark 2. The progression of γ is equivalent to the progression of a virtual vehicle along the given path. The control objective is to steer the actual vehicle towards this virtual vehicle that is conveniently moving through the path, thereby solving the path following problem.

C. Path Following Control Design

We follow the controller presented in [16] and define an error variable $\mathbf{e} = R^T(\theta)(\mathbf{p} - \mathbf{p}_d(\gamma)) - \boldsymbol{\epsilon}$, where $\epsilon = [\epsilon_1 \quad \epsilon_2]^T$ is a given small vector (see Remark 3). The error dynamics of the path following system is given by

$$\begin{aligned}\dot{\mathbf{e}} &= \dot{R}^T(\theta)(\mathbf{p} - \mathbf{p}_d(\gamma)) + R^T(\theta)(\dot{\mathbf{p}} - \dot{\mathbf{p}}_d(\gamma)) \qquad (4)\\ &= -S(\omega)\mathbf{e} + \Delta\mathbf{u} - R^T(\theta)\dot{\mathbf{p}}_d(\gamma)\\ &= -S(\omega)\mathbf{e} + \Delta\mathbf{u} - R^T(\theta)\frac{\partial \mathbf{p}_d(\gamma)}{\partial \gamma}\left[v_d(\gamma) + v_r\right]\end{aligned}$$

where $\Delta = \begin{bmatrix} 1 & -\epsilon_2 \\ 0 & \epsilon_1 \end{bmatrix}$ and we have imposed the following condition (which will be explained in the next section) for the dynamics of path parameter γ

$$\dot{\gamma} = v_d(\gamma) + v_r \qquad (5)$$

where v_r is the formation speed actuation signal that will be viewed as an input control signal for the coordination system. Consider now the realistic situation that $\mathbf{e}(t)$ is not precisely known, but through the UAV sensors it is possible to obtain an estimate of $\mathbf{e}(t)$, denoted as $\hat{\mathbf{e}}(t)$. Let $\tilde{\mathbf{e}} = \hat{\mathbf{e}} - \mathbf{e}$ be the estimation error and assume that $\boldsymbol{\epsilon}$ is selected such that Δ is invertible and the term $|\frac{\partial \mathbf{p}_d}{\partial \gamma}|$ is bounded. Then, the following result holds.

Theorem 1 (Path Following). *Given the error dynamics for the path following system described by* (4)*, the control law*

$$\mathbf{u} = \Delta^{-1}\left(-K_p\hat{\mathbf{e}} + R^T(\theta)\frac{\partial \mathbf{p}_d(\gamma)}{\partial \gamma}v_d(\gamma)\right) \qquad (6)$$

makes the closed-loop system Input-to-State Stable (ISS) with respect to the estimation error $\tilde{\mathbf{e}}(t)$ and the formation speed actuation signal $v_r(t)$. Moreover, there exists class $\mathcal{K}$ functions $\alpha_1, \alpha_2, \alpha_3, \alpha_4$ and an ISS Lyapunov function $V(\mathbf{e})$ such that

$$\alpha_1(\|\mathbf{e}\|) \le V(\mathbf{e}) \le \alpha_2(\|\mathbf{e}\|) \qquad (7)$$

$$\dot{V}(\mathbf{e}) \le -\alpha_3(\mathbf{e}) \;\; \forall \;\; \|\mathbf{e}\| \ge \alpha_4(\|\mathbf{d}\|) > 0 \qquad (8)$$

where $\mathbf{d} = [\tilde{\mathbf{e}}^T \quad v_r]^T$.

Proof. Consider the ISS Lyapunov function

$$V(\mathbf{e}) = \frac{1}{2}\mathbf{e}^T\mathbf{e}$$

Taking the time derivative and using (4) - (6), we have

$$\begin{aligned}\dot{V} &= \mathbf{e}^T\dot{\mathbf{e}}\\ &= \mathbf{e}^T\left(-S(\omega)\mathbf{e} + \Delta\mathbf{u} - R^T(\theta)\frac{\partial \mathbf{p}_d(\gamma)}{\partial \gamma}\left[v_d(\gamma) + v_r\right]\right)\\ &= -\mathbf{e}^TS(\omega)\mathbf{e} - \mathbf{e}^TK_p\hat{\mathbf{e}} - \mathbf{e}^TR^T(\theta)\frac{\partial \mathbf{p}_d(\gamma)}{\partial \gamma}v_r\\ &= -\mathbf{e}^TK_p(\mathbf{e} + \tilde{\mathbf{e}}) - \mathbf{e}^TR^T(\theta)\frac{\partial \mathbf{p}_d(\gamma)}{\partial \gamma}v_r\\ &= -\mathbf{e}^TK_p\mathbf{e} - \mathbf{e}^T\bar{K}\mathbf{d}\\ &= -\mathbf{e}^TK_p\mathbf{e} + \beta\mathbf{e}^T\mathbf{e} - \beta\mathbf{e}^T\mathbf{e} - \mathbf{e}^T\bar{K}\mathbf{d}\\ &\le -\lambda_{\min}(K_p - \beta I)\|\mathbf{e}\|^2 \;\; \forall \;\; \|\mathbf{e}\| \ge \frac{\|\bar{K}\|\|\mathbf{d}\|}{\beta}\end{aligned}$$

where $\beta > 0$ is a constant chosen such that $K_p - \beta I > 0$, $\bar{K} = \begin{bmatrix} K_p & R^T(\theta)\frac{\partial \mathbf{p}_d}{\partial \gamma} \end{bmatrix}$ and $\mathbf{d} = \begin{bmatrix} \tilde{\mathbf{e}} \\ v_r \end{bmatrix}$. Consequently, the system (4) with controller (6) is ISS with respect to $\mathbf{d}$ (Theorem 4.19, [17]). $\square$

Remark 3. It must be noted that if $\boldsymbol{\epsilon} = 0$, we do not have direct control over the heading (and hence the roll angle) of the UAV i.e., the control input ω does not appear in the error dynamics (4). Hence, it is necessary to have the condition $\boldsymbol{\epsilon} \neq 0$ satisfied for this controller. Note however that when the error $\mathbf{e}(t)$ converges to zero, this implies that $\|\mathbf{p} - \mathbf{p}_d(\gamma)\| \to \|\boldsymbol{\epsilon}\|$ provided that $\|\mathbf{d}\| \to 0$ as $t \to \infty$.

III. Self-triggered Cooperative Controller

In this section, we develop an algorithm for implementation of self-triggered consensus controller. In order to develop a self-triggered controller, we follow the work of [13] and first design an event-triggered controller followed by the development of a self-triggered implementation from the triggering condition. However, we differ in the convergence analysis of [13] where here we prove that the event-triggered controller is ISS with respect to the measurement errors. The background information necessary for derivation of results for the self-triggered cooperative controller is presented in the following subsection followed by the problem formulation. Finally, we conclude the section with the event-based cooperative control design.

A. Graph Theory and Consensus Protocol

Graph theory is used extensively in the literature to model the underlying communication network among the group of robots. References [18], [19] form a good source for results on graph theory and their application for cooperative control and consensus in multi-agent systems. An undirected graph is defined as a pair $G = (V, E)$, where $V = \{v_i | i = 1, \cdots, N\}$ is the vertex set with each vertex representing a robot in a multi-robot system. The set $E \subseteq V \times V = \{(v_i, v_j) | i, j = 1, 2, \cdots, N \text{ and } i \neq j\}$ is called the Edge set. Vertices v_i and v_j are said to be adjacent if $(v_i, v_j) \in E$, that is, v_i can communicate with v_j and vice versa in case of undirected graphs. The adjacency relations are represented by the *adjacency matrix* $A \in \mathbb{R}^{N \times N}$, defined as

$$a_{ij} := \begin{cases} 1 & \text{if } (v_i, v_j) \in E \\ 0 & \text{otherwise} \end{cases}$$

The neighborhood of the vertex v_i is defined as the set $N_i = \{v_j \in V | (v_i, v_j) \in E\}$ and the cardinality of the neighborhood set is the degree of the given vertex, $d(v_i)$. The degree matrix, D of the graph G is a diagonal matrix, containing the vertex-degrees of G on the diagonal. The graph Laplacian L defined as $L = D - A$, is one of the important matrices used in the consensus algorithms described in the following subsection. For an undirected graph, the Laplacian is symmetric and positive semi-definite; hence its real eigenvalues can be ordered as $\lambda_1(L) \leq \lambda_2(L) \leq \cdots \leq \lambda_N(L)$. A graph is connected if $\lambda_1(L) = 0$ and $\lambda_2(L) > 0$.

Consider a set of N agents with single integrator dynamics

$$\dot{\gamma}_i = u_i \; \forall \; i = 1, \cdots, N \tag{9}$$

where $\gamma \in \mathbb{R}$ and $u \in \mathbb{R}$. Furthermore, assume that these agents are connected to each other over a static network modeled as an undirected graph G. The following theorem from [18] provides the result for consensus.

Theorem 2 (Consensus)**.** *The set of N agents modeled as* (9) *converge to an agreement set defined by $\mathcal{A} = \{\gamma \in \mathbb{R}^N | \gamma_i = \gamma_j \; \forall i, j\}$ under the consensus protocol*

$$\dot{\gamma}_i = -\sum_{j \in N_i} (\gamma_i - \gamma_j), \quad i = 1, \cdots, N \tag{10}$$

only if the graph G is connected. Moreover, the rate of convergence is dictated by $\lambda_2(L)$.

Remark 4. The consensus protocol (10), is a distributed law and depends only on the agent's own measurements and that of its neighbors. For the purpose of analysis, the agent states can be stacked together into a single vector $\boldsymbol{\gamma} = [\gamma_1, \gamma_2, \cdots, \gamma_N]^T$. The consensus protocol (10) can equivalently be written as

$$\dot{\boldsymbol{\gamma}} = -L\boldsymbol{\gamma} \tag{11}$$

where L is graph Laplacian.

Remark 5. The agents under the consensus protocol (10) or (11) converges to the initial average of the states i.e., $\boldsymbol{\gamma}(t) \to \frac{\mathbf{1}^T \boldsymbol{\gamma}(0)}{N} \mathbf{1}$ as $t \to \infty$, where vector $\mathbf{1}$ is a N-dimensional vector with all entries valued at 1 and corresponds to the eigenvector of $\lambda_1(L) = 0$.

B. Problem Formulation

Problem 2 (Self-triggered Cooperative Control). Consider a group of N vehicles and associate with each one, a desired reference path $\mathbf{p}_d^i(\gamma_i)$ parameterized by γ_i for $i = 1, 2, \cdots, N$. It is assumed that each vehicle is able to follow the specified reference path using a path following controller. The cooperative control objective consists in designing a decentralized, self-triggered control law such that the position of the virtual vehicles along the reference path maintains a desired formation. Mathematically, the objective can be specified as $\|\gamma_i - \gamma_j\| \to 0$ for all $i, j = 1, \cdots, N$ and $i \neq j$ as $t \to \infty$. Additionally, we require that the speed of the virtual vehicle matches the desired formation speed assignment $v_d(\gamma_i)$ i.e., $\|\dot{\gamma}_i - v_d(\gamma_i)\| \to 0$ for all $i = 1, \cdots, N$ as $t \to \infty$.

In order to achieve this objective, we propose to set the evolution of γ_i as

$$\dot{\gamma}_i = v_d(\gamma_i) + v_r^i \tag{12}$$

where $v_r^i(t)$ is a new control variable. The control objective is to compute the corrective action $v_r^i(t) = v_r^i(t_k^i)$ for all $t \in \bigcup_{k \in \mathbb{Z}_{\geq 0}} [t_k^i, t_{k+1}^i)$ where t_k^i is the time instant at which event occurs for agent i.

Additionally, another objective is to compute the next time instant t_{k+1}^i at which the event should occur, thereby triggering the controller update and transmission over the network. The candidate event time can be selected as

$$t_{k+1}^i = t_k^i + \max\{\tau_k^i, b_i\} \tag{13}$$

where b_i is a lower bound and hence needs to be positive in order to have a zeno free computation of next event time instant. τ_k^i is computed later in this section, such that the stability of the system is not compromised.

C. ISS of the Event-triggered Cooperative Controller

We start with design of decentralized, event triggered consensus controller similar to the one presented in [20] and prove that the result holds in presence of a desired formation speed specification (12). Additionally, we also prove the ISS of the event-triggered consensus controller with respect to the measurement error arising out of the triggering condition. Given the single integrator model (9), the distributed consensus protocol can be written as,

$$\begin{aligned} \dot{\gamma}_i &= -\sum_{j \in N_i} (\gamma_i - \gamma_j) \quad \forall \quad i = 1, \cdots, N \\ &= -q_i(t) \end{aligned}$$

Introduce the event time for the agent i as t_k^i and a measurement error, $\tilde{q}_i(t) = q_i(t_k^i) - q(t)$ for all $t \in \bigcup_{k\in\mathbb{Z}_{\geq 0}} [t_k^i, t_{k+1}^i)$. The event-triggered controller for agent i is

$$u_i(t) = -q_i(t_k^i) \quad \forall \quad t \in \bigcup_{k\in\mathbb{Z}_{\geq 0}} [t_k^i, t_{k+1}^i) \tag{14}$$

In order to prove the ISS of the event-triggered controller, define the disagreement vector [19], as[1]

$$\boldsymbol{\delta} = \boldsymbol{\gamma} - \alpha\mathbf{1} \tag{15}$$

where $\alpha = (1/N)\mathbf{1}^T\boldsymbol{\gamma}$ is an invariant for the consensus problem and $\mathbf{1}^T\boldsymbol{\delta} = 0$. Consider the vector of states and measurement error stacked together, $\boldsymbol{\gamma} = [\gamma_1, \gamma_2, \cdots, \gamma_N]^T$, $\mathbf{q} = [q_1, q_2, \cdots, q_N]^T$ and $\tilde{\mathbf{q}} = [\tilde{q}_1, \tilde{q}_2, \cdots, \tilde{q}_N]^T$. The disagreement vector dynamics satisfies

$$\begin{aligned}\dot{\boldsymbol{\delta}} &= \dot{\boldsymbol{\gamma}} - \frac{1}{N}\mathbf{1}^T\dot{\boldsymbol{\gamma}}\mathbf{1} \\ &= -(\mathbf{q} + \tilde{\mathbf{q}}) + \frac{1}{N}\mathbf{1}^T(\mathbf{q} + \tilde{\mathbf{q}})\mathbf{1} \\ &= -(\mathbf{q} + \tilde{\mathbf{q}}) + \bar{q}\mathbf{1}\end{aligned} \tag{16}$$

where $\bar{q} = (1/N)\mathbf{1}^T\tilde{\mathbf{q}}$ and $\mathbf{1}^T\mathbf{q} = 0$ since $\mathbf{q} = L\boldsymbol{\gamma}$.

Remark 6. In fact it is easy to see that α is an invariant quantity. Differentiating α we have $\dot{\alpha} = \frac{1}{N}\mathbf{1}^T\dot{\boldsymbol{\gamma}} = -\frac{1}{N}\mathbf{1}^T L\boldsymbol{\gamma} = 0$.

Remark 7. Since α is an invariant and $L\mathbf{1} = 0$, the relation $\mathbf{q} = L\boldsymbol{\delta}$ holds.

Lemma 1 (Algebraic connectivity of graphs)**.** *Let G be an undirected graph with Laplacian L, then, $\lambda_2 = \min_{\mathbf{1}^T\boldsymbol{\delta}=0} \frac{\boldsymbol{\delta}^T L\boldsymbol{\delta}}{\boldsymbol{\delta}^T\boldsymbol{\delta}}$ with $\lambda_2 = \lambda_2(L)$, i.e,*

$$\boldsymbol{\delta}^T L\boldsymbol{\delta} \geq \lambda_2\|\boldsymbol{\delta}\|^2$$

Similarly, $\lambda_{\max} = \max_{\boldsymbol{\delta}\neq 0} \frac{\boldsymbol{\delta}^T L\boldsymbol{\delta}}{\boldsymbol{\delta}^T\boldsymbol{\delta}}$ with $\lambda_{\max} = \lambda_{\max}(L)$ i.e.,

$$\boldsymbol{\delta}^T L\boldsymbol{\delta} \leq \lambda_{\max}\|\boldsymbol{\delta}\|^2$$

Proof. Since the Graph Laplacian L is a symmetric matrix, the proof follows from direct application of Courant-Fischer theorem from [21]. □

Theorem 3 (Event-triggered Consensus)**.** *The event triggered consensus controller* (14) *achieves consensus for the system* (9)*, given the triggering condition*

$$\|\tilde{q}_i(t)\| \leq \beta_i\|q_i(t)\| \tag{17}$$

for $0 < \beta_i < 1$. Additionally, the controller is ISS with respect to the measurement error $\tilde{q}_i(t)$.

Proof. Consider the candidate ISS Lyapunov function

$$V_{\text{CC}}(\boldsymbol{\delta}) = \frac{1}{2}\boldsymbol{\delta}^T\boldsymbol{\delta}$$

[1]The notation of continuous dependence variables on time is dropped for the sake of brevity.

Taking the derivative with respect to time, using (16) and the relation $\mathbf{q} = L\boldsymbol{\delta}$, yields

$$\begin{aligned}\dot{V}_{\text{CC}}(\boldsymbol{\delta}) &= \boldsymbol{\delta}^T\dot{\boldsymbol{\delta}} \\ &= \boldsymbol{\delta}^T\left[-(\mathbf{q} + \tilde{\mathbf{q}}) + \bar{q}\mathbf{1}\right] \\ &= -\boldsymbol{\delta}^T L\boldsymbol{\delta} - \boldsymbol{\delta}^T\tilde{\mathbf{q}} \\ &= -\boldsymbol{\delta}^T L\boldsymbol{\delta} + \beta\boldsymbol{\delta}^T L\boldsymbol{\delta} - \beta\boldsymbol{\delta}^T L\boldsymbol{\delta} - \boldsymbol{\delta}^T\tilde{\mathbf{q}} \\ &= -(1-\beta)\boldsymbol{\delta}^T L\boldsymbol{\delta} - \beta\boldsymbol{\delta}^T L\boldsymbol{\delta} - \boldsymbol{\delta}^T\tilde{\mathbf{q}} \\ &\leq -(1-\beta)\lambda_2(L)\|\boldsymbol{\delta}\|^2 \;\; \forall \;\; -\boldsymbol{\delta}^T\tilde{\mathbf{q}} \leq \beta\boldsymbol{\delta}^T L\boldsymbol{\delta} \\ &\leq -\lambda_{\text{CC}}V_{\text{CC}}(\boldsymbol{\delta}) \quad \forall \quad \|\tilde{\mathbf{q}}\| \leq \beta\|L\boldsymbol{\delta}\| = \beta\|\mathbf{q}\|\end{aligned}$$

where $0 < \beta < 1$ and $\lambda_{\text{CC}} = (1-\beta)2\lambda_2(L)$. Consequently, we can conclude ISS of system (9) with the event triggered consensus controller (14) given the event triggering condition $\|\tilde{\mathbf{q}}\| \leq \beta\|L\boldsymbol{\delta}\| = \beta\|\mathbf{q}\|$. Since the objective is to derive the decentralized controller and hence the triggering condition, the centralized triggering condition can be written in decentralized manner for each agent i as

$$\|\tilde{q}_i(t)\| \leq \beta_i\|q_i(t)\| = \beta_i\|(L\boldsymbol{\delta})_i\|$$

where $(L\boldsymbol{\delta})_i$ is i^{th} component of the vector $L\boldsymbol{\delta}$. β_i can be chosen as β/N. Hence the system (9) under the decentralized event-triggered consensus controller (14) is ISS with respect to the measurement error $\tilde{q}_i(t)$. □

Given the result for the event-triggered consensus, the cooperative controller that solves the event-triggered version of the Problem 2 will be developed in the following. Note that the cooperative control problem is a consensus problem with an additional desired speed assignment for each agent.

Proposition 1. *Given the dynamics of the path variable* (12)*, the event-triggered, decentralized, cooperative control law*

$$v_r^i(t) = -\sum_{j\in N_i}(\gamma_i(t_k^i) - \gamma_j(t_k^i)) \tag{18}$$

defined over $t \in \bigcup_{k\in\mathbb{Z}_{\geq 0}}[t_k^i, t_{k+1}^i)$ along with the triggering condition (17) *solves the Problem 2 and does so exponentially fast. Additionally, the disagreement error $\boldsymbol{\delta}(t)$ is ISS with respect to the measurement error $\tilde{\mathbf{q}}(t)$*

Proof. Consider the dynamics of the path variable given by

$$\dot{\boldsymbol{\gamma}} = v_d(\gamma)\mathbf{1} + \mathbf{v_r}$$

The cooperative control law (18) can be written in compact form as

$$\mathbf{v_r} = -\mathbf{q}(t_k) = -(\mathbf{q} + \tilde{\mathbf{q}})$$

Consequently, the dynamics of the path variable satisfies

$$\dot{\boldsymbol{\gamma}} = v_d(\gamma)\mathbf{1} - (\mathbf{q} + \tilde{\mathbf{q}})$$

The dynamics of the disagreement vector $\boldsymbol{\delta}$ in presence of desired speed assignment is then given as

$$\begin{aligned}\dot{\boldsymbol{\delta}}(t) &= \dot{\boldsymbol{\gamma}} - \frac{1}{N}\mathbf{1}^T\dot{\boldsymbol{\gamma}}\mathbf{1} \\ &= v_d(\gamma)\mathbf{1} - (\mathbf{q}+\tilde{\mathbf{q}}) - \frac{1}{N}\mathbf{1}^T\left[v_d(\gamma)\mathbf{1} - (\mathbf{q}+\tilde{\mathbf{q}})\right]\mathbf{1} \\ &= v_d(\gamma)\mathbf{1} - (\mathbf{q}+\tilde{\mathbf{q}}) - v_d(\gamma)\mathbf{1} + \frac{1}{N}\mathbf{1}^T(\mathbf{q}+\tilde{\mathbf{q}})\mathbf{1} \\ &= -(\mathbf{q}+\tilde{\mathbf{q}}) + \bar{q}\mathbf{1}\end{aligned}$$

Clearly the disagreement vector dynamics remain unchanged in the presence of the desired speed assignment $v_d(\gamma)$. Hence, using the Lyapunov function $(1/2)\boldsymbol{\delta}^T\boldsymbol{\delta}$ and following the proof of Theorem 3, we can conclude this proposition. □

D. Self-triggered Cooperative Control Algorithm

From the derived results of decentralized event-triggered consensus, where we have shown that consensus holds in presence of a desired formation speed assignment, the event triggering condition (17) can now be used to compute the next possible event-time instant, which is necessary to have a self-triggered controller implementation. We follow the steps presented in [13]. As mentioned earlier, the problem of computation of the next event time instant reduces to the computation of upper and lower bound to the inter event time. Using (13) to compute the next event time, eliminates the possibility of exhibition of zeno behavior by any agent i. The inter-event time is bounded from below by a strictly positive value b_i. The lower bound is computed as $b_i = \frac{\beta}{N(1+\beta)}$, see [13] for detailed computation.

In order to compute the next possible event time τ_k^i, consider the time evolution of $\tilde{q}_i(t)$ and $q_i(t)$ for each agent i over the time interval $[t_k^i, t_{k+1}^i)$. From the definition of $q_i(t)$, it is clear that it is differentiable and satisfies

$$\begin{aligned}\dot{q}_i(t) &= -\sum_{j\in N_i}\dot{\gamma}_i(t) - \dot{\gamma}_j(t) \\ &= -\sum_{j\in N_i} q_i(t^i_{k_i(t)}) - q_j(t^j_{k_j(t)}) \\ &= \xi_i(t)\end{aligned}$$

where $k_i(t) = \arg\max_{k\in\mathbb{Z}_{\geq 0}}\{t_k^i | t_k^i \leq t\}$, $q_i(t^i_{k_i(t)})$ is the value of the state q_i of agent i at the latest event time $t^i_{k_i(t)}$ before the current time t and $\xi_i(t)$ is an auxiliary variable. Computation of τ_k^i is based on the information available to the agent i in the interval $[t_k^i, t_{k+1}^i)$. From the definition of $\tilde{q}_i(t)$, we know that it is continuous in the interval $[t_k^i, t_{k+1}^i)$ and $\dot{\tilde{q}}_i(t) = -\dot{q}_i(t) = -\xi_i(t)$. Since $\tilde{q}_i(t_k^i) = 0$, we can compute $\tilde{q}_i(t)$ as

$$\tilde{q}_i(t) = -\int_{t_k^i}^{t}\xi_i(s)ds \quad \forall \quad t > t_k^i \tag{19}$$

Consequently, $q_i(t)$ can be computed as

$$q_i(t) = q_i(t_k^i) + \int_{t_k^i}^{t}\xi_i(s)ds \quad \forall \quad t > t_k^i \tag{20}$$

The key issue in determining the next event time instant $t_{k+1}^i = t_k^i + \max\{b_i, \tau_k^i\}$ and hence τ_k^i is computation of the term $\Xi(t) := \int_{t_k^i}^{t}\xi_i(s)ds$. Note that $\xi_i(t)$ is always piecewise constant and following the procedure of [13], we obtain the following equations to compute $\Xi(t)$

$$\Xi(t) = \sum_{l=1}^{p}\xi_i(\tau_{l-1})(\tau_l - \tau_{l-1}) + \xi_i(\tau_p)(t-\tau_p) \tag{21}$$

The term $\Xi(t)$ is written as

$$\Xi(t) = \Xi_p + \xi_i(\tau_p)(t-\tau_p) \tag{22}$$

where $\Xi_0(t) = 0$, $\Xi_p = \sum_{l=1}^{p}\xi_i(\tau_{l-1})(\tau_l - \tau_{l-1})$ and τ_p captures the latest time instant at which any measurement is received from the neighbor of vehicle i. Consequently, the variables $q_i(t)$ and $\tilde{q}_i(t)$ can be computed using (20) and (19) respectively. The event triggering condition $\|\tilde{q}_i\| \leq \beta_i\|q_i\|$ for each agent i equivalent to

$$\|\Xi(t)\| \leq \beta_i\|q_i(t_k^i) + \Xi(t)\| \tag{23}$$

The next event time is defined as

$$t_{k+1}^i = \inf\{t > t_k^i | \quad \|\Xi(t)\| = \beta_i\|q_i(t_k^i) + \Xi(t)\|\} \tag{24}$$

In order to avoid repeated checking of the triggering condition, let $y = t - \tau_p$ and $\xi_p = \xi_i(\tau_p)$, then

$$\|\Xi_p + \xi_p y\| = \beta_i\|q_i(t_k^i) + \Xi_p + \xi_p y\| \tag{25}$$

After some algebraic manipulations, the triggering condition given above can be written as a quadratic equation of y as

$$ay^2 + by + c = 0$$

where

$$\begin{aligned}a &= (1-\beta_i^2)\xi_p^T\xi_p \\ b &= 2\xi_p^T\Xi_p - 2\beta_i^2\xi_p^T(q_i(t_k^i) + \Xi_p) \\ c &= \Xi_p^T\Xi_p - \beta_i^2(q_i(t_k^i)+\Xi_p)^T(q_i(t_k^i)+\Xi_p)\end{aligned}$$

In [13], it is shown that the above given quadratic equation has exactly one positive solution given by $y_p = (-b + \sqrt{b^2-4ac})/2a$ provided that $\xi_p \neq 0$. The next candidate event time is then given as $t_{k+1}^i = y_p + \tau_p$.

At each event time t_k^i, agent i receives its neighbor's states $\gamma_j, j \in N_i$ and computes the control input $v_r^i(t)$ according to the equation (18). It also computes the next event time t_{k+1}^i according to equation (13) using the computed average $q_i(t_k^i)$ and previously received $q_j(t^j_{k_j(t)})$. Finally it transmits over the network, $q_i(t_k^i)$ which would be used by its neighbors for control computation and event generation. Consequently at each event there are $2N_i + 1$ data exchanges on each agent. The iterative algorithm used to implement the self triggered controller is given in [13] and requires that the receivers on the UAVs are always active.

IV. Self Triggered CPF

In Section II we have proved ISS of the Path following system in presence of estimation errors and in Section III, the ISS of the event-based cooperative controller. In this section, we present the main result of the paper, where we view the two subsystems (PF and CC system) as a cascade of two ISS systems. To this end, consider N Path Following subsystems, each one ISS with respect to the inputs as shown in Section II, then we can construct a single Lyapunov function $V_{\mathrm{PF}}(\boldsymbol{\varepsilon})$ by stacking the error states into a vector $\boldsymbol{\varepsilon} = [\mathbf{e}_1^T, \cdots, \mathbf{e}_N^T]^T$. Similarly define the estimation error $\tilde{\boldsymbol{\varepsilon}}$. The single, combined ISS Lyapunov function for the Path Following subsystems can be written as

$$V_{\mathrm{PF}}(\boldsymbol{\varepsilon}) = \frac{1}{2}\boldsymbol{\varepsilon}^T\boldsymbol{\varepsilon} \tag{26}$$

which satisfies

$$\dot{V}_{\mathrm{PF}}(\boldsymbol{\varepsilon}) \leq -\lambda_{\mathrm{PF}} V_{\mathrm{PF}}(\boldsymbol{\varepsilon}) \;\; \forall \;\; \|\boldsymbol{\varepsilon}\| \geq \frac{\|I_N \otimes \bar{K}\| \|\mathbf{d}\|}{\beta} \tag{27}$$

where $\lambda_{\mathrm{PF}} = 2\lambda_{\min}(K_p - \beta I)$. Clearly the combined path following system is still ISS with respect to the input perturbations $\mathbf{d} = [\mathbf{d}_1^T, \cdots, \mathbf{d}_N^T]^T$. The main result of this paper is stated in the following theorem.

Theorem 4. *The path following controller* (6) *along with the self-triggered cooperative controller of equation* (18)*, collectively termed as the self-triggered CPF controller makes the system with dynamics* (4) *and* (12) *ISS with respect to* $\mathbf{d}$. *Additionally the cooperative controllers are executed in decentralized fashion on each agent at time instants computed by* (13).

Proof. The proof follows directly from the results of Theorem 1 and Proposition 1, since the overall CPF system can be viewed as cascaded interconnection of two ISS subsystems. More precisely, construct a combined Lyapunov function

$$V(\boldsymbol{\varepsilon}, \boldsymbol{\delta}) = \frac{1}{2}\boldsymbol{\varepsilon}^T\boldsymbol{\varepsilon} + \frac{1}{2}\boldsymbol{\delta}^T\boldsymbol{\delta}$$

Taking the time derivative and proceeding as before we have

$$\dot{V}(\boldsymbol{\varepsilon}, \boldsymbol{\delta}) \leq -\lambda_{\mathrm{PF}} V_{\mathrm{PF}} - \lambda_{\mathrm{CC}} V_{\mathrm{CC}} \;\; \forall \;\; \|\boldsymbol{\varepsilon}\| \geq \frac{\|I_N \otimes \bar{K}\| \|\mathbf{d}\|}{\beta}$$
$$\|\tilde{\mathbf{q}}\| \leq \beta \|L\boldsymbol{\delta}\|$$

Clearly, the system is ISS with respect to formation speed actuation signal $\mathbf{v_r}$ and the estimation error $\tilde{\boldsymbol{\varepsilon}}$, collectively written as $\mathbf{d}$ since, the condition $\|\tilde{\mathbf{q}}\| \leq \beta\|L\boldsymbol{\delta}\|$ is enforced by the self-triggered implementation. □

V. Simulation Results

Having established the main results, the proposed control algorithms are validated through simulations in this section. Simulations are performed in Matlab using the VirtualArena[2] toolbox [22].

[2]A Matlab based simulation environment developed by Andrea Alessandretti for simulation of decentralized controllers for multi-agent systems. Visit https://github.com/andreaalessandretti/VirtualArena

A. Path Following

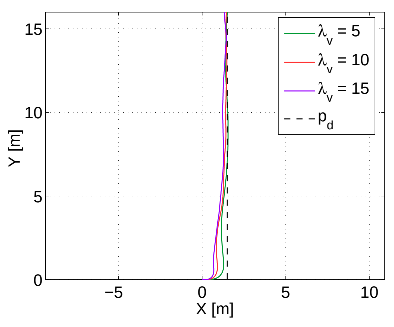
The simulation results of the path following controller developed in section II are presented and the performance of the controller is illustrated taking into account the effects of the inner-loop autopilots, and measurement noise that is reflected by having nonzero state estimation errors.

The path following controller (6) assumes that the control commands, $\mathbf{u}(t) = [v_f, \phi_r]^T$ are tracked perfectly. Such an assumption does not hold in practice and therefore we consider that the autopilots characteristics in closed- loop can be locally described (for simplicity) by a first order dynamics. The model used for simulating the UAV is given by

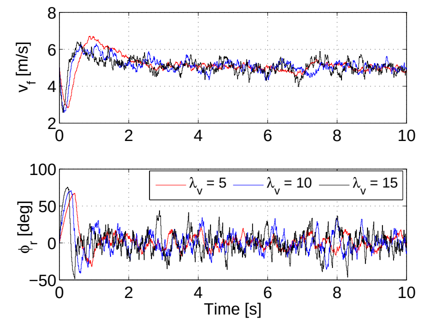
$$\begin{aligned}
\dot{p}_x &= v\cos\psi \\
\dot{p}_y &= v\sin\psi \\
\dot{\psi} &= \frac{g\tan\phi}{v} \\
\dot{v} &= -\lambda_v v + \lambda_v v_f \\
\dot{\phi} &= -\lambda_\phi \phi + \lambda_\phi \phi_r
\end{aligned} \tag{28}$$

where λ_v and λ_ϕ are positive constants. The control inputs v_f and ϕ_r are the command inputs to the internal dynamics. Note that the roll angle command ϕ_r is generated from intermediate control command ω through $\arctan(v_f\omega, g)$. In the above model it is assumed that the UAV autopilot keeps the sideslip angle and the flight path angle near zero over the entire maneuver of the aircraft. Also we assume planar flight at a constant altitude. Additionally, zero mean, white Gaussian noise is injected into the error states $\mathbf{e}(t)$ in order to simulate the estimated states $\hat{\mathbf{e}}(t)$ which is assumed to be obtained from the navigation system on-board the UAV.

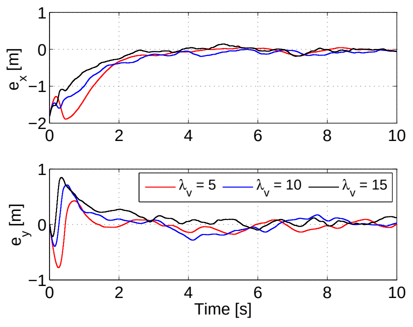
We select three scenarios with $\lambda_v = \lambda_\phi = \{5, 10, 15\}$, to denote the different time constants of the first order internal dynamics of the UAV and analyze the performance of the path following controller in presence of the estimation errors. The simulations were conducted for a period of 10 seconds with discretization of 0.01 seconds with the following parameters: $K_p = I$, $\boldsymbol{\epsilon} = [0.3 \quad 0]^T$, and desired speed $v_d = 5$ [m/s]. From Fig. 2a it can be seen that the path following controller performs adequately and converges to the desired path in the presence of internal dynamics and estimation errors. However, the effects of the inner-loop autopilot dynamics (not considered in control design) can be seen in terms of oscillations during the transients for slower internal dynamics. Also the error states are as expected more oscillatory, especially in the lateral direction of the UAV for the slower internal dynamics case as can be seen from Fig 2c. The error states $\mathbf{e}(t)$ converge to zero and are bounded around zero in the presence of the bounded estimation errors $\tilde{\mathbf{e}}(t)$. This implies that $\|\mathbf{p} - \mathbf{p}_d\| \to \|\boldsymbol{\epsilon}\|$. More precisely, the UAV lags behind the virtual vehicle by a distance of ϵ_1 [m]. Clearly the path following controller is able to steer the UAV towards the path in the presence of internal dynamics and estimation errors while achieving the desired speed of 5 [m/s] (see Fig 2b)



(a) X - Y position of the UAV



(b) Applied control inputs



(c) Error states $\mathbf{e}(t)$

Fig. 2: Path following control performance with different internal dynamics $\lambda_v = \lambda_\phi = \{5, 10, 15\}$ and bounded estimation error.

B. Self-triggered CPF

In this subsection, the simulation results of Self-triggered CPF are presented and discussed. The simulation settings of the Path following controller are the same as presented in previous subsections. The internal dynamics of the UAVs were set to $\lambda_v = \lambda_\phi = 15$. The simulation was conducted for a period of 20 seconds with $N = 3$ agents. Each UAV was assigned a straight line path separated by distance of 10, 20 and 30 meters respectively. All the UAVs start at origin (with different altitudes, not considered in this paper) as the initial condition. The corresponding initial values of the path variables were set as $\boldsymbol{\gamma} = [0 \quad 2 \quad 4]^T$. The tuning parameter specific to the triggering condition of the cooperative controller was set to $\beta_i = 1/N$ for $i = \{1, 2, 3\}$ which is eventually used to compute the lower bound to the inter-event time. The underlying communication topology is considered to be fixed with UAV 1 communicating with UAV 2 and UAV 3 through bidirectional links. There is no direct communication between UAV 2 and UAV 3.

Fig 3a shows the plot of UAV positions for the given straight line formation. As it can be seen, the vehicles are able to follow the desired path and also converge to the desired formation. Note that the path following controllers are being simulated on each UAV in the presence of internal dynamics and estimation errors. The cooperation among the virtual vehicles are evident in Fig 3b, where the error between the path variable γ_i for $i = \{1, 2, 3\}$ of each UAV asymptotically converges to zero. The formation reaches the desired speed assignment $v_d(\gamma) = 5$ [m/s] as can be seen from Fig 3c. All three UAVs achieve the desired speed as expected.

The motivation in this research is to reduce the frequency of controller executions and exchange of information among the agents at the cooperative control level. Fig 3d shows the plot of event time instants for the UAVs. In an ideal, periodic, time-triggered implementation of the CPF, a simulation of 20 seconds with sampling period of 0.01 seconds would require $20/0.01 = 2000$ controller executions and instances of information exchanges. In contrast with the periodic implementation the self-triggered cooperative controller simulated in this paper generates 19, 16 and 20 events for UAV 1, 2 and 3 respectively. The results are not surprising, as the cooperative controller does not need to be executed once the consensus is achieved. Another important observation is that the events are not synchronized showing the true decentralized operation of the agents. The results obtained are encouraging, especially from the point of view of embedded control systems. The controllers can be scheduled to be executed independently and hence the computational resource could be allocated for more demanding tasks.

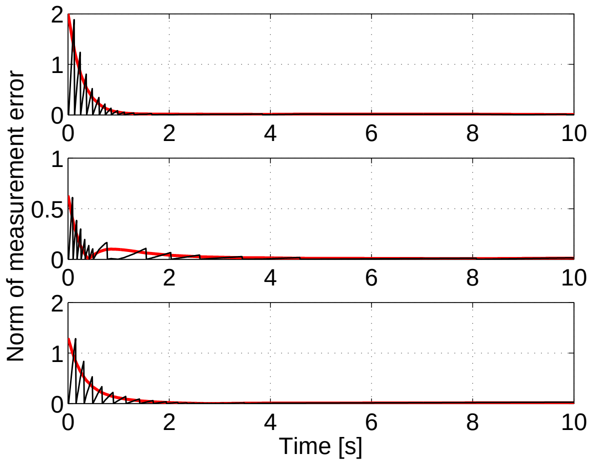
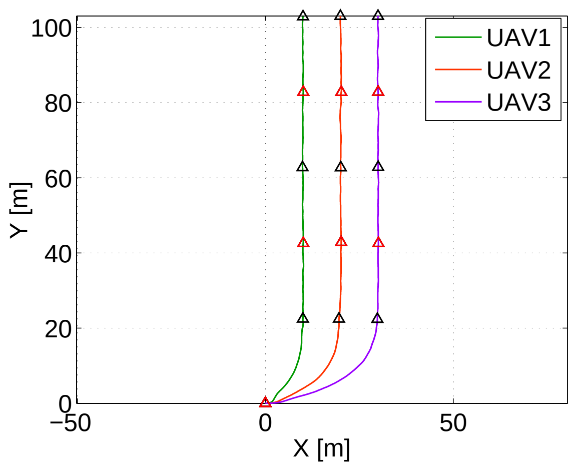
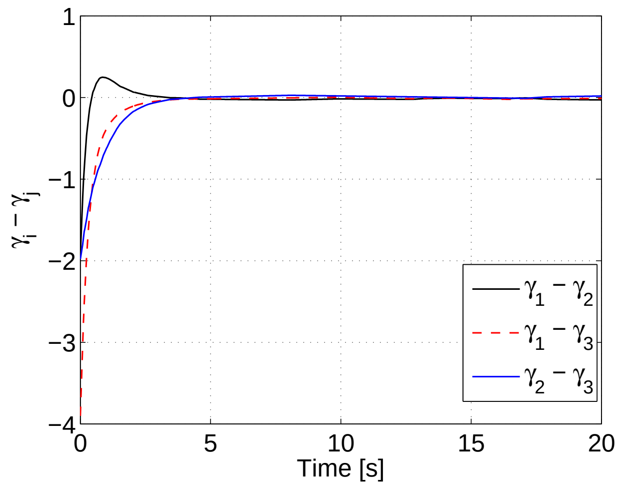


Fig. 4: Plot of triggering condition for $N = 3$ agents

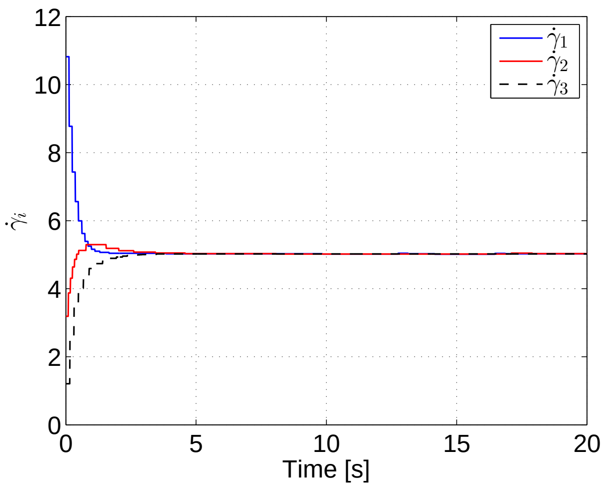
Fig 4 shows the plot of the triggering condition (17) for all the three UAVs. Clearly, the self triggered algorithm is able to replicate the triggering condition arising out of the event-triggered controller presented in Section III. It can be seen that the measurement error $\tilde{q}_i$ accumulates until it hits the bound $\beta_i \|q_i\|$ and resets to zero indicating the occurrence of the event. The measurement errors exceeed the bounds in the plot and this discrepancy can be attributed to the fact that our analysis has been in continuous time and simulation in matlab always happens with certain discretization and hence subjected to numerical errors. Important fact to notice is that the bound and as well as the norm of measurement



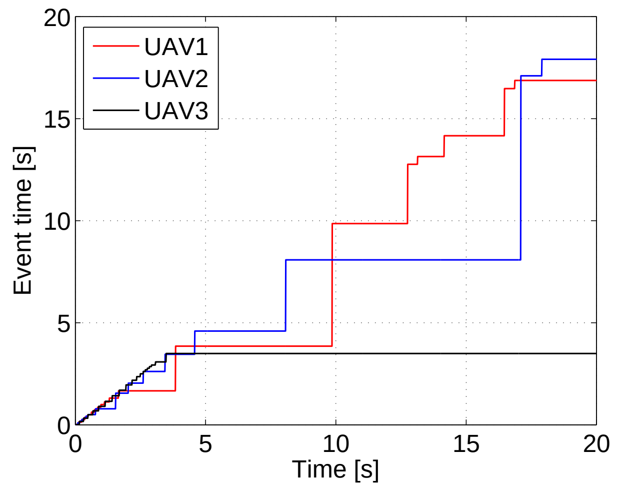
(a) X-Y position plot of UAVs



(b) Error in virtual vehicle positions



(c) Virtual vehicle speed



(d) Event times

Fig. 3: Performance of the Self-triggered Cooperative Path Following Control for $N = 3$ UAVs

errors asymptotically converge to zero indicating the vehicles reaches a consensus. Consequently the number of events during steady state is drastically reduced.

VI. Conclusion

A decentralized, self-triggered CPF controller was developed for fixed wing UAVs and illustrated through computer simulations considering autopilot internal dynamics of UAV and estimation errors. A path following controller and event-triggered cooperative controller was developed and the stability properties were proven using the ISS framework. A self-triggered cooperative control implementation was applied to solve the CPF control problem for fixed wing UAVs. The stability of overall system was proven by using the fact that the path following subsystem and the cooperative control subsystem can be viewed as cascade connection of two ISS systems. Through simulations it was shown that the self-triggered cooperative controller achieve consensus with significantly less number of communication and control updates in comparison with the periodic, time-triggered implementations. The proposed CPF control method uses several simplifying assumptions such as lossless communication links, static communication topology which do not hold in practice. The communication network is plagued by intermittent communication, packet drops, switching topology and delays which need to be taken into consideration during the control design.

The results presented in the current paper, exposes many interesting problems that could be addressed. Investigation of the self-triggered CPF in presence of external disturbances like wind gusts and failure of one of more UAVs would be an interesting extension to the present work. Additionally many practical issues regarding the communication network could be addressed. Experimental validation of the proposed method is perhaps the most important extension to this work.

References

[1] V. V. Klemas, "Coastal and environmental remote sensing from unmanned aerial vehicles: An overview," *Journal of*

Coastal Research, pp. 1260–1267, 2015. [Online]. Available: http://dx.doi.org/10.2112/JCOASTRES-D-15-00005.1

[2] K. Klausen, T. I. Fossen, T. A. Johansen, and A. P. Aguiar, "Cooperative path-following for multirotor uavs with a suspended payload," in *2015 IEEE Conference on Control Applications (CCA)*, Sept 2015, pp. 1354–1360.

[3] A. P. Aguiar and J. P. Hespanha, "Trajectory-tracking and path-following of underactuated autonomous vehicles with parametric modeling uncertainty," *IEEE Transactions on Automatic Control*, vol. 52, no. 8, pp. 1362–1379, Aug 2007.

[4] R. Ghabcheloo, A. P. Aguiar, A. Pascoal, C. Silvestre, I. Kaminer, and J. Hespanha, "Coordinated path-following in the presence of communication losses and time delays," *SIAM Journal on Control and Optimization*, vol. 48, no. 1, pp. 234–265, 2009. [Online]. Available: http://dx.doi.org/10.1137/060678993

[5] K. J. Åström and B. Wittenmark, *Computer-controlled systems: theory and design*. Courier Corporation, 2013.

[6] P. Tabuada, "Event-triggered real-time scheduling of stabilizing control tasks," *IEEE Transactions on Automatic Control*, vol. 52, no. 9, pp. 1680–1685, 2007.

[7] A. Anta and P. Tabuada, "To sample or not to sample: Self-triggered control for nonlinear systems," *IEEE Transactions on Automatic Control*, vol. 55, no. 9, pp. 2030–2042, 2010.

[8] W. Heemels, K. H. Johansson, and P. Tabuada, "An introduction to event-triggered and self-triggered control," in *IEEE 51st Annual Conference on Decision and Control (CDC)*. IEEE, 2012, pp. 3270–3285.

[9] D. V. Dimarogonas, E. Frazzoli, and K. H. Johansson, "Distributed event-triggered control for multi-agent systems," *IEEE Transactions on Automatic Control*, vol. 57, no. 5, pp. 1291–1297, 2012.

[10] D. V. Dimarogonas and K. H. Johansson, "Event-triggered control for multi-agent systems," in *Proceedings of the 48th IEEE Conference on Decision and Control, held jointly with the 28th Chinese Control Conference CDC/CCC 2009*. IEEE, 2009, pp. 7131–7136.

[11] G. S. Seyboth, D. V. Dimarogonas, and K. H. Johansson, "Event-based broadcasting for multi-agent average consensus," *Automatica*, vol. 49, no. 1, pp. 245–252, 2013.

[12] M. Mazo and P. Tabuada, "Decentralized event-triggered control over wireless sensor/actuator networks," *IEEE Transactions on Automatic Control*, vol. 56, no. 10, pp. 2456–2461, 2011.

[13] Y. Fan, L. Liu, G. Feng, and Y. Wang, "Self-triggered consensus for multi-agent systems with zeno-free triggers," *IEEE Transactions on Automatic Control*, vol. 60, no. 10, pp. 2779–2784, 2015.

[14] E. D. Sontag, "Input to state stability: Basic concepts and results," in *Nonlinear and optimal control theory*. Springer, 2008, pp. 163–220.

[15] A. Rucco, A. P. Aguiar, and J. Hauser, "Trajectory optimization for constrained uavs: A virtual target vehicle approach," in *2015 International Conference on Unmanned Aircraft Systems (ICUAS)*, June 2015, pp. 236–245.

[16] A. Alessandretti, A. P. Aguiar, and J. Colin N., "Trajectory-tracking and path-following controllers for constrained underactuated vehicles using model predictive control," in *2013 European Control Conference (ECC) July 17-19, 2013, Zurich, Switzerland*, 2013, pp. 1371–1376.

[17] H. K. Khalil, *Nonlinear systems*. Upper Saddle River, (N.J.): Prentice Hall, 1996. [Online]. Available: http://opac.inria.fr/record=b1091137

[18] M. Mesbahi and M. Egerstedt, *Graph theoretic methods in multiagent networks*. Princeton University Press, 2010.

[19] R. Olfati-Saber, J. A. Fax, and R. M. Murray, "Consensus and cooperation in networked multi-agent systems," *Proceedings of the IEEE*, vol. 95, no. 1, pp. 215–233, Jan 2007.

[20] Y. Fan, G. Feng, Y. Wang, and C. Song, "Distributed event-triggered control of multi-agent systems with combinational measurements," *Automatica*, vol. 49, no. 2, pp. 671–675, 2013.

[21] R. A. Horn and C. R. Johnson, Eds., *Matrix Analysis*. New York, NY, USA: Cambridge University Press, 1986.

[22] A. Alessandretti, A. P. Aguiar, and C. N. Jones, "VirtualArena: An object-oriented matlab toolkit for control system design and simulation," in *2017 International Conference on Unmanned Aircraft Systems (ICUAS)*, June 2017.