Event-triggered Consensus Control Approach for Guaranteed Finite Time Convergence – *Progress Report*

Liam Dale[1], n9741283
Dr. Aaron McFadyen[2]

***Executive Summary* – This document outlines the initial stages of an investigation into event-triggered consensus of multi agent systems guaranteeing finite time convergence. Based on a literature review basic state-dependent and state-independent triggering protocols are simulated. These conserve system resources while having satisfactory performance. They are augmented with open loop estimation techniques to improve effectiveness. A sky-highway use case is defined in the context of urban air mobility. Improvements considering practicality and robustness are planned for future work.**

I. Project Introduction

A century of aerospace innovation is culminating to the development of urban air mobility (UAM), which includes fully automated vertical take-off and landing aircraft (VTOL) for intra-city transportation [1]. When compared to road transport this technology may alleviate congestion and decrease travel time while faring at par or better regarding fuel costs and carbon dioxide emissions [2]. This has piqued interest from many entities including Uber and Airbus, which are already rolling out services on a small scale [3, 4]. Due to its infancy there are still many open questions regarding regulations and technical implementations [5].

Successful UAM realisation will see an increase in vehicle density along major aerial routes and highways which will eclipse the capability of contemporary air traffic management (ATM) systems. Thus, a challenge is posed to provide technologies which autonomously maintain safety and efficiency for dense multi-agent systems (MASs) with networking capabilities [1]. This will require all agents coordinate to observe a collective behaviour.

This project will focus on the topic of consensus, which typically refers to the problem of reaching an agreement among a group despite their initial state trajectories [6]. This is an integral component of group decision making and motion coordination. A typical example is platooning on highways, where vehicles negotiate a collective speed and heading to maintain a safe separation [6]. This project report focusses on decentralised event-triggered consensus as this has practical implications relevant to UAM.

This report details progress, and is structured as follows. Section II provides a background and literature review to further introduce the topic. Findings inform the research problem, which is outlined within section III. A design goal to contextualise the project is proposed in section IV which progress example in section V are working towards. Finally, avenues for future work and a timeline for completion is detailed.

II. Background/Literature Review

Information sharing between MAS participants is essential for consensus such they may all agree on a common goal [6]. It may occur in a centralised manner, which assumes all agents communicate global knowledge over a fully connected network and solve the problem with a central node. This approach introduces a single point of failure and as communication topologies are rarely fully connected becomes increasingly impractical as the number of agents scales [7]. Alternatively there is a decentralised scheme, where information may only be shared between neighbours and nodes perform individual decision making. This method is pragmatic by virtues of robustness and scalability, however is more complex in structure and organisation [7].

Continuous communication between agents is unrealistic for practical applications, as it is an unnecessary drain on limited resources [8]. Alongside corresponding computation and actuator updates it consumes energy to drain batteries and curtail flight time. Networks constraints such as packet loss, latency and throughput worsen with congestion, endangering performance and stability [8]. Drawbacks are exacerbated as the network size scales. As such, it is important to conservatively transmit information while preserving the overall system performance [9]. Event triggered control is a solution where transmissions are spread sporadically based on current system measurements and an event-triggering threshold. This on-demand strategy significantly reduces transmission frequency while upholding performance [9].

While this investigation is tailored towards UAM the applications extend to networked control systems where resource conservation is important. Subterranean environments are critical to modern infrastructure. Due to harsh physical conditions and unforeseeable dangers such as fire or collapse it is preferable to use autonomous vehicles for many related operations [10]. A challenge is the limited communication infrastructure. In [10] autonomous mobile radio nodes traverse a collapsed mine to form a daisy chain and create an ad-hoc communications network. This enables

[1] **Liam Dale** is with Science and Engineering Faculty (SEF), Queensland University of Technology (QUT), Australia. This report is in partial fulfilment of EGH400-1 unit assessment requirements and submitted on 06-September-2020. liam.dale@connect.qut.edu.au.
[2] **Dr. Aaron McFadyen** is also with Science and Engineering Faculty (SEF), Queensland University of Technology (QUT), Australia. He is the supervisor for this research project.

tele-operated robots to clear rubble. An event-triggered solution would crucially extend node lifetime and preserve network resources.

A. Multi-Agent Consensus

The network topology plays an integral role in the consensus of a multi-agent system, determining the rate of convergence, negotiated result and whether consensus is possible [6]. The topology is not necessarily fixed, being affected by vehicle motion and communication dropouts. Suppose there are n agents where the communication topology is modelled by a directed graph $G_n \triangleq (V_n, E_n)$. $V_n = \{1, 2 \dots n\}$ is the set of vertices and $E_n \subseteq V_n \times V_n$ is the set of edges. This system must have negotiations arbitrated by a consensus algorithm.

The most prevalent continuous-time consensus algorithm [6] is given as

$$\dot{x}_i(t) = -\sum_{j=1}^{n} a_{ij}(t)\left[x_i(t) - x_j(t)\right], \qquad i = 1, \dots, n$$

where $a_{ij}(t)$ is an entry in the associated adjacency matrix. Variables i and j respectively denote receiving and transmitting agents. State x denotes the information state, which relates to the consensus variable of interest. The corresponding matrix form is

$$\dot{x}_i(t) = -L_n(t)x(t)$$

where $L_n(t)$ corresponds to the G_n associated Laplacian matrix. Consensus is achieved when for each initial state $x_i(0)$ and for all $i, j = 1, \dots, n$ the difference of states $\left|x_i(t) - x_j(t)\right| \to 0$ as $t \to \infty$.

The continuous time algorithm is discretised with a difference equation [6]

$$x_i[k+1] = \sum_{j=1}^{n} d_{ij}[k]x_j[k], \qquad i = 1, \dots, n$$

where $d_{ij}[k]$ is an entry in the associated row-stochastic matrix D, and k denotes a sampling event. Information states are held constant between triggers. The corresponding matrix form is

$$x[k+1] = D[k]x[k]$$

Similarly, consensus is achieved when for each initial state $x_i[0]$ and for all $i, j = 1, \dots, n$ the difference of states $\left|x_i[k] - x_j[k]\right| \to 0$ as $k \to \infty$.

For both continuous and discrete time with invariant topologies and constant a_{ij} gains consensus is achieved when the directed topology has a directed spanning tree, or the undirected topology is connected. In these scenarios the Laplacian matrix eigenvalues $\lambda_i(L_p)$ are positive semidefinite, starting from zero and increasing. As such, non-zero eigenvalues $\lambda_i(-L_p)$ are all negative. The second smallest $\lambda_2(L_p)$ is the algebraic connectivity and defines the asymptotic convergence rate [6].

These algorithms force x_i to be driven towards the information state of its neighbours. Equilibrium is determined by vehicles which are the root of a directed spanning tree, equal to a weighted average of their initial states [6]. This property gives rise to leaderless, and leader-follower strategies. When there is a single leader it becomes a reference for all followers, which is often the case in platooning. For leaderless networks if a reference is desired an intangible virtual leader may be created to emulate this behaviour [6]. For undirected connected graphs all nodes are leaders and followers giving average consensus [6].

With controllers being connected over a network, the consensus should be robust to wireless channel constraints and stochasticity [7]. Issues such as fading, variable latency, packet misses and quantisation cause the system and controller inputs to be an approximation of measured values. These issues have the potential to degrade performance and stability [7], and alongside network topology changes are not addressed in the general algorithm mentioned.

B. Consensus for Single-Integrator Dynamics

Consider agents where the information state dynamics are given as the control input [6].

$$\dot{\xi}(t) = u_i(t), \qquad i = 1, \dots, n$$

The fundamental consensus algorithms are then altered as follows

$$\mathrm{u_i(t)} = -\sum_{j=1}^{n} a_{ij}(t)\left[\xi_\mathrm{i}(t) - \xi_j(t)\right], \qquad i = 1, \dots, n$$

$$\xi_i[k+1] = \sum_{j=1}^{n} d_{ij}[k]\xi_j[k], \qquad i = 1, \dots, n$$

and in matrix forms

$$\dot{\xi} = -|L_n(t) \otimes I_m|\xi$$
$$\xi[k+1] = -|D_n[k] \otimes I_m|\xi[k]$$

where $\otimes$ denotes the Kronecker product. The concepts for stability and convergence are analogous to before.

This algorithm may be extended to guarantee that information states converge to a set difference Δ_ij, such that $(\xi_\mathrm{i} - \xi_j) \to \Delta_{ij}(t)$ as $t \to \infty$ [6].

$$\mathrm{u_i(t)} = \dot{\delta}_\iota - \sum_{j=1}^{n} a_{ij}\left[(\xi_\mathrm{i} - \xi_j) - (\delta_i - \delta_j)\right], \qquad i = 1, \dots, n$$

Here $\Delta_{ij} \triangleq \delta_i - \delta_j, \forall i \neq j$ where δ describes the relative state reference of the corresponding agent. This algorithm is useful for maintaining relative vehicle positions, which has applications in formation control.

C. Event-Triggered Consensus Protocols

For resource conservation communications and corresponding actuator updates may be spread over discrete time intervals, between which measured values are locked and held. There must be a sufficient frequency to effectively capture state changes and preserve performance goals. Conversely there must be a guaranteed lower bound on inter-event execution times to exclude Zeno behaviour, where infinite events happen on a finite time horizon [11].

The sampled-data approach is where transmissions are delayed by a fixed period, although improving utilisation this still results in an over-provisioning of resources. High frequency triggers required to maintain stability and provide new information in the transient phase are excessive as the system converges [11]. A dynamic approach is required to overcome this limitation.

An event-triggered protocol transmits information on demand to reduce wastage without jeopardising performance. A core component is the error measurement e between the current states x and latest broadcast states $\hat{x}$

$$\hat{x}_i(t) = x_i(t_k^i), \qquad t \in [t_k^i, t_{k+1}^i)$$
$$e(t) = \hat{x}(t) - x(t)$$

where $t_0^i, t_1^i, \ldots$ are the triggering instants for agent i. This provides a measure on how valuable the state of an agent is to maintaining overall closed loop behaviour. Events are triggered when this crosses a dynamic threshold e_T and so at all times the following condition holds.

$$error \leq threshold \triangleq e(t) \leq e_T(t)$$

On triggering agents broadcast their states and alongside receivers update their control inputs accordingly. As controllers use periodically broadcast values the input is redefined as

$$\mathrm{u_i(t)} = -\mathrm{K}\sum_{j=1}^{n} a_{ij}(t)[\hat{x}_i(t) - \hat{x}_j(t)], \qquad i = 1, \ldots, n$$

where K is a gain matrix and again agent i receives from agent j. A basic operating procedure is represented in Figure 1 below. The error threshold is selected to preserve underlying consensus stability concepts such as Lyapunov stability, input-to-state stability (ISS) and L_2 stability. The latter two are preferred in practical systems due to their robustness against external disturbances.

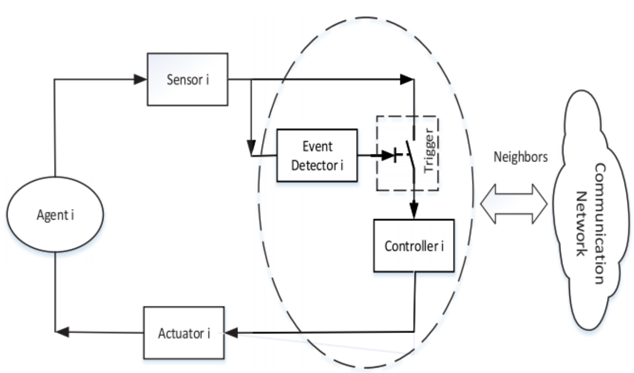

Figure 1 Event Triggering Process [11]

An early event triggered consensus strategy for a linear multi-agent system of N agents is considered in [12] for undirected, connected topologies. The solution is extensible to high order dynamics, described via

$$\dot{x}_i(t) = Ax_i(t) + Bu_i(t), \qquad i = 1,2, \ldots N$$

assuming

$$rank(AB) = rank(A)$$

It is also robust to switching topologies, and enforces a lower boundary on the inter-execution time to exclude Zeno behaviour for all agents. The event triggering-threshold given as

$$e_T(t) = k_i|z_i(t)|, i = 1,2, \ldots N$$

where

$$0 < k_i < 1/\max(\lambda_N[L_p(t)])$$
$$z_i(t) = \textstyle\sum_{j=1}^{N} a_i(x_i(t) - x_j(t))$$

This relaxes the actuation update requirements, however $z_i(t)$ still requires constant knowledge of neighbouring states and global topology – contradicting motivations and leaving continuous communication as a requirement. Communication is relaxed in [13] to use only broadcasted states at the expense of a more complex triggering function. Assumptions of linearity and an undirected topology simplify event triggering, however, raise practical concerns.

An alternative strategy using a state-independent threshold for relative error is proposed in [14] for an undirected, connected topology. The triggering threshold is given as

$$e_T(t) = c_0 + c_1 e^{-\alpha t}$$

with constants

$$c_0 \geq 0, c_1 \geq 0, c_0 + c_1 > 0$$
$$0 < \alpha < \lambda_2(L_p)$$

By only needing local information to compute the state-triggering condition the need for continuous communication is eliminated. Design choices are required for c_0, c_1 and α. If the choice for $c_0 \neq 0$ then consensus is not exact but bounded by a radius but Zeno behaviour is automatically excluded. Details are also given on extensions to accommodate for network stochasticity and double-integrator agents. The assumptions of linear dynamics and network topologies are again impractical.

These strategies will continue to trigger when consensus is reached unless agents are at an equilibrium point. The error thresholds trend towards zero as either the agents converge or time elapses. When they are arbitrarily small if any agent deviates from its latest broadcast value the error is enough to trigger an event [15]. These triggers are not crucial to maintaining consensus, and so are a contradictory wastage of resources. In many autonomous vehicle scenarios this is a hard limitation as agents are constantly moving.

D. *Model-Based Event-Triggered Consensus Protocols*

Model-based event-triggered control protocols employ observer estimation techniques to the predict error and reduce triggering frequency [16]. There is a computational cost as agents must be equipped with estimators for themselves and each of their leaders . If the network is not homogenous agents must also communicate their dynamics. As the network scales these burdens grow [15]. There are open loop and closed loop estimation techniques.

In the open loop estimation approach the latest broadcasts are projected forwards without control inputs. They may be rewritten as

$$\hat{x}_i(t) = e^{A(t-t_k^i)} x_i(t_k^i), \qquad t \in [t_k^i, t_{k+1}^i)$$

where A is a linear state dynamics matrix for agent i. This redefines the control input and error [16]

$$\mathrm{u_i(t)} = -\mathrm{K}\sum_{j=1}^{n} a_{ij}(t)\left[e^{A_i\left(t-t_k^i\right)}x_i\left(t_k^i\right) - e^{A_j\left(t-t_k^j\right)}x_j\left(t_{k'}^j\right)\right]$$

$$e_i(t) = e^{A\left(t-t_k^i\right)}x\left(t_k^i\right) - x(t)$$

where $t_{k'}^j$ denotes a triggering instant for agent j. State dependent and independent protocols utilising this technique are studied in [16] and [17] respectively.

The strategy given in [17] is for discrete time. The estimation is given recursively from the latest broadcast as

$$\hat{x}_i[k] = G^{(k-k_q^i)}x_i[k_q^i], \qquad k \in \left[k_q^i, k_{q+1}^i\right)$$

where G is the discrete state space dynamics matrix and k_q^i denotes triggering iterations of agent i. The event threshold is taken as

$$e_T(k) = c\alpha^{kT}$$

with T as the discrete time step and the following constants.

$$c > 0$$

$$\max(|\lambda(\Xi)|) < \alpha < 1$$

For N agent states and n network agents matrix $\Xi \in \mathbb{C}^{(\mathrm{n}-1)\mathrm{N}\times(\mathrm{n}-1)\mathrm{N}}$ is given as

$$\Xi \triangleq I_{n-1} \otimes G + \Delta \otimes HK$$

where H is the discrete time input matrix and I is an identity matrix. Finally, $\Delta \in \mathbb{C}^{(\mathrm{n}-1)\times(\mathrm{n}-1)}$ is the submatrix of the Jordan canonical form of $I_n - D_{in}$ without row and column one. D_{in} is the topology in-degree matrix (see Appendix I matrix preliminaries). Compared to the previously surveyed state-dependent protocol this threshold requires more calculation. An advantage is that agent dynamics are considered. Discrete sampling inherently provides a lower boundary on inter-event times to exclude Zeno behaviour and so there is no bounding constant needed. Perfect consensus is enforced.

The closed loop approach makes estimates with a control input. It follows that agents must additionally broadcast their inputs on event triggers. This requires drastic alteration of the generic strategy; if $\hat{x}$ reflects the measured states there is no error to trigger. Strategies in [18] and for agents become functions of their edge [19] define edges as virtual systems which consider the differences in states and inputs of their two vertices. Triggers states. Advantages are only seen in the transient phase. When consensus is reached control inputs are zero, and this technique becomes analogous to the open loop estimation. Input broadcasting merits further review and consideration. If agents can estimate the states of their neighbours the continuous monitoring requirements of protocols surveyed in the previous subsection are relaxed.

E. *Finite Time Convergence*

The convergence rate is a significant performance metric for consensus control, so a typical problem is to develop controllers which drive the system in as little time as possible [6]. Most control schemes use Lipschitzian dynamics, leading to exponential convergence with infinite settling time [20] and dependence on initial state conditions [11]. With an event-triggering protocol which aims to reduce control updates, this may suffer further [11]. This motivates practical MAS design to reach consensus in finite time. Research in this area is attracting attention and evolving rapidly, with recent approaches in [21, 22, 23, 24].

An early event-triggered finite time consensus control scheme is proposed in [25] for a mobile sensor network. This designs a finite-time consensus algorithm and determines a state-dependent event-triggering rule to preserve stability. The triggering condition has increased complexity, and still requires continuous communication. Results show that when compared to a standard benchmark performance is maintained while reducing computational resource usage and control updates.

III. Research Problem

The aim of this project is to investigate a practical approach for a MAS to achieve consensus. As a foundational concept in unmanned traffic management (UTM) this is quintessential for the successful realisation of future UAM.

This research will simulate average consensus amongst homogeneous agents under an undirected network. Communication will be event triggered and guarantee convergence within finite time to achieve a balance of feasibility and performance. This will lay the foundation to explore a pressing design challenge which may come under one of the succeeding threads.

a) Robustness to disturbances
b) Robustness to network stochasticity
c) Relaxing network assumptions
d) Robustness to hardware limitations

The latter component is intentionally left open due to the rapid evolution of research in this realm.

IV. Design Program

Working towards a practical implementation requires that a realistic scenario be modelled. This will define the goal for the final outcome.

A. *Vehicle Dynamics*

The idealised rigid-body dynamics of a quadrotor vehicle [26] are a candidate to contextualise this project within UAM. There are three degrees of freedom with cartesian x-y motion and rotation about a single axis. States considered are x position, x momentum, y position, y momentum, pitch angle, and angular inertia. These are as follows.

$$\boldsymbol{z}' \triangleq [x \quad p_x \quad y \quad p_y \quad \theta \quad L]^T$$

They have the following state derivates.

$$\dot{\boldsymbol{z}}' = \begin{bmatrix} \dot{x} \\ \dot{p}_x \\ \dot{y} \\ \dot{p}_y \\ \dot{\theta} \\ \dot{L} \end{bmatrix} = \begin{bmatrix} \dot{z_1} \\ \dot{z_2} \\ \dot{z_3} \\ \dot{z_4} \\ \dot{z_5} \\ \dot{z_6} \end{bmatrix} = \begin{bmatrix} m^{-1}z_2 \\ -(F_F + F_R)\sin(z_5) \\ m^{-1}z_4 \\ (F_F + F_R)\cos(z_5) - mg \\ J^{-1}z_6 \\ l_F F_F - l_R F_R - b_t J^{-1} z_6 \end{bmatrix}$$

Table 1. Quadrotor Variable Descriptions

Variable	**Description**
m	Vehicle mass
g	Gravity constant
J	Moment of inertia around centre mass
b_t	Aero-dynamic damping
F_F, F_R	Thrust force of front rotors, rear rotors
l_F, l_R	Distance from centre mass of front rotor, rear rotor thrust vectors

The model is pictured below.

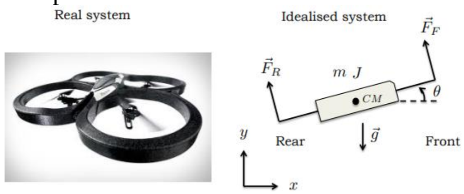

Figure 2 Quadrotor Model

This is a nonlinear model, however the project has only progressed to use linear dynamics. To simplify it for use and to study the consensus problem in a clear way the nonlinear pitch and gravitational disturbance will be removed. Net rotor force will be broken into its x and y components F_x and F_y. This clearly invalidates the goal model is a constraint to be removed in future. This gives the resulting system.

$$\boldsymbol{z} \triangleq [x \quad p_x \quad y \quad p_y]^T$$
$$\boldsymbol{F} \triangleq [F_x \quad \mathrm{F_y}]^T$$

With changed state derivates this is now a linear model. Momentums are single integrator dynamics, and positions are double integrator dynamics.

$$\dot{\boldsymbol{z}} = \begin{bmatrix} \dot{x} \\ \dot{p_x} \\ \dot{y} \\ \dot{p_y} \end{bmatrix} = \begin{bmatrix} \dot{z_1} \\ \dot{z_2} \\ \dot{z_3} \\ \dot{z_4} \end{bmatrix} = \begin{bmatrix} m^{-1} z_2 \\ \mathrm{F_x} \\ m^{-1} z_4 \\ \mathrm{F_y} \end{bmatrix}$$

This may be broken down into the continuous state space matrices.

$$A = \begin{bmatrix} 0 & m^{-1} & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & m^{-1} \\ 0 & 0 & 0 & 0 \end{bmatrix}, \qquad B = \begin{bmatrix} 0 & 0 \\ 1 & 0 \\ 0 & 0 \\ 0 & 1 \end{bmatrix}$$

$$C = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}, \qquad D = \begin{bmatrix} 0 & 0 \\ 0 & 0 \\ 0 & 0 \\ 0 & 0 \end{bmatrix}$$

To give open loop dynamics as follows.

$$\dot{z} = Az + BF$$
$$y = Cz + DF$$

B. *Use Case*

Future UAM may be realised with a free flight style which is not governed by an organisational system [27]. Vehicles may be freely launched and compete for available airspace. They travel the shortest path to their destination and when anticipating collision employ sense and avoid technologies [27]. This will require minimal infrastructure to implement however presents significant safety challenges.

The air highway method of air traffic control provides a safer and more orderly alternative to free flight [27]. It emulates land freeways by geofencing air traffic paths for UAVs to follow. These are splits into directed lanes for travel, passing and damaged vehicles. Paths are carefully planned to preclude agents from crossing. With designated paths the route from source to destination is less efficient than free flight [27]. This idea has not been thoroughly tested or studied, so an optimal design is not known.

The project considers the air highway method of traffic control due to its safety merits. It models a sky highway separated into five lanes without speed limits. Using the cartesian XY plane lanes span the horizontal direction and are stacked vertically. Agents in each lane communicate with their two adjacent neighbours. They aim to reach a consensus on horizontal velocity while specifying a dynamic formation with minimum offsets avoid loss of separation. New agents are periodically spawned around the entry point with random initial conditions and merge into their closest lane. This causes the network topology to switch. Lane boundaries are predefined so agents do not need to use consensus techniques to negotiate the vertical position or velocity.

A preliminary simulation has been performed with the linear quadrotor dynamics discussed. The protocol prescribes to a state-dependent event-triggered strategy. Specific operative examples are given in section V. Updates to actuators are coupled, however with the simplified dynamics they could be separated. Vertical position and velocity are not decided via consensus and so should be triggered independently of network status. The exact triggering mechanisms will be subject to change when dynamics are update and the use case evolves. A demonstrative animation is available for viewing at https://github.com/ldale1/EGH400-Event-Triggered-Consensus/tree/master/Animations.

V. PROGRESS

All simulations are performed in discrete time. For each agent their operations are dictated by the following algorithm. Samples t_k are synchronised for all agents, however this may not be the case in practice.

Algorithm 1. Network Simulation

$t_k \leftarrow 0$
$\hat{x} \leftarrow x_0$
Broadcast $\hat{x}$
While $t_k < t_{sim}$
    Check for transmissions
    If (receiving transmission from any leader j)
        $\hat{x}_j \leftarrow x_j$
        Recalculate control input u
    End if
    Check for an event
    If ($\|e\| > \|e_T\|$)

$\hat{x} \leftarrow x$
Broadcast $\hat{x}$
Recalculate control input u
End if
Decide on control input
If (recalculated control input)
Update actuator
Else
Hold actuator
End if
Ready for the next sample
Step agent
If (model-based protocol)
Project states $\hat{x}$
Project leader states $\hat{x}_j$
End if
$t_k \leftarrow t_{k+1}$
End while

This models a time-invariant topology. When this switches the new topology is modelled as a contiguous scenario. The algorithm restarts with the initial states as the final states of the previous topology.

Consider a strongly connected network with five agents. The dynamics of each are described as

$$\dot{x}_i = \begin{bmatrix} 0 & 1 \\ 0 & 0 \end{bmatrix} x_i + \begin{bmatrix} 0 \\ 1 \end{bmatrix} u_i$$

where x_i and u_i denote states and control input for agent i. The Laplacian communications matrix and associated eigenvalues λ_L are as follows.

$$L = \begin{bmatrix} 4 & -1 & -1 & -1 & -1 \\ -1 & 4 & -1 & -1 & -1 \\ -1 & -1 & 4 & -1 & -1 \\ -1 & -1 & -1 & 4 & -1 \\ -1 & -1 & -1 & -1 & 4 \end{bmatrix}$$
$$\lambda_L = [0, 5, 5, 5, 5]$$

The system is modelled in discrete time, with a sampling period of 0.01 seconds. The corresponding discrete time dynamics are given as follows.

$$x_i[k+1] = \begin{bmatrix} 1 & 0.01 \\ 0 & 1 \end{bmatrix} x_i[k] + \begin{bmatrix} 0.0001 \\ 0.01 \end{bmatrix} u_i[k]$$

The row-stochastic communications matrix and associated eigenvalues λ_D are given below.

$$D = \begin{bmatrix} 0.2 & 0.2 & 0.2 & 0.2 & 0.2 \\ 0.2 & 0.2 & 0.2 & 0.2 & 0.2 \\ 0.2 & 0.2 & 0.2 & 0.2 & 0.2 \\ 0.2 & 0.2 & 0.2 & 0.2 & 0.2 \\ 0.2 & 0.2 & 0.2 & 0.2 & 0.2 \end{bmatrix}$$
$$\lambda_D = [0, 0, 0, 0, 1]$$

Initial state values are given as $x_1(0) = [5.5; 9.5]$, $x_2(0) = [-4.5; -6.5]$, $x_3(0) = [12.5; -0.5]$, $x_4(0) = [9.5; -1.5]$ and $x_5(0) = [-0.5; 2.5]$. Their controller gains are $K = [1.00, 1.73]$ as decided by a linear quadratic regulator which penalises states and inputs equally.

A. State-Dependent Event Trigger

Using the state-dependent event trigger in [12] adapted for discrete time agent trajectories are pictured as follows. Consensus is successfully reached. The settling value for state two is 0.7, which is the average of initial values.

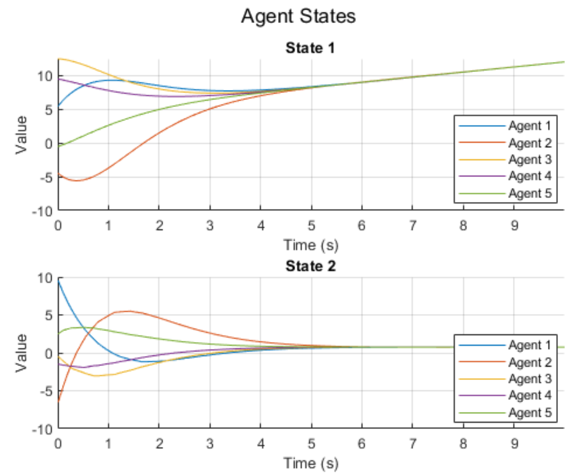

Figure 3 Scenario V.A State Trajectories

During the transient phase triggering instants observe intended behaviour. Agents one and two have the largest initial velocities. Their rapid movement causes frequent early triggers. As they adjust their speed this frequency reduces. When agents near consensus their triggering frequency increases rapidly, contradicting design goals. On average 40.7% of samples are triggered. Actuator updates are more common as they also relate to neighbour transmissions. This ratio will worsen as time draws out due to extreme steady state triggers.

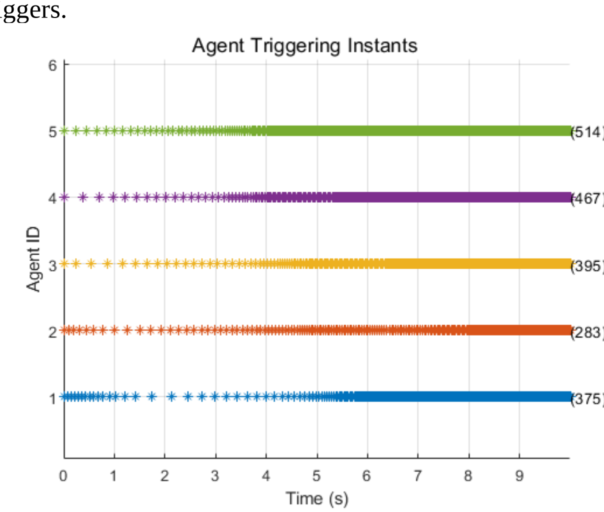

Figure 4 Scenario V.A Triggering Instants

Rapid triggering as consensus is approached is mirrored by an arbitrarily small error threshold. As state two settles on a nonzero value state one will always be drifting from its latest broadcast. Without a predictive factor the error grows and consistently triggers events. Magnitude of state two corresponds to steady state triggering frequency.

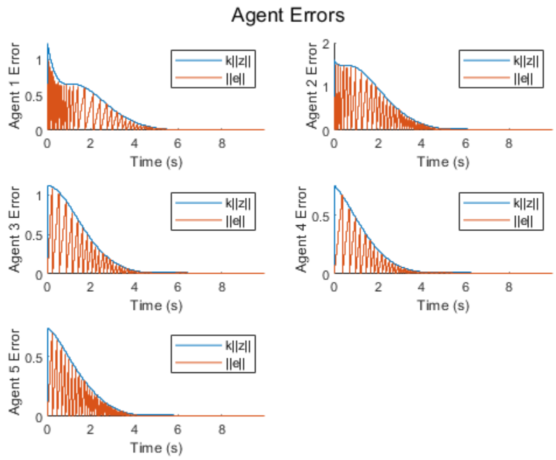

Figure 5 Scenario V.A Error and Threshold Norms

B. Model-Based State Dependent Event Trigger

This scenario uses the same event trigger, however predicts error with an open loop estimation approach. State trajectories and differences to the previous scenario are pictured below. Despite transient differences agents settle at the same states.

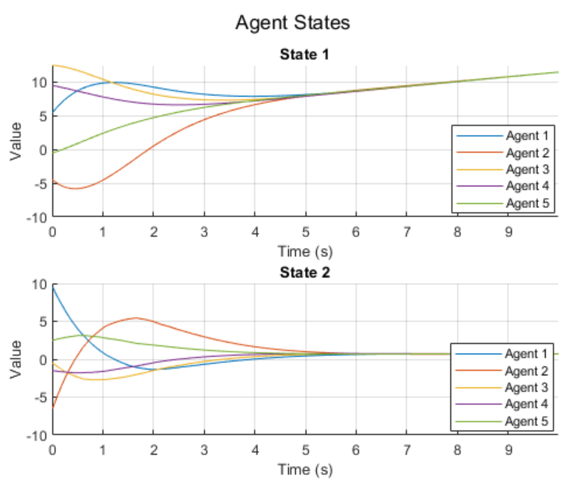

Figure 6 Scenario VII.BV.B State Trajectories

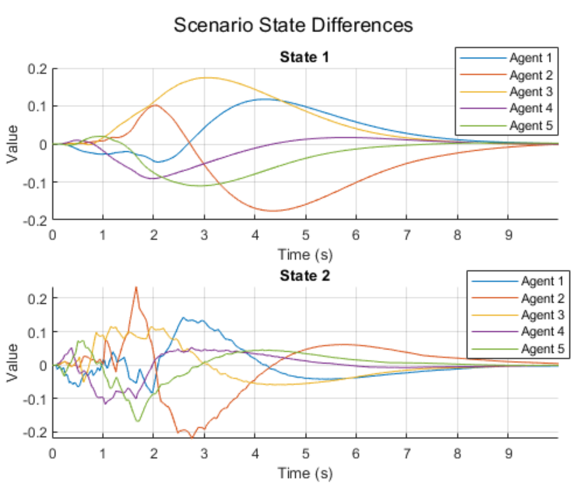

Figure 7 Scenario V.A and V.B State Differences

With the predictive factor agents stops triggering when consensus is reached. Net triggering has been reduced to 4.9% of all samples to better align with the intent. This ratio will improve as time draws out and the network is undisturbed..

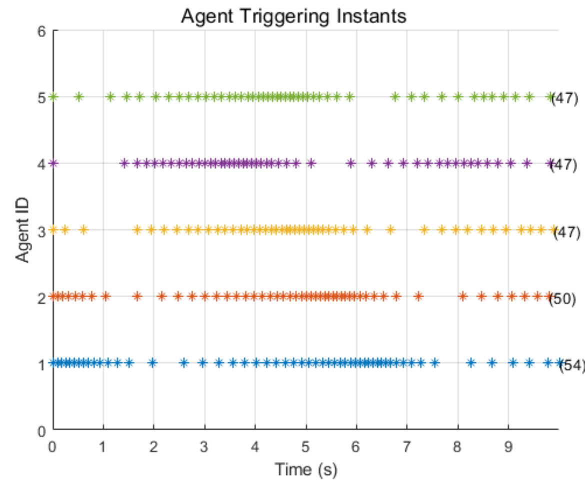

Figure 8 Scenario V.B Triggering Instants

C. Model-Based State Dependent Event Trigger under Switching Topologies

This scenario models a network which switches to a randomly generated, undirected topology every second. No restrictions are placed on their layout. The progression is shown in the succeeding figure.

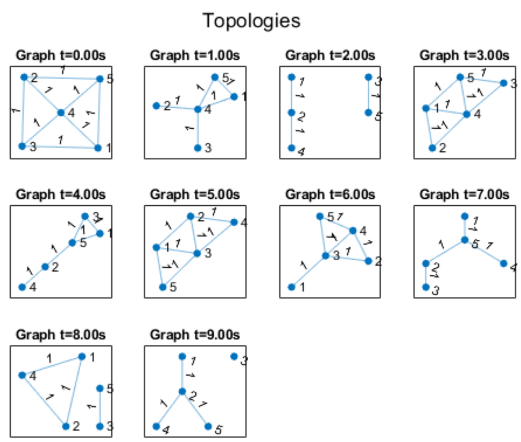

Figure 9 Scenario V.C Topology Progression

Agents and protocol are the same as in scenario V.B. Simulated trajectories are shown in the figure below. These have become jagged, as agent constantly switch leaders. When completely disconnected agents have zero control input and state two stays unchanged. Consensus is still achieved as with enough random changes agents inevitably transmit states across the network. As switching frequency increases the result becomes more analogous to the previous scenario. Settling has been drawn out, and the final state two value has dropped 17.1% to a value of 0.58.

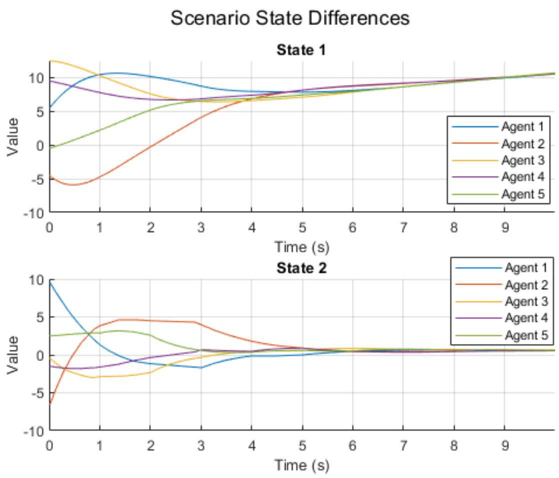

Figure 10 Scenario V.C State Trajectories

Topology changes trigger an event. Whether events keep triggering depends on state derivatives and updated error thresholds. When thresholds quickly drop agents adjust rapidly with frequent triggers.

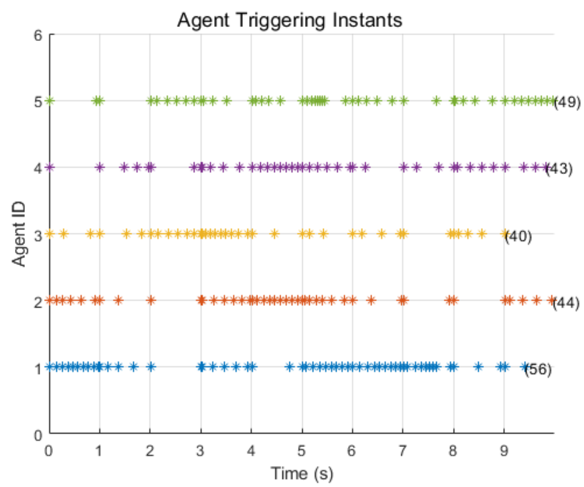

Figure 11 Scenario V.C Triggering Instants

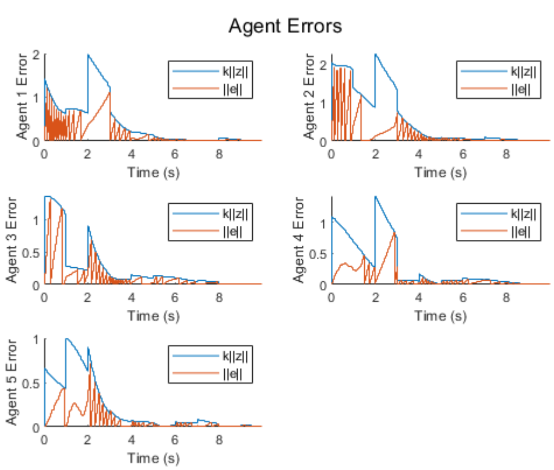

Figure 12 V.C Scenario Error and Threshold Norms

When there is a topology switch agent control inputs can change quite drastically. This is especially relevant when an agent suddenly becomes led by a single neighbour with different states. A mild example is seen in the succeeding figure at two seconds. A practical worry may be actuator switching speed and saturation.

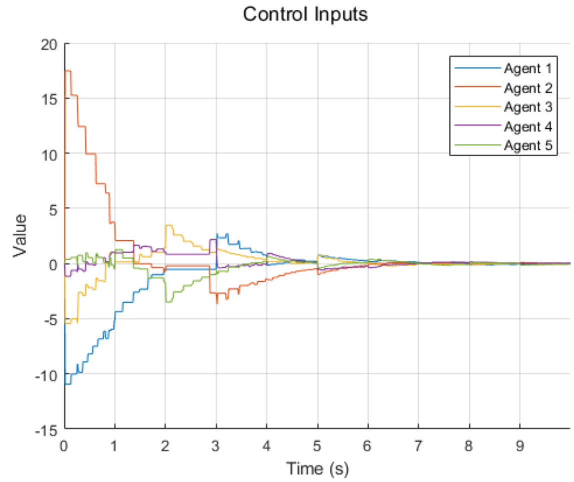

Figure 13 Scenario V.C Control Inputs

D. Model-Based State-Independent Event-Trigger

This scenario implements the model-based state-independent protocol surveyed in II.D with an invariant strongly connected topology. Constant c is chosen as 0.5 arbitrarily. The eigenvalues of matrix Ξ are all $0.9913 \pm 0.05i$ giving constant α as 0.992. State trajectories are similar to previous scenarios.

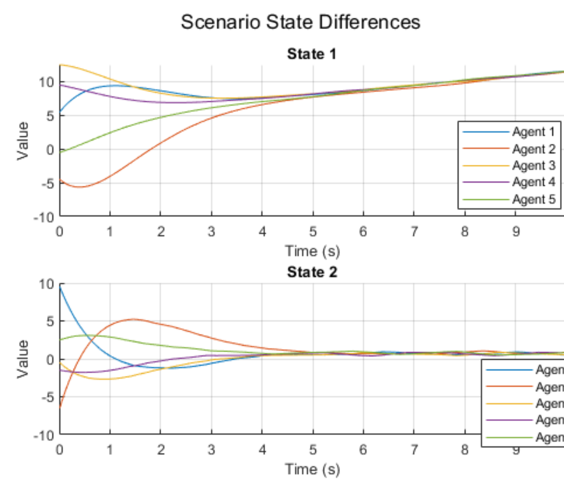

Figure 14 Scenario V.D State Trajectories

Event triggers are very infrequent as shown in the figure overleaf. As a consequence agents are not pushed to a precise consensus speedily; there is still a spread at ten seconds. This is reflected by the error threshold, which decays slowly uninfluenced by neighbours. Empirically the rate for perfect convergence is an inherent challenge for state-independent thresholds. While this may be affected by design choices if the exponential term decays too quickly the threshold goes to zero and appears as a fixed trigger. Given topology switches the threshold does not adapt.

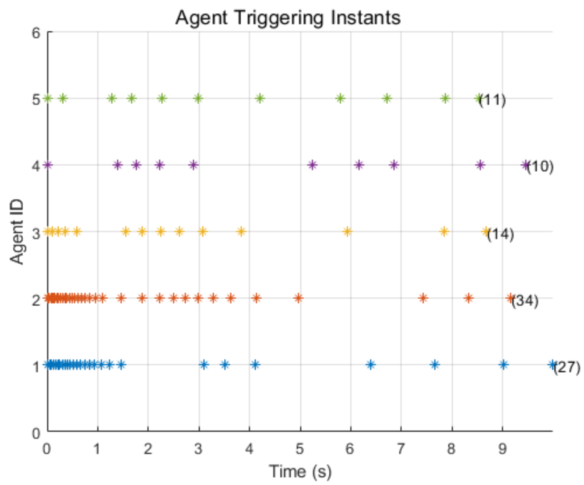

Figure 15 Scenario V.D Triggering Instants

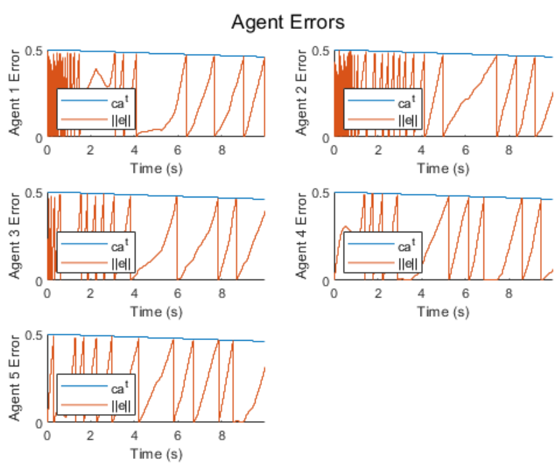

Figure 16 Scenario V.D Error and Threshold Norms

VI. FUTURE WORK

The major deliverables remaining are modelling of nonlinear dynamics and development of a finite time protocol. These work towards the goal UAV model and better performance metrics for the sky highway use case. Secondary deliverables correspond to relaxation of assumptions and simplifications which are impractical. These correspond to research threads discussed in section III. When modelled imperfections have drastic repercussions.

Consider the network and event triggering protocol given in given in section V.B. Using the same initial conditions this is simulated again but with network delay and sensor noise. Ideally the controller should be robust and produce an analogous result. Outcomes reveal that the current approach is quite fragile.

When there is a stochastic transmission delay between 50-100ms introduced the system does not settle to the expected value. State two regulates to zero as if the dynamics include a negative feedback term. There are also excessive triggers, as indicated by the markers.

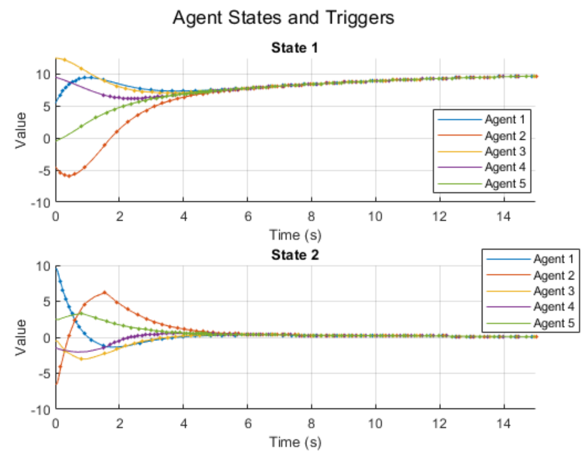

Figure 17 Scenario V.B States and Triggers With Packet Delay

When the measurements have a white noise power of -50dBW events excessively trigger as consensus is approached. The protocol aims to enforce perfect consensus, however the noise consistently repudiates this.

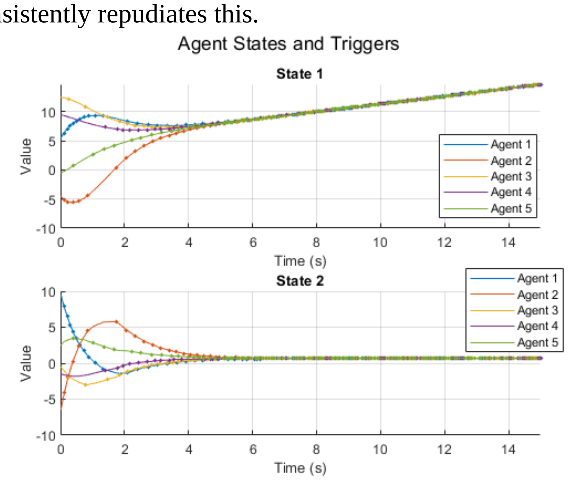

Figure 18 Scenario V.B States and Triggers with Sensor Noise

Layers of complexity will be iteratively added and tested within project phase two to work towards the research goal. This will involve modelling nonlinearity, selecting a contemporary implementation of event triggered finite time consensus and analysing a gap in the work that corresponds to one of the research threads mentioned in section III. This is left general due to the project time scale and rapid rate at which new approaches are published. The literature review will be extended at the beginning of this phase to properly focus the investigation. Work updates will be saved to leave a trail of artefacts which may be used to benchmark performance and gain insights into advantages added.

A high-level process flow displaying tasks and interdependencies for both phases is pictured within Figure 19. This is broken down into a Gantt chart projected timeline, documented within appendix II.F.

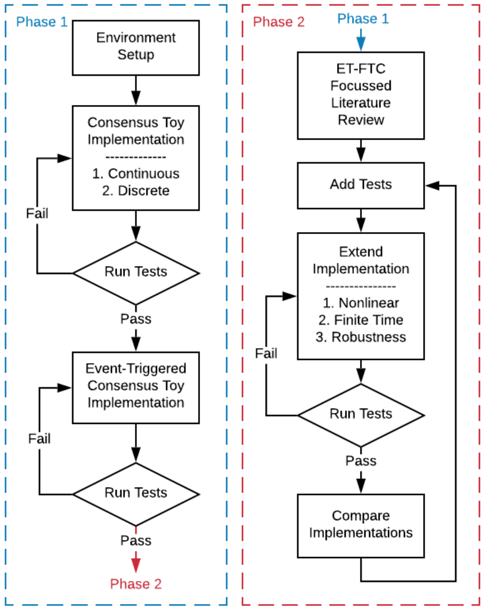

Figure 19 Project Workflow

Due to being an iterative process new learnings and insights will come as progress is made. This may cause the research question to evolve towards a more specific or interesting niche. In parallel there be regular updates to the literature review and math preliminaries to explore new strategies and develop comprehensiveness. With completed milestones new sections will be added to document work and results.

All relevant simulation materials will be stored within the GitHub repository https://github.com/ldale1/EGH400-Thesis-Project

VII. Conclusion

This progress report details an initial exploration into event-triggered consensus. Simulations performed for state-dependent and state-independent triggering protocols show an overall reduction in triggering instants. However, if agents do not settle at equilibrium they trigger wastefully. Using open loop estimation techniques this limitation may be overcome. Future work will drive the project towards the sky highway use case.

APPENDIX I[3]

A. Graph Preliminaries

A team of p vehicles may have information exchanges modelled by graphs (V_p, E_p) which are either directed or undirected. Both cases have a node set $V_p = \{1, \ldots, p\}$ and edge set $E_p \subseteq V_p \times V_p$. In the directed case edge (i, j) denotes that child node j may receive information from parent node i, although not vice versa. In the undirected case edge (i, j) signifies both nodes may receive information from one another. This may be categorised as a special directed graph case, where undirected (i, j) implies a directed couple (i, j) and (j, i). Self-edges are not allowed, unless stated. A weighted graph maps a weight to every edge.

Directed and undirected paths are sequences of edges (i_1, i_2), (i_2, i_3), … in directed and undirected graphs respectively. These are cycles instead if starting and ending at the same node.

A directed tree is where every node has a parent except for a root node, which consequentially has s directed path to its all its descendants. A directed graph is strongly connected if every node is the root of a directed tree reaching all other nodes. An undirected tree is where every pair of nodes is connected by a path. This is analogous to an undirected graph being connected.

A. Matrix Preliminaries

The adjacency matrix $\mathrm{A}_p = [\alpha_{ij}] \in \mathbb{R}^{p \times p}$of a graph reflects the weights of edges (j, i). Value a_{ij} is 0 if $(j, i) \notin E_p$, as self-edges are not allowed this causes the diagonal to be zeros. For the undirected graph the adjacency matrix is symmetrical as $(j, i) \in E_p$ requires $(i, j) \in E_p$ causing $a_{ij} = a_{ji}$. They are also balanced, meaning $\sum_{j=1}^{p} a_{ij} = \sum_{j=1}^{p} a_{ji}$ for all i.

There is a corresponding Laplacian matrix $L_p = [l_{ij}] \in \mathbb{R}^{p \times p}$ which is defined as $L_p \triangleq D_{in} - \mathrm{A}_p$. Here the in-degree matrix $D_{in} = [d_{ij}] \in \mathbb{R}^{p \times p}$ is given as $d_{ij} = 0, i \neq j$ and $d_{ii} = \sum_{j=1}^{p} a_{ij}\,, i = 1, \ldots, p$. This is not the common Laplacian matrix definition, however it has relevance to consensus algorithms.

There is also a corresponding row stochastic matrix $D_p = [d_{ij}] \in \mathbb{R}^{p \times p}$ which is nonnegative and has every row sum to one. It may be calculated as $D_p = (I + D_{in})^{-1} \cdot (I + A)$. The identity matrix $I = [i_{ij}] \in \mathbb{R}^{p \times p}$ is given as $i_{ij} = 0, i \neq j$ and $i_{ii} = 1$. D_{in} is the in-degree matrix as before. The product of these are still row stochastic. It is indecomposable and aperiodic (SIA) if $\lim_{k \to \infty} D^k = \mathbf{1} y^T$ for $y \in \mathbb{R}^n$. They have the eigenvalue 1 for eigenvector $\mathbf{1}_n$. This matrix is used to study consensus in the discrete time setting.

[3] The work in this appendix is adapted exclusively from [6].

APPENDIX II

A. Testing Suite

To verify the integrity of the implementation, a suite of unit tests was developed using the inbuilt MATLAB testing framework.

Automated unit tests will be used to verify the solution correctness and coverage. These will check whether consensus is achieved using a range of total agents, initial state trajectories and network topologies. Convergence can be checked with the literature review definition, however with a sufficiently small error allowable for bounded consensus. With a limited set of numerical examples to base tests off, conclusions about novel test outcomes must be drawn from the principles reviewed.

It is impractical to verify the correct replication of a research paper via automated testing, as there is typically not exact data to compare results with. Important metrics such as the state or error over time are consistently displayed in plots rather than in a numerical form. As such, manual inspection will be required to draw conclusions.

The following table shows a preliminary list of tests. This will be expanded upon throughout the iteration process as complexity increases.

Table 2. Unit Tests

No.	Description	Pass Criteria	Result
A	***Automated Test Suite***		
A.1	*Agents*: 5 *Initial Values:* linear distribution *Network:* Undirected, connected	Consensus Reached	Pass
A.2	*Agents:*5 *Initial Values:* linear distribution *Network:* Undirected, disconnected	Consensus Not Reached	Pass
A.3	*Agents:* 5 *Initial Values:* exponential distribution *Network:* Undirected, connected	Consensus Reached	Pass
A.4	*Agents:* 10 *Initial Values:* linear distribution *Network:* Undirected, connected	Consensus Reached	Pass
B	***Manual Test Suite***		
B.1	State trajectory plot	Mirrored Plot	

B.2	Transmission triggering plot	Mirrored Plot	
B.3	State error plot	Mirrored Plot	

Automated unit tests are summarised with the following results. The two succeeding figures show the network topology and state trajectories for single integrator agents.

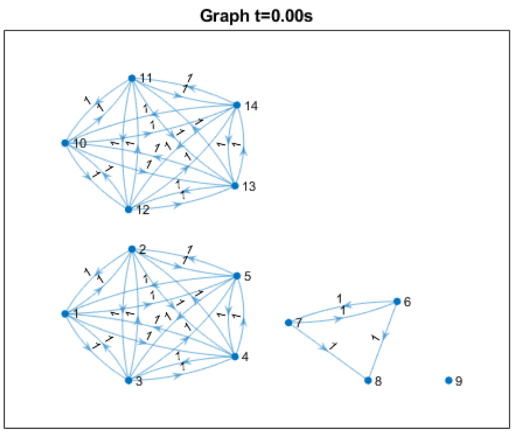

Figure 20 Test Representation Graph

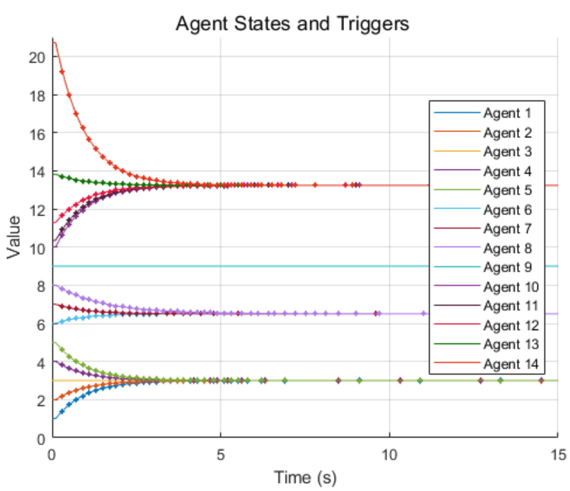

Figure 21 Test Representation States

An example manual unit test is given with succeeding figures. These show an attempted replication of first problem in [17], where related results are displayed in Fig 2. and Fig 4. A visual comparison reveals that results have similar shapes, however there is not an exact comparison. It is difficult to discern where differences stem from. The paper reports the error threshold exponent as the sampling iteration k. It is assumed this should be given as time instead to match displayed values and have consistency across different sampling periods. The network graph in Fig 1. does not match the given communications matrix. Agent three is pictured receiving from four, yet the matrix has this weighted as zero. Separate from the scenario and implementation there are likely differences in the simulation environment and algorithm. Due to ambiguities and result similarity it is assumed the replication is working as intended. It is impossible to confirm correctness.

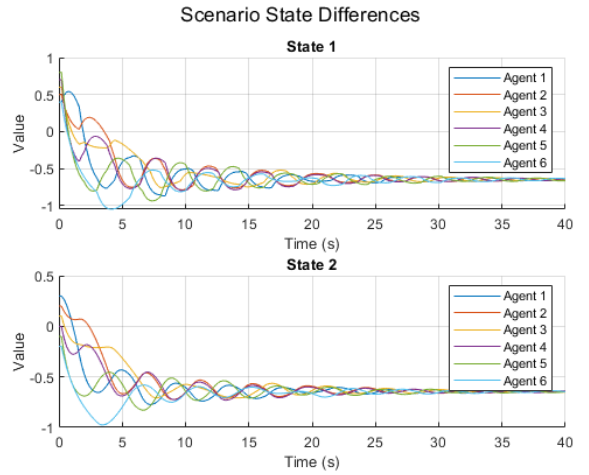

Figure 22 Toy State Trajectories

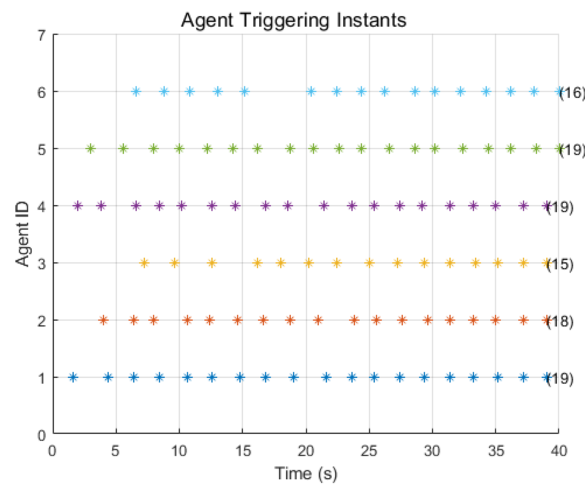

Figure 23 Toy Triggering Instants

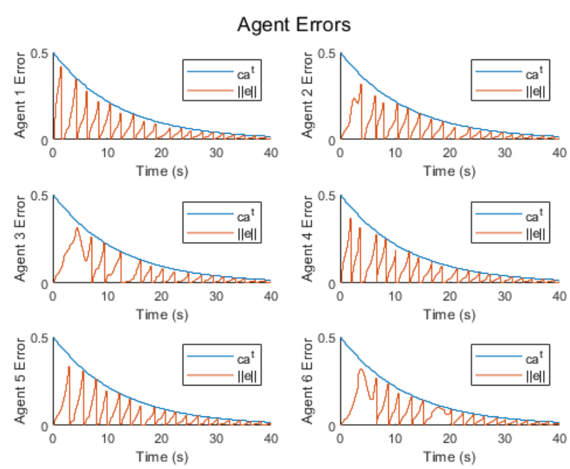

Figure 24 Toy Error Norm and Threshold

Appendix III

A. Progress Summary

The project has progressed to the development of simple event-triggered consensus control strategies. Quarter one was used to build a theory-based foundation. Quarter two was used to develop an extensible simulation environment for the initial attempts and future work.

B. Quarter 1 (Semester 2, 2020 - Project Proposal)

This quarter the initial engineering literature review worksheet and project proposal were completed. A clear plan was established for the remainder of the investigation.

C. Quarter 2 (Semester 2, 2020 - Progress Report)

A strong foundation was established in this quarter to base more complex work to come. Core deliverables of continuous and discrete time consensus were complete. Initial event-triggered and model-based event-triggered protocols were built on this. An advance on phase two started where switching topologies, sensor noise and transmission delays were modelled. The empirical learnings not gleamed from the quarter one theory have been added to the literature review.

The MATLAB simulation environment was written with an object-oriented approach to facilitate the iterative nature of this project. Core concepts such as inheritance and abstraction ensure that modifications and extensions are simple.

D. Quarter 3 (Semester 1, 2021 - Progress Report)

N/A

E. Quarter 4 (Semester 1, 2021 – Final Report)

N/A

F. Project Timeline

The updated Gantt timeline is shown as follows. The main change is the additional of subtask 1.4 Modelling Imperfections. Some minor changes have been made to subtask 1.3 Event-Triggered Implementation to better reflect the process.

I am happy with the first phase progress. The initial project proposal planned for phase one to model an event triggered protocol for single integrator agents under a strongly connected time invariant topology. These deliverables were finished and extended. There have been several approaches explored which model double integrator agents under switching topologies. The case for phase two importance has been made by modelling imperfections and observing the detriments. I have a positive outlook for successful completion and strong outcomes. There is still plenty of work to do and the triggering strategy will be replaced. The technical learnings and observations from phase one will be invaluable for this.

A nontechnical takeaway is that the exact replication of research papers is very difficult. This follows the trend given in Appendix II. Numerical examples do not always detail all the specifics and required values. Simulation algorithms are rarely given and very high level; small changes such as whether agents broadcast on start-up have a butterfly effect to completely change the outcome. Subtask 1.3.6 Testing and Refinement took much longer than anticipated for these reasons. To improve on this in future desired preciseness of replication will be relaxed. The timeline for the next phase has been altered to expand the initial literature review and shrink development time so the final strategy is understood deeply and the refinement will be brief. The first pass at implementation can be feasibly done in a brief period due to the code base extensibility.

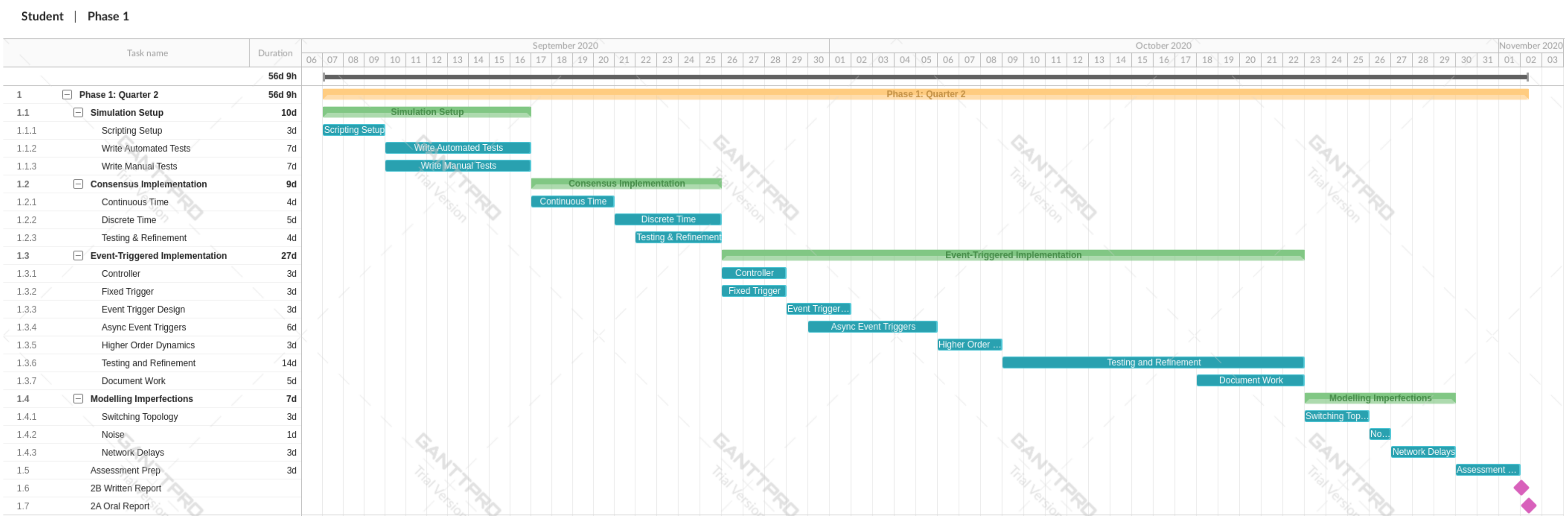

Figure 25 Gantt Chart Remainder Phase 1

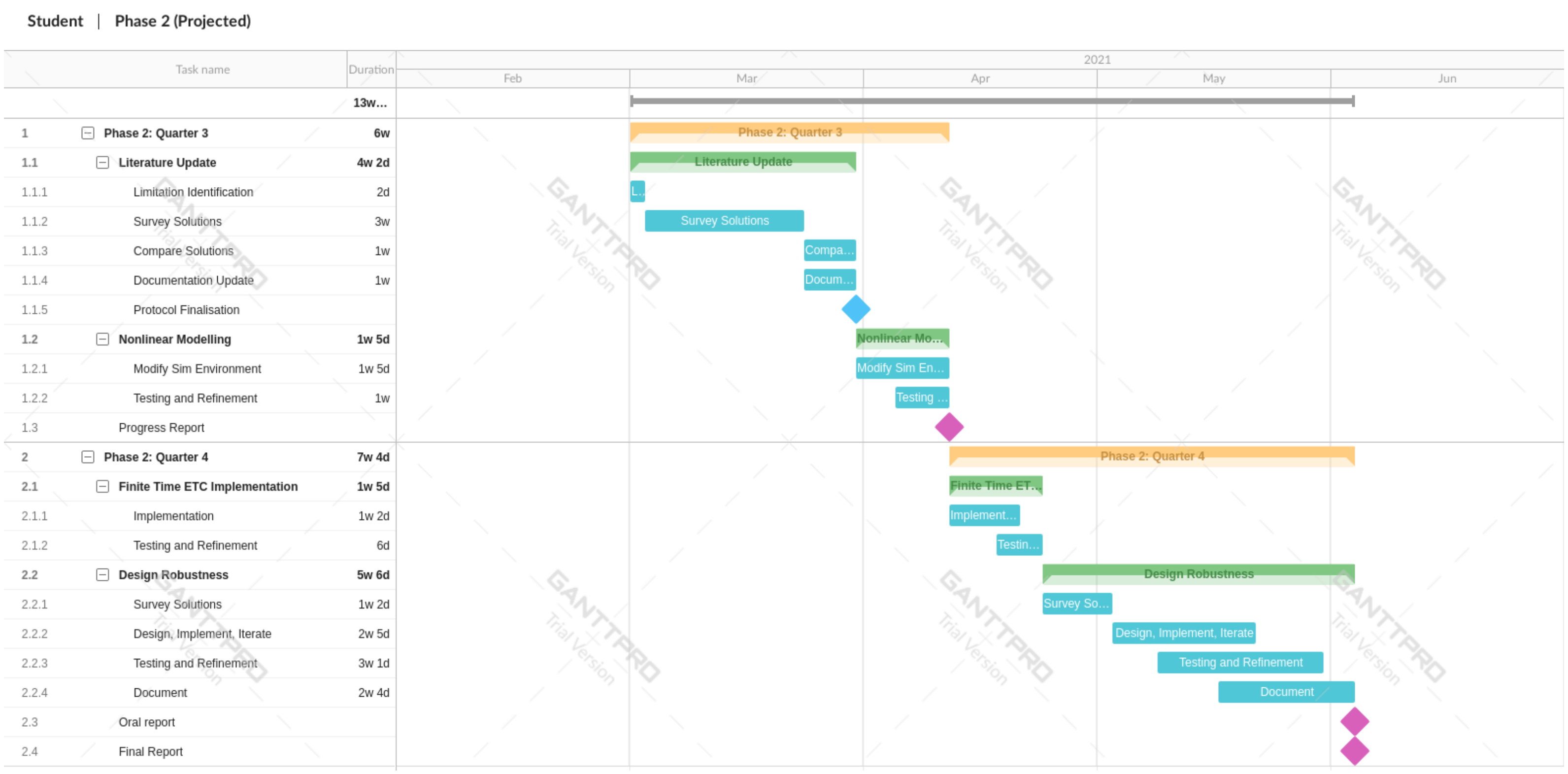

Figure 26 Gantt Chart Projected Phase 2

REFERENCES

[1] A. Straubinger, R. Rothfeld, M. Shamiyeh, K.-D. Buechter, J. Kaiser and K. O. Ploetner, “An overview of current research and developments in urban air mobility – Setting the scene for UAM introduction,” *Journal of Air Transport Management,* vol. 87, 2020.

[2] V. Bulusu, “Urban Air Mobility: Deconstructing the Next Revolution in Urban Transportation - Feasibility, Capacity and Productivity,” ProQuest LLC, Ann Arbor, 2019.

[3] Uber Elevate, "Fast forwarding to the future of on-demand, urban aviation," Uber Technologies Inc, 2020. [Online]. Available: https://www.uber.com/in/en/elevate/uberair/. [Accessed 30 August 2020].

[4] Airbus, “Urban Air Mobility – the sky is yours,” Airbus Group, 27 November 2018. [Online]. Available: https://www.airbus.com/newsroom/stories/urban-air-mobility-the-sky-is-yours.html. [Accessed 30 August 2020].

[5] X. Yang and P. Wei, “Scalable Multi-Agent Computational Guidance with Separation Assurance for Autonomous Urban Air Mobility,” *American Institute of Aeronautics and Astronautics,* vol. 43, no. 8, pp. 1473-1486, 2020.

[6] W. Ren and R. W. Beard, Distributed Consensus in Multi-vehicle Cooperative Control, London: Springer, 2008.

[7] A. Bemporad, M. Heemels and M. Johansson, Networked Control Systems, Berlin: Springer, 2010.

[8] X. Wang and M. D. Lemmon, “Event-Triggering in Distributed Networked Control Systems,” *IEEE Transactions on Automatic Control,* vol. 56, no. 3, pp. 586-601, 2011.

[9] L. Ding, Q.-L. Han, X. Ge and X.-M. Zhang, “An Overview of Recent Advances in Event-Triggered Consensus of Multiagent Systems,” *IEEE Transactions on Cybernetics,* vol. 48, no. 4, pp. 1110-1123, 2018.

[10] K. L. Moore, M. D. Weiss, J. P. Steele, C. Meehan, J. Hulbert, E. Larson and A. Weinstein, “Experiments with autonomous mobile radios for wireless tethering in tunnels,” *Journal of Defense Modeling and Simulation,* vol. I, no. 9, p. 45–58, 2012.

[11] X. Ge, Q.-L. Han, D. Ding, X.-M. Zhang and B. Ning, “A survey on recent advances in distributed sampled-data cooperative,” *Neurocomputing,* vol. 275, pp. 1684-1701, 2018.

[12] Z. Zhang, F. Hao, L. Zhang and L. Wang, “Consensus of linear multi-agent systems via event-triggered control,” *International Journal of Control,* vol. 87, no. 6, pp. 1243-1251, 2014.

[13] S. S. Kia, J. Cortes and S. Martinez, “Distributed event-triggered communication for dynamic average,” *Automatica,* vol. 59, pp. 112-119, 2015.

[14] G. S. Seyboth, D. V. Dimarogonas and K. H. Johansson, “Event-based broadcasting for multi-agent average consensus,” *Automatica,* vol. 49, no. I, pp. 245-252, 2012.

[15] E. Garcia, P. J. Antsaklis and L. A. Montestruque, Model-Based Control of Networked Systems, New York: Springer International Publishing , 2014.

[16] Y. Liu and X. Hou, “Event-triggered consensus control of disturbed multi-agent systems using output feedback,” *ISA Transactions,* vol. 91, pp. 166-173, 2019.

[17] D. Yang, X. Liu and W. Chen, “Periodic event/self-triggered consensus for general continuous-time linear multi-agent systems under general directed graphs,” *IET Control Theory and Applications,* vol. 9, no. 3, p. 428–440, 2015.

[18] Y. W. Meng, L. Xie and R. Lua, “An input-based triggering approach to leader-following problems; Hongye Su; Zheng-Guang Wu,” *Automatica,* vol. 75, p. 221–228, 2017.

[19] D. Liuzza, D. V. Dimarogonas, M. d. Bernardo and K. H. Johanssona, “Distributed model based event-triggered control for synchronization of multi-agent systems,” *Automatica,* no. 73, pp. 1-7, 2016.

[20] V. T. Haimo, “Finite Time Controllers,” *SIAM Journal on Control and Optimization,* vol. 24, no. 4, pp. 760-770, 1986.

[21] J. Lui, Y. Yu, H. He and C. Sun, “Team-Triggered Practical Fixed-Time Consensus of Double-Integrator Agents With Uncertain Disturbance,” *IEEE Transactions on Cybernetics,* vol. Early Access, pp. 1-10, 2020.

[22] X. Lu, “Distributed Event-Triggered Control for Prescribed Finite-Time Consensus of Linear Multi-Agent Systems,” *IEEE Access,* vol. 8, pp. 129146 - 129152, 2020.

[23] C. Du, X. Liu, W. Ren, P. Lu and H. Liu, “Finite-Time Consensus for Linear Multiagent Systems via Event-Triggered Strategy Without Continuous Communication,” *IEEE Transactions on Control of Network Systems,* vol. 7, no. 1, pp. 19-29, 2020.

[24] J. Liu, Y. Zhang, Y. Yu and C. Sun, “Fixed-Time Event-Triggered Consensus for Nonlinear Multiagent Systems Without Continuous Communications,” *IEEE Transactions on Systems, Man, and Cybernetics: Systems,* vol. 49, no. 11, pp. 2221-2229, 2018.

[25] Q. Lu, Q.-L. Han, B. Zhang, D. Liu and S. Liu, “Cooperative Control of Mobile Sensor Networks for Environmental Monitoring: An Event-Triggered Finite-Time Control Scheme,” *IEEE Transactions on Cybernetics,* vol. 47, no. 12, pp. 4134-4147, 2017.

[26] T. Perez, “Modelling of Physical Dynamical Systems,” Queensland University of Technology, Brisbane, 2017.

[27] A. Battista and D. Ni, “A Comparison of Traffic Organization Methods for Small Unmanned Aircraft Systems,” *Transportation Research Record*, vol. 2672, pp. 21-30, 2018.

[28] P. Tabuada, “Event-Triggered Real-Time Scheduling of Stabilizing Control Tasks,” *IEEE Transactions on Automatic Control*, vol. 52, no. 9, pp. 1680-1685, 2007.