Controlling Swarms of Unmanned Aerial Vehicles using Smartphones and Mobile Networks; an evaluation of the Latency requirements

Bruno Olivieri[1], **Marcos Roriz Junior**[1], **Markus Endler**[1]

[1]Department of Informatics – Pontifical Catholic University of Rio de Janeiro
Rio de Janeiro – RJ – Brazil

{bolivieri,mroriz,endler}@inf.puc-rio.br

Abstract. *Swarms of Unmanned Aerial Vehicles (UAV) can be useful to several applications, such as urban surveillance and monitoring of mass events. To build and maintain a swarm formation, the UAVs need to compute, communicate, and coordinate their actions. However, it is complex to design and integrate computing and communicating chips into the UAV control board. Thus, in this paper, we argue that out-of-the-box Smartphones can be coupled to UAV and fulfill this role as a proxy computing unit, i.e., they can be attached to UAVs to do their computing and communicating tasks. Based on this hypothesis, we present a framework to coordinate the UAV swarms using conventional mobile networks. Then, we evaluate if this approach, under different latencies and swarm speed, is fast enough to coordinate the UAV Swarms. The results indicate that the proposed approach present stable swarm formations under* $200\,ms$*, a higher limit than standard 3G and 4G latencies. Further, the results show that the ratio* $\frac{\textit{leader speed}}{\textit{communication latency}}$ *is the key aspect that determines the swarm formation accuracy.*

1. Introduction

Unmanned **A**erial **V**ehicles (**UAV**) are useful in numerous application scenarios, specifically those unreachable or dangerous to humans, such as area patrol, disaster rescue missions, and surveillance monitoring of massive events. They are useful for these scenarios because the aircraft can be remotely operated by a computer through a radio-link, rather than by a human pilot [Austin 2010]. Such UAV's are common referred as flying robots or drones, due to the fact that their computer system are capable of a certain level of self-control, *e.g.*, run arbritary code, move to a given direction, or fly in a given altitude.

In general, one needs several UAVs to perform these tasks, for example, to cover a larger area, or to get multiple views of a given site. A set of UAV that work together to execute a given task is called a *swarm*. Swarm members work together by communicating their position and other useful information in predefined intervals. Spatial formation is a key element for swarm collaboration, *i.e.*, how the UAV are disposed in space (*e.g.*, circular, rectangular, flock, *etc*).

To enable such coordination, UAV swarm members need to communicate with each other. The communication infrastructure used by UAVs can be based on non-existing or existing network infrastructures, such as dedicated backbones or privated repeaters. In the first case, the swarm itself establishes and maintains an ad-hoc communication network to each other through a radio signal, while in the latter case the UAV communicate to

each other through an ubiquitous and uniform wireless communication medium, such as mobile networks (*e.g.*, 3G, 4G). Although non-existing communication network may be mandatory for certain applications, especially those in non urban locations, the existing network infrastructure is a natural fit for urban environments, where one can take advantage of the wide-range coverage of mobile cellular networks, as this works relies.

Nonetheless, it can be complex to design and integrate computing and communicating chips with the UAV control board. Thus, we argue that out-of-the-box Smartphones, *e.g.* Android and iPhone, can fulfill this role as a proxy computing unit, that is, they can be attached to the UAVs to do their computing and communicating tasks. Not only several UAVs are powerful enough to carry an out-of-the-box smartphone, these devices also present a rich set of sensors, such as location, altitude, accelerometer, *etc*, which can be used to enrich/complement the UAV's sensored data. Further, smartphones are constantly becoming lighter (in terms of weight) and cheaper to manufacture.

Using the communication infrastructure, several UAV swarm systems coordinate themselves based on the concept of **P**oint **o**f **I**nterest (POI), where the swarm formation must cover, patrol, or explore the set of POIs [Şahin and Spears 2005]. However, it is difficult to instrument an area with sensor or static routes, especially because POIs can dynamically change depending on the application context. A solution for this problem is to enable the UAV swarm to move towards POIs steered by an (human) operator, and if their sensing capabilities around a POI need to be improved or enlarged, then the swarm should move in a rigid formation.

However, movement coordination in UAV swarms is a challenging task, since it requires reliable communication and timely mutual sharing of the UAV's states, *e.g.*, to ensure a good coverage, avoid collision, *etc*. Further, if a specific swarm formation (shape) should be maintained, then it must be ensured that their current positions and other movement parameters are timely communicated, the robots are always kept within safe distance from each other so that potential collisions are avoided.

Concerned with the requirement of timely swarm coordination and communication, in this paper we investigate **if mobile networks (*e.g.*, 3G, 4G) are fast enough to coordinate Smartphone-based UAV applications**. To do that, by considering premises and limitations of the mobile network infrastructure, we built a coordination framework (master-slave) based on a mobile message-oriented middleware. We simulated an application instance of the proposed framework with several network latencies to validate the model. By examining the coordination accuracy compared to the different network latency values we suggest the range and scenarios where mobile networks can be viable for UAV swarm applications. This can be summarized in the following contributions:

- Answer the question: Is it possible to control a swarm of UAVs using mobile networks and smartphones? If positive, what are the primary constraints?
- Synthetic evaluation of the proposed approach using mobile networks to discover the latency requirements.

The remainder of the paper is structured as follows. Section 2 overviews the fundamental concepts used throughout the paper, while Section 3 presents the core concepts and algorithms of the proposed framework. In Section 4 we evaluate an instance (application) of the model using a serious of network latencies. Section 5 presents the related works. Finally, in Section 6 we present the conclusion and final remarks.

2. Fundamental Concepts

In this section we briefly review the different concepts used throughout the paper, which can be subdivided into those related to UAV systems, those related to the underlying communication infrastructure, and the ones related to the existing coordination models.

2.1. Unmanned Aerial Vehicles (UAV)

A UAV can be simply described as an aircraft which replaces the aircrew by a computer system and a radio-communication link. UAV can have different shapes, which will drastically vary conforming the intended application, as illustrated by Figure 1. For example, zeppelins are used for long and slow movement aerial surveillance, airplanes for long distances, and quadcopters for agile movement hovering over dense urban areas.

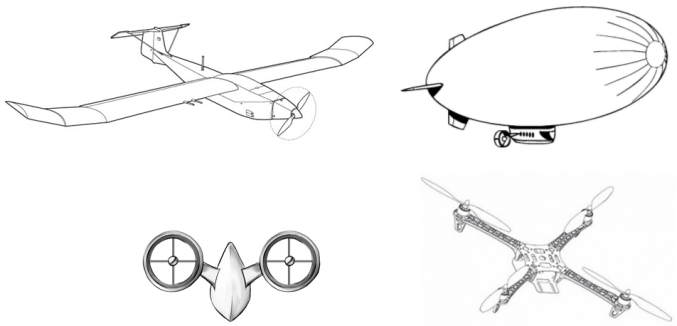

Figure 1. Diverse UAV models: starting with a fixed wing (top left); a Zeppelin (top-right); a bicopter (bottom-left) and a quadcopter (bottom-right).

Based on the existing literature, UAVs can be referred by many different names, such as Drones, Flying Robots or RPAS (remote piloted aircraft systems). In all cases, these definitions consider that the airship has an computing component, a control subsystem, and possibly a radio-link to comunicate with the ground units. Thus, flight is controlled either autonomously by onboard programs and/or through remote control by a pilot located on the ground. Despite this variety of UAV models and a wide range of possible applications, this work approaches UAVs mainly from a computing perspective, instead of its flying characteristic, by focusing on UAVs that are able of executing certain autonomous tasks based on simple computing primitives (operator commands) such as takeoff, landing and traveling to a Point of Interest (POI).

Precisely, in this paper we use a specific type of UAV, a quadcopter – a kind of multirotors (illustrated by Figure 1 bottom-right) – as it present some advantages for executing simple autonomous tasks based on computing commands. First, because it is not a fixed-wing aircraft, quadcopters are capable of easily hovering over a stationary POI. Thus, the device can hold its position replicating helicopter capabilities.

Second, a quadcopter has four propellers with simple motor connection without helicopters rotors. As a result, the aircraft is capable of moving around with four degrees of freedom, flying from a three dimension point $\langle latitude, longitude, height \rangle$ to another one. This feature by itself is very useful to implement simple autonomous tasks (the building blocks of coordination algorithms), without concern of the aerodynamics, since the position and directional vector describing UAV motion are independent of aerodynamic lift pressure (which is an essential factor for airplanes), or even temperature or winds (as is the case for Zepelins).

2.2. Communication Infrastructure

Regarding the communication infrastructure, as mentioned before, we propose to use the Internet access provided by mobile networks to establish communication between several UAVs, in order to enable and control the UAV system. The current mobile network technologies (2G/3G/4G) provide wide-range Internet connection through IP networks for mobile phones. Hence these phones may play the role of universal communication (and processing) hubs for the UAVs. This decision to use the mobile Internet brings some advantages and limitations, as noted below. The primary positive aspects are:

1. Reliance on a established set of wireless communication standards and protocols;
2. Ability to benefit from already installed and tested networks;
3. Wide (and growing) wireless coverage in metropolitan areas;
4. Continuous improvements and optimizations of the used protocols;
5. Potential to utilize a large set of smartphones as radios for communication.

The negative aspects of mobile networks for UAV are:

1. Higher and less predictable network latency compared with the latencies in other ad-hoc networks used for MANs as privated WLANs;
2. Higher package loss rates due to the common handover process between cellular base-stations providing the telecom data services;
3. Due to more frequent handovers caused by the high mobility of UAVs, it is harder to maintain continuous data streams in these networks.

To tackle the second and third negative issues presented, we base our work in existing mobile message-oriented middleware platforms, such as **S**calable **D**ata **D**istribution **L**ayer (SDDL) [David et al. 2013b]. These platforms present lightweight mobile protocols, such as RUDP and MR-UDP [David et al. 2013a], that address intermittent connections, including soft handover. Such protocols can also maintain data-streams to the UAVs through the usage of low-rate keep-alive (heartbeat) messages.

Further, the event-based programming models provided by message-oriented middleware, *e.g.*, *publish/subscribe*, seems a natural fit for the distributed nature of UAV systems. A primary advantage of this programming model is the asynchronous and topic-based messaging primitives, enabling decoupled elements (the UAVs) to easily share state between them by associating a pub/sub topic to the desired event or coordination message type. For example, an UAV leader can easily share (publish) its state with the remainder UAVs in the swarm using an 1-to-N subscription topic, where the swarm is subscribed to the UAV leader topic. By choosing this programming model, we are able to reduce the number of messages, ease the distributed UAV coordination using topics to represent the events, and do "groupcast" messages.

2.2.1. Coordination Model

During the last three decades, multiple coordination models for robots (including UAVs) have been proposed [Murray 2007]. This growing interest in these models was motivated by the premise that certain tasks could be better performed by employing multiple simple robots rather a single (or a few) powerful robot [Pal et al. 2005].

Severe models (e.g. [Prencipe 2005]) follow a distributed coordination approach, where the swarm state is distributed, and the members communicate the state only through message passing. In this work, we opted for using a semi-synchronous model and the coordination life cycle can be summarized as follows:

- **Wai**t: A not synchronized phase, when a UAV waits for information of the leader's new location;
- **Look**: A synchronous phase that starts when a UAV sense its local environment, *e.g.*, its own position and contextual situation (such as battery level or actual direction or status), and is able to use this data to take decisions within the scope of the algorithm that control its behavior.
- **Compute**: Using the sensed and external information, the UAV logically execute the coordination algorithm, and possibly move to a new state. For example, using the sensed location, the UAV identify that it is now in the "outside formation" state and it should retrieve to a given point.
- **Move**: Physically move/execute a given task, such as move to a given latitude and longitude, accelerate, etc.

3. Coordination Framework

In this section we describe the coordination framework for coordinating UAV swarm, which is based on three pillars: an UAV Smartphone-centric hardware architecture, the mobile message-oriented middleware publish/subscribe programming model, and a distributed coordination algorithm. We discuss and explain these elements throughout this section.

3.1. System Premises

In order to fully understand the coordination framework proposed in this work and its limitation, it is necessary to outline the assumptions made, as well as the main hypotheses related to the characteristics of the UAVs, the communication links and the location technology. The premises is that each UAV:

- Contains a location sensor (such as GPS, digital compass, etc);
- Is able to carry a Smartphone with the required sensors;
- Do not fail or deplete their batteries during operation.
- Has a reliable wired connection with the carried Smartphone, and the communication latency between the smartphone and the control board is negligible;
- UAV' control Board provides and reacts reliably to steering controls through an API, such as `MoveToPosition()`, `IncreaseSpeed()`, `DecreaseSpeed()`, `GoToAltitude()`, etc. The interaction with the control board is logically made by the Smartphone and executed analogously by the Flight Control Board;
- Has access to a mobile network through the entire geographic area of interest.

The framework uses a role-based concept (master and slave) and it is leader-oriented, in a sense that the UAVs steers the motion of the entire swarm around the leader in an orderly manner. Once a leader is appointed (master) by the user, the remaining UAVs are queried and some are designated as slaves of this leader UAV and thus gather around it. In other words, it is the user's responsibility to determine the number of slaves and the leader UAV. However, the algorithm is the one responsible for "recruiting" the designated number of slaves.

The swarm is said to be properly formed when the desired number of slaves surround the leader and assume their expected relative positions. This arrangement focuses on positioning the leader in the middle of a circle before arranging the slaves around it, thus increasing the covered area. The user, acting as an overseer, can, just by controlling the leader, direct several UAVs over a POI, thus increasing the visibility in that area.

3.2. Hardware Integration

A central aspect of this paper is the Smartphone-based UAV architecture, where the smartphone acts as proxy computing and communicating unit for the UAV control board. The idea is to decouple the architecture in two core parts, taking advantage of the processing and communicating power of smartphone devices and the well-tested UAV flight control boards. With respect to the airship functionality, the implicit assumption is that it has sufficient energy to perform the tasks, *i.e.*, perform the takeoff and landing, reach the swarm, and do the swarm flight task area autonomously.

An overview of the hardware architecture is illustrated in Figure 2, where the smartphone send and receive commands to the UAV control board. The smartphone receives commands from the coordination framework, such as go forward or goes for a coordinate. This solution is better described on a previous work [Olivieri and Endler 2014]

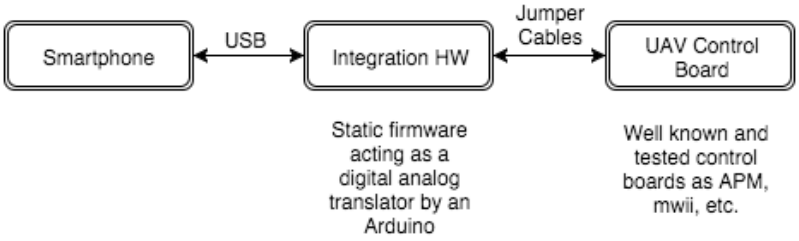

Figure 2. An overview of the UAV Smartphone-based hardware architecture.

3.3. Roles and Distributed Algorithm

To coordinate the swarm, we propose a distributed algorithm, that is, each UAV process the incoming commands itself without requiring a central processing node. Commonly, the elements in a distributed algorithm execute the same code, thus, we assign roles to distinguish them. First, we categorize UAV in two flying modes: Patrol and Swarm. In Patrol mode, the UAVs is autonomously patrolling the region within the borders of interest or executing user commands, while in Swarm mode, the UAV is aiding other UAV in a given task.

The Swarm mode can be further subdivided in two other ones: Leader and Slave. The user interact directly with the Leader UAV by sending control commands, such as move to a position, speed up, slow down, *etc*. Once the command is done, the leader UAV will hover and propagates its own coordinates to its slaves via groupcast (1-to-N message topics). Slave UAVs aid the leader one by responding to its commands, for example, to calculate and reach their expected position in the swarm.

These roles are a central part of the publish/subscribe communication platform. Each role is assigned to a different topic, precisely:

- $Patrol$: to communicate with patrolling (free) UAVs;
- $Swarm$: swarm slaves subscribe to this topic using the leader ID as filter.

Aside from the roles topics, we consider a generic topic (UAV) that every aircraft is subscribed to, but using their ID as filter. This is useful to communicate directly ("unicast") with the given UAV, for example, an user operator sending commands to a given UAV.

Whenever the operator needs that a particular UAV should be escorted by a swarm, *e.g.*, a relevant event occurred, it elect that UAV as a swarm leader. In addition, it also indicates the number n of additional UAVs (from the total of m devices) that will be requested to establish the swarm (where $n < m$). At this moment, all the UAVs subscribed to the $Patrol$ topic receive a message request to send (publish) their current position coordinates to the UAV designated as leader. Algorithm 1 and 2 presents the UAV main loop and the reaction routines for each role topic.

When the leader receives the first slave candidates' messages, it waits for a time interval equal to $2 \times \Delta$ to receive the remaining or late slave candidates messages. Once in possession of this data, the leader selects the n "most appropriate" UAVs that are to become part of its swarm. The criteria used for selecting the slave UAVs vary, depending on the task: it might be the distance to the leader, the airship's residual energy, or any combination of these or other airship data. After the selection is done, the leader UAV publish a command to the elected swarm. When the selected members receives this message, they immediately switch to the slave role, unsubscribe from the $Patrol$ topic, and subscribe it to the leader location updates in the $Swarm$ topic. Consequently, any location update message sent by the leader to this topic is automatically "groupcast" to the slave UAVs subscribed to the topic.

The swarm formation will assume a circular form around the leader, with radius r, in order to widen the visual coverage of the detected occurrence at the ground. All n slave UAVs will be positioned on this circle $\theta = \frac{360}{n}$ degrees apart from each other. To sustain this formation, all slave UAVs receive at the start of the swarm a leader message containing a θ angle and their respective relative positions on this circle (clockwise) around the leader's position (l_x, l_y). To exemplify, the first slave will be positioned at $(r \times \cos(\theta), r \times \sin(\theta))$, while the 2^{nd} slave will be placed at $(r \times \cos(2\theta), r \times \sin(2\theta))$, and the i-th slave will position itself at $(r \times \cos(i\theta), r \times \sin(i\theta))$, as illustrated by Figure 3.

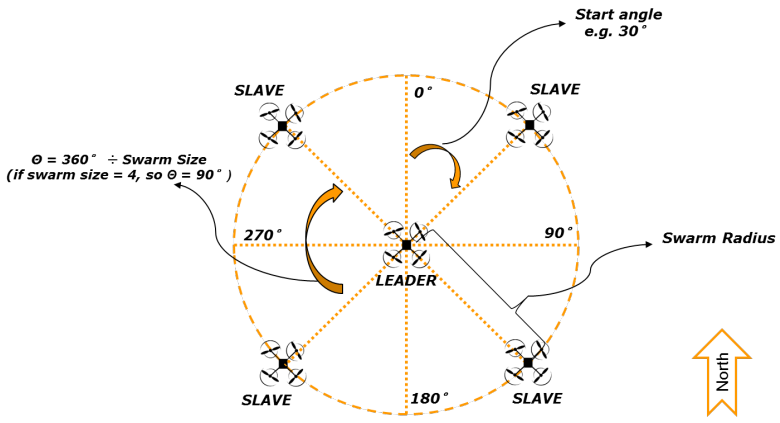

Figure 3. Swarm sample describing its relative positions and main formation.

Algorithm 1: UAV Reactive Loop

on *initialization(*uav_{id}*)*:
    $id \leftarrow uav_{id}$
    $state \leftarrow Patrol$
    subscribe to $Patrol$ and UAV topic

on *receive message* msg *from* UAV *topic*:
    switch msg_{type} **do**
        case *Elected as Leader by Operator*
            BECOMELEADERROUTINE (msg)
        case *Slave Response*
            HANDLESLAVERESPONSE (msg) ▷ *Slave Candidates Response*
        case *Release Command*
            if $status = Leader$ **then**
                publish *Release* Command to each slaves (UAV topic)
            $status \leftarrow Patrol$
            $leader \leftarrow null$
            $angle \leftarrow null$
            $speed \leftarrow speed \times 0.6$
            unsubscribe from $Swarm$ topic
            subscribe to the $Patrol$ topic

on *receive message* msg *from* $Patrol$ *topic*:
    switch $messagetype$ **do**
        case *Slave Request*
            HANDLESLAVEREQUEST (msg)
        case *Slave Confirmation*
            HANDLESLAVERESPONSE (msg)

on *receive message* msg *from a leader of the* $Swarm$ *topic*:
    $p_x \leftarrow r.\cos(angle)$
    $p_y \leftarrow r.\sin(angle)$
    $destination \leftarrow p \oplus msg_{position}$ ▷ $\oplus$ = *coordinates* (x, y) *addition*
    move to $destination$

main loop:
    while *true* **do**
        if $status = Patrol$ **then**
            $hoverAround()$ ▷ *Stays patrolling*
        if $status = Leader$ **then**
            wait $2 \times \Delta$ ▷ *Disseminate leader position*
            publish new position to $Swarm$ topic

Algorithm 2: UAV procedures

procedure BECOMELEADERROUTINE *(msg)*:
    $status \leftarrow Leader$
    $swarmsize \leftarrow msg_{swarmsize}$
    $swarmcandidates \leftarrow \emptyset$
    publish to $Patrol$ topic a $SlaveRequest$ message from id
    wait for $2 \times \Delta$ for slave responses
    $slaves \leftarrow$ **choose** $swarmsize$ candidates from $swarmcandidates$
    for $i \leftarrow 1$ *to* $swarmsize$ **do**
        $cmsg \leftarrow$ create a new $SlaveConfirmation$ message
        $cmsg_{leader} \leftarrow id$
        $cmsg_{angle} \leftarrow (\frac{360}{swarmsize} \times i\text{++}) + 30$
        publish cms to $Patrol$ topic using c id

procedure HANDLESLAVEREQUEST *(msg)*:
    publish to UAV topic of msg_{id} a $SlaveResponse$ message with $\langle id, pos \rangle$

procedure HANDLESLAVERESPONSE *(msg)*:
    $swarmcandidates \leftarrow swarmcandidates \cup \langle msg_{id}, msg_{pos} \rangle$

procedure HANDLESLAVECONFIRMATION *(msg)*:
    unsubscribe from $Patrol$ topic
    subscribe to $Swarm$ topic of leader msg_{id}
    $status \leftarrow Slave$
    $leader \leftarrow msg_{id}$
    $angle \leftarrow msg_{angle}$

All slave UAVs conforms to this configurations, which remain unchanged during the swarm, and use it to move to their relative position (in the circular formation) when the leader moves. The θ angle is measured from the same referential, the geographical north, which is known by each UAV through their compass and is referred to as mutual North. From this moment on, the leader will continuously transmit its absolute position (l_x, l_y) to all its slaves through the *Swarm* topic. This position-update message will be received by all slave UAVs within the $2 \times \Delta$ time limit, and will allow them to constantly update their absolute positions and keep the swarm formation.

The algorithm will continue the logic until the user operator send a release message to the leader and the swarm, which instruct all UAVs in the swarm to return to $Patrol$ mode. Once they change their states, all UAVs will resume their task of ordinary patrolling, which they perform autonomously.

3.4. Algorithm Complexity

The coordination algorithm proposed here is a distributed algorithm with low message complexity, both in the phase of swarm formation and during the swarm mode flight. More specifically, as soon as a new leader is determined with n slaves, it will publish a *Slave Request* to all *Patrol* UAVs ($\leq m$), some of which will reply, yielding a maximum of

$2 \times (m-1)$ messages. Next, the leader sends out unicast messages (*Slave Confirmation*) for the n selected slaves (resulting in n messages), which determine the members of the swarm. A swarm formation is considered acting correctly when its slaves were recruited and are following its leaders as well. To ensure that the chosen UAVs are flying in swarm formation, the leader only needs to send publish its location in the *Swarm* topic every time it changes its position, as each of its slave UAVs will be able to recalculate its own absolute position. Thus, each slave recruitment process and swarm update requires only a small and predictable number of messages, $\mathcal{O}(m)$ and $\mathcal{O}(n)$, respectively, where m is the total number of UAV and n is the number of UAV in a swarm ($n < m$).

4. Evaluation

To evaluate the feasibility of the proposed approach, we evaluated an application instance of the coordination framework with different network latencies and swarm speeds[1]. This evaluation allowed us to identify some issues and constraints of the proposed approach, including an insight of the latency requirements and swarm speed, specifically if fast swarms are able to be controlled in the presence of mobile network latency.

4.1. Evaluation Setup

To evaluate our approach we opted for simulating several UAVs using parallel process communicating with each other. Workarounds were implemented in order to emulate the two main problems that might have impact the scenario: network latency and eventual message loss. These issues were artificially inserted in the simulation environment to test their respective consequences on the accuracy of the swarm formation.

The mobile network latency was emulated by using an internal variable (Δt) in the message-sending functions. Thus, each time a UAV sends a message, rather than being processed immediately, it is buffered into the mobile protocol queue. The elements in this queue are processed with a delay of Δt time period. For example, assuming an average latency of 50 ms for 4G networks, each time a UAV sends a message, it takes 50 ms before it is actually transmitted. So, a message takes $2 \times \Delta t$ from UAV A to UAV B, one Δt from UAV A to the middleware and another Δt to reroute from the middleware to B.

Message loss is the second network behavior that was emulated, and it was approached in a similar way as the introduction of network latency. Another variable (ϕ) hold the probability (in percentage) of message-delivery success. Thus, each time a UAV tries to send a message, the simulator uses the ψ value to determine whether the message will or not be actually transmitted. For instance, for $\psi = 90\,\%$, every time an UAV send a message there is $90\,\%$ chance of the message being transmitted.

We also simulated the UAV "control board", which includes function to alter the UAV movement, altitude, direction, etc. These attributes are continuously updated by a thread that describes the airship's movement physics and processes external flight control commands received through the coordination framework. It also includes high-level routines, such as `GoToPoint()`, `IncreaseSpeed()`, `ChangeAltitude()`.

To verify if the swarm spatial formation is correct we implement a server side element that subscribes the leader and swarm location updates, relying on a Complex Event

[1] An on-line video describing the experiment is available at `https://www.youtube.com/watch?v=3phH-5e7l9c`

Processing engine [EsperTech 2015] framework. This element is useful since it enables us to verify how the swarm formation change with the experiments, for example, when a leader change its position, if and the elapsed the spatial formation becomes invalid.

Finally, all experiments were performed in the same environment, consisting of a single machine running the Ubuntu GNU/Linux 14.04 operating system, and the Java distribution OpenJDK 7. The machine has an Intel i7 quad-core (with Hyperthread Technology, enabling the execution of up to eight processes simultaneously), and 16 GB of RAM memory. The coordination framework was implemented in Java, and used the Scalable Data Distribution Layer (SDDL) mobile message-oriented middleware.

4.2. Swarm Delay

The simulation scenario is composed by a UAV leader that is escorted by four slaves, equally distributed around a radius of 30 meters, that means, the slaves are positioned at the following angles $30°$, $120°$, $210°$, and $300°$. To evaluate how the network latency affects the swarm formation, we emulated eight network latencies and defined four different UAV speeds for the leader, while fixing the slave UAV speed to $120\,\%$ of the leader one.

In this test, in addition to using the coordination framework topics, all UAVs publish the location pair $\langle current, target \rangle$ to a ***Measurement*** topic. Messages sent to this topic are delivered without delay to the measurement server, ***i.e.***, without any artificial transmission delay, while messages in ordinary topics are artificially delayed by Δt period, as explained in subsection 4.1. We considered eight different Δt delays: $0\,ms$, $50\,ms$, $100\,ms$, $150\,ms$, $200\,ms$, $210\,ms$, $225\,ms$, and $250\,ms$. For each latency, we also considered different speeds for the leader UAV, $2\,m/s$, $5\,m/s$, $10\,m/s$, $15\,m/s$.

The measurement of the swarm formation accuracy starts when the UAV swarm is first formed. From this point on, the measurement tool computes the swarm's formation accuracy each time it receives the location pair $\langle current, target \rangle$ of an UAV from the *Measurement* topic. By considering the difference between current and target location of an UAV, and the other UAVs records, we are able to detect if a swarm formation is accurate. To handle minor imprecision, we consider a formation valid if all UAV slaves are within either $90\,\%$ to $110\,\%$ of the swarm's R radius as more discussed on [Olivieri and Endler 2015]. Each test run analyzes $10\,000$ messages from each UAV, in addition, each test were executed four times, resulting in $10\,000 \times 5 \times 4 = 200\,000$ messages being analyzed.

The simulation results are illustrated in Figure 4, which presents the relationship between latency and swarm correctness, for speeds of 2, 5, 10, and 15 meters per second. Emulated latencies were increased by $50\,ms$. The vertical bars show the standard deviation range of each test suite. The graph shows that the best case scenario are those where the UAV is moving slower ($2\,m/s$) with a lower latency ensuring 100% of correctness, while the worst-case scenario happens with a combination of fast speed ($15\,m/s$) and higher latency, cause higher imprecision in the swarm formation with only 40% of correctness. Further, slower swarms yield better formation, since delayed messages will impact them less than faster ones. For example, swarms moving at $2\,m/s$ could present perfect ($100\,\%$) formation, even at the presence of $200\,ms$ latencies. On the other hand, swarms moving at $5\,m/s$, presented correct formation on only $80\,\%$ of the cases. This lead to the conclusion that the $\frac{\text{leader speed}}{\text{communication latency}}$ ratio is the key aspect that determines the swarm formation accuracy.

The experimental results showed that all series exhibited a non-linear and non-proportional shape. Interesting, for all speeds tested, the results shows that one can identify a maximum latency that is supported, after which the swarm formation's accuracy decreases significantly. These limits appear somewhat close to each other on the graph, indicating the presence of other factors potentially contributing to this behavior as discussed below.

In the scope of the present analysis, the results indicate at least another parameter that influences the swarm formation accuracy, the swarms' radius R, *i.e.*, the distance between the leader and the slave UAVs in the swarm. In order to evaluate if and how the swarm's formation accuracy depends on R, we repeated the previous experiment, but using $R = 50$ meters. The graph illustrated by Figure 5 shows the experiment result.

When comparing this graph with the one of Figure 4 it is clear that, when considering a larger radius, the overall swarm formation accuracy increases for higher speeds. On the other hand, increasing the swarm's radius does not result in a better network latency tolerance. Similarly, the significant decrease in the swarm formation's accuracy that was observed at $200\,ms$ did not change with the wider swarm radius. Nonetheless, when the swarm radius increases, all network latencies resulted in higher accuracy levels.

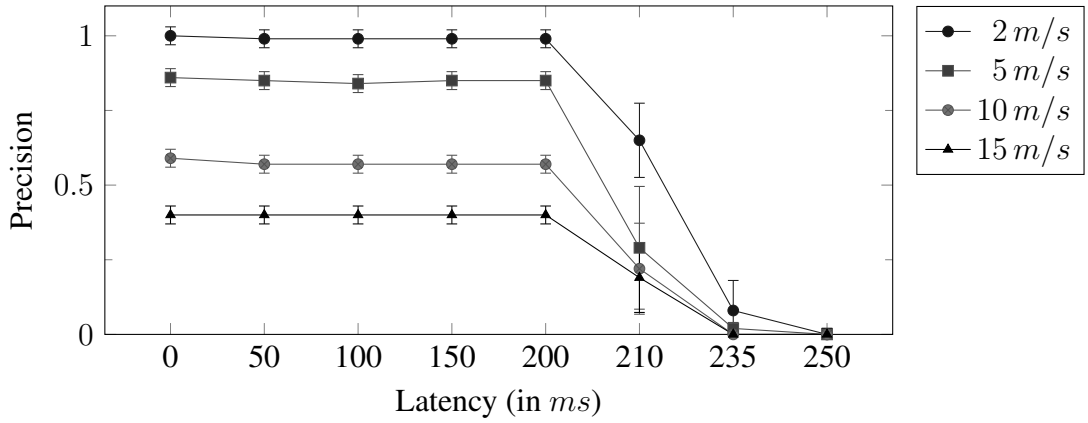

Figure 4. Graph showing results of different leader speeds and network latencies and their combined impact on the accuracy for the swarm formation.

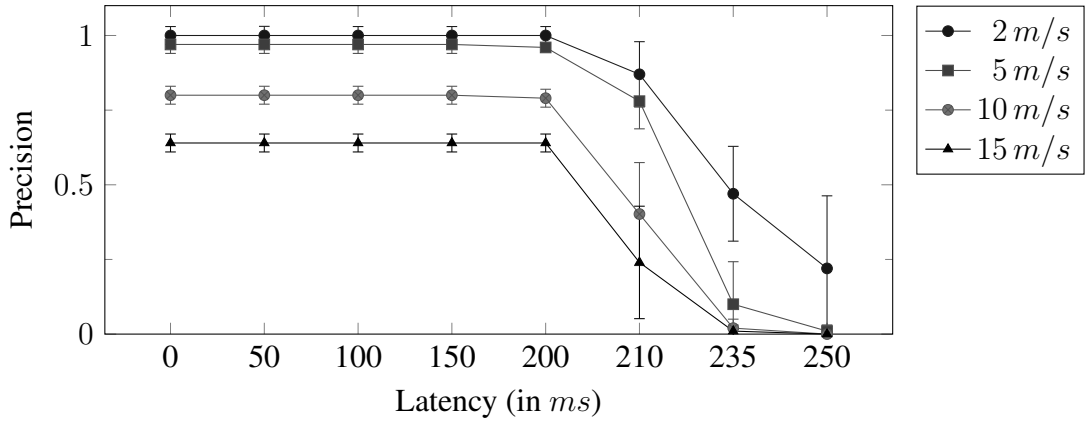

Figure 5. Same experiment of Figure 4, but with swarm radius equal to 50 meters instead 30 meters.

This improvement is due to the higher tolerance for the swarm accuracy. As previously noted, when the swarm radius increases, it make easier for all slaves to reach their target location. Further, the radius tolerance, here $10\,\%$, is also an important variable to be taken into consideration in the experiment. Nevertheless, this evaluation does not intend to explore all possible factors that can affect the swarm formation accuracy. However, we identified the three factors that seem to be more influential on the swarm's behavior, at least in a simulation environment, are: swarm speed, swarm radius, and network latency.

5. Related Work

Flying POIs surveillance using UAV is an active research area and usually been theoretically investigated under TSP and VRP problems optimizations [Pascarella et al. 2013]. Darrah *et al.* in their work [Darrah et al. 2015] introduces a Genetic Algorithm to perform a faster TSP like approach to use UAV on surveillance POIs problem but does not discuss the communication aspect. Barton *et al.* [Barton and Kingston 2013] focus on rapidly creating the swarm formation. They consider a V2I communication infrastructure and a machine learning architecture to process the execution, perform, and tune the trajectory. However, both works does not consider the usage of multiple UAV working together to conquer a common goal, e.g., patrolling a set of POI.

Regarding swarm of UAVs, [Ryan McCune and Madey 2013] presents a review and a newer approach to the Clean Sweep problem [Wagner et al. 2008] by using multiple UAVs to perform target searches on an area. The UAV swarm runs three distinct search algorithms in order to converge on a target search. The work presents evaluations regarding the three algorithms performance and compares them. Unfortunately, it does not explore the communication model or how it impacts on the swarm performance.

In the work [Kumar et al. 2013] presents a very agile[2] swarm of UAVs, working together in several tasks such as dynamic formation, block assembling, music playing and obstacle avoidance. Its communication model relies on 802.4.15 star network with a central processing node responsible to process all movements and all UAV acts as remote controlled aircrafts. Despite our work and other above related, this work runs only in indoor areas under a camera vision system and cannot supposed to work outdoor.

6. Conclusion and Future Works

This work proposed and detailed a framework for coordination swarm of UAV relying on smartphones and mobile networks. This framework is capable to establish swarm of UAVs controlled by one user with a detailed distributed algorithm which was evaluated. This evaluation tested its accuracy against several mobile networks usual latencies and showed that is possible to maintain such swarm of UAV until latencies around $200ms$ which is acceptable for 3G and 4G technologies. Further, based on results reached, we aim to prosecute with prototyping and physical evaluations with a real swarm of UAVs.

References

[Austin 2010] Austin, R. (2010). *Unmanned Aircraft Systems*. John Wiley & Sons, Ltd, Chichester, UK, first edit edition.

[2]Video available on https://www.youtube.com/watch?v=YQIMGV5vtd4

[Barton and Kingston 2013] Barton, K. and Kingston, D. (2013). Systematic surveillance for UAVs: A feedforward iterative learning control approach. *2013 American Control Conference*, pages 5917–5922.

[Darrah et al. 2015] Darrah, M., Wilhelm, J., Munasinghe, T., Duling, K., Yokum, S., Sorton, E., Rojas, J., and Wathen, M. (2015). A Flexible Genetic Algorithm System for Multi-UAV Surveillance: Algorithm and Flight Testing. *Unmanned Systems*, 03(01):49–62.

[David et al. 2013a] David, L., Endler, M., and Roriz, M. (2013a). MR-UDP: Yet another Reliable User Datagram Protocol, now for Mobile Nodes. Technical report, Pontifícia Universidade Católica do Rio de Janeiro, MCC-06/13, ISSN 0103-9741.

[David et al. 2013b] David, L., Vasconcelos, R., Alves, L., André, R., and Endler, M. (2013b). A DDS-based middleware for scalable tracking, communication and collaboration of mobile nodes. *Journal of Internet Services and Applications*, 4(1):16.

[EsperTech 2015] EsperTech (2015). EsperTech - Complex Event Processing.

[Kumar et al. 2013] Kumar, V. U. o. P., Bhattacharya, S., and Ghrist, R. (2013). Multi-robot Coverage and Exploration on Riemannian Manifolds with Boundary. *International Journal of Robotics Research*.

[Murray 2007] Murray, R. M. (2007). Recent Research in Cooperative Control of Multivehicle Systems. *Journal of Dynamic Systems, Measurement, and Control*, 129(5):571.

[Olivieri and Endler 2014] Olivieri, B. and Endler, M. (2014). An ubiquitous based approach for Movement Coordination of Swarms of Unmanned Aerial Vehicles using mobile networks. In *V Simpósio de Computação Ubíqua e Pervasiva, 2014, Brasília. (SBC)*, number JULY.

[Olivieri and Endler 2015] Olivieri, B. and Endler, M. (2015). Coordinating Movement within Swarms of UAVs through Mobile Networks. In *6th IEEE Workshop on Pervasive Collaboration and Social Networking, (PerCol), IEEE PerCom Workshops, At Saint Louis*, number MARCH, pages 154–159.

[Pal et al. 2005] Pal, A., Kshemkalyani, A. D., Kumar, R., and Gupta, A., editors (2005). *Distributed Computing – IWDC 2005*, volume 3741 of *Lecture Notes in Computer Science*. Springer Berlin Heidelberg, Berlin, Heidelberg.

[Pascarella et al. 2013] Pascarella, D., Venticinque, S., and Aversa, R. (2013). Agent-based design for UAV mission planning. *Proceedings - 2013 8th International Conference on P2P, Parallel, Grid, Cloud and Internet Computing, 3PGCIC 2013*, pages 76–83.

[Prencipe 2005] Prencipe, G. (2005). The Effect of Synchronicity on the Behavior of Autonomous Mobile Robots. *Theory of Computing Systems*, 38(5):539–558.

[Ryan McCune and Madey 2013] Ryan McCune, R. and Madey, G. R. (2013). Swarm control of UAVs for cooperative hunting with DDDAS. *Procedia Computer Science*, 18:2537–2544.

[Wagner et al. 2008] Wagner, I. A., Altshuler, Y., Yanovski, V., and Bruckstein, A. M. (2008). Cooperative Cleaners: A Study in Ant Robotics. *The International Journal of Robotics Research*, 27(1):127–151.

[Şahin and Spears 2005] Şahin, E. and Spears, W. M., editors (2005). *Swarm Robotics*, volume 3342 of *LNCS*. Springer Berlin Heidelberg, Berlin, Heidelberg.