A Distributed Antenna Orientation Solution for Optimizing Communications in a Fleet of UAVs

Rémy Grünblatt[a,*], Isabelle Guérin Lassous[b], Olivier Simonin[c]

[a] *Inria Lille-Nord Europe, France*
[b] *Univ Lyon, Université Claude Bernard Lyon 1, ENS de Lyon, Inria, CNRS, LIP, France*
[c] *Univ Lyon, INSA de Lyon, Inria, CITI, France*

Abstract

In this article, we describe how a fleet of unmanned aerial vehicles (UAVs) can optimize communication performances by having its members independently change their orientations. This distributed solution, based on a hill-climbing approach, relies on information available locally at each node, namely the reception power of the received frames. The solution is evaluated using the ns–3 network simulator, whose source code is modified to be able to deal with non-isotropic antennas in the context of Wi-Fi networks, as well as simulate angular movements. As isotropic antennas are only theoretical objects, this step is mandatory in order to increase the realism of network simulations. The results, obtained using realistic antenna models, highlight that controlled mobility, in particular controlled orientation needs to be considered in order for UAV networks to provide better performances.

Keywords: `elsarticle.cls`, LaTeX, Elsevier, template
2010 MSC: 00-01, 99-00

1. Introduction

The use of UAVs is nowadays not restricted to niche applications. Indeed UAVs are now used for military purposes, commercial applications but also for entertainment. The applications are diverse ranging from civil or scientific monitoring to emergency monitoring, including the search and rescue application, the tracking of people and objects, or the delivery of goods. The use of several UAVs cooperating within a fleet of UAVs is considered for different objectives: monitoring a larger area, visualizing a scene with different viewpoints provided by the UAVs of the fleet, sharing different pieces of information to enrich the knowledge on the system monitored by the UAVs or to facilitate the cooperation within the fleet, etc.

UAVs are often equipped with one or several radio interfaces in order to exchange data between them and/or with a ground controller. The radio communications can be realized directly between the devices when possible (when the communication technology offers such direct communications and when the

*Corresponding author
Email address: `remy@grunblatt.org` (Rémy Grünblatt)

devices are within communication range) or via a communication infrastructure. There is a large range of radio communication technologies on the market, each technology having their advantages and drawbacks.

Some UAV fleet applications require high efficiency communications. This is for instance the case for applications based on the transmission of high quality videos, which require, among others, high throughput communications. Some recent wireless technologies like, for instance, Wi-Fi 4 or Wi-Fi 5 (or the upcoming version Wi-Fi 6), or mobile broadband technologies, like 4G or 5G, or proprietary solutions like, for instance, AirMax, offer, at least theoretically, such a high throughput. Practically, the obtained throughput is often less than the maximal one touted by the wireless devices' manufacturers [1, 2, 3]. This cut of throughput can be explained by the protocol's overhead (*e.g.* for control data or medium access) but also by the dynamic rate adaptation that is often implemented in the new wireless communication technologies whose the goal is to find the data rate (also called the physical transmission rate) adapted to the radio environment quality encountered during the transmissions. The radio environment has thus a strong impact on the communications' quality. This environment is not easily controlled and is most of the time undergone by the wireless devices.

In the case of UAVs, often equipped with directional antennas, the antennas' orientation has an impact on the communication quality as shown in [4]. Thanks to some features like motion or orientation, UAVs have the possibility to reach some positions and some orientations that could improve the radio environment quality, and consequently the reached throughput between the UAVs and/or with a ground controller. In this paper, we explore this possibility by leveraging the orientation of directional antennas in order to improve the network performance of a fleet of UAVs. The main contributions of this paper are summarized as follows:

- We propose a controlled mobility algorithm based solely on the orientation of the UAVs, without an *a priori* knowledge of the radiation pattern of the used antennas. The proposed algorithm is distributed and each UAV asynchronously runs its own algorithm based on local measurements on the power of the received packets.

- We have extended the ns–3 network simulator to simulate a network of UAVs equipped with directional antennas of any possible radiation pattern. The extension allows for simulating nodes orientations and *angular* movements, allowing to simulate moving UAVs and the effects of their movements over link quality.

- Based on ns-3 simulations, we have evaluated the proposed antenna orientation solution. Different network topologies with different number of UAVs are considered, as different antennas with different gains and with several rate adaptation algorithms. The solution convergence time and the obtained throughput for each transmitted flow have been studied.

We think that such a solution is of great interest for applications that, at some time periods, need high throughput between UAVs hovering on fixed locations when monitoring an area or an event and exchanging, for instance, videos. The paper is organized as follows: the studied problem is formally modeled in

Section 2. From this modeling, we describe the proposed antenna orientation algorithm in the same section. Then, the simulation environment developed to simulate UAVs and antenna orientation is described in Section 3. In Section 4, we describe the different scenarios that have been evaluated as the obtained results. Papers that relate to the studied problem are discussed in Section 5. We conclude in Section 6.

2. Problem Modeling and Optimization of Antenna Orientation

In this section, we first introduce the studied problem, the assumptions and we then describe the proposed solution for the antenna orientation.

We consider a set of UAVs (also named as agents or nodes hereafter), each equipped with a wireless network interface controller using WiFi and a directional antenna whose the radiation pattern (also named the antenna gain pattern) is unknown. All the agents use the same Wi-Fi channel to communicate. The studied problem is the following: given a UAV fleet spatial configuration can each agent change its local antenna orientation in order to optimize the communication performance, such as throughput? We focus on multi-rotor UAVs because their three-dimensional positions and orientations can be fully controlled and maintained through time by the flight controller, while, for example, fixed-wing UAVs cannot hover at a given position. In this article we consider UAVs whose 3D positions are static, but whose orientations in their horizontal plane, around the normal axis, named yaw, can be changed. Indeed, as the 3D UAVs positions are often application dependent, we focus on parameters that can be modified without interactions with the applications, for the sake of generality. These requirements cover, in particular, the class of area coverage applications, such as surveillance, continuous monitoring or network coverage. As changing the roll (orientation along the longitudinal axis) or the pitch (orientation along the transverse axis) of a multi-rotor changes its 3D position when it is not subject to external forces apart from gravity, we assume those two quantities are also fixed.

In this study, we want to optimize the overall network throughput by changing the agent antenna orientations when the agents are in fixed positions. The throughput obtained by each transmitted flow in the network depends, among others, on the transmission rate used to transmit the flow, the quality of the channel used for the communication, the transmission power and the antenna types and orientations. The transmission rate is very often adaptive and regulated by a rate adaptation algorithm (also noted RAA hereafter). There exist many different rate adaptation algorithms [5] and a large number of Wi-Fi interfaces use proprietary solutions for which the used algorithm is unknown. These algorithms can lead to very different performance for the same scenario, as shown in [6]. We thus think that it is difficult to design a generic antenna orientation solution leveraging the used rate adaptation algorithm. We have therefore decided to use, in our solution, a simpler metric like the power of the received signal. This metric is impacted by the antenna orientation and the channel quality, but it also has an indirect impact on the used rate adaptation algorithm. In Section 4, we will study the performance of our proposed solution with different rate adaptation algorithms.

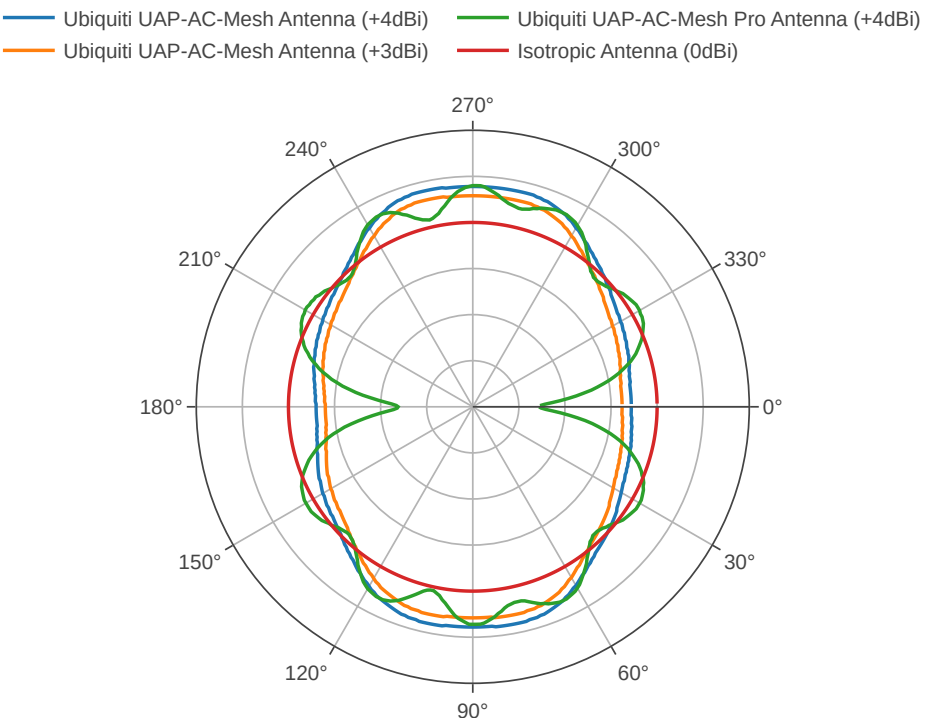


Figure 1: Radiation pattern of the antennas used during the simulations for $\theta = 90°$ (horizontal plane), in dBi (decibel relative to the isotropic antenna).

2.1. Problem Modeling

Let $G = \{V, E\}$ be an undirected graph representing a set of N networked agents, where $V = \{A_1, A_2, \ldots, A_N\}$ and $E \subset V \times V$ denote respectively the set of vertices and the set of edges. We denote by $Ad = (a_{i,j})_{(i,j)\in N\times N}$ the adjacency matrix of the graph: $a_{i,j} = 1$ if $(i, j) \in E$ meaning that agents A_i wants to communicate with agent A_j, and $a_{i,j} = 0$ if $(i, j) \notin E$ meaning that agent A_i does not wish to communicate with agent A_j.

Each agent is equipped with a directional antenna. The antenna radiation pattern is represented by a function g. As g can be different from one agent to another, we use g_i representing the antenna radiation pattern of agent A_i. It is expressed in decibels and in the spherical coordinates system described in [7, Chapter 2.2]. Figure 1 gives an example of four antenna gain patterns in a plane: one for an isotropic antenna and three for non-isotropic antennas. Depending on the antenna orientation between two neighbor agents (*i.e.* there exists a link between these 2 agents in G), these two agents may be able to communicate or not. When they are able to communicate, this orientation has also an impact on the power of the received signal. The higher the received power, the more likely the communication will be of good quality and will use a high transmission rate.

The objective of the agents is then to cooperatively solve the following optimization problem:

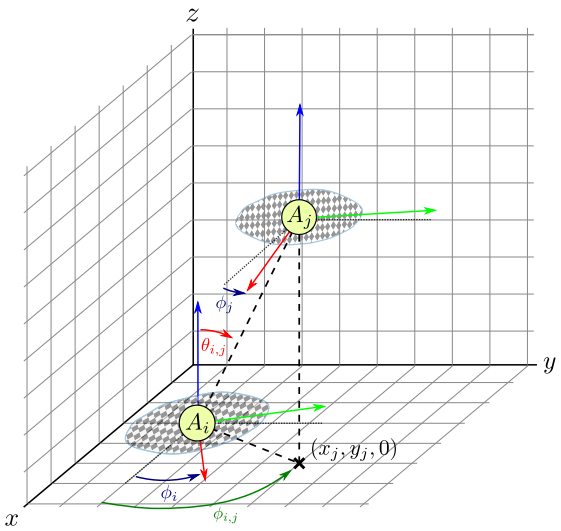


Figure 2: 3D view of two agents A_i and A_j

$$\max_{\boldsymbol{\phi} \in [0;360[^N} f(\boldsymbol{\phi}) := \sum_{i \in \{1,\dots,N\}} \sum_{\substack{j \in \{1,\dots,N\} \\ j \neq i}} a_{i,j} * S_{i,j}$$

with

$$S_{i,j} = e_j + g_j(\pi - \theta_{i,j}, \phi_{i,j} + \pi - \phi_j) + g_i(\theta_{i,j}, \phi_{i,j} - \phi_i) - C_{i,j}$$
$$\text{if } S_{i,j} \geq Th$$
$$= 0 \text{ otherwise}$$

$S_{i,j}$ represents the received power, at agent A_i, of the signal sent by agent A_j and $\boldsymbol{\phi}$ is the yaw orientation vector giving the yaw orientation of each agent (ϕ_i is the yaw orientation of agent A_i). The scalar e_j represents the transmission power of agent A_j in dBm and the scalar $C_{i,j}$ represents the loss induced by the channel between agents A_j and A_i, in dB. The antenna gains used during the communication between agent A_i and agent A_j depend on their position and their relative orientation. Assuming agent A_i is located at (x_i, y_i, z_i) and agent A_j is located at (x_j, y_j, z_j), we have

$$\theta_{i,j} = \text{atan2}(\sqrt{(x_j - x_i)^2 + (y_j - y_i)^2}, z_j - z_i)$$

and

$$\phi_{i,j} = \text{atan2}(y_j - y_i, x_j - x_i)$$

which represent respectively the relative polar and the relative azimuth angles between agents A_i and A_j. These quantities are represented on Figure 2. $S_{i,j}$ is a non null value if $S_{i,j}$ is higher than a given threshold Th representing the minimal signal-to-noise ratio required to receive data. The main parameters used in our model are listed in Table 1.

Finding a solution to this optimization problem involves determining the different agent antenna orientations to optimize the sum of the powers of the

N	Number of agents
A_i	Agent i
V	Set of agents
E	Set of edges
Ad	Adjacency matrix ($Ad = (a_{i,j})_{(i,j)\in N\times N}$)
g_i	Antenna radiation pattern of agent A_i
$S_{i,j}$	Received power at agent A_i of the signal sent by agent A_j
ϕ_i	Yaw orientation of agent A_i
e_i	Transmission power of agent A_i
$C_{i,j}$	Channel loss between agents A_j and A_i
Th	Minimal signal-to-noise ratio to receive data
$M_{i,j}$	Power measurement vector of Agent A_i for its neighbor agent A_j
$m_{i,j,k}$	Mean power, measured at Agent A_i with a yaw orientation k, on signals sent by agent A_j
$Window$	Search space, initially set to $[0; 359]$
$Count$	Maximum number of loop passages when an agent stays in a same orientation
$Goal$	Orientation to reach

Table 1: Main used parameters.

received signals in the network. In the next section, we propose a distributed solution in which each agent determines its antenna orientation based on local measurements, without knowing its antenna gain pattern nor the ones of the other agents, or their positions.

2.2. Optimization based on Antenna Orientation

As the explicit expression of g is unknown from the agents, the proposed solution will be based on measurements that each agent can carry out by rotating on itself. More precisely, agent A_i can measure $S_{i,j}(t)$ at time t if the following conditions are met: agent A_j is transmitting at time t, $a_{i,j} = 1$ such a measurement, it knows its yaw orientation $\phi_i(t)$. These measurements will be stored in a measurement vector M: each agent A_i maintains $M_{i,j} = [m_{i,j,0}, m_{i,j,1}, \ldots, m_{i,j,359}]$ for each agent A_j such that $(i, j) \in E$. The scalar components $m_{i,j,k}$ corresponds to the measurement of the mean received power, at agent A_i, of the signals sent by agent A_j when A_i has a yaw orientation equals to k. Because we are not requiring the knowledge of G and E from the agent A_i, $M_{i,j}$ is created "on the fly" when the connection between A_i and A_j is first established *i.e.* when A_j sends for the first time a packet to A_i. It is then initialized to $M_{i,j} := [\textsc{None}, \ldots, \textsc{None}]$.

Each agent executes its own algorithm without being synchronized with its neighbors. The proposed algorithm consists of an infinite loop (see Algorithm 1). In each passage in the loop (of line 2), each agent realizes different steps. First, the agent fetches the frames it has received since the last loop execution, in its network interface queue, and updates its measurement vectors (line 3 or 4). Then, if the agent lacks some data in its measurement vector with at least one neighbor, it seeks which orientation to move to, to get this measurement (lines 7 and 8). Finally, if its measurement vectors are complete, it tries to optimize its orientation based on their values (lines 16 and 17). Each agent

Algorithm 1: Antenna Orientation Optimization (agent A_i)

$Window \leftarrow 360$; $Goal \leftarrow None$; $Count \leftarrow 0$;
while *true* **do**
  % *Measurement updates*
  if $m_{i,j,\lfloor\phi_i(t)\rfloor} = \text{None}$ **then** $m_{i,j,\lfloor\phi_i(t)\rfloor} \leftarrow S_{i,j}(t)$
  else $m_{i,j,\lfloor\phi_i(t)\rfloor} \leftarrow \text{mean}\left(m_{i,j,\lfloor\phi_i(t)\rfloor}; S_{i,j}(t)\right)$
  % *New orientations to explore*
  if *there exists k such that there exists j such that* $m_{i,j,k} = \text{None}$ **then**
    find the k minimizing $|\lfloor\phi_i(t)\rfloor - k|$
    $Goal \leftarrow k$
    if $k == \phi_i(t)$ **then** $Count \leftarrow Count + 1$
    else $Count \leftarrow 0$
    if $Count \geq 10$ **then**
      $Count \leftarrow 0$
      $m_{i,j,k} \leftarrow -100$ for any j such that $m_{i,j,k} == \text{None}$
  % *Orientation optimization*
  if *there exists no j or k such that* $m_{i,j,k} = \text{None}$ *and (*$Goal$ *is* None *or* $Goal == \lfloor\phi_i(t)\rfloor$*)* **then**
    Find l in $[\lfloor\phi_i(t)\rfloor - Window/2; \lfloor\phi_i(t)\rfloor + Window/2]$ maximizing: $\sum_{j\in[1;N]} a_{i,j} * m_{i,j,l}$
    $Goal \leftarrow l$
    $Window \leftarrow Window/2$
  % *Change of orientation*
  Set d_i to reach $Goal$

runs the algorithm while it changes its orientation according to online results and while it communicates with its neighbors if required by the data traffic.

The proposed algorithm is based on the hill climbing approach [8]. We have chosen hill climbing for two reasons: 1) it is an anytime algorithm (it can find better and better solutions as long as it keeps running) and 2) even if it does not guarantee convergence towards a global optimum, it provides an efficient way to find a good solution in a decentralized multi-agent problem. Algorithm 1 describes the algorithm executed by agent A_i.

As the orientation $\phi_i(t)$ and the power measurement $S_{i,j}(t)$ of the received signal depend on the instant at which these 2 parameters are considered, they are expressed in function of the time t.

The *Goal* variable represents the orientation the agent is currently trying to reach. In the first loop passages, *Goal* corresponds to unexplored orientations for which no measurement value has been collected (line 8). Once measurements have been collected for all the orientations and neighbors (from lines 14 and 15), then an optimal orientation (in respect to the defined objective function) can be computed (line 16). Then, the parameter d_i, representing the direction to follow (*i.e.* right, left, or do not move), is updated in order to reach the orientation *Goal* (line 17). The $Window$ variable represents the search space. Initially, the search space includes all the possible orientations ($[0; 360[$). In order to speed up the algorithm convergence, the size of the space search is divided by 2 as soon as a maximal solution is found in the current space search (line 18 of Algorithm 1). The $Count$ variable represents the maximum number of loop passages during which the agent stays in the same orientation. If the agent stays in a given orientation for too long while trying to fill its measurement vector, the agent considers that it is not a good orientation and sets a very low value to the corresponding measurement element (line 13 of Algorithm 1). The main parameters of Algorithm 1 are listed in Table 1.

Note that finding an optimal orientation does not mean the end of the algorithm. The search continues with new possible measurements and on a reduced search space. As long as the search window is not reduced to a singleton, a new optimal solution can thus be found.

For each agent, the complexity of each execution of the algorithm loop in terms of base operations (additions, multiplications, assignments and look-ups) is linear in the number of the agent's neighbors (*i.e.* in $O(N)$) as the line 5 from Algorithm 1 implies checking the power measurement vectors for all the neighbors and the line 16 requires a summation of at most N elements. In this algorithm, we use a fixed granularity of 1 degree for the measurement vectors, granularity which is hidden in the use of the floor function at lines $3-4$, 7 and $15-16$. As such, we consider the search from line 16 to be executed in constant time. If we were to make this granularity variable, for example by using intervals of d degrees, the complexity would be $O(N/d)$.

2.3. Illustration of the solution

We now show an example of an execution of the algorithm on agent A_i. Fig. 3 shows that agent A_i already received packets from two neighbours (Neighbour 1 and Neighbour 2) and measured the power level of the received packets on two orientations $0°$ and $1°$. The power level given for each neighbour corresponds to a mean power (exponentially weighted moving average) computed from the

packets received from this neighbour. At this stage of the algorithm, we can note that agent A_i has not received any packet from Neighbour 2 in the orientation 2°.

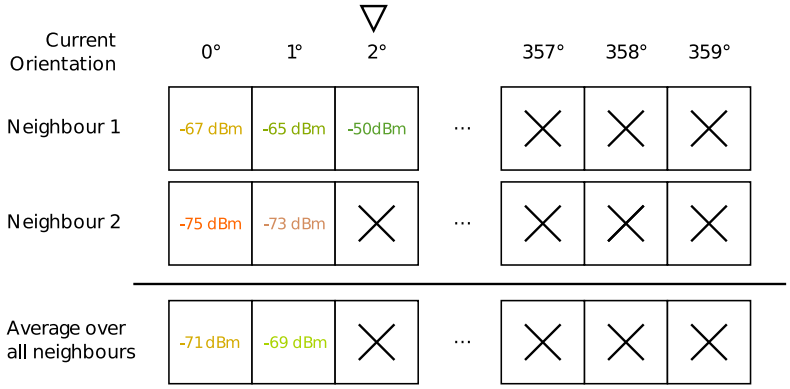


Figure 3: Example of our algorithm execution on agent A_i with two neighbours after a few steps.

After a while, agent A_i has collected power measurements on all its possible orientations and with all its known neighbours, as shown in Fig. 4. Note that the power measurement may not be possible with all the neighbours on all the orientations due to the radio environment, the relative orientation of antennas and the data traffic (that is not controlled in our algorithm). On those cases, agent A_i sets the power level to -100 dBm. In our example, those cases appear with Neighbour 2 on orientations 2° and 359°. It means that, on those orientations, either Neighbour 2 can not communicate with Agent A_i or Neighbour 2 had no data packets to send when agent A_i was measuring.

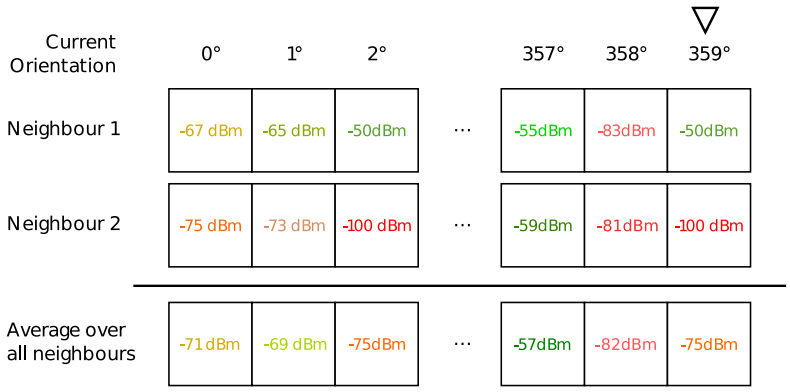


Figure 4: Example of our algorithm execution on agent A_i after a while.

From Fig. 4 we see that the best position in terms of average power over all neighbours corresponds to the orientation 357°. Agent A_i then decides to reach this orientation as shown on Fig. 5.

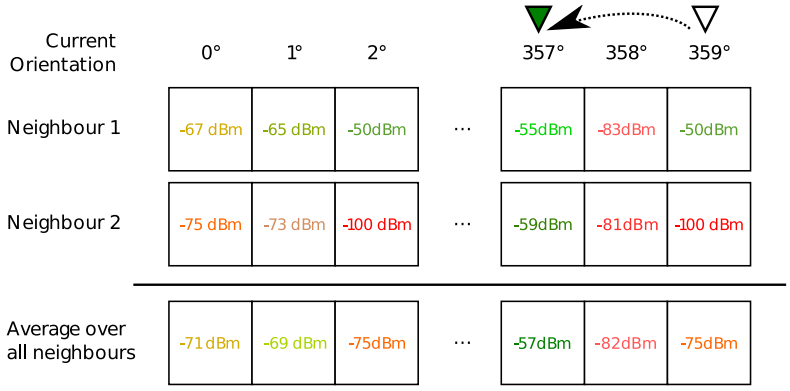


Figure 5: Example of our algorithm execution on agent A_i: orientation selection.

3. Implementation in the ns–3 simulator

Evaluating this algorithm is a difficult task because the algorithm is distributed and executed in parallel by all the agents in an asynchronous way, but also because it depends on the data traffic, the medium access, the used transmission rates, the channel quality and the agent controller. Moreover, we are interested in the network performance. In order to evaluate the proposed antenna orientation solution in a realistic context, a dedicated simulation framework, based on the ns–3 simulator and a modular approach relying on message passing was proposed and developed in [9]. However, for performance and maintainability reasons, we have re-implemented this solution directly in the ns–3 simulator. All the simulation results provided in Section 4 were obtained with this new implementation of our solution in ns–3. We therefore explain this choice and the ns–3 implementation.

3.1. Simulator choice

To accurately simulate networks, and in our case, Wi-Fi networks, and study their performances, the use of a well-known network simulator is in our opinion mandatory. Re-implementing an *ad-hoc* simulator for networking certainly means more control over the whole simulation and processing chain, as well as a mastered complexity, but it also means less accuracy in the simulation, because it is almost impossible to re-implement a significant part of the 802.11 standards on such a project without time, patience, and dedicated engineering resources. And without such resources, it is almost impossible to maintain the development of such a simulator across years, simulator which therefore becomes obsolete and cannot be re-used.

In this work, and because we want to simulate Wi-Fi networks, mobility and antennas, we could have used either the ns–3 or the OMNeT++/INET simulators. On the one hand, OMNeT++/INET provides simulation tools for Wi-Fi networks, antennas, multiple mobility models as well as some obstacles simulation. The simulator's *nodes* are modeled as point-like, and also have an orientation that can be acted upon by the mobility models, for example for a *smart* orientation. On the other hand, OMNeT++/INET lacks recent rate adaptation algorithms used in the field for recent PHYs (*e.g.* the HT and VHT

PHYs), whereas ns–3 supports rate adaptation algorithms such as Minstrel-Ht or Iwl-Mvm-Rs. However, while ns–3 comes with antenna simulation and models, this simulation is only compatible with LTE (Long-Term Evolution) networks, which makes it unsuitable "as is" to evaluate our proposition. Moreover, ns–3 *nodes* are point-like without any notion of orientation.

Because building over ns–3 to implement what was needed to evaluate our solution was deemed easier than building over OMNeT++/INET, we created a specific simulation framework in [9], based on a modular approach offloading the simulation of the antennas and UAV controllers to an external component. Communication between ns-3 and this external component was done using message-passing library, ZeroMQ, and a serialization and de-serialization library, Protocol Buffers. While this framework allowed us to achieve our goals, from a performance point-of-view, this was far from perfect because of the heavy lifting required for the computation of the effective path-loss for each transmitted frame, involving multiple and costly serialization, and message-passing operations.

3.2. Re-implementation in pure *ns–3*

In order to transcribe our simulations in *pure* ns–3, whose programming language is C++, we added and modified multiple ns–3 modules. Quaternion objects and operations over said quaternions were introduced in the core module of ns–3. Quaternions, which are special numbers which encompasses the real and the complex numbers, and specifically unit quaternions, allow us to represent and manipulate spatial rotations and orientations in a convenient way, preventing gimbal locks effects. The base mobility model used by ns–3 to represent the physical nodes, in particular their coordinates, was then enriched by adding a quaternion representing the spatial orientation of the node. More advanced mobility models, such as the ConstantVelocityMobilityModel, and the ConstantAccelerationMobilityModel, have been further enriched by the addition of quaternions representing the nodes rotational speed and rotational acceleration, allowing to model nodes (and UAVs) changing their orientation at a constant speed or constant acceleration. The ns–3 antenna module was modified to add a customizable antenna model, allowing users to specify an antenna model as a custom function, in their scripts, without modifying ns–3 code or re-creating a specific module. Finally, a specific propagation loss model was added to model the antenna effects, modifying the link budget computation done by ns–3 according to the node orientations and antenna models. The modifications to the ns–3 3.33 source tree and examples are available at https://github.com/rgrunbla/ns-3-33.

4. Simulation results

In this section, we present the simulation results obtained on different scenarios in order to evaluate the performance of our solution (Algorithm 1). The different scenarios share some parameters, described in Table 2, but differ in the number of nodes and their positions. We use the ns–3 Friis propagation loss model, also known as the free-space path loss model, coupled with the Nakagami-m fast fading model. These two models accurately model the radio propagation of air-to-air communications between UAVs [1]. All of the

simulations rely either on an isotropic antenna or a directional antenna whose orientation is regulated by Algorithm 1. The first directional antenna used in our simulation represents the Ubiquiti UAP-AC-Mesh Antenna, also named as mesh antenna hereafter, whose radiation pattern is shown on Figure 1. The radiation pattern is provided by the constructor on its website [10] as an *ant* file type, covering the horizontal plane with a granularity of 1°. This antenna has been chosen for its small size and weight, making it compatible with airborne applications, as well as its balanced radiation pattern suitable for mesh applications. The tested antenna has a maximal gain of 4 dBi. Considering a link with two agents equipped with the directional antenna with a 4 dBi gain, 63% of all the possible orientations between the two agents yield a higher gain than a link with two isotropic antennas.

Three rate adaptation, named MinstrelHT, Intel and Ideal, have been considered. MinstrelHT is implemented in the mac80211 component of the Linux kernel and is open source [11]. MinstrelHT is used in the ath9k driver. The Intel rate adaptation algorithm is used on Intel WiFi interfaces and on Intel Aero Ready-to-Fly UAV [6]. Ideal is another rate adaptation algorithm implemented in ns–3 and supporting 802.11ac. These three algorithms have different behaviors and lead to different performance, as studied in [6]. Our evaluation will thus also study the impact of these three algorithms on the performance of our solution.

We present the obtained results in Sections 4.1, 4.2 and 4.3. The initial orientations of the nodes are distributed uniformly over $[0; 2\pi]$, and each simulation is repeated 20 times with different initial orientations. The UAVs start the antenna orientation algorithm after a random time drawn in $[1s; 20s]$. Thus, the UAVs do not necessarily start the algorithm at the same time, mimicking what would be the case practically in a network without any external synchronization. As the initial orientations of the UAVs may or may not allow for communications, UAVs that did not receive any data from neighbours rotate on themselves, at a random speed in the $\pm[0.85; 1.25]$ rad/s interval. This effectively prevents the presence of *sub-optimal* synchronizations between agents, without requiring coordination across the fleet. While in the considered scenarios, the application bitrate is constant, in real scenarios, there could be periods where no UAV needs to transmit. To prevent the penalization of potentially good orientations when this happens, as a UAV could think receiving no data means its own orientation is bad, a minimal background traffic is needed. We therefore decided to have UAVs broadcast minimum size beacons 10 times per second, allowing for such scenarios to be potentially implemented later on.

Given the UAV rotation speed (a complete rotation can be achieved in at most 7.39s at 0.85 rad/s), the worst case in terms of convergence time for the algorithm happens when bad orientations are encountered. In this case, the longer the initial full rotation can take is at most 36s, as one UAV will spent at most 10 execution of the loop in a bad angle, and the algorithm runs at 100 Hz. Given that in the orientation optimization step, the window is halved every time the temporary "optimal" orientation is reached, the final optimal orientation is reached in at most 7.39s after the end of the initial rotation. The maximum convergence time of the algorithm (in terms of orientation) is therefore around 43.39s, which explains why we chose the simulation duration to be 100s (a round number) as this gives plenty of time to the rate adaptation algorithms to be able to converge.

Simulation Parameter	Value
ns–3	version 3.33
Simulation Duration	100s
Wi-Fi Standard	802.11 ac
Wi-Fi MAC type	Ad-Hoc
Rate adaptation algorithm	MinstrelHt, Ideal or Intel
Spatial Streams	2x2:2
Channel Width	20 Mhz
Antennas	Isotropic or Ubiquiti UAP-AC-Mesh or Ubiquiti UAP-AC-Mesh-pro
Propagation Loss Model	Friis
Fast fading	Nakagami
Routing	Static
Application	Constant bitrate, UDP
UAV Rotation Speed	0 rad/s, 1.05 ± 0.20 rad/s
Controller Frequency	100 Hz

Table 2: ns–3 simulation parameters

Several results are reported with the box plot representation. Note that the plain lines represent the quartiles while the dotted line represents the mean value.

4.1. Scenario #1: Simple

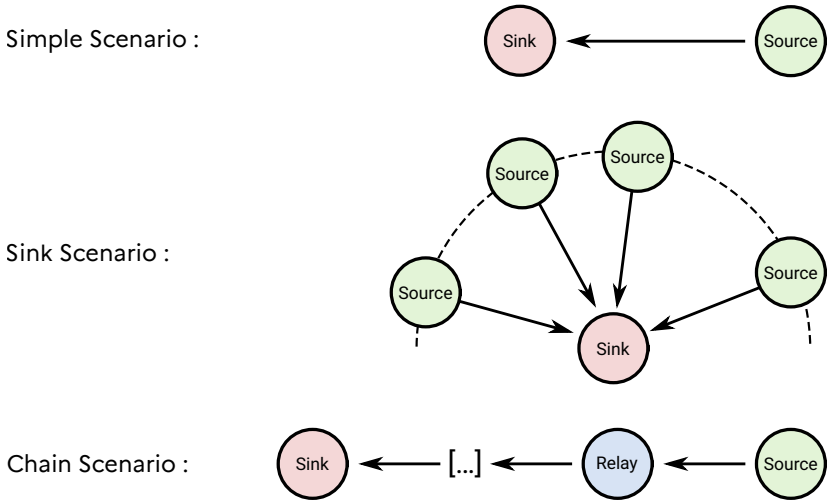


Figure 6: Overview of the studied scenarios

In this scenario, two nodes are separated by a fixed distance, with one node acting as a source and one node acting as a sink, as shown on top of Figure 6. The two nodes are either both equipped with omnidirectional antennas, in which case Algorithm 1 is not used, or both equipped with directional antennas

using our antenna orientation algorithm. The throughput of the source is set to 180 Mbps, which exceeds the maximum physical throughput for the WiFi physical layer parameters used in the simulation, which is 173.3 Mbps.

To highlight the convergence phase of our algorithm, we first show the evolution of the received throughput when no fast fading is adding in the simulation. Figure 7 depicts the received throughput at the sink as a function of time, rate adaptation algorithm and used antenna, for a single simulation, when the distance between the two nodes is $100m$ and without fast fading. We can see that, when the directional antenna is used, : the evolution follows two main phases. The first phase, where the throughput varies a lot, corresponds to the execution of the antenna orientation algorithm: as the channel between the two nodes changes due to the changes of antenna orientation, the rate adaptation algorithms react and change the transmission rates, affecting the received throughput. The second phase starts after the antenna orientation algorithm has converged to its best solution in terms of received power, which results in a rather stable throughput without fast fading. On the other hand, the throughput achieved with the omnidirectional antenna remains stable throughout the simulation except for MinstrelHT which shows a two-phase behavior.

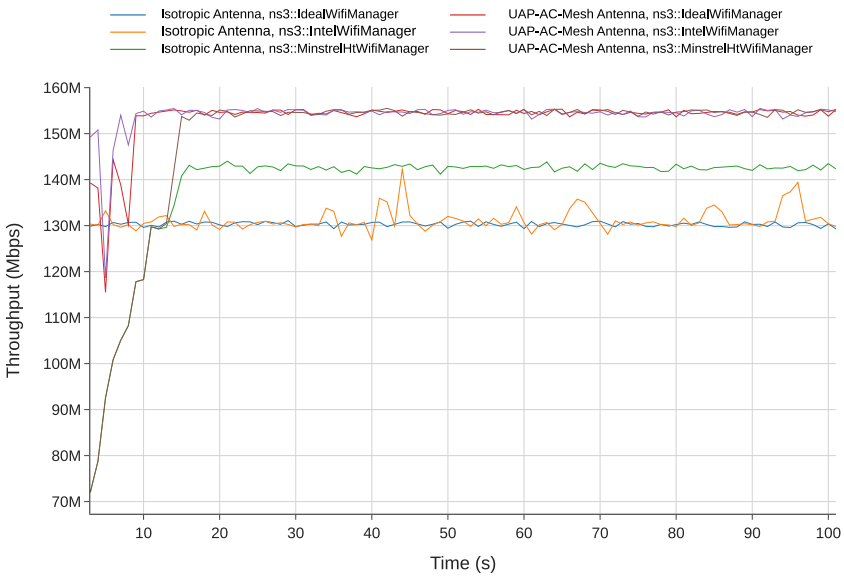


Figure 7: Evolution of the application throughput in function of time for Scenario #1, with 2 nodes 100m apart and a saturating UDP application rate of 180 Mbps, and without fast fading.

From now on, all the simulation results have been obtained with fast fading. We now show the received throughput evolution at the sink when fast fading is active. Figure 8 depicts the received throughput as a function of time, rate adaptation algorithm, and the used antenna, for a single simulation and when the distance between the two nodes is $100m$. We can observe that the three rate adaptation algorithms obtained a lower throughput when omnidirectional antennas are used compared to directional antennas orientated with our

algorithm, except at the beginning of the simulation which corresponds to the algorithm's first phase during which the nodes evaluate different antenna orientations. During this phase, the large variations in throughput correspond to changes in channel quality, which makes the rate adaptation algorithms react implying changes in transmission rates and thus in received throughput. The second phase starts after the antenna orientation algorithm has converged to its best solution in terms of received power. In this simulation, the convergence time is around 10s and is almost the same for the three rate adaptation algorithms. We can note that the Intel rate adaptation algorithm shows more variation than the two other evaluated rate adaptation algorithms. This is consistent with the previous observations made about the algorithm in [6].

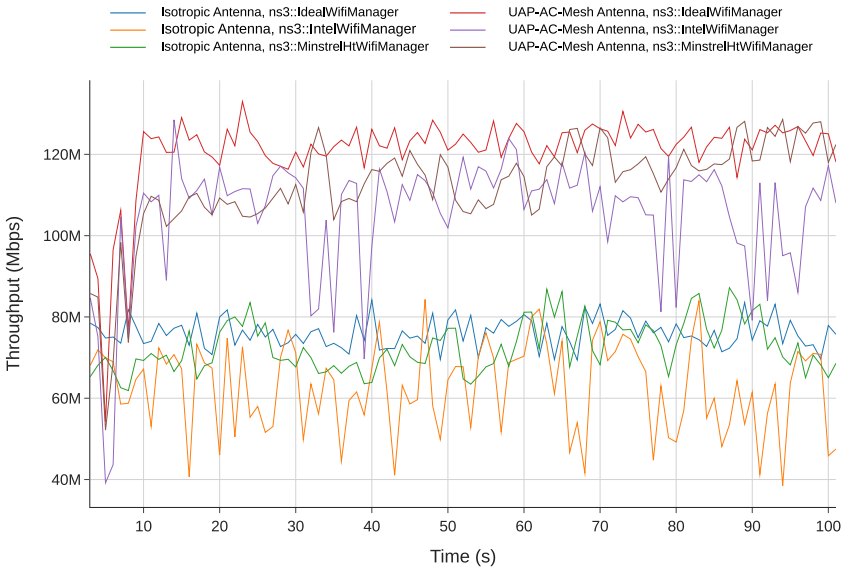


Figure 8: Evolution of the application throughput in function of time for Scenario #1, with 2 nodes 100m apart and a saturating UDP application rate of 180 Mbps.

The convergence time for the antenna orientation algorithm and the convergence time on the received throughput for the simulations using the directional antenna are plotted on Figure 9 (with the box plot representation). The convergence time for the antenna orientation algorithm is the elapsed time between the start of the algorithm and the time when the last agent stops to change its orientation. The convergence time on the received throughput is the elapsed time between the start of the algorithm and the time when the received throughput on the sink is different to at most 20% of the final achieved received throughput. We can note that the convergence time of our algorithm is always smaller than 20s in Scenario #1. The convergence time on the received throughput is also smaller than 20s for Ideal and Minstrel-Ht, underlying than when the orientation of the UAVs converge, those rate adaptation algorithms quickly adapt. Actually, for Minstrel-Ht, the throughput convergence time is better with non-isotropic antennas and mobility than with isotropic antennas and no mobility. Yet, for the Intel rate adaptation algorithm, even if the antenna orientation al-

gorithm converges as quickly as with the other rate adaptation algorithms, the throughput does not converge, as it is still evolving at $t = 100s$, which is the duration of our simulations. As this phenomenon also happens with an isotropic antenna and no controlled orientation algorithm, this can be linked to the bad management of the fast-fading of the Intel RAA we already observed in our previous works.

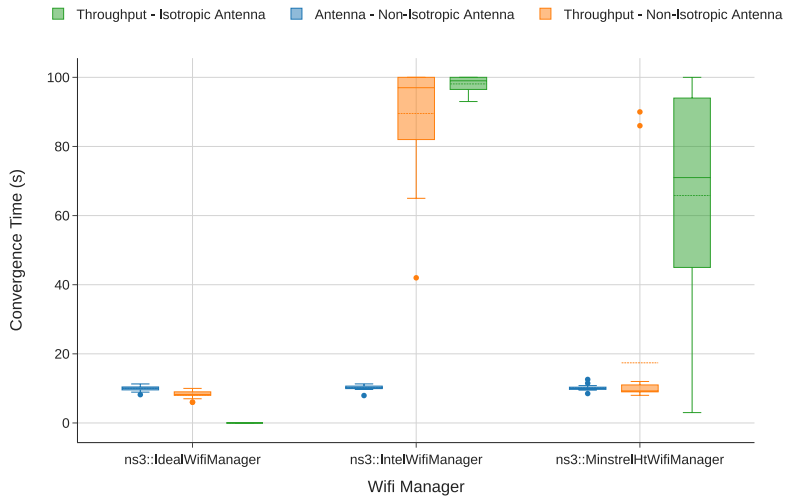


Figure 9: Comparison of the convergence time for the antenna orientation algorithm and application throughput for Scenario #1 at $d = 100m$.

Figure 10 shows the distribution of the achieved throughput for Scenario #1 when the two nodes are $100m$ away. For the oriented antenna, we show the throughput obtained at the initial phase of our algorithm measured between 2s and 4s, corresponding to the results in orange denoted '*Non-Isotropic (Initial Throughput)*', and the achieved throughput measured when the antenna orientation has converged, corresponding to the results in green denoted '*Non-Isotropic (Final Throughput)*'. The obtained results show that our antenna orientation solution improves the achieved throughput whatever the used RAA. As expected, our algorithm always converges to a better throughput than at the beginning of the antenna orientation algorithm. Moreover, the results are also better than when using omnidirectional antennas. For instance, with the Ideal RAA, the mean achieved throughput is 112.9 Mbps with directional antennas compared to 76.3 Mbps with omnidirectional antennas (which represents a gain of +48%), whereas it is 102.5 Mbps with directional antennas compared to 64.2 Mbps with isotropic antennas for the Intel RAA (corresponding to a gain of +60%). For MinstrelHT the use of directional antenna with our orientation algorithm leads to 107.4 Mbps compared to 74.2 Mbps with ominidirectional antennas (gain of +48%). We analyzed the antenna orientations obtained when our algorithm has converged for the different simulation repetitions and for the different RAAs. The values obtained on the antenna orientations vary but are mainly scattered on good positions. These orientations lead to better link qualities which also lead to a use of higher transmission rates, which, at the end,

results in higher achieved throughput. Finally, one can note that the obtained values on the throughput are more dispersed with directional antennas than with isotropic antennas. This is explained by the fact that the obtained orientations vary, which results in different link budgets implying more various throughput, though these latter are, most of the time, better than the ones obtained with isotropic antennas.



Figure 10: Comparison of the achieved throughput for Scenario #1 at $d = 100m$.

4.2. Scenario #2: Sink

In this scenario, one node serves as a sink while 5 other nodes serve as sources. The sources are located on a circle with a fixed radius r, while the sink is located at the center of the circle, as shown on the middle of Figure 6. The sink can be seen as a UAV receiving video feeds from the sources, and sending them to the ground using another network component not studied here. The sink is equipped with an isotropic antenna. We have observed, on the different simulations, that the antenna orientation algorithm converges in less than 40s. The distribution of the average received throughput per link, at the sink, for a radius of 100m and for an application rate at each source of 50 Mbps different application rates at the source, is shown on Figure 11. As for Scenario #1, we compare the results obtained with the isotropic antenna with the non-isotropic antenna for which we study the initial throughput obtained between 2s and 4s and the throughput obtained after the convergence of our algorithm. One can observe an increase in the obtained throughput when using the directional antenna, no matter which RAA is used. Looking at the means, the gains are similar for MinstrelHt and Intel, but a bit smaller with IdealWifi, which nevertheless achieves the best throughput of the three RAAs. We can observe that the final throughputs obtained for MinstrelHt are more scattered than for Intel and Ideal, as expected from the inner working of the algorithm.

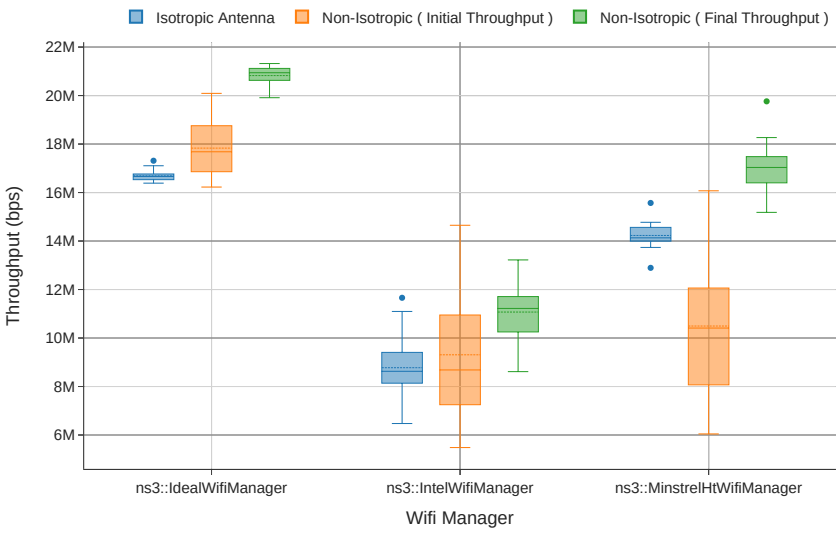


Figure 11: Comparison of the average received throughput per link for Scenario #2 with $r = 100$m and 5 nodes.

4.3. Scenario #3: Chain

In this scenario, one node serves as a source, one node serves as a sink, and the other nodes serve as relays between the source and the sink as depicted in the bottom of Figure 6. The source and the sink are separated by a fixed distance d, and the relays are equidistantly placed between them.

We plot the distribution of the received throughput at the sink on Figure 12, for a distance between the source and the sink of $d = 1000$m, and for 5 nodes in total, that is to say for 3 relays, for an application throughput of 10, 50 or 100 Mbps. The use of the directional antenna with the antenna orientation algorithm improves the overall throughput, for any RAA. The gain is more important for Ideal and for MinstrelHT than for Intel, which can be explained by Intel underperforming in multi-hops scenarios: the gains brought by a controlled orientation are smaller because the maximum throughput achieved by Intel in a multi-hop scenario is smaller. We can also note that, in this scenario, the use of a directional antenna may be harmful, compared to an omnidirectional one, if not used correctly. Indeed, without the use of an antenna orientation algorithm that seeks to optimize the antennas' orientation, the obtained results show that the directional antennas lead to smaller throughput than isotropic ones.

Figure 13 shows the obtained throughput when the chain consists of 10 nodes. With Ideal, our solution still improves the performance compared to the use of an ominidirectional antenna. With MinstrelHT, our antenna orientation algorithm improves the throughput obtained with the oriented antenna, but not when compared to the use of ominidirectinal antenna. With Intel there is no improvement with the directional antennas. With 10 nodes, we observe a high number of retransmissions leading to low data rate selected by the rate adaptation algorithms and thus low throughput. The poor performance of Intel

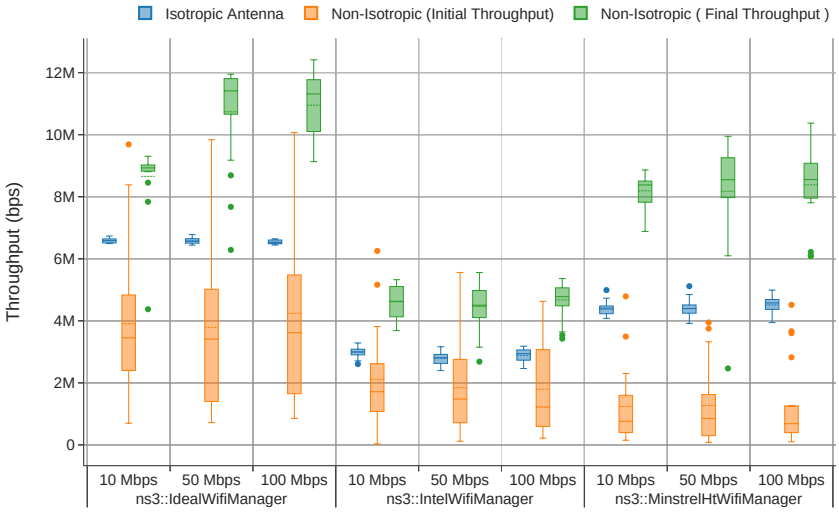


Figure 12: Comparison of the average received throughput at the sink for Scenario #3 with $d = 1000$m, 5 nodes and an application rate of 10, 50 or 100 Mbps.

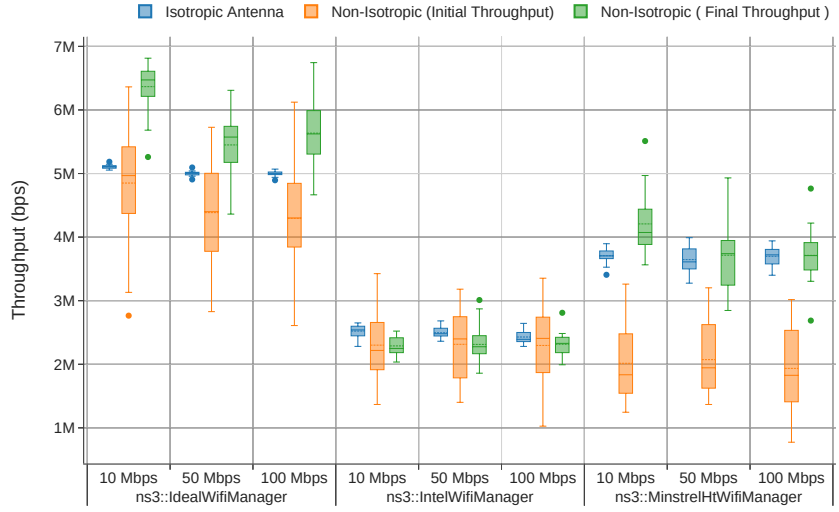


Figure 13: Comparison of the average received throughput at the sink for Scenario #3 with $d = 1000$m, 10 nodes and an application rate of 10, 50 or 100 Mbps.

can be explained by its conservative behavior when too many retransmissions are triggered [6].

4.4. Results with a 3 dBi antenna

We also evaluated our solution with the same pattern radiation as the Ubiquiti UAP-AC-Mesh antenna but with a maximal gain reduced to 3dBi (see Figure 1). With such an antenna, only 38% of the possible antenna orientations between two agents will result in higher network performance than with the isotropic antenna. As for the previous section, we compare the results achieved with an omnidirectional antenna with the ones obtained with the non-isotropic antenna at the early stage of our algorithm (denoted 'Non-Isotropic (Initial Throughput)') and when our algorithm has converged (denoted 'Non-Isotropic (Final Throughput)').

Figure 14 reports the distribution of the achieved throughput between two nodes (Scenario #1) separated of 100m and when the source throughput is 180 Mbps for 20 repetitions of the simulation with different initial orientations. The results show that even if the achieved throughput is smaller than the one obtained with a 4dBi antenna, the antenna orientation algorithm is able to find good orientations resulting in better performance than with omnidirectional antennas. The results also show that it is imperative to optimize the antenna orientation to leverage the antennas' directionality, as does our algorithm.



Figure 14: Comparison of the achieved throughput for Scenario #1 at $d = 100m$ over 20 random initial orientations with the 3 dBi directional antenna.

Figure 15 reports the distribution of the average achieved throughput per link for Scenario #2 with 5 nodes and each source transmitting with an application rate of 50 Mbps. Once more, the results show that our algorithm is able to find good orientations and can improve the overall throughput compared with the omnidirectional antenna. This improvement is more limited with Intel. This

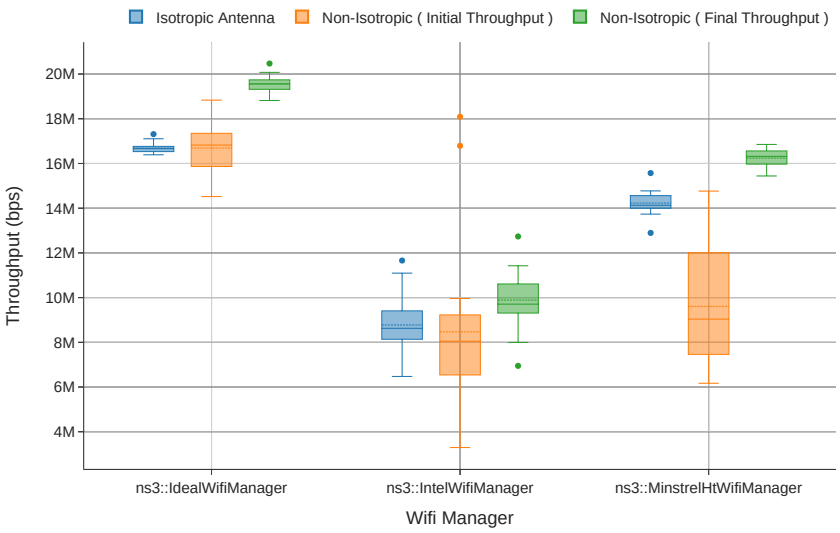


Figure 15: Comparison of the average received throughput per link for Scenario #2 with $r = 100$m, 5 nodes with the 3 dBi directional antenna.

can be explained by the fact that, globally, achieves smaller throughput, which impacts the overall network performance when the performance anomaly arises [12] as this can be the case in this scenario.

Figure 16 reports the distribution of the average achieved throughput per link for Scenario #3 with 5 nodes on a chain whose the source and the sink are separated by 1000m. The source transmits with an application rate of 10, 50 or 100 Mbps. As for the 4 dBi antenna, the results show that our algorithm improves the end-to-end throughput by finding good orientations between neighboring nodes. Such an algorithm is necessary to take advantage of directional antennas.

4.5. Results of an antenna with an irregular radiation pattern

We have also evaluated our antenna orientation solution when used on an oriented antenna with an irregular radiation pattern. We have tested Ubiquiti UAP-AC-Mesh-Pro antenna with a maximal gain of 4dBi (see Figure 1 and [10] for the constructor website). With such an antenna, 59.5% of the possible antenna orientations between two agents will result in higher network performance than with the isotropic antenna, but the good positions are close (in terms of orientation) to worst solutions. Our algorithm has been evaluated on Scenarios 1, 2 and 3 with this antenna. As shown on Figures 17, 18, and 19 our solution is able to improve the throughput of the communicating links, even when the antenna has an irregular radiation pattern.

4.6. Lessons learned from the obtained results

As underlined by Figure 9 and by the the small execution time analysis provided in the beginning of this section, the actual convergence time of the

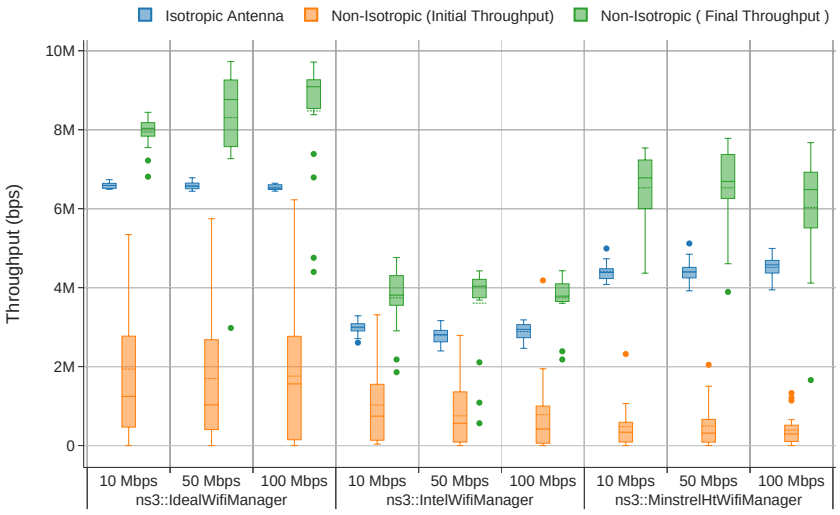


Figure 16: Comparison of the average received throughput at the sink for Scenario #3 with $d = 1000$m, 5 nodes and an application rate of 10, 50 and 100 Mbps, with the 3 dBi directional antenna.

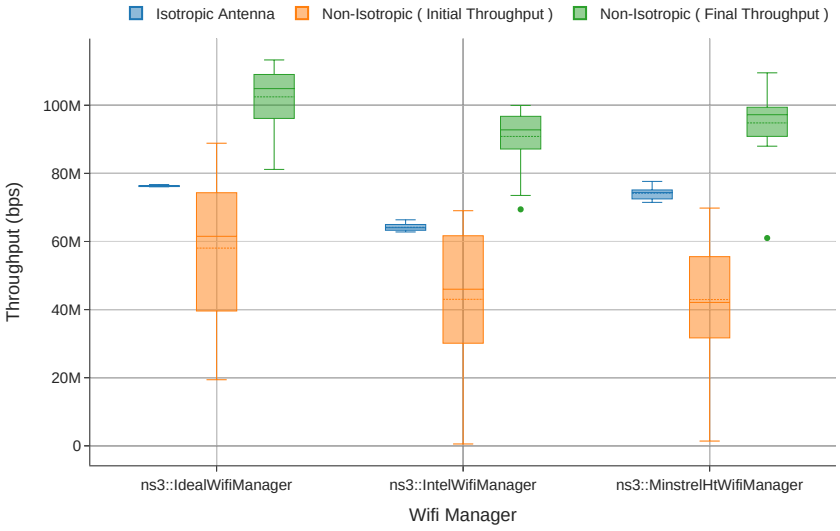


Figure 17: Comparison of the achieved throughput for Scenario #1 at $d = 100m$ over 20 random initial orientations with the UAP-AC-Mesh-Pro directional antenna.

antenna orientation algorithm in the simulation is well below the worst convergence time. This can be linked to the chosen simulation parameters in terms of channel model, antenna gain, and distance between the nodes, which prevents

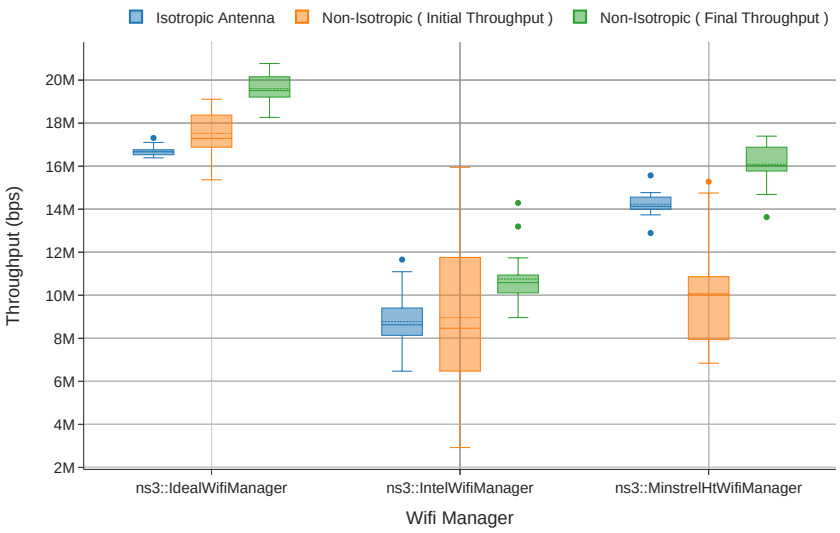


Figure 18: Comparison of the average received throughput per link for Scenario #2 with $r = 100$m, 5 nodes with the UAP-AC-Mesh-Pro directional antenna.

too many orientation combinations to result in some impossibility to establish connections in a non-synchronized way. Indeed, a *perfect* orientation of two nodes who want to communicate is not required to establish communication, which allows to avoid a need for synchronisation. Such synchronisation would be needed in the case of very directional antennas, such as a Yagi antennas for example.

The ability to distinguish between a bad position and an absence of transmission, provided by the use of beacons, is also primordial to our algorithm as it allows to mark bad positions as being bad by verifying no such beacons are received. Even if in the case of our simulation scenarios constant traffic are used, if we were to consider scenarios where traffic alternates between "on" and "off" patterns, without beacons, good positions would be marked as being bad by the inability to identify the absence of a signal as normal.

In the tested scenarios, the obtained results show that the use of directional antennas may be inefficient, compared to the use of isotropic antennas, if the antenna orientation is not controlled. Without any control, the antennas may be in bad relative positions leading to low performance results, in terms of throughput for instance. Taking advantage of the antennas' directionality requires to search the good orientations corresponding to good quality communications.

The proposed antenna orientation solution is agnostic with respect to the rate adaptation algorithms used in the Wi-Fi drivers. The obtained results show that our solution is more efficient than the use of isotropic antennas when Ideal and MinstrelHT are used whatever the targeted scenarios. With the Intel rate adaptation algorithm, the efficiency of our solution is not observed on all the tested scenarios.

We think that this kind of antenna orientation solution would also have an

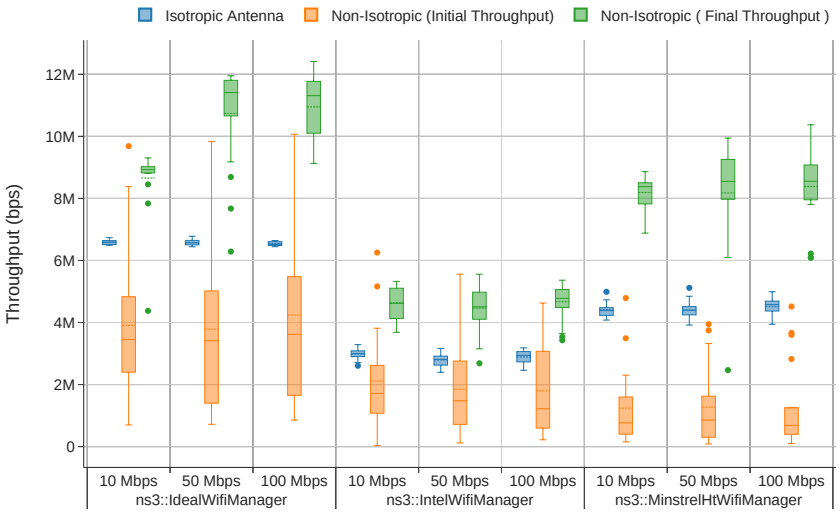


Figure 19: Comparison of the average received throughput at the sink for Scenario #3 with $d = 1000$m, 5 nodes and an application rate of 10, 50 and 100 Mbps, with the UAP-AC-Mesh-Pro directional antenna.

interest when isotropic antennas are used as the environment (UAVs, buildings, etc.) impacts the effective radiation pattern of the antennas. We also think that this solution could be used with other wireless communication technologies than Wi-Fi as soon as they offer point-to-point communications since the proposed algorithm do not use any specific Wi-Fi feature.

5. Related Work

The interest of using directional antennas in UAV networks has been shown in some experiments. In [13], the authors show that, when using Wi-Fi directional antennas, 2 UAVs can communicate with an acceptable throughput (of the order of several Mb/s) even if the distance between the 2 UAVs is large (around 1 km). In these experiments, IEEE 802.11g is used and there is no indication on the used rate adaptation algorithm or if any such algorithm is disabled. In [4], the authors experimentally show the impact of the antenna orientations on the UAV communications. Different effects are studied like the UAV body, the UAV relative direction and the elevation angle on different parameters like the Received Signal Strength and the cross-polarization discrimination. The obtained results show the importance of the antenna orientation and their impact on the expected communication performance.

In [14], an antenna heading control system is proposed for UAVs equipped with directional antenna. This system is mainly based on GPS information, but also on a RSSI (Received Signal Strength Indicator) scan when GPS data are not available. The directional antennas are used for the communications between UAVs, communications realized with the proprietary protocol AirMax.

The RSSI scan algorithm is a two-stage algorithm in which the first step finds a first orientation via a coarse-grained scanning. From this first orientation, the second step reduces the scan step as the search space. The algorithm is designed for a link with 2 UAVs that are able to schedule the start of the algorithm: as one UAV runs the algorithm, the second UAV waits for the execution completion before starting its own run of the algorithm. This synchronization is realized with handshaking signals (via the Xbee system). MATLAB simulations and real experiments are carried out on a scenario with 2 UAVs. The same authors study the same problem in [15], but, in this paper, the authors use a reinforcement learning approach to learn the communication channel model. The proposed solution is validated with 2 UAVs and with a focus on the reached angles and the learned antenna radiation pattern. These two studies are the closest to our work but they differ from ours on different aspects: we consider the WiFi communication protocol and more general scenarios with possibly more than 2 UAVs; the antenna orientation algorithm we propose is local to each UAV without any synchronization between UAVs; we consider the possible use, by the WiFi interface, of a rate adaptation algorithm that may significantly impact the communication performance.

In some papers, the authors consider the possible mobility of the devices to improve the communication performance. This is for instance the case in [16, 17]. But in most cases, terrestrial robots are considered and the objective is to move some robots in some "good" locations in order to obtain more efficient communications. Very often, the antenna orientation and the WiFi rate adaptation are not taken into account as we do in our study.

The current paper is an extended version of a previous paper by the same authors [6]. Compared to [6], we provide, in this work, a more realistic evaluation framework. First the whole solution, initially developed in a dedicated simulation framework, has been directly re-implemented in the ns–3 simulator for performance and maintainability reasons, implying the creation of new modules in ns–3 to make the simulation of spatial rotations possible. Second, we have considered a more realistic propagation model with the addition a fast fading model (Nakagami-m model) to the considered path loss model (Friis model). Third, we desynchronize the start of the antenna orientation algorithm among the nodes with a random start. We think that such an approach is more realistic, in addition to avoid possibly sub-optimal configurations. Finally, contrary to [6], we tested three different antennas with different radiation patterns.

6. Conclusion

In this paper, we have proposed a distributed antenna orientation algorithm to optimize communications inside a fleet of stabilized UAVs. Based on power measurements collected with its neighbours, each UAV seeks to orientate its antenna in a position that optimizes the sum of the powers of the received signals. The optimization phase is based on a distributed hill climbing approach.

The proposed algorithm has been implemented in the ns–3 simulator. This requires to add and modify several ns–3 modules, like, for instance, the use of quaternion objects for simulating spatial rotations and the possibility to manipulate any antenna model. Our solution has been evaluated, via ns–3, on three different scenarios, three different antenna profiles and three different rate adaptation algorithms.

The obtained results show that, on those scenarios, our antenna orientation algorithm significantly improves the overall throughput compared to the use of the same directional antennas whose orientation is not controlled and optimized. The results also show that our algorithm improves, on most of the tested scenarios, the throughput compared to the one obtained with an omnidirectional antenna. We can note that the throughput's improvement happens for the three tested rate adaptation algorithms, even if the Intel algorithm leads to more limited performance, which can be explained by its conservative behavior when the radio channel quality strongly varies.

We think that this work has shown the utmost importance of applying a smart antenna orientation algorithm to leverage the antennas' directed radiation patterns in order to improve the network performance. We now plan to test this solution on a real fleet of UAVs.

Acknowledgment

The authors would like to thank the ANR and the AID for their funding support via the CONCERTO project (ANR-20-ASTR-0003), as the Direction Générale de l'Armement (DGA).

References

[1] E. Yanmaz, R. Kuschnig, C. Bettstetter, Achieving air-ground communications in 802.11 networks with three-dimensional aerial mobility, in: 2013 Proceedings IEEE INFOCOM, IEEE, 2013, pp. 120–124.

[2] S. Hayat, E. Yanmaz, C. Bettstetter, Experimental analysis of multipoint-to-point uav communications with ieee 802.11 n and 802.11 ac, in: 2015 IEEE 26th Annual International Symposium on Personal, Indoor, and Mobile Radio Communications (PIMRC), IEEE, 2015, pp. 1991–1996.

[3] E. Yanmaz, S. Hayat, J. Scherer, C. Bettstetter, Experimental performance analysis of two-hop aerial 802.11 networks, in: 2014 IEEE Wireless Communications and Networking Conference (WCNC), IEEE, 2014, pp. 3118–3123.

[4] M. Badi, J. Wensowitch, D. Rajan, J. Camp, Experimental evaluation of antenna polarization and elevation effects on drone communications, in: Proceedings of the 22nd International ACM Conference on Modeling, Analysis and Simulation of Wireless and Mobile Systems, 2019, pp. 211–220.

[5] H. Qi, Z. Hu, X. Wen, Z. Lu, Rate adaptation with thompson sampling in 802.11ac wlan, IEEE Communications Letters 23 (2019) 1888–1892.

[6] R. Grünblatt, I. Guérin-Lassous, O. Simonin, Simulation and performance evaluation of the intel rate adaptation algorithm, in: Proceedings of the 22nd International ACM Conference on Modeling, Analysis and Simulation of Wireless and Mobile Systems, 2019, pp. 27–34.

[7] C. A. Balanis, Antenna theory: analysis and design, John wiley & sons, 2016.

[8] R. Stuart, N. Peter, et al., Artificial intelligence: a modern approach (2003).

[9] R. Grünblatt, I. Guérin Lassous, O. Simonin, Leveraging antenna orientation to optimize network performance of fleets of uavs, in: Proceedings of the 23rd International ACM Conference on Modeling, Analysis and Simulation of Wireless and Mobile Systems, 2020, pp. 253–260.

[10] U. Networks, UAP antenna radiation patterns (2019). URL http://web.archive.org/web/20200613210252/https://help.ui.com/hc/en-us/articles/115005212927-UniFi-UAP-Antenna-Radiation-Patterns

[11] F. Fietkau, Minstrel ht: New rate control module for 802.11n, https://lwn.net/Articles/376765/.

[12] M. Heusse, F. Rousseau, G. Berger-Sabbatel, A. Duda, Performance anomaly of 802.11b, in: IEEE INFOCOM 2003. Twenty-second Annual Joint Conference of the IEEE Computer and Communications Societies (IEEE Cat. No.03CH37428), Vol. 2, 2003, pp. 836–843 vol.2. doi:10.1109/INFCOM.2003.1208921.

[13] Y. Gu, M. Zhou, S. Fu, Y. Wan, Airborne wifi networks through directional antennae: An experimental study, in: 2015 IEEE Wireless Communications and Networking Conference (WCNC), IEEE, 2015, pp. 1314–1319.

[14] J. Chen, J. Xie, Y. Gu, S. Li, S. Fu, Y. Wan, K. Lu, Long-range and broadband aerial communication using directional antennas (acda): Design and implementation, IEEE Transactions on Vehicular Technology 66 (12) (2017) 10793–10805.

[15] S. Li, C. He, M. Liu, Y. Wan, Y. Gu, J. Xie, S. Fu, K. Lu, Design and implementation of aerial communication using directional antennas: learning control in unknown communication environments, IET Control Theory & Applications 13 (17) (2019) 2906–2916.

[16] M. Gowda, A. Dhekne, R. Roy Choudhury, The case for robotic wireless networks, in: Proceedings of the 25th International Conference on World Wide Web, 2016, pp. 1317–1327.

[17] K. Miranda, N. Mitton, T. Razafindralambo, On the impact of routers' controlled mobility in self-deployable networks, 2015.