Dynamic Routing for Flying Ad Hoc Networks

Stefano Rosati, *Member, IEEE*, Karol Krużelecki, *Member, IEEE*, Grégoire Heitz, Dario Floreano, *Senior Member, IEEE*, and Bixio Rimoldi, *Fellow, IEEE*

Abstract—**This paper reports experimental results on self-organizing wireless networks carried by small flying robots. Flying ad hoc networks (FANETs) composed of small unmanned aerial vehicles (UAVs) are flexible, inexpensive and fast to deploy. This makes them a very attractive technology for many civilian and military applications. Due to the high mobility of the nodes, maintaining a communication link between the UAVs is a challenging task. The topology of these networks is more dynamic than that of typical mobile ad hoc networks (MANETs) and of typical vehicle ad hoc networks (VANETs). As a consequence, the existing routing protocols designed for MANETs partly fail in tracking network topology changes.**

In this work, we compare two different routing algorithms for ad hoc networks: optimized link-state routing (OLSR), and predictive-OLSR (P-OLSR). The latter is an OLSR extension that we designed for FANETs; it takes advantage of the GPS information available on board. To the best of our knowledge, P-OLSR is currently the only FANET-specific routing technique that has an available Linux implementation. We present results obtained by both Media Access Control (MAC) layer emulations and real-world experiments. In the experiments, we used a testbed composed of two autonomous fixed-wing UAVs and a node on the ground. Our experiments evaluate the link performance and the communication range, as well as the routing performance.

Our emulation and experimental results show that P-OLSR significantly outperforms OLSR in routing in the presence of frequent network topology changes.

I. Introduction

In the case of a calamitous event, when ordinary communication infrastructure is out of service or simply not available, a group of small flying robots can provide a rapidly deployable and self-managed ad hoc Wi-Fi network to connect and coordinate rescue teams on the ground. Networks of small unmanned aerial vehicles[1] (UAVs) can also be employed for wildfire monitoring [1], border surveillance [2], and for extending ad hoc networks on the ground [3], [4], [5].

The recent technological progress in electronics and communication systems, especially due to the the widespread availability of low-cost micro-embedded computers and Wi-Fi radio interfaces, has paved the way for the creation of inexpensive flying ad hoc networks (FANETs), however, a challenging networking problem arises.

As pointed out in [6], FANETs are a special case of mobile ad hoc networks (MANETs) characterized by a high degree of mobility. In a FANET, the topology of the network can change more frequently than in a typical MANET or vehicle ad hoc network (VANET). As a consequence, the network routing becomes a crucial task [7], [8]. The network routing algorithms, which have been designed for MANETs, such as BABEL [9] or the optimized link-state routing (OLSR) protocol [10], [11], fail to follow the evolution of the network topology. It is possible to bypass this problem by considering star networks with static routing [12]. However, star architectures restrict the operative area of groups of UAVs, because the nodes cannot fly out of the communication range of the control center. In this paper, we focus on partially-connected mesh ad hoc networks that enable the UAVs to use multi-hop communication to extend the operative area. In this case, the problem of highly dynamic routing must be faced.

A few methods for overcoming the problem caused by a rapidly changing network topology have recently been proposed: Guo at al. [13] present a UAV-aided cross-layer routing protocol (UCLR) that aims at improving the routing performance of a ground MANET network with aid from one UAV. In [14], the authors propose the use of directional antennas and two cross-layer schemes, named Intelligent Media Access Control (MAC), and Directional-OLSR. The latter is an extension of OLSR. The choice of the route is based on flight information (such as attitude variations, pitch, roll and yaw). The authors report only OPNET simulation results. Benzaid et al [15], [16] present an OLSR extension denoted Fast-OLSR that aims at meeting the need for highly dynamic routing in MANETs composed of *fast-moving* and *slow-moving* nodes. This extension increases the rate of the

S. Rosati, K. Krużelecki, and B. Rimoldi are with the Mobile Communications Laboratory (LCM), Swiss Federal Institute of Technology (EPFL), Lausanne, Switzerland. G. Heitz and D. Floreano are with Laboratory of Intelligent Systems (LIS), EPFL, Lausanne, Switzerland. Email: {stefano.rosati, karol.kruzelecki, gregoire.heitz, dario.floreano, bixio.rimoldi}@epfl.ch

[1]Small UAVs are also knows as micro-air vehicles (MAVs)

Hello messages only for the nodes that move faster than a given speed. If the *fast-moving* nodes are a small percentage of the network's nodes, the additional overhead is limited. Otherwise, if the network is composed mainly of UAVs, the overhead grows significantly. In [17], we present an extension to the OLSR protocol named predictive-OLSR (P-OLSR). The key idea of this extension is to use GPS information available on board and to weigh the expected transmission count (ETX) metric by a factor that takes into account the direction and the relative speed between the UAVs.

In this paper, we present field experiments that compare P-OLSR against OLSR. The experiments involved two UAVs and a ground station. The carriers were fixed-wing autonomous planes called *eBees* and developed by SenseFly [18]. Each plane carried an embedded computer-on-module and an 802.11n radio interface. The field-test results show that P-OLSR can follow rapid topology changes and provide a reliable multi-hop communication in situations where OLSR mostly fails. In order to assess the behavior of P-OLSR in larger networks, we carried out MAC-layer emulations considering a network composed of 19 UAVs. To the best of our knowledge, P-OLSR is currently the only FANET-specific routing technique that has an available Linux implementation. The open-source software of the P-OLSR daemon can be downloaded from [19].

The rest of the paper is organized as follows. In Section II, we highlight the differences between OLSR and P-OLSR. In Section III, we specify the modified structure of the *Hello* messages and describe the implementation of the P-OLSR daemon. In Section IV, we describe the testbed and in Section V we describe the experiments and present the results. In Section VI, we present the MAC-layer emulation we used to assess the P-OLSR performance in larger networks. Finally, we draw our conclusions in Section VII.

II. Routing for Flying Ad Hoc Networks

In [12], Frew and Brown analyze the networking for systems of small UAVs. They characterize four different network architectures: *direct-link*, *satellite*, *cellular*, and *ad hoc* (also called *mesh networking*). The *direct-link* and the *satellite* architecture are star networks where all the UAVs are either directly connected to the ground control or to a satellite connected to the ground control. This is a simple network architecture: it does not require dynamic routing, because all the nodes are directly connected with the control center. The nodes require, however, long-range (terrestrial or satellite) links, hence they are not suitable for small UAVs. Furthermore, UAV-to-UAV communication is inefficiently routed through the control center, even if the nodes operate the same area. This might cause traffic congestion in the control center, which is also a weakness of the system in case of attack.

In the *cellular* architecture, the UAVs are connected to a cellular system with many base stations scattered on the ground. UAVs can do a handover between different base stations during the flight. This architecture does not need a single vulnerable control center. However, the operating area of the UAVs is limited by the cellular network extension. In the case of a catastrophic event, the UAV system can be deployed only if the cellular network is present and functioning in the area.

In the *ad hoc* architecture, every node can act as a router. These networks are also know as FANETs: they have no central infrastructure, therefore, they are very robust against isolated attacks or node failures. Moreover, as these networks do not rely on any external support they can be rapidly deployed anywhere. These characteristics, on one hand, make FANETs the most suitable solution for many applications, but on the other hand, they raise a challenging networking problem.

In fact, due to the rapid and erratic movement of the UAVs, the topology of a FANET can vary rapidly and the nodes must react by automatically updating their routing tables. Therefore, in a FANET it is crucial to employ a fast and reactive routing procedure. In [17] we show that some of the most popular routing algorithms for MANETs, such as OLSR and BABEL, fail to track the fast topology changes of a FANET. Similar conclusions are also drawn in [7]. For this reason, in [17], we introduce P-OLSR. To predict how the quality of the wireless links between the nodes is likely to evolve, P-OLSR exploits the GPS information, which is typically available from the UAV's autopilot. For the sake of completeness, in this section we briefly report the definition of ETX, as well as, the key concepts behind P-OLSR.

A. Link-Quality Estimation

OLSR is currently one of the most popular proactive routing algorithms for ad hoc networks. It is based on the link-state routing protocol. The original OLSR design does not consider the quality of the wireless link. The route selection is based on the hop count metric, which is inadequate for mobile wireless networks. However, by using the ETX metric [20], the OLSR *link-quality* extension enables us to take into account the quality of the wireless links. The ETX metric was introduced in [21], and it is defined as

$$\mathrm{ETX}(\mathcal{R}) = \sum_{\eta\in\mathcal{R}} \mathrm{ETX}(\eta) = \sum_{\eta\in\mathcal{R}} \frac{1}{\phi(\eta)\rho(\eta)}, \quad (1)$$

where, $\mathcal{R}$ is a route between two nodes of the network, and η is a hop of the route $\mathcal{R}$. $\phi(\eta)$ is the forward receiving ratio, i. e., the probability that a packet sent through the hop η is successfully received. $\rho(\eta)$ is the reverse receiving ratio, i.e., the probability that the corresponding ACK packet is successfully received. In other words, ETX estimates the expected number of transmissions (including re-transmissions) necessary to deliver a packet from the source to its final destination. Then OLSR selects the route that has the smallest ETX, which is not necessarily the one with the least number of hops. If all the hops forming $\mathcal{R}$ are errorless (i.e., $\phi(\eta) = \rho(\eta) = 1$) the $\mathrm{ETX}(\mathcal{R})$ is equal to the number of hops of $\mathcal{R}$.

The receiving ratios are typically estimated by link-probe messages. The OLSR link-quality extension uses the control messages named *Hello* messages as a link-probe. ϕ is computed by means of an exponential moving average, as follows,

$$\begin{cases} \phi_l = \alpha h_l + (1-\alpha)\phi_{l-1} \\ \phi_0 = 0 \end{cases} , \quad 0 \le \alpha \le 1 , \tag{2}$$

where

$$h_l = \begin{cases} 1 & \text{if the } l\text{-th } \textit{Hello} \text{ message is received} \\ 0 & \text{otherwise} \end{cases} ; \tag{3}$$

and α is an OLSR parameter, named *link-quality aging*, that drives the trade-off between the accuracy and responsiveness of the receiving ratio estimation. On one hand, with a greater α, the receiving ratios will be averaged for a longer time, thus yielding a more stable and reliable estimation. On the other hand, with a lower α, the system will react faster. Another important OLSR parameter is the *Hello Interval* (HI) that indicates how frequently *Hello* messages are broadcasted.

B. Speed-Weighted ETX

In a FANET the nodes move rapidly, for example, the cruising speed of our flying robots is around 12 meters per second. The network topology changes rapidly, and a wireless link between two UAVs can break suddenly. In these networks, the ETX metric might be inadequate, because it is not reactive enough to follow the link variations. Due to the delay introduced by the exponential moving average, a node notices that a certain wireless link has broken with a non-negligible delay. Therefore, for a significant amount of time, it will continue routing packets on a link that is actually broken.

To solve this problem, we modify the ETX metric to take into account the position and the direction of the UAV, with respect to its neighbors. The $\mathrm{ETX}(\eta)$ metric of the hop η between the nodes i and j is weighed by a factor that accounts for the relative speed between i and j as follows:

$$\mathrm{ETX}(\eta) = \frac{\mathrm{e}^{v_\ell^{i,j}\beta}}{\phi(\eta)\rho(\eta)} , \tag{4}$$

where $v_\ell^{i,j}$ is the relative speed between nodes i and j, and β is a non-negative parameter.

If the nodes i and j move closer to each other, the relative speed is negative, thus the ETX will be weighted by a factor smaller than 1. Otherwise, if the nodes i and j move apart from each other, the relative speed is positive, thus the ETX will be weighted by a factor greater than 1. In other words, a hop between two nodes that move closer to each other is preferred rather than a hop between two nodes that move apart, even if they have the same values of ϕ and ρ.

In order to compute the speed-weighted ETX, we assume that every node knows the position of its neighbors. As illustrated in the Appendix III, to distribute the GPS coordinates across the network we add a field to the *Hello* messages.

The instantaneous relative velocity between i and j at time t_i is computed as

$$\tilde{v}_\ell^{i,j} = \frac{d_\ell^{i,j} - d_{\ell-1}^{i,j}}{t_\ell - t_{\ell-1}} , \tag{5}$$

where, t_ℓ and $t_{\ell-1}$ are, the arrival time of the last and second to last *Hello* message. $d_\ell^{i,j}$ and $d_{\ell-1}^{i,j}$ are the corresponding distances between the nodes i and j. As the GPS positions are subject to errors, and gusts of wind can perturb the motion of the UAVs, it is preferable to average the instantaneous speed using a exponential moving average as follows:

$$\begin{cases} v_\ell^{i,j} = \gamma \tilde{v}_\ell^{i,j} + (1-\gamma) v_{\ell-1}^{i,j} \\ v_0^{i,j} = 0 \end{cases} , \quad 0 \le \gamma \le 1 , \tag{6}$$

where γ is a P-OLSR parameter. Acting on the P-OLSR parameter β and γ, we can optimize the routing selection to the cruising speed of the UAVs, and to the chosen HI.

Fixed-wing UAVs require forward motion for flying. They also require a minimum air-speed and turning radius. Therefore the direction of the UAV is a good indicator for predicting its position in the near future, and then to foresee how the link quality is likely to evolve.

III. Implementation Details

In order to implement the P-OLSR protocol, we forked an open-source implementation of OLSR called OLSRd [22]. In the modified version, the *Hello* messages are augmented to contain position information. Thus, every node knows its neighbors' positions and can compute the corresponding ETX according to (4).

A. *OLSRd with Link-Quality Extension*

OLSRd uses link-quality sensing and ETX metrics through the so-called *link-quality* extension [20]. It replaces the hysteresis mechanism of the OLSR protocol with link-quality sensing algorithms that are intended to be used with ETX-based metrics. To do so, the *link-quality* extension uses the OLSR *Hello* messages to probe link quality and to advertise link-specific quality information, (i.e. receiving ratios, ϕ and ρ), in addition to detecting and advertising neighbors. Likewise, it includes the link-quality information also in OLSR Topology Control (TC) messages that are to be distributed to the whole network. Clearly the modified messages are not RFC-compliant anymore because they include new fields for link-quality information. Therefore, all the nodes in the network have to use the link-quality extension.

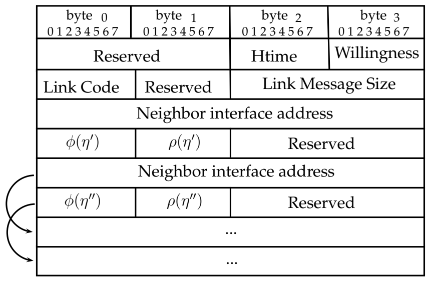


Fig. 1. Format of the original (i.e. OLSRd) *Hello* message.

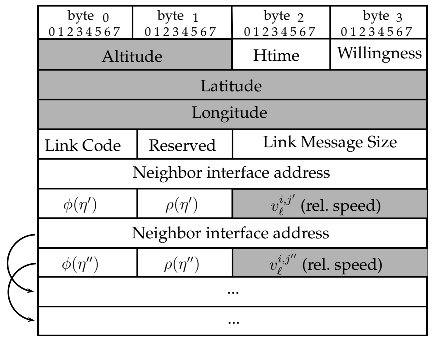


Fig. 2. Format of the modified (i.e. P-OLSRd) *Hello* message.

B. *P-OLSRd Implementation*

To implement P-OLSR, we have to share the coordinates (i.e. longitude, latitude, and altitude) of each node with its neighbors. This is done through the *Hello* messages. Subsequently, each node uses its neighbors' coordinates to compute the corresponding relative speeds, and share them across the whole network via both *Hello* and TC messages.

Fig. 1 depicts the structure of the original *Hello* message in OLSRd. The first block of 8 bytes carries information about the node itself. Note that in this block there are 3 reserved bytes that are not used by the OLSR daemon and are filled with zeros. Then, a block of 8 bytes is appended for each neighbor seen by the node. This block is formatted as follows: 4 bytes are for the IP address of the neighbor, 1 byte is for the forward receiving ratio[2] $\phi(\eta')$, 1 byte is for the reverse receiving ratio $\rho(\eta')$, and 2 bytes that are not used, hence filled with zeros.

Fig. 2 depicts the modified structure of the *Hello* message. We highlight in gray the fields that were added or modified. The first part of the message contains 16 bytes instead of 8. It contains the latitude and the longitude, formatted as single-precision floating-point numbers that occupy 4 bytes each[3]. The altitude is formatted as a 16-bit fixed-point number that replaces the 2 reserved bytes not used by the OLSR daemon. Similarly to the original *Hello* message, a block of 8 bytes is appended for each neighbor seen by the node. The only difference here is that we use the 2 empty bytes to communicate the averaged relative speed between the nodes. The speed is formatted as a 16-bit fixed-point number.

The size difference between the original and the modified *Hello* message is 8 bytes, independently of the numbers of nodes in the network. The *Hello* message is encapsulated into a UDP datagram that, in turn, is encapsulated in an IP packet and then into a 802.11 frame. For medium and large networks, the additional 8 bytes constitute a negligible overload compared to the total size of the frame.

Figs. 3 and 4 illustrate the structure of the TC message for OLSRd and P-OLSRd respectively. They differ only by one field: P-OLSRd exploits the 2 reserved bytes to convey the averaged relative speed formatted as a 16-bit fixed-point number. The original and the modified TC message have the same size.

[2] η' is the hop between the node the is producing the *Hello* message and its neighbor.

[3] In order to have at least one-meter precision, we could have formatted the latitude and longitude with 26-bit fixed-point representation each. This, however, would have required rearranging the latitude and longitude, and adding some padding in order to complete the byte.

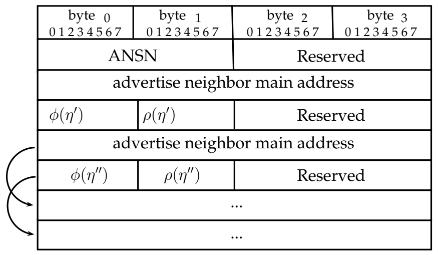


Fig. 3. Format of the original (i.e. OLSRd) *Topology Control* message. ANSN stands for Advertised Neighbor Sequence Number.

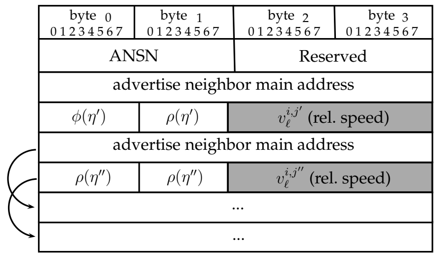


Fig. 4. Format of the modified (i.e. P-OLSRd) *Topology Control* message.

IV. UAV Testbed

We can distinguish two main types of small UAVs: rotary-blade and fixed-wing. Rotary-blade UAVs, which are also know as mini-copters or multi-copters, fly using the lift force generated by one or more blades, like helicopters. Typically, rotary-blade UAVs are composed of four blades, in this case they are know as quadri-copters. They are able to takeoff and land vertically, fly in every direction and maintain a fixed position in the air.

Fixed-wing UAVs fly by exploiting the lift force generated by the forward speed of the vehicle in the air, hence, forward motion is required for flying. They have a minimum speed and turning radius and can fly at higher speeds and with a higher energy efficiency compared to rotary-blade UAVs. Therefore, they can cover a greater area, but the network topology is likely to change very rapidly. This makes the ad hoc networking of fixed-wing UAVs more challenging and, at the same time, more interesting.

In this work, we address fixed-wing UAVs instead of those with rotary-blades. For the experiments we use two fixed-wing UAVs, named *eBee*, developed by SenseFly [18]. The vehicle's body is made of expanded polypropylene (EPP), and it has a single rear-mounted propeller powered by an electric motor. The *eBee* platform is illustrated in Fig. 5(a). It has an integrated autopilot capable of flying with winds up to 12 m/s, at a cruising speed of about 57 km/h, with an autonomy of 45 minutes. In case of emergency, they can be remotely controlled up to a distance of 3 km via a Microhard Systems Nano n2420 [23] link connection, and this low data-rate control link is only used for controlling the plane. The data produced by the UAV is instead carried by the 802.11n link. Due to its small dimensions (the wingspan is 96 centimeters) and weight (under 630 g), flying *eBees* are not considered to be dangerous. In some countries (e.g., Switzerland) they can be used without specific authorization.

On each plane, we mount a Gumstix Overo Tide [24], an ARM-based computer-on-module produced by Gumstix Inc. The computer runs a customized Linux distribution (kernel version 3.5.0), and it is connected to a compatible expansion board that we designed and produced. The expansion board incorporates a USB 2.0 hub with four powered ports and a serial port connected to the autopilot. Using this serial port, the Gumstix can fetch the current GPS data, as well as other flight parameters. The computer is also connected via USB to a HD camera and to a Wi-Fi radio interface. The embedded computer setup is shown in Fig. 5(b).

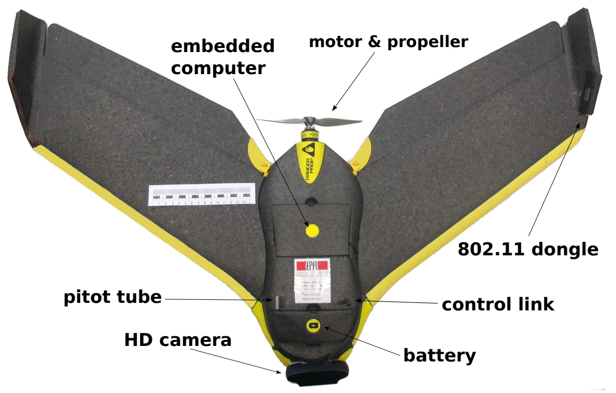
The Wi-Fi radio interface is the Linksys AE3000 USB dongle, using 802.11n. Despite the small dimensions, they support a MIMO system with three antennas. During the experiments, we enabled 802.11n space-time block coding (STBC) to exploit channel diversity. The transmission bandwidth was set to 20 Mhz in the 5 Ghz unlicensed band. We used QPSK modulation, code rate 1/2, and one spatial stream (MODCOD 1).

V. Experiments

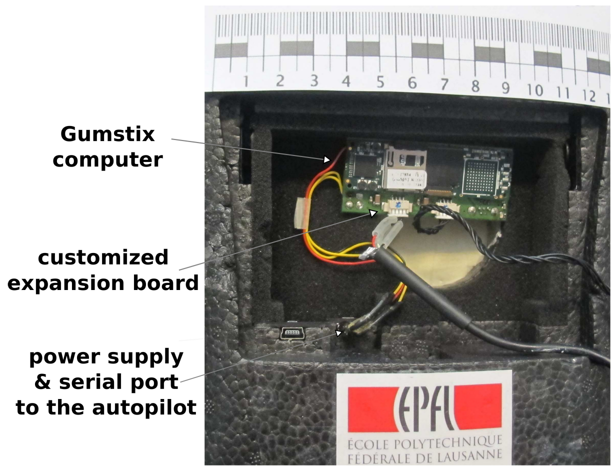
In this section, we describe the two experiments that we carried out on the EPFL campus. The aim of the first experiment was to characterize the end-to-end link performance between an UAV and a node on the ground. Specifically, we determined the communication range in which the two nodes can communicate with a tolerable amount of lost packets. In the second experiment, we compared the OLSR protocol with P-OLSR protocol.

A. Link Performance Assessment

In this experiment, we had a 2-node network composed of a node on the ground and a flying UAV. The UAV flew between two checkpoints positioned 450 meters apart. For both checkpoints, we set a turning radius of 30 meters, and the height-above-ground was 75 meters. The node on the ground was located at the center of one of the checkpoints. Using *iperf*, we took one measurement of the link quality per second. Every second, the UAV sent 85 UDP datagrams (totaling 1



(a) SenseFly *eBee* with HD camera, Linksys AE3000 USB dongle, and Gumstix computer-on-module.



Gumstix computer
customized expansion board
power supply & serial port to the autopilot

(b) Computer and expansion board in the rear compartment of the UAV.

Fig. 5. UAV platform used in the experiments.

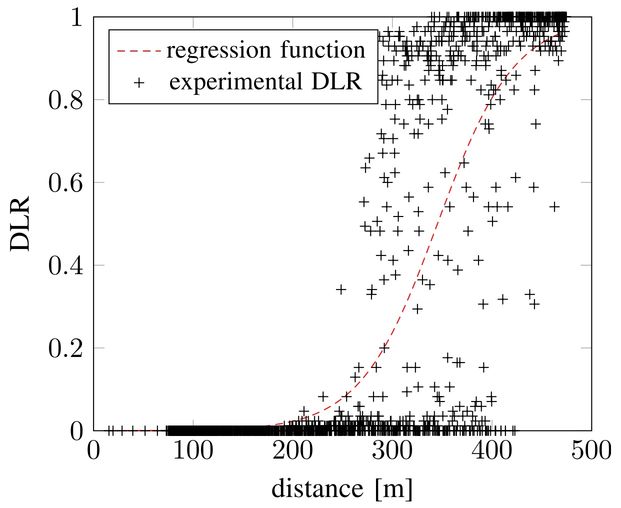


Fig. 6. Experimental datagram loss rate vs. distance. The black crosses are the DLR measured values, and the dashed red line is the corresponding non-linear least-square regression function.

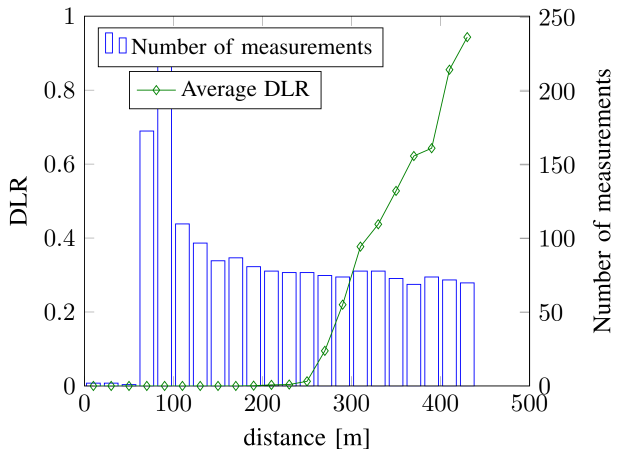


Fig. 7. Average datagram loss rate vs. distance. The blue bars represent the number of measurements per distance interval. The green solid line is the average DLR.

Mbit) to the node on the ground. All the datagrams received with a delay greater than 5 seconds were considered lost. We also counted as correctly received the datagrams that arrived out of order (provided that their delay was smaller than 5 seconds). This is what can be tolerated by a video streaming, where the video is played with a 5-second delay. Every second, we computed the datagram loss rate (DLR), which is the ratio between the datagram lost and their total number.

The results are reported in Fig. 6. The black crosses represent the DLR measured at a given distance. The dashed line represents the corresponding non-linear least-square regression function. The function that we use as a model is

$$\frac{1}{1+\mathrm{e}^{-(p_1+p_2 d)}}\,, \tag{7}$$

where d is the distance in meters and p_1 and p_2 are the coefficients to be estimated. Using a non-linear fitting method, we compute the value of p_1 and p_2, thus yielding the best fitting in the least square sense. They are $p_1 = 8.9$ and $p_2 = 0.025$.

In Fig. 7, we quantize the distance between transmitter and receiver with a bin width of 20 meters. For each bin we compute the average DLR. The resulting curve is plotted in green. We also report the number of measurements per bin.

As we can see from these results, the connection is good when the distance is shorter than 250 meters. In this region, the observed DLR is always lower than 0.2. We observe a transition from 250 meters to 300, where the DLR can be as high as 1, but on average it is lower than 0.3. When the distance is greater than 300 meters, the connection degrades significantly and the DLR is often close to 1. The average DLR is 0.5 at 350 meters. We observe some sporadic cases of a good connection even when the distance is greater than 400 meters.

B. Routing Performance Assessment

In the second experiment, we compared the routing performance of OLSR with link-quality extension and P-OLSR. For both OLSR and P-OLSR, we set the HI equal to 0.5 seconds. This value is a good trade-off between the amount of overhead and the reactivity-speed of the algorithm. Using the default HI value, which in OLSRd is equal to 2 seconds, the algorithm would have been too slow to pursue the topology changes. As P-OLSR exploits the GPS information to estimate the link-quality evolution, it could work also with a longer HI. Nevertheless, for the sake of fairness, we use the same value for both algorithms.

We set the *link-quality aging*, α, equal to 0.2 for OLSR, and to 0.05 for P-OLSR. This parameter controls the trade-off between the accuracy and the responsiveness of the receiving ratio estimation. As before, as P-OLSR can predict the link-quality evolution, it can work with a smaller α, yielding a more accurate estimation of the receiving ratio. OLSR, on the contrary, needs an greater aging value to increase its responsiveness, at the expense of accuracy. The other P-OLSR parameters were $\beta = 0.2$, and $\gamma = 0.04$.

We had a network of three nodes: one fixed destination on the ground (node 1), one flying UAV source (Node 2), and one flying UAV relay (Node 3). Node 1 was on the terrace of the BC building on the EPFL campus (latitude: 46.51843° N; longitude: 6.561591° E). The UAV source, Node 2 flew following a straight trajectory of 600 meters west from Node 1, then it returned to the starting point. The UAV relay, Node 3, followed a circular trajectory of radius 30 meters and centered at 250 meters west from Node 1. Both the UAVs flew about 75 meters above the ground, and the terrace of the BC building is approximately 10 meters above the ground. The actual trajectories followed by the planes are illustrated in Fig. 8.

We performed 10 loops for each routing algorithms. The loop-time is affected by the weather conditions, especially the wind. During our experiment, the eBee took on average 120 seconds to complete a loop. The time difference between the fastest and the slowest loop was around 10 seconds. In each loop, Node 2 changed its routing to reach Node 1, from a direct connection to a two-hop connection, and vice versa. Therefore the network topology was expected to change two times during the each loop.

Fig. 9 shows the evolution of the DLR during the 10 runs. For OLSR, we notice some peaks of the DLR that correspond to the moments when the routing algorithm has to switch from the direct link to a two-hop. This

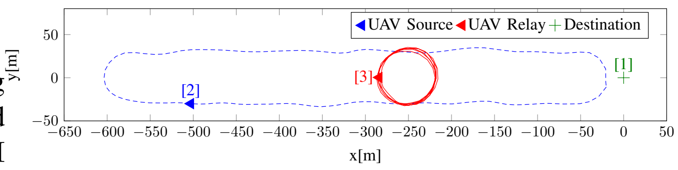


Fig. 8. Trajectories of the UAVs during the network field experiments.

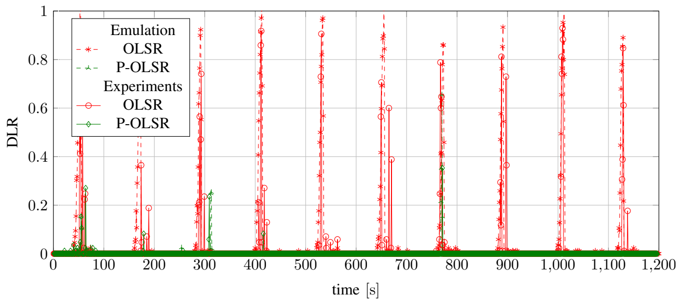


Fig. 9. Evolution of the average DLR. Dashed lines report the results of MAC-layer emulations, solid lines report the results of field experiments.

happens because OLSR takes several seconds to detect that a wireless direct-link is broken. This translates into an interruption of the service. P-OLSR, however, reacts promptly to topology changes. As we can see from the field-test results, P-OLSR is able to predict a change in the topology and reacts before the previous link breaks. The DLR peaks that we see on the P-OLSR results are due to the fading of the wireless channels rather than to incorrect routing.

VI. MAC-layer Emulations with Larger Networks

In this section, we examine the behavior of P-OLSR operating on larger FANETs. When many UAVs are involved, field experiments become expensive. For this reason, we analyze the performance of P-OLSR via a network emulation platform that integrates all the testbed aspects.

A. Emulation Platform

In order to analyze the routing performance in medium/large FANETs, we developed the emulation platform illustrated in Fig. 10. It creates a Linux container (LXC) for each node of the network. The nodes are connected using a MAC-layer real-time emulator called Extendable Mobile Ad-Hoc Network Emulator (EMANE). EMANE is an open-source framework, developed primarily by Naval Research Laboratory [25]. The MAC and the physical layers are emulated, whereas

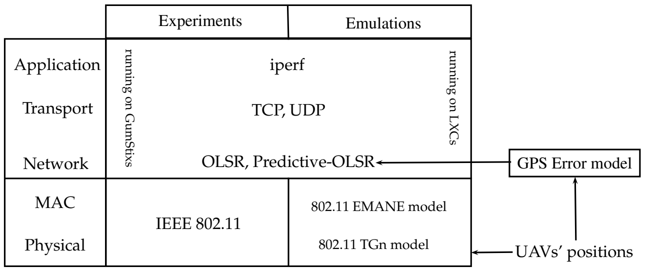


Fig. 10. Emulation platform.

the remaining layers use the real software implementations used by the Linux machine. For a propagation channel model, we use the IEEE 802.11 TGn model D, defined in [26]. This model was proposed for outdoor environments with line-of-sight conditions. EMANE imports the positions of the UAVs from log files and uses them to compute the pathloss for each link. These log files can be obtained from real-flight data logs, or by a flight simulator that reproduces realistic flight conditions. Before passing the UAV's position to the P-OLSR daemon, we add an error to take into account imperfect GPS receivers. We model GPS errors following the statistical characterization of GPS error provided in [27].

B. Emulation Results

We carried out two network emulation campaigns. In the first one, we reproduced the conditions we had in the experiments. This was done to test the emulator. We used the positions reported in the log files of the actual experiments. Fig. 9 shows emulation results (dashed lines) side-by-side with the experimental results (solid lines) for OLSR and for P-OLSR. The emulation results match with the experimental ones.

In the second emulation campaign, we assessed the behavior of P-OLSR in larger networks and we analyzed the role of various parameters, namely the *Hello interval* (HI), the link-quality aging (α), and the P-OLSR specific parameters (β and γ). The network consisted of 19 moving UAVs. One UAV (Node 2) scanned a rectangular area of 1200 square meters, by following the trajectory plotted with a dashed line in Fig. 11. It took 380 seconds to complete the trajectory. The other UAVs (Nodes $1, 3, \dots, 19$) are uniformly spread within the area. They circulated around the respective waypoints, reproducing the behaviour of actual fixed-wing UAVs. The radius of the circular trajectories was 30 meters, the speed was 12 m/s and the initial phase was uniformly distributed between zero and 2π. The distance between the relays was such that only the closest neighbors had a direct

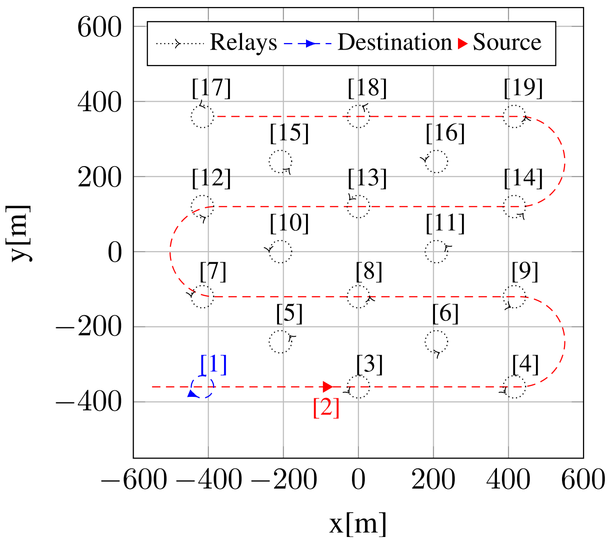


Fig. 11. Emulations set-up. A network composed of 19 nodes. Node 2 is flying with a speed of 12 m/s following the trajectory markers with red dashed line.

link. For example, node 10 can communicate directly only with Nodes 5, 7, 8, 12, 13, and 15.

As we did in the experiments, we sent 85 UDP datagrams per second (totaling 1 Mbit) from Node 2 to Node 1, using *iperf*. In order to have a more compact representation of the results, we compared the network performance in terms of average outage time. We say that an outage occurs when the DLR becomes greater than 0.2. The outage duration of such an event is the length of the interval during which the DLR stays above 0.2. The outage time of a run is the sum of all the outage durations. In order to average the results, we repeated the emulation 10 times for each configuration.

In Figs. 12 and 13 we compare OLSR and P-OLSR for different durations of the HI while keeping the aging parameter α equal to 0.2. We present several P-OLSR configurations: In Fig. 12, we maintain a fixed value of β, ($\beta = 0.2$) and we vary the speed aging parameter γ. In Fig. 13, we fix γ ($\gamma = 0.08$) and we vary the value of β. Notice that P-OLSR reduces drastically the outage time compared to OLSR. In all its configurations, P-OLSR cuts down the outage time by at least 85%. Considering only the best performing P-OLSR configurations, the outage reduction is about 95%, 92%, and 90%, for HI durations equal to 0.5, 1, and 2 seconds, respectively.

In Fig. 12, we compare P-OLSR configurations for different values of γ. We see that when the HI is short, it is convenient to decrease the value of γ, and vice versa. The reason is that if the *Hello* message rate is low, it is convenient to have a fast aging of the previous speed estimates. In Fig. 13, we examine the role of the parameter β. We notice that when the HI is long, it is

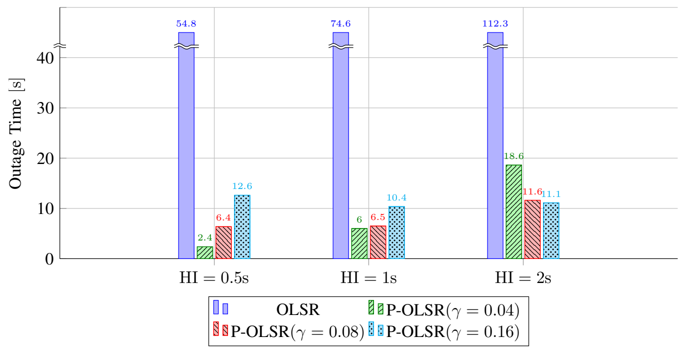


Fig. 12. Average outage time for OLSR and P-OLSR with $\alpha = 0.2$, $\beta = 0.2$, and different HI values.

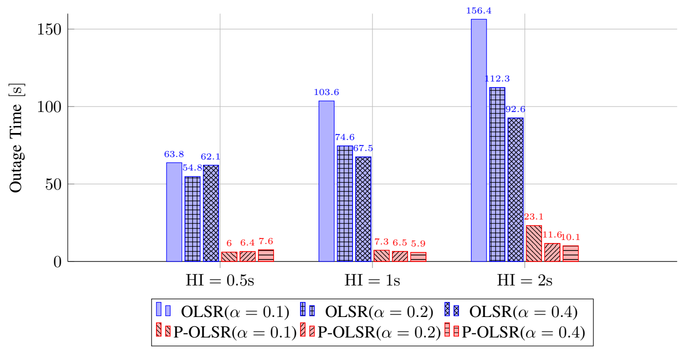


Fig. 14. Average outage time for OLSR and P-OLSR with different HI values and α.

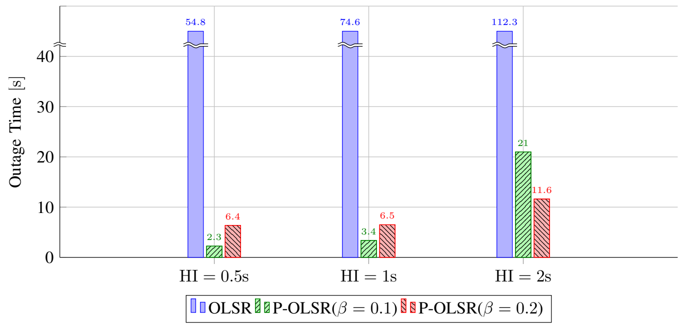


Fig. 13. Average outage time for OLSR and P-OLSR with $\alpha = 0.2$, and $\gamma = 0.08$, and different HI values.

more convenient to increase β, meaning that the speed term has a greater weight in the EXT computation, (see Eq. (4)). Intuitively, when the *Hello* message rate is low, the link-quality estimation is slow in tracking the channel fluctuations, therefore it is convenient to give more weight to the speed term than to the link-quality term.

In Fig. 14, we can examine the role of the HI and the link quality aging, α. We compare P-OLSR and OLSR considering different values of aging and different HI durations. To avoid overcrowding the plots, we consider only one P-OLSR configuration, that is ($\beta = 0.2$ and $\gamma = 0.08$).

As we notice in Fig. 14, the shorter the HI the better the performance. However the improvement brought by halving the HI from 1 to 0.5 seconds might not be large enough to justify the increase of the signalling traffic. This is more evident when P-OLSR is used. In fact, by exploiting the GPS information, P-OLSR is able to track the evolution of the network topology also when the *Hello* message rate is lower. As an example, we note that P-OLSR with HI of 2 seconds reduces the outage time by 80% with respect to OLSR with HI = 0.5 seconds, even if it generates about 1/4 of the *Hello* messages.

We also compare the performance in terms of goodput. In perfect link conditions, the goodput is equal to 1 Mbit/s. In Fig. 15, we plot the goodput achieved by OLSR and P-OLSR during an emulation with HI = 1 second, $\alpha = 0.1$, and the specific-P-OLSR parameters $\gamma = 0.08$, $\beta = 0.2$. As we can see from the plot, the goodput achieved by OLSR is unstable. It often drops below 0.6 Mbit/s, and it also reaches zero. This is because OLSR does not reacts promptly to the topology changes. On the other hand, P-OLSR, which reacts promptly to topology changes and avoids outages, achieves a more stable goodput. It never drops under 0.6 Mbit/s and most of the time it is above 0.8 Mbit/s. The goodput is lower than 1 Mbit/s because some packets are lost due to random channels fluctuations. In the plot, we report also the average goodput over the whole emulation duration, which is 0.83 Mbit/s for OLSR and 0.95 Mbit/s for P-OLSR. All the other cases behave similarly. For the sake of more compact representations, for the other cases we report the average goodput over the emulation duration (Figs. 16-18). As before, we repeated the emulation 10 times for each configuration and report the average results. In Figs. 16 and 17, we compare OLSR and P-OLSR for several HI values while keeping $\alpha = 0.2$. In Fig. 16, we fix the value of β, ($\beta = 0.2$) and vary γ. In Fig. 17, we fix the value of γ ($\gamma = 0.08$) and vary β. We see that in all cases, P-OLSR increases the average goodput compared to OLSR. The gap is greater when the *Hello* message rate is lower (HI = 2 seconds), because OSLR is slower to reacts to topology changes. In Fig. 18, we compare P-OLSR and OLSR considering different values of aging and different HI durations. Again, to avoid overcrowding the plots, we consider only the P-OLSR configuration with $\beta = 0.2$ and $\gamma = 0.08$. Once again, P-OLSR outperforms OLSR in every configuration. The gap is smaller when the *Hello* message rate is high.

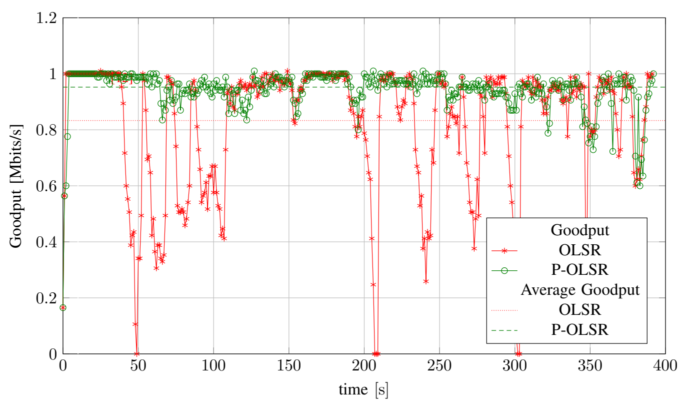


Fig. 15. Goodput during an emulation, considering HI = 1 second, $\alpha = 0.1$, $\beta = 0.2$, and $\gamma = 0.08$.

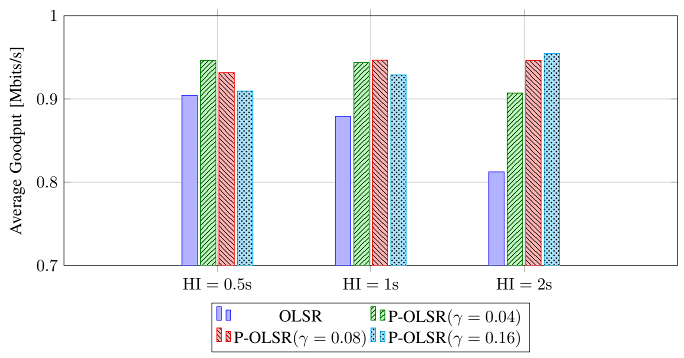


Fig. 16. Average goodput for OLSR and P-OLSR with $\alpha = 0.2$, $\beta = 0.2$, and different HI values.

VII. Conclusion

In this paper we compared the performance of P-OLSR and that of OLSR in a FANET composed of small fixed-wing UAVs.

Such networks are characterized by a high degree of mobility, which constitutes a challenge to the routing protocol. Routing protocols designed for MANETs mostly fail in tracking the evolution of the network topology. We address this problem by designing an OLSR extension, called P-OLSR: it takes advantage of the GPS information to predict how the quality of the wireless links will evolve. The network emulations and fields experiments confirm our expectations. With P-OLSR, the routing follows the topology changes without interruptions, which is not the case with OLSR. We make use of P-OLSR to improve the routing in the SMAVNET II project [19]. SMAVNET II is a research project for developing a self-organizing UAV network.

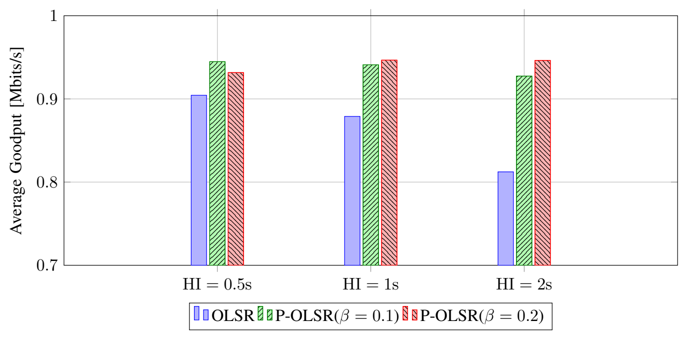


Fig. 17. Average goodput for OLSR and P-OLSR with $\alpha = 0.2$, and $\gamma = 0.08$, and different HI values.

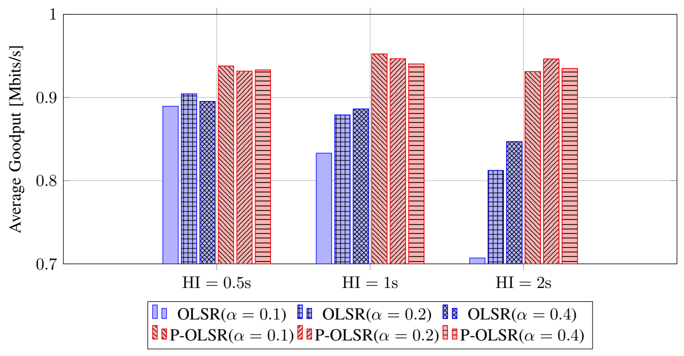


Fig. 18. Average goodput for OLSR and P-OLSR with different HI values and α.

Acknowledgments

This work is supported by armasuisse, competence sector Science+Technology for the Swiss Federal Department of Defense, Civil Protection and Sport.

References

[1] C. Barrado, R. Messeguer, J. Lopez, E. Pastor, E. Santamaria, and P. Royo, "Wildfire monitoring using a mixed air-ground mobile network," *Pervasive Computing, IEEE*, vol. 9, no. 4, pp. 24–32, October 2010.

[2] Z. Sun, P. Wang, M. C. Vuran, M. Al-Rodhaan, A. Al-Dhelaan, and I. F. Akyildiz, "BorderSense: Border patrol through advanced wireless sensor networks." *Ad Hoc Networks*, vol. 9, no. 3, pp. 468–477, 2011. [Online]. Available: http://dx.doi.org/10.1016/j.adhoc.2010.09.008

[3] I. Rubin and R. Zhang, "Placement of UAVs as Communication Relays Aiding Mobile Ad Hoc Wireless Networks," in *Military Communications Conference, 2007. MILCOM 2007. IEEE*, Oct 2007, pp. 1–7.

[4] E. P. de Freitas, T. Heimfarth, I. F. Netto, C. E. Lino, C. E. Pereira, A. M. Ferreira, F. R. Wagner, and T. Larsson, "UAV relay network to support WSN connectivity." in *ICUMT*. IEEE, 2010, pp. 309–314. [Online]. Available: http://dx.doi.org/10.1109/ICUMT.2010.5676621

[5] F. Jiang and A. Swindlehurst, "Dynamic UAV relay positioning for the ground-to-air uplink," in *GLOBECOM Workshops (GC Wkshps), 2010 IEEE*, Dec 2010, pp. 1766–1770.

[6] I. Bekmezci, O. K. Sahingoz, and S. Temel, "Flying Ad-Hoc Networks (FANETs): A survey." *Ad Hoc Networks*, vol. 11, no. 3, pp. 1254–1270, 2013. [Online]. Available: http://dx.doi.org/10.1016/j.adhoc.2012.12.004

[7] O. Sahingoz, "Mobile networking with UAVs: Opportunities and challenges," in *Unmanned Aircraft Systems (ICUAS), 2013 International Conference on*, May 2013, pp. 933–941.

[8] K. Zhang, W. Zhang, and J.-Z. Zeng, “Preliminary Study of Routing and Date Integrity in Mobile Ad Hoc UAV Network,” in *Apperceiving Computing and Intelligence Analysis, 2008. ICACIA 2008. International Conference on*, Dec 2008, pp. 347–350.

[9] J. Chroboczek, “The Babel Routing Protocol,” RFC 6126, Apr. 2011. [Online]. Available: http://www.rfc-editor.org/rfc/rfc6126.txt

[10] T. Clausen and P. Jacquet, “Optimized Link State Routing Protocol (OLSR),” RFC 3626, Oct. 2003. [Online]. Available: http://www.rfc-editor.org/rfc/rfc3626.txt

[11] C. Dearlove, T. Clausen, and P. Jacquet, “The Optimized Link State Routing Protocol version 2,” IETF Draft RFC draft-ietf-manet-olsrv2-10, 2009. [Online]. Available: http://tools.ietf.org/id/draft-ietf-manet-olsrv2-19.txt

[12] E. W. Frew and T. X. Brown, “Networking Issues for Small Unmanned Aircraft Systems,” *Journal of Intelligent and Robotic Systems*, vol. 54, no. 1-3, p. 21, 2008. [Online]. Available: http://dx.doi.org/10.1007/s10846-008-9253-2

[13] Y. Guo, X. Li, H. Yousefi’zadeh, and H. Jafarkhani, “UAV-aided cross-layer routing for MANETs,” in *Wireless Communications and Networking Conference (WCNC), 2012 IEEE*, April 2012, pp. 2928–2933.

[14] A. Alshbatat and L. Dong, “Cross layer design for mobile Ad-Hoc Unmanned Aerial Vehicle communication networks,” in *Networking, Sensing and Control (ICNSC), 2010 International Conference on*, April 2010, pp. 331–336.

[15] M. Benzaid, P. Minet, and K. Al Agha, “Integrating fast mobility in the OLSR routing protocol,” in *Mobile and Wireless Communications Network, 2002. 4th International Workshop on*, 2002, pp. 217–221.

[16] M. Benzaid, P. Minet, and K. Al-Agha, “Analysis and simulation of fast-OLSR,” in *Vehicular Technology Conference, 2003. VTC 2003-Spring. The 57th IEEE Semiannual*, vol. 3, April 2003, pp. 1788–1792 vol.3.

[17] S. Rosati, K. Krużelecki, L. Traynard, and B. Rimoldi, “Speed-Aware Routing for UAV Ad-Hoc Networks,” in *4th International IEEE Workshop on Wireless Networking & Control for Unmanned Autonomous Vehicles: Architectures, Protocols and Applications*, 2013.

[18] Sensefly eBee. [Online]. Available: http://www.sensefly.com/drones/ebee.html

[19] SMAVNET II website. [Online]. Available: http://smavnet.epfl.ch

[20] olsrd Link Quality Extensions. [Online]. Available: http://www.olsr.org/docs/README-Link-Quality.html

[21] D. S. J. De Couto, D. Aguayo, J. Bicket, and R. Morris, “A High-throughput Path Metric for Multi-hop Wireless Routing,” in *Proceedings of the 9th Annual International Conference on Mobile Computing and Networking*, ser. MobiCom ’03. New York, NY, USA: ACM, 2003, pp. 134–146. [Online]. Available: http://doi.acm.org/10.1145/938985.939000

[22] Optimized Link State Routing Deamon. [Online]. Available: http://www.olsr.org/

[23] Microhard Systems Inc. Spread Spectrum Wireless Modem. [Online]. Available: http://www.microhardcorp.com/n2420.php

[24] Gumstix Web Page: Overo Tide COM Product overview. [Online]. Available: https://www.gumstix.com/store/app.php/products/257/

[25] Extendable Mobile Ad-hoc Network Emulator (EMANE). [Online]. Available: http://cs.itd.nrl.navy.mil/work/emane/

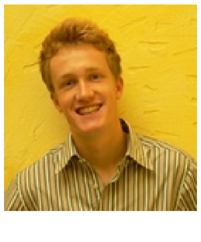
[26] IEEE P802.11 Wireless LANs, “TGn Channel Models,” IEEE Std 802.11 11-03/940r4, Tech. Rep., 2004.

[27] E. Akim and D. Tuchin, “GPS errors statistical analysis for ground receiver measurements,” *Keldysh Institute of Applied Mathematics, Russia Academy of Sciences*, 2002.

Stefano Rosati (S’07-M’11) is a Post-Doc Research Engineer at the École Polytechnique Fédérale de Lausanne (EPFL), Switzerland. He received the Laurea degree (summa cum laude) and Ph.D. degree in Telecommunications Engineering from the University of Bologna, Italy, in 2007 and 2011, respectively. From 2007 to 2011, he was with the Advanced Research Center for Electronic Systems (ARCES) of the University of Bologna. In 2010, he was an Intern Engineer at Corporate R&D Department of Qualcomm Inc. (San Diego, CA). In 2011 he joined the Information Processing Group (IPG) and the Mobile Communications Laboratory (LCM) at EPFL.

His interests are in various aspects of digital communications, in particular next-generation cellular communication systems, and both terrestrial and satellite broadcast networks. His research activities are also focused on flying ad-hoc networks and self-organizing networks of Unmanned Aerial Vehicles (UAVs). He has authored several scientific papers and internationals patents regarding these topics.

Karol Krużelecki (M’09) is a research engineer working at EPFL, Lausanne, Switzerland. He received his Msc. Eng. degree in Computer Science and in Computational Mechanics from Cracow University of Technology, Poland, in September and November 2008 respectively. Between 2008 and 2011 he was working at The European Organization for Nuclear Research (CERN) on the development of automatic build and test system used to improve the software quality and reliability for the Large Hadron Collider beauty experiment (LHCb). In 2012 he joined the Information Processing Group (IPG) and the Mobile Communications Laboratory (LCM) of the EPFL.



Grégoire Heitz is an engineer working at the Laboratory of Intelligent System at EPFL since September 2012. He obtained a M.Sc in Electronic from the ENSCPE of Lyon (France) in 2012. He was working in senseFly SA for his master thesis, working on the development of a new flying robot in 2011. His research interest lies in embedded systems development.

Dario Floreano (SM'06) received the M.A. and Ph.D. degrees from the University of Trieste, Trieste, Italy, in 1988 and 1995, respectively, and the M.S. degree from the University of Stirling, Stirling, Scotland, in 1991. He is Full Professor at the École Polytechnique Fédérale de Lausanne, Lausanne, Switzerland, where he is Director of the Laboratory of Intelligent Systems and Director of the Swiss National Center of Competence in Robotics. His research interests are at the convergence of biology, artificial intelligence, and robotics. He authored more than 300 peer-reviewed articles and 3 books on the topics of evolutionary robotics, bio-inspired artificial intelligence, and biomimetic flying robots, and spun two companies off

Bixio Rimoldi (S'83-M'85-SM'92-F'00) received his Diploma and his Doctorate from the Electrical Engineering department of the Eidgenoössische Technische Hochschule in Zurich (ETHZ). During 1988-1989 he held visiting positions at the University of Notre Dame and Stanford. In 1989 he joined the faculty of Electrical Engineering at Washington University, St. Louis, and since 1997 he is a Full Professor at the École Polytechnique Fédérale de Lausanne (EPFL) and director of the Mobile Communications Lab. He has spent sabbatical leaves at MIT (2006) and at the University of California, Berkeley (2003-2004).

In 1993 he received a US National Science Foundation Young Investigator Award. In 2000 he was elected to the grade of Fellow of the IEEE. During the period 2002-2009 he has been on the Board of Governors of the IEEE Information Theory Society where he served in several offices including President. He was co-chairman with Bruce Hajek of the 1995 IEEE Information Theory Workshop on Information Theory, Multiple Access, And Queueing (St Louis, MO), and co-chairman with Jim Massey of the 2002 IEEE International Symposium in Information Theory (Lausanne, Switzerland). He was a member of the editorial board of "Foundations and Trends on Communications and Information Theory," and was an editor of the European Transactions on Telecommunications. During 2005 and 2006 he was the director of EPFL's undergraduate program in Communication Systems.

His interests are in various aspects of digital communications, in particular information theory, and software-defined radio.