Speed-Aware Routing for UAV Ad-Hoc Networks

Stefano Rosati, Karol Krużelecki, Louis Traynard, and Bixio Rimoldi.
Mobile Communications Laboratory (LCM), Information Processing Group (IPG),
École Polytechnique Fédérale de Lausanne (EPFL), Lausanne, Switzerland.
Email: {stefano.rosati, karol.kruzelecki, louis.traynard, bixio.rimoldi}@epfl.ch

***Abstract*—In this paper we examine mobile ad-hoc networks (MANET) composed by unmanned aerial vehicles (UAVs). Due to the high-mobility of the nodes, these networks are very dynamic and the existing routing protocols partly fail to provide a reliable communication. We present Predictive-OLSR an extension to the Optimized Link-State Routing (OLSR) protocol: it enables efficient routing in very dynamic conditions. The key idea is to exploit GPS information to aid the routing protocol. Predictive-OLSR weights the expected transmission count (ETX) metric, taking into account the relative speed between the nodes. We provide numerical results obtained by a MAC-layer emulator that integrates a flight simulator to reproduce realistic flight conditions. These numerical results show that Predictive-OLSR significantly outperforms OLSR and BABEL, providing a reliable communication even in very dynamic conditions.**

I. Introduction

Unmanned aerial vehicle (UAV) networks are emerging as a valuable and suitable platform for many civilian and military applications. UAV networks can be used to connect users on the ground, to collect data from sensors, to provide a fast-deployable Wi-Fi coverage in remote areas that are hardly accessible otherwise (e.g., high mountain areas).

In order to carry out challenging tasks, UAVs must be able to communicate reliably. We show in this paper that due to the high mobility of the nodes, sometimes the existing networks routing algorithms, which have been designed for mobile ad-hoc networks (MANETs), such as BABEL [1] or the Optimized Link-State Routing (OLSR) protocol [2], [3], fail to provide a reliable communication.

In this paper, we present an extension to OLSR, which provides reliable communication even in case of very dynamic UAV networks. The key idea is to exploit the GPS information. In particular, we weight the expected transmission count (ETX) metric by a factor that takes into account the relative speed between the nodes.

We consider an ad-hoc IEEE 802.11n network of embedded mini computers, such as the ARM-based computers produced by Gumstix Inc. [4], mounted on eBee drones [5] that are produced by SenseFly. We test the performance of the novel algorithm by MAC-layer emulation by using the extendable mobile ad-hoc network emulator (EMANE) [6] combined with eMotion 2.0 flight simulator from SenseFly. The numerical results show that Predictive-OLSR outperforms OLSR and BABEL and provides a reliable communication.

II. UAV platform

The platform is based on eBee drones [5] developed by SenseFly and on mini ARM-based computers by Gumstix Inc. [4]. The drones are fixed-wing aircrafts with an electric motor and integrated autopilot capable of flying with winds of up to 12 m/s, at a cruising speed of up to 57 km/h, with an autonomy of up to 45 minutes. In case of emergency, they can be remotely controlled up to a distance of 3 km via a Microhard Systems Nano n2420 [7] link connection. Within this distance, if necessary, the flight mission can be modified on the fly. The autopilot has access to an inertial measurement unit, a barometer, a pitot-tube for airspeed, an optical-flow sensor and GPS receiver.

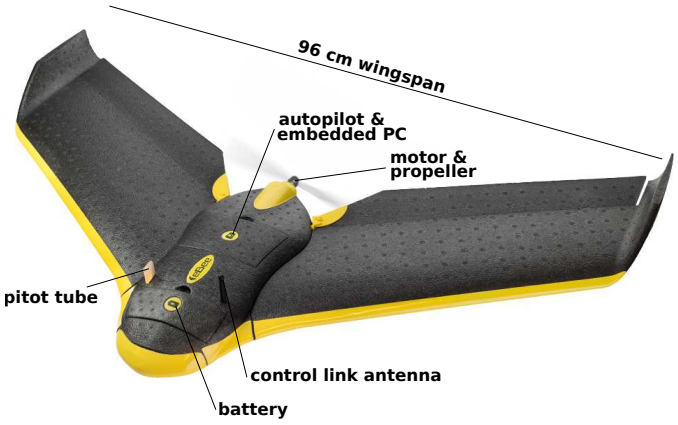

Fig. 1. SenseFly eBee drone.

Each eBee also carries a Gumstix Overo Tide [4] computer with a custom embedded Linux distribution and a standard USB WiFi (802.11n) card. We use this embedded computer to establish the wireless network. A serial connection between the auto-pilot module and the embedded computer allows us to access the sensors attached to the autopilot (including the GPS reading) and to give commands to the autopilot, e.g., modify the aircraft mission according to routing needs. Thanks to its small dimensions and weight (under 630 g), flying eBees are not considered a threat. In many countries (e.g., Switzerland) they can be used without specific authorization.

III. Enhanced Routing for UAV Ad-Hoc Networks

OLSR is a proactive routing algorithm based on the link-state routing protocol. It is currently the most employed routing algorithm for ad-hoc networks. Initially OLSR selected the route with the least number of hops [2]. As it is well known, the hop count metric is not suitable for wireless links. However, using the OLSR *link-quality extension* [8], OLSR can take into account the quality of the wireless links, using the ETX metric.

BABEL, presented more recently [1], is also a routing algorithm designed for ad-hoc networks. Like OLSR, it is a proactive routing algorithm, but it is based on distance-vector routing protocol and adopts EIGRPs loop avoidance techniques. Like OLSR, Babel also uses the ETX metric.

A. Expected Transmission Count

ETX measures the quality of the wireless link between the nodes i and j. It was introduced in [9] and it is defined as

$$\mathrm{ETX}^{i,j} = \frac{1}{r^{\mathrm{f}} r^{\mathrm{r}}}\,, \tag{1}$$

where r^{f} is the forward receiving ratio that is the probability that a packet successfully arrives at the recipient; and r^{r} is the reverse receiving ratio that is the probability that the ACK packet is successfully received. In other words ETX estimates the expected transmissions (including re-transmissions) of a packet necessary for it to be received without error at its destination. Then the ETX of a route $\mathcal{R}$ is defined as the sum of the ETX metrics of the links composing the route

$$\mathrm{ETX}^{\mathcal{R}} = \sum_{(i,j)\in\mathcal{R}} \mathrm{ETX}^{i,j}\,. \tag{2}$$

The receiving ratios r^{f}, r^{r} are measured using as a link probe a packet called *Hello packet*[1]. The *Hello Interval* is a parameter that indicates how frequently the Hello packets are broadcast. In OLSR, the receiving ratio r^{f} is computed using a exponential moving average, as

$$\begin{cases} r_l^{\mathrm{f}} = \alpha h_l + (1-\alpha) r_{l-1}^{\mathrm{f}} \\ r_0^{\mathrm{f}} = 0 \end{cases}, \quad 0 \le \alpha \le 1\,, \tag{3}$$

where

$$h_l = \begin{cases} 1 & \text{if } l\text{-th Hello packet received} \\ 0 & \text{Otherwise} \end{cases} \tag{4}$$

and α is a OLSR parameter named *Link-Quality Aging*. It is worthwhile to note that α sets the trade-off between accuracy and responsiveness of link-quality estimation. On one hand, with a higher α the receiving ratios will be averaged for yielding a more stable and reliable estimation; on the other hand, with a lower α the receiving ratios will be faster for tracking the current link-quality.

B. Speed-Weighted ETX

ETX is an efficient measure of the quality of a link in quasi-static wireless ad-hoc networks, but is not reactive enough to cope with very dynamic wireless ad-hoc networks, such as UAV networks. Due to the exponential moving average, which is necessary for a stable and reliable link-quality estimation, a node takes a certain amount of time before noticing that a wireless link quality has decreased. During this time it will continue to route packets on a broken link, thus yielding an interruption of the service. The key idea for overcoming this problem is to use GPS information to improve the routing. In particular, to predict how the link quality is likely to evolve, we use the relative speed between two nodes. Assuming that every node knows the position of its neighbors[2], we redefine ETX as

$$\mathrm{ETX}^{i,j} = \frac{\mathrm{e}^{v_\ell^{i,j}\beta}}{r^{\mathrm{f}} r^{\mathrm{r}}}\,, \tag{5}$$

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

[1] In OLSR, The Hello packets are dedicated OLSR control messages that are also used as link probes by the OLSR link-quality extension

[2] In the following section we discuss how to distribute this information among the nodes.

If the node 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. On the contrary, if the node i and j move away from each other, the relative speed is positive, thus the ETX will be weighted by a factor greater than 1. As a consequence, a link between two nodes that move closer will be preferred to a link between two nodes that move apart, even if they have the same values of r^{f} and r^{r}. The best value of β depends on the cruise speed of the UAVs of the link coverage extension.

C. Computation of the Speed

As discussed in the following section, in our implementation the GPS coordinates are conveyed by the Hello packets. Thus every time the algorithm computes the ETX, it has available fresh GPS information. The instantaneous relative velocity between i and j at time t_i is computed as

$$\tilde{v}_\ell^{i,j} = \frac{d_\ell^{i,j} - d_{\ell-1}^{i,j}}{t_\ell - t_{\ell-1}}\,, \tag{6}$$

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

$$v_\ell^{i,j} = \gamma \tilde{v}_\ell^{i,j} + (1-\gamma) v_{\ell-1}^{i,j}, \quad 0 \le \gamma \le 1\,, \tag{7}$$

where γ is a Predictive-OLSR parameter.

IV. Implementation Details

To implement this speed-weighed ETX, we used the already-existing, open-source and actively developed OLSR daemon called OLSRd [10]. We modified it to share position information in addition to receiving ratios in Hello packets. Thus, every node knows its neighbor's position and can compute the corresponding ETX.

A. OLSRd Link-Quality Extension

OLSRd uses link-quality sensing and ETX metrics, through the so-called Link Quality extension [8]. They replace the hysteresis mechanism of the OLSR protocol with link-quality sensing algorithms, intended to be used with ETX-based metrics. To do so, such an extension uses OLSR Hello messages to probe links quality and advertise link-specific quality information (receiving ratios), in addition to their previous role in the protocol (detecting and advertising neighbors). Likewise, the link-quality extension include link-quality information in OLSR TC messages, to distribute it 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 should use the link-quality extension.

B. Extending Link-Quality Extension

Position information can be thought as additional link-quality information, which is shared the same way as receiving ratios. The OLSRd link-quality extension mechanism was tailored for simple ETX metrics. Now, in our case, we have two new requirements: (i) store more information per link to compute the ETX; (ii) to share the position information that is not link-specific. We extended the OLSR link-quality extension mechanism to enable the implementation of more complex metrics. In particular, we modified again the Hello message to include the GPS positions that are non-link-specific information. The modified format of Hello packets is reported in Figure 2. The fields in gray are not part of the original OLSR RFC. The fields *neighbor-specific link-quality data* have been introduced by the OLSRd link-quality extension. We added a new field named *non-link-specific quality data*.

C. Obtaining GPS Position

OLSRd comes with a handy networking toolkit that we used to implement an OLSRd plug-in able to listen to GPS sentences on a given interface, parse them (with the open-source NMEA library) and update the node position, directly in OLSRd.

V. Routing Performance

We measure the evolution of the datagram loss rate (DLR) of a multi-hop route. As we use the network to transmit a continuous stream of data (e.g., high-quality video stream) in real time, our goal is to minimize the DLR during the transmission. We measure the DLR every second by sending 85 UDP datagrams having, in total, a size of 1 Mbit[3]. DLR is the ratio between the lost and the total number of datagrams. We consider two different multiple-hop scenarios: (i) involving a UAV

[3]We use *iperf* to obtain this measurements.

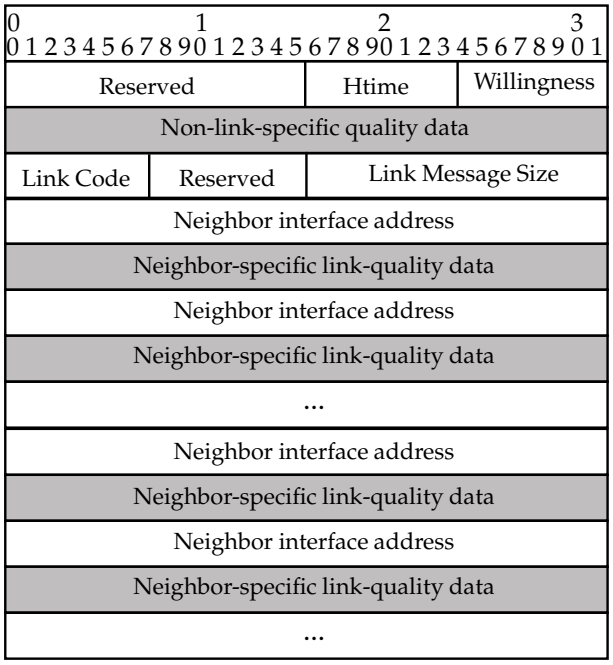

Fig. 2. Format of the modified Hello packet.

source node, two UAV relays, and a fixed destination node; (ii) involving a UAV node flying around rectangle area sized 1200x1500 meters and covered by 32 static relays. We compare Predictive-OLSR, OLSR that uses the link-quality extension, and BABEL. We set the Hello interval to 0.5 second for all the analyzed algorithms. For OLSR we set $\alpha = 0.2$, which is the best trade-off between stability and responsiveness in our scenarios. As Predictive-OLSR is inherently more responsive, we can adopt a lower α in order to improve the stability. We choose $\alpha = 0.05$. The others Predictive-OLSR parameters are set as follows: $\beta = 0.2$, $\gamma = 0.04$. All the measurements are obtained using the MAC-layer emulator presented in the following section.

A. Emulation Platform

Field experiments are expensive and require the involvement of people, transportation, and costly equipment. For this reason we developed an emulation platform that integrates all the test-bed aspects, as illustrated in Figure 3. The emulator creates a Linux Container (LXC) for each node of the network. The nodes are connected using a MAC-layer emulator called extendable mobile ad-hoc network emulator (EMANE). EMANE is an open-source framework, developed mainly by Naval Research Laboratory for real-time modeling of mobile network systems. Regarding the channel model, we

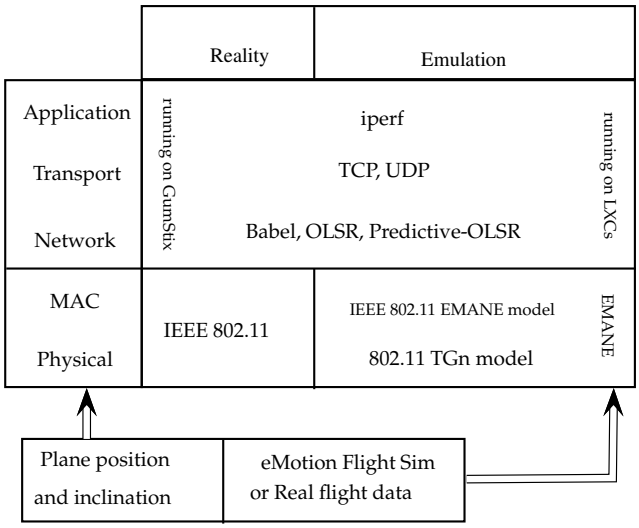

Fig. 3. Emulation platform.

consider the IEEE 802.11 TGn model defined in [11]. EMANE imports the positions, speeds, and orientations of the UAVs from log files. These log files can be obtained from real flight data logs, or by a flight simulator called eMotion, provided by SenseFly, that simulates realistic flight condition of the eBee drone. All network layers, except the MAC and the physical layer, use the actual implementations that run in the Linux machine hosting the emulation. To obtain numerical results we run emulation on Fedora 15, kernel 2.6.43.8. The tested version of OLSRd is 0.6.5.3, and the tested version of BABELd is 1.3.4.

B. 2-relay scenario

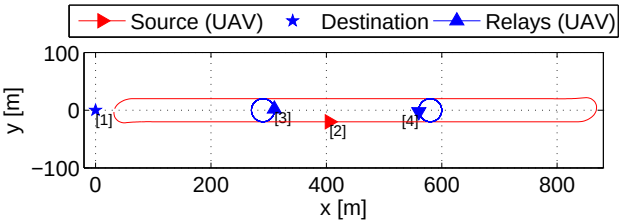

Fig. 4. 2-relay scenario.

A multiple-hop scenario, illustrated in Figure 4, consists of a mobile UAV source (node 2), two mobile UAV relays (nodes 3 and 4), and a fixed destination (node 1). Both the source and the relay nodes are embedded in eBee UAVs. The relays keep circling around the given position with the circular trajectories of 20 meter radius,

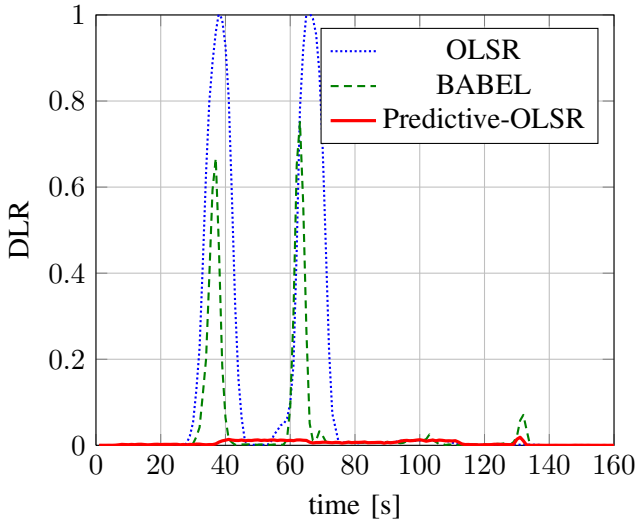

Fig. 5. Evolution of average DLR in the 2-relay scenario.

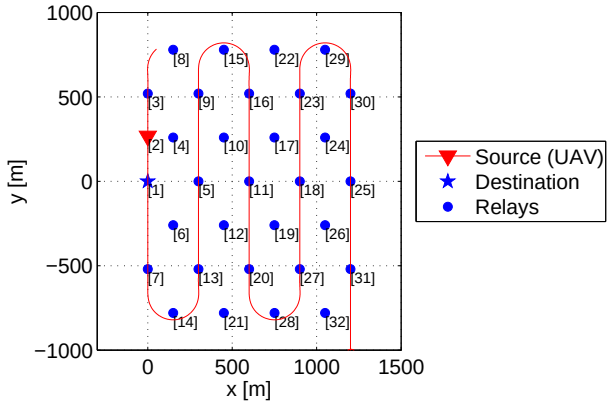

Fig. 6. Open area coverage scenario.

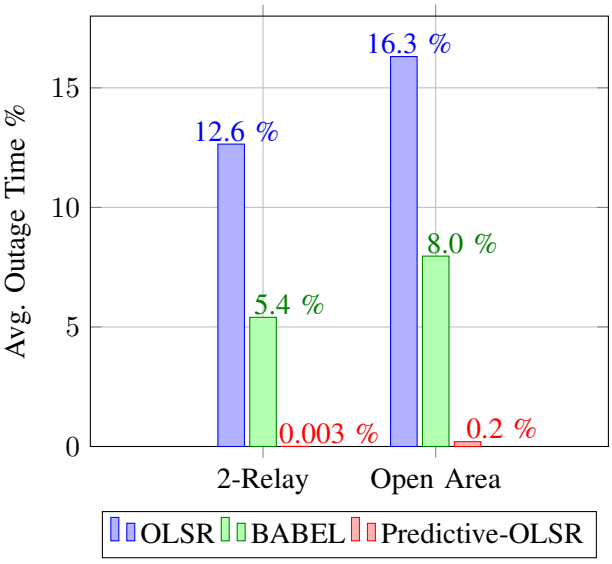

Fig. 8. Average outage time for both the analyzed scenarios.

extending the range of the network coverage for the mobile source from around 300 meters up to around 900 meters from the target. Node 2 follows a straight trajectory of around 850 meters, then it returns to the starting point. It takes 160 seconds to finish the loop. During this period, node 2 continuously transmits data to node 1. During one loop, node 2 changes its routing to node 1 from a direct connection to a 2-hop and then to a 3-hop route, and back. Therefore the network topology will change 4 times during the loop.

For each routing algorithm, we run 200 loops in order to obtain an average of the results. Figure 5 shows the evolution of the average DLR during the loop. For both OLSR and BABEL, we notice the two peaks of the DLR that correspond to the moments when the routing algorithm has to switch from the direct link to a 2-hop and then to a 3-hop connection. This happens because the routing algorithm takes a certain amount of time to notice that the wireless direct-link quality is broken. So it reacts late, this translates into an interruption of the service. Whereas, Predictive-OLSR reacts promptly to the topology changes. Figure 5 shows that there are no peaks of the average DLR. It is interesting to note that BABEL outperforms OLSR. Similar results were reported [12] where BABEL and OLSR were experimentally compared.

C. Open Area Coverage

In the second scenario, consisting of one moving UAV (node 2), one fixed destination (node 1) and 30 relays (nodes 3 to 32) that are located uniformly within a rectangle area of size 1200x1500 meters. As before, we transmit the data continuously from node 2 to node 1. Node 2 scans the area by following the trajectory shown in Figure 6. It takes 870 seconds to complete the trajectory. The distances between the relays has been chosen to have a good quality direct wireless link only among the closest neighbors that are 300 meters away. For example, node 5 can communicate directly only with nodes 1, 4, 6, 10, 11, 12. It cannot reach the other nodes directly. In order to average the results, we repeat the emulation 50 times for each routing algorithm.

Figure 7 shows the evolution of the average DLR. Again, we notice that, for both OLSR and BABEL, the DLR has several peaks during the mission, which trans-

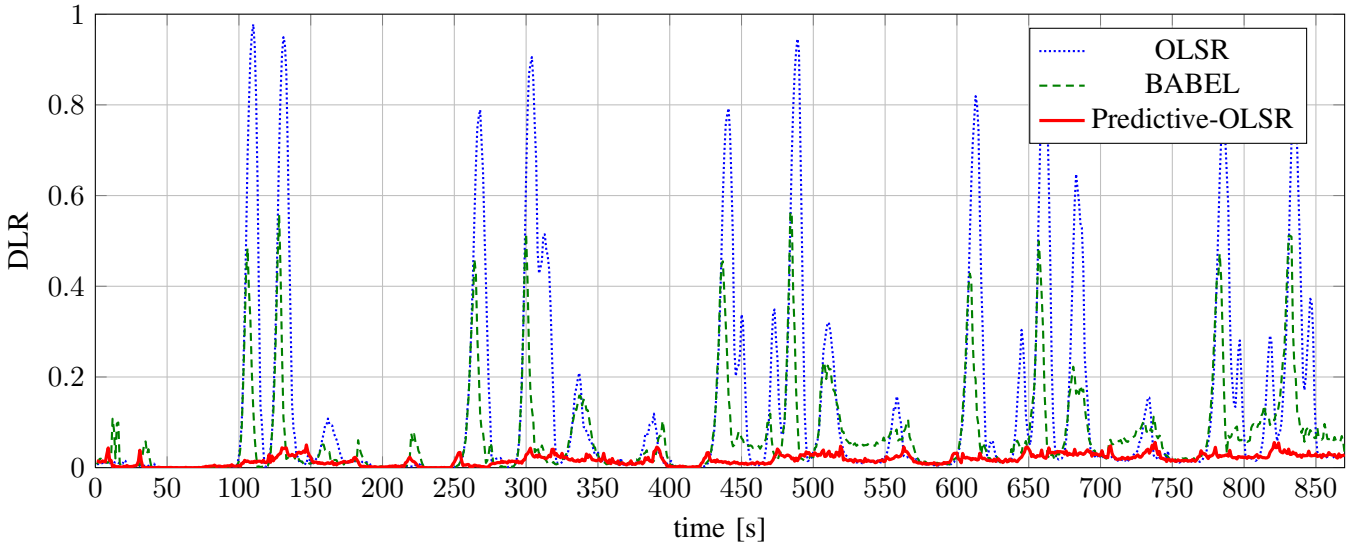

Fig. 7. Evolution of average DLR in the open area coverage scenario.

lates into a service interruption. This happens because the routing algorithm is not fast enough to reach to the topology changes. Whereas, with Predictive-OLSR the average DLR is never higher than 0.1.

Let us assume that there is an outage event during the time the DLR is greater than 0.2. We compute the outage time percentage for each run, and then we average all the runs. As shown in Figure 8, node 2 experiences an average outage time of 16.3% with OLSR: and 8% with BABEL; where as with Predictive-OLSR the outage time is 0.2%.

VI. Conclusions

In this paper, we have presented an extension, named Predictive-OLSR, to the OLSR routing protocol, that enables efficient routing in very dynamic ad-hoc networks composed of UAVs. This extension exploits GPS information. The numerical results, obtained by MAC-layer emulation, show that Predictive-OLSR succeeds in providing a reliable multi-hop communication, even in such a dynamic ad-hoc network, whereas, other state-of-the-art routing protocols, such as BABEL and OLSR, mostly fail.

Acknowledgments

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

References

[1] J. Chroboczek, "The Babel routing protocol," RFC 6126, Apr. 2011. [Online]. Available: http://www.rfc-editor.org/rfc/rfc6126.txt

[2] 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

[3] 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

[4] Gumstix web page: Overo Tide COM product overview. [Online]. Available: https://www.gumstix.com/store/product_info.php?products_id=257

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

[6] Extendable mobile ad-hoc network emulator (EMANE). [Online]. Available: http://cs.itd.nrl.navy.mil/work/emane/

[7] Microhard systems inc. spread spectrum wireless modem. [Online]. Available: http://www.microhardcorp.com/n2420.php

[8] olsrd link quality extensions. [Online]. Available: http://www.olsr.org/docs/README-Link-Quality.html

[9] D. S. J. De Couto, D. Aguayo, J. Bicket, and R. Morris, "A high-throughput path metric for multi-hop wireless routing," 2003.

[10] Optimized link state routing deamon. [Online]. Available: http://www.olsr.org/

[11] IEEE P802.11 Wireless LANs, "TGn channel models," IEEE Std 802.11 11-03/940r4, Tech. Rep., 2004.

[12] D. Murray, M. Dixon, and T. Koziniec, "An experimental comparison of routing protocols in multi hop ad hoc networks," in *Telecommunication Networks and Applications Conference (ATNAC), 2010 Australasian*, 2010, pp. 159–164.