This paper is a preprint of the published work and restricted to personal use only. The copyright and all rights therein are retained by the author(s) and/or other copyright holders. The definite version of this work is published as: Mahdi Asadpour, Mario Burger, Fabian Schuiki, and Karin Anna Hummel. Needle in a Haystack: Limiting the Search Space in Mission-aware Packet Forwarding for Drones. In Proceedings of the 1st International Workshop on Experiences with the Design and Implementation of Smart Objects (SmartObjects '15). ACM, 31-36. doi:10.1145/2797044.2797057

Needle in a Haystack: Limiting the Search Space in Mission-aware Packet Forwarding for Drones

Mahdi Asadpour, Mario Burger, Fabian Schuiki, Karin Anna Hummel
ETH Zurich
Zurich, Switzerland
{amahdi, marioburger, fschuiki, khummel}@ethz.ch

ABSTRACT

Micro aerial vehicles may build a collective of smart sensor objects cooperating in civilian missions such as search and rescue, farmland monitoring, or surveillance. Wireless connectivity is a prerequisite for transferring images and other sensor data to the ground. Though aerial vehicles may set up a multi-hop wireless network on their own, vehicle movement causes frequent changes of the wireless signal quality and intermittent connectivity, which poses challenges to end-to-end data delivery and renders traditional routing approaches impractical. We address this problem by including delay-tolerant packet forwarding. Further, we make forwarding mission-aware, i.e., aware of future positions and connection opportunities derived from the waypoints of the MAVs' mission. The resulting path options for packet forwarding open a vast search space. We present a solution to find a path efficiently based on the A^* search algorithm. We study the performance of our mission-aware algorithm compared to a delay-tolerant variant of geographic routing in simulation and in a testbed of quadcopters with IEEE 802.11n aerial links. Our first results reveal that for simple scenarios, the benefit of mission-aware forwarding is limited, yet, in more sophisticated scenarios, mission-aware forwarding can alleviate inefficient forwarding and improve performance.

Categories and Subject Descriptors

C.2.1 [**Computer-Communication Networks**]: Network Architecture and Design—*Wireless communication*

Keywords

Micro Aerial Vehicle Networks; Mission-aware Packet Forwarding; IEEE 802.11; Measurements

1. INTRODUCTION

Drones, also known as unmanned aerial vehicles (UAVs) or micro aerial vehicles (MAVs), are increasingly used in civilian missions such as surveillance, search and rescue, entertainment, 3D mapping, etc. The major strength of MAVs is their ability to provide images, videos, and other sensor information from an aerial perspective. Furthermore, MAVs are small flying robots that are aware and in control of their movement observed by GPS (Global Positioning System) and IMU (Inertial Measurement Unit) on-board modules. To cover larger areas, MAVs may cooperate and jointly set up a wireless ad-hoc multi-hop *aerial network of things*.

Networking and in particular packet forwarding is challenged in aerial networks as MAV mobility and light-weight communication solutions such as small antennas cause limited coverage, unstable link conditions, and intermittent connectivity. Classical routing protocols cannot cope well with this setting as they require end-to-end connectivity and some degree of link stability to converge [2, 7]. Delay-tolerant networking (DTN) concepts supporting store-carry-forward transmission master intermittent connectivity well. In aerial networks, DTN can be further supported by dedicated ferry MAVs, which are sent to carry data physically into transmission range of other MAVs. Additionally, mission-defined trajectories of MAVs are known to a large extent and can be leveraged for smart packet forwarding.

We assume in this work that MAVs move according to mission-driven, pre-defined waypoints. In a brute-force approach the MAV connectivity graph can be constructed for all future points in time and traversed. Assuming one hop at a point in time, the number of path options in this space is in the range of up to N^T, where N is the number of nodes and T is the number of points in time. As exhaustive search is computationally expensive, cf. also the arguments presented in [10], there is a need for providing an efficient search approach in order to make mission-aware packet forwarding a reality. In this paper, we make the following contributions:

- We introduce an MAV network model considering the positions of MAVs and the time as additional dimension. We model link cost basically in terms of packet forwarding delay, which consists of the time the data are carried and the transmission time of the data. The transmission delay is calculated for Wi-Fi 802.11n links based on an empirically derived throughput function of the geographical link distance (Section 3).
- We design and implement an algorithm based on the A^* search algorithm, which finds existing delay-tolerant pathways for a packet and selects the path with expected minimum transfer time. The algorithm operates with exhaustive search based on real costs combined with a heuristic that guides the search towards a solution (Section 4).

SmartObjects'15, September 7, 2015, Paris, France.

DOI: http://dx.doi.org/10.1145/2797044.2797057.

- We analyze the A^* mission-aware forwarding scheme in simulation and in a field test and compare the results to geographic routing that has been extended to tolerate disconnections (cf. [2, 9]). We discuss the potential and limitations of mission-aware forwarding for different scenarios (Sections 5 and 6).

2. RELATED WORK

Position-based, geographic, or geometric routing [8] is a well-suited approach for packet forwarding in mobile networks with known node location. The routing aim is to bring packets geographically closer to the destination by selecting next hops accordingly. Pure geographic routing is, however, not adequate for networks that are subject to intermittent connectivity. A known approach to target intermittent connectivity is delay-tolerant networking. Extensions to incorporate DTN in location-aware aerial networks have been proposed in [9,11]. Simulation results indicate that the approach is promising.

Beyond the current location, motion and navigation information can be leveraged. GeoDTN+Nav [5] developed for ground vehicular ad-hoc networks makes use of navigation information. An efficient store-carry-forward concept is introduced that can predict future disconnections and connectivity opportunities. For MAV networks, an ad-hoc trajectory-aware routing protocol is presented in [10]. In this work, the full trajectories of all MAVs at each point in time are known a priori. Simulation results show the potential of trajectory-awareness by improved throughput in comparison to two ad-hoc routing algorithms, AODV and LAR. Yet, the algorithm requires large storage for maintaining the path information of all the MAVs as well as high computational cost for calculating all routes. This result confirms that, on the one hand, awareness about the waypoints of MAVs (as defined by the mission) improves packet forwarding decisions. On the other hand, search algorithms with limited complexity are required to make mission-aware forwarding practical.

3. MAV NETWORK MODEL

The MAV network is modeled as a temporal graph over mission time T_m. At each point in time T_i, a snapshot of the network graph can be drawn with vertices representing MAVs (and ground devices) and edges corresponding to the wireless links of nodes that are in transmission range of one another. The throughput and connectivity model is based on a disk/sphere range model for wireless transmission in free space. As the network topology changes over time and connectivity is intermittent, the network model includes links that reflect carrying (storing) of the data by one MAV.

3.1 Link Types

The MAV network graph includes two link types, traditional wireless *transmission links*, and *carry links*.

- **Transmission link:** When two nodes k and l are in communication range of each other at time T_i, a transmission link exists. The link quality is expressed by the weight w_{k-l}. In our model, we assume transmission links to be symmetric, i.e., $w_{k-l} = w_{l-k}$. The weight w is calculated as follows:

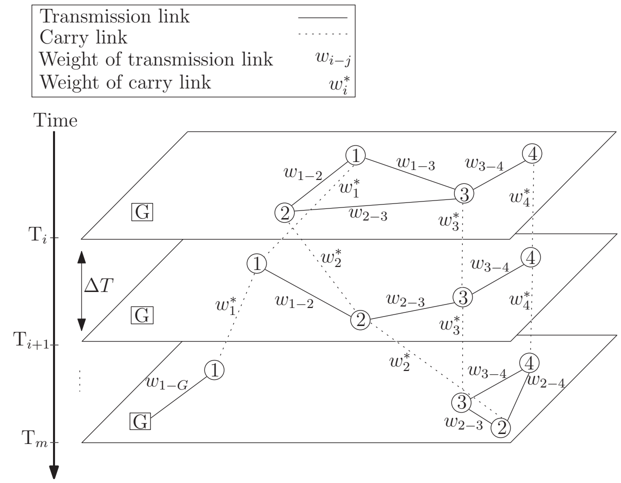

Figure 1: Sample MAV graph representation in discrete time planes over mission time T_m.

$$w = T_{tx} = \frac{M}{s(d)}, \tag{1}$$

$$s(d) = 10^6 \times (-7.952 \times \log_2(d) + 66.94), \tag{2}$$

where M is the size of a data packet and $s(d)$ is the empirically derived throughput as a function of the geometric distance of two MAVs (cf. [3], coefficients adapted to the testbed used in this paper).

- **Carry link:** Links that reflect storing and carrying of data by MAVs are named carry links. For two given points in time T_i, T_j ($T_i < T_j$), we assign a carry link a weight w_k^*, where k is the carrying MAV. The weight w^* is calculated as follows:

$$w^* = T_j - T_i. \tag{3}$$

3.2 Time-discrete Representation

To ease graph traversal, we use a discrete time approach with time step ΔT. The temporal MAV network model is sliced into spaces at points in time $T_i, T_{i+1} = T_i + \Delta T$. ΔT is chosen to cover one packet transmission. Figure 1 visualizes a sample MAV graph over discrete time; MAV locations are represented in two dimensions. An example transmission link exists between MAV 1 and 2 with a weight of w_{1-2} at time T_i. A carry link, here, a movement of MAV_2 can be observed between time steps T_i and T_{i+1}, the corresponding weight is denoted by w_2^*.

3.3 Packet Forwarding Cost

Given the discrete time MAV graph model with time step ΔT, we calculate the expected packet forwarding cost for all possible path options. For a given observation or mission time T_m, all MAV waypoints are known and a connectivity matrix (adjacency matrix of transmission links) at each point in time T_i is created with weights $w = T_{tx}$ ($T_{tx} \leq \Delta T$).

Algorithm 1 takes as input the connectivity matrices of all time steps T_i and calculates a tree of all path options (with tree level i corresponding to T_i). Each path option is

Algorithm 1: Construction of packet path space.

Data: connectivity matrix at all T_i
Result: tree of of all route options with cost T
$i = 1$
set for all nodes: $T = 0, T_d = 0$
while $(T_i \leq T_m)$ **do**
  <construction of i-th tree level>
  <calculate costs for all routes on i-th level>
  $T_d = (i-1) \times \Delta T$
  forall the *nodes* **do**
    if $i \neq 1$ **then**
      set T to cost of node on level $i-1$
      $T = T + \Delta T$
    end
    forall the *neighboring nodes* **do**
      calculate T_{tx}
      $T = T + T_{tx}$
      store T, T_d
    end
  end
  $i = i + 1$
end

assigned a cost T calculated as the sum of the route delay T_d and a *transmission cost* T_{tx} in terms of the (multi-hop) transmission time of a packet. At each step from T_i to T_{i+1}, ΔT is added to T_d.

We make use of the transmission time to differentiate between routes with small and large transmission overhead (few or many hops), favoring those with small transmission overhead. Yet, the influence of transmission time on the cost T should be small. This can be achieved with a feasible setting of ΔT, e.g., in the range of seconds when using Wi-Fi as wireless transmission technology.

A brute-force search for a given source node k and a destination node l can now be performed on the path tree with the aim to return the best path option. Figure 2 visualizes two different path options for sending a packet from a source to a destination node in a sample scenario of one hovering MAV_2 (source), two ferrying MAVs (MAV_1 and MAV_3), and a ground station (destination). Given the two route options, route 2 is preferable as it results in the minimum delay of $T_d = 50\,\text{s}$ and a cost of $T = 50\,\text{s} + 2T_{tx}$. The open question is how to find a low-cost path avoiding exhaustive search.

4. MISSION-AWARE FORWARDING

Mission-aware packet forwarding relies on the current time as well as on the positions and future waypoints of all MAVs. Further, the forwarding algorithm needs to efficiently traverse the temporal MAV network graph.

4.1 Mission-awareness

Position and motion information of MAVs is provided by on-board GPS and IMU modules, the time is provided by the on-board embedded system (synchronized on all MAVs). The future waypoints of MAVs are disseminated between the MAVs and each MAV constructs a local representation of the temporal graph (cf. Section 3). To assure reliable mission dissemination, we leverage an out-of-band network channel dedicated to transmission of mission and telemetry data (cf. Section 5).

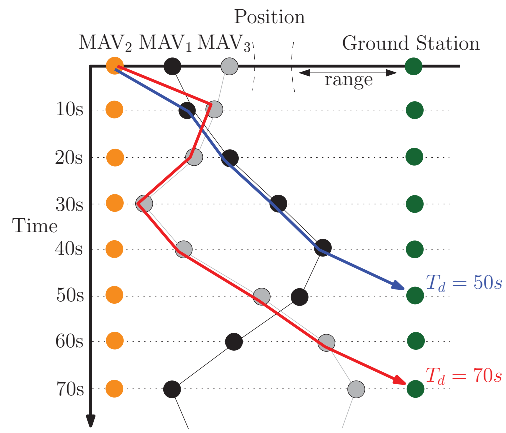

Figure 2: Example packet forwarding paths in an MAV network (discrete time representation). MAV_2 sends data to the ground station using MAV_1 and MAV_3 as data ferries.

4.2 Graph Traversal by A^*

To efficiently traverse the MAV graph, we propose an A^*-based algorithm. The original A^* algorithm uses a best-first search to find a least-cost path from a source to a destination [6]. The algorithm traverses the graph based on the order defined by a cost function $f()$ that combines the cost traveled in the graph from the source to node k ($g(k)$) with a heuristic estimating the cost to reach the destination ($h(k)$), $f(k) = g(k) + h(k)$.

In the MAV graph, $g(k)$ corresponds to the cost T from the source node to the current node k (precisely, the representation of node k in one time space/plane). The costs are calculated as detailed in Algorithm 1. Our heuristic estimate $h(k)$ is calculated as the transmission delay that would occur when transmitting the packet via *hypothetical relay MAVs* that are placed in a straight line between node k and the destination. (We use an equal distance of 10 m between the MAV relays.)

For each packet, the following steps are executed:

1. The MAV graph (cf. Section 3) is constructed for the current point in time T_i based on the current locations of MAVs, i.e., the links are assigned weights corresponding to the expected T_{tx} of the links. The source node is selected as the current node k.

2. Each neighbor n of the current node k at the current point in time is traversed. All neighbors are then ordered along their cost values defined by $f(n)$, consisting of the actual costs T and the heuristic estimate.

3. If either the execution time exceeds a threshold or the number of nodes in the graph reaches a maximum, the algorithm stops. Otherwise, the algorithm selects the node n with the lowest cost value $f(n)$ as the current node k.

4. If node k is the destination, the algorithm returns the path together with the cost value. Else, the algorithm expands the graph by one time step, setting the new point in time to T_{i+1} and continues with step 2.

The worst case performance of our algorithm is expressed by $\mathcal{O}(|V|^2 \times m)$ (V is the set of the nodes in the graph at

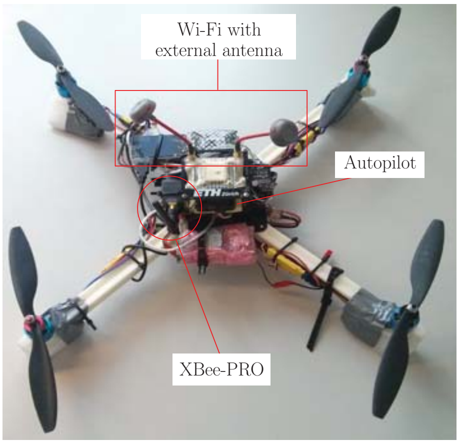

Figure 3: Quadcopter platform with on-board wireless package and two external circular antennas.

a point in time, m is the number of points in time). By configuring the execution time threshold, the runtime of the algorithm can be limited.

5. EVALUATION SETUP

We intend to provide first performance insights of our mission-aware A^*-based algorithm. Thus, we compare it with geographic routing that is extended to support intermittent connectivity. Geographic forwarding is a greedy approach based on sending a packet simply to the neighboring node that is geographically closest to the destination.

5.1 Implementation

The mission-aware A^*-based algorithm as well as the geographic forwarding algorithm are implemented in C++. To execute the algorithms in simulation and on the embedded system, we proceed as follows:

Simulation: We implement a discrete event simulator to run, study, and fine-tune the mission-aware algorithm. The simulator uses a communication and mobility model simplified for this purpose. The communication range is set based on a disk model with adjustable cut-off range (here 200 m). Communication is ideal when in range (no packet loss, no interference). To be realistic, the transmission time is calculated based on the experimentally derived throughput function $s(d)$ (cf. Section 3). The MAV mobility model describes linear MAV movement along straight lines at a speed similar to real settings (4.5 m/s); turns are immediate.

Embedded system: The algorithm source code is optimized for the embedded system (cf. Section 5.3), in particular to save CPU-time and memory.

5.2 Metrics

The following metrics are used for evaluation:

- **Delay:** the time difference between message creation time and the time a message is received at the destination calculated as the sum of transmission delays occurred on each hop and the time a message remains in the queue before transmission.
- **Hop count:** the number of hops a message passes until it reaches the destination. This metric allows to discuss the efficiency of a forwarding algorithm.
- **Processing overhead:** the execution time needed by an algorithm to take a forwarding decision for a packet.

5.3 Flying Platform

We use a quadcopter platform termed "Arducopter" [1] (Figure 3) equipped with a Wi-Fi communication package. The main electronic system of Arducopter is an Arduino-based autopilot, which integrates a GPS unit, IMU, pressure sensors, etc. The copter's typical cruise speed is 4.5 m/s. The safe flying altitude is up to 100 m and flight endurance is about $12 - 15$ minutes. The autopilot enables the copter to take off and land autonomously and to navigate through GPS waypoints.

5.4 Wireless Network

To separate control and data traffic, a hybrid wireless network consisting of two wireless technologies is used:

- **XBee-PRO**: long range (about 1.5 km), low throughput (less than 80 kb/s) communication channel used for light-weight data such as telemetry and control commands. It operates in the 2.4 GHz frequency band and connects every MAV to the ground station. All MAVs overhear transmissions.
- **Wi-Fi IEEE 802.11n**: shorter range (about $200 - 300$ m), high-throughput (about 100 Mb/s) communication technology for large-sized data. To avoid interference with XBee-PRO, the 5 GHz frequency band is used to connect MAVs in ad-hoc mode. We use SparkLAN WUBR-507N USB dongles with Ralink 3572 chipset and mount two external circular antennas for improved link performance [4].

Status information including GPS (latitude, longitude, altitude), orientation, and speed is broadcasted periodically with a tunable frequency (here, every two seconds) via XBee-PRO. This way, the ground station and other MAVs receive status information and become aware of the network topology. The mission-related information (waypoints) of all MAVs are exchanged at the beginning of a test.

6. EVALUATION RESULTS

We compare the performance of our mission-aware A^*-based algorithm with geographic forwarding. The evaluation scenarios are inspired by a search and rescue mission, in which MAVs gather data such as images and send the data to the ground. We assume a sparse placement of MAVs and intermittent connectivity. Ferry MAVs are employed to carry and transmit data to the ground station.

The first part of the investigation is based on *simulation* with the aim to analyze the principle behavior of the A^*-based algorithm exposed to scenarios with different numbers of MAVs (up to 14 nodes). Then, we study the algorithms in a testbed with three quadcopters (four nodes) and present the measurement results of a *field experiment*. The field results give first insights into the performance of our A^*-based algorithm when implemented on the embedded system and exposed to real world conditions.

6.1 Simulation

Figure 4 depicts the setting of the simulation scenarios. Up to nine hovering MAVs are placed in an area of 800 m×800 m outside of the communication range of the ground station.

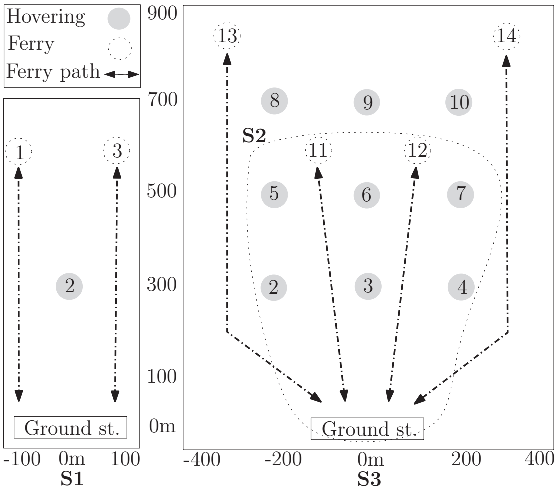

Figure 4: Simulation scenarios, S1 (four nodes), S2 (nine nodes), and S3 (14 nodes).

To reach the ground station, MAVs make use of up to four ferry nodes. The simulation time is 5 minutes; each hovering MAV generates 5 messages per second. Three scenarios – S1 (four nodes), S2 (nine nodes), and S3 (14 nodes) – are defined, see Figure 4. In all scenarios, the respective symmetric ferries fly always in opposite direction.

Delay. Figure 5 (top) visualizes the delay for the simulated scenarios S1, S2, and S3. In the simple scenarios, the A^*-based algorithm and geographic forwarding achieve a similar median delay in the range of 60 s (scenario S1) and 90 s (scenario S2). In the most sophisticated scenario S3 (larger network graph, many forwarding options), we observe a median delay of 137 s for A^*-based forwarding and an improvement of about 22 s when compared to geographic forwarding. Further, the variability of the delay increases in both algorithms with increased number of nodes as an effect of different (intermittent) connectivity experienced by the MAVs.

Hop count. As shown in Figure 5 (middle), the A^*-based algorithm outperforms geographic forwarding in all scenarios. Whereas almost all messages are transferred via two hops by the A^*-based algorithm meaning that the data are transferred by exactly one ferry to the ground, geographic forwarding routes the messages along considerably more hops (median hop count is about ten hops in scenarios S2 and S3). This outcome is expected as the A^*-based algorithm leverages trajectory knowledge and can optimize transmission, while geographic forwarding only considers the current location information.

Processing overhead. As shown in Figure 5 (bottom), when running the algorithms in simulation on a standard PC the A^*-based algorithm causes substantially larger processing overhead than geographic forwarding. The execution time of geographic forwarding is always less than 0.003 s. The A^*-based algorithm shows a maximum median execution time of 2.45 s. As expected, the execution time of the A^*-based algorithm grows with growing number of nodes and path options.

Discussion. The results of the simulation study confirm that the A^*-based algorithm can improve geographic-based packet forwarding by avoiding unnecessary hops at the cost of processing overhead. The more path options are available, the more the benefits of the algorithm are observable. In particular packet forwarding delay could be only reduced in larger scenarios.

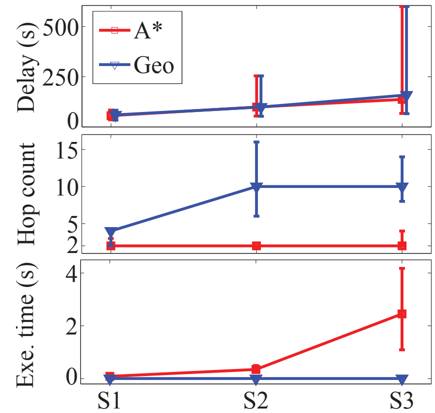

Figure 5: Simulation results in terms of delay, hop count, and processing overhead: median values and error bars (25% and 75% quartiles).

6.2 Field Experiment

We chose a scenario with three quadcopters and one ground station as visualized in Figure 6: MAV_2 hovers at a waypoint that is about 250 m away from the ground station (out of range as defined by the Wi-Fi cut-off range[1] of 150 m). Two ferries, MAV_1 and MAV_3, carry (and transmit) data generated by MAV_2 to the ground station. The ferries move in opposite directions. We chose this scenario to expose the algorithms to different path options even in a simple setting with few nodes. The test flight lasts for about 5 min; MAV_2 generates 5 messages per second.

Delay. Similar to simulation results for the simple scenario S1, the A^*-based algorithm and geometric forwarding show a high similarity in terms of delay (Figure 7). Larger delays observed for both algorithms correspond to disconnection periods induced by the scenario.

Hop count. Figure 8 shows the hop count. We observe that 100% of the messages are transferred via two hops to the ground station when using the A^*-based algorithm, either traversing MAV_1 or MAV_3, which is the best solution possible. In the case of geographic routing, 16.1% of the messages need four hops. This means that some messages are experiencing ping-ponging between MAVs. Using mission-aware forwarding this effect is counteracted by looking into the future.

Processing overhead. The processing overhead of the implementation of the A^*-based algorithm, optimized for the embedded platform, is promising: the median of the measured execution time is 1.92 ms, though we measured peak values of more than 10 ms as well. The processing overhead for geographic forwarding is negligible, with a median value of about 30 μs. It is worth noting that we investigate a small scenario with only three MAVs. An increase in overhead is expected for larger fleets.

[1] We introduce a cut-off range to assure that links with very low link quality are not utilized and due to practical tractability of the field test.

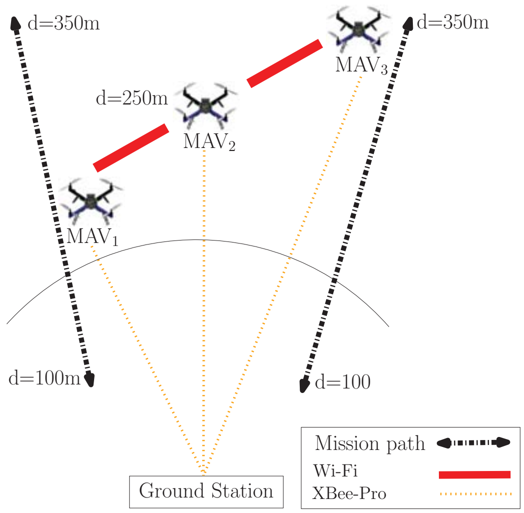

Figure 6: Field experiment scenario: hovering MAV$_2$, ferrying MAV$_1$ and MAV$_3$, and ground station (with d as respective distance to the ground station; arc visualizes the cut-off range of $150\,\mathrm{m}$).

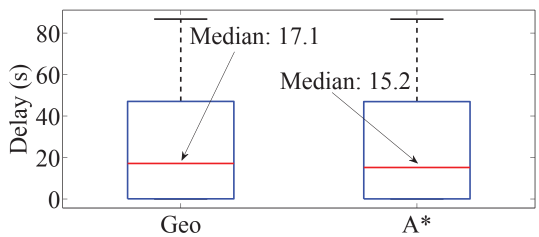

Figure 7: Delay of forwarding algorithms (field experiments).

Discussion. The field experiments confirm the feasibility to run our mission-aware algorithm in a real setting, successfully counteracting ping-ponging of packets. Although the processing performance is promising, a study is needed to answer the question whether the improved performance justifies the increase in execution time by A^* for larger fleets and longer missions in practice.

7. CONCLUSION

We provided an efficient packet forwarding method for micro aerial vehicle networks by leveraging future vehicle waypoints and connection opportunities as defined by a mission. To limit the search complexity in the temporal graph describing vehicle connectivity, we introduced a search algorithm based on A^*. Our first results observed in a simulation study and a field test with quadcopters reveal that for simple topologies, the benefits of mission-aware forwarding compared to geographic forwarding are limited, yet, unnecessary hopping of messages can be avoided. In presence of more complex topologies, mission-aware forwarding shows promising results outperforming simple geographic forwarding in terms of delay and inefficiencies in routing decision. Whereas the processing overhead is suitable for the embedded system for smaller fleets, the trade-off between better forwarding performance and computational overhead has to be studied for more scenarios and larger fleets.

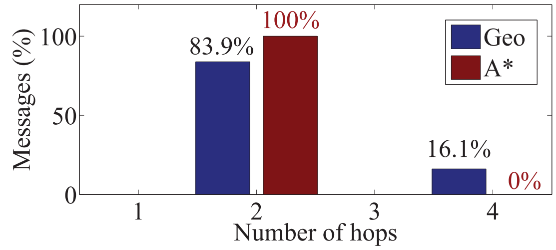

Figure 8: Hop count distribution of forwarding algorithms (field experiments).

8. ACKNOWLEDGMENTS

We thank our pilots Matthias Grob and Stefan Mach for their extraordinary support during our flight experiments.

9. REFERENCES

[1] Arducopter: Arduino-based Autopilot. `http://copter.ardupilot.com/`.

[2] M. Asadpour, S. Egli, K. A. Hummel, and D. Giustiniano. Routing in a Fleet of Micro Aerial Vehicles: First Experimental Insights. In *Proc. of 3rd ACM AIRBORNE*, pages 9–10, 2014.

[3] M. Asadpour, D. Giustiniano, K. A. Hummel, S. Heimlicher, and S. Egli. Now or Later?: Delaying Data Transfer in Time-critical Aerial Communication. In *Proc. of 9th ACM CoNEXT*, pages 127–132, 2013.

[4] M. Asadpour, B. Van den Bergh, D. Giustiniano, K. Hummel, S. Pollin, and B. Plattner. Micro Aerial Vehicle Networks: An Experimental Analysis of Challenges and Opportunities. *IEEE Communications Magazine*, 52(7):141–149, 2014.

[5] P.-C. Cheng, K. Lee, M. Gerla, and J. Härri. GeoDTN+Nav: Geographic DTN Routing with Navigator Prediction for Urban Vehicular Environments. *Mobile Networks and Applications*, 15(1):61–82, 2010.

[6] P. Hart, N. Nilsson, and B. Raphael. A Formal Basis for the Heuristic Determination of Minimum Cost Paths. *IEEE Trans. on Systems Science and Cybernetics*, 4(2):100–107, 1968.

[7] A. Jimenez-Pacheco, D. Bouhired, Y. Gasser, J.-C. Zufferey, D. Floreano, and B. Rimoldi. Implementation of a Wireless Mesh Network of Ultra Light MAVs with Dynamic Routing. In *Proc. of IEEE Globecom Workshops*, pages 1591–1596, 2012.

[8] F. Kuhn, R. Wattenhofer, Y. Zhang, and A. Zollinger. Geometric Ad-hoc Routing: Of Theory and Practice. In *Proc. of 22nd PODC*, pages 63–72, 2003.

[9] E. Kuiper and S. Nadjm-Tehrani. Geographical Routing in Intermittently Connected Ad hoc Networks. In *Proc. of AINAW Workshops*, pages 1690–1695, 2008.

[10] R. Muzaffar and E. Yanmaz. Trajectory-aware Ad hoc Routing Protocol for Micro Aerial Vehicle Networks. In *Proc. of IMAV*, 2014.

[11] R. Shirani, M. St-Hilaire, T. Kunz, Y. Zhou, J. Li, and L. Lamont. The Performance of Greedy Geographic Forwarding in Unmanned Aeronautical Ad-hoc Networks. In *Proc. of 9th CNSR*, pages 161–166, 2011.