Predictive Routing for Dynamic UAV Networks

Arnau Rovira-Sugranes, Abolfazl Razi
School of Informatics, Computing and Cyber Security (SICCS)
Northern Arizona University, Flagstaff, AZ 86011
Emails:{ar2832, abolfazl.razi}@nau.edu

***Abstract*—The emerging Flying Ad-Hoc Networks (FANETs) provide efficient infrastructure solutions for a wide range of military and commercial applications. These networks are typically composed of high-speed Unmanned Aerial Vehicles (UAVs) and form dynamic network topologies. As such, conventional routing algorithms do not efficiently accommodate these continued topology changes and exhibit poor delay performances. In this paper, we introduce an optimal routing algorithm for UAV networks with queued communication systems based on Dijkstra's shortest path algorithm as a primary step towards developing a fully predictive communication platform.**

The core idea is to incorporate the anticipated locations of intermediate nodes during a transmission session into the path selection criterion. We further evaluate the impact of measurement uncertainties on choosing the optimal path and on the resulting end-to-end delay. The simulation results confirm the superior delay performance of the proposed algorithm compared to the conventional routing algorithms. The enhancement is more significant, when the network size is larger, the relative node velocities are higher, and the average waiting times in transmit buffers are longer.

I. Introduction

In recent years, Unmanned Aerial Vehicles (UAV) are emerging as valuable and suitable platforms for many civilian and military applications including transportation, wildlife monitoring, fire management, precision agriculture, surveillance, border patrolling, and many more yet to come [1]. UAVs demonstrated several advantages over conventional piloted aircrafts. For example, a UAV has a lower production cost and is smaller in size and weight. Moreover, it has an increased maneuverability level and requires minimal human interventions. With the recent technological advances in design and implementation of drones with improved flight stability, this technology is replacing conventional methods in several applications. More than 2 million drones shipped in 2016 have realized a global market revenue of $4.5 billion and is expected to reach $11.2 billion by 2020 [2].

The future trend of UAV technology is known to be utilizing networks of autonomous UAVs with complementary sensing and actuation equipment to collectively perform complicated tasks. For instance, US Navy has recently launched the LOCUST project to perform military attacks using UAV swarms [3]. Cooperation among networks of UAV swarms will facilitate long range missions across a vast coverage area and may involve massive sensory and imagery information exchange. A majority of current UAV communication protocols are borrowed from terrestrial wireless networks and hence are extremely inefficient in accommodating continued topology changes of UAV networks [4]. The main reason is that decision making occurs at several stages of a communication protocol solely based on the current network status and as such it falls behind abrupt network topology changes [5]–[7]. An immediate consequence is frequent link losses and non-optimal packet forwarding algorithms. This project is an effort to address this issue as a beginning step towards developing a fully predictive communication platform for UAV networks. The main idea is to incorporate predicted network topology in routing algorithms.

Most commercial UAV systems either utilize Air to Ground (A2G) communication with start configuration, where all UAVs directly communicate to a ground station, or employ Air to Air (A2A) communication with Ad-hoc networking. Both approaches are extremely inefficient in heavy traffic modes (e.g. video traffic) for the obvious reason of repeated traffic payloads. Therefore, it is advantageous to use A2A communications with targeted routing protocols in UAV networks [8]–[10].

Satisfactory results have been obtained implementing routing protocols based on Dijkstra's shortest path algorithm and optimized multicast communication routing protocols for Vehicular Ad-Hoc Networks (VANETs) [11], [12]. In [13], the impact of network topology change is taken into account through a stable routing protocol which uses a cognitive agent based routing method. It finds the most stable path by considering velocity, direction and the link expiry time using fuzzy logic. Also routing algorithms based on *reinforcement learning* (e.g. Q-routing) are proposed in order to gradually learn the link behaviors based on the current transmission experiences [14]. These methods heavily rely on the assumption of low-speed vehicles moving in a 2D space with a predefined patterns determined by the roads. As such network topologies are slow varying and hence link variations can be learned in a relatively long training phases. Secondly, the network topology changes are highly linear. These assumptions are not legitimate in UAV networks with freely flying potentially high-speed UAVs.

There have been a few works considering the problem of optimal routing for dynamic UAV network topologies. In [15], a Fountain-code based Greedy Queue and Position Assisted (FGQPA) routing protocol is introduced to deal with dynamic network topologies. It first designs a Power Allocation and Routing (PAR) policy to relieve the effect of the queue backlog and then it employs a *nearest span* scheme to direct packets to the destination with a small delay. However, this paper considers a particular scenario with stationary destination node, which is not applicable to the general case of mobile destination nodes. Moreover, it only maintains one physical queue for all network flows, whereas in reality each node might have a different waiting time.

Another routing method proposed for dynamic conditions is Predictive-OLSR [16]. The key idea is to exploit GPS information and weight the Expected Transmission (ETX) count

metric to estimate the quality of the link. The numerical results show that Predictive-OLSR significantly outperforms conventional routing methods by providing a reliable communication in dynamic conditions. Nevertheless, this method assumes that only the source node is moving, hence not applicable to fully dynamic UAV networks. Further, it requires GPS-based positioning. In this work, we assume that the network topologies are anticipated either based on the access to the online path-planning information by UAVs (e.g. [17]) or using model-based object motion trajectory prediction methods (e.g. [18]. As such, the anticipated locations of network nodes with some prediction error are available to network nodes.

To approach this problem, we consider a hierarchical setup where the networks comprises several UAV clusters. Within-cluster routing is performed using conventional routing algorithms since the relative distances are almost constant for UAVs due to their correlated motion trajectories. However, predictive routing is designed for between-cluster communications. The rest of this paper organized as follows. In Section II, the utilized system model is explained. The proposed algorithm is provided in full detail in Section III. Simulations results are presented in section IV followed by concluding remarks in section V.

II. Network Mobility Model

We consider a large-scale network of UAVs that is composed of multiple UAV swarms. A UAV swarm includes a set of UAVs with different and complementary equipment that fly together and collectively perform a complicated task. Therefore, a swarm forms a local subnetwork with relatively mild topology changes. Therefore, we use conventional routing for within-swarm communications. On the other hand, different swarms are assigned with independent motion trajectories and hence they form an extremely dynamic topology. Therefore, predictive routing is used for inter-swarm communications. We follow the popular clustered infrastructure, where an information flow includes three parts: i) from the source node to the respective cluster-head, ii) from the source cluster-head to the destination cluster-head and iii) form the destination cluster-head to the destinations node; these parts are shown by black, red and green colors in Fig. 1. The problems of choosing proper cluster-head and performing conventional routing are well studied in the literature and the focus of this paper is on developing a predictive routing algorithm for the second part of the communication chain.

The network of cluster-heads is composed of N nodes, each of which representing a swarm. Hereafter, we call a cluster-head node and denote it by $n_i,\ i = 1, 2, \ldots, N$ for notation convenience. The network of N nodes form a time-varying undirected weighted graph $\mathcal{G}(t) = \{V, E(t)\}$, the so called *contact graph*. Here, $V = \{n_1, \ldots, n_N\}$ is the set of vertices and $E(t)$ is set of edges $e_{ij},\ i, j \in \{1, 2, \ldots, N\}$ with respective time-varying weights $w_{ij}(t)$.

In order to generate a representative contact graph with a predefined number of nodes, we take the following assumptions. The edges follow an iid Bernoulli distribution with parameter s, i.e. $\Pr(e_{ij} = 1) = 1 - \Pr(e_{ij} = 0) = s$, where s controls the sparsity of the contact graph. We consider bi-way communications with symmetric channels and hence $e_{ij} = e_{ji}$ and $w_{ij}(t) = w_{ji}(t)$. The weights $w_{ji}(t)$ are arbitrary edge

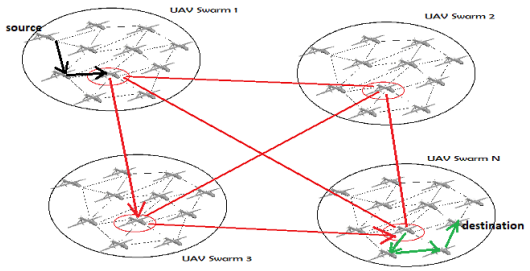

Fig. 1. The proposed system model for a large scale UAV network, which consists of multiple UAV swarms.

metrics based on the optimization objective and depend on the pairwise distances $d_{ij}(t)$. In order to simulate node locations $\vec{l}_i(t) = [x_i(t)\ y_i(t)]$, we randomly place the nodes at initial locations $l_i(0)$ using *uniform* distribution within a rectangular coverage area. For notation convenience, we use 2D space but extension to 3D space is trivial. We further assign nodes with randomly generated initial velocities $\vec{v}_i(0) = [v_i^{(x)}(0)\ v_i^{(y)}(0)]$ and acceleration vectors $\vec{a}_i = [a_i^{(x)}\ a_i^{(y)}]$. Then, we use Durbin's curve equations to represent the motion trajectories in terms of a discrete state transition model as follows:

$$\begin{cases} \vec{s}_i(k+1) & = A\vec{s}_i(k) + B\vec{a}_i(k) + \vec{w}_i(k), \\ \vec{o}_i(k) & = \gamma_i(k) C\vec{s}_i(k) + \vec{z}_i(k), \end{cases} \tag{1}$$

where we have:

$$\vec{s}_i(k) = \left[x_i(k)\ y_i(k)\ v_i^{(x)}(k)\ v_i^{(y)}(k)\right]^T, \quad \text{(state vector)}$$

$$\vec{a}_i(k) = \left[a_i^{(x)}(k)\ a_i^{(y)}(k)\right]^T, \quad \text{(input vector)}$$

$$\vec{o}_i(k) = \left[o_i^{(x)}(k)\ o_i^{(y)}(k)\right]^T, \quad \text{(observation vector)}$$

$$A = \begin{bmatrix} \vec{I}_{2\times 2} & dt\vec{I}_{2\times 2} \\ \vec{0}_{2\times 2} & \vec{I}_{2\times 2} \end{bmatrix},$$

$$B = \left[\vec{0}_{2\times 2} \quad \vec{I}_{2\times 2}\right], C = \left[\vec{I}_{2\times 2} \quad \vec{0}_{2\times 2}\right]^T,$$

$$\vec{w}_i(k) \sim \mathcal{N}(\vec{0}, \vec{R}_i), \quad \vec{z}_i(k) \sim \mathcal{N}(\vec{0}, \vec{Q}_i). \tag{2}$$

Here k represents the discrete time point $k$dt for an arbitrarily chosen time step dt. Also, $\vec{w}_i(k)$ and $\vec{z}_i(k)$ are zero mean Gaussian distributed random vectors of covariances R_i and Q_i that respectively represent the system and observation noise terms. Finally, $\gamma_i(k)$ is a Bernoulli distributed random variable $\Pr(\gamma_i(k) = 1) = \lambda$ to capture object tracking success. The optimal estimation of the node locations when measurement is available ($\gamma_i(k) = 1$) as well as the optimal online prediction of locations when the measurement is absent ($\gamma_i(k) = 0$) for known input vector $\vec{a}_i$ can be obtained using Kalman filtering with intermittent observation [19]. Therefore, we assume that the location estimates and predictions $\hat{l}_{ij}(t)$ for all neighbor nodes are available to each node n_i using a proper tracking system. The prediction error is modeled as $\vec{e}_i(t) = \vec{l}_i(t) - \hat{l}_i(t) \sim \mathcal{N}(0, \sigma^2 \mathbf{I}_{2\times 2})$, where σ^2 is the prediction error variance. Path selection decisions are made by UAVs based on the predicted locations of neighboring nodes $\hat{l}_{ij}(t)$, whereas the actual delay calculations are made based on the actual locations $l_{ij}(t)$.

III. Predictive Routing Algorithms

Before elaboirating on the details of the proposed algorithm, we first review the celebrated Dijkstra's shortest path algorithm as follows.

A. Dijkstra's shortest path algorithm

The conventional routing method is based on the standard Dijkstra's shortest path algorithm, which finds the shortest path from the source node to the destination (or to all existing nodes in another variant of this algorithm) in a contact graph. In this algorithm, the contact graph and associated edges are considered to be fixed during during the execution of the algorithm. This assumption comes from the negligible variation of pairwise node distances during a transmission session in connection-oriented communications or during a packet transmission in connection-less communications. The operation of this algorithm is based on assigning distance values to all nodes. All nodes are marked unvisited and all are initialized with infinity values except the source node which is assigned with a zero distance value. Then, the distance values of all neighbors of the source are update simply by adding the source nodes distance value to the weight of the connecting edge. Once all neighbors of the source node are update the source node is marked visited and one of the neighbor nodes with the smallest distance value is chosen as the current node. The same process is repeated for the new current node. When updating a node distance value, if the new value is larger than the previous value, we avoid updating. The algorithm finishes when the destination node is marked visited (or all nodes marked visited in another variant of this algorithm).

This intuitive algorithm is very efficient and is the fundamental core of most shortest path algorithms. However, this algorithm fails in finding the shortest path in extremely dynamic networks, where the edge metrics are subject to substantial changes especially in queuing communication systems. Fig. 2 illustrates a failure scenario.

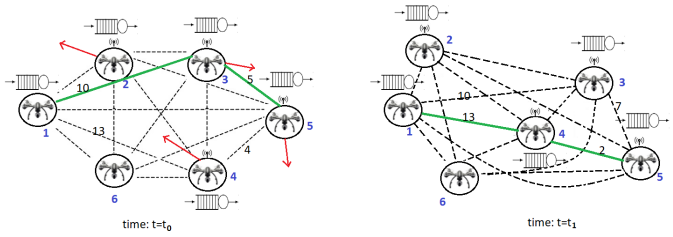

Fig. 2. The impact of UAV network topology change on the optimal path from source node n_1 to destination node n_5. In a stationary network in left hand side, the optimal path is $1-3-5$ with distance value of $10+5=15$. However, if the network topology changes, this path results in a distance value of $10+7=17$, which is not optimal anymore. In fact, the path $1-4-5$ yields a lower distance value of $13+2=15$ as shown in the right hand side.

Suppose that we are interested in finding a shortest path for a packet transmission from source node n_1 to destination node n_5. Executing a convention shortest path algorithm based on the edge metric at time $t=t_0$ will results in choosing the path $1-3-5$ with a minimum distance value of $d_{13}(t_0)+d_{35}(t_0)=10+5=15$, which is less than all alternative routs including $1-4-5$ with distance value of $d_{14}(t_0)+d_{45}(t_0)=13+4=17$, as depicted in left hand side of Fig. 2. However, in a dynamic network, once the packet reaches the intermediate node, the network topology is changed and the previously selected path $1-3-5$ will result in an actual distance value of $d_{13}(t_0)+d_{35}(t_1)=10+7=17$, which is not optimal anymore. If the packet would have chosen the path $1-4-5$ at the first place, the distance value of $d_{14}(t_0)+d_{45}(t_1)=13+2=15$ would have been achieved. This non-optimality is due to the lack of predictive information about the upcoming network topology change, when the algorithm is executed at time $t=t_0$. However, if fairly reliable anticipated edge metrics were present at time $t=t_0$, the globally optimal path could have been found. This is the main motivation behind the proposed algorithm as detailed next.

B. Proposed shortest path algorithm for dynamic networks

The proposed algorithm is a modification of the Dijkstra's shortest path algorithm by incorporating predicted pairwise distances into the path selection criterion. Dijkstra's algorithm is chosen because it reduces the amount of computation among other shortest path algorithms. The proposed algorithm involves starting from the source node at time 0 and finding the next intermediate node considering the predicted network topology (i.e. considering the intermediate node locations when the packet is expected to reach that node). Once the packet reaches the second node, network topology is updated based on the actual pairwise distances. Then, the problem reduces to finding the optimal path from the second node to the destination using a degraded contact graph excluding the source node. At each step, the edge metrics are calculated by incorporating predicted node locations. This procedure repeats until the remaining network degrades to the destination node. The following is the edge update procedure.

As mentioned earlier, a UAV network can be represented by an undirected contact graph, where the edges are associated with time-varying weights $w_{ij}(t)$. In a general formulation, we can consider edge weights as $w_{ij}(t)=f(d_{ij}(t))+g_{ij}(t)$, where $f(d_{ij}(t))$ mimics the distance-related terms (e.g. propagation delay) and $g_{ij}(t)$ represents other terms (waiting delay). In this paper, the objective is to minimize the transmission delay. Thus, an edge metrics $w_{ij}(t)$ represents the delay associated with the edge e_{ij} accounting for time span from the epoch that a packet reaches node i denoted by t_i until the epoch it is delivered to node j, denoted by t_j. This time includes two components: i) waiting time in transit buffer of node i at time t_i, denoted by $w_i(t_i)$ and the actual propagation time. The propagation time for a packet that leaves node n_i at time t_i+w_i and reaches the mobile node j, denoted by $p_{ij}(t_i+w_i)$ is calculated by solving the following equations:

$$\begin{cases} d_{ij}=|\hat{l}_j(t_i+w_i+p_{ij}(t_i+w_i))-l_i(t_i+w_i)|_2, \\ p_{ij}(t_i+w_i)=d_{ij}/c, \end{cases} \tag{3}$$

where the first equations characterizes the predicted distance between node n_i (when the packet leaves this node at time t_i+w_i) and node n_j (when the packet reaches this node at time $p_{ij}(t_i+w_i)$). The second equations relates this distance to the propagation time $p_{ij}(t_i+w_i)$ using the wave propagation phenomenon with the speed of light c. The operator $|\vec{x}|_2=\sqrt{x_1^2+x_2^2}$ is the second norm of vector $\vec{x}=[x_1 x_2]$. Note that (3) may require numerical methods to solve the nonlinear

equation, since $\hat{l}_j(t)$ is the solution of (1) and in general is not linear. Once, we solve (3), edge metrics are determined as

$$w_{ij}(t_i) = w_i + p_{ij}(t_i + w_i), \quad t_j = t_i + w_{ij}. \tag{4}$$

The total transmission delay of a K-hop path $\mathcal{P} = (n_{i_1}, n_{i_2}, \ldots n_{i_K})$ is calculated as:

$$d(\mathcal{P}) = \sum_{k=1}^{K-1} w_{i_k i_{k+1}}(t_{i_k}) = \sum_{k=1}^{K-1} w_{i_k} + p_{i_k i_{k+1}}(t_{i_k} + w_{i_k}) \tag{5}$$

The following is the summary of the proposed algorithm.

Algorithm 1: Optimal shortest path algorithm

Data: N: number of nodes, n_s: source, n_d: destination, $\mathcal{G} = (V, E_{ij}(t))$, $\vec{l}_i(t)$, $\hat{l}_i(t)$, $w_i(t)$, c
Result: $\mathcal{P}$: optimal path, $d(\mathcal{P})$: path delay
(initialization):
$\{Visited\} \leftarrow \{n_s\}$; $t_{n_s} = 0$;
$\{Unvisited\} \leftarrow V \setminus \{n_s\}$; $t_n = \infty \forall n \in \{Unvisited\}$;
$n_i \leftarrow n_s$;
(finding optimal path:)
while $n_d \notin \{Visited\}$ **do**
  for $n_j \in \{Unvisited\}$ **do**
    if $e_{ij} = 1$ **then**
      update $w_{ij}(t_i)$ using (4);
      update d_{ij} using (3);
      $t_j = t_i + w_{ij}(t_i)$
    $\{Visited\} \leftarrow \{Visited\} \cup \{n_i\}$;
    $\{Unvisited\} \leftarrow V \setminus \{n_i\}$;
    update next node: $n_i \leftarrow \underset{j \in \{Unvisited\}}{\text{argmin}} \{t_j\}$;
    Previous(n_j)=n_i
(form the optimal path:)
$\mathcal{P} = \{n_d\}$; $n = n_d$
while $n \neq n_s$ **do**
  $n \leftarrow$ Previous(n);
  $\mathcal{P} \leftarrow n \cup \mathcal{P}$
(calculate delay:)
t=0;
for $n \in \mathcal{P}$ **do**
  $i = n; j =$ next(n);
  calculate d_{ij} using (3) by substituting $\hat{l}_j(t)$ with $l_j(t)$;
  update $w_{ij}(t)$ using (3) and (4);
  $t \leftarrow t + w_{ij}(t)$
$d(\mathcal{P}) = t$

Note that predicted node locations $\hat{l}_j(t)$ are used to find the optimal path, while actual node locations $l_j(t)$ are used to calculate the actual end-to-end delay $d(\mathcal{P})$. In the case of linear motions, we have $B = 0$ in (1) and the state transition equations are reduced to $\vec{l}_i(t+1) = \vec{l}_i(t) + dt\vec{v}_i$+noise, which further simplifies the calculations by linearizing (3).

IV. Experimental Setup and Simulation Results

For both conventional and optimal methods, random networks have been generated using *uniform* distributions for the initial real positions $\vec{l}_i(t = 0)$, initial velocities $\vec{v}_i(0)$, initial accelerations$\vec{a}_i(0)$ and the waiting time $w_i(0)$. The motion trajectories then simulated using state transition equations in (1). The edge weights $w_{ij}(t)$ are obtained based on the pairwise distances as detailed in Section III-B.

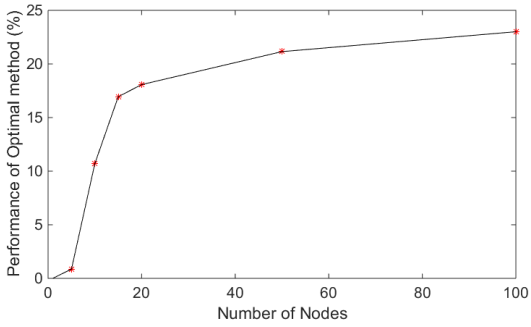

Fig. 3. Performance of the optimal method in percentage of end-to-end delay when the number of nodes is increasing.

Shortest path for the dynamic networks are obtained using both the standard version of Dijkstra's algorithm and the proposed algorithm. For both algorithms, the actual node locations are used to quantify the end-to-end delay.

Fig. 3 represents the impacts of the number of nodes in the performance of the proposed algorithms. It is shown that the improvement of the proposed algorithm with respect to the standard version increases with the number of nodes. An intuitive justification is that in a larger network, the number of links in the optimal path is larger, which results in a larger performance difference between the two algorithms. An interesting results is that the improvement follows a logarithmic behavior, since the number of links from the source to destination also follows a logarithmic relation with the number of nodes. For a network of $N = 100$ nodes, the performance improvement is about 25%.

Fig. 4 shows the impact of the average node velocity on the end to end delay for both methods. For higher average node velocities, the network is more dynamic and hence the proposed algorithm exhibits a better performance compared to the conventional method. Therefore, this algorithm is more appropriate for UAVs networks composed of high-speed flying objects.

Another important parameter is the average waiting time of the node. Intuitively, the larger the waiting time, the more

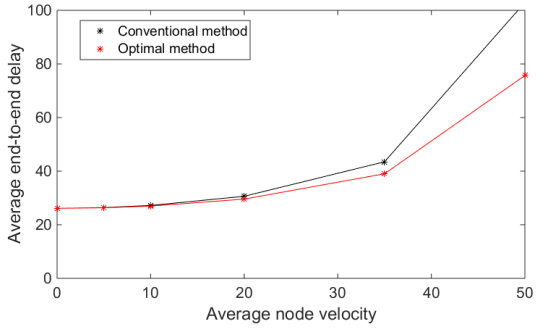

Fig. 4. The impact of the average node velocity on the end to end delay for the selected paths using the proposed and the conventional Dijkstra's shortest path algorithms.

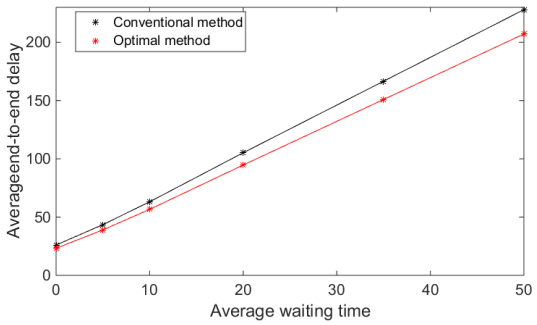

Fig. 5. The impact of the average waiting time of the nodes on the end to end delay.

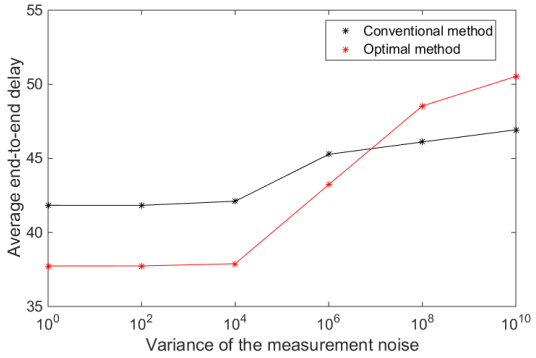

Fig. 6. Average end-to-end delay for both conventional and proposed algorithms under prediction uncertainties.

deviant is the current location of the subsequent node from its initial value. As such, the improvement gain for the proposed method is larger as shown in Fig. 5.

Lastly, we evaluate the impact of the uncertainty of predictions, as depicted in Fig. 6. It is seen that for small prediction error variance, the average end-to-end delay of the optimal method is lower than that of the conventional method. Once we increase the variance of the prediction error, the optimal method becomes less effective than the conventional method. This happens because the probability of choosing incorrect paths increases as the proposed algorithm selects more incorrect intermediate nodes. Incorporating the prediction results when the measurement system is extremely noisy, is irrelevant.

V. Conclusions

In this paper, an optimal routing method based on Dijkstra's shortest path algorithm is proposed for UAV networks by incorporating predicted network topology. The simulation results confirm that incorporating the anticipated network topology into the path selection criterion significantly improves the end-to-end delay performance compared to the conventional Dijkstra's algorithm. The performance improvement is more significant for larger networks, larger average waiting time and higher node velocities. Moreover, for UAVs with limited communication ranges, the proposed algorithm can be used to prolong the network connectivity by excluding the links that are likely to get disconnected due to exceeding the predefined communication range and solve the problem of frequent link losses. A potential future research direction is to assess the impact of prediction imperfectness through characterizing the probability of selecting non-optimal links.

References

[1] S. Hayat, E. Yanmaz, and R. Muzaffar, "Survey on unmanned aerial vehicle networks for civil applications: a communications viewpoint," *IEEE Communications Surveys & Tutorials*, vol. 18, no. 4, pp. 2624–2661, 2016.

[2] "Gartner says almost 3 million personal and commercial drones will be shipped in 2017," Feb 2017. [Online]. Available: http://www.gartner.com/newsroom/id/3602317

[3] "LOCUST: autonomous, swarming uavs fly into the future," April 2015. [Online]. Available: http://www.onr.navy.mil/Media-Center/Press-Releases/2015/LOCUST-low-cost-UAV-swarm-ONR.aspx

[4] S. Hayat, E. Yanmaz, and R. Muzaffar, "Survey on unmanned aerial vehicle networks for civil applications: A communications viewpoint," *IEEE Communications Surveys Tutorials*, vol. 18, no. 4, pp. 2624–2661, Fourth quarter 2016.

[5] F. Afghah, A. Razi, and A. Abedi, "Stochastic game theoretical model for packet forwarding in relay networks," *Telecommunication Systems*, pp. 1–17, 2013.

[6] A. Abedi, F. Afghah, and A. Razi, "Resource management in cyber-physical systems," *Cyber-Physical System Design with Sensor Networking Technologies*, vol. 2, p. 177, 2016.

[7] A. Razi, F. Afghah, and A. Abedi, "Power optimized dstbc assisted dmf relaying in wireless sensor networks with redundant super nodes," *IEEE Transactions on Wireless Communications*, vol. 12, no. 2, pp. 636–645, 2013.

[8] T. Zaza and A. Richards, "Intelligent foresight for uav routing problems," in *2016 UKACC 11th International Conference on Control (CONTROL)*, Aug 2016, pp. 1–6.

[9] H. Ding and D. Castan, "Fast algorithms for uav tasking and routing," in *2016 IEEE Conference on Control Applications (CCA)*, Sept 2016, pp. 368–373.

[10] Y. Cai and K. Sekiyama, "Subgraph matching route navigation by uav and ground robot cooperation," in *2016 IEEE Congress on Evolutionary Computation (CEC)*, July 2016, pp. 4881–4886.

[11] J. d. Zhang, Y. j. Feng, F. f. Shi, G. Wang, B. Ma, R. s. Li, and X. y. Jia, "Vehicle routing in urban areas based on the oil consumption weight - dijkstra algorithm," *IET Intelligent Transport Systems*, vol. 10, no. 7, pp. 495–502, 2016.

[12] W. Farooq, M. A. Khan, and S. Rehman, "Amvr: A multicast routing protocol for autonomous military vehicles communication in vanet," in *2017 14th International Bhurban Conference on Applied Sciences and Technology (IBCAST)*, Jan 2017, pp. 699–706.

[13] H. C. Premkumar, V. R. Budyal, and M. S. Kakkasageri, "Cognitive agent based stable routing protocol for vehicle-to-vehicle communication," in *2016 IEEE Annual India Conference (INDICON)*, Dec 2016, pp. 1–5.

[14] S. P. Choi and D.-Y. Yeung, "Predictive q-routing: A memory-based reinforcement learning approach to adaptive traffic control," in *Advances in Neural Information Processing Systems*, 1996, pp. 945–951.

[15] C. Yin, Z. Xiao, X. Cao, X. Xi, P. Yang, and D. Wu, "Enhanced routing protocol for fast flying uav network," in *2016 IEEE International Conference on Communication Systems (ICCS)*, Dec 2016, pp. 1–6.

[16] S. Rosati, K. Kruelecki, G. Heitz, D. Floreano, and B. Rimoldi, "Dynamic routing for flying ad hoc networks," *IEEE Transactions on Vehicular Technology*, vol. 65, no. 3, pp. 1690–1700, March 2016.

[17] G. D. Goez, R. A. V. Velez, and J. S. B. Valencia, "Uav route planning optimization using pso implemented on microcontrollers," *IEEE Latin America Transactions*, vol. 14, no. 4, pp. 1705–1710, April 2016.

[18] G. Aoude, J. Joseph, N. Roy, and J. How, "Mobile agent trajectory prediction using bayesian nonparametric reachability trees," in *Infotech@ Aerospace 2011*, 2011, p. 1512.

[19] J. Rohde, J. E. Stellet, H. Mielenz, and J. M. Zllner, "Localization accuracy estimation with application to perception design," in *2016 IEEE International Conference on Robotics and Automation (ICRA)*, May 2016, pp. 4777–4783.