Hou et al. EURASIP Journal on Wireless Communications and Networking (2020) 2020:140
https://doi.org/10.1186/s13638-020-01707-3

EURASIP Journal on Wireless Communications and Networking

RESEARCH Open Access

Improving aerial image transmission quality using trajectory-aided OLSR in flying ad hoc networks

Chen Hou, Zhexin Xu, Wen-Kang Jia, Jianyong Cai and Hui Li*

* Correspondence: hli@fjnu.edu.cn
Fujian Provincial Engineering Research Center for Optoelectronic Sensing Technology, College of Photonic and Electronic Engineering, Fujian Normal University, Fuzhou 350117, Fujian, China

Abstract

UAVs have been widely used in various applications. Auto coordination of multiple UAVs through AI or mission planning software can provide significant improvements in many applications, including battlefield reconnaissance, topographical mapping, and search and rescue missions. Under such circumstances, the trajectory information is known for a set amount of time, and the system's performance relies on the network between UAVs and their base. Here, a new protocol is proposed that takes the trajectory of UAVs as a known factor and uses it to improve optimized link state routing (OLSR). In this protocol, *Q*-learning is adopted to find the best route for the system. Additionally, a packet forwarding arrangement is described that addresses the common problem of deteriorating image quality often faced by UAVs. The simulation results show significant improvements over OLSR and GPSR under a sparsely distributed scenario, with the packet delivery ratio improved by over 30% and over 40 s reduction in the end-to-end delay.

Keywords: Trajectory-OLSR, Flying ad hoc network, Store and forward, *Q*-learning, Progressive transmission

1 Introduction

Unmanned aerial vehicles (UAVs) feature high mobility, low cost, accessibility, and complex terrain compatibility, as they are widely used in battlefield reconnaissance, topographical mapping, and search and rescue missions. Among such applications, search and rescue missions are of great potential. One important advantage of utilizing UAVs when compared to other search and rescue methods is that with the help of AI and mission planning, it is possible to apply multiple UAVs to accomplish the mission more efficiently. However, such a method highly relies on network availability to transmit large amounts of data between each independent UAV and the UAVs to their base, where the data mostly consist of images. Images transmitted through UAVs often suffer from packet loss and noise, which lead to image deterioration, frame loss and a significant increase in latency. Moreover, search and rescue missions conducted by UAVs need to be sparsely distributed to cover vast areas and make the void problem of the routing algorithm more serious. Hence, to support the AI and mission planning system

to reliably send images to base, such applications require a specifically designed protocol to tackle this problem.

When compared to other protocols, multi-hop data transmission shows great potential due to its flexibility, robustness and ability to transmit data among nodes when applied in vehicles; however, short link lifetime, sparsely distributed nodes and fast changing 3D topology still requires the flying ad hoc network to continue to evolve to better suit the scenario.

There are many ways to improve the performance of ad hoc networks other than hardware, and error correction codes, i.e., Low-density Parity-check(LDPC)codes, are of great importance to enhance the transmission quality in all wireless scenarios [1, 2] and could guarantee the transmission reliability. Improving the mechanism of ad hoc protocols and making use of information from UAVs is another important approach. In this paper, we propose a novel multi-hop protocol based on OLSR to handle the short link lifetime caused by the UAV's high-speed movement. The protocol utilizes the trajectory information of the UAVs and adopts the deep learning tool *Q*-learning to improve its performance, and the issues regarding decreased image quality are addressed using a special optimization for forwarding of the image payload packet.

The multi-hop routing protocol of FANET faces many challenges, such as three-dimensional movement, which leads to complicated topology, and high speeds, which lead to short link survival time. Furthermore, sparse UAV networks often result in void problems and lead to outdated topology information in active routing protocols. Singh, Kuldeep, and A.K. Verma [3] noted that OLSR outperformed ad hoc on-demand distance vector routing (AODV) and destination-sequenced distance vector routing (DSDV) in terms of the packet delivery ratio, end-to-end delay and average throughput as 20 UAVs moved at a speed of 50 m/s. However, according to the results of M. T. Hyland et al. [4], the performance of greedy perimeter state routing (GPSR) is superior to that of AODV and OLSR in most aspects as the density of nodes increases to 200. In summary, the multi-hop routing protocol faces severe void problems when the node density is sparse; hence, OLSR is implemented in our study because our scenario is characterized by a sparse distribution and faces a serious void problem when FANET is applied.

OLSR has been widely used in VANET because VANET and FANET have similar topology change problems, and many improvements made to handle VANET dynamic topology changes are of great value to FANET. For instance, Ge et al. [5] presented a method to select multipoint relay (MPR) nodes according to bandwidth. Hakim Badis and Khaldoun Al Agha [6] proposed modifying the hello message and propagating QoS information, and MPR selection is based on such information. HÄRRIJ et al. [7] proposed a method to select the MPR nodes according to their predicted link state. In addition to the MPR selection process, the utilization of node position information is also a valuable approach. H. Menouar et al. [8] proposed a method that used the position, velocity and range to calculate the link's lifetime and applied the link lifetime as a weighing factor in MPR selection as well as routing table construction. Sachin Sharma [9] proposed a protocol called P-OLSR that uses speed to predict the future position of the nodes and rule out the nodes moving out of the radio range. Mi, Zhichao et al. [10] presented a method that uses position and distance to weight the link error ratio. Then, the link error ratio is applied to determine the route and select MPRs.

Apparently, more information being incorporated into the MPR selection and routing processes for OLSR could greatly help the protocol handle the changing topology problem.

However, further improvements are needed for OLSR to handle FANET, and many adjustments and improvements to OLSR have been developed. For instance, Benzaid et al. [11, 12] presented an OLSR extension called Fast-OLSR. Fast-OLSR was designed for dynamic topology using increasing hello message rates. A. Alshbatat and L. Dong [13] proposed a method that adopted directional antennas and two cross-layer schemes to make the OLSR directional, namely, directional-OLSR. This mechanism takes advantage of flight information, such as altitude variations, pitch, roll, and yaw. Stefano Rosati et al. [14] proposed the predictive-OLSR, which uses the GPS information from the UAVs. In addition, this scheme weights hops in the expected transmission count metric (ETX) by a factor that considers direction and relative speed between UAVs. Pu, C [15] then took the link-quality and traffic-load factors into account in the route-finding process of OLSR. Li, C, et al. [16] presented a method that combined location, distance and neighbor changing rates in the MPR selection process and improved the OLSR performance in the FANET.

The aforementioned improved versions of OLSR tend to use present movement or location information, and some of the studies have attempted to predict future movement of UAVs with highly dynamic topology changes. However, the prediction error of these methods becomes unacceptable when the UAVs suffer from trajectory problems. Furthermore, the prediction error degrades dramatically as the predicted duration increases. In this case, the predictions are obviously unreliable. Even worse, the connectivity of links among UAVs cannot be guaranteed for sparse FANETs; in this case, a large end-to-end transmission delay leads to intolerant prediction errors of UAV movement since the prediction has to cover a long time interval.

Fortunately, the track of each UAV could be pre-planned through a mission planning system [17]. Studies performed by Xianfeng Li [18] presented an adaptive track-explore system suitable for FANET. In such a system, the UAV track is certain in the next several seconds or even minutes. Inspired by this, modified OLSR is proposed in this paper, namely, trajectory-OLSR (T-OLSR). In this proposed mechanism, each node shares its pre-planned short-term trajectory in the hello message and TC message, resulting in a slight increase in overhead. T-OLSR is specifically designed to accomplish collaborative operation in FANETs. This method has a robust and effective routing mechanism for the highly dynamic topology of FANET. The performance of T-OLSR is shown to be superior to traditional methods, especially in sparsely distributed networks.

Furthermore, to cope with the packet loss problem, which greatly affects the image quality and occurs frequently in UAVs missions, we determine the priorities of the image payload packets to decide the packet forwarding order to further improve the protocol's performance.

The remainder of this paper is organized as follows. The motivation of the proposed T-OLSR is presented in Section 2. In Section 3, the system model for T-OLSR and the relative assumptions are described. Then, our sophisticated routing mechanism is elaborated in Section 4. Furthermore, the performance of the proposed mechanism is evaluated and compared through simulation with other related classic approaches in Section 5. Finally, the paper is concluded in Section 6.

2 Motivation

OLSR is a proactive routing mechanism that relies on the link state information. The key feature of OLSR is the MPR. The topology control messages are only forwarded through MPR nodes, which effectively restrain the overhead of broadcasting. The transmission of hello messages and topology control (TC) messages are crucial procedures in OLSR to detect the network topology and estimate the link state. Each node broadcasts the hello message to all its neighbors periodically, and the message contains the sender node's information and its neighbors' information. Through the exchange of hello messages, each node detects its 1-hop neighbors and 2-hops neighbors.

In OLSR, only the MPR nodes are allowed to forward the TC message to the whole network. The TC message contains the information about the selectors of each MPR node. The selectors refer to the nodes that select the nodes as their MPR nodes. This information on selectors is embedded in its neighbor's hello message, and the messages are then written into the TC messages. In addition, the nodes detect the topology of the whole network through the received TC message. Once a TC message arrives, the node can update its routing table and calculate the shortest path to the destinations by the Dijkstra algorithm.

However, the traditional broadcast mechanism for hello messages and TC messages in OLSR is not suitable in FANET since the speed of UAVs can exceed 100 mph and the topology of the network may change dramatically. The information in the received hello message and TC message may be outdated. For a TC message, its default interval is 5 seconds in the classic OLSR design, which faces a serious problem of being outdated in a sparsely distributed network and becomes almost useless for estimating the topology and calculating the routing paths. In this case, the end-to-end delay and packet delivery ratio of the collaborate operation in sparse FANETs are unacceptable.

Moreover, the routing strategy of OLSR, which employs Dijkstra, is not suitable in FANET for the various durations of the link existent time, which leads to further deteriorated performance in a sparse FANET. Fortunately, the trajectory of UAVs can be pre-planned. Although the planned trajectory may be adjusted during the duration of cooperation among UAVs, the short-term planned trajectories are considered to be certain within a few seconds. Thus, it is possible to improve the accuracy of the topology estimation by sharing the short-term trajectories through hello messages and TC messages. Importantly, reasonable modeling of the trajectories can effectively compress the additional overhead of packets.

Hence, we proposed trajectory-OLSR, which aims to improve the end-to-end delay and packet delivery ratio in sparse FANETs. In T-OLSR, the short-term trajectories of each node within several seconds are modeled using a three-order polynomial. Then, the coefficients of these polynomials are embedded as an additional section into the hello message and TC message. In other words, the short-term trajectory of a UAV is shared as several coefficients of three-order polynomials, which slightly increase the extra overhead of the packets. Based on these short-term coefficients of the trajectories, the nodes can accurately estimate the position of other nodes and reconstruct the real-time topology of the network once they receive the hello packets or TC packets. To utilize the trajectory information in the route-finding process, *Q*-learning is used as the route-finding mechanism to further refine the network's performance, while a specifically designed packet forwarding sequence that considers progressive image transmission

is also adopted to improve the image quality. This proposed mechanism can efficiently enhance the performance of traditional OLSR in sparse FANETs in terms of the end-to-end delay and packet delivery ratio.

3 System model

3.1 Application of the proposed system in search and rescue missions

Our proposed protocol can be used in search and rescue systems. These systems consist of several UAVs that simultaneously search a target area in circle patterns as showed in Fig. 1, which are decided by the upper layer application. During the search procedure, the UAVs send control messages and aerial images to their neighbors or the back-end control centre. The system consists of N UAV circles at the same speed v and radius R above the target area where all UAVs are considered equal nodes in the system. In addition, a commercial GPS system is equipped on the UAVs to provide meter-level location accuracy. Such system mainly transmits the images of spots of interest through the whole network to support functions such as mission planning and object identification.

3.2 Trajectory description

The trajectory of each UAV should be calculated first by the upper layer in the proposed system. In our paper, circulating flight is employed, and other complex flight patterns are not considered since we focus on developing a routing mechanism, and the developed mechanism can be extended to solve other more complicated scanning patterns.

The key idea of the proposed T-OLSR is utilizing the UAVs' trajectory information in the routing mechanism. The trajectory of each UAV can be described as the functions between three-dimensional Descartes coordinates and time as follows:

$$x(\mathrm{t}) = a2 + \mathrm{a0}{*}\sin(\mathrm{a1}{*}\mathrm{t}) \tag{1}$$

$$y(\mathrm{t}) = b2 + b0{*}\cos(\mathrm{b1}{*}\mathrm{t}) \tag{2}$$

$$z(t) = t{*}c0 + c1 \tag{3}$$

where $t = t_1 - t_0$. a0, a1, a2 and b0, b1, b2 together describe the trajectory of the UAV in the x, y plane, $c0$, $c1$ describe its height, ($a2$, $a2$) is the starting point, $a0$, $b0$ are the curvature parameter, $a1$, $b1$ are the speed parameter, $c1$ is the starting height and $c0$ is the altitude changing rate. Here, we assume that the plane's height increases or decreases monotonically during this short period. It should be noted that t_0 denotes initial time when the node shares its trajectory, and $t_1$denotes an arbitrary time within the next T_s seconds. After obtaining trajectory information from the upper layer, nodes will then share the coefficients of these functions via hello packets and TC packets. As a result, the nodes that receive these packets can reconstruct the trajectory of the senders for the next T_s seconds using these coefficients. Furthermore, the routing processes are further optimized and updated according to these reconstructed trajectories.

3.3 Packet structure

The hello message packet of T-OLSR adds 128 bits per neighbor node for the eight coefficients $a0$, $a1$, $a2$, etc., as in Eq.1, 2, and 3, which fits the trajectory of each neighbor

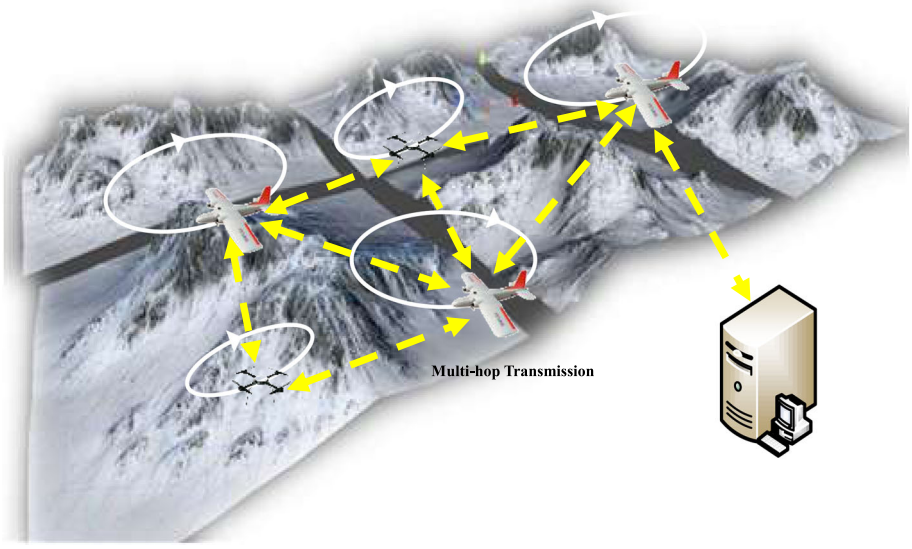

Fig. 1 System overview

in the following T_s seconds, and 4 bytes for the timestamp labeled as "initial time" to the traditional hello packet, as shown in Fig. 2. The timestamp records the time at which the nodes send a hello packet. As a result, 20 additional bytes are used to describe the trajectory of each neighbor. This structure of the hello message packet is specifically designed for sparse networks, which means that the numbers of neighbors are

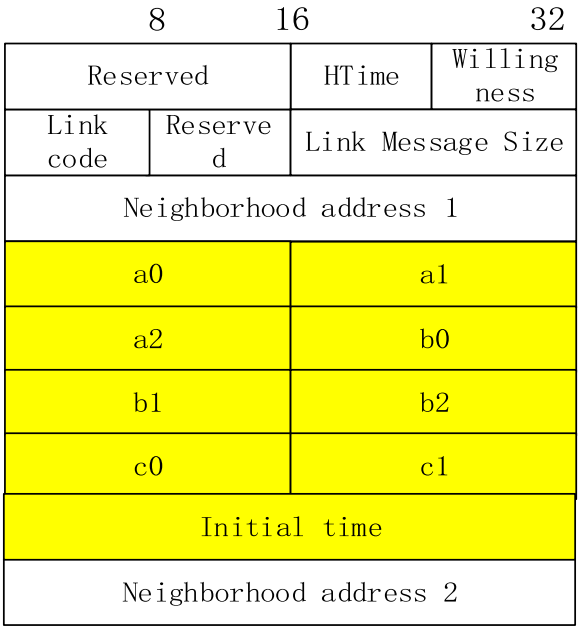

Fig. 2 Hello Message structure of T-OLSR

relatively limited. Thus, the sizes of the hello message packets are not large and remain practical.

Figure 3 depicts the structure of the TC message packet in T-OLSR. The classical TC message packet carries the information that regards each MPR node and its MPR selector. In addition, the TC message packet of the proposed mechanism contains an additional 20 bytes for each advertised neighbor. The contents of these 20 bytes are similar to the hello message.

Through the shared information of the trajectory in the hello message and TC message, the nodes can effectively estimate the change of topology in a short-term interval and calculate the position of all advertised neighbors in the following T_s seconds, while the overhead of the additional fields is acceptable in the sparse network, such as the stated search and rescue mission.

4 Trajectory-OLSR (T-OLSR)

T-OLSR's overall flow diagram is shown in Fig. 4. Overall, this process basically follows the classical OLSR. The nodes update their MPR nodes upon receiving the hello message from neighbors and update the routing table upon receiving the TC message from MPR nodes. Each node periodically sends out a hello message, and the nodes that are selected as MPR nodes send TC messages when they sense topology changes.

Compared to the classical OLSR, T-OLSR improves three procedures—hello message sending, MPR selection, and routing table updating—which is highlighted in Fig. 4; these procedures are the core of the protocol, and their performance determines the entire transmission performance. In OLSR, the sparse distributed network mainly causes the hello message and MPR selection result outdated and the routing table based on it cannot reflect the link's overall latency while the topology also suffers from outdate problem. In the T-OLSR, due to the shared trajectory information, the hello message interval can be longer, which leads to the reduction in overhead; MPR selection could consider the topology changes about to occur; more importantly, the routing table could take the trajectory of the nodes into account. The routing paths of T-OLSR

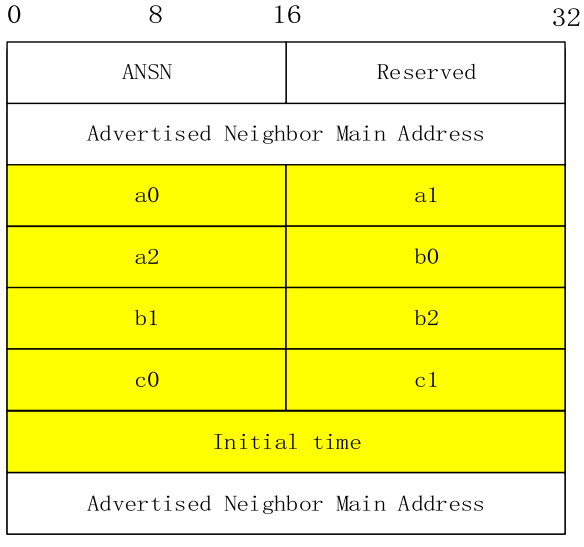

Fig. 3 TC Message structure of T-OLSR

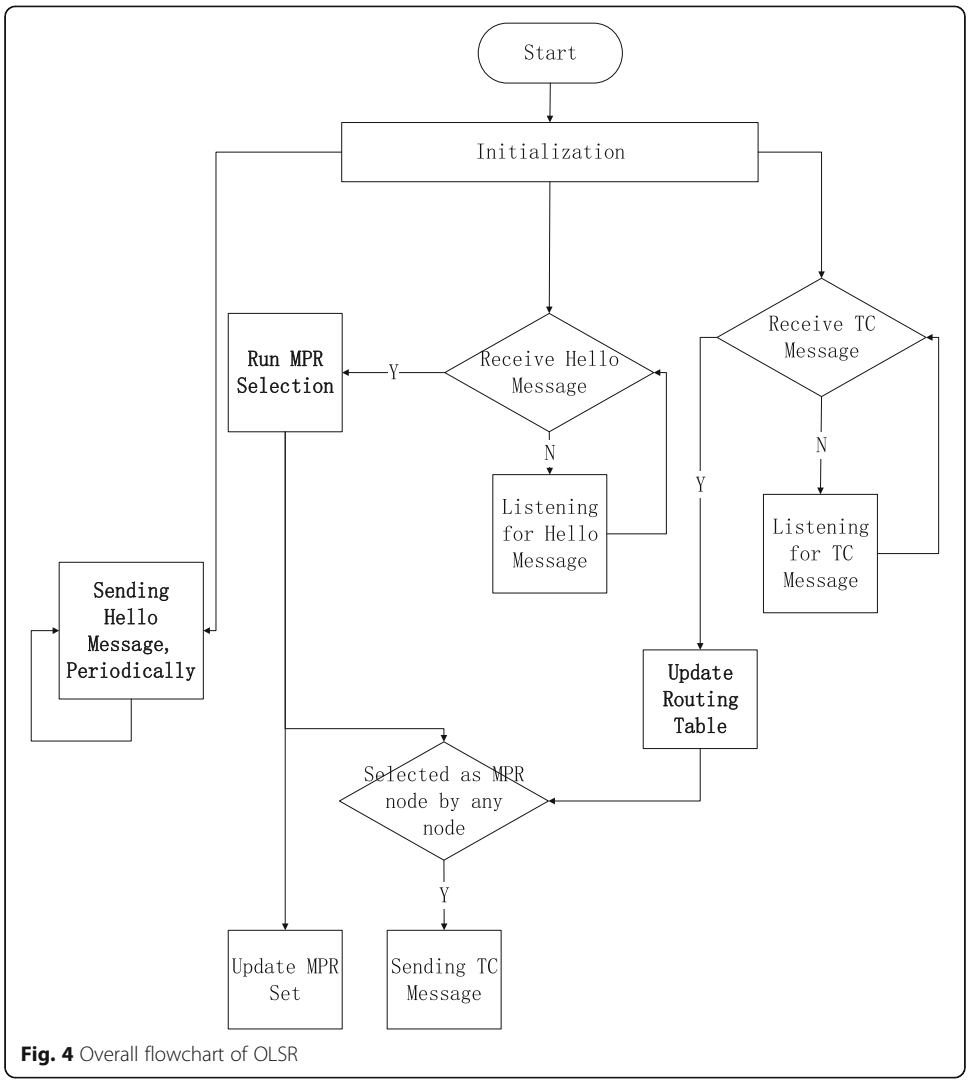

Fig. 4 Overall flowchart of OLSR

are determined through the existing links or the links that will be established in the future; such a mechanism solves the problem of outdated topology in sparse networks.

4.1 Time interval for a hello message

The time interval for the hello message, denoted as HELLO_INTERVAL or Φ in the following part, determines the awareness accuracy of the change in links and topology. The HELLO_INTERVAL of classical OLSR is set to a constant value such as 1 s. However, HELLO_INTERVAL should be shorter for UAVs due to their high speed [19]. Unfortunately, the overhead caused by frequent hello messages increases undesirably. In contrast, the HELLO_INTERVAL for T-OLSR could be set longer despite the high speed of UAVs for trajectory information embedded in the hello message. In this way, the overhead caused by the hello message can be effectively restrained. Furthermore, the awareness of the accuracy of changes in links and topology is still satisfactory. This method benefits from the shared trajectories since the nodes can estimate the links from the topology in the following T_s

seconds once they receive the hello message from T-OLSR. Thus, Ts should be optimized to create a trade-off between the reduction in overhead and the estimation error. The impact of T_s and HELLO_INTERVAL will be discussed in Section 5 through simulations.

4.2 MPR selection

The modified MPR selection mechanism is showed in Fig. 5, it also ensures the coverage of all two-hop neighbors, as in classical OLSR. However, in T-OLSR, instead of updating the MPR set until receiving the hello message, the nodes utilize the trajectory information to achieve continuous tracking of topology changes in the following T_s seconds once they receive the hello message. Then, the nodes adjust the MPR selection results according to the topology changes in sparse scenarios.

For instance, it is assumed that node A receives a hello message from its neighbors at time t_0. As shown in Eq. 1–Eq. 3, the three-dimensional positions of one-hop and two-hop neighbors can be calculated at arbitrary time $t \in [t_0, t_0 + T_s]$. Nodes i and j denote the members of the one-hop and two-hop neighbors of node A. The three-dimensional Euclidean distances between node A and node i are denoted as D_Ai. Similarly, D_ij can be defined. The links are established when the three-dimensional Euclidean distances between the two nodes are less than the communication radius R. Thus, the time when D_Ai or D_ij is equal to R can be considered the critical time, denoted as t_{link_w} $(w = 1,2...,W)$. Furthermore, the derivative of D_Ai or D_ij with respect to time

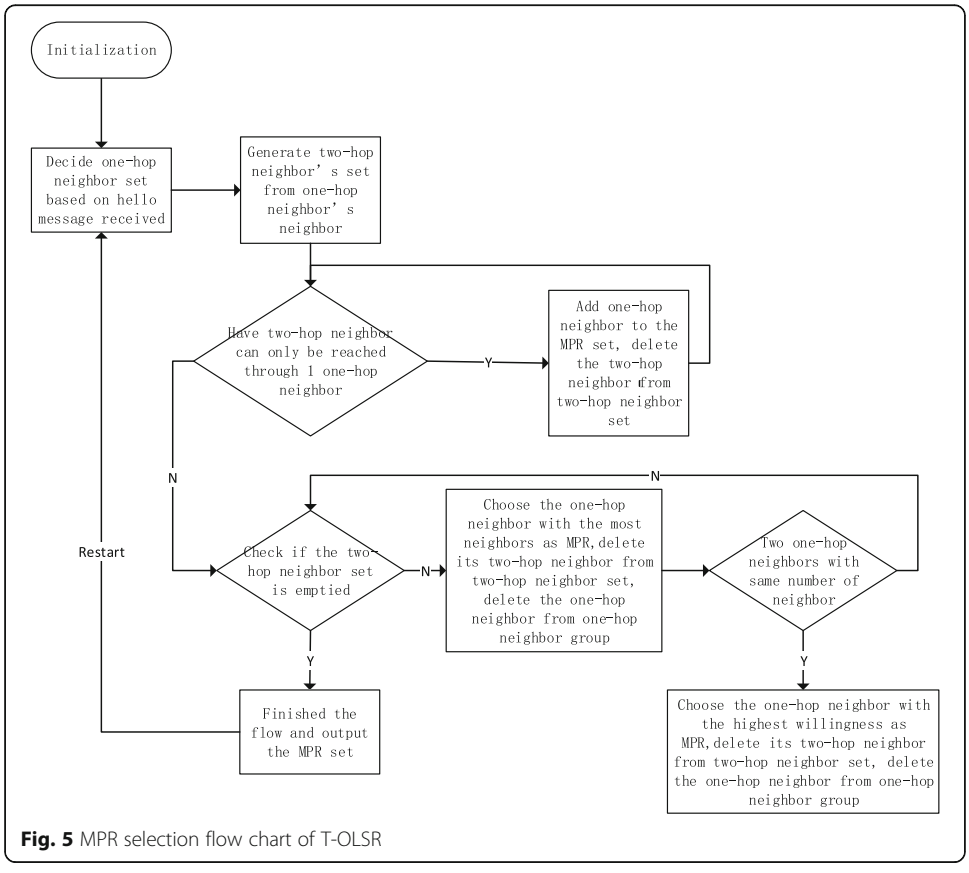

Fig. 5 MPR selection flow chart of T-OLSR

can be used to detect whether links exist between two nodes during the time interval $[t_{\text{link}_w}, t_{\text{link}_{w+1}}]$ For instance, if $\frac{d(\text{Distant}_{ij}(t))}{dt}_{t=t_{\text{link}_w}} < 0$, the link between node i and node j established at $t=t_{\text{link}_w}$, and the link continues to exist during $[t_{\text{link}_w}, t_{\text{link}_{w+1}}]$, where the link disconnects at $t=t_{\text{link}_{w+1}}$ when $\frac{d(\text{Distant}_{ij}(t))}{dt}_{t=t_{\text{link}_{w+1}}} > 0$. The one-hop and two-hop neighbor sets are updated at $t = t_{\text{link}}$; then, the MPR selector sets are subsequently obtained during the interval $[t_{\text{link}_w}, t_{\text{link}_{w+1}}]$ in the following T_s seconds and hello message will be sent at each t_{link_w}, and if no hello message is sent in one hello interval, one hello message will be sent despite t_{link_w} exist or not. If only one t_{link_w} exist, then the link exists during time interval $[0, t_{\text{link}_w}]$ or $[t_{\text{link}_w}, Ts]$ which depending on the $\frac{d(\text{Distant}_{ij}(t))}{dt}_{t=t_{\text{link}_w}}$.

4.3 Routing based on the trajectory information

Another important mechanism of OLSR is route-finding through information transmitted by TC messages. The TC messages in the original OLSR suffer from the outdated problem as stated previously. Original OLSR cannot take a link that does not currently exist into account. The T-OLSR routing process is improved by incorporating the trajectory information of the store-forward mechanism. The combination of the T-OLSR routing process not only considers the link that currently exists but also the future link to determine the route in sparse scenarios.

The OLSR routing process consists of two parts: (1) calculate and obtain all links' starting time and ending time occur during the trajectory coverage time through the trajectory information embedded in the TC message. (the trajectory is assumed to be stable in our study for the next T_s seconds). 2. Find the route through Q-learning with route time sequence validity check.

The T-OLSR routing process incorporates trajectory information by adopting the store-forward mechanism; this process is shown in Fig. 6. Traditionally, the store and

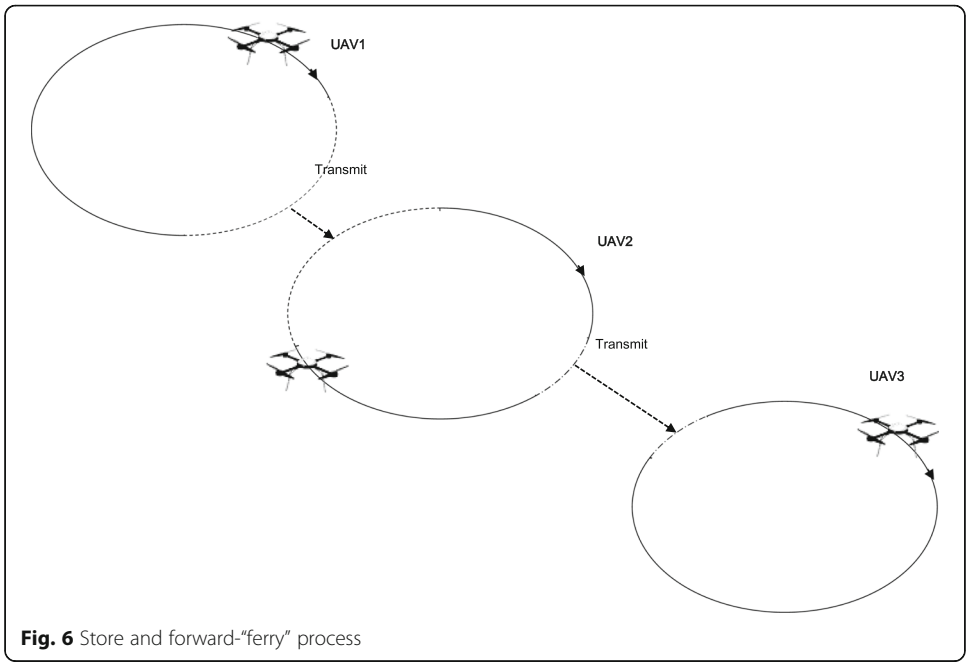

Fig. 6 Store and forward-"ferry" process

forward process is a reactive process—data are stored in the nodes and wait for the next hop to show up without knowing whether next hop will show up. T-OLSR takes all link exist and will establish into account in the route-finding process, and actively send the data to the nodes with knowing when target node will show up.

4.3.1 Calculate and obtain all links

In T-OLSR, the routing process first selects all possible routes according to the trajectory information. T-OLSR considers all links whether they existed now or will be established according to the received trajectory information and can be represented by the equation below:

$$\min\left(D_{ij}(t)\right) < R_T \tag{4}$$

Where the D_{ij} is the Euclidean distant defined in Section 4.2 and the R_T is the transmission radius of the node. Eq. 4 considers all nodes within the transmission radius to establish a link. By solving the equation, we obtain a range of t that could fulfil the condition. Through a similar procedure described in the Section 4.2 to get the $[t_{\text{link}_w}, t_{\text{link}_{w+1}}]$ of each link we could have the $T=\{\gamma_s(i,j),\ \gamma_e(i,j)\}$, where the starting time of the link as the first part and the ending time of the link as the second part. T-OLSR then creates "link starting time" $\gamma_s$and "link ending time" $\gamma_e$matrixes applying such information.

$$\gamma_s = \begin{bmatrix} -1 & 0 & -1 \\ 0 & -1 & 1 \\ -1 & 1 & -1 \end{bmatrix},$$

As per showed the γ_s is the starting time matrix γ_e is the ending time matrix, − 1 in both matrixes stands for link does not exist, 1 in γ_s stands for the link start at 1 s after current time, 1 in γ_e stands for the link end at 1 s after current time. The row 1 and column 2 in γ_s and γ_e stand for the value of the link between node 1 and node 2.

4.3.2 Adopt Q-learning as the route-finding algorithm

Q-learning as a deep learning tool is applied in OLSR to handle route finding problem, and it is efficient in dealing with the route-finding process with T-OLSR's parameters that could continuously update.

Q-learning's equation could be expressed intuitively as

$$Q(state, action) = R(state, action) + Gamma* Max[Q(next\,state,\quad all\ \ action)] \tag{5}$$

where the state stands for the node packet currently stay at, the action stands for the link of node with its neighbor, e.g., state 1 action 3 stands for link between node1 and node3. R(state, action) is R table's element and Q(state, action) is Q table's element. Gamma stands for the learning parameter (taken as 0.8 in our study). We set the reward table (R table) for the route-finding process as follows:

1. Each number in the matrix stands for a link's reward value;
2. Links that do not exist are marked as −1;
3. The R values of other existing links or links to be established are defined as r. Definition of r will be given in following sections Fig. 7.

An example of R table is given in Fig. 7 with values stated above. Q-learning will update Q value through the iteration of Eq. 5 and choose the route from each state's links which have highest Q value according to the final Q table.

4.3.3 Check the validity of the route

In T-OLSR's Q-learning process update the Q value through $\mathrm{Max}[Q(nextstate, all\ \ actions)] = \max[Q(next\ \ state, all\ \ possible\ actions)]$. Instead of finding maximum Q value of one link's next action, T-OLSR getting the maximum Q value of one link's next action which have valid time sequence. During such process, T-OLSR checks the validity of the route—whether the later hop vanished before the former hop before it had the chance to send the packet through.

Such process is curried through looking up the "link starting time" and "link ending time" matrixes and checking if the process's next hop could fulfil below condition

$$\gamma_s(\mathrm{link}_{n+1}) < \gamma_e(\mathrm{link}_n) \tag{6}$$

Where the link_n is current link's next action and link_{n+1} are all possible following actions of current link's next action. Q value of link_{n+1} could not fulfil Eq. 6 will be taken as − 1 and hence not considered in Q value's iteration and hence guarantee Q-learning only consider valid route.

Taking the Fig. 8 and Fig. 9's scenario as example, T values from t = 0001 to t = 0010, the route is from node 2 to node 8. Figure 8 is a valid route, Q value of all links in the route will be updated accordingly through the iteration of Q-learning. Figure 9's route is invalid for its wrong time sequence; the Fig. 9 scenario's Q value iteration process is as follows, T-OLSR try to update the Q value of state 2 action 3 labeled as link 1, its next action is link 2, possible action of link 2 is link 3, link 3, and link 2 cannot fulfil the Eq. 6; hence, Q value of node 3 will not be taken into account during the iteration, the maximum Q value of link 1 either be updated from other route or stay the same. Such process would guarantee the invalid route will not be chosen in the Q-learning's route finding process.

$$\mathbb{R} = \begin{array}{c|cccccc} \frac{State}{Action} & 0 & 1 & 2 & 3 & 4 & 5 \\ 0 & -1 & -1 & -1 & -1 & r & -1 \\ 1 & -1 & -1 & -1 & r & r & -1 \\ 2 & -1 & -1 & -1 & r & r & -1 \\ 3 & -1 & r & r & -1 & -1 & r \\ 4 & r & r & r & -1 & -1 & -1 \\ 5 & -1 & -1 & -1 & r & -1 & -1 \end{array}$$

Fig. 7 An example of R table

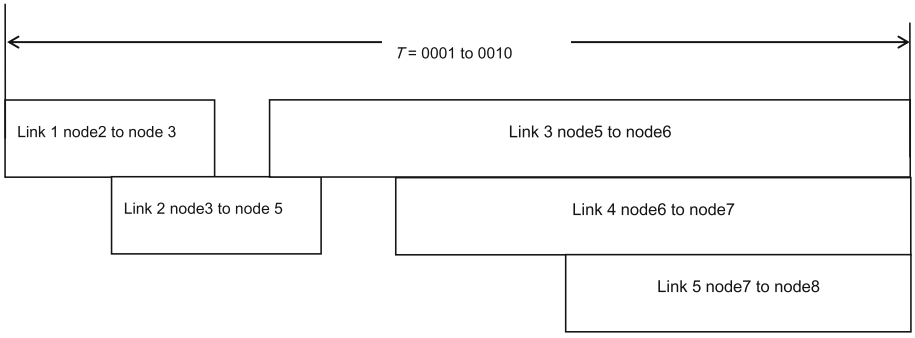

Fig. 8 An example of a link sequence that leads to valid route

4.3.4 Define the r value through route delay and connectivity

Reward Table R's r value is defined by two parts, the connectivity reward value of all links and route delay reward value of all link. Connectivity rewards is set to be

$$\alpha_{ij} = \begin{cases} 0 & \text{normal link between } i, j \\ 1 & i \text{ and } j \text{ is the target node} \\ 0.8 & i \text{ or } j \text{ is the target node} \end{cases}$$

And the route delay reward value is set to be

$$\beta_{ij} = w1(\gamma_e(\mathrm{i},\mathrm{j}) - \gamma_s(\mathrm{i},\mathrm{j})) + (1-w1)\left(1 - \frac{\gamma_s(\mathrm{i},\mathrm{j}) - T0}{Ts}\right) \tag{7}$$

$T0$ is the current time, w1 is a constant value range from [0,1], Hence r is defined as

$$\mathrm{r}_{ij} = w2 \times \alpha_{ij} + (1-w2)\beta_{ij}. \tag{8}$$

Where the $w2$ is a constant value range from [0,1]. β_{ij} part will help Q-learning choose the route established early, last relatively longer which leads to less waiting time while α_{ij} part taken the connectivity into account; adjusting $w1$ and $w2$ will emphasize the connectivity, lasting time or starting time during the route-finding. Together with the validity check and the r value gives above, Q-learning will find more reliable route with less latency for T-OLSR.

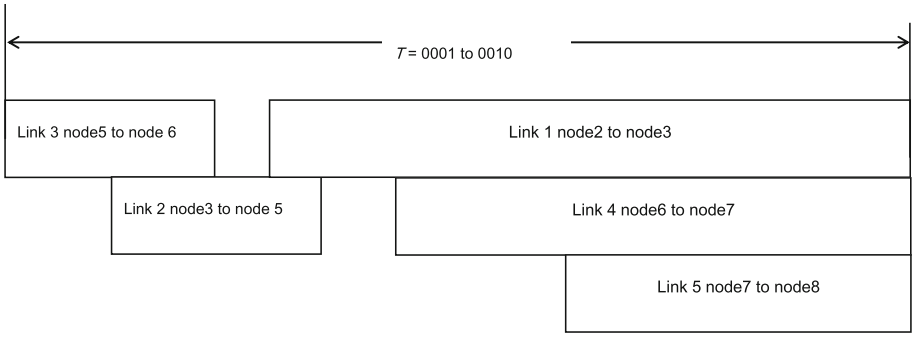

Fig. 9 An example of a link sequence that leads to an invalid route

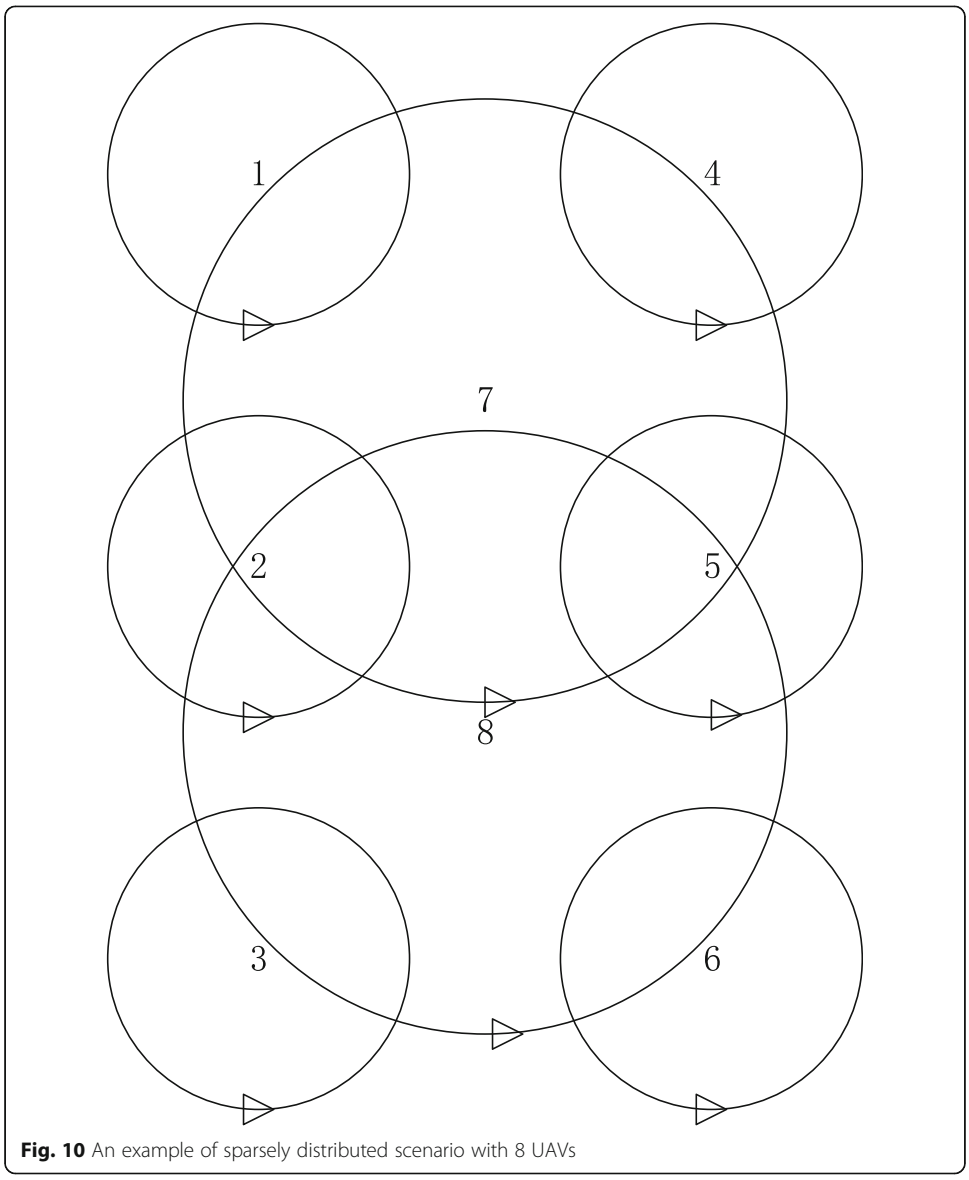

Fig. 10 An example of sparsely distributed scenario with 8 UAVs

4.3.5 Optimization of the image payload packet forwarding sequence by utilizing image progressive transmission

The T-OLSR mechanism we proposed features the ability to store and forward—"ferry" the packet; during this process, the forwarding sequence of the packets requires consideration. As the packets that are transmitted through the network mainly consist of images and the packet loss problem is a critical problem faced by UAVs, an improvement targeting UAV image transmission is proposed to further improve the network's performance. We take JPEG 2000 as the image file type–JPEG2000 is the new international standard for image compression, and it is widely used in various applications, including UAV systems. JPEG 2000 has "progressive transmission" properties—its packets are not arranged through the division of the picture; instead, it allows images to be reconstructed with increasing pixel accuracy or spatial resolution as more packets are received. Hence, under a UAV system application that suffers from the packet loss problem, proper arrangement of the JPEG2000

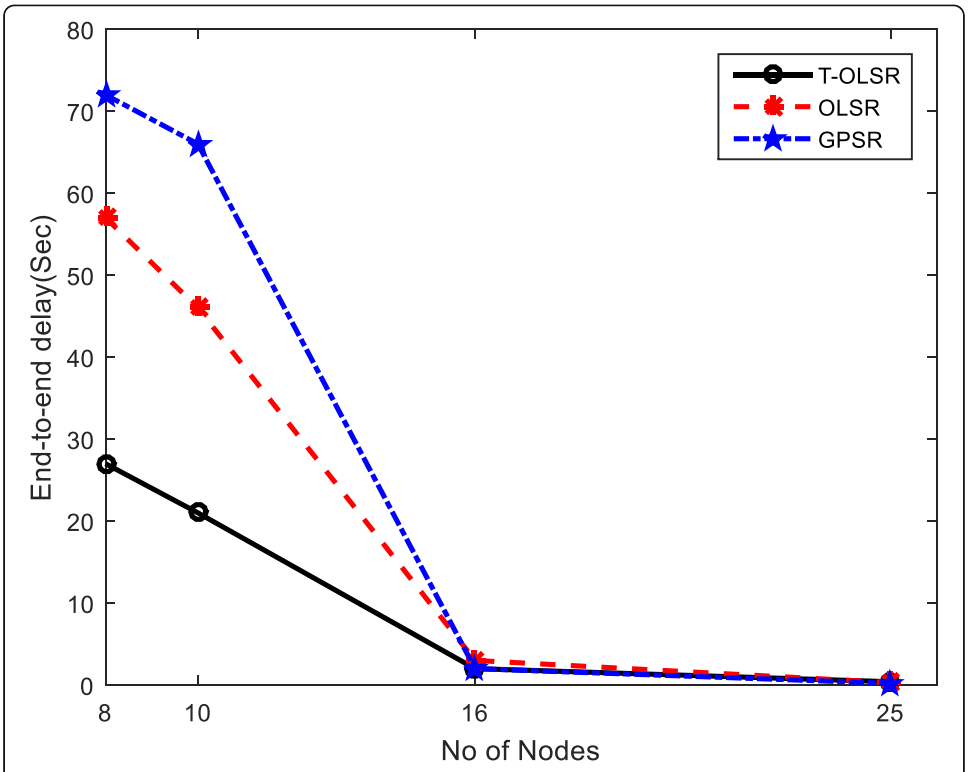

Fig. 11 End-to-end delays performance of T-OLSR ($\mathbf{\Phi}$ = 2 s, τ = 10 s), OLSR, and GPSR

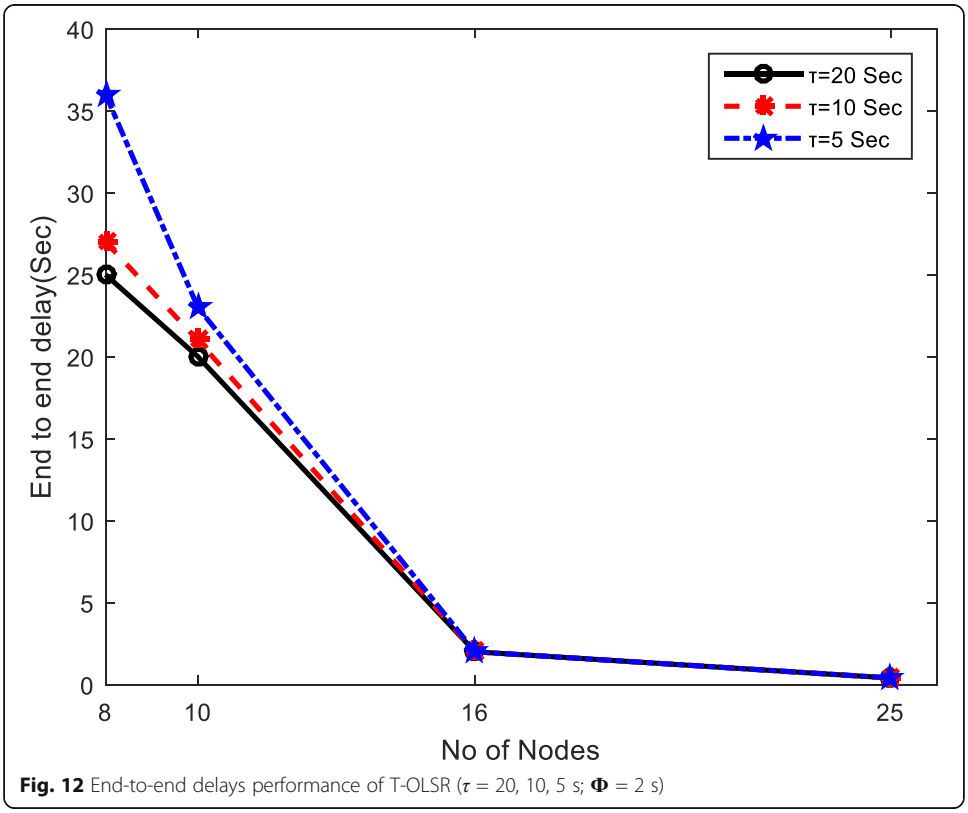

Fig. 12 End-to-end delays performance of T-OLSR (τ = 20, 10, 5 s; $\mathbf{\Phi}$ = 2 s)

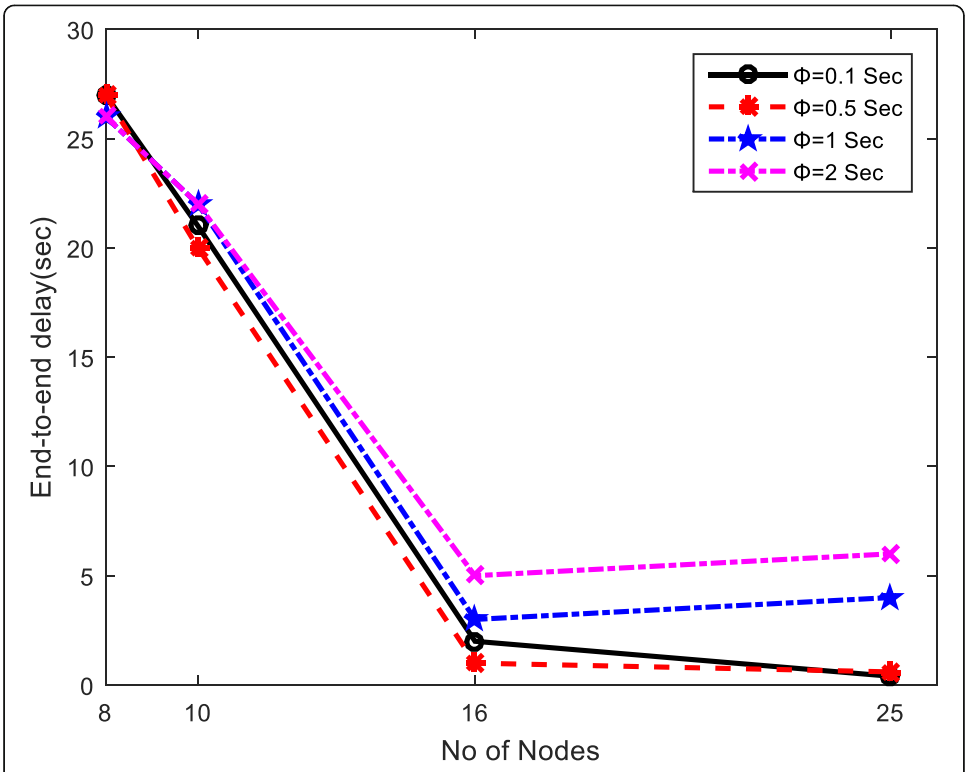

Fig. 13 End-to-end delays performance of T-OLSR ($\mathbf{\Phi}$ =0.1, 0.5, 1, 2 s; τ = 10 s)

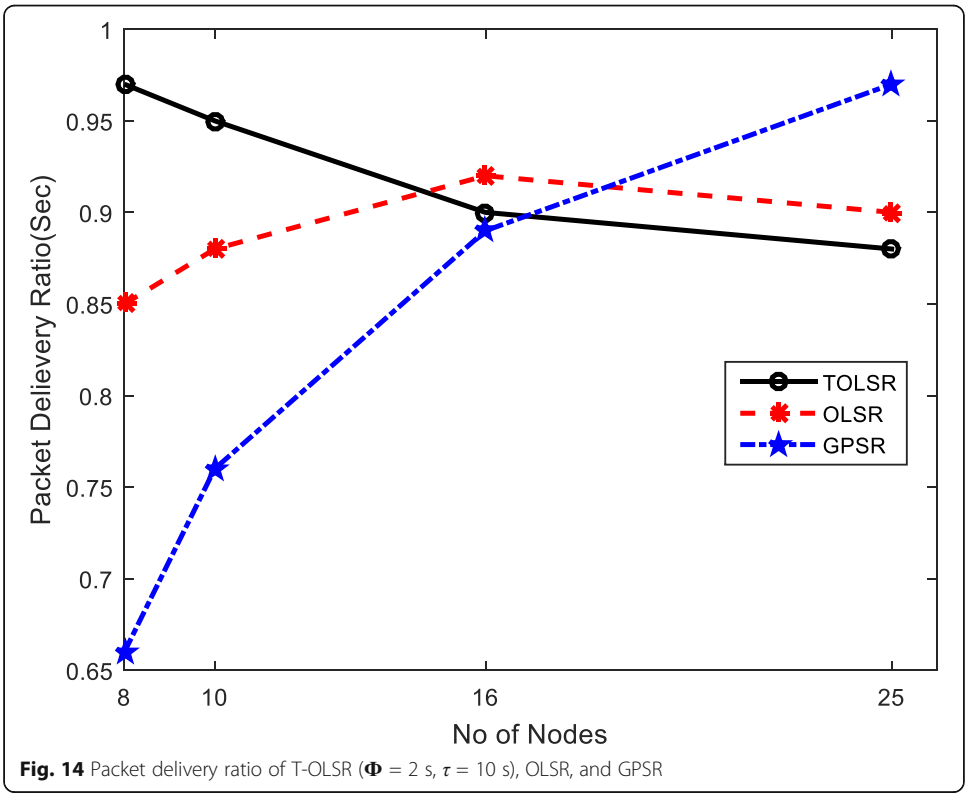

Fig. 14 Packet delivery ratio of T-OLSR ($\mathbf{\Phi}$ = 2 s, τ = 10 s), OLSR, and GPSR

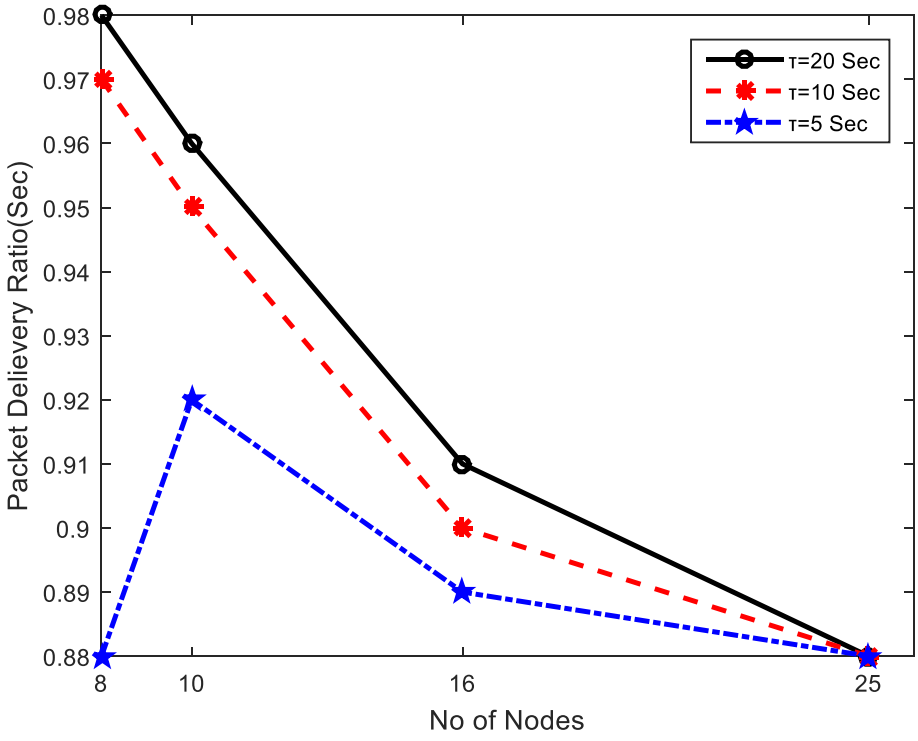

Fig. 15 Packet delivery ratio of T-OLSR (Packet delivery ratio of T-OLSR (τ = 20, 10, 5 s; Φ = 2 s))

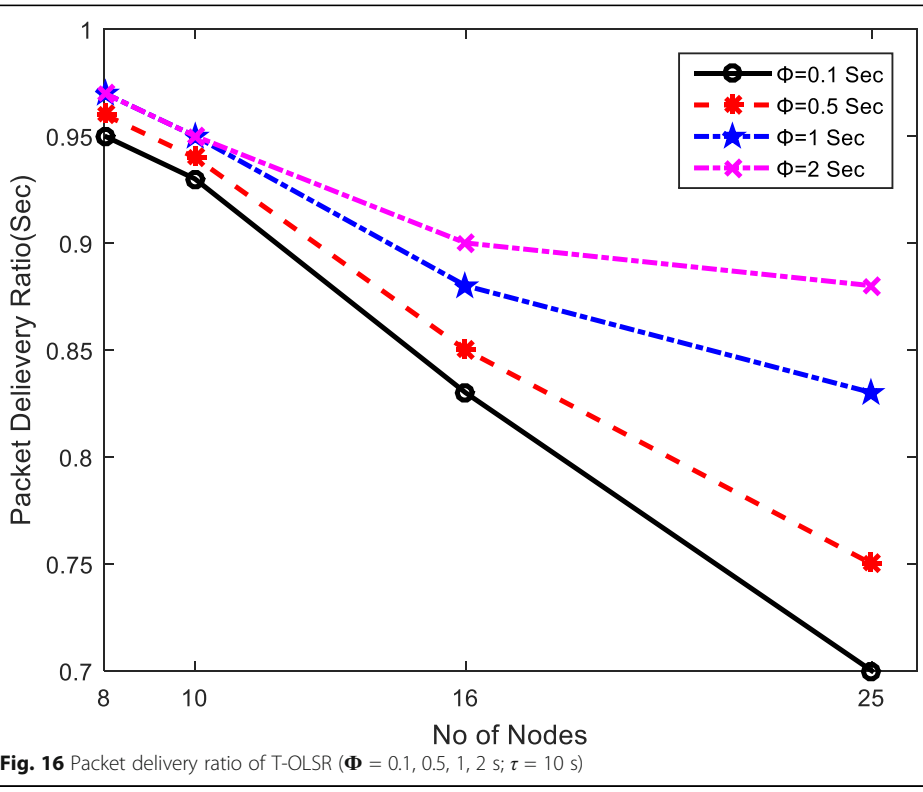

Fig. 16 Packet delivery ratio of T-OLSR (Φ = 0.1, 0.5, 1, 2 s; τ = 10 s)

image payload packet could greatly improve the image quality. With the priority of the critical image payload in packet forwarding, the image quality can be improved greatly.

As stated in the standard, the header part of the image is important; it depicts the outline of the picture, and the remaining part improves the details of the picture [20, 21]. From the simulation of several pictures, we found that with 5% error in the first 45% of the picture, the PSNR is less than 25 dB, and in the last 55% of the picture, the PSNR is much higher than 25 dB. However, the first 300 bytes are critical, and a 5% error in the first 300 bytes would make the reconstruction of the image impossible. Hence, regarding the image files with the format JPEG2000, we divide all packets into three categories: 1—critical part, first 300 byte of the image; 2—important part, first 40% of the image; 3—other part, other parts of the image. When forwarding the image payload's packets, the queue of sending packets is arranged to be first category 1, then category 2, then category 3, and in each category, the packets follow the first in-first out (FIFO) order. This mechanism assures that the important packets are transmitted first, and when facing serious packet loss situations, the image can still be reconstructed to the greatest extent.

5 Simulation experiment

5.1 Simulation parameters

We used MATLAB to perform the simulation, taking a 3 km × 6 km area as the search and rescue mission coverage area, and every plane followed a circle to perform the scanning. Eight planes are involved in the mission. Following the trajectory shown in the Fig. 10, the planes travel at different speeds, varying from 50 m/s to 90 m/s. Each plane travels in a circle, the smallest circle has the same radius as the transmission radius, and planes 7 and 8 travel in a large circle with a radius double the transmission radius. Planes 1 and 2 and planes 2 and 3 can communicate with each other when close enough, planes 4 and 5 and planes 5 and 6 can communicate with each other when close enough; plane 7 can communicate with planes 1, 2, 4, 5 and 8 at certain times, and plane 8 can communicate with planes 2, 3, 5, 6 and 7. The whole setup follows the scenario given in Section 3.1, with several UAVs cooperatively searching the given area.

Such a setup emphasizes the sparse problem: one plane cannot directly send messages to other nodes, and most of the nodes need to wait for other nodes to "ferry" the packet, and the nodes need to store and forward the packet in order to transit the packet.

We assume that all planes need to send pictures to one other, the throughput rate is 10 Mbps. The transmission radius is set to 500 m. To see the node density's effect on the result, we add nodes randomly to the original scenario to represent an increased node density. When node is added to the network, it was randomly dropped into the search area and their trajectories are same as node1–6. $w1$ and $w2$ of Eq. 7 and Eq. 8 is set to be 0.8 and 0.6 according to simulation trails.

5.2 Result analysis

5.2.1 End-to-end latencies

End-to-end delay is the time a packet takes to be transmitted from the source to destination; it reflects the delay of the network and is a direct indicator of the network's performance. In our simulation, we compared T-OLSR with OLSR, and GPSR under

different choices of hello intervals (**Φ**), trajectory information times (**τ**), and different numbers of nodes.

From the results in Figs. 11, 12 and 13, we can find that T-OLSR with **Φ** = 2 s and **τ** = 20 s provides the best result for the sparse scenario. For the trajectory information with **τ** = 20 s in the described scenario where the planes move in circles, a longer trajectory length did not cause an increase in overhead and hence provided a better result; however, as the "store-forward" waiting time is the main factor affecting the overall latency, and the **τ** = 10 s information is already enough to cover most of the topology changes; hence, the **τ** = 20 s and **τ** = 10 s results are quite similar regarding the end-to-end delay. The **Φ** = 2 s result appears to be better overall than shorter **Φ** ; for a certain path, longer **Φ** appear to be better, especially when more nodes involved. T-OLSR with **Φ** = 2 s and **τ** = 10 s shows better results when the node density is low, and this result is similar to GPSR and OLSR when the node density increases.

5.2.2 Packet delivery ratio

The packet delivery ratio refers to how many packets can successfully arrive at the destination; in a sparse scenario, it could effectively evaluate the network's performance.

The results in Figs. 14, 15 and 16 reveal that T-OLSR with **Φ** = 2 s and **τ** = 20 s gives the best result. A shorter **Φ** interval did not have a significant impact on the packet delivery ratio in the sparse scenario because the overhead effect was not critical, but with more nodes, a smaller **Φ** interval causes the network's performance to deteriorate rapidly. The trajectory length greatly affects the packet delivery ratio, a **τ** = 10 s trajectory information duration gives a better result than 5 s, the τ = 20 s trajectory result is similar to the **τ** = 10 s duration trajectory, as the information is already enough to counter most of the topology changes under the simulated scenario. Comparing T-OLSR with **Φ** = 2 s and **τ** = 10 s with GPSR and OLSR, T-OLSR outperforms both when the scenario is sparse and shows similar results when nodes are dense.

6 Conclusion

In conclusion, we propose a protocol that aims to solve common issues that arise when utilizing multiple cooperating UAVs under a sparsely distributed network while considering image quality. Judging from the end-to-end delay and packet delivery ratio, T-OLSR has demonstrated that it exhibits better performance when used in coordinated search and rescue missions with multiple UAVs in sparse distribution scenarios. Additionally, T-OLSR has the "store and forward" ability that enables UAVs to address the sparse scenario's link outage problem with more options. With *Q*-learning, T-OLSR takes multiple factors into account during the route-finding process, which further improves the system's performance. During the simulation, T-OLSR shows a significant improvement in the packet delivery ratio with an increase over 30.2% when compared to GPSR in 8-node sparsely distributed scenarios and the end-to-end delay result, as well as with a reduction in the delay up to 46.3 s for end-to-end delays compared with GPSR and 32.7 s compared with OLSR, depending on the scenario. Utilizing the image payload packet forwarding mechanism, the image quality shows significant improvement with same packet loss ratio. Hence, in scenarios where the trajectory is a known factor, T-OLSR outperforms all traditional ad hoc protocols. However, our study only covers the simplest trajectory patterns

and the plane didn't change its trajectory during movement, which may not be sufficient in real-world search and rescue missions. Additionally, the trajectory transmission method used in this paper is still elementary, and more sophisticated coding could help to reduce the overhead of the T-OLSR. Moreover, such trajectory information could be applied to various protocols to improve their performance. With further development of AI and coordinated UAV control systems, we have little doubt that protocols that utilize trajectories predetermined by mission control software will outperform traditional protocols.

Abbreviations
UAV: unmanned aerial vehicles; FANET: flying ad hoc network; OLSR: optimized link state routing; AODV: ad hoc on-demand distance vector routing; DSDV: destination-sequenced distance vector routing; GPSR: greedy perimeter stateless routing; TC: topology control; T-OLSR: trajectory-OLSR; MPR: multipoint relay

Acknowledgements
Special thanks to IEEE Life Fellow Trieu-Kien Truong's valuable comments.

Authors' contributions
Chen HOU is the main author of the paper. Zhexin Xu contributed to the design of the study, theory, result analysis, and article writing. Chen HOU carried out the experimental work and the data collection and interpretation. Wen-Kang Jia finished the analysis and interpretation of the data. Jianyong Cai conceived and designed the experiments, and Hui Li contributed to the development of the ideas and revised the paper. All authors read and approved the final manuscript.

Funding
This work was supported by the National Science Foundation of China (grant no. 61675043).

Availability of data and materials
Data sharing was not applicable to this article, as no data sets were generated or analyzed during the current study.

Ethics approval and consent to participate
Not applicable

Consent for publication
The authors all agree on the publication of the paper.

Competing interests
The authors declare that they have no competing interests.

Received: 27 January 2020 Accepted: 15 April 2020
Published online: 02 July 2020

References

1. Y. Fang, P. Chen, G. Cai, F. C. M. Lau, S. C. Liew, G. Han, Outage-limit-approaching channel coding for future wireless communications: Root-protograph low-density parity-check codes," IEEE Vehicular Technology Magazine **13**(2), 85–93 (2019).
2. P. Chen, Z. Xie, Y. Fang*, Z. Chen, S. Mumtaz, J. Rodrigues, Physical-layer network coding: An efficient technique for wireless communications," IEEE Network Magazine, **99**, 1–7 (2019).
3. K. Singh, A.K. Verma, *Experimental analysis of AODV, DSDV and OLSR routing protocol for flying adhoc networks (FANETs). IEEE International Conference on Electrical* (Computer and Communication Technologies, IEEE, 2015)
4. M.T. Hyland, B.E. Mullins, R.O. Baldwin, & M.A. Temple, Simulation-based performance evaluation of mobile ad hoc routing protocols in a swarm of unmanned aerial vehicles. International Conference on Advanced Information NETWORKING and Applications Workshops .Vol.2 (IEEE, 2007), pp.249-256.
5. Y. Ge, T. Kunz, & L. Lamont, Qualityof service routing in ad-hoc networks using OLSR. Hawaii International Conference on System Sciences, (IEEE Computer Society, 2003), pp.300.2.
6. H. Badis, K. Al Agha, Qolsr, qos routing for ad hoc wireless networks using olsr. Transactions on Emerging Telecommunications Technologies **16**(5), 427–442 (2005)
7. Härri, J., Filali, F., & Bonnet, C. (2012). Performance testing of olsr using mobility predictions. Eurecom.
8. H. Menouar, M. Lenardi, & F. Filali, Improving proactive routing in VANETs with the MOPR movement prediction framework. Telecommunications, 2007. Itst '07. International Conference on ITS, (IEEE, 2007), pp.1-6.
9. S. Sharma, P-OLSR: Position-based optimized link state routing for mobile ad hoc networks. Local Computer Networks, 2009. LCN 2009. IEEE, Conference on, (IEEE, 2009), pp.237-240.
10. Mi, Z., Yu, W., Niu, D., & Dong, C. (2012). The position-aware routing protocol for pre-handoff on OLSR in vehicular networks. Second International Conference on Intelligent System Design and Engineering Application .pp.1156-1159. IEEE Computer Society.
11. 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.
12. 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.

13. 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.
14. S. Rosati, K. Kruželecki, G. Heitz, D. Floreano, B. Rimoldi, Dynamic routing for flying ad hoc networks. IEEE Transactions on Vehicular Technology **65**(3), 1690–1700 (2016)
15. C. Pu, Link-quality and traffic-load aware routing for UAV ad hoc networks. 2018 IEEE 4th International Conference on Collaboration and Internet Computing (CIC), (IEEE, 2018).
16. C. Li, L. Zheng, W. Xie, & P. Yang, Ad hoc network routing protocol based on location and neighbor sensing. 2018 IEEE International Conference on Computer and Communication Engineering Technology (CCET), (IEEE, 2018).
17. L. Evers, A.I. Barros, H. Monsuur, & A. Wagelmans, Uav mission planning: from robust to agile. , 56, 1-17 (2015).
18. X. Li, J. Chen, & J. Li, FATES: a framework with adaptive track-explore strategy for moving targets search by a FANET. 2018 IEEE Intl Conf on Parallel & Distributed Processing with Applications, Ubiquitous Computing & Communications, Big Data & Cloud Computing, Social Computing & Networking, Sustainable Computing & Communications (ISPA/IUCC/BDCloud/SocialCom/SustainCom), (IEEE, 2018).
19. E. Larsen, J. Flathagen, V. Pham, F. Norway, & L. Landmark, Iolsr: olsr for wsns using dynamically adaptive intervals. 18-23 (2011).
20. A.N. Skodras, C.A. Christopoulos, & T. Ebrahimi, Jpeg2000: the upcoming still image compression standard. Pattern Recognition Letters, 22(12), 1337-1345 (2001).
21. C. Christopoulos, A. Skodras, & T. Ebrahimi, The jpeg2000 still image coding system: an overview. IEEE Transactions on Consumer Electronics, 46(4), 0-1127 (2000).

Publisher's Note