

Received October 19, 2020, accepted October 29, 2020, date of publication December 9, 2020, date of current version January 5, 2021.

Digital Object Identifier 10.1109/ACCESS.2020.3043672

# Skeleton-Based Swarm Routing (SSR): Intelligent Smooth Routing for Dynamic UAV Networks

NILOOFAR TOORCHI<sup>1</sup>, (Member, IEEE), FEI HU<sup>ID1</sup>, (Member, IEEE), ELIZABETH SERENA BENTLEY<sup>2</sup>, (Member, IEEE), AND SUNIL KUMAR<sup>ID3</sup>, (Senior Member, IEEE)

<sup>1</sup>Department of Electrical and Computer Engineering, The University of Alabama, Tuscaloosa, AL 35487, USA

<sup>2</sup>Air Force Research Laboratory, Rome, NY 13441, USA

<sup>3</sup>Electrical and Computer Engineering Department, San Diego State University, San Diego, CA 92182, USA

Corresponding author: Fei Hu (fei@eng.ua.edu)

This work was supported by the Air Force Research Laboratory (AFRL) under Contract FA8750-18-1-0023. Distribution A. Approved for public release: Distribution unlimited; AFRL-2020-0517 on 15 Dec 2020.

**ABSTRACT** A swarm of unmanned aerial vehicles (UAVs) requires the transmission of mission-related data across the network. The resource constraints and dynamic nature of the swarm bring critical challenges to the design of UAV routing protocols. Most of the conventional ad hoc routing schemes are not intelligent and cannot adapt to the dynamic nature of UAV swarming networks. On the other hand, some artificial intelligence (AI)-based routing schemes may consume significant computational resources in the UAVs. In this article, a low-cost, adaptive routing protocol, namely *skeleton-based swarm routing (SSR)*, is proposed, which exploits an intelligent online learning algorithm and the topology features of the mission-driven UAV swarm to distribute the traffic over optimal routes. Here, the *skeleton* represents the most stable parts of the swarm formation. *SSR* architecture consists of three modules: 1) A geometric addressing module, which assigns geometric coordinates to each node based on the swarm skeleton structure; 2) A leaf-like routing pipe which allows the selection of multiple candidate routes around the shortest path; 3) An intelligent low-complexity learning model which determines how to distribute the packets inside the routing pipe to achieve load-balanced, high-throughput transmissions. The proposed skeleton-based scheme can also facilitate the UAV formation construction and morphing. The simulation results show that the proposed *SSR* protocol can noticeably improve the network performance (up to 100% throughput improvement) compared to the single path routing schemes, such as the ad-hoc on-demand distance vector (AODV) and link-quality and traffic-load aware optimized link state routing (LTA-OLSR) protocols.

**INDEX TERMS** Geometric routing, quality of service, reinforcement learning, stochastic dynamic programming, swarm networks, UAV communication.

## I. INTRODUCTION

The airborne networks composed of unmanned aerial vehicles (UAVs), as illustrated in Fig. 1, have been deployed in different civilian, commercial and military applications, such as disaster management, border surveillance, search and rescue operations, goods delivery, etc. [1], [2]. In such networks, it is often needed to transmit data (e.g., high-resolution surveillance videos) among UAVs or to the control station. Hence, establishing reliable end-to-end paths among UAVs is critical for many applications, which demand high quality-of-service (QoS).

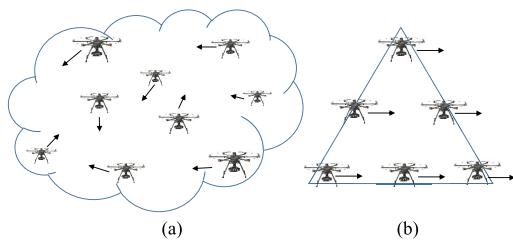
The associate editor coordinating the review of this manuscript and approving it for publication was Muhammad Maaz Rehan<sup>ID</sup>.



**FIGURE 1.** UAV swarm network in civilian applications.

For effective coordination and collaboration, the UAVs usually communicate in an ad hoc fashion and form flying ad hoc network (FANET). A subset of UAVs may link up

with the control station. Based on the degree of coordination among UAVs, FANETs can have different application architecture. For instance, the swarm cooperation architecture (Fig. 2(a)) requires a lower coordination among UAVs to accomplish tasks such as the target search. On the other hand, physically-linked architecture (e.g., lifting and transportation of objects) and mission-driven (formation-based) architecture (Fig. 2(b)) require a higher degree of coordination and collaboration [2], [3]. This article focuses on *mission-driven* swarm networks where certain network structure is desired. Based on the mission requirements, UAVs are often deployed in a certain spatial *formation* and the whole network topology may change from one formation to another (called *swarm morphing* hereafter). The formation control can be done through different methods such as leader-follower [4]. Since the UAVs have limited resources in terms of power, memory, computations, etc. Hence, the low complexity protocols should be designed for swarm networks.



**FIGURE 2.** Different swarm architectures: (a) swarm cooperation, (b) mission-driven formation.

In this article, an intelligent, high-throughput routing scheme is designed that adapts to the dynamics of swarm formation. In general, the proposed scheme is distributed and can be used to improve different QoS metrics (such as throughput, delay, load-balancing, etc.) as well as network life time. It exploits the swarm structure (formation) to reduce the complexity of smart routing. Many existing routing schemes typically search the shortest path and cannot adapt to the network dynamics. On the other hand, the centralized solutions and routing schemes, which rely on the frequent update of link-state information to construct the topology database, introduce a high complexity and overhead, which is not suitable for resource-limited swarm networks.

To add intelligence and adaptivity to network applications, the artificial intelligence (AI) techniques, such as deep learning (DL) [5] and deep reinforcement learning (DRL) [6], are becoming increasingly popular. Despite the effectiveness of these techniques, many AI schemes may not be practically applicable to the resource-limited UAV swarm networks in which powerful computing platforms are not accessible and energy consumption is a critical issue.

The routing scheme proposed in this article uses a special *geometric addressing* system to identify the roles and location of different nodes based on the swarm structure, in order to reduce the routing overhead and latency. In fact, the mission-driven swarm networks, which follow a specific formation,

can be represented by a framework, called *skeleton* in this article. Here, the skeleton refers to the main structure of the swarm which consists of several relatively stable nodes, called bones. The nodes located in the outer area of the swarm typically have higher mobility than the core/inner area nodes. In terms of the node addressing model, here the term “geometric” is preferred to “geographic”, since the geographic coordinates of the nodes may not be accessible. Note that the geometric address represents the area where the node is located in.

The proposed routing protocol is *hybrid*: 1) *Geometric* forwarding: Packets are forwarded to the area that the node is expected to reach based on a greedy forwarding scheme. 2) *Reactive* search: A local search is conducted to find the destination’s location. Here it is assumed that the UAV formation’s shape changes smoothly based on the mission requirements. We call such a process as *swarm morphing*, similar to the concept of image/polygon morphing in the field of computer vision.

Particularly, a distributed dynamic-programming-based online routing scheme, called *skeleton-based swarm routing (SSR)*, is proposed which uses a leaf-like routing *pipe* to transmit the packets through the nodes that experience less traffic load (thus have a lower probability of getting congested) and hence, improves QoS. The proposed routing scheme is flexible and can be used to improve other networking metrics, such as network lifetime, when QoS requirements are not tight. To the best of our knowledge, it is the first work that benefits from the geometric addressing derived from the UAV swarm structure in order to improve the routing procedure. The main contributions of the paper are as follows:

- **Swarm skeleton-based geometric addressing:** A novel geometric addressing model is designed based on the swarm skeleton structure, which represents the UAV’s approximate location. A formation morphing strategy is implemented to guide the node into the position with the minimum impact on its geometric address, when the entire network changes the formation.
- **Adaptive pipe routing:** A leaf-like routing pipe is constructed according to the addressing model. The pipe serves as the main framework of the routing scheme and can adapt to the changes of the network skeleton structure.
- **Dynamical-programming-based route optimization:** A novel distributed, low-cost, intelligent routing protocol is proposed to achieve the high-throughput and load-balanced data forwarding inside the pipe.

The rest of the paper is organized as follows: Section II presents an overview of the related work. Section III provides the system assumptions and briefly explains different components of the proposed routing scheme. The procedure for developing a novel geometric addressing model and constructing the leaf-like pipe, is discussed in Section IV. Section V provides the swarm morphing procedure based on the star-like skeleton structure. In Section VI, the details of

*SSR* that can intelligently route data inside the pipe are thoroughly discussed. To demonstrate the high QoS performance of the *SSR* scheme, extensive simulation results are presented in Section VII. Section VIII analyzes the *SSR* overhead, followed by the conclusions in Section IX.

## II. RELATED WORK

UAV swarm network has specific characteristics, in terms of mobility pattern, computation power, energy consumption and radio propagation, which make it different from other types of ad hoc networks. Due to the increasing popularity, many studies have been conducted on the suitable protocols and the corresponding challenges of such networks.

In general, the routing schemes can be classified into five categories: 1) *Reactive* routing: These schemes, such as Ad-hoc On-demand Distance Vector (AODV) [7], are on-demand and mainly designed for mobile ad-hoc networks. For route discovery, they rely on flooding the control messages throughout the network which results in extra overhead and latency. 2) *Proactive* routing: These protocols are based on routing tables that are regularly updated (even when there is no data to transmit) and, hence are faster. However, frequent update of the link-state information throughout the network causes high overhead [2]. 3) *Geographic/geometric* routing: These schemes are sometimes considered as proactive routing as they do not perform the initial route discovery phase. However, they do not need the periodic update of the routing tables and the information about the entire network link states. Instead, they depend on the geographic location of the nodes for greedy (distance-based) forwarding. As the geographic locations of nodes may not be always known, the recent routing designs on this topic focus on virtual coordinates and naming approaches [8]–[10]. 4) *Store-carry-forward*: This category is more applicable to sparse and mobile networks where nodes can move in order to deliver data packets. This solution is mainly suitable for centralized and delay tolerant applications [2]. 5) *Hybrid*: Hybrid routing protocols combine the attributes of the other categories to better adjust to the network features [11].

The integration of software defined networking (SDN) with UAV swarm network is investigated in [12] where the communication and routing policies are managed by the SDN controller. A QoS-based disjoint multi-path routing scheme based on distributed SDN architecture is studied in [13], which can exclude the energy-exhausted UAVs and re-select new RF links, if some links are broken. In [14], a centralized traffic-differentiated routing protocol is proposed for SDN-based hierarchical FANET architecture aiming to meet specific QoS requirements, where each UAV cluster is controlled by an upper stationary airship. A centralized SDN-based topology and routing management scheme is introduced in [15], where the controller positions the relay nodes to optimize the link availability and constructs the routing table for each node based on the length of the links. The deployment of SDN requires the collection of network information and the use of specific network infrastructure.

An autonomous flocking control scheme is proposed in [16] to maintain the hierarchical network structure. Due to the high cost of deploying GPS and the possibility of losing GPS signals, it uses the received signal strength (RSS) to estimate the distance. The work in [17] solves the problem of communication and control of a triangular swarm of three cellular-connected UAVs and mathematically derives the reliability of the wireless system in terms of meeting the delay requirements. [18] investigates the effect of UAVs sharing the same spectrum with the uplink of cellular users. The result shows that the presence of UAV links may slightly degrade the cellular users' uplink performance. The quality of UAV links as well as the user links degrades as UAVs fly higher, due to the possibility of larger line-of-sight interference.

Some recent studies on swarm networks have focused on new routing schemes since the existing ones for ad hoc or sensor networks may not be sufficiently mobility-adaptive, communication and computation efficient, or supportive of UAV-to-ground-station communications. It is mentioned in [19] that nodes can estimate the time and energy consumption for data transmission in each path by accessing the position information. They construct a weighted directed graph for UAV cluster architecture. Based on the graph analysis, the optimal relay path can be found through Bellman-Ford algorithm. An adaptive scheme is proposed in [20] which dynamically adjusts the HELLO packet interval and the timeout timer based on the swarm mission information and the network condition, in order to minimize the energy consumption in FANET routing schemes.

An enhanced version of AODV, called robust and adaptive reliable predictive (RARP) scheme [21], is proposed for UAV networks which combines the omnidirectional and directional transmissions, and uses a modified RREQ format that includes sender's trajectory information (in 3D), minimum expected connection time of the path and maximum of nodes' failure probability. The destination waits for a specific time to receive several RREQs and selects the path based on a utility function which is a weighted sum of the metrics in RREQ and the hop count to the destination. The authors in [22] propose the PSO-GLFR protocol, which improves greedy forwarding routing (GFR) using particle swarm optimization (PSO) and limited flooding. Besides the distance factor, the PSO-GLFR protocol considers the number of neighbors and deflection angle to find the next forwarding relay.

Some studies have extended OLSR [23] in order to adapt it for FANET. To address the high-mobility of UAVs, [24] weights the expected transmission count (ETX) metric based on the relative speed between the nodes using the GPS information. In [25], a mobility and load aware OLSR (ML-OLSR) is proposed which assigns a stability degree to the links based on the statistical information of the distance. Moreover, a load factor is calculated using the buffer load of the node and its neighbors, which avoids selecting the congested paths in the path selection phase. In a similar approach, a link-quality and traffic-load aware OLSR (LTA-OLSR) protocol is proposed in [26], in which the

statistical information of received signal strength is used to find the link quality. The traffic load of each UAV is obtained using the node's buffer load and the channel utilization (which is an indication of neighbors' traffic load). Despite the effectiveness of such proactive schemes, the need for frequently updating the topology state introduces high overhead, and the computational complexity of path selection algorithm may be a hindrance, especially in large networks of battery powered UAVs [2].

In swarm networks with sparse density, the store-carry-forward technique is a popular solution that benefits from the node mobility to forward packets to suitable nodes. The authors in [27] propose the location-aided delay tolerant routing (LADTR) protocol, which combines geographic forwarding with store-carry-forward strategy to improve the availability of paths between searching UAVs in post-disaster operations. The GPS information is used to estimate the future locations of UAVs. To create a reliable end-to-end communication in fast and random flying UAV networks, [28] uses the queue backlog information along with the geographic information of the nodes to select the next forwarding relay in the store-carry-forward method. Moreover, the RaptorQ code technique is used to further reduce the packet loss rate. Although these routing schemes improve the delivery ratio, they are more suitable for delay-tolerant applications due to the latency caused by store and carry procedure.

A relatively lightweight stochastic packet forwarding scheme is proposed in [29], where the forwarding probability to each neighbor is found using the link throughput and expiration time. These two metrics are estimated based on geographic coordination and moving information propagated via HELLO packets. The stochastic forwarding can distribute the packets among several forwarders based on their weights and hence, reduces the interference among nodes compared to single path routing. Distributing packets over a routing "pipe" (instead of a routing path) is another approach to mitigate the inter-node interference, which is investigated in [30] using multi-beam directional antenna (MBDA). However, the use of MBDA may be costly in large-scale UAVs networks.

With popularity of online learning and AI algorithms, some researchers have used such techniques in swarm networks. For example, in [31] an online reinforcement learning (RL) scheme is used for transferring time-insensitive packets in sparse networks, where UAVs help to transfer data. Inspired by geographical routing, it defines a reward function and learns whether to relay data among UAVs or move UAVs to a new location. Although several studies have been conducted to improve the convergence speed of RL algorithms [32], they still face the convergence problem in large state space. The use of DRL has shown more promising results for large state and action sets. The deep Q-learning technique is used in [33] in an airborne network composed of powerful airplanes at the higher layer and high-density UAV swarm at the lower layer. The scheme adjusts the locations of some UAVs to make up for the broken RF links. However, due to its

high computation complexity, DRL may not be applicable in resource-constrained UAV networks. Table 1 compares some of the major routing protocols, in terms of routing category, the metrics being used, load balancing, communication overhead and computation complexity.

**TABLE 1. Routing protocols comparison for UAV network.**

Reference	Routing strategy	Metrics	Load bal.	comm. overh.	compu. compl.
STFANET [15]	centralized SDN	length of each link	×	High	High
RARP [21]	reactive	# hop, failure prob., connection time	×	High	Low
PSO-GLFR [22]	geographic, reactive	distance, # neighbors, deflection angle	×	Low	High
P-OLSR [24]	proactive	relative speed-weighted ETX	×	High	High
ML-OLSR [25]	proactive	stability degree of node, buffer load	✓	High	High
LTA-OLSR [26]	proactive	link quality, traffic load	✓	High	High
LADTR [27]	geographic, SCF	position information	×	Low	High
FGQPA [28]	SCF	queue backlog, position information	✓	High	High
SPA [29]	stochastic forward.	link throughput, link expiration time	×	Low	Low

SCF: store-carry-forward.

The scheme proposed in this article is distributed and avoids the overhead and complexity of the centralized solutions such as SDN and AI-based approaches, while adapting to the network conditions. It uses the geometric forwarding and hence, does not rely on the geographic coordinates (which may be sometimes unavailable). To the best of our knowledge, it is the first work that benefits from the geometric addressing derived from the UAV swarm structure. Compared to [29], SSR forwards data to only a subset of neighbors in the direction of the destination (according to the geometric addressing) instead of all the neighbors, so that the data is not diverted from the desired trend in large networks.

SSR does not perform route discovery and thus, avoids the latency and overhead of flooding RREQs. Moreover, it dispatches data through a routing pipe which significantly improves the throughput. Similar to [25], [26], it is a load-balanced routing scheme, but does not require frequent update of the link state information required by proactive approaches. However, it requires the distribution of the updated geometric address table, which is not very frequent in the formation-based UAV network, considered in this article.

### III. SYSTEM MODEL

In this section, a swarm network model is described and the system assumptions for the morphing and routing schemes are provided. The UAVs are interchangeably referred to as the network "nodes" in the paper.

## A. NETWORK ASSUMPTIONS

### 1) NODE RESOURCES

As the GPS data may not be available at some nodes, the proposed scheme does not rely on the exact geographic location information. To stay in a proper relative position, the nodes estimate the angle and distance of their neighbors by using inexpensive equipment, such as compass and sensors. The inter-node distance can also be estimated via RSS-based methods. Nodes have limited power and computational resources and are thus not able to execute complex algorithms. Simple omnidirectional antennas are used here and nodes' time clocks are synchronized.

### 2) SWARM TOPOLOGY

Swarm nodes fly in a 2D environment. One of the nodes is assumed to be the swarm leader, which is pre-selected and fixed in this article. The leader can access larger resources. However, in order to make the scheme applicable to networks with limited resources, it is assumed that only the leader has access to the mission commands. The leader may decide about the swarm topology itself or receive the command from another entity, e.g., the control station.

The information about the new structure can be represented as the basic information about the skeleton, such as the length and angle of the bones and is transmitted by the leader to the nearby bone nodes. The swarm acts in a distributed manner and each node guides its child node to the proper position (parent-child formation control strategy). Hence, the need for the geographic information (e.g. GPS data) and the trajectory of all the swarm nodes is eliminated. This reduces the complexity of the formation control algorithm, specially in large swarms, and improves the scalability and the flexibility of the network.

### 3) NODE MOBILITY

The UAVs are assumed to move at a speed of 10~50m/s. In mission-driven swarm networks, the UAVs cooperate with each other to fulfill a mission. However, the mission-driven formation doesn't mean that all the nodes have a pre-determined trajectory and all the locations are pre-known and exact. Nodes can still move freely in their proximity, but they need to maintain the overall topology and, therefore, an approximate swarm "framework" should be kept (called skeleton in this article). When a new swarm mission command is received, nodes may move together towards a target area or locally to properly fill a region. They may also move to help with the route establishment. Hence, the network dynamics can be described by a topology prediction model.

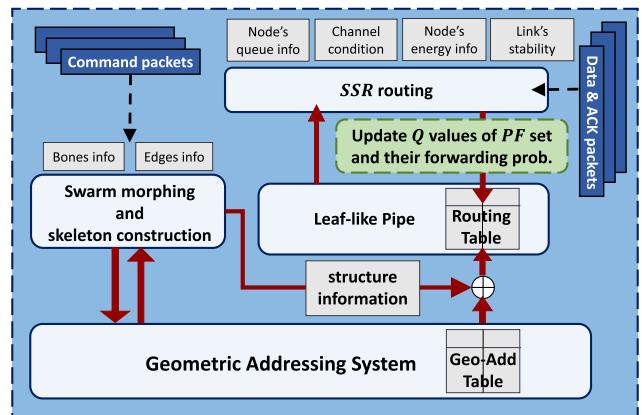
### 4) TABLES MAINTAINED IN NODES

In addition to the general neighbor table that records the 1-hop neighbors' information (such as their IDs, distances, etc.), each node also maintains a geo-address table that contains all nodes' ID and their geo-addresses, as well as the time-stamp when the geo-address is updated.

The geo-address table is distributed throughout the network (by the leader) when the swarm formation is constructed, and may be updated if there are major changes in the swarm (e.g., after each shape morphing). Meanwhile, each node may also update the geo-address extracted from a packet's header, if the corresponding time-stamp is newer than the one saved in the table. Nodes also maintain a routing table for one or more destinations, which includes the next-hop relay nodes (called *potential forwarders* or *PF*) and the cost of the path to the destination, initiated by each *PF* (denoted as  $Q_j$ ).

## B. SSR COMPONENTS

Fig. 3 illustrates the multi-tier architecture of the proposed *SSR* scheme. At the very bottom layer, geometric addressing system provides the approximate location of each node. The formation information, such as the length and the angle of the bones, is distributed through the skeleton by the leader via mission command messages, helping to construct and morph the formation. The geo-address table, containing the nodes' geometric addresses, is constructed based on the skeleton structure and a parent-child relation model among the nodes. Using this table and some basic information on the structure (can also be extracted from the table), the nodes construct their routing tables which include the set of next-hop forwarders *PF* and their qualification values  $Q_j$ . Forwarding to a set of next-hop nodes leads to a leaf-like routing pipe between a source-destination pair.



**FIGURE 3. SSR modules.**

*SSR* is a distributed scheme. To make its forwarding decisions, a node needs information only from the nodes in the *PF* set, i.e. their  $Q_j$  values, instead of all the pipe nodes or the whole network. This is why the *SSR* protocol has a lower complexity compared to the conventional RL-based schemes or proactive link-state routing schemes. The  $Q_j$  values can be derived based on a weighted combination of some metrics, such as queue status (including service delay, queue length, etc.), channel condition, the node's remaining power, link stability or expiration time, etc. When the data packets are dispatched through the pipe, nodes gradually update their

value and the forwarding probability to the *PF* set, leading to the selection of high-quality paths.

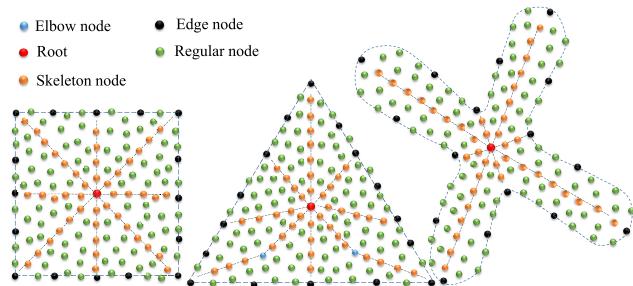
#### IV. LEAF-LIKE PIPE FRAMEWORK

In order to speed up the data forwarding process and reduce the route discovery overhead, the *SSR* protocol uses the greedy forwarding concept by exploiting the swarm structure. Based on the approximate geometric position of the destination, a relay node transmits the packets to a set of its neighbors called *potential forwarders* that are in the direction to the destination based on their geo-addresses.

##### A. SWARM SKELETON STRUCTURE

The skeleton is used as a reference to facilitate the geometric addressing and shape morphing. A similar concept has been previously used for static sensor network [8], where the extracted skeleton was the medial axle of the network shape. In our case, due to the mission-guided UAV movement, the skeleton is a pre-known structure that can guide the swarm to form a new shape.

Inspired by the study in [34], a star-shaped skeleton with the swarm leader as its root is used, as shown in Fig. 4. The reasons behind choosing star skeleton are threefold: 1) it facilitates the propagation of leader's commands uniformly and quickly throughout the network, 2) it helps to guide the nodes to build a new formation in a distributed manner during the swarm morphing process, 3) it provides a 2D coordinate system suitable for the geo-addressing framework.

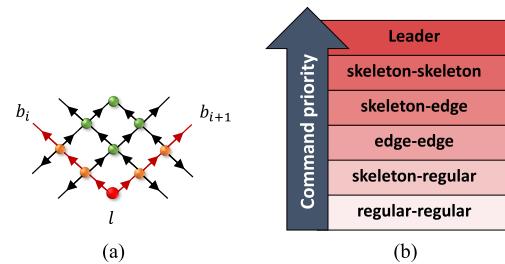


**FIGURE 4.** Examples of swarm formations with skeleton consisting of 8 bones.

The skeleton consists of branches (Fig. 4), which form the skeleton bone and are not necessarily arranged in a line. The use of “elbow” nodes enables the skeleton to represent more complex shapes and divide them into approximately equal-area pieces. In general, the nodes are classified into four main categories: 1) leader or root  $l$ ; 2) skeleton node set  $S$  that consists of bone nodes  $B_i \subseteq S$ ,  $i = 1, \dots, k$ , where  $k$  is the number of skeleton bones; 3) edge nodes located in the border of the swarm, and 4) regular nodes. Each bone can have up to  $e$  elbows.

The skeleton nodes can guide other nodes to form a desired swarm shape. To manage the formation in a distributed manner, nodes follow a hierarchical parent-children structure as

shown in Fig. 5(a). The leader  $l$  transmits the abstract information on the desired swarm structure to the nearby bone nodes and guides them to reach the desired positions. Then this information is passed through the bones and each bone node guides the next bone node (its child) until the complete framework is constructed (parent-child formation control strategy). Transmitting this geometric structure information only to the skeleton nodes instead of the entire network, reduces the communication overhead.



**FIGURE 5.** (a) Parent-children relation between different node types. Here an arrow points from a parent to its children. (b) Priorities of different swarm messages.

The network can first make a rough representation of the desired shape by constructing the skeleton and some main edges. Then other (regular) nodes can be placed, starting from the inner layers. Some UAVs, called free nodes, may move towards inner parts to fill any vacant area there. A node can leave its parent node, if it receives a positioning command with a higher priority.

##### B. SWARM MESSAGES

Swarm messages have different priorities as illustrated in Fig. 5(b). Here “A-B” command means that a node with role “A” commands another node to accept the role “B”. For instance, if a node has already been accepted as a skeleton node (by receiving a leader or skeleton-skeleton message), it will discard any skeleton-regular or regular-regular messages. Among all command messages, the leader and skeleton-skeleton messages have different formats as they contain the information on the skeleton’s shape, such as the angles and the lengths of the bones. The swarm messages are briefly described below.

###### 1) SKELETON MESSAGE

Every bone node that receives a skeleton command message records all the information about the skeleton, and updates its position based on its assigned bone index. Then it passes the message to the next bone node. Fig. 6 illustrates the command message format. Suppose the number of bones is 8, then the skeleton information can be fit into a 72-Byte (72B) packet, containing 8B information for each bone and an additional 8B for the packet type and sender and receiver information.

In Fig. 6, the 4-bit *Num* field specifies the total number of bones  $k$  ( $k \leq 16$ ) in the skeleton. The *bIdx* contains the *bone index* that the sender is located in, where  $1 \leq bIdx \leq k$ .

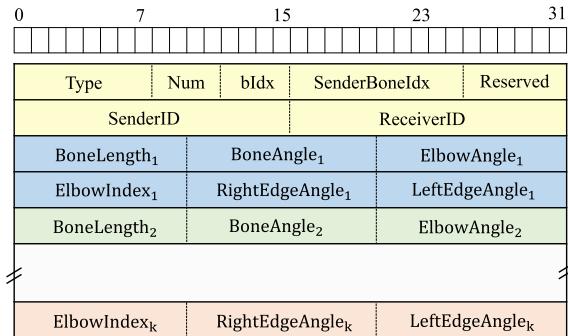


FIGURE 6. Format of the command message.

*SenderBoneIdx* contains the bone index of the sender. Initially, the root sets it to zero, and it increases by one when passing through each bone node. The *BoneLength<sub>i</sub>* represents the number of nodes in bone *i* (denoted as  $|B_i|$ ). By assigning 10 bits to it,  $|B_i|$  can reach 1024, which represents a large swarm network. The 11-bit *BoneAngle<sub>i</sub>* field specifies the initial angle of the bone *i*, in terms of a pre-defined reference line such as the East or North direction. The *ElbowIndex<sub>i</sub>* and *ElbowAngle<sub>i</sub>* represent the bone index and the angle of the elbow node in *i<sup>th</sup>* bone, respectively. *ElbowIndex<sub>i</sub>* = 0 means that *i<sup>th</sup>* bone does not have any elbow.

Upon receiving a command message, the node first checks *bIdx* and increases it by one. Then it finds its own bone index by increasing *SenderBoneIdx* field and compares it with *ElbowIndex<sub>i</sub>*. If the node is located after an elbow node, it takes the corresponding *ElbowAngle<sub>i</sub>* as its bone parent; otherwise, it adjusts its position corresponding to *BoneAngle<sub>i</sub>*'s value. If the node's bone index is equal to the value of *BoneLength<sub>i</sub>*, it is located at the end of bone *i*. The end bone node guides left- and right-edge nodes to reach their desired positions, based on the fields *LeftEdgeAngle<sub>i</sub>* and *RightEdgeAngle<sub>i</sub>*, respectively. Then every edge node that has been adjusted in the determined location guides the next edge node (by sending an edge-edge command message).

## 2) EDGE AND REGULAR COMMAND MESSAGES

These messages do not contain much information about the skeleton. They only tell about the new position of the receiver node. Besides common information on packet type and sender/receiver IDs, an edge-edge command includes the direction and distance that the receiver needs to travel, in order to reach the desired location in the swarm edge.

## C. SKELETON-BASED GEOMETRIC ADDRESSING

The geometric addressing of each node is determined based on its distance to the adjacent bones. Here the number of hops is used as the distance to a node since the exact geographic location of the nodes may not be accessible. Still nodes are distinguished by their unique IDs. If the destination has recently moved in the region, a local search is performed to find the new geometric position (geo-address) of the destination.

The proposed addressing procedure is inspired by the naming method of the medial axis based naming and routing protocol (MAP) [8]. However, the addressing procedure proposed in this article overcomes three drawbacks of MAP scheme: (1) MAP may get too far from the shortest path when it first moves along the latitude and then along the longitude, as shown in Fig. 7. The proposed scheme moves along the diameter direction to reduce the path length. (2) Unlike MAP, the proposed scheme does not require that every node has a complete knowledge of the skeleton. Note that frequently updating the skeleton information and flooding it all through the network introduce a high overhead. (3) MAP-based routing cannot achieve load balancing. For instance, two sources with the same height *h* will transmit their data along the same *h*-latitude route, which may lead to a congested path. In contrast, the SSR-based leaf-like routing pipe (shown in Fig. 7(b)) can easily achieve load balancing by using different paths.

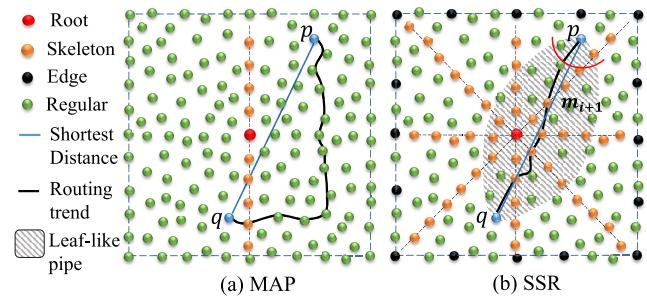


FIGURE 7. Comparison of data forwarding paths in SSR and MAP.

A given node *g* is addressed using the tuple  $G(g) = (R_i, n_i, n_{i+1}, h_i, h_{i+1})$ . Here  $R_i$  is the index of the region surrounded by two bones  $b_i$  and  $b_{i+1}$ . The number of skeleton bones,  $k$ , stays constant during the swarming process. These bones are indexed in order, but their sizes and shapes may change.  $n_i$  and  $n_{i+1}$  are the indices (hop counts) of the nodes located in bones  $b_i$  and  $b_{i+1}$ , respectively. These are the roots of the shortest-path trees from the bones to *g*.  $h_i$  and  $h_{i+1}$  are the hop-count distances of *g* from  $n_i$  and  $n_{i+1}$ , respectively. They also reflect the depth of the entire tree. If *g* is a bone node, then the geo-address can be simplified as  $G(g) = (R_i, n_i, 0, 0, 0)$ , where  $n_i$  is the index of the bone node. The leader's geo-address is a tuple of zeros.

By using such an addressing system, a 2D virtual coordination system is generated, where the skeleton bones are located in the coordination axis. Fig. 8 illustrates an example of geometric addressing for a given node *g*. Considering  $i = 2$  (i.e., the node is located in the region  $R_i = 2$ , which is surrounded by  $b_2$  and  $b_3$ ), the address of node *g* is  $G(g) = (2, 4, 2, 2, 3)$  in this example.

## D. LEAF ESTABLISHMENT AND POTENTIAL FORWARDERS

Similar to AODV scheme, every node maintains a routing table for one or more destinations (denoted as  $q_i$ ). However, there are some differences between SSR and AODV routing

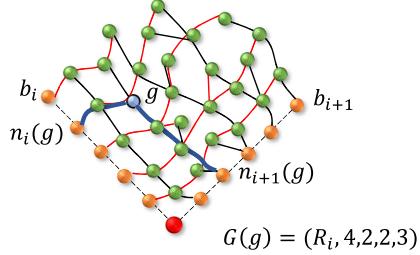


FIGURE 8. An example of assigning geo-addresses.

Dst-ID	Geo-Add	Update Time	$PF_v(q)$	Q-value
$q_1$	$G(q_1)$	$t_1$	$u_1$	$Q_1$
			$u_2$	$Q_2$
			$u_3$	$Q_3$
			$u_4$	$Q_4$
$q_2$	$G(q_2)$	$t_2$	$u_1$	$Q_1$
			$u_2$	$Q_2$
			$u_3$	$Q_3$
			$u_4$	$Q_4$

FIGURE 9. An example of SSR routing table.

tables. Fig. 9 illustrates the structure of a typical routing table in *SSR*. Since there is no route discovery phase and consequently no RREQ and RREP packets, the “destination sequence number” and “hop count” fields are not used. Another major difference is that the “next hop” field is now a multi-input field that can record up to  $J$  ( $=4$  in this example) potential forwarders (i.e., the next-hop relay nodes, denoted as  $u_j \subseteq PF(q_i)$ ) to reach the destination. This routing table has another field that records the cost of the path (denoted as  $Q_j$ ) initiated by each potential forwarder, which is described in Section VI. It is used to calculate the forwarding probability for different next-hop nodes.

After knowing the geo-address of the destination  $q$ , the source node  $p$  starts data transmission. In order to not divert the packets much from the shortest path between nodes  $p$  and  $q$ , the routing scheme dispatches the packets inside a leaf-like pipe around the shortest path, shown by the shaded area in Fig. 7(b). Upon receiving the data, the relay node  $g$  keeps an entry in its routing table for destination  $q$ , similar to the case shown in Fig. 9. The node finds the  $PF$  set, and distributes the received packets among them, based on the forwarding probability (to be explained in Section VI).

Each node that receives a packet addressed to the destination  $q$  estimates the direction of  $q$ , called *trend*, and then selects the set of next relay nodes with equal or shorter distance to  $q$ . It is worth mentioning again that by distance we mean hop-count distance based on the geo-address. By comparing the first element of their geo-addresses, the node will know that it is in the same or different region as  $q$ . If the node  $g$  and destination node  $q$  (with  $G(g) = (R_i, n_i, n_{i+1}, h_i, h_{i+1})$  and  $G(q) = (R_j, n_j, n_{j+1}, h_j, h_{j+1})$ ) are in the same region  $R$ ,

the hop-count distance corresponding to the bone  $b_i$  can be found as follows:

$$d(g, q) = \sqrt{(n_i - n_j)^2 + (h_i - h_j)^2} \quad (1)$$

Note that (1) can be rewritten with respect to bone  $b_{i+1}$  as well. Estimating the distance of nodes located in different regions may need a more complex calculation. However, a good estimation can be obtained using trigonometry (illustrated in Fig. 10). The distance  $d(g, q)$  can be found through the following equations:

$$\alpha = \sum_{k=i+1}^{j-1} \alpha_k + \tan^{-1} \frac{h_j}{n_j} + \tan^{-1} \frac{h_{i+1}}{n_{i+1}} \quad (2)$$

$$d(g, q) = \sqrt{n_{i+1}^2 + h_{i+1}^2 + n_j^2 + h_j^2 - 2\sqrt{n_{i+1}^2 + h_{i+1}^2}\sqrt{n_j^2 + h_j^2} \cos \alpha} \quad (3)$$

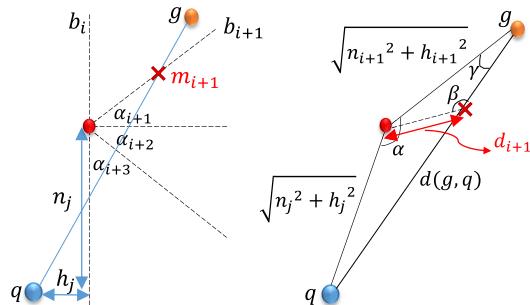


FIGURE 10. Finding hop-count distance based on geo-address information and skeleton structure.

where  $\alpha_k$  is the angle between the skeleton bones located in the areas between the  $R_i$  and  $R_j$  regions. If the node does not know the precise values of  $\alpha_k$ , it can use an approximate value;  $\alpha_k = 360/k$ . The above equations are derived by assuming that the node estimates a *trend* in a clockwise direction around the root by comparing  $R_i$  and  $R_j$ . If the trend is anti-clockwise, the equations should be updated by substituting  $(h_j, n_j)$  with  $(h_{j+1}, n_{j+1})$ , and  $(h_{i+1}, n_{i+1})$  with  $(h_i, n_i)$ , respectively. To make the search of  $PF$  easier, the node  $g$  can first find the closest bone node in the trend ( $m_{i+1}$  in Fig. 10), and then use (1) to locally compare the distance of the neighbors to  $m_{i+1}$ . Here  $m_{i+1}$  can be found by using the equations below:

$$\gamma = \sin^{-1} \frac{\sqrt{n_j^2 + h_j^2} \sin \alpha}{d(g, q)}, \quad \beta = 180 - (\gamma + \tan^{-1} \frac{h_{i+1}}{n_{i+1}}) \quad (4)$$

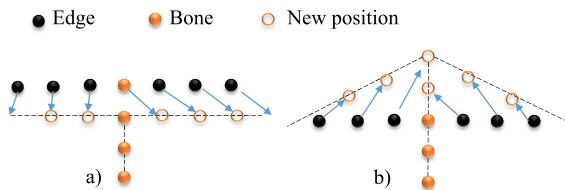
$$m_{i+1} = \left[ \frac{\sin \gamma \sqrt{n_{i+1}^2 + h_{i+1}^2}}{\sin \beta} \right] \quad (5)$$

## V. SWARM MORPHING

In the initial phase of the skeleton construction, the leader propagates the positioning commands to the first layer of bone nodes, which can be a set of the neighbors close to the specified position in the desired skeleton structure. Every bone node is responsible to find the next bone node. A parent-children relation is established across the bone. The last bone node in bone  $b_i$  adjusts the locations of the first edge nodes in its left and right sides, based on the edge angle information in the command message. The edge construction proceeds across the edge until it is completed. A rough layout of the formation can be detected after the construction of the swarm bones and edges. Then the (regular) nodes can be arranged to fill out the entire swarm.

The morphing phase initializes by propagated a new command message through the bones. Bone child position is adjusted based on the information held in the command message. Upon moving, a node sends a FOLLOW message to its children containing its movement information. If a child receives such messages from both parents (one parent is in the tree rooted in bone  $b_i$  and the other is in the tree rooted in bone  $b_{i+1}$ , see Fig. 8), it chooses one of them and the parent-children relation with the other parent may break.

Fig. 11 illustrates how a bone shrinks or expands. If the length of the new bone is shorter than the previous bone, extra bone nodes will be added to the edges, as shown in Fig. 11(a). If it is longer, then the closer edge nodes will join the bone to expand it (see Fig. 11(b)). The reason for exchanging only the bone and edge nodes during expansion or shrinking, is to keep the boundaries closed and reduce the changes in the nodes' geometric addresses.



**FIGURE 11.** An example of a) bone shrinking, and b) bone expansion.

After adjusting its child in the bone, every bone node adjusts the position of the left or right (regular) children. This procedure continues through the regular nodes. Nodes select new child only from its own region (which is known through the first element of the geometric address). This helps to keep a UAV in the same region during the morphing. This also helps to reduce the overhead of searching a displaced node and simplify the routing process. When a node cannot be found in the expected place, a local search limited to the region  $R_i$  is carried out to find the new location, as explained in Section VI. Here, a region refers to the area surrounded by two bones.

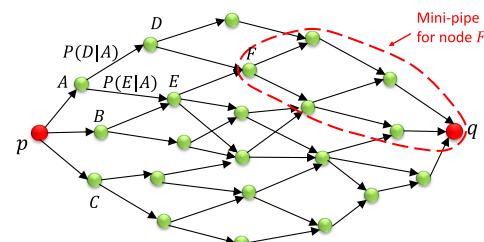
If a node updates its geo-address, its children need to update theirs accordingly. Upon changes in its geo-address, every node  $g$  broadcasts the new geo-address (via HELLO

packets) to its neighbors. The leader gathers all the nodes' geometric addresses after each shape morphing, as the address of several nodes might have changed. The leader constructs an updated geo-address table and propagates it throughout the network.

It is worth mentioning again that UAVs in mission-driven formation move in an organized and controlled manner to maintain the overall structure. When the structure is changing, nodes move in a way that has the minimum impact on swarm instability (it is called *distortion* hereafter). Nodes located in the inner parts of the swarm, experience the lowest displacement while those in the outer parts may have to move further. Considering that SSR establishes a data forwarding "pipe" instead of a single or several separate paths, multiple active paths are available to the destination, even if several links are broken. The routing pipe is discussed further in the following section.

## VI. SKELETON-BASED ROUTING

The SSR scheme aims to achieve an intelligent swarm-adaptive, load-balanced, and high-throughput routing. It builds a leaf-like routing pipe (from the source to the destination) composed of inter-connected paths, as illustrated in Fig. 12. Based on the feedback received from the potential forwarders, nodes gradually adjust the frequency of data transmission through these paths.



**FIGURE 12.** A leaf-like pipe composed of inter-connected paths. Each link is associated with a forwarding probability.

When a source node  $p$  decides to send packets to a destination  $q$ , it first looks for a matching entry in its routing table and extracts  $G(q)$  from the geo-address table. If there is a corresponding forwarder set  $PF$  that is up-to-date (SSR can find this by comparing  $G(q)$  and its time-stamp with those in the routing table), it starts to dispatch packets among  $PF(q)$  based on the corresponding  $Q$ -values. Otherwise, it first updates  $PF(q)$  and their corresponding  $Q$ -values, as described in Sections III and IV.

Besides the source and destination IDs, the data packet's header also includes  $G(p)$  and  $G(q)$  (as well as the time-stamp) when the source has updated  $G(q)$ . When a relay node receives a packet, it updates  $G(p)$  in its geo-address table. If the node has a newer  $G(q)$  compared to the one in the packet, it inserts the updated one in the packet header and notifies the source about the changes in  $G(q)$ . In general, every node that hears a new geo-address will update its table. Upon significant changes in the geo-address

of the node or destination  $q$ , the set of next-hop nodes,  $PF(q)$ , is updated and the  $Q$ -values of newly added nodes is initialized.

#### A. BUILDING SKELETON ROUTING VIA A DISTRIBUTED STOCHASTIC DYNAMIC PROGRAMMING APPROACH

The mesh architecture shown in Fig. 12 inspires us to formulate the problem of skeleton-based routing as a stochastic shortest-path problem [35], which can be solved using a dynamic programming approach [36]. A cost (or reward) is assigned to each link inside the leaf-like pipe, and the cumulative cost is calculated in a backward recursion fashion. However, the cost values are not known in advance, and are subject to the changes due to the dynamic nature of the swarm network. Hence, an online dynamic-programming-based approach is used which can adapt quickly to the changes of the network conditions. It has two main advantages:

- 1) It is solved in a distributed manner. Instead of having a central point which gathers all the state information frequently and solves the whole problem, every node calculates its own value.
- 2) Nodes gradually update their forwarding probabilities based on the values of their forwarding nodes and hence, the scheme will learn how to dispatch the data packets inside the pipe in order to get the best performance with a low computation complexity.

To dispatch the cumulative cost (or reward) through the path in the backward manner, one can take advantage of the ACK messages that are popularly used in routing protocols. A mini-pipe is initiated from each relay node  $g$  to the destination  $q$ , as shown in Fig. 12. The “expected” cost (or reward) of the sending packets from the node  $g$  through the mini-pipe, is denoted as  $V(g)$ , which can be piggybacked onto  $g$ ’s ACK messages. Upon receiving it, the sender updates the  $Q$ -value that corresponds to the forwarder node  $g$  in the routing table.

The problem can be modeled as a Markov decision process (MDP) in which every node is a state,  $s \in S = N$  ( $N$  is the set of the nodes in the pipe), with the action space  $A \triangleq PF(s)$  and the next state  $s' = a(s)$ . Here the transition probability function is the probability of choosing a possible forwarder, i.e.,  $P(s'|s, a(s)) = P(a(s)|s)$ . Then the expected cost (value) at each node  $v$  can be found by using the Bellman equation [36]:

$$V(v) = \sum_{u \in PF(v)} [c_{vu} + V(u)]P(u|v) \quad (6)$$

where  $c_{vu}$  is the immediate cost of transferring packets from  $v$  to  $u$ , and  $V(u)$  is the cost value of node  $u$ .  $P(u|v)$  is the probability of forwarding packets to node  $u$ , where  $\sum_{u \in PF(v)} P(u|v) = 1$ . Initially, when a new forwarder is added to the pipe, its associated cost is set to the initial value  $V_0$  which gets updated as the packets pass through the node. Considering  $\alpha_Q$  as the learning rate, the  $Q$ -value

in the routing table can be updated by using the following equation:

$$Q(u) = (1 - \alpha_Q)Q(u) + \alpha_Q V(u) \quad (7)$$

#### B. PACKET FORWARDING PROBABILITY

A node  $u \in PF(v)$  may become the next forwarder based on a probability distribution function. There are different action-selection strategies in the reinforcement learning (RL)-based schemes, such as random action-selection and  $\epsilon$ -greedy approaches. Here, the Boltzmann approach is adopted which selects the actions based on a probability distribution function, such that the more rewarding actions are selected with higher probabilities (more frequently).

The major advantage of the Boltzmann approach over the  $\epsilon$ -greedy approach is that the non-optimal actions are not selected with equal probability. Instead, they are taken with a frequency corresponding to their estimated rewards. Hence, more rewards can be gathered in the exploration phase. In this approach, the probability function is represented by the Softmax function (or normalized exponential function) with the temperature parameter  $\tau$ . Here, the proper choice of  $\tau$  is important. The higher values lead to almost equiprobable actions as in random approach, while the lower values can make a big difference in action selection probability [37].

Other exploration approaches such as Bayesian neural networks or deep RL (DRL) might work better but introduce a high computation complexity (thus not suitable to UAVs networks). Thus, the Boltzmann action-selection approach is adopted where the probability of choosing node  $u$  as the next forwarder (i.e., forwarding probability) can be found through the following equation [38]:

$$P(u|v) = \frac{e^{Q(u)/\tau}}{\sum_{y \in PF(v)} e^{Q(y)/\tau}} \quad (8)$$

Note that in (8), the  $Q$  values represent the reward associated with the actions. Thus, if the problem is formulated as minimizing a cost, the cost values  $Q_c$  can be first converted to reward values  $Q_r$  before calculating the forwarding probability. For instance,  $Q_c$  can be normalized and then  $Q_r = 1 - Q_c$ .

#### C. COST FUNCTION

The definition of  $c_{vu}$  is application-specific and depends on QoS requirements. It can be a weighted combination of several metrics. For instance, video streaming demands a high data rate and very low latency. Control packets hold sensitive information and hence require timely delivery. Some IPv4 applications and offline transfer of texts or documents can be classified as the best-effort services.

For the best-effort traffic, the data packets can be routed through the paths with a higher remaining energy in the intermediate nodes, although they may not provide low-latency communications. For instance, the work in [39] has proposed an energy-aware reward function which, with respect to our problem, can be represented as follows:

$$r_{vu}^{EA} = \text{sigm}(\Delta E(u) \times \beta_e) \quad (9)$$

where  $\Delta E(u)$  is the difference between the energy of the node  $v$  and the next forwarder node  $u$ , and  $\beta_e$  is an energy difference scaling factor. The Sigmoid function rescales the energy difference with  $[-1, 1]$ .

To make *SSR* support high-throughput and low-latency communication for high-priority traffic, the per-hop service delay is defined as the cost, i.e., the time duration from the moment the packet enters the queue to the moment it is successfully delivered, denoted by  $c_{vu}^{SD}$ . It represents the queue length and the channel utilization (i.e., the medium access delay in contention based protocols) and hence, leads to a load-balanced solution [25], [26]. It also indirectly tells about other situations such as RF interference, channel collisions and link handoff. If a node is experiencing high interference or poor SINR (due to the long distance), the packet drop rate increases, which leads to increase in packet delivery delay. A node can find the expected service delay through the statistics of the past experiences.

#### D. LOCAL SEARCH

If the destination  $q$  moves away from the expected location specified in the packet header, a node that expects  $q$  to be in its neighborhood (by comparing their geometric addresses) may not be able to find it. In this case, the node initiates a local search to find the current address of  $q$  by broadcasting an *address request* (AREQ) packet. AREQ contains the initiator's ID and address,  $q$ 's ID and its previous address, and the time-stamp. Any node with a newer address should reply and every node hearing a newer address updates its geo-address table.

In fact, a newer address represents  $q$ 's trajectory towards the new position. The initiator is responsible for gathering the addresses and sending an update on the latest address to the source. AREQ flooding only occurs in a small region ( $R_i$ ) in the skeleton-based routing protocol.

#### E. TWO-STEP DISCARDING POLICY (TSDP)

For service delay as the cost function,  $V(v)$  in (6) represents the expected latency before reaching the destination. It can also be used for active queue management that aims at relieving the network congestion or reducing the end-to-end latency by discarding some packets in the queue before it gets full. Tail drop and random early detection (RED) are examples of widely used discarding approaches (RFC 2309 and 7567). However, for delay-sensitive applications with large queue sizes, it is preferable to use a discarding policy that drops packets based on the priority levels and the probability of on-time delivery to the destination. Inspired by weighted random early detection (WRED) approach [40], here a two-step discarding policy (TSDP) is proposed that operates based on the service delay.

TSDP consists of two steps:

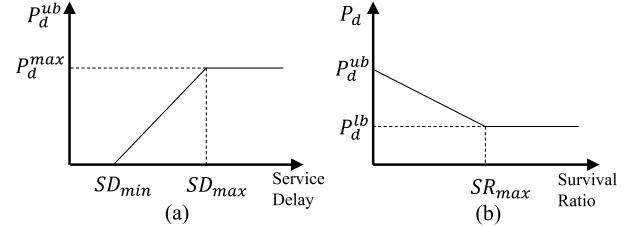
- 1) Finding  $P_d^{ub}$ , the upper bound of the packet discard probability. As illustrated in Fig. 13(a), this value is low (close to 0) for low-latency queues, but increases linearly as the service delay rises until reaching  $P_d^{max}$ . One can determine

how to discard the packets with different traffic types through different configurations of  $P_d^{max}$ ,  $SD_{min}$  and  $SD_{max}$ . The calculated  $P_d^{ub}$  will be used in the second step.

2) In this step, the discarding probability of packets (in the same traffic type) is differentiated by estimating the possibility of timely delivery. Assume that  $t_l$  is the packet lifetime,  $t_{vq}$  is the expected latency from node  $v$  to destination  $q$ , and  $t_{pkt}$  is the packet age. Then the *survival ratio* of packet  $pkt$  entering node  $v$ 's queue can be defined as follows:

$$SR(pkt, v) = \frac{\max(t_l - t_{pkt} - t_{vq}, 0)}{t_l} \quad (10)$$

When the packet is new and a low-latency route is estimated, the survival ratio increases, which leads to a decrease in  $P_d$ , as shown in Fig. 13(b). However, a low  $SR(pkt)$  indicates that the packet has a small chance of arriving at the destination in time and hence, this packet is dropped with a higher probability.

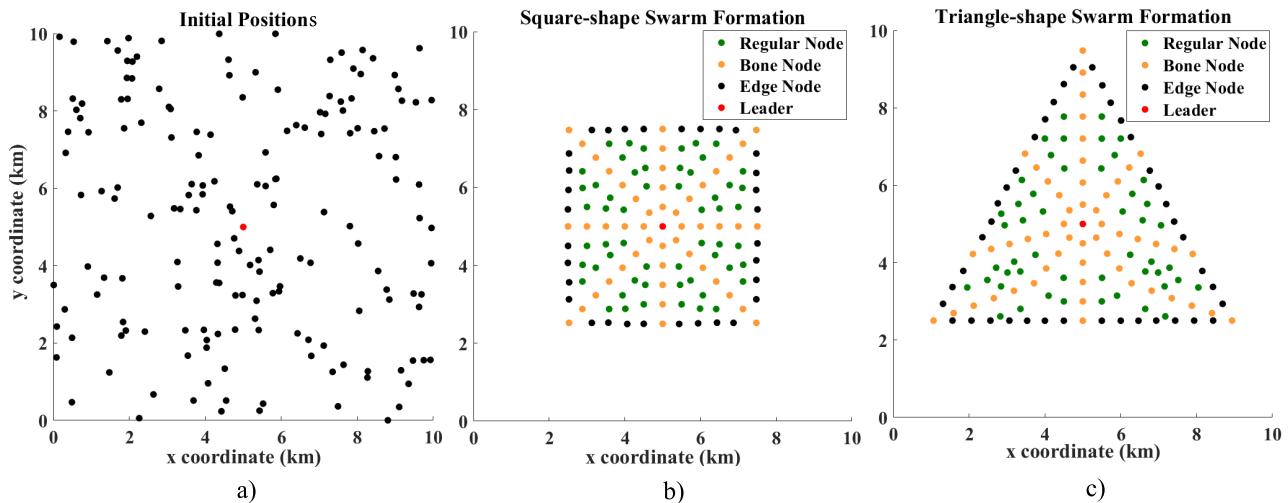


**FIGURE 13.** (a) Upper bound on the drop probability in the first step of TSDP, (b) Finding the drop probability of a packet in the second step based on its probability of on-time delivery to the destination.

$P_d^{lb}$  is a fraction of  $P_d^{ub}$ , i.e.  $P_d^{lb} = \alpha P_d^{ub}$ . If a node does not have enough information to estimate  $t_{vq}$ ,  $\alpha$  can be set to 1 (i.e., the survival ratio is not counted). In general,  $t_{vq}$  can be estimated through the recursive equation (6), i.e., it is equal to  $V(v)$  if  $c_{vu}$  is the per-hop service delay. Each node  $v$  in the pipe can evaluate the service delay to the next forwarder  $u$ , calculate  $t_{vq}$  and send it to its upstream node.

#### F. RESILIENCE TO INTENTIONAL INTERFERENCE

*SSR* can be combined with the detection schemes for intentional interference to detour the data around the interfered areas. If UAVs are capable of detecting the intentional interference and scoring themselves based on the probability of being interfered (similar to [30]), this score can be reflected in the  $Q$  value of nodes. Hence, in (8), the probability of forwarding to a potential forwarder  $u \in PF(v)$  increases, if the mini-pipe initiated from  $u$  has a lower chance of being involved in intentional interference activities (see Fig. 12). In this case, *SSR* can route the packets through the “safe” areas in the pipe. Please note that boundary nodes, i.e. the nodes close to the interfered area, are more vulnerable to intentional interference in the near future due to mobility. Thus, they should be assigned a lower score.



**FIGURE 14.** a) Initial positions of UAVs randomly distributed in a 10km × 10km rectangular area; b) Square formation based on the skeleton concept; c) Triangular formation.

## VII. PERFORMANCE EVALUATION

In this section, the performance of the proposed UAV swarm morphing and *SSR* routing scheme is evaluated. A wireless swarm networking platform is developed in Matlab, where the nodes can morph from one shape into another and exchange packets through IEEE 802.11 channel access model. The simulation parameters are summarized in Table 2.

**TABLE 2.** Simulation parameters.

parameter	value
Max UAV speed	50m/s
Radio transmission Range	1km
Minimum distance between nodes	500m
Number of nodes	137
Number of skeleton bones	8
Data packet size	2k Bytes
ACK packet size	14 Bytes
Max link rate	5Mbps
Packet lifetime	5s
Channel access CW	15
Source node's CW	7
Initial node's forwarding cost $V_0$	0

### A. SWARM CONSTRUCTION

The swarm formation of two example shapes (square and triangular) is examined using the procedures explained in Sections IV and V. To construct the swarm formation from randomly distributed UAVs within a 10km × 10km rectangular area, as depicted in Fig. 14(a), the leader in the center propagates command messages to the nearby nodes and guides them to move towards the desired locations to form the first layer. Then, the skeleton nodes construct the core of swarm based on the information in the command message (Fig. 6).

The final shapes of the square and triangle formations are illustrated in Fig. 14(b) and (c), respectively. The results validate the ability of the simulator in terms of using the skeleton nodes to facilitate the swarm formation.

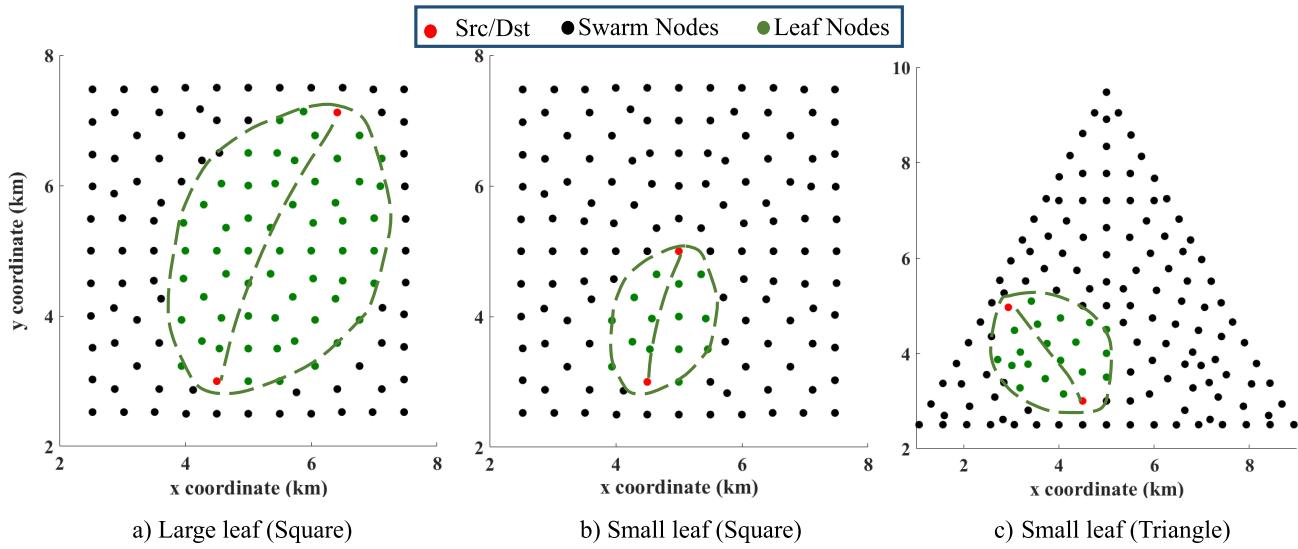
### B. ESTABLISHMENT OF LEAF-LIKE ROUTING PIPE

In this part, several scenarios with different source-destination pairs are considered, and the ability of the proposed model to construct leaf-like routing pipes based on the skeleton infrastructure and geometric addressing is evaluated. Fig. 15(a) illustrates the pipe constructed between two relatively far-away nodes (in two opposite regions in a square formation). The pipe starts from the source at the top, passes through several regions and gets merged again near the destination. Fig. 15(b) illustrates a smaller pipe from the leader to destination.

As seen in Fig. 15(a) with (b), if the distance between the source and destination is longer, the formed leaf could become wider and allows the establishment of more backup paths in the pipe. A wider pipe also leads to a higher throughput and better traffic balancing performance since more non-interfering paths can be built. Fig. 15(c) depicts another example of the pipe within a triangle formation. The pipe may not be like a leaf when the inter bones' angles are different or the bone elbows are presented. However, this is not the main concern as long as a group of nodes can be found to build the optimal routes among them.

### C. SSR PERFORMANCE IN LIGHT EXTERNAL TRAFFIC

In this section, performance of the *SSR* scheme is examined when the external (or background) network traffic is light and has a lower priority than the flow under investigation and is generated by other applications. Only one source and



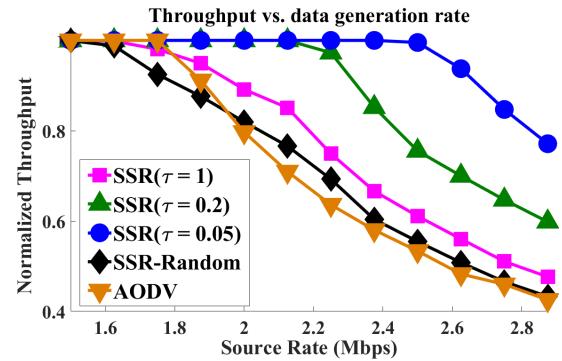
**FIGURE 15.** Establishing a leaf-like routing pipe based on the proposed geometric addressing: a) a large leaf in square formation; b) a small leaf in square formation; c) a small leaf in triangular formation.

destination pair is active in the routing pipe. Fig. 16 compares the (normalized) throughput of *SSR* protocol with that of *AODV* scheme, for the formation and the routing pipe illustrated in Fig. 15(a). Moreover, the impact of different values of  $\tau$  (the thermal factor in Boltzman model) on the routing performance is investigated. Here,  $\tau$  is a hyper-parameter and determines the sensitivity of action selection in Markov decision model (smaller  $\tau$  means higher sensitivity). In this figure, *SSR-random* represents the case when the data forwarders (*PF*) are chosen randomly.

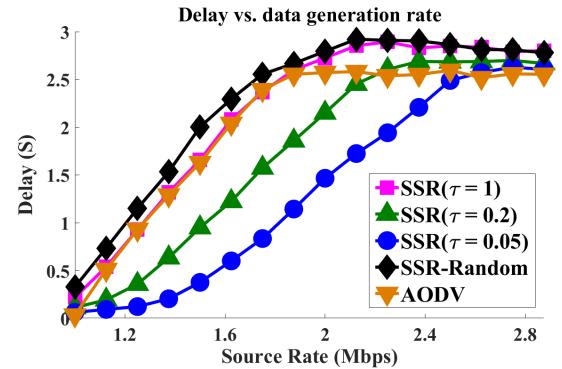
As shown in Fig. 16, all schemes work well for a low data generation rate. When the source rate increases, *SSR* with  $\tau = 0.05$  still shows a satisfactory performance while others have degrading throughput. The reason is that *SSR* with  $\tau = 0.05$  distributes packets to the paths that have a lower RF interference with each other. However, when source data generation rate exceeds 2.5Mbps, *SSR* with  $\tau = 0.05$  also exhibits throughput degradation. The main reason for throughput drop in high data rates is that the source node cannot timely flush out all the generated packets due to the congested channel.

Uniformly distributing packets among the forwarders in *SSR-random* scheme may not be a good solution since the inner nodes will suffer a stronger interference from nearby nodes. The result shows that *SSR-random* and *SSR* with  $\tau = 1$  exhibit a similar performance. The throughput of *AODV*, on the other hand, falls more rapidly since it uses only a single path at the center of the pipe.

Fig. 17 compares the delay of packets received at the destination for the above-mentioned scenarios. The result reveals that a higher  $\tau$  causes longer delay since the service time grows in queues. The *AODV*'s expected delay is lower than the other schemes at higher data generation rates as it passes the packets through the shortest path in the middle of the pipe.



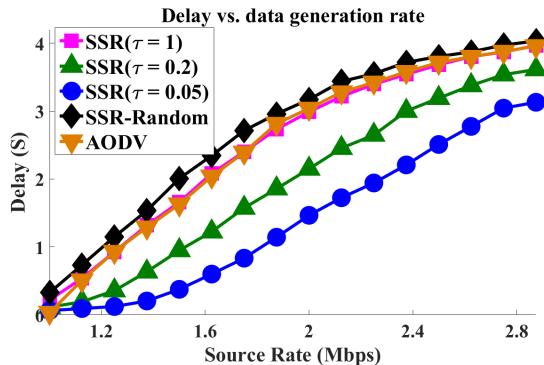
**FIGURE 16.** Throughput comparison for different forwarding action selection schemes when network traffic is light.



**FIGURE 17.** Received packet delay comparison for different forwarding action selection schemes when traffic is light.

However, a large portion of the packets are dropped on their way to the destination.

To account for the dropped packets in delay performance analysis, Fig. 18 illustrates the *total delay* as the expected



**FIGURE 18.** Total delay comparison for different forwarding action selection schemes when traffic is light.

delay of all packets, where the delay associated with a dropped packet is considered to be the packet lifetime. Based on the figure, AODV delay is similar to that of *SSR* with  $\tau = 1$  and *SSR-random*, while *SSR* with  $\tau = 0.05$  has the best performance in terms of routing the data through faster paths. The *total delay* is used in delay analysis hereafter.

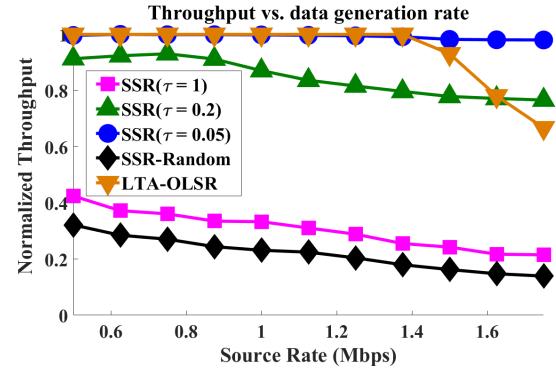
#### D. SSR PERFORMANCE WHEN NETWORK IS CONGESTED

Next, the performance of *SSR* in terms of its load-balancing capability is investigated under network congestion, for the network topology shown in Fig. 15(a). The external traffic could cause certain congestion around the leader node, as depicted in Fig. 21(a).

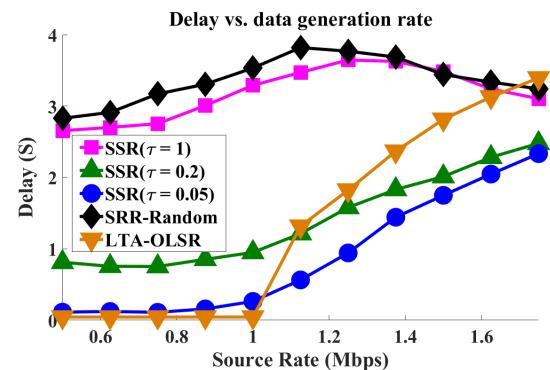
The throughput and delay for several configurations are evaluated in Figs. 19 and 20, respectively. Again, *SSR* shows a better performance when using smaller values of  $\tau$ , as the probability of choosing the optimal actions in Markov model increases. Though  $\tau = 1$  performs poorly, it is still slightly better than the random selection of the forwarders. Higher value of  $\tau$  leads to almost equal probability of actions in Markov model, and hence, the result is closer to that of *SSR-Random*.

The *SSR* performance is compared with FANET routing protocols ML-OLSR [25] and LTA-OLSR [26], which are load-aware proactive routing schemes. Both the ML-OLSR and LTA-OLSR find the best route to the destination which experiences the lowest traffic load. Averaging over buffer load and channel occupation, LTA-OLSR performance is comparable to *SSR* with  $\tau = 0.05$  for smaller data generation rates, as shown in Fig. 19. However, when the source rate rises, LTA-OLSR performance degrades due to the inter-node interference in single path routing, whereas *SSR* with  $\tau = 0.05$  outperforms LTA-OLSR as it dispatches the data packets through a routing pipe which can lead to a higher throughput.

As shown in Fig. 20, the packet delay in *SSR-Random* and  $\tau = 1$  may not always have an increasing trend when the source rate increases. In Fig. 21(b) and (c), some packets may be dropped in the path, which helps to mitigate the congestion and reduces the delay for the rest of the traffic. This fact is also mentioned in [30]. Due to the use of routing pipe, *SSR* with  $\tau = 0.05$  can balance the traffic load and decrease the



**FIGURE 19.** Throughput comparison for different action selection schemes (with congestion).



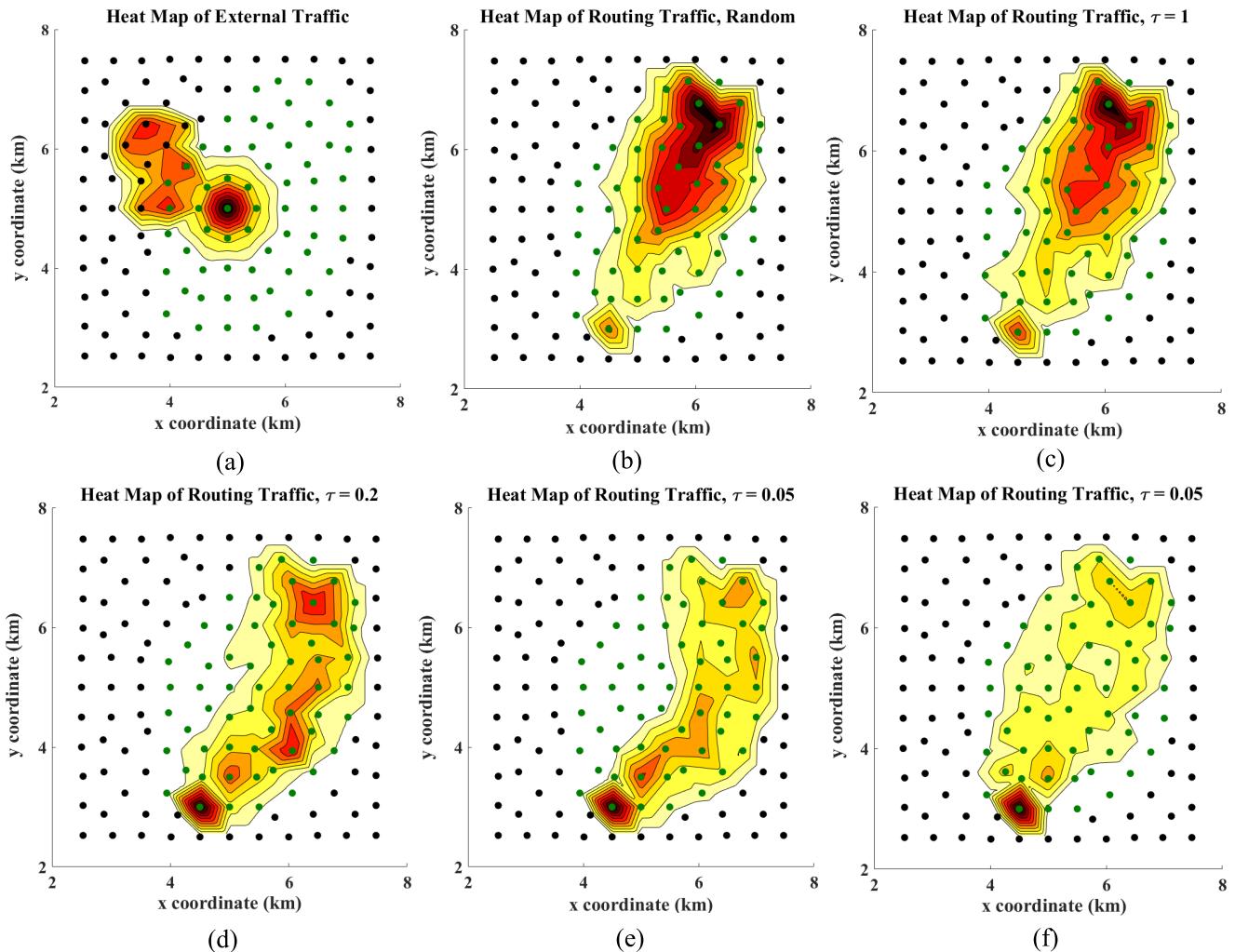
**FIGURE 20.** Delay comparison for different action selection schemes (with congestion).

delay more efficiently compared to LTA-OLSR which uses a single-path routing.

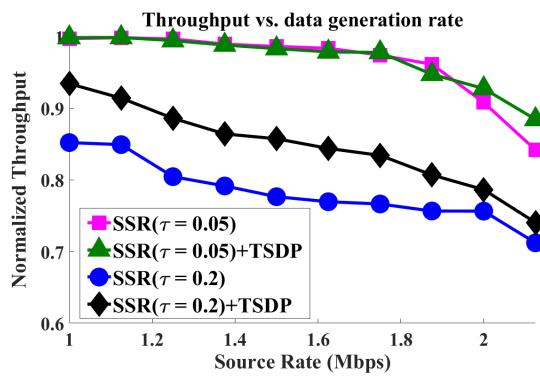
Fig. 21 compares the load distribution of *SSR* traffic across the entire leaf-like pipe, for the topology shown in Fig. 15(a). Here the routing traffic is almost evenly distributed in *SSR-Random* scheme or when using  $\tau = 1$ , regardless of the external traffic. However, the traffic load is heavier in the start section of the pipe but it decreases when it travels through the pipe, because some packets are dropped before reaching the destination. In these cases, *SSR* ignores the location of the congested areas and therefore cannot avoid them. On the contrary, the traffic associated with smaller values of  $\tau$  can successfully get away from the congested area, as depicted in Fig. 21(d) and (e).

Fig. 21(f) reveals that when there is no congestion *SSR* can evenly distribute traffic in the entire pipe, and the center of the pipe experiences slightly less traffic than the outer areas (thus the nodes in the center experience less interference). Here the destination receives more packets from the outer areas than the center.

Although small  $\tau$  increases the throughput by selecting the optimal Markov actions more frequently, it also reduces the possibility of exploring other actions. Hence the routing scheme may not be able to quickly adapt to the changes in network conditions. Thus, selection of a suitable value of  $\tau$  is a trade-off between performance and learning speed



**FIGURE 21.** (a) External load distribution due to other applications; (b) Load distribution of SSR traffic with random action selection under congestion; (c) Load distribution of SSR traffic for  $\tau = 1$  under congestion; (d) Load distribution of SSR traffic for  $\tau = 0.2$  under congestion; (e) Load distribution of SSR traffic for  $\tau = 0.05$  under congestion; (f) Load distribution of SSR traffic for  $\tau = 0.05$  without congestion.

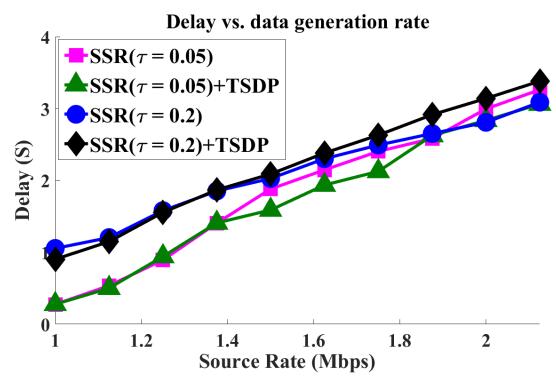


**FIGURE 22.** Throughput comparison with and without the use of TSDP in congested scenario.

in dynamic swarm environment. In the experiments, using  $\tau = 0.05$  achieves an optimal routing performance.

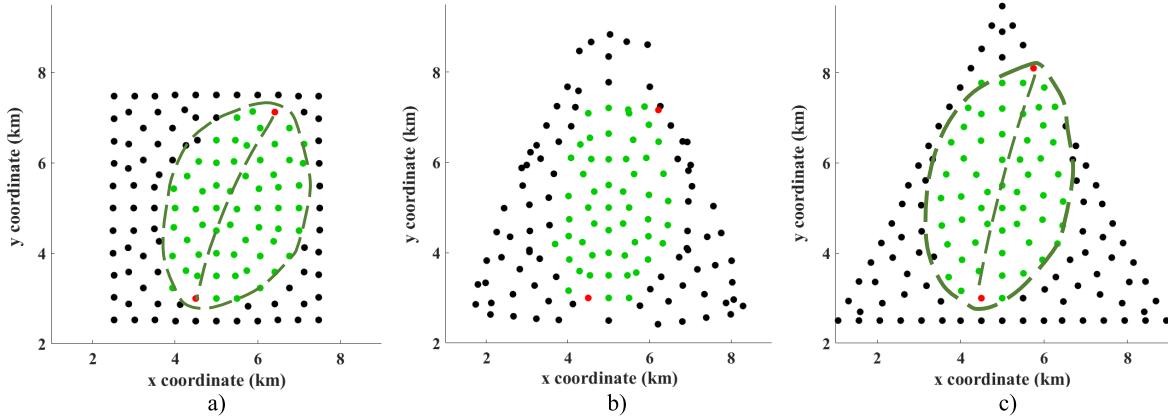
#### E. SSR PERFORMANCE WITH TSDP

Figs. 22 and 23 illustrate the effect of TSDP on the SSR throughput and delay, for the congestion scenario shown



**FIGURE 23.** Delay comparison with and without TSDP in congested scenario.

in Fig. 21(a). Table 3 summarizes the TSDP parameters for the SSR and the low-priority external traffic. In Fig. 22, TSDP does not noticeably affect the performance when the throughput is high. However, when the queues get



**FIGURE 24.** Swarm morphing from a square to triangular formation and the changes of the leaf-like routing pipe: a) Initial leaf-like pipe in the square formation; b) A snapshot of swarm at the middle of morphing; c) The end pipe in the triangular formation. The morphing process takes about 125s.

**TABLE 3.** TSDP simulation parameters.

Parameters	SSR Traffic	External Traffic
$P_d^{max}$	0.4	0.7
$SD_{min}$	0.25s	0.5s
$SD_{max}$	1s	1s
$SR_{max}$	0.5	N/A
$\alpha$	0.25	1

congested and the throughput degrades at higher source rates, TSDP improves the performance by filtering out part of the low-priority traffic as well as packets having a low chance of reaching the destination in time. Therefore, the competition for the channel access is moderate and more high-priority packets can reach the destination.

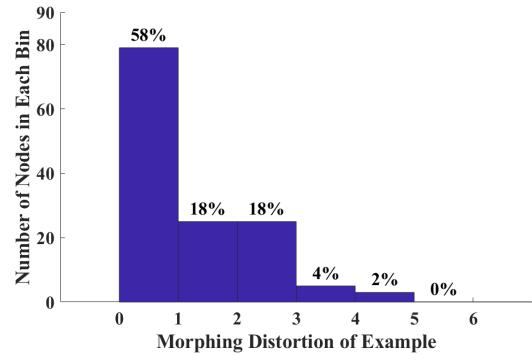
Fig. 23 reveals that TSDP may slightly increase the expected delay of the received packets, because more packets can reach the destination before expiration.

#### F. SWARM MORPHING

In this section, the performance of the *SSR* scheme is evaluated when the swarm morphs from one shape to another. Here, morphing from the square formation to triangular shape takes around 125s. The morphing results at three different time instances are illustrated in Fig. 24.

During morphing, the geometric addresses of some nodes may change and some other nodes may join or leave the pipe. This may also lead to an undesired change in the *PF* list. Generally, minimizing the topology distortion (in terms of RF connectivity) during morphing is preferred. We thus define the *morphing distortion* as the difference (in terms of the hop-count distance) between the previous and new geometric address.

Fig. 25 shows that about 58% of the nodes moved only 1-hop away in the geometric addressing system during morphing, and only 6% of them moved more than 3-hops away. This implies that due to localized movement of nodes, most



**FIGURE 25.** Morphing distortion measurement for the scenario shown in Fig. 24.

paths inside the pipe still have good RF connectivity. Most displacements occur in the outer areas of swarm, and the inner nodes can still maintain their established links. Moreover, the nodes do not leave their region  $R_i$  during morphing which significantly helps the local search (Section VI).

#### VIII. OVERHEAD AND COMPLEXITY ANALYSIS

The communication overhead incurred by transmitting the formation command message is  $O(N_s)$ , where  $N_s$  is the number of skeleton nodes. It can be rewritten as  $O(k \times d)$ , where  $k$  is the number of bones and  $d$  is the average length of a bone. These control messages are only transmitted along the skeleton bones. Other nodes are guided through the parent-child formation control strategy.

Updating the geo-address locally does not introduce a noticeable communication overhead. A child node updates its geo-address upon changes in the parent's address which is transferred via the conventional HELLO packets. It is not necessary to flood the new address if it has changed by only one or two hops due to using routing pipe and local search. Updating the geo-address table of all nodes after a major change in formation shape (which is not frequent) can

have an overhead comparable to OLSR topology database update. Alternatively, the leader can gather the addresses in each region through the parent-child relations, combine them all and flood it to the network which have communication complexity of  $O(2N)$ , where  $N$  is the number of nodes.

The communication complexity of local search in case of node displacement is  $O(2N/k)$ , since the route-seeking messages are only flooded inside a particular region. The overhead can be further reduced if a multi-point relaying technique is used. Lastly, the *SSR* routing has a low computation complexity, as it is executed in a distributed manner. Every node in the pipe updates the forwarding probability of its *PF* set when the value corresponding to a potential forwarder has changed. The complexity of this operation is  $O(|PF|)$ .

## IX. CONCLUSION

In this article, a novel UAV routing and morphing scheme based on the skeleton structure, called *SSR*, was proposed for swarm network. *SSR* includes an addressing system that provides nodes with geometric coordinates. Based on that, a pipe with interconnected paths is constructed between a source-destination pair, which provides a foundation for the adaptive online routing algorithm. This routing scheme has low complexity, can avoid congested areas, and achieves traffic balancing in the pipe. *SSR* uses geometric forwarding and avoids the flooding of route search messages throughout the network. It only needs a local search in a specific region, whenever the destination moves away.

The simulation results verified that the *SSR* scheme can successfully establish a load-balanced high-throughput leaf-like routing pipe. *SSR* outperformed the single path throughput by up to two times and noticeably reduced the latency. The proposed skeleton structure can also simplify and facilitate the UAV swarm morphing process. The future research will investigate the application of DRL for seeking the optimal routing pipe parameters such as pipe width and trend in a centralized large-scale UAV swarm network.

## ACKNOWLEDGMENT AND DISCLAIMER

The authors acknowledge the Government's support in the publication of this article. Any opinions, findings and conclusions or recommendations expressed in this material are those of the authors and do not necessarily reflect the views of AFRL.

## REFERENCES

- [1] S. Hayat, E. Yanmaz, and R. Muzaffar, "Survey on unmanned aerial vehicle networks for civil applications: A communications viewpoint," *IEEE Commun. Surveys Tuts.*, vol. 18, no. 4, pp. 2624–2661, 4th Quart., 2016.
- [2] D. S. Lakew, U. Sa'ad, N.-N. Dao, W. Na, and S. Cho, "Routing in flying ad hoc networks: A comprehensive survey," *IEEE Commun. Surveys Tuts.*, vol. 22, no. 2, pp. 1071–1120, 2nd Quart., 2020.
- [3] I. Maza, A. Ollero, E. Casado, and D. Scarlatti, "Classification of multi-UAV architectures," in *Handbook of Unmanned Aerial Vehicles*, K. Valavanis and G. J. Vachtsevanos, Eds. Dordrecht, The Netherlands: Springer, 2015, pp. 953–975.
- [4] B. Yun, B. M. Chen, K. Y. Lum, and T. H. Lee, "Design and implementation of a leader-follower cooperative control system for unmanned helicopters," *J. Control Theory Appl.*, vol. 8, no. 1, pp. 61–68, Feb. 2010.
- [5] Y. LeCun, Y. Bengio, and G. Hinton, "Deep learning," *Nature*, vol. 521, pp. 436–444, May 2015.
- [6] V. Mnih, K. Kavukcuoglu, D. Silver, A. Graves, I. Antonoglou, D. Wierstra, and M. Riedmiller, "Playing atari with deep reinforcement learning," 2013, *arXiv:1312.5602*. [Online]. Available: <http://arxiv.org/abs/1312.5602>
- [7] C. Perkins, E. Belding-Royer, and S. Das, *Ad Hoc On-Demand Distance Vector (AODV) Routing*, document RRFC 3561, 2003. [Online]. Available: <https://tools.ietf.org/html/rfc3561>
- [8] J. Bruck, J. Gao, and A. Jiang, "MAP: Medial axis based geometric routing in sensor networks," *Wireless Netw.*, vol. 13, no. 6, pp. 835–853, Dec. 2007.
- [9] Q. Fang, J. Gao, L. J. Guibas, V. de Silva, and L. Zhang, "GLIDER gradient landmark-based distributed routing for sensor networks," in *Proc. IEEE 24th Annu. Joint Conf. IEEE Comput. Commun. Societies*, vol. 1, Dec. 2005, pp. 339–350.
- [10] K. Cai, Z. Yin, H. Jiang, G. Tan, P. Guo, C. Wang, and B. Li, "Onion-Map: A scalable geometric addressing and routing scheme for 3D sensor networks," *IEEE Trans. Wireless Commun.*, vol. 14, no. 1, pp. 57–68, Jan. 2015.
- [11] J. Jiang and G. Han, "Routing protocols for unmanned aerial vehicles," *IEEE Commun. Mag.*, vol. 56, no. 1, pp. 58–63, Jan. 2018.
- [12] Z. Yuan, X. Huang, L. Sun, and J. Jin, "Software defined mobile sensor network for micro UAV swarm," in *Proc. IEEE Int. Conf. Control Robot. Eng. (ICCRE)*, Apr. 2016, pp. 1–4.
- [13] F. Xiong, A. Li, H. Wang, and L. Tang, "An SDN-MQTT based communication system for battlefield UAV swarms," *IEEE Commun. Mag.*, vol. 57, no. 8, pp. 41–47, Aug. 2019.
- [14] W. Qi, Q. Song, X. Kong, and L. Guo, "A traffic-differentiated routing algorithm in flying ad hoc sensor networks with SDN cluster controllers," *J. Franklin Inst.*, vol. 356, no. 2, pp. 766–790, Jan. 2019.
- [15] T. D. E. Silva, E. D. Melo, P. Cumino, D. Rosário, E. Cerqueira, and E. P. D. Freitas, "STFANET: SDN-based topology management for flying ad hoc network," *IEEE Access*, vol. 7, pp. 173499–173514, 2019.
- [16] F. Dai, M. Chen, X. Wei, and H. Wang, "Swarm intelligence-inspired autonomous flocking control in UAV networks," *IEEE Access*, vol. 7, pp. 61786–61796, 2019.
- [17] T. Zeng, M. Mozaffari, O. Semiari, W. Saad, M. Bennis, and M. Debbah, "Wireless communications and control for swarms of cellular-connected UAVs," in *Proc. 52nd Asilomar Conf. Signals, Syst., Comput.*, Oct. 2018, pp. 719–723.
- [18] M. M. Azari, G. Geraci, A. Garcia-Rodriguez, and S. Pollin, "Cellular UAV-to-UAV communications," in *Proc. IEEE 30th Annu. Int. Symp. Pers., Indoor Mobile Radio Commun. (PIMRC)*, Sep. 2019, pp. 1–7.
- [19] D.-T. Ho, E. I. Groth, S. Shimamoto, and T. A. Johansen, "Optimal relay path selection and cooperative communication protocol for a swarm of UAVs," in *Proc. IEEE Globecom Workshops*, Dec. 2012, pp. 1585–1590.
- [20] I. Mahmud and Y.-Z. Cho, "Adaptive hello interval in FANET routing protocols for green UAVs," *IEEE Access*, vol. 7, pp. 63004–63015, 2019.
- [21] G. Gankhuyag, A. P. Shrestha, and S.-J. Yoo, "Robust and reliable predictive routing strategy for flying ad-hoc networks," *IEEE Access*, vol. 5, pp. 643–654, 2017.
- [22] F. Wang, Z. Chen, J. Zhang, C. Zhou, and W. Yue, "Greedy forwarding and limited flooding based routing protocol for UAV flying ad-hoc networks," in *Proc. IEEE 9th Int. Conf. Electron. Inf. Emergency Commun. (ICEIEC)*, Jul. 2019, pp. 1–4.
- [23] P. Jacquet, P. Muhlethaler, T. Clausen, A. Laouiti, A. Qayyum, and L. Viennot, "Optimized link state routing protocol for ad hoc networks," in *Proc. IEEE Int. Multi Topic Conf. Technol. 21st Century*, Dec. 2001, pp. 62–68.
- [24] S. Rosati, K. Kruzelecki, L. Traynard, and B. Rimoldi, "Speed-aware routing for UAV ad-hoc networks," in *Proc. IEEE Globecom Workshops (GC Wkshps)*, Dec. 2013, pp. 1367–1373.
- [25] Y. Zheng, Y. Jiang, L. Dong, Y. Wang, Z. Li, and H. Zhang, "A mobility and load aware OLSR routing protocol for UAV mobile ad-hoc networks," in *Proc. Int. Conf. Inf. Commun. Technol. (ICT)*, 2014, pp. 1–7.
- [26] C. Pu, "Link-quality and traffic-load aware routing for UAV ad hoc networks," in *Proc. IEEE 4th Int. Conf. Collaboration Internet Comput. (CIC)*, Oct. 2018, pp. 71–79.
- [27] M. Y. Arifat and S. Moh, "Location-aided delay tolerant routing protocol in UAV networks for post-disaster operation," *IEEE Access*, vol. 6, pp. 59891–59906, 2018.

- [28] C. Yin, Z. Xiao, X. Cao, X. Xi, P. Yang, and D. Wu, "Enhanced routing protocol for fast flying UAV network," in *Proc. IEEE Int. Conf. Commun. Syst. (ICCS)*, Dec. 2016, pp. 1–6.
- [29] C. Pu, "Stochastic packet forwarding algorithm in flying ad hoc networks," in *Proc. MILCOM IEEE Mil. Commun. Conf. (MILCOM)*, Nov. 2019, pp. 490–495.
- [30] N. Toorchi, F. Hu, S. Pudlewski, E. Bentley, and S. Kumar, "Volcano routing: A multi-pipe high-throughput routing protocol with hole avoidance for multi-beam directional mesh networks," *IEEE Trans. Mobile Comput.*, vol. 19, no. 12, pp. 2981–2996, Dec. 2020.
- [31] P. Smith, R. Hunjet, A. Aleti, and J. C. Barca, "Adaptive data transfer methods via policy evolution for UAV swarms," in *Proc. 27th Int. Telecommun. Netw. Appl. Conf. (ITNAC)*, Nov. 2017, pp. 1–8.
- [32] N. Toorchi, J. Chakareski, and N. Mastronarde, "Fast and low-complexity reinforcement learning for delay-sensitive energy harvesting wireless visual sensing systems," in *Proc. IEEE Int. Conf. Image Process. (ICIP)*, Sep. 2016, pp. 1804–1808.
- [33] A. M. Koushik, F. Hu, and S. Kumar, "Deep  $Q$ -learning-based node positioning for throughput-optimal communications in dynamic UAV swarm network," *IEEE Trans. Cognit. Commun. Netw.*, vol. 5, no. 3, pp. 554–566, Sep. 2019.
- [34] M. Shapira and A. Rappoport, "Shape blending using the star-skeleton representation," *IEEE Comput. Graph. Appl.*, vol. 15, no. 2, pp. 44–50, Mar. 1995.
- [35] E. L. Porteus, *Foundations of Stochastic Inventory Theory*. Palo Alto, CA, USA: Stanford Univ. Press, 2002.
- [36] M. L. Puterman, *Markov Decision Processes: Discrete Stochastic Dynamic Programming*. Hoboken, NJ, USA: Wiley, 2014.
- [37] R. S. Sutton and A. G. Barto, *Reinforcement Learning: An Introduction*. Cambridge, MA, USA: MIT Press, 2018.
- [38] I. Goodfellow, Y. Bengio, and A. Courville, *Deep Learning*. Cambridge, MA, USA: MIT Press, 2016.
- [39] B. Debowski, P. Spachos, and S. Arcibi, "Q-learning enhanced gradient based routing for balancing energy consumption in WSNs," in *Proc. IEEE 21st Int. Workshop Comput. Aided Model. Design Commun. Links Netw. (CAMAD)*, Oct. 2016, pp. 18–23.
- [40] K. Wallace, *Cisco IP Telephony Flash Cards: Weighted Random Early Detection (WRED)*. Indianapolis, IN, USA: Cisco Press, 2004.



**NILOOFAR TOORCHI** (Member, IEEE) is currently pursuing the Ph.D. degree with the Department of Electrical and Computer Engineering, The University of Alabama, Tuscaloosa, AL, USA. Her research interests include wireless networking, QoS-aware and cross-layer protocols, UAV swarming, artificial intelligence, software defined networking, programmable switches, and others.



**FEI HU** (Member, IEEE) received the Ph.D. degree in signal processing from Tongji University, Shanghai, China, in 1999, and the Ph.D. degree in electrical and computer engineering from Clarkson University, New York, NY, USA, in 2002. He is currently a Professor with the Department of Electrical and Computer Engineering, The University of Alabama, Tuscaloosa, AL, USA. He has published more than 200 journal/conference papers and book (chapters) in the field of wireless networks and machine learning. His research interests include wireless networks, machine learning, big data, network security, and their applications. His research has been supported by U.S. NSF, DoE, DoD, Cisco, and Sprint.



**ELIZABETH SERENA BENTLEY** (Member, IEEE) received the B.S. degree in electrical engineering from Cornell University, Ithaca, NY, in 1999, the M.S. degree in electrical engineering from Lehigh University, Bethlehem, PA, in 2001, and the Ph.D. degree in electrical engineering from The State University of New York at Buffalo, in 2007. Since 2008, she has been employed by the Air Force Research Laboratory, Information Directorate, Rome, NY. Her research interests include cross-layer optimization, directional networking, wireless video transmission, and network modeling and simulation.



**SUNIL KUMAR** (Senior Member, IEEE) received the Ph.D. degree in electrical and electronics engineering from the Birla Institute of Technology and Science, Pilani, India, in 1997. He is currently a Professor and a Thomas G. Pine Faculty Fellow with the Electrical and Computer Engineering Department, San Diego State University, CA, USA. He has published more than 160 research papers in refereed journals and conferences, seven books, and book chapters. His research interests include robust video compression, including H.264 and HEVC, and QoS-aware and cross-layer protocols for wireless networks. His research has been funded by the U.S. Air Force Research Laboratory, the National Science Foundation, the Department of Energy, California Energy Commission, and industry.

• • •