Formation Tracking in Sparse Airborne Networks

Yongcai Wang, *Member, IEEE*, Tianyuan Sun, Guoyao Rao, and Deying Li, *Member, IEEE*

***Abstract*— A swarm of unmanned air vehicles (UAVs) may form a dynamic 3-D network whose topology changes frequently. Tracking the geometric formation of the network is a critical problem. Recent advantage of wireless ranging technologies (e.g., ultrawideband) enables inter-UAV distance measurement up to hundreds meters with errors in centimeter level. This makes it possible to track the network topology by the partially measured distance matrix among the UAVs, which is known as the *formation tracking* problem. But the measured distances are generally sparse and noisy, and the topology of UAV network is changing continuously. These cause the formation tracking highly challenging. Existing methods are generally fragile to the measurement noises and network sparsity. This paper exploits a fact that well-connected subcomponents, whose local structures can be calculated reliably, exist widely because the unevenness of node distribution in sparse networks. Therefore, a weighted component stitching (WCS) method to find the reliable components and stitch their local structures with weights is proposed for calculating the formation of the network accurately. In particular, we propose efficient two-center four-vertex-connected star-graph (2-4-star) detection and merging algorithms to extract the reliable global rigid components. A WCS algorithm and a weighted component-based Kalman filter algorithm with complexity both $O(n^3)$ are proposed for robust formation tracking in n vertex UAV networks. Extensive experiments were conducted, showing that the proposed methods can improve the formation tracking accuracy 21%–48% over existing state-of-the-art methods, especially in sparse, noisy UAV networks under different parameter settings.**

***Index Terms*— Airborne networks, formation tracking, network localization, dynamic, 3D, UAV swarm, rigidity, graph realization.**

I. Introduction

RECENT advantages of unmanned air vehicles (UAVs), quad-copters, and smart drones have enabled swarming of UAVs [1], [2] to carry out specific military or civil applications. In flying a swarm of UAVs, tracking and controlling the formation of the 3D UAV network is critically important [3]–[7] for not only formation control [8], collision avoidance [9], but also for topology-based communication scheduling and optimization [10]–[13].

Manuscript received January 2, 2018; revised May 1, 2018; accepted June 5, 2018. Date of publication August 10, 2018; date of current version November 21, 2018. This work was supported in part by the National Natural Science Foundation of China Grant No. 11671400, 61672524. The Fundamental Research Funds for the Central University, and the Research Funds of Renmin University of China, 2015030273. *(Corresponding author: Yongcai Wang.)*

The authors are with the Department of Computer Sciences, Renmin University of China, Beijing 100872, China (e-mail: ycw@ruc.edu.cn)

Color versions of one or more of the figures in this paper are available online at http://ieeexplore.ieee.org.

Digital Object Identifier 10.1109/JSAC.2018.2864374

One way to get location information is to equip GPS (GLONASS, or Beidou) on the small UAV devices, which is not proper for the limited battery and for indoor applications, such as the smart tiny drones flying in buildings [14], [15]. One potential way is to exploit recent advantages of wireless localization technologies [16]. For example, the Ultra Wide Band (UWB) [17], [18] can measure inter-node distance up to hundreds of meters with error less than 10 centimeters in 10Hz updating rate. This enables the possibility to track the formation of a drone swarm by the inter-distance measurements among the drones. In this way, the drones can be built in smaller size and be more energy efficient.

If the drones in a swarm collaborate to form a dynamic 3D network, it will cover a large area. For keeping safe distance for collision avoidance, and for the limited ranging scope of the wireless devices, the measured distance matrix among the drones is generally sparse and the distances are noisy. This poses great challenges to the real-time formation tracking of the 3D UAV networks.

Existing studies generally approach 3D formation tracking from two aspects: 1) on the solution uniqueness problem, and 2) on the formation tracking algorithms. We give a detailed introduction to these related theories and algorithms in Section II-B. Remaining challenges include: (1) an exact combinatorial characterization of generic rigidity is currently not available in $\mathbb{R}^d (d \geq 3)$ [19], which leaves difficulty to check solution uniqueness of UAV formation tracking. (2) Existing algorithms generally produce accurate formation results in densely connected networks. However, their common drawbacks are being highly sensitive to the network sparseness and ranging noises. The formation calculation accuracy collapses when the distance measurements become sparse and when the ranging noises increase. But these are general scenarios in practical UAV networks.

In sparse networks, the degradation of the formation calculation accuracy is mainly because the under-determinant optimization. This paper proposes novel methods to address this challenge. At first, we observe that: for the uneven node distribution, even in highly sparse networks, there are densely connected components whose local structure can be calculated rather reliably. By a Weighted Component Stitching (WCS) method, the reliable local structure of these components can benefit the formation calculation of other parts of the graph. The key problems are: 1) how to find these reliable components efficiently since it encounters combinatorial exploration of the subgraphs; 2) how to evaluate a component's reliability, and 3) how to design the weighted component stitching method.

To solve these problems, at first, an efficient algorithm is proposed to search the *double-center four-vertex-connected subgraphs (2-4-star)* in the UAV network, which is a special global rigid component in $\mathbb{R}^3$ [20]. The 2-4-star is sufficient for having a unique local formation in noiseless condition but may also produce unsatisfied result when the edges are noisy. Secondly, we propose a redundant ratio (RR) metric to evaluate the reliability of a component. The intuition is that the component with higher redundant ratio produces more accurate local coordinates by having more distance constraints in the local optimization problem. Then RR is utilized to merge the 2-4-stars into Local Mostly Condensed Components (LMCC) until the RR of the merged component cannot be further improved. Thirdly, a weighted component-stitching algorithm (WCS) is proposed, which assigns weights to the locally calculated coordinates in each LMCC by the LMCC's RR metric. The local coordinates in the LMCCs are then synchronized to produce the network's global coordinates by iteratively adjusting the rotation and transformation matrices of the LMCCs, and the global coordinates. The weighted synchronization helps to enhance the benefits of the condensed components and reduce the impacts of the sparse parts. Further, an integration of WCS and Kalman Filter, i.e., (WCKF) is presented for robust formation tracking. The detailed contributions are as following:

1) An efficient algorithm is designed to detect the 2-4-stars in the UAV network.
2) An edge redundant ratio (RR) metric is proposed, and merging algorithm is developed to merge the 2-4-stars to LMCC. We prove two components can be merged only if they share more than 4 vertices and the merged component remains global rigid.
3) A weighted component stitching (WCS) algorithm is developed to utilize the reliable components to improve formation tracking accuracy. Different weighting schemes are examined.
4) A weighted-component based Kalman filter algorithm (WCKF) is proposed to integrate the WCS into the Kalman Filter by using the local coordinates in components as observations.
5) The WCS algorithm, WCKF algorithm are extensively compared with current component stitching algorithm (ARAP), centralized algorithms (SDP), and EKF algorithm. 21% to 48% accuracy improvements are shown in both static and dynamic networks by simulations in Unity under different parameter settings.
6) The computation complexity of WCS and WCKF are shown $O(hn^3)$ and $O(D^3n^3)$ respectively, (notations can be referred to Table I), being comparable to the traditional ARAP and EKF algorithms.

Note that we study mainly the anchor-less case, focusing on the formation or relative structure calculation. If there are more than four non-collinear anchor nodes, the relative formation can be fixed by the anchors to the earth frame.

The remaining sections of this paper are organized as following. Problem model and related works are introduced in section II. Preliminaries are introduced in Section III. The 2-4-star detection algorithm, RR metric, and LMCC merging algorithmare introduced in section IV. The WCS and WCKF algorithms are presented in Section V. Complexity analysis and extensive comparisons are presented in Section VI. The paper is concluded with remarks in Section VII.

TABLE I
LIST OF NOTATIONS

n	number of vertices	m	number of edges
$\mathbf{G}$	graph	$\mathbf{R}_i$	rotation matrix of $\mathbf{G}_i$
$\mathbf{V}$	vertex set	$\mathbf{R}_k^c$	rotation matrix of $\mathbf{G}_k^c$
$\mathbf{E}$	edge set	$m(t)$	number of edges at t
$\mathbf{G}_i$	patch i	$\mathbf{D}(t)$	distance matrix at t
$\mathbf{G}_k^c$	component k	$r(\mathbf{G})$	redundant ratio of $\mathbf{G}$
$\mathbf{p}_i(t)$	location of node i at t	R	the maximum range
$\mathbf{q}_i(t)$	local location of i in $\mathbf{G}_i$	$\mathbf{X}_i(t)$	state of node i at t
$\mathbf{q}_i^{c,k}(t)$	local location of i in $\mathbf{G}_k^c$	$f(r)$	weight function
$d_{i,j}(t)$	distance between i, j at t	$\mathbf{A}$	state matrix
h	# of iterations in WCS	D	maximum node degree
$\mathbf{H}$	observation matrix	$\mathbf{K}$	Kalman gain
$\mathbf{P}$	variance matrix	$\mathbf{Q}$	observation noise matrix
$\mathbf{B}$	control gain	$\mathbf{z}$	observation vector

II. BACKGROUND AND PRELIMINARIES

A. Problem Model

The problem is to track the formation of a dynamic airborne network in $\mathbb{R}^3$ by partially measured inter-node distances. Let set $\mathbf{V}$ denote the UAV nodes and $|\mathbf{V}| = n$. The locations of these UAVs may change over time. Their locations at time t are denoted by $\{\mathbf{p}_1(t), \cdots, \mathbf{p}_n(t)\}$, where $\mathbf{p}_i(t) = [x_i(t), y_i(t), z_i(t)]^T$ is the location of a node i at t. By equipped wireless ranging device, such as UWB module, we assume two UAVs can measure real-time inter-distance $d_{i,j}(t)$ when $d_{i,j}(t) \leq R$, where R is the maximum ranging scope.

A graph $\mathbf{G}(t) = (\mathbf{V}, \mathbf{E}(t))$ can be constructed according to the measured edges at time t, in which $(i,j) \in \mathbf{E}(t)$ if $d_{i,j}(t) \leq R$. $m(t) = |\mathbf{E}(\mathbf{t})|$ is the number of edges at t. The goal of formation calculation at t is to find 3-dimensional embedding $\{\mathbf{p}_1(t), \ldots, \mathbf{p}_n(t)\} \in \mathbb{R}^3$ such that the following *stress function* is minimized [21].

$$Stress(\mathbf{p}_1(t), \ldots, \mathbf{p}_n(t)) = \sum_{(i,j)\in\mathbf{E}(t)} (\|\mathbf{p}_i(t) - \mathbf{p}_j(t)\|_2 - d_{ij}(t))^2 \quad (1)$$

For continuously tracking, the state and the motion model of UAV is used.

Definition 1 (State and Motion Model of a Node): The state of a node is defined by its real-time location and velocity. $\mathbf{X}_i(t) = [\mathbf{p}_i(t), \mathbf{v}_i(t)] = [x_i(t), y_i(t), z_i(t), \dot{x}_i(t), \dot{y}_i(t), \dot{z}_i(t)]^T$. *Its motion model is represented by* $\mathbf{X}_i(t+1) = \mathbf{A}_i\mathbf{X}_i(t) + \mathbf{B}_i\mathbf{u}_i(t) + \xi$, *where* $\mathbf{A}_i$ *is the system matrix,* $\mathbf{B}_i$ *is the control gain, and* ξ *is the system noise vector. The control vector is carried out by the accelerations,*

i.e., $\mathbf{u}_i(t) = [\ddot{x}_i(t), \ddot{y}_i(t), \ddot{z}_i(t)]$ *given by the drone's engine.*

$$\mathbf{A}_i = \begin{bmatrix} 1 & 0 & 0 & \Delta t & 0 & 0 \\ 0 & 1 & 0 & 0 & \Delta t & 0 \\ 0 & 0 & 1 & 0 & 0 & \Delta t \\ 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 \end{bmatrix}$$

$$\mathbf{B}_i = \begin{bmatrix} \frac{\Delta t^2}{2} & & \\ & \frac{\Delta t^2}{2} & \\ & & \frac{\Delta t^2}{2} \\ \Delta t & & \\ & \Delta t & \\ & & \Delta t \end{bmatrix}$$

Δt is the time interval between two updates. We assume the control is transformed from body frame to the global frame by the UAV's attitude information, e.g., by gyroscope [22], [23]. Gaussian noises are assumed for the motion model with variance matrix $\mathbf{R}$. The goal of the formation tracking problem is to maximize the posterior probability of the drones' states at t based on the distance measurements and controls obtained from 1 to t:

$$\mathbf{X}^*(t) = \arg\max_{\mathbf{X}(t)} \; p\left(\mathbf{X}(t) | \mathbf{X}(1:t-1),\; \mathbf{D}(1:t),\; \mathbf{u}(1:t)\right) \tag{2}$$

where $1:t$ is an abbreviation of $\{1, \cdots, t\}$. The notations of the model and the variables used in this paper are listed in Table I.

B. Related Works

Formation tracking provides real-time topology information of the UAV swarm, which is a crucial foundation for many applications.

1) Formation Control: At first, formation control and collision avoidance are highly dependent on the formation information. Dong *et al.* [5] and Dong [24] exerted the necessary and sufficient conditions to achieve time-varying formations and proposed formation control protocols. Seo *et al.* [9] proposed to use line-of-sight vectors and relative velocity vectors to determine available plane and direction for UAV collision avoidance. Monajjemi *et al.* [25] studied commanding a team of UAVs with a vision-mediated gestural interface. More related works on formation control can be referred to a recent survey [26].

2) UAV Communication Optimization: Transmission scheduling and data routing in UAV networks are required to adapt to high mobility, dynamic topology, and changing link quality. Formation is the key to predict neighborhood links and link qualities. Ny *et al* [10]. proposed a primal-dual gradient descent algorithm to jointly optimize the steady-state positions of UAVs to route packets through the network for supporting the communication rates desired for the applications. Wu *et al.* [11] proposed mixed integer nonconvex optimization model to jointly optimize the UAV trajectory, communication power, and user association to serve ground users by UAV-mounted wireless stations. Kwon and Hailes [12] proposed dynamic positioning algorithm to drive UAVs automatically to work as communication relays to best serve participating ground nodes. Zeng *et al.* [13] studied the UAV trajectory optimization problem to minimize the mission completion time in transmitting files to ground nodes by multicast. Formation changes also impact the multi-hop transmission delay and packet loss rate. More related work can be referred to the survey Gupta *et al.* [8].

3) Sensing Technologies: Sensing and ranging technologies among UAVs highly impact the formation calculation algorithms. Schiano and Giordano [27] studied bearing rigidity maintenance for formations of quadrotor UAVs based on the measurements from onboard cameras. Michieletto *et al.* [6] studied the necessary and sufficient conditions for forming rigidity graph when devices are gathering bearing measurements from their neighbors. Wang *et al.* [7] used Ultra-Wind Band (UWB) based indoor navigation system (INS) to achieve time-varying formations of four omni-directional mobile robots. Tiemann and Wietfeld [28] studied time-difference of arrival (TDOA) based ultra-wideband (UWB) indoor multiple UAV localization. Perez-Grau *et al.* [29] studied multi-modal mapping and localization of unmanned aerial robots based on UWB and RGB-D sensing. Wang *et al.* [30] studied simultaneously narrow-band ultrasound ranging (in chorus mode), which has centimeter level accuracy and high location refreshing rate, which can be used in UAV indoor localization. Both ultrasound and UWB has the advantages of low cost, low power, and accurate ranging in open space. UWB has larger ranging distance, which is the considered ranging technology in this paper.

4) Formation Calculation Algorithms: Formation calculation investigates the problem in (1), which calculates formation using distance matrix measured among vertices at one time instance, which is reduced to a graph realization problem. There are mainly two categories of graph realization algorithms.

a) Centralized optimization algorithms: The representative centralized algorithms include Multi-dimensional Scaling (MDS) [31], SMACOF [32] and Semi-definite programming (SDP) [33], etc. MDS [31] solves the localization problem by matrix decomposition, which requires the network to be a complete graph. SMACOF [32] constructs a quadratic function to approximate the *stress function* in each iteration step. SDP [33] relaxes the constraints of the distance matrix to approach localization by semidefinite programming. *These algorithms perform well in densely connected networks, but their performances degrade quickly as the increasing of network sparseness and measurement noises.*

b) Component stitching (CS) method: To tolerate the network sparseness and measurement noises, a series of component stitching algorithms were proposed for graph realization. The representative algorithms include: RCGR [34], AAAP [21], ARAP [21], ASAP [35]. RCGR proposes to find robust components with low flipping possibility by evaluating topology robustness against ranging noises using a ranging

sensitivity matrix (RSM) [34]. ARAP and ASAP adopt iterative component stitching method (CS) to fight against ranging noises and edge sparseness. *The CS-based methods use a local calculation and global synchronization process, which provide the state-of-the-art for tolerating ranging noises and link sparseness (will be detailed in the next section). But how to exploit structural unevenness to further tolerate the network sparsity and ranging noises is not reported in previous studies.*

5) Formation Tracking Algorithms: Formation tracking investigates the problem in (2). Extended Kalman Filter (EKF), Unsented Kalman Filter (UKF) [36] and Particle Filter (PF) [37] are the most widely used methods for formation tracking. Their common routine is to predict the states of nodes based on the motion model, and update the states by measured distance matrix. The tracking performances are highly dependent on the number of measurements and the noise characteristics. *But the states are updated only one time per interval, which are shown sensitive to the measurement sparsity (will be detailed in Section III).*

In this paper, we in particular investigate accurate and efficient formation tracking for UAV networks with sparse and noisy edges, which are general scenario in applications.

III. Experiments on Existing Methods

A. Component Stitching (CS) Methods

We detail the most related component stitching (CS) methods [21], [35] for self-containment.

Definition 2 (Patch): A patch $\mathbf{G}_i$ is the subgraph induced by node i and its neighbors. Let $\mathbf{N}_i = \{j : (i,j) \in \mathbf{E}\} \cup \{i\}$ be node i's neighbors and itself; $\mathbf{G}_i$ is the subgraph of $\mathbf{G}$ induced by $\mathbf{N}_i$.

A common routine of component stitching algorithm is as following:

1) Divide the graph into patches $\{\mathbf{G}_1, \mathbf{G}_2, \cdots, \mathbf{G}_n\}$;

2) In each patch $\mathbf{G}_i$, node coordinates are calculated by the inter-node distances in a patch's local coordinate system, carried out by MDS [31] or SMACOF [32].

3) The local coordinates of nodes in different patches are synchronized to estimate coordinates in a global coordinate system, by solving an optimization problem.

B. Component Synchronization

Let $\mathbf{q}_j^i$ be node j's locally calculated coordinates in patch $\mathbf{G}_i$. Let $\mathbf{P} = \{\mathbf{p}_1, \mathbf{p}_2, \ldots, \mathbf{p}_n\}$ denote the unknown global coordinates of nodes. $\mathbf{q}_i^i$ is the local coordinates of i in $\mathbf{G}_i$. Then $\mathbf{p}_i$ and $\mathbf{q}_i^i$ is related by a rotation matrix $\mathbf{R}_i$ and a transition vector $\mathbf{T}_i$.

$$\mathbf{p}_i = \mathbf{R}_i \mathbf{q}_i^i + \mathbf{T}_i \tag{3}$$

Note that $\mathbf{R}_i$ is a 3×3 rigid transformation matrix, and $\mathbf{T}_i$ is a 3×1 translation vector. The synchronization is to calculate $\mathbf{R}_i$ and $\mathbf{T}_i$ for each component $\mathbf{G}_i$. $\mathbf{T}_i$ can be eliminated if a node j is also in $\mathbf{N}_i$, by utilizing $\mathbf{p}_i - \mathbf{p}_j = \mathbf{R}_i \mathbf{q}_i^i - \mathbf{R}_i \mathbf{q}_j^i$. So the synchronization problem becomes to minimize the following cost function [21]:

$$F = \min_{\mathbf{P}, \mathbf{R}_1, \ldots, \mathbf{R}_n} \{ \sum_{i=1}^{n} \sum_{j \in \mathbf{N}_i} \| (\mathbf{p}_i - \mathbf{p}_j) - \mathbf{R}_i (\mathbf{q}_i^i - \mathbf{q}_j^i) \|^2 : \mathbf{R}_i^{\mathrm{T}} \mathbf{R}_i = I \} \tag{4}$$

Eqn. (4) is generally solved using an iterative local-global method. In the local phase, $\mathbf{P}$ is assumed fixed to calculate each $\mathbf{R}_i$. In the global phase, all the $\mathbf{R}_i$ are assumed fixed to solve $\mathbf{P}$. The process repeats until convergence of $\mathbf{P}$.

C. Experiments Against Sparsity and Noises

We firstly investigate how the existing algorithms can tolerate the network sparseness and ranging noises. Simulations were conducted for ARAP, SDP and EKF for networks with increasing sparsity and ranging noises. In simulation, the network contains 60 nodes, randomly distributed in a $600(m) \times 100(m) \times 100(m)$ 3D area. We vary the maximum ranging radius R from $115(m)$ to $55(m)$ to check how the formation errors of these three algorithms vary as the network becomes more and more sparse. The network becomes sparser as the maximum ranging radius decreases. The distance measurements are assumed impacted by Gaussian noises, i.e, $d_{i,j} \sim \mathcal{N}(d_{i,j}^0, (\sigma d_{i,j}^0)^2)$. We vary σ from 0.01 to 0.3 to check how the algorithms are impacted by the ranging noises.

Fig.1e) summarizes the formation calculation errors when R varied from $115(m)$ to $55(m)$ and $\sigma = 0.01$. The average node degree in the graph $degree(\mathbf{G})$ decreases almost linearly as the graphs become sparse. In dense cases, i.e., $R > 100$, EKF, SDP and ARAP all provided good realization results with small errors. But the localization errors of EKF and SDP increase sharply when the network becomes sparse. ARAP shows more tolerance to the network sparseness for the slowly increasing of the localization error with the network's sparseness. But its errors in sparse networks are also large.

Fig.1f) shows the formation errors when σ varies from 0.01 to 0.3 for $R = 100$ *(dense case)* and $R = 65$ *(sparse case)* respectively. In dense case, the formation errors of SDP and ARAP increase almost linearly with the ranging noises. EKF is more noise sensitive. In sparse case, the impacts of edge sparseness dominate the formation errors for all the three algorithms.

Fig.1a) to Fig.1d) visually compare the formation calculation results of different algorithms with the network formation groundtruth when $R = 65, \sigma = 0.01$. Fig.1a) is the ground truth. Fig.1b)c)d) are the results of EKF, SDP, ARAP respectively. It can be seen that when the network is sparse, i.e., $R = 65$, EKF and SDP provides unsatisfactory results. ARAP gives reasonable results in some dense parts, but the overall synchronized result is still not good.

Motivated by these observations, this paper investigates topology unevenness in the graph to pursue accurate formation calculation in sparse networks.

IV. Weighted Component Stitching (WCS)

A. Mathematical Model of WCS

Instead of regularly dividing the patches as stated in Definition 2, WCS considers to detect reliable components in

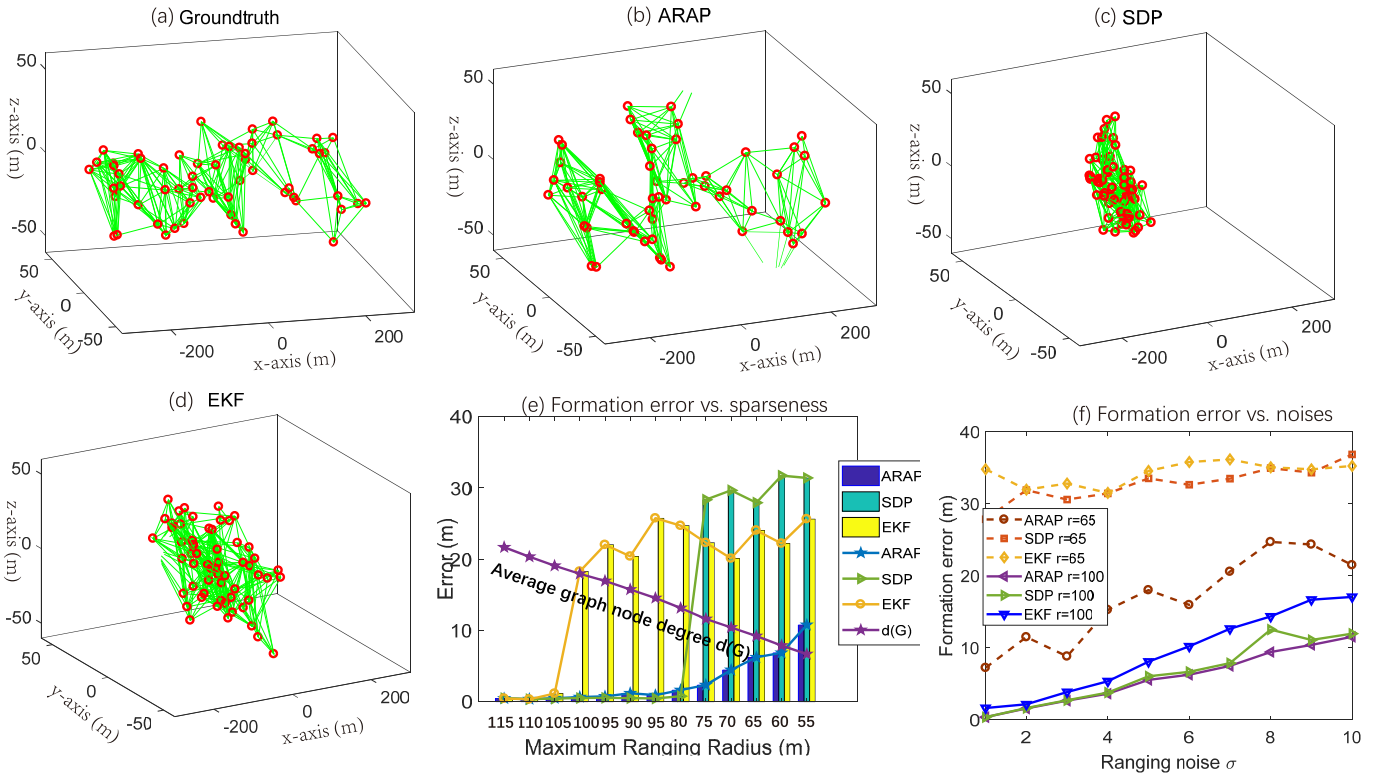


Fig. 1. Illustration of current algorithms' performances.

the graph. Suppose $\mathbf{G}^c = \{\mathbf{G}_1^c, \mathbf{G}_2^c, \mathbf{G}_3^c, \ldots, \mathbf{G}_l^c\}$ be the detected reliable components which are ordered by redundant ratio (RR, will be defined soon). Let $\{r_1, r_2, \cdots, r_l\}$ be the redundant ratios of these components, and $\mathbf{R}_k^c$ be the rotation matrix of the component $\mathbf{G}_k^c$. The local coordinates of node i in component $\mathbf{G}_k^c$ are calculated by MDS and SMACOF, which are denoted by $\mathbf{q}_i^{c,k}$. Let $w_k = f(r_k)$ be the weight function calculated for component $\mathbf{G}_k^c$. We propose a weighted component stitching (WCS) framework to synchronize the components based on the RR metrics:

$$F = \min_{\mathbf{P},\mathbf{R}_1^c,\ldots,\mathbf{R}_l^c} \{\sum_{k=1}^{l} \sum_{(i,j)\in\mathbf{G}_k^c} f(r_k)\|(\mathbf{p}_i - \mathbf{p}_j) - \mathbf{R}_k^c(\mathbf{q}_i^{c,k} - \mathbf{q}_j^{c,k})\|^2 \mathbf{R}_k^{c\mathrm{T}}\mathbf{R}_k^c = I\} \quad (5)$$

Although having a similar form as (4), the reliable component detection is no longer trivial in WCS. It involves to detect the mostly condensed subgraph in $\mathbf{G}$, which is similar to the quasi-clique detection problem, which is known NP-complete even in 2D networks [38]. We propose to detect special global rigid, 2-centered, 4-vertex-connected subgraphs (2-4-stars) efficiently. Then these 2-4-stars are merged to form reliable components adaptively.

B. Global Rigid Property

The necessary and sufficient condition for a graph $\mathbf{G}$ having unique realization is that $\mathbf{G}$ is global rigid. *Global rigid* is interpreted by the following three definitions.

Definition 3 (Isometry [39]): In three-dimensional Euclidean space, two geometric graphs are congruent (or related by an isometry), if and only if they are related by a rigid motion (translation or rotation), or a composition of a rigid motion and a reflection.

Let the term *framework* (in d-space) denote a triple $(V, E, \mathbf{P})$, where (V, E) is the graph and $\mathbf{P}$ represents the vertex coordinates in d-dimension space.

Definition 4 (Rigid in $\mathbb{R}^3$ [39]): A framework $(V, E, \mathbf{P})$ is rigid if the continuous motions of the vertices in $\mathbb{R}^3$ that preserve the edge lengths arise only from its isometric frameworks in $\mathbb{R}^3$.

Definition 5 (Global Rigid in $\mathbb{R}^3$ [39]): A framework $(V, E, \mathbf{P})$ is global rigid in $\mathbb{R}^3$ if every realization $(V, E, \mathbf{Q})$ with the same edge lengths arises from an isometry of $(V, E, \mathbf{P})$ in $\mathbb{R}^3$.

Although global rigid graphs have unique realization, unfortunately, the combinatorial conditions for checking a graph's global rigidity in $\mathbb{R}^3$ remains open [19], [19]. But recent results [20] show that some special graphs are global rigid.

C. Finding Global Rigid 2-4-Star Subgraphs

The recent results [20] show that there are special graph structures, which are global rigid in $\mathbb{R}^d$ $(d \geq 3)$. In particular, in $\mathbb{R}^3$ the 2-star, 4-vertex-connected subgraphs, called **2-4-stars** are global rigid.

Definition 6 (k-Star Graph [20]): A k-star graph is a connected component which contains at least k vertices (centers) that are connected to all the other nodes in the component.

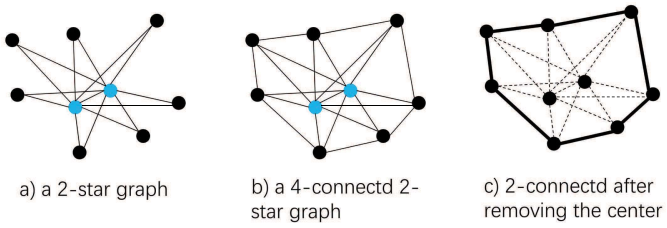


Fig. 2. Illustration of (a) 2-star graph; (b) 4-connected, 2-star graph; (c) 2-connected after removing the center.

The motivation of k-star graph is that the 1-hop neighborhood graph may have multiple centers (especially in random geometric graphs). As shown in Fig.2a), it is very close to the one-hop neighborhood of a node, but has 2 centers. Unlike the conventional definitions of star graphs, the edges between non-central nodes are included in the k-star graph.

Theorem 1: A 2-star graph is generic global rigid in $\mathbb{R}^3$ ***if and only if*** *it is 4-vertex-connected.*

We refer readers to [20] for the proof. The 4-vertex-connected graphs have four vertex disjoint paths between any two nodes. Fig.2b) gives an example of 2-4-star. Because a 2-4-star is global rigid, it has the unique formation solution in noiseless case. Note that the 2-4-stars can be detected by neigborhood search. Theorem 2 provides hint on how to efficiently detect the 2-4-stars.

Theorem 2: Give a k*-star graph* $\mathbf{G}(\mathbf{V}, \mathbf{E})$ *and the center nodes* $\mathbf{V}_{star} = \{v_1^{star}, v_2^{star}, \ldots, v_k^{star}\}$, $\mathbf{G}$ *is* L*-vertex-connected if* $(\mathbf{G} - \mathbf{V}_{star})$ *is* $(L-k)$*-vertex-connected.*

Proof: In k-star graph $\mathbf{G}$, $\forall v_i, v_j \in \mathbf{V}$, we need to find L disjoint paths to connect these two nodes. When $v_i, v_j \in \mathbf{V} - \mathbf{V}_{star}$, there are k disjoint paths connecting these two nodes, $\{v_i \to v_1^{star} \to v_j, v_i \to v_2^{star} \to v_j, \ldots, v_i \to v_k^{star} \to v_j\}$. And the graph $(\mathbf{G} - \mathbf{V}_{star})$ is $(L-k)$-vertex-connected, so there are $(L-k)$ disjoint paths in $(\mathbf{G} - \mathbf{V}_{star})$. Finally, we get L disjoint paths connecting v_i and v_j in $\mathbf{G}$.

If $v_i \in \mathbf{V}_{star}$ and $v_j \in \mathbf{V} - \mathbf{V}_{star}$, we can find k disjoint paths by center nodes. Then we select any $L-k$ nodes in $\mathbf{V} - \mathbf{V}_{star}$, denoted by $v_1', v_2', \ldots, v_{L-k}'$, and form $L-k$ disjoint paths$\{v_i \to v_1' \to \ldots \to v_j, v_i \to v_2' \to \ldots \to v_j, \ldots, v_i \to v_{L-k}' \to \ldots - v_j\}$.

If $v_i, v_j \in \mathbf{V}_{star}$, $\forall v_1, v_2, \ldots, v_L \in V - \{v_i, v_j\}$ can form L disjoint paths, $\{v_i \to v_1' \to v_j, v_i \to v_2' \to v_j, \ldots, v_i \to v_L' \to v_j\}$. □

Based on Theorem 2, in any 2-4-star, after removing the two centers, the remaining nodes should be 2-connected. This is because the two centers connect all the other nodes, which provide two paths between any pair of nodes. An 2-4-star detection algorithm, i.e., Algorithm1 is therefore designed based on this fact.

The algorithm contains two steps: 1) finding all the 2-star subgraphs by examining each edge (i, j); 2) verifying whether the 2-star graph is 4-vertex connected. The first step can be accomplished in $O(mn)$ time. For each edge $e = (i, j)$ in the graph $\mathbf{G}$, we find the node v_i, v_j and their common neighbor nodes $\mathbf{V}_e = \{v_l | (i,l) \in \mathbf{E} \text{ and } (l,j) \in \mathbf{E}\}$. We then find the graph induced by the common neighbors of $\{v_i, v_j\}$, denoted by $\mathbf{S}_e$. Then $\mathbf{S}_e$ is divided to find its 2-connected components. $\mathbf{S}^2 = \{\mathbf{S}_1^2, \cdots, \mathbf{S}_l^2\}$. The induced graphs of the vertices in the 2-connected components together with $\{v_i, v_j\}$ are merged to output the 2-4-star graph $\mathbf{S}_\mathbf{e}^\mathbf{c}$ centered at $\{v_i, v_j\}$.

Algorithm 1 2-4-Stars Detection

```
Input: A graph G = (V, E)
Output: S^c = {S_1^c, S_2^c, ..., S_K^c}
S^c ← ∅;
for each edge e = (i, j) ∈ E do
    V_e = {v_l | (i, l) ∈ E and (l, j) ∈ E}, S_e^c ← ∅;
    S_e is the subgraph induced by V_e;
    Divide S_e into 2-connected subgraphs S^2 = {S_1^2, ..., S_l^2};
    for each S_i^2 ∈ S^2 do
        S^c' ← the induced graph of (V(S_i^2) + {v_i, v_j});
        S_e^c ← S_e^c ∪ S^c'
    end for
    Add S_e^c into S^c if S_e^c ≠ ∅
end for
return S^c
```

D. Redundant Ratio (RR) Metric

Although each detected 2-4-Star is global rigid, they may also produce unreliable local formation result when the edge lengths are noisy. The way to further improve the local realization reliability is to merge the 2-4-stars to find more condensed components. The redundant edges in the component can help to improve accuracy for providing over-constraints in the local optimization.

Definition 7 (Redundant Rigidity): A graph $\mathbf{G} = (\mathbf{V}, \mathbf{E})$ *with realization in* $\mathbb{R}^3$ *having* $n \geq 4$ *nodes is* redundant rigid *if and only if it remains rigid after the removal of any one edge* $(i, j) \in \mathbf{E}$.

Since each redundant rigid graph $\mathbf{G} = (\mathbf{V}, \mathbf{E})$ must be rigid, it contains a rigid spanning graph, denoted by $\mathcal{L}$. The edges contained in the rigid graph are independent, which is denoted by set $\widetilde{\mathcal{E}}$. An edge $e \in \mathbf{E} \setminus \widetilde{\mathcal{E}}$ must be dependent with some edges in $\widetilde{\mathcal{E}}$. Gortler [19] showed that $|\widetilde{\mathcal{E}}| = 3n-6$ in $\mathbb{R}^3$, i.e., the number of independent edges in a rigid graph $\mathbf{G}$ with n vertices is $3n-6$.

Definition 8 (Number of Redundant Edges): For a redundant rigid graph $\mathbf{G} = (\mathbf{V}, \mathbf{E})$*, the number of redundant edges is defined as the total number of edges minus the number of independent edges, i.e.,* $|\mathcal{F}| = |\mathbf{E}| - |\widetilde{\mathcal{E}}|$. *In 3D networks, it is* $|\mathcal{F}| = m - (3n-6)$.

We therefore propose redundant ratio (RR) to indicate the proportion of redundant edges in a component.

Definition 9 (Redundant Ratio (RR)): Considering a redundant rigid graph $\mathbf{G} = (\mathbf{V}, \mathbf{E})$ *in* $\mathbb{R}^3$ *with* $|\mathbf{V}| = n$. *The* redundant rate (RR) *of this graph is calculated as* $r(\mathbf{G}) = \frac{m-(3n-6)}{C(n,2)}$, *where* $C(n,2)$ *is the edges number of* n*-complete graph, i.e.,* $C(n,2) = \frac{n(n-1)}{2}$. *So* $r(\mathbf{G}) = \frac{2(m-(3n-6))}{n(n-1)}$

Note that the 2-4-star graphs are global rigid, which must be redundant rigid. Let $m_i = |E(\mathbf{S}_i)|$ be the number of edges and

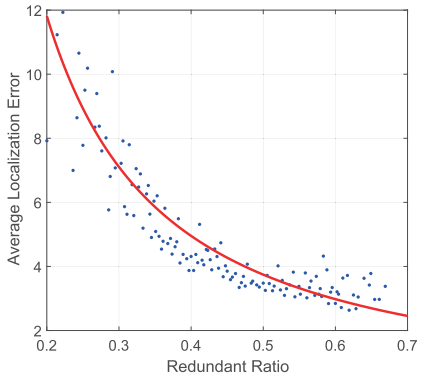


Fig. 3. Comparison of the results in different topologies.

$n_i = |V(\mathbf{S}_i)|$ be the number of vertices in a 2-4-star graph $\mathbf{S}_i$, then the number of redundant edges is $m_i - (3n_i - 6)$, and the redundant ratio is: $r(\mathbf{S}_i) = \frac{2(m_i-(3n_i-6))}{n_i(n_i-1)}$.

The validity of RR metric is checked by investigating its relationship with the formation calculation errors. Fig. 3 shows the average error of node localization in a component as a function of the RR of the component. 200 redundantly rigid components are generated and their location errors and RR are evaluated. The red line is the results of polynomial fitting. It can be seen clearly the average localization error decreases with the increasing of RR metric. Statistically, the correlation coefficients of the localization errors and the proposed RR metric in these experiments is -0.84, which indicates that the redundant ratio has a strong negative correlation with the average localization error.

E. Merging the 2-4-Stars for Better Reliability

Let the detected 2-4-stars be $\mathbf{S^c} = \{\mathbf{S_1^c}, \mathbf{S_2^c}, \ldots, \mathbf{S_K^c}\}$. We merge these 2-4-stars to form local mostly condensed components (LMCC) for achieving more reliable formation calculation in the LMCCs. Let $\mathbf{G}_i^c$ denote a component, which is a 2-4-star or a merged component from several 2-4-stars.

1) Sufficient Condition to Merge Two Components: Considering two components $\mathbf{G}_i^c$ and $\mathbf{G}_j^c$. The condition that they can be merged is:

$$r(\mathbf{G}_i^c \cup \mathbf{G}_j^c) > \max\ \left\{r(\mathbf{G}_i^c), r(\mathbf{G}_j^c)\right\} \tag{6}$$

which means that the union of these two components has larger RR metric.

2) Necessary Condition to Merge Two Components: Suppose $\mathbf{G}_i^c$ contains r_i redundant edges; $\mathbf{G}_j^c$ contains r_j redundant edges. When they are merged, $\mathbf{G}_i^c \cup \mathbf{G}_j^c$ contains $r_{i,j} \leq r_i + r_j$ redundant edges, because some redundant edges may be overlapped. Let η be the number of overlapped vertices in the merged components. So the condition in (6) can be expanded as: $\frac{r_{i,j}}{C(n_i+n_j-\eta,2)} > \frac{r_i}{C(n_i,2)}$ and $\frac{r_{i,j}}{C(n_i+n_j-\eta,2)} > \frac{r_j}{C(n_j,2)}$, where n_i and n_j are the number of vertices in $\mathbf{G}_i^c$ and $\mathbf{G}_j^c$ respectively. So

$$\left(C(n_i,2)+C(n_j,2)\right) r_{i,j} > C(n_i+n_j-\eta,2)\left(r_i+r_j\right) \tag{7}$$

Since $r_{i,j} \leq r_i + r_j$, if two components can be merged, there must be:

$$C(n_i,2) + C(n_j,2) > C(n_i+n_j-\eta,2) \tag{8}$$

Theorem 3: When $n_i \geq 4$, $n_j \geq 4$ and $\eta \leq \min\{n_i, n_j\}$, if the necessary condition in (8) is true, i.e., if two components can be merged to a component with higher RR, there must be $\eta \geq 4$.

Proof: The condition in (8) can be expanded as

$$f(n_i, n_j, \eta) = 2n_in_j - 2(n_i+n_j)\eta + \eta + \eta^2 < 0$$

In a component merged from 2-4-stars, $n_i \geq 4$, $n_j \geq 4$ and $\eta \leq \min\{n_i, n_j\}$. All n_i, n_j, and η are integers. Since $n_i \geq 4$ and $n_j \geq 4$, we can enumerate the value space of $f(n_i, n_j, \eta)$ using η as a parameter. The value of $f(n_i, n_j, \eta)$ is smaller than 0 for $n_i \geq 4, n_j \geq 4$, only when $\eta \geq 4$. □

Theorem 4: If $\mathbf{G}_1$ and $\mathbf{G}_2$ are two global rigid components in $\mathbb{R}^3$ and they share at least 4 non-co-surface vertices, then $\mathbf{G}_1 \cup \mathbf{G}_2$ is global rigid in $\mathbb{R}^3$.

Proof: Each global rigid component has a unique realizable formation. Using the overlapped four vertices as anchors, the locations of other vertices in $\mathbf{G}_1$ are uniquely determined, as well as for other vertices in $\mathbf{G}_2$. So the locations of all the nodes in $\mathbf{G}_1 \cup \mathbf{G}_2$ are uniquely localized, so that $\mathbf{G}_1 \cup \mathbf{G}_2$ is global rigid. □

Therefore, by merging two components (2-4-stars, or union of 2-4-stars) who share at least four non-co-surface vertices and satisfy (6), we can guarantee the merged component not only has better redundant ratio, but also is global rigid.

Therefore, a greedy algorithm is designed to merge any two components who have at least 4 non-co-surface vertices and satisfy (6). The algorithm is given in Algorithm 2. The input of the algorithm is the set of 2-4-stars. The output is a set of merged components. In each step, we merge two components in a greedy manner. In particular, the two components whose union has the highest RR will be selected. If $r(\mathbf{G}_i^c \cup \mathbf{G}_j^c) > r(\mathbf{G}_i^c)$ and $r(\mathbf{G}_i^c \cup \mathbf{G}_i^c) > r(\mathbf{G}_j^c)$, $\mathbf{G}_i^c$ and $\mathbf{G}_j^c$ will be merged into one component. The merging process repeats until any two components cannot be merged. The result is a set of LMCCs. Algorithm 2 has complexity of $O(K^3)$, where K is the number of the 2-4-stars.

F. Weighted Component and Patch Stitching

By Algorithm 2, we have found the global rigid LMCCs and calculated their RR metrics. The local coordinates of nodes in these LMCCs can be calculated rather reliably, which are denoted by $\{\mathbf{q}_i^{c,l}\}$ for node i's local coordinate in LMCC l. But there are some sparse parts in the graph which are not included in these LMCCs. To synchronize the nodes in the sparse parts, we calculate their local coordinates in patches, i.e., $\{\mathbf{q}_j^i\}$ for node j's local coordinate in patch i. The local coordinates calculated in patches and local coordinates calculated in LMCCs are synchronized together by

Algorithm 2 Component Merging Algorithm

Input: $\mathbf{S^c} = \{\mathbf{S_1^c}, \mathbf{S_2^c}, \ldots, \mathbf{S_K^c}\}$
Output:$\mathbf{G}^c = \{\mathbf{G}_1^c, \mathbf{G}_2^c, \ldots, \mathbf{G}_L^c\}$
$Lastnum = 0$, $num = |\mathbf{S^c}|$, $\mathbf{G}^c \leftarrow \mathbf{S^c}$;
while $Lastnum \neq num$ **do**
  $Lastnum = num$;
  $\{i, j\} := \max_{\mathbf{G}_i^c, \mathbf{G}_j^c \in \mathbf{G}^c} r(\mathbf{G}_i^c \cup \mathbf{G}_j^c)$;
  $T = r(\mathbf{G}_i^c \cup \mathbf{G}_j^c)$
  if $T > r(\mathbf{G}_i^c)$ and $T > r(\mathbf{G}_j^c)$ **then**
    $\mathbf{G}_i^c \leftarrow \mathbf{G}_i^c \cup \mathbf{G}_j^c$;
    $\mathbf{G}^c \leftarrow \mathbf{G}^c - \{\mathbf{G}_j^c\}$;
    $num = num - 1$;
  end if
end while
return $\mathbf{G}^c$

merging (4) and (5). The revised cost function is given in (9).

$$F = \min_{\mathbf{P}, \mathbf{R}_1, \ldots, \mathbf{R}_1^c, \ldots} \{\sum_{i=1}^{n} \sum_{j \in \mathbf{N}_i} \|(\mathbf{p}_i - \mathbf{p}_j) - \mathbf{R}_i(\mathbf{q}_i^i - \mathbf{q}_j^i)\|^2 + \sum_{l=1}^{L} \sum_{(i,j) \in \mathbf{G}_l^c} f(r_l) \|(\mathbf{p}_i - \mathbf{p}_j) - \mathbf{R}_l^c(\mathbf{q}_i^{c,l} - \mathbf{q}_j^{c,l})\|^2 : \mathbf{R}_i^{\mathrm{T}} \mathbf{R}_i = I, \quad (\mathbf{R}_l^c)^T \mathbf{R}_l^c = I\} \tag{9}$$

In (9), the local coordinates calculated in the LMCCs are weighted by redundant ratio $f(r_l) > 1$.

1) Alternating Least-Squares (ALS) Method: The weighted synchronization in (9) is solved using an Alternating Least-Squares (ALS) method. In the local phase, $\mathbf{P}$ is assumed fixed to solve (9) for each $\mathbf{R}_i$ and $\mathbf{R}_k^c$, which can be solved by Procrustes analysis [40]. In the global phase, all the $\mathbf{R}_i$ and $\mathbf{R}_k^c$ are assumed fixed to solve $\mathbf{P}$ by Eqn. (9). This is obtained by setting the gradient $\frac{\partial F}{\partial \mathbf{p}_i} = 0, \forall i = 1, 2, \cdots, n$, which sets up n linear equations of $\mathbf{p}_i$ and $\mathbf{p}_j$, where $\mathbf{p}_j \in \mathbf{N}_i$.

$$\sum_{j \in \mathbf{N}_i} (\mathbf{p}_i - \mathbf{p}_j) = \frac{1}{2 + \sum_{\mathbf{G}_l^c : (i,j) \in \mathbf{G}_l^c} f(r_l)} \cdot \Big(\sum_{j \in \mathbf{N}_i} [\mathbf{R}_i(\mathbf{q}_i^i - \mathbf{q}_j^i) + \mathbf{R}_j(\mathbf{q}_i^j - \mathbf{q}_j^j) + \sum_{\mathbf{G}_l^c : (i,j) \in \mathbf{G}_l^c} f(r_l) \mathbf{R}_l^c(\mathbf{q}_i^{c,l} - \mathbf{q}_j^{c,l})] \Big) \tag{10}$$

Note that $\sum_{\mathbf{G}_l^c : (i,j) \in \mathbf{G}_l^c} f(r_l)$ is the summation in all LMCCs that contain edge (i, j). By solving these n linear equations, the global coordinates are obtained, which are then substituted into (9) for solving $\{\mathbf{R}_i\}$ and $\{\mathbf{R}_l^c\}$. The iteration repeats until convergence of $\mathbf{P}$ or reaching a maximum number of iterations.

2) Selection of the Weight Function $f(r)$: Four types of weight functions are investigated, namely, *linear function, logarithmic function, cubic function and exponential function*,

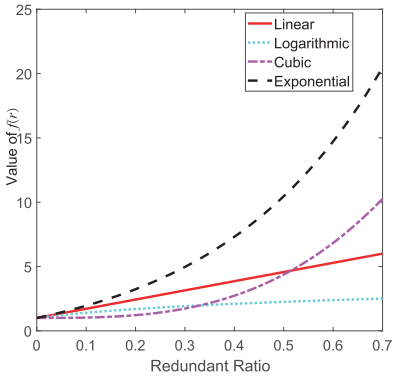


Fig. 4. Four forms of weight functions.

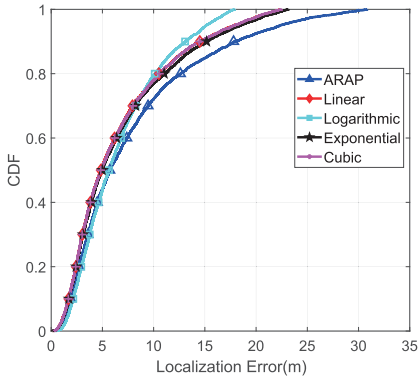


Fig. 5. The CDF of localization results by different weight functions.

which are designed as below:

$$\begin{aligned} f_{linear}(r) &= 1 + \alpha * r \\ f_{logarithmic}(r) &= 1 + ln(1 + \alpha * r) \\ f_{cubic}(r) &= (1 + \alpha * r)^3 \\ f_{exponential}(r) &= 1 - e + e^{1 + \alpha * r} \end{aligned} \tag{11}$$

The parameter α in Equation (11) is a scaling factor, which can be changed to control the scale of the weight. To make the starting point of the four functions be the same and higher than 1, we have added a constant to each function. The designed four weighting functions are plotted in Fig. 4. The impacts of the weighting functions on the localization results are investigated by simulations. In simulations, the localization results weighted by these four functions are compared and are also compared with the result of ARAP. The cumulative density function of localization errors are plotted in Fig. 5. All of these four weighting methods perform better than ARAP, and the logarithmic weighting function performs best for having small proportion of large errors. Therefore the logarithmic weighting is used in the simulation sections.

V. Integrating WCS and Kalman Filter(WCKF)

WCS provides reliable formation calculation $\{\mathbf{p_1}, \cdots, \mathbf{p_n}\}$ based on the distance measurements $\mathbf{D}(t)$ at one time instance. When the UAV swarm is flying continuously, we need to design tracking algorithms to track the formation evolving

over time, based on the distance matrix measured from 1 to t. Kalman filter-based tracking algorithms have been widely applied in vehicles, robot, and aircraft tracking applications [36]. But as shown in Fig.1, the traditional EKF-based algorithm provides poor performance in 3D sparse networks. The main reason is that the rigid and non-rigid observations are equally treated.

To utilize the advantages of the WCS to enhance the formation tracking performance in sparse networks, we propose to integrate WCS and Kalman Filter, i.e., WCKF. WCKF has two steps in each iteration, i.e., *state prediction* and *state correction*. The positions and velocities of nodes are denoted as state variables: $\mathbf{X_t} = \{\mathbf{p_1(t)}, \cdots, \mathbf{p_n(t)}, \mathbf{v_1(t)}, \cdots, \mathbf{v_n(t)}\}^{\mathbf{T}}$, where $\mathbf{p}_i(t) = (x_i(t), y_i(t), z_i(t))^T$ is the position of node i; $\mathbf{v}_i(t) = (\dot{x}_i(t), \dot{y}_i(t), \dot{z}_i(t))^T$ is the velocity vector.

A. State Prediction

Let $\overline{\mu}_t$ be the predicted mean and $\overline{\mathbf{P}}_t$ be the predicted variance matrix of $\mathbf{X_t}$ respectively. The state prediction is based on the motion model of UAV:

$$\overline{\mu}_t = \mathbf{A}\mu_{t-1} + \mathbf{B}\mathbf{u}_t \tag{12}$$

$$\overline{\mathbf{P}}_t = \mathbf{A}^T\mathbf{P}_{t-1}\mathbf{A}^T + \mathbf{R} \tag{13}$$

where $\mathbf{A} = diag\,(\mathbf{A}_1, \mathbf{A}_2 \cdots \mathbf{A}_n)$ is the system matrix; $\mathbf{B} = [\mathbf{B_1}, \mathbf{B_2} \cdots \mathbf{B_n}]^T$ is the control gain, where $\mathbf{A}_i$ and $\mathbf{B}_i$ are defined in Definition 1. $\mathbf{R}$ is the covariance matrix of the noises of the motion model.

B. State Correction

State correction is the key step. By integrating WCS, the local coordinates obtained at each time instance t are used as observations. The observations include the local coordinates and the rotation matrices of the patches and the components. Their relations to the state variable can be set up as:

$$\begin{cases} \mathbf{p}_i - \mathbf{p}_j = \mathbf{R}_i(\mathbf{q}_i^i - \mathbf{q}_j^i), & \forall i \in 1:n,\ j \in \mathbf{N}_i \\ \mathbf{p}_i - \mathbf{p}_j = \mathbf{R}_l^c(\mathbf{q}_i^{c,l} - \mathbf{q}_j^{c,l}), & \forall l \in 1:L,\ (i,j) \in E(\mathbf{G_l^c}) \end{cases} \tag{14}$$

They totally provide $M_t = \sum_{i=1,\ldots,n} |\mathbf{N_i}| + \sum_{l=1,\ldots,k} |E(\mathbf{G_l^c})|$ observation functions. If these M_t observations are directly used to build observation matrix $\mathbf{H}_t$, the complexity of matrix inversion in calculating the Kalman gain in (18) will be $O(M_t^3)$, which is quit high. To reduce the size of the observation matrix, the observations on each common edge are aggregated into one observation. We can treat this as a sensor fusion problem. We assume the observations are independent normally distributed. Denote $\mathbf{O}_{i,j}^{p,i} = \mathbf{R}_i(\mathbf{q}_i^i - \mathbf{q}_j^i) \sim \mathcal{N}(0, \mathbf{\Sigma_p})$; $\mathbf{O}_{i,j}^{c,l} = \mathbf{R}_l^c(\mathbf{q}_i^{c,l} - \mathbf{q}_j^{c,l}) \sim \mathcal{N}(0, \mathbf{\Sigma_c})$. Both $\mathbf{\Sigma_p}$ and $\mathbf{\Sigma_c}$ are 3×3 diagonal matrix. Then, for an edge $(i,j) \in E$, let $\omega_{i,j}^{p,i} = 1$ if j is in patch $\mathbf{N}_i$ ($\omega_{i,j}^{p,i} = 0$ otherwise); Let $\omega_{i,j}^{c,l} = f(r_l)$ if $(i,j) \in E(\mathbf{G_l^c})$, ($\omega_{i,j}^{c,l} = 0$, otherwise). Then the aggregated observation for edge (i,j) are:

$$\mathbf{O}_{i,j} = \mathbf{p}_i - \mathbf{p}_j = \frac{1}{\varpi_{ij}} \left(\sum_{i=1}^{n} \varpi_{i,j}^{p,i} \mathbf{O}_{i,j}^{p,i} + \sum_{l=1}^{L} \varpi_{i,j}^{c,l} \mathbf{O}_{i,j}^{c,l} \right) \tag{15}$$

where $\varpi_{ij} = \sum_{i=1}^{n} \varpi_{i,j}^{p,i} + \sum_{l=1}^{L} \varpi_{i,j}^{c,l}$ is the sum of weights. The variance of the aggregated obervation on edge (i,j) is

$$\mathbf{\Sigma}_{i,j} = \frac{1}{(\varpi_{ij})^2} \left(\sum_{i=1}^{n} \left(\varpi_{i,j}^{p,i}\right)^2 \mathbf{\Sigma}_p + \sum_{l=1}^{L} \left(\varpi_{i,j}^{c,l}\right)^2 \mathbf{\Sigma}_c \right) \tag{16}$$

So the observation vector and observation matrix can be constructed as following.

$$\mathbf{z}(t) = \mathbf{H_t}\mathbf{X_t} + \mathbf{v} \tag{17}$$

where $\mathbf{z}(t) = [\cdots, \mathbf{p_i} - \mathbf{p_j}, \cdots]^T$ are observation vector formed by $(i,j) \in E$.

$$\mathbf{H_t} = \begin{pmatrix} \cdots & \mathbf{I}_3 & \cdots & -\mathbf{I}_3 & \cdots & 0 & \cdots & 0 \\ & \vdots & & & & & \vdots & \\ & \mathbf{I}_3 & & -\mathbf{I}_3 & \vdots & 0 & \cdots & 0 \\ \cdots & \vdots & & & & & \vdots & \end{pmatrix}$$

$\mathbf{H_t}$ is a $3m \times 6n$ sparse matrix, which contains $-\mathbf{I}_3$ at $3(i-1)+1 : 3(i-1)+3$ columns and $-\mathbf{I}_3$ at $3(j-1)+1 : 3(j-1)+3$ columns if $\mathbf{p}_i - \mathbf{p}_j$ is observed. The covariance matrix $\mathbf{Q}_t$ is a $3m \times 3m$ diagonal matrix. $\mathbf{Q}_t = diag(\mathbf{\Sigma}_{i,j}), \forall (i,j) \in E$, where $\mathbf{\Sigma_{i,j}}$ is in (16). There are different ways to assign $\mathbf{R}$. In our approaches, we assign $\sigma_{\mathbf{p_i x}}^{\mathbf{2}} = \sigma_{\mathbf{p_i y}}^{\mathbf{2}} = \sigma_{\mathbf{p_i z}}^{\mathbf{2}} = \sigma_{\mathbf{v_i x}}^{\mathbf{2}} = \sigma_{\mathbf{v_i y}}^{\mathbf{2}} = \sigma_{\mathbf{v_i z}}^{\mathbf{2}} = \sigma^{\mathbf{2}}$.

Then the states are updated by KF iteration:

$$\mathbf{K}_t = \overline{\mathbf{P}}_t\mathbf{H}_t^T(\mathbf{H}_t\overline{\mathbf{P}}_t\mathbf{H}_t^T + \mathbf{Q}_t)^{-1} \tag{18}$$

$$\mu_t = \overline{\mu}_t + \mathbf{K}_t\left(\mathbf{z}(t) - \mathbf{H}_t\overline{\mu}_t\right) \tag{19}$$

$$\mathbf{P}_t = (\mathbf{I} - \mathbf{K}_t\mathbf{H}_t)\overline{\mathbf{P}}_t \tag{20}$$

Eqn. (18) calculates the Kalman gain; (19) and (20) update the state mean and variance matrix respectively. The prediction and correction steps update μ_t and $\mathbf{P}_t$ in each time.

VI. Performance Analysis and Evaluations

A. Complexity Analysis

Given a rigid graph $G = (V,E)$,$|V| = n$ and $|E| = m$, the graph generally has a low node degree for the limited communication range of the UAVs and the safe distance for collision avoidance, i.e., the maximum node degree D is generally upper bounded. We also denote the maximum iteration times in ARAP and WCS by h.

Theorem 5: Given $|V| = n$, $|E| = m$, the maximum iteration times h, the worst case time complexity of WCS is $O(hn^3)$.

Proof: In WCS, a set of 2-4-stars are firstly detected by Algorithm 1. Its time complexity is $O(mn)$ for searching 2-connected subgraphs induced by each edge. Suppose K 2-4-stars are got in this step. In sparse UAV networks, K is generally smaller than n. Then we merge the 2-4-stars into LMCCs by Algorithm 2, whose time complexity is $O(K^3) < O(n^3)$. Finally, we synchronize the local coordinates calculated in LMCCs and in patches by (10). The ALS step needs a traversal of all edges in all patches and LMCCs. The time complexity of this process is $O((n+L)m)$, which is smaller than $O(n^3)$. Because the maximum iteration times is h, the time

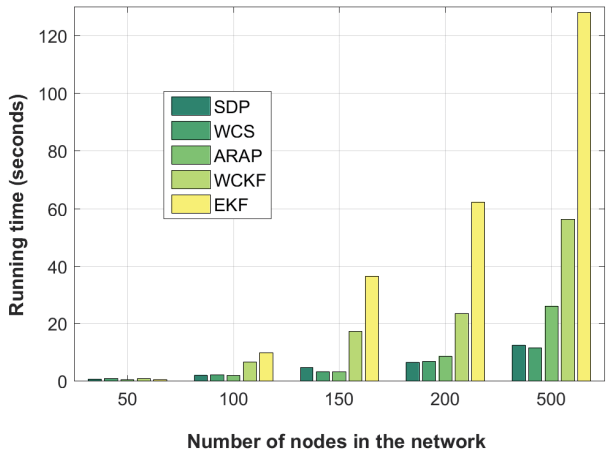


Fig. 6. Comparing of solving time of different algorithms for networks with different node number.

TABLE II
THE TIME COMPLEXITY OF SELECTED ALGORITHM

Algorithm	WCS	ARAP	SDP	WCKF	EKF
Complexity	$O(hn^3)$	$O(hn^3)$	$O(Dn^3)$	$O(D^3n^3)$	$O(D^3n^3)$

complexity of the synchronization step is $O(h \cdot n^3)$. So the complexity of WCS is totally $O(n^3)+O(n^3)+O(h{\cdot}n^3)$, which is $O(h \cdot n^3)$. □

Theorem 6: Given $|V| = n$, $|E| = m$, the maximum node degree D, the worst case time complexity of WCKF in a time instance is $O(D^3n^3)$.

Proof: In WCKF, the process of finding the LMCCs is the same as WCS, which is $O(mn + K^3)$. Then we use the calculated local coordinates to update the Kalman Filter model. The most time-consuming process in WCKF is to calculate the Kalman gain in Eqn.(18). It involves calculating the inversion of a $3m \times 3m$ matrix. The time complexity of this step is $O(m^3)$, which equals to $O(D^3n^3)$, since $m \leq \frac{D}{2}n$. So the time complexity of WCKF in a time instance is $O(D^3n^3)$. □

B. Complexity Comparison

We compare the computation complexity of different algorithms and also verify the computation times by simulations. The time complexity of the algorithms is summarized in Table. II. WCS and ARAP have the same complexity. WCKF's and EKF have the same complexity. We then compare the running times of these algorithms in networks with different number of nodes. The average running time of each algorithm is evaluated in 50 random networks for each setting of node number. The result is shown in Fig. 6.

When the node number equals 50, all algorithms finish in 1 second. When the nodes number increase, WCS and SDP are the mostly efficient. In Table. II, the time complexity of WCS and ARAP is the same, but WCS performs better in practice, because it converges quickly than ARAP, i.e., using less number of iterations. The reason is that the weighting of the module speeds up the convergence. WCKF and EKF are generally more time consuming. We use the LU decomposition to complete the matrix inversion in both EKF and WCKF. The matrix $\mathbf{H_t}$ in WCKF is sparser than EKF's, and entries of $\mathbf{H_t}$ in WCKF are integers, which leads to the practical gap between these two algorithms.

VII. PERFORMANCE EVALUATION

The proposed WCS and WCKF are extensively verified by simulations and compared with existing methods.

A. Simulation Settings

In particular, we compare the performances of WCS, WCKF with that of:

1) the state-of-the-art of component stitching method ARAP [21];
2) the centralized network locating method SDP [33];
3) traditional EKF-based tracking algorithm;

We've documented the localization error of each node as $\|\mathbf{p_i} - \mathbf{p_i'}\|$, where $\mathbf{p}_i'$ is the estimated localization of $\mathbf{v}_i$; $\mathbf{p}_i$ is the ground-truth location. The algorithms are implemented in UAV networks under various network and parameter settings.

B. Simulations in Static Environment

Firstly, we compare these methods when nodes are static. That is, we only calculate the localization results when nodes are static without formation tracking. The performances are compared under different network settings.

1) Simulation Settings: The considered network settings include 1) edge density; 2) ranging noises. In simulations, 50 nodes are randomly deployed into a space of $300(m) \times 100(m) \times 100(m)$ area. The edge density of the graph is controlled by the ranging radius R. $d_{i,j}$ between nodes v_i and v_j can be measured if $\|\mathbf{p}_i - \mathbf{p}_j\| < R$. Therefore, by varying R between $[40, 75]$, we can control the density of edges. The distance measurements are impacted by noises $d_{i,j} = d_{j,i} = \|\mathbf{p}_i - \mathbf{p}_j\| + \varepsilon_{ij}$, $\varepsilon_{ij} \sim N(0, \sigma^2)$, where the noise level is controlled by varying σ. The simulations are carried out in Matlab2015b.

2) Simulation Results: We present the experimental results in the form of cumulative distribution and average nodes' localization errors, as shown in the Fig.7. For each parameter setting, we run experiments in 200 random topologies to calculate the localization results. We controlled the ranging noises in three levels, respectively, $\sigma = 1$, $\sigma = 2$ and $\sigma = 5$. We select 8 ranging radius between $[40, 75]$ and calculate the average error for each level. As shown in Fig.7c)f)i), rather accurate formation results can be obtained for all the four methods when $R \geq 60$, i.e., when the network is well-connected. Therefore, we focus on the sparse cases. In particular, we plot the cumulative error distribution when $R = 40$ and $R = 50$, which are two sparse settings.

The first column and the second column subgraphs of Fig.7 are CDF of formation errors when $R = 40$ and $R = 50$ respectively. EKF-based algorithm provided highly unsatisfactory results in these two cases. SDP provides rather accurate formation result for the case when $R = 50$, but provides unsatisfied result when $R = 40$, which shows its sensitivity to the network sparsity. ARAP and the proposed WCS method

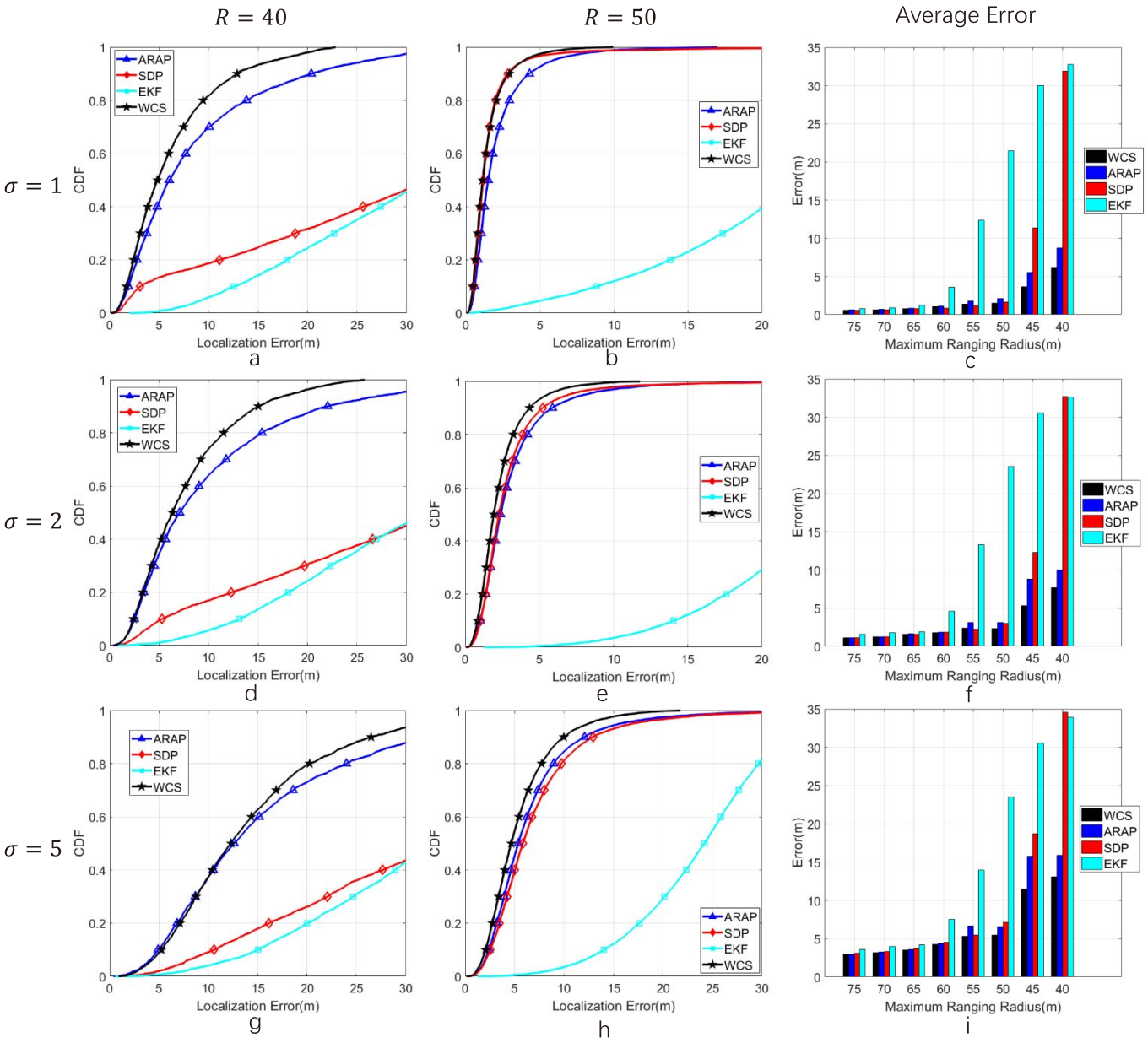


Fig. 7. Comparing of location error of different algorithms for networks with different connectivity and noise level settings.

both show better tolerance to network sparsity. But it can be seen that WCS provides better localization accuracy than ARAP in these sparse networks. When $R = 40$ which is the more sparse case, the difference between WCS and ARAP is more clear. The formation errors all increase as the ranging noises increase.

Each row of Fig.7 is corresponding to a noise level. In all these three noise levels, WCS has the best performances. As shown in Fig.7(b)(e)(h), our method shows more advantages over SDP and ARAP when σ increases from 1 to 5. Fig.7(c)(f)(i) plot the average formation errors of different algorithms in these settings. It shows component stitching based methods are much better in sparse networks, and WCS is more robust than ARAP algorithm.

C. Experiments for Formation Tracking

1) Simulation Setting: We conduct simulations in Unity3D 5.0, which is a well-known 3D game engine to generate dynamic UAV formations. The 3D environments built in Unity3D are shown in Fig9. In simulating the formation dynamics, at the beginning, 30 UAVs are deployed randomly in a small space, as shown in Fig9(a). Then each UAV begins to move randomly in each time slot, with the speed between $0m/s$ and $20m/s$. For the reason of entropy increasing, the UAV network will become increasingly sparse due to the randomness of motion. We control all UAVs to prevent collision and prevent them leaving the prescribed area, which is a $100m \times 100m \times 100m$ space. Fig9(b) was captured at

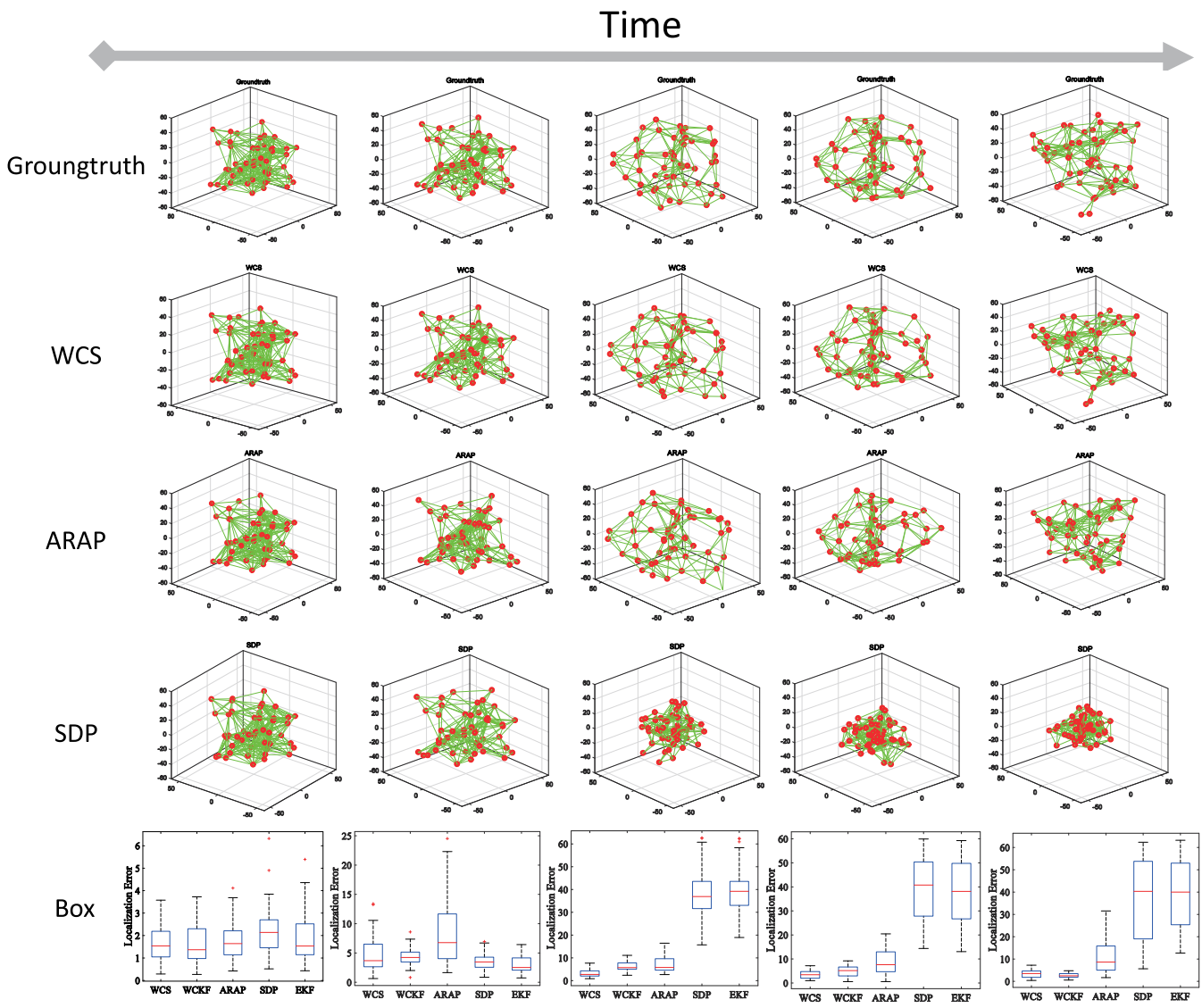


Fig. 8. Comparing of tracking result of different algorithms.

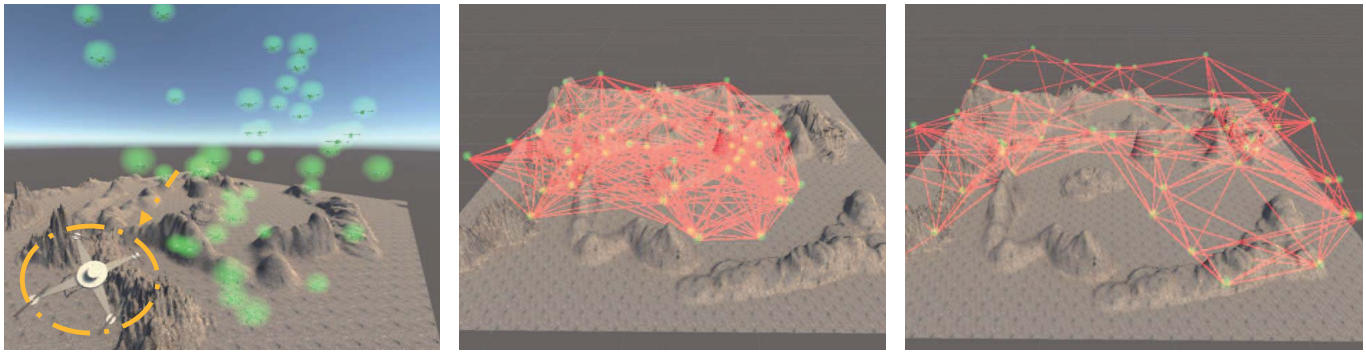
$t = 100$ during the formation expansion, and Fig9(c) was captured at $t = 200$, in which the network has been much sparser.

2) Tracking Results: We show formation tracking results in Fig 8. The corresponding parameter settings are $R = 40, \sigma = 5$ and the UAV number is 50. We evenly select five time slots during the experiments, and each column represents the result of one time slot. The first row is the ground truth of the real-time network structure, which is offered by Unity3D. The second row, third row, and the $4th$ row show the tracking results solved by WCS, ARAP, and SDP algorithms at the corresponding time instances. The $5th$ row shows the statistical results in the form of boxplot.

At the beginning of the tracking, the network is dense and all algorithms perform well. With the dynamic motions of the UAVs, the errors of the tracking results increase as the network becomes more and more sparse. But we can see that WCKF and WCS keep rather reasonable tracking accuracy during all these time slots.

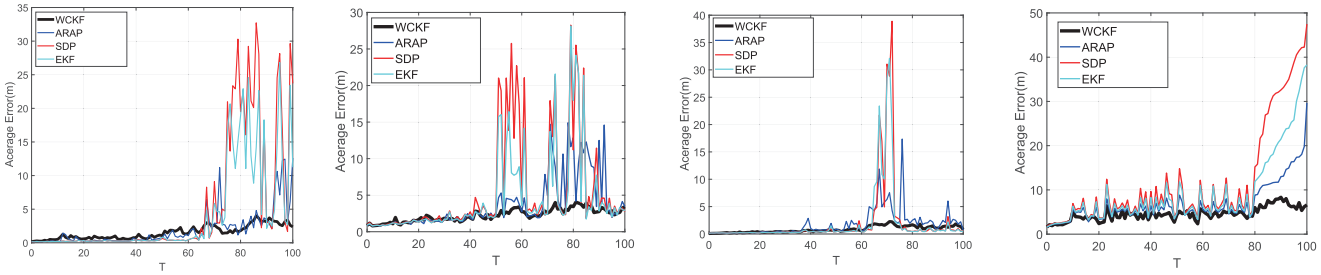
3) Compared With Others Tracking Algorithms: We conduct further experiments to compare the tracking performances with WCKF. The compared algorithms include ARAP, SDP and EKF. In particular we evaluate the cases when $(R = 45, \sigma = 1)$, $(R = 45, \sigma = 5)$, $(R = 55, \sigma = 1)$, $(R = 55, \sigma = 5)$ respectively. In each setting, we run experiments for 100 time slots. The beginning formation is generated randomly according to the parameter setting and the UAVs follow random motion in the following time slots to make the network sparser and sparser. In each slot, the average localization errors of all nodes are calculated and plotted.

The results are shown in Fig.10(a) to Fig.10(d). It can be seen that WCKF algorithm provides the best performances in all these settings than the other three algorithms. At the beginning stage of all these four experiments, all algorithms offer accurate results because of the networks are well-connected. With the motions of UAVs, the network becomes increasingly sparser and WCKF begins to show better accuracy than others. After the network becomes highly sparse, the location errors



a b c

Fig. 9. The simulation environments by Unity3D for formation tracking.



(a) R=45, σ=1 (b) R=45, σ=5 (c) R=55, σ=1 (d) R=55, σ=5

Fig. 10. Comparing of location error during the tracking of different algorithms for networks with different connectivity and noise level settings.

TABLE III
AVERAGE LOCALIZATION ERRORS IN TRACKING EXPERIMENTS

Parameter Setting	%	WCKF	ARAP	EKF	SDP
$R = 55, \sigma = 1$	**21.6%**	**0.98**	1.32	**1.25**	1.35
$R = 55, \sigma = 5$	**32.5%**	**1.79**	**2.87**	3.14	3.67
$R = 45, \sigma = 1$	**27.9%**	**1.58**	**2.19**	3.91	4.68
$R = 45, \sigma = 5$	**35.9%**	**3.28**	6.36	**5.12**	7.36

TABLE IV
AVERAGE LOCALIZATION ERRORS DURING 50-100 SECONDS IN TRACKING EXPERIMENTS

Parameter Setting	%	WCKF	ARAP	EKF	SDP
$R = 55, \sigma = 1$	**28.2%**	**1.35**	**1.88**	1.92	1.91
$R = 55, \sigma = 5$	**41%**	**2.32**	**3.93**	4.28	5.62
$R = 45, \sigma = 1$	**48.2%**	**2.14**	**4.13**	5.91	7.33
$R = 45, \sigma = 5$	**46.5%**	**4.27**	**7.98**	12.55	14.71

of the other three algorithms begin to increase. But benefited by the smoothing effects of the Kalman Filter and reliable component stitching, WCKF has much smaller error and variance than the others.

In Table III, we further summarize the average location errors of different algorithms under different parameter settings in the tracking experiments. Each value in the table is the average error of the whole 100 time slots in each experiment and averaged over 30 experiments. WCKF provides improvements than the state-of-the-art algorithms from 21.6% to 35.9% for different parameter settings. Table IV summarizes the results when the network becomes sparser. Each value is the average error during 50 to 100 time slots and averaged over 30 experiments. WCKF provides improvements than the state-of-the-art algorithms from 28% to 48% in sparse cases.

VIII. CONCLUSION

The paper has investigated the 3D formation tracking problem for dynamic and sparse graphs formed by UAV swarm. It investigated the unevenness structure of the graphs and presented novel weighted component stitching (WCS) and WCS based Kalman filter (WCKF) methods to address the formation tracking challenges in sparse networks. It shows the key to deal with the sparseness is to find and utilize the well-connected components. A 2-4-star detection and merging method is developed, which finds the reliable global rigid subgraphs (LMCC) in the sparse graph. A novel redundant ratio metric is proposed for indicating the graph's reliability based on the connectivity properties. Extensive simulations have shown that the proposed WCS and WCKF methods outperform the state-of-the-art ARAP, SDP and EKF methods in sparse networks. As accurate network formation is important to predict link availabilities and qualities in the dynamic UAV networks, accurate formation tracking may play an important role in UAV network communication.

The remaining problem is how to track the 3D formation when the network becomes even sparser so as to there are only a few redundant rigid components. It needs further investigation to guess edges in the sparse graph. Adding UAV attitude information will help the formation tracking in

the EKF prediction step. Time synchronization and system implementation in practical UAV swarm are also important future direction.

REFERENCES

[1] S. Sand *et al.*, "Swarm exploration and navigation on mars," in *Proc. Int. Conf. Localization GNSS (ICL-GNSS)*, Jun. 2013, pp. 1–6.

[2] N. Mahdoui, E. Natalizio, and V. Fremont, "Multi-UAVs network communication study for distributed visual simultaneous localization and mapping," in *Proc. Int. Conf. Comput., Netw. Commun. (ICNC)*, Feb. 2016, pp. 1–5.

[3] M. A. Guney and M. Unel, "Formation control of a group of micro aerial vehicles (MAVs)," in *Proc. IEEE Int. Conf. Syst., Man, Cybern.*, Oct. 2013, pp. 929–934.

[4] L. Barnes, R. Garcia, M. Fields, and K. Valavanis, "Swarm formation control utilizing ground and aerial unmanned systems," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst.*, Sep. 2008, p. 4205.

[5] X. Dong, B. Yu, Z. Shi, and Y. Zhong, "Time-varying formation control for unmanned aerial vehicles: Theories and applications," *IEEE Trans. Control Syst. Technol.*, vol. 23, no. 1, pp. 340–348, Jan. 2015.

[6] G. Michieletto, A. Cenedese, and A. Franchi, "Bearing rigidity theory in SE(3)," in *Proc. IEEE 55th Conf. Decis. Control (CDC)*, Dec. 2016, pp. 5950–5955.

[7] Z. Wang, C. Yang, X. Dong, Q. Li, and Z. Ren, "Time-varying formation control for mobile robots: Algorithms and experiments," in *Proc. 43rd Annu. Conf. IEEE Ind. Electron. Soc. (IECON)*, Oct./Nov. 2017, pp. 7239–7244.

[8] L. Gupta, R. Jain, and G. Vaszkun, "Survey of important issues in UAV communication networks," *IEEE Commun. Surveys Tuts.*, vol. 18, no. 2, pp. 1123–1152, 2nd Quart., 2016,

[9] J. Seo, Y. Kim, S. Kim, and A. Tsourdos, "Collision avoidance strategies for unmanned aerial vehicles in formation flight," *IEEE Trans. Aerosp. Electron. Syst.*, vol. 53, no. 6, pp. 2718–2734, Dec. 2017.

[10] J. L. Ny, A. Ribeiro, and G. J. Pappas, "Adaptive communication-constrained deployment of unmanned vehicle systems," *IEEE J. Sel. Areas Commun.*, vol. 30, no. 5, pp. 923–934, Jun. 2012.

[11] Q. Wu, Y. Zeng, and R. Zhang, "Joint trajectory and communication design for multi-UAV enabled wireless networks," *IEEE Trans. Wireless Commun.*, vol. 17, no. 3, pp. 2109–2121, Mar. 2018.

[12] J. Kwon and S. Hailes, "Scheduling UAVs to bridge communications in delay-tolerant networks using real-time scheduling analysis techniques," in *Proc. IEEE/SICE Int. Symp. Syst. Integr. (SII)*, Dec. 2014, pp. 363–369.

[13] Y. Zeng, X. Xu, and R. Zhang, "Trajectory design for completion time minimization in UAV-enabled multicasting," *IEEE Trans. Wireless Commun.*, vol. 17, no. 4, pp. 2233–2246, Apr. 2018.

[14] M. Logan, T. Vranas, M. Motter, Q. Shams, and D. Pollock, "Technology challenges in small UAV development," in *Proc. Infotech@Aerospace*, 2005, pp. 1–5.

[15] J. Wan, L. Zhong, and F. Zhang, "Cooperative localization of multi-UAVs via dynamic nonparametric belief propagation under GPS signal loss condition," *Int. J. Distrib. Sensor Netw.*, vol. 10, no. 2, p. 562380, 2014.

[16] A. Benini, A. Mancini, and S. Longhi, "An IMU/UWB/vision-based extended Kalman filter for mini-UAV localization in indoor environment using 802.15.4a wireless sensor network," *J. Intell. Robot. Syst.*, vol. 70, pp. 461–476, May 2013.

[17] Y. Ma, N. Selby, and F. Adib, "Minding the billions: Ultra-wideband localization for deployed RFID tags," in *Proc. 23rd Annu. Int. Conf. Mobile Comput. Netw.*, New York, NY, USA, 2017, pp. 248–260.

[18] K. Guo *et al.*, "Ultra-wideband-based localization for quadcopter navigation," *Unmanned Syst.*, vol. 4, no. 1, pp. 23–34, 2016.

[19] S. J. Gortler, A. Healy, and D. P. Thurston, "Characterizing generic global rigidity," *Amer. J. Math.*, vol. 132, no. 4, pp. 897–939, 2007.

[20] M. Cucuringu, "Graph realization and low-rank matrix completion," Ph.D. dissertation, Dept. Appl. Math., Princeton Univ., Princeton, NJ, USA, 2012.

[21] L. Zhang, L. Liu, C. Gotsman, and S. J. Gortler, "An as-rigid-as-possible approach to sensor network localization," *ACM Trans. Sens. Netw.*, vol. 6, no. 4, p. 35, 2010.

[22] H. G. de Marina, F. J. Pereda, J. M. Giron-Sierra, and F. Espinosa, "UAV attitude estimation using unscented Kalman filter and TRIAD," *IEEE Trans. Ind. Electron.*, vol. 59, no. 11, pp. 4465–4474, Nov. 2012.

[23] B. Tian, L. Liu, H. Lu, Z. Zuo, Q. Zong, and Y. Zhang, "Multivariable finite time attitude control for quadrotor UAV: Theory and experimentation," *IEEE Trans. Ind. Electron.*, vol. 65, no. 3, pp. 2567–2577, Mar. 2018.

[24] X. Dong, *Formation Control of Swarm Systems*. Berlin, Germany: Springer, 2016, pp. 53–103.

[25] V. M. Monajjemi, J. Wawerla, R. Vaughan, and G. Mori, "HRI in the sky: Creating and commanding teams of UAVs with a vision-mediated gestural interface," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst.*, Nov. 2013, pp. 617–623.

[26] B. Zhu, L. Xie, and D. Han, "Recent developments in control and optimization of swarm systems: A brief survey," in *Proc. 12th IEEE Int. Conf. Control Automat. (ICCA)*, Jun. 2016, pp. 19–24.

[27] F. Schiano and P. R. Giordano, "Bearing rigidity maintenance for formations of quadrotor UAVs," in *Proc. IEEE Int. Conf. Robot. Automat. (ICRA)*, May/Jun. 2017, pp. 1467–1474.

[28] J. Tiemann and C. Wietfeld, "Scalable and precise multi-UAV indoor navigation using TDOA-based UWB localization," in *Proc. Int. Conf. Indoor Positioning Indoor Navigat. (IPIN)*, Sep. 2017, pp. 1–7.

[29] F. J. Perez-Grau, F. Caballero, L. Merino, and A. Viguria, "Multi-modal mapping and localization of unmanned aerial robots based on ultra-wideband and RGB-D sensing," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst. (IROS)*, Sep. 2017, pp. 3495–3502.

[30] Y. Wang, L. Song, and S. S. Iyengar, "An efficient technique for locating multiple narrow-band ultrasound targets in chorus mode," *IEEE J. Sel. Areas Commun.*, vol. 33, no. 11, pp. 2343–2356, Nov. 2015.

[31] X. Ji and H. Zha, "Sensor positioning in wireless ad-hoc sensor networks using multidimensional scaling," in *Proc. 23rd Annu. Joint Conf. IEEE Comput. Commun. Soc. INFOCOM*, vol. 4, Nov. 2004, pp. 2652–2661.

[32] S. Korkmaz and A.-J. van der Veen, "Robust localization in sensor networkswith iterative majorization techniques," in *Proc. IEEE ICASSP*, Apr. 2009, pp. 2049–2052.

[33] A. M.-C. So and Y. Ye, "Theory of semidefinite programming for Sensor Network Localization," *Math. Program.*, vol. 109, nos. 2–3, pp. 367–384, 2006.

[34] T. Sun, Y. Wang, D. Li, W. Chen, and Z. Gu, "Robust component-based network location with noisy range measurements," in *Proc. 27th Int. Conf. Comput. Commun. Netw.*, Hangzhou, China, 2018, pp. 1–9.

[35] M. Cucuringu, Y. Lipman, and A. Singer, "Sensor network localization by eigenvector synchronization over the Euclidean group," *ACM Trans. Sensor Netw.*, vol. 8, no. 3, p. 19, 2012.

[36] C. Luo, S. I. McClean, G. Parr, L. Teacy, and R. De Nardi, "UAV position estimation and collision avoidance using the extended Kalman filter," *IEEE Trans. Veh. Technol.*, vol. 62, no. 6, pp. 2749–2762, Jul. 2013.

[37] M. S. Arulampalam, S. Maskell, N. Gordon, and T. Clapp, "A tutorial on particle filters for online nonlinear/non-Gaussian Bayesian tracking," *IEEE Trans. Signal Process.*, vol. 50, no. 2, pp. 174–188, Feb. 2002.

[38] J. Pattillo, A. Veremyev, S. Butenko, and V. Boginski, "On the maximum quasi-clique problem," *Discrete Appl. Math.*, vol. 161, nos. 1–2, pp. 244–257, 01 2013.

[39] B. Jackson and A. Nixon, "Stress matrices and global rigidity of frameworks on surfaces," *Discrete Comput. Geometry*, vol. 54, no. 3, pp. 586–609, 2015.

[40] J. M. F. Ten Berge, "J.C. Gower and G.B. Dijksterhuis.*Procrustes problems*. New York: Oxford University Press," *Psychometrika*, vol. 70, no. 4, pp. 799–801, 2005.

Yongcai Wang received the B.S. and Ph.D. degrees from the Department of Automation Sciences and Engineering, Tsinghua University, in 2001 and 2006, respectively. He was an Associated Researcher with NEC Laboratories China from 2007 to 2009. He was a Research Scientist with the Institute for Interdisciplinary Information Sciences, Tsinghua University, from 2009 to 2015. He was a Visiting Scholar with the Department of Electronic and Computer Engineering, Cornell University, in 2015. He is currently an Associate Professor with the Department of Computer Sciences, Renmin University of China. His current research interests include network localization algorithms, simultaneously locating and mapping algorithms, and combinatorial optimization and applications.

Tianyuan Sun received the B.S. degree from the Department of Computer Sciences, Renmin University of China, in 2015, where he is currently pursuing the master's degree. His research interests include network localization algorithms and vision-based perception algorithms for robot and VR.

Guoyao Rao received the B.S. degree from the Department of Mathematics, Beijing Normal University, in 2016. He is currently pursuing the Ph.D. degree with the Department of Computer Sciences, Renmin University of China. His research interests include network localization algorithms and vision-based localization algorithms.

Deying Li received the M.S. degree in mathematics from Huazhong Normal University in 1988 and the Ph.D. degree in computer science from the City University of Hong Kong in 2004. She is currently a Professor with the Department of Computer Science, Renmin University of China. Her research interests include wireless networks, mobile computing, and algorithm design and analysis.