Understanding Node Localizability in Barycentric Linear Localization

Haodi Ping, *Graduate Student Member, IEEE*, Yongcai Wang, *Member, IEEE*, Deying Li, *Member, IEEE*, and Wenping Chen

***Abstract*— The barycentric linear localization (BLL) methods provide a lightweight, distributed way to calculate locations for resource-limited IoT devices. A crucial requirement for BLL is that the nodes participating in the iterative location propagation are localizable. Otherwise, the unlocalizable nodes will continuously pose error information in the location propagation process, making even the theoretically localizable nodes converge to the wrong locations. However, the research on node localizability in BLL is much lacked, greatly limiting the application scope of BLL. In specific, BLL node localizability is detected on a *generated graph* $\mathcal{G}^{\mathcal{A}}$. For any node, its neighbors appear in $\mathcal{G}^{\mathcal{A}}$ only when the neighbors can form triangle(s), so that $\mathcal{G}^{\mathcal{A}}$ is much sparser than the original $\mathcal{G}$. Thus, the node localizability condition in BLL is harder to be satisfied than that in traditional localization methods. Moreover, the distributed algorithm to detect BLL localizable nodes is still open. This paper thoroughly investigates the node localizability conditions and distributed localizable node detection algorithms in BLL. At first, an efficient and fully distributed Negative Edge Inference (NEI) algorithm is proposed for each node to infer implicit edges in its neighborhood. NEI strengthens the distance graph by revealing more distance constraints so that enables more neighboring triangles. Then a new sufficient condition, i.e., the recursive three disjoint path condition (Recursive-3DP) on the strengthened distance graph is proposed to identify BLL localizable nodes much more accurately. Secondly, a distributed Path Extension and Pruning (PEP) algorithm is proposed for distributed localizable node detection. PEP is proved to detect all the theoretically Recursive-3DP nodes in the strengthened distance graph. A Fast-PEP algorithm is further proposed, which misses very limited Recursive-3DP nodes while bringing significant improvement in efficiency. PEP and Fast-PEP guarantee to identify BLL localizable nodes in $2H$ rounds, where H is the maximum hop number of the node disjoint paths. Finally, by using NEI and PEP (Fast-PEP), a localizability-aware BLL (LABEL) method is proposed, which correctly identifies localizable nodes and guarantees their correct location convergence. Extensive analysis and experiments show the advantages in localizability and location accuracy of the proposed schemes over the state-of-the-art methods.**

***Index Terms*— Barycentric coordinate, distributed localization, node localizability condition, distributed localizability detection.**

Manuscript received 30 December 2021; revised 4 August 2022; accepted 12 October 2022; approved by IEEE/ACM TRANSACTIONS ON NETWORKING Editor T. He. This work was supported in part by the National Natural Science Foundation of China under Grant 61972404, Grant 12071478, and Grant 61732006; and in part by the Public Computing Cloud, Renmin University of China. *(Corresponding author: Yongcai Wang.)*

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

This article has supplementary downloadable material available at https://doi.org/10.1109/TNET.2022.3216204, provided by the authors.

Digital Object Identifier 10.1109/TNET.2022.3216204

I. Introduction

THE recent advances in wireless ranging techniques such as received signal strength [1], [2], WiFi RTT [3], [4], UWB [5], [6], and LoRa RTT [7] are promoting the development of *ranging-based localization* techniques [8]. For example, during the COVID-19 pandemic, the UWB ranging technique has been adopted for social distancing and contact tracing [9]. The UWB modules can also be found in the latest smartphones, smartwatches, and IoT devices [10], implying the broad application prospects of the ranging technique for location-based services.

In ranging-based localization, a certain number of objects with known locations are called *anchors*. The objects whose locations are unknown are called *agents*. Localization is to calculate the agent locations based on inter-agent and agent-anchor distance measurements, and anchor locations. Regarding where the locations are inferred, ranging-based localization methods are roughly categorized into two categories: *centralized* and *distributed*.

The centralized ranging-based localization algorithms can be further divided into: 1) *optimization-based*, including multidimensional scaling (MDS) [11], [12], general graph optimization (G2O) [13], [14], semidefinite programming (SDP) [15]; 2) *probabilistic-based*, [16], [17], [18], [19], [20]; 3) *geometric-based* [21], [22], [23]; and 4) *component stitching* [24], [25], [26], [27], [28], etc. A common limitation of centralized localization methods is the cost and latency for collecting ranging data to a central processor. The optimization-based algorithms further require proper initialization to avoid being trapped into local optima.

Distributed localization methods involve only local communication and local computation at each agent, which can be broadly divided into *sequential algorithms* and *concurrent algorithms* regarding the sequence of location updating. In sequential algorithms [29], [30], [31], [32], [33], [34], [35], [36], each agent determines its location w.r.t. location-known neighbors starting from the anchors. They typically require three anchors to have common neighbors for starting the trilateration. Concurrent algorithms [37], [38], [39], [40], [41] update the location estimations concurrently based on distance measurements and latest estimated locations of neighbors.

In the distributed and concurrent localization branch, recently, a *Barycentric coordinate-based Linear Localization* (BLL) schema has drawn great attention [42], [43], [44], [45], [46], [47], [48], [49], [50]. Through an efficient reparametrization with barycentric coordinates, the localization problem is reduced to distributed linear location updating at each node by a linear function of neighbors' location estimations.

The overall localization updating of all nodes becomes a large linear system. So the concurrent location evolvement inherits the property of the iterative linear system, including convergence and solution uniqueness, when the system matrix satisfies the convergence conditions [42], [43], [46]. The BLL methods are attractive for having the following properties. 1) It is fully distributed and concurrent; 2) It does not require good initialization of the agent locations; 3) It works with the minimal number of anchors (not less than $d+1$ in $\Re^d$) and does not require special anchor distribution (such as anchors forming triangles); 4) It is highly efficient for each agent to update only a local linear equation. Moreover, the convergence of BLL has been certified against noisy distance measurements, communication link failures, and communication channel noise [42], [44].

Despite the above benefits, we observe that a key drawback of BLL is that it works properly only when all the nodes participate in the location iteration are *localizable* (or equivalently, the linear system is nonsingular) [46]. We show that if the unlocalizable nodes are included in the iterative updating process, even the locations of the localizable nodes may converge to wrong locations. In ranging-based localization, a network is *localizable* [29] if the locations of all the nodes can be uniquely determined by respecting the constraints of the measured distances and anchor locations. A specific node is *localizable* [51] if its location can be uniquely determined by respecting the constraints. The existence of unlocalizable nodes has been verified to be very general in practical applications such as in OceanSense [52] and GreenOrbs [53]. This motivates the following question: In the commonly seen unlocalizable networks, how to ensure only the BLL localizable nodes participating in the BLL iteration, so the nodes' locations can converge correctly.

Identifying whether a network/node is localizable is called the network/node *localizability* problem. There exist a series of studies, including the *network localizability* conditions [29] and detection algorithms [54]; the *node localizability* conditions [51] and localizable node detection algorithms [29], [55], [56]. However, the above localizability conditions and algorithms are found to be no longer applicable in BLL.

To distinguish related studies, we define the concept of *raw ranging-based localizability* (RRL). The *RRL localizability* refers to localizability studies using the edges (ranging information) in the original distance graph. Meanwhile, the term *BLL localizability* is defined as referring to the localizability problem in the BLL schema. BLL localizability depends on a *generated graph* [46] associated with the barycentric coordinate representation. It is generally sparser than the original distance graph, which will be detailed in Section II. To find the BLL localizable nodes, existing work is very limited. To the best of our knowledge, there is only a sufficient condition called recursive-3DP and a centralized detection algorithm called iterative max-flow (IMF) [57]. But the recursive-3DP condition may miss many localizable nodes, which will be detailed in Section III. The IMF algorithm is centralized, which is not suitable for distributed localizable node detection as BLL requires.

Without distributed localizable node detection methods, existing BLL algorithms impose different restrictions for correct localization. DILOC [42] and its variants [43], [47] require that all agents are within the convex hull of anchors and each agent is within the convex hull of neighbors. If there is not such a convex hull, an agent is allowed to increase its ranging scope. ECHO [46] assumes that the whole network is localizable to make the system nonsingular.

This paper investigates distributed conditions and algorithms to identify BLL localizable nodes, so as to select only localizable nodes in local updating to exclude the impact of unlocalizable nodes. It firstly shows that knowing the implicit edges in the neighborhood is crucial for accurate localizability detection in BLL. An efficient and distributed *negative edge inference (NEI)* scheme is proposed to reveal true neighborhood distance constraints. A new node localizability detection condition is proposed for BLL. Then, a distributed *path extension and pruning (PEP)* algorithm is presented, which is shown to have the same localizable node detection performance as the centralized IMF algorithm. The key contributions are as follows:

- The issues of BLL localization are firstly investigated: (1) we show when some unlocalizable nodes participate in the BLL iteration, even the localizable nodes may converge to wrong locations. (2) The importance of implicit edge to BLL localizability detection is illustrated. (3) The RRL node localizability detection algorithms are shown to fail to detect some BLL localizable nodes and they require special anchor distribution to launch.
- A distributed *negative edge inference* (NEI) algorithm is proposed to infer the implicit edges (the unmeasured edges with unique length) in each node's neighborhood. NEI is highly efficient and uses only one-hop neighborhood distance information.
- Based on the information proved by NEI, a new sufficient condition, i.e., *Recursive-3DP on the strengthened distance graph* is proposed for BLL node localizability detection. It detects much more BLL localizable nodes than the condition without using NEI.
- A distributed *path extension and pruning* (PEP) algorithm is designed for detecting BLL localizable nodes distributively. It does not need special anchor distribution and theoretically guarantees to detect all the Recursive-3DP nodes in the strengthened graph.
- A *Fast-PEP* algorithm is also proposed to detect disjoint paths using the shortest paths. It greatly improves efficiency while preserving the detection capability. Experiments show that Fast-PEP miss very limited number of Recursive-3DP nodes than PEP.
- Then a *Localizability Aware Barycentric linEar Localization framework (LABEL)* is proposed, in which each node runs NEI and PEP (Fast-PEP) to select localizable neighbors to form the distributed linear equation. LABEL can guarantee the correct location convergence. Benefited by PEP and NEI, it discovers and correctly localizes much more localizable nodes than existing methods and thus improves the application scope of BLL algorithms.

The rest of the paper is organized as follows. Preliminary and related works are introduced in Section II. The key observations are presented in Section III. NEI is presented in Section IV. PEP and Fast-PEP are presented in Section V. Evaluations through both real and simulated experiments are presented in Section VI. The paper is concluded with a discussion of future work in Section VII.

II. Preliminaries and Related Work

Let the terminology "node" refer to both the agents and anchors. For a network of $m+n$ nodes in $\Re^{dim}$, let $\mathcal{V} = \mathcal{A} \cup \mathcal{S}$

denote the node set. This paper is presented for $dim = 2$, whose idea can be further extended to $dim = 3$. We assume $\{v_1, \cdots, v_m\} \in \mathcal{A}$ are *anchors*, whose locations $\mathbf{P}_{\mathcal{A}} = [\mathbf{p}_1 \ \cdots \ \mathbf{p}_m]^T$ are known. The nodes $\{v_{m+1}, \cdots, v_{m+n}\} \in \mathcal{S}$ are *agents*, whose locations $\mathbf{P}_{\mathcal{S}} = [\mathbf{p}_{m+1} \ \cdots \ \mathbf{p}_{m+n}]^T$ are to be determined. Let R be the ranging radius, the inter-node distance d_{ij} can be measured and $(i,j) \in \mathcal{E}$ if $||\mathbf{p}_i - \mathbf{p}_j||_2 \leq R$. The anchors, agents, and inter-node distance measurements can be represented by a distance graph $\mathcal{G} = (\mathcal{V}, \mathcal{E}, \mathbf{d})$. $\mathcal{N}_i$ represents the neighbors of v_i in $\mathcal{G}$, $v_j \in \mathcal{N}_i$ if $(i,j) \in \mathcal{E}$. The ranging-based localization problem is to calculate $\mathbf{P}_{\mathcal{S}}$ by anchor locations $\mathbf{P}_{\mathcal{A}}$ and the inter-node distances $\mathbf{d}$.

A. Ranging-Based Localization and Localizability

We firstly introduce different categories of ranging-based localization algorithms. Then we introduce the localizability results in ranging-based localization.

1) Ranging-Based Localization Algorithms: Ranging-based localization algorithms can be divided into *centralized* and *distributed* two categories. Representative centralized localization algorithms include (1) optimization-based methods [11], [12], [13], [14], [15], where a quadratic least-squares objective function is adopted; (2) probabilistic based algorithms, where the measurement data is characterized by a statistical model [16], [17], [18], [19], [20]; (3) geometric-based methods where the noise and ambiguity issues are discussed from the network geometry perspective [21], [22], [23]; and (4) component stitching methods [24], [25], [26], [27], [28] that divide the network into sub-structures and merge the local results, etc. Centralized algorithms need high communication costs and latency for data collection, and optimization-based methods generally face the trouble of local optima due to the non-convexity of the resulting optimization problem.

Distributed localization methods can be broadly categorized into *sequential* and *concurrent* two categories. The representative sequential method is trilateration [29]. When an agent has at least three location-known neighbors, it calculates its location by trilateration. Then, it serves as new location-known node to help localize other agents. To make trilateration more robust, robust quadrilateral [30] and safe triangles [31] are proposed to prune unreliable distance measurements. Another idea is to import redundancy, i.e., using multilateration to improve accuracy [32], [33], [34]. Another direction is to expand trilateration in sparse networks. Biliteration [35] and shadow edge [36] methods are proposed. The idea of trilateration also finds applicability in ranging-free localization methods [58], [59], [60], [61], where the main difference is that the agent-to-anchor distance is obtained by the estimated average hop distance and hop count. In overall, sequential methods are easy to carry out. But they require the anchors to be densely deployed to start trilateration. Moreover, the sequential manner inevitably faces the location error accumulation problem.

In concurrent distributed methods [37], [38], [39], [40], [41], each agent starts with an initial location estimation and then updates its location using distance measurements to its neighbors and the neighbors' estimated locations. The process iterates until the estimations converge. In [38], each node repeatedly and concurrently receives distances and locations from neighbors and calculates a local coordinate system with a random orientation relative to the global coordinate system. Two distributed expectation-conditional maximization (ECM) algorithms have been designed using the idea of average consensus in [39]. A distributed algorithm for least square localization has also been proposed in [40]. BLL is also in the distributed and concurrent branch, which will be introduced separately in Section II-B.

More detailed survey of localization algorithms with various taxonomies can be found in [47], [62], [63], [64], and [65], and references therein.

2) RRL Localizability: We refer to the localizability studies using edge information in the original graph as RRL localizability. A network is *RRL localizable* if all the node locations in the network can be uniquely determined by respecting the distance measurements and the anchor locations. A node is *RRL localizable* if its location can be uniquely determined by the distance and anchor constraints.

• **RRL Network Localizability Theories and Algorithms.** RRL network localizability is generally characterized by the graph's global rigidity property. In $\Re^2$, $\mathcal{G}$ is *global rigid* if and only if it is 3-connected and redundantly rigid [66]. RRL Network localizability detection algorithms return a *yes-or-no* answer for a given network. A polynomial algorithm has been designed by checking 3-connectivity and redundant rigidity [54].

• **RRL Node Localizability Theories and Algorithms.** To characterize node localizability, even in $\Re^2$, the necessary and sufficient condition remains unsolved. For necessary condition, a localizable node needs at least three node disjoint paths to anchors, called 3P condition [67]. Combining 3P and redundant rigidity (the node is in a redundantly rigid component with at least 3 anchors), another necessary condition called RR-3P is proposed in [51]. For sufficient condition, if a node belongs to a redundantly rigid component and it has three vertex disjoint paths to three anchor nodes, the node is RRL localizable, which is called RR3P condition [51]. Existing node localizability detection algorithms are mainly designed based on sufficient conditions.

The network flow and the pebble game algorithm [68] are combined to detect localizable nodes based on RR3P in a centralized manner. Distributed algorithms for localizable node detection have also been proposed, including Trilateration Protocol (TP) [29], Wheel Extension (WE) [55], and Triangle Extension (TE) [56]. But these detection algorithms are based on sufficient conditions, they may still miss to detect some localizable nodes [56].

B. Barycentric Coordinate-Based Linear Localization (BLL) and BLL Localizability

1) BLL Algorithms: • **Construction of Linear System.** For each agent, its location can be expressed as a linear function of neighbors' locations through the barycentric coordinate introduced by $M\ddot{o}bius$ [69]. For an agent v_i, its function is:

$$\mathbf{p}_i = a_{ij}\mathbf{p}_j + a_{ik}\mathbf{p}_k + a_{il}\mathbf{p}_l, \tag{1}$$

where $\{a_{ij}, a_{ik}, a_{il}\}$ are the *barycentric coordinates* of v_i with respect to neighbors v_j, v_k, and v_l. In the seminal work of BLL, i.e., DILOC [42] and its variants [43], [47], v_i is required to locate inside the triangle formed by v_j, v_k, and v_l. Consequently, the barycentric coordinates are always positive. To compute the barycentric coordinates with the neighbors that do not enclose v_i, Diao et al. [46] designed a complex geometric approach by determining the relative position between v_i and neighbors; Some work resorts to RRL methods in neighborhood to calculate barycentric coordinates

TABLE I
Representative Ranging-Based Localization Methods

Centralized				Distributed	
optimization	probabilistic	geometric	component	sequential	concurrent
MDS [11] [12] G2O [13] [14]SDP [15]	[16] [17] [18] [19] [20]	[21] [22] [23]	[24] [25] [26] [27] [28]	[29] [30] [31] [32] [33] [34] [35] [36]	[37] [38] [39] [40] [41] **BLL** [42] [43] [44] [45] [46] [47] [48] [49] [50]

indirectly [48], [49]; Recently, Tecchio et al. [50] introduced a direct and efficient calculation approach. For any three neighbors $\{v_j, v_k, v_l\}$, the coefficients are:

$$\begin{cases} a_{ij} = D(v_j, v_k, v_l; v_i, v_k, v_l)/D(v_j, v_k, v_l; v_j, v_k, v_l), \\ a_{ik} = D(v_j, v_k, v_l; v_j, v_i, v_l)/D(v_j, v_k, v_l; v_j, v_k, v_l), \\ a_{il} = D(v_j, v_k, v_l; v_j, v_k, v_i)/D(v_j, v_k, v_l; v_j, v_k, v_l). \end{cases} \tag{2}$$

where $D(\cdot)$ is the *Cayley-Menger bideterminant* [70]. It takes the distances between two node sets with equal cardinality as input and outputs a signed scalar.

$$\begin{aligned} &D(s_1, \cdots, s_k; t_1, \cdots, t_k) \\ &= 2(-\frac{1}{2})^k \begin{vmatrix} 0 & 1 & 1 & \cdots & 1 \\ 1 & d^2_{s_1,t_1} & d^2_{s_1,t_2} & \cdots & d^2_{s_1,t_k} \\ 1 & d^2_{s_2,t_1} & d^2_{s_2,t_2} & \cdots & d^2_{s_2,t_k} \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 1 & d^2_{s_k,t_1} & d^2_{s_k,t_2} & \cdots & d^2_{s_k,t_k} \end{vmatrix}, \end{aligned} \tag{3}$$

where d_{s_i,t_i} means the distance between node s_i and node s_i. It can be seen that the three neighbors are required to be **mutually connected** (i.e., forming a triangle) to calculate (3). Let Θ denote three mutually connected neighbors. By permutation, v_i can construct a set $\{\Theta_1, \cdots, \Theta_M\}$ where M is the number of triangles formed by neighbors. Then, the barycentric coordinate representation becomes:

$$\mathbf{p}_i = \frac{1}{M} \sum_{k=1}^{M} \sum_{j \in \Theta_k} a_{ij} \mathbf{p}_j, \tag{4}$$

where the barycentric coordinates are calculated as in (2).

Using the barycentric representations (4), from the system point of view, a linear system can be constructed. In a matrix–vector notation, the linear system is:

$$\begin{bmatrix} \mathbf{P}_{\mathcal{A}} \\ \mathbf{P}_{\mathcal{S}} \end{bmatrix} = \begin{bmatrix} \mathbf{I} & \mathbf{0} \\ \mathbf{B} & \mathbf{C} \end{bmatrix} \begin{bmatrix} \mathbf{P}_{\mathcal{A}} \\ \mathbf{P}_{\mathcal{S}} \end{bmatrix}. \tag{5}$$

Let $\mathbf{A} = \begin{bmatrix} \mathbf{I} & \mathbf{0} \\ \mathbf{B} & \mathbf{C} \end{bmatrix} \in \mathbb{R}^{(m+n)\times(m+n)}$. The ith row of $\mathbf{A}$ is the barycentric coordinate of v_i w.r.t. its neighbors. So the localization problem is transformed to the linear system:

$$(\mathbf{I} - \mathbf{C})\mathbf{P}_{\mathcal{S}} = \mathbf{B}\mathbf{P}_{\mathcal{A}}. \tag{6}$$

where $(\mathbf{I} - \mathbf{C})$ is called the coefficient matrix. Based on the linear system, a *generated graph* of $\mathcal{G}$ is defined.

Definition 1 (The Generated Graph of $\mathcal{G}$): Given $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ and the matrix $\mathbf{A}$ constructed by (1)-(5) using neighbor triangle permutation and barycentric coordinates, the generated graph $\mathcal{G}^{\mathcal{A}} = (\mathcal{V}, \mathcal{E}^{\mathcal{A}})$ is defined as a graph with the same $\mathcal{V}$ and $(i, j) \in \mathcal{E}^{\mathcal{A}}$ if $\mathbf{A}_{(i,j)} \neq 0$.

The neighbors of v_i in $\mathcal{G}^{\mathcal{A}}$ are called *barycentric neighbors*, denoted by $\mathcal{N}_i^{\mathcal{A}}$. $v_j \in \mathcal{N}_i^{\mathcal{A}}$ if $(i, j) \in \mathcal{E}^{\mathcal{A}}$.

• **Distributed Linear Iteration.** After constructing the linear system locally, each node $v_i \in \mathcal{S}$ updates $\mathbf{p}_i$ in an iterative updation and propagation manner. Let $\hat{\mathbf{p}}_i$ represent the estimated location of v_i. For example, ECHO [46] adopts the Richardson iteration to update $\hat{\mathbf{p}}_i$ with the help of an intermediate variable:

$$\begin{cases} \boldsymbol{\eta}_i(t) = \hat{\mathbf{p}}_i(t) - \sum_{v_j \in \mathcal{N}_i^{\mathcal{A}}} a_{ij} \hat{\mathbf{p}}_j(t), \\ \hat{\mathbf{p}}_i(t+1) = \hat{\mathbf{p}}_i(t) - \epsilon \boldsymbol{\eta}_i(t) + \epsilon \sum_{v_j \in \mathcal{N}_i^{\mathcal{A}}} a_{ji} \boldsymbol{\eta}_j(t), \end{cases} \tag{7}$$

where a_{ij} is the barycentric coordinate of v_i w.r.t. v_j and ϵ is a precalculated scalar to control convergence. Other distributed updating algorithms include distributed gradient descent [70], distributed conjugate residual [71], and distributed conjugate gradient (DCG) [72], etc. Note that such solvers in BLL are independent to the initialization $\hat{\mathbf{p}}_i(0)$. Moreover, BLL in noisy environments [44], [50], dynamic networks [73], and 3D spaces [49] have also been explored. A survey about BLL can be seen in [64].

It can be observed that the routine of BLL involves only neighborhood information and local concurrent calculation for each node. If the linear system is non-singular, the locations of agents will converge under any initialization.

2) BLL Localizability: • **BLL Network Localizability Theory and Algorithm.** BLL localizability is characterized by the *generated graph* $\mathcal{G}^{\mathcal{A}}$.

Lemma 1 (BLL Network Localizability Condition [46]): In BLL algorithms, all nodes in a graph $\mathcal{G}$ are localizable in $\Re^2$ if and only if every node can find at least three node-disjoint paths to anchors through only edges in $\mathcal{G}^{\mathcal{A}}$ generated from $\mathbf{A}$. Or equivalently, $\mathbf{I} - \mathbf{C}$ constructed from (1)- (6) is non-singular.

For BLL network localizability, our early work presented a flow-based algorithm to test whether Lemma 1 is satisfied [57] for judging whether a network is BLL localizable.

• **BLL Node Localizability Theory and Algorithm.** The necessary BLL node localizable condition requires the node to have at least 3 mutually connected neighbors. The sufficient BLL node localizability condition was firstly given in [57], which is called *Recursive-3DP*.

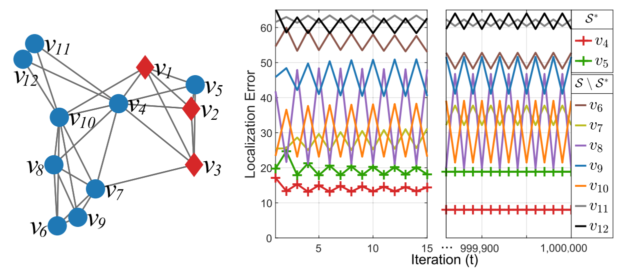
Lemma 2 (Sufficient BLL Node Localizability Condition (Recursive-3DP) [57]): In a graph $\mathcal{G} = \{V, E\}$ in $\Re^2$, suppose $\mathcal{G}^{\mathcal{A}}$ is obtained through neighbor triangle permutation. If 1) v_i has three node disjoint paths to anchors in $\mathcal{G}^A$, and 2) nodes on the disjoint paths all have three node disjoint paths to anchors in $\mathcal{G}^A$, then, node v_i is BLL localizable in $\mathcal{G}$.

An Iterative Max-Flow (IMF) algorithm for detecting BLL localizable nodes has been proposed in [57]. But IMF is centralized, which cannot be run-timely used by BLL which requires distributed localizable node detection algorithm.

Table I summarizes the representative centralized and distributed ranging-based algorithms. A summary of the localizability conditions, the centralized and distributed localizability detection algorithms for RRL and BLL are given in Table II.

TABLE II
REPRESENTATIVE WORK ABOUT LOCALIZABILITY IN DISTANCE GRAPHS. ×: UNSOLVED

	Network Localizability		Node Localizability				
	Necessary and Sufficient Condition	Testing Algorithm	Necessary and Sufficient Condition	Necessary Condition	Sufficient Condition	Centralized Detection Algorithm	Distributed Detection Algorithm
RRL	Global rigid [29]	A polynomial algorithm [54]	×	3P [67]RR-3P [51]	RR3P [51]	network flow and pebble game [68]	TP [29]WE [55]TE [56]
BLL	Each agent has 3DP to anchors [46]	Max-Flow Algorithm [57]	×	3 mutually connected neighbors [57]	Recursive-3DP [57]	Iterative Max-Flow (IMF) [57]	×



(a) (b)

Fig. 1. The locations calculated by ECHO in a network consisting BLL unlocalizable nodes. (a) The graph topology. (b) The evolution of localization error w.r.t. iteration rounds.

Although BLL is attractive for the four properties mentioned in Section I, the research about BLL node localizability is deficient, which will be addressed in this paper.

III. KEY OBSERVATIONS

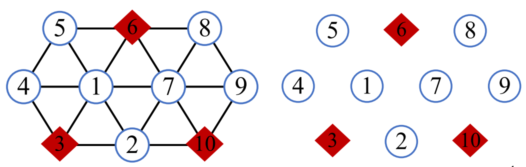
Why BLL node localizability is crucial is explained by showing the impact of unlocalizable nodes in BLL. Then two key difficulties for BLL localizable node detection are illustrated.

A. The Impact of Unlocalizable Nodes in BLL

In existing BLL methods, an implicit requirement is the nodes participating in location iteration are BLL localizable. But in practical applications, it is generally inevitable that some nodes are unlocalizable, especially in sparse networks. If we do not follow the restricts in DILOC [42] and its variants [43], [47] that all nodes locating inside a convex hull composed of anchors, and do not follow the assumption in ECHO that all nodes are localizable, let's check what will happen. We show that the unlocalizable nodes will damage the localization correctness of the localizable nodes. To proceed, the concept of correct convergence is clarified.

Definition 2 (Correct Convergence): In a generic network running BLL algorithms, a node v_i's location $\hat{\mathbf{p}}_i$ converges correctly, if $\forall \varepsilon > 0$, $\exists$ a time τ, such that $||\hat{\mathbf{p}}_i(t) - \mathbf{p}_i||_2 \leq \varepsilon$ for all $t \geq \tau$.

It means the node's calculated location $\hat{\mathbf{p}}_i$ converges towards its true location $\mathbf{p}_i$ as time passing. A network consisting unlocalizable nodes is shown in Fig. 1(a), where $\{v_1, v_2, v_3\}$ are anchors. Using the IMF algorithm, $\{v_4, v_5\}$ are BLL localizable while $\{v_6, \cdots, v_{12}\}$ are BLL unlocalizable. Let $||\hat{\mathbf{p}}_i(t) - \mathbf{p}_i||_2$ denote the localization error at the tth iteration. Fig. 1(b) shows the localization error evolution of each v_i when the BLL algorithm ECHO is adopted. Each $\hat{\mathbf{p}}_i$ is initialized from $\hat{\mathbf{p}}_i(0) = [0, 0]$. Let $\mathcal{S}^*$ represent the BLL localizable nodes. It can be seen that even the BLL localizable nodes $\hat{\mathbf{P}}_{\mathcal{S}^*}$ cannot converge to their true locations after even 10^8 iterations.



(a) A wheel extension graph. (b) The corresponding $\mathcal{G}^{\mathcal{A}}$.

Fig. 2. A wheel extension graph and its $\mathcal{G}^{\mathcal{A}}$.

The reason is that the unlocalizable nodes v_6 to v_{12} do not have enough constraints to converge to correct locations. But they continuously broadcast their wrong locations to impact neighbors' location updates. This makes even the theoretically localizable nodes cannot converge correctly.

So it is critical for each node to identify its own and its neighbors' BLL localizability. Only localizable nodes should select localizable neighbors to construct the linear model. However, the localizable node detection theories and algorithms are lacking in BLL. We next explain the main theoretical and algorithmic difficulties, respectively.

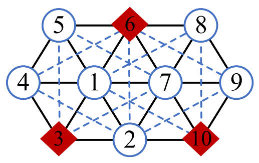
B. Requiring Stronger Neighborhood Connectivity

Since the necessary and sufficient condition for node localizability is unsolved, existing identification methods are all based on sufficient conditions. Thus, it is inevitable that some truly localizable nodes will be wrongly characterized as unlocalizable by sufficient conditions. This problem is especially serious in BLL localizable node detection. An example is shown in Fig. 2, where nodes {3, 6, 10} are anchors. Since the graph is a wheel extension graph, all agents are RRL localizable [55]. But in BLL, no node can find three mutually connected neighbors when constructing $\mathcal{G}^{\mathcal{A}}$. Then, $\mathbf{A}$ is an all-zero matrix and $\mathcal{E}^{\mathcal{A}} = \emptyset$. Thus, no node is BLL localizable using Recursive-3DP.

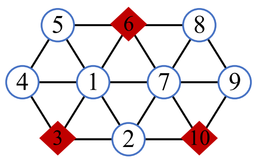
The key problem is that the BLL node localizability requires more edges in each node's neighborhood. The lack of edges leads to an inability to compute the Cayley-Menger bi-determinant when a node has not three mutually connected neighbors. Thus, the localizable nodes are wrongly marked as unlocalizable.

In distance graphs, although some edges cannot be directly measured, many of them are "implicitly" existing. In specific, when $(v_i, v_j) \notin \mathcal{E}$ (i.e., $||\mathbf{p}_i - \mathbf{p}_j||_2 > R$), there is an *implicit edge* between v_i and v_j if the possible distance between them is unique. Yang et al. [51] provided a way to infer some of the implicit edges based on graph partition. If $\mathcal{E}$ can be partitioned into two parts, and v_i and v_j are constrained in a rigid component in each partition, then (v_i, v_j) is implicit. The implicit edges are added into $\mathcal{G}$ to form an extended graph $\mathcal{G}^I$ before counting localizable nodes through RR3P. With these implicit knowledge, the RRL localizable nodes can be more accurately detected.

However, Yang's method needs graph-level information. Whereas in BLL, each node knows only connectivity among

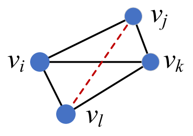


(a) $\mathcal{G}^+$ after NEI.

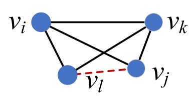


(b) The corresponding $\mathcal{G}^{\mathcal{A}+}$.

Fig. 3. A wheel extension graph after NEI and its new generated graph. The blue dotted lines represent edges inferred by NEI.



(a) d_{jl}^+



(b) d_{jl}^-

Fig. 4. The two possible lengths of the negative edge in $\mathcal{BFG}_{ijkl}$.

neighbors and the BLL localizability also depends on the neighborhood connectivity. Further, the partition-based method misses many implicit edges, e.g., (v_2, v_4) in Fig. 2(a). We cannot find a partition of edges such that v_2 and v_4 are in rigid components in each partition, but (v_2, v_4) is actually implicit since its length can be uniquely determined.

This motivates the design of a fully distributed approach to infer implicit connectivity among neighbors, which is called Negative Edge Inference (NEI) and will be detailed in Section IV. Then, a strengthened distance graph is defined.

Definition 3 (Strengthened Distance Graph): For a distance graph $\mathcal{G} = \{\mathcal{V}, \mathcal{E}\}$, its corresponding strengthened distance graph is $\mathcal{G}^+ = \{\mathcal{V}, \mathcal{E}^+\}$, where $\mathcal{E}^+$ is the union of $\mathcal{E}$ and the implicit edges added by NEI.

Given $\mathcal{G} = \{\mathcal{V}, \mathcal{E}\}$ with at least 3 anchors in $\Re^2$, a new sufficient BLL node localizability condition is proposed based on $\mathcal{G}^+$ and its corresponding generated graph $\mathcal{G}^{\mathcal{A}+}$.

Theorem 1 (Recursive-3DP in $\mathcal{G}^{\mathcal{A}+}$): If 1) a node v_i has 3 node disjoint paths (3DP) to anchors in $\mathcal{G}^{\mathcal{A}+}$, and 2) the nodes on the three disjoint paths to anchors all have 3DP to anchors in $\mathcal{G}^{\mathcal{A}+}$. Then, v_i is BLL localizable in $\mathcal{G}$.

Proof: In $\mathcal{G}^+$, more edges with unique lengths are involved. If both 1) and 2) are satisfied, the graph induced by v_i and the nodes it passes by on its paths from $\mathcal{G}^{\mathcal{A}+}$ satisfies Lemma 1. Then all these nodes are BLL localizable. □

Fig. 3(a) shows the $\mathcal{G}^+$ of Fig. 2(a), where the dashed lines are the implicit edges inferred by NEI. Then the generated graph $\mathcal{G}^{\mathcal{A}+}$ in BLL is shown in Fig. 3(b). It can be verified that all nodes become BLL localizable using Recursive-3DP in $\mathcal{G}^{\mathcal{A}+}$. Comparing with Recursive-3DP in $\mathcal{G}$, NEI helps to find much more theoretically BLL localizable nodes. A necessary condition for BLL node localizability detection is proposed based on the property of $\mathcal{G}^+$.

Theorem 2 (Mutually Connected 3 Neighbors (MC3N) in $\mathcal{G}^+$): If a node is BLL localizable in $\mathcal{G}$, it has at least 3 mutually-connected neighbors in $\mathcal{G}^+$.

Proof: From (1)-(3), the construction of the BLL model is based on Cayley-Menger bideterminant, where the distance between each pair of nodes from the input $\{v_1, \cdots, v_k\}$ is involved. If a node is BLL localizable, it must have at least 3 mutually-connected neighbors in $\mathcal{G}^+$, otherwise the linear equation for location updating cannot be set up. □

Thus theoretically, if a node meets the Recursive-3DP in $\mathcal{G}^{\mathcal{A}+}$, it is BLL localizable. If a node does not meet the MC3N condition, it is BLL unlocalizable.

Please also note that in Fig. 2(a), the three anchors don't have common neighbors, so trilateration cannot startup although all nodes are RRL localizable. BLL does not require special anchor distribution to launch.

C. Limitations in Distributed BLL Node Localizability Detection Algorithms

Distributed RRL localizable node detection algorithms, including TP [29], WE [55], and TE [56] can be transformed to detect BLL localizable nodes if they are applied onto $\mathcal{G}^{\mathcal{A}}$ rather than $\mathcal{G}$. Because these algorithms are detecting localizable nodes based on sufficient localizable conditions on $\mathcal{G}^{\mathcal{A}}$, their detected subgraph satisfied the network localizable condition in Lemma 1. So their detected nodes on $\mathcal{G}^{\mathcal{A}}$ are BLL localizable. But for the following limitations, they may miss to detect many BLL localizable nodes.

• **(Launch condition).** TP, WE, and TE require special anchor distribution to start, which is called *launch condition*.

1) In TP, for starting trilateration, at least three anchors are required and the anchors need to have at least one common neighbor;
2) In WE, there need at least one wheel containing at least three anchors, otherwise, the wheel structure cannot be extended so that no localizable node can be detected;
3) In TE, for starting localizability detection by branching, at least two anchors need to have common neighbors.

These launch conditions greatly limit the application scope of these algorithms, since the anchor nodes are may not be deployed as expected.

• **(Miss to detect Recursive-3DP nodes).** Another problem of TP, WE, and TE is the miss of detection of many BLL localizable nodes satisfying Recursive-3DP, since TP, WE, and TE are designed based on conservative sufficient conditions, i.e., trilateration and bilateration.

We can see more intuitive examples in Section VI-C.1. Their detection capability will also be detailed in the evaluation section.

IV. Negative Edge Inference by Neighborhood Information

This section details the Negative Edge Inference (NEI) approach.

A. The Building Block of NEI

Recall that any three neighbors v_j, v_k, v_l of a node v_i contribute to v_i's BLL localizability only if they are mutually connected. We consider the case where only two pairs of the three neighbors are connected. Without loss of generality, say that (v_j, v_k) and (v_k, v_l) are connected while (v_j, v_l) is not measured as shown in Fig. 4(a). Although v_j, v_k, v_l are in $\mathcal{N}_i$, none of them is the neighbor of v_i in $\mathcal{G}^{\mathcal{A}}$ due to the absence of (v_j, v_l). We regard such a graph with four vertices and five edges as a building block for negative edge inference, which is called a *basic flipping graph* ($\mathcal{BFG}$). The missed edge (v_j, v_l) is the *negative edge* under investigation. Such $\mathcal{BFG}$s can be widely seen in practical networks.

In our early work [65], it has been proved that the length of the negative edge d_{jl} in the $\mathcal{BFG}$ has and only has two possible lengths, denoted by d_{jl}^+ (Fig. 4(a)) and d_{jl}^- (Fig. 4(b)). In this paper, we re-derive the calculation of d_{jl}^+ and d_{jl}^- using the Cayley–Menger bideterminants constructed by the

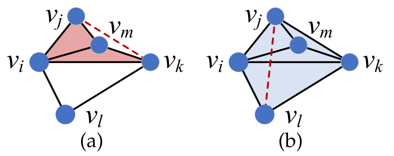


Fig. 5. Negative edge inference in $\{v_i, v_j, v_k, v_l\}$ with the help of $\mathcal{BFG}_{ijmk}$. (a) Infer d_{jk} in $\mathcal{BFG}_{ijmk}$; (b) Infer d_{jl} in $\mathcal{BFG}_{ijkl}$.

five known edges. In $\mathcal{BFG}_{ijkl}$, write $\angle jik, \angle lik$ as α and β, respectively. Their cosine values can be calculated as [74]:

$$\cos\alpha = \frac{D(v_i, v_j; v_i, v_k)}{\sqrt{D(v_i, v_j)D(v_i, v_k)}},$$
$$\cos\beta = \frac{D(v_i, v_l; v_i, v_k)}{\sqrt{D(v_i, v_l)D(v_i, v_k)}}. \tag{8}$$

$D(\cdot)$ is the Cayley–Menger bideterminant as in (3). Then α and β are known by $arc\cos(\cdot)$. In Δijl, d^+_{jl} and d^-_{jl} are obtained using the law of cosines:

$$d^+_{jl} = \sqrt{d^2_{ij} + d^2_{il} - 2d_{ij}d_{il}\cos(\alpha+\beta)},$$
$$d^-_{jl} = \sqrt{d^2_{ij} + d^2_{il} - 2d_{ij}d_{il}\cos(\alpha-\beta)}. \tag{9}$$

Then a condition to judge whether the negative edge has a unique length, i.e., whether the negative edge is implicit, is presented.

Theorem 3 (Negative Edge Inference [65]): For an arbitrary $\mathcal{BFG}_{ijkl}$, the length of edge (v_j, v_l) can be uniquely inferred as d^+_{jl} if $d^-_{jl} \le R$, where R is the maximum ranging radius.

Thus, the *negative edge inference* is detailed as:

$$d_{jl} = d^+_{jl},$$
$$\text{If } \{v_i, v_j, v_k, v_l\} \text{ is a } \mathcal{BFG}, \text{and } d^-_{jl} \le R. \tag{10}$$

B. Negative Edge Inference (NEI)

$\mathcal{BFG}$ can be further constructed with the help of newly identified implicit edges. In Fig. 5(a), two edges, i.e., (v_j, v_l) and (v_j, v_k), are not measured, so (10) cannot be applied in $\{v_i, v_j, v_k, v_l\}$. But if (v_j, v_k) is inferred by $\mathcal{BFG}_{ijmk}$, $\{v_i, v_j, v_k, v_l\}$ becomes a new $\mathcal{BFG}$ and (10) can be applied to infer d_{jl} as in Fig. 5(b).

The above scenario implies that negative edge inference should be conducted in an iterative manner. Algorithm 1 shows the procedure for iterative negative edge inference at a node v_i. The statement "local distances" of v_i means the distances to its neighbors and the inter-neighbor distances. If a node v_i has less than three neighbors, NEI terminates since it cannot construct any $\mathcal{BFG}$ (Line 1-2). But such v_i can launch NEI if its neighbors successfully infer implicit edges and inform it; Then v_i enumerates any three-node combinations in its neighbors and checks whether they can form a $\mathcal{BFG}$ with v_i (Line 4-6). Whenever an implicit edge is inferred, neighbors should be informed (Line 9-10). A variable c is used to control the inference loop. Node v_i repeats the edge inference procedure if 1) v_i successfully infers at least one edge, i.e., $c \ge 1$, or 2) v_i receives informed messages from neighbors who successfully infers implicit edges using $\mathcal{BFG}$ containing v_i (Line 11-13).

C. Properties of NEI

• **NEI Improves Node Localizability.** NEI better reveals the true connectivity among neighbors, which is exactly what

Algorithm 1 Negative Edge Inference (NEI)

```
Input: neighbors: $\mathcal{N}_i$; local distances;
Output: updated $\mathcal{N}_i$; updated local distances;
1  if $|\mathcal{N}_i| < 3$ then
2      return // cannot construct $\mathcal{BFG}$
3  $c \leftarrow 0$;
4  for each three-node combinations $\{v_j, v_k, v_l\}$ in $\mathcal{N}_i$ do
5      if $\{v_i, v_j, v_k, v_l\}$ is a $\mathcal{BFG}$ then
6          try to infer negative edge as (10);
7          if $d_{jl}$ is successfully inferred then
8              $c \leftarrow c + 1$;
9              inform $v_j$ to add $v_l$ to $\mathcal{N}_j$ and record $d_{jl}$;
10             inform $v_l$ to add $v_j$ to $\mathcal{N}_l$ and record $d_{jl}$;
11 if $c > 0$ or receives any inform message from $\mathcal{N}_i$ then
12     update $\mathcal{N}_i$ and local distances;
13     go back to Line 3;
14 return updated $\mathcal{N}_i$ and updated local distances.
```

BLL node localizability requires as we have mentioned. Thus, the BLL node localizability can be significantly improved. For example, v_i in Fig. 5(a) has four neighbors but none of them can contribute to v_i's BLL localizability since they cannot form triangle to calculate barycentric representation. After NEI, two triangles can be found, i.e., Δjkm and Δjkl. Then all the four neighbors also become barycentric neighbors of v_i in $\mathcal{G}^{\mathcal{A}+}$. In addition, it will be shown that NEI also greatly helps in improving RRL localizability.

• **Convergence of NEI.** For a graph $\mathcal{G} = \{\mathcal{V}, \mathcal{E}\}$, $|\mathcal{V}| = n$ and $|\mathcal{E}| = m$. Suppose $\mathcal{G}^+$ has Λ edges after adding all negative edges. If no edge is added in a certain round, NEI converges; In the worst case, only one edge is added in each round. NEI converges within $\Lambda - m$ rounds. Since $\Lambda \le n(n-1)/2$ edges, thus NEI theoretically converges at most $n(n-1)/2 - m$ rounds. In actual experiments, NEI adds more than one edge in one round and converges much faster.

• **Communication message cost.** The calculation in both NEI and barycentric coordinates uses Cayley-Menger bideterminants are oriented to a node v_i and its three neighbors, so no additional communication cost is needed in conducting NEI. Although NEI does not consider multi-hop implicit edges, they do not impact the construction of $\mathcal{G}^{\mathcal{A}}$, since each node's location is represented only by direct neighbor's locations. For a node v_i, only the IDs of the endpoints of the inferred edge are involved in the *'inform'* message. Suppose the network has n nodes and each node's neighbor count is bounded by Δ. Thus, in the worst case, v_i has to send a message consisting of only IDs of $n - \Delta$ pairs of nodes.

• **Time complexity.** For complexity analysis, we consider the node degree of a network $\mathcal{G} = \{\mathcal{V}, \mathcal{E}\}$ is bounded by a threshold Δ, i.e., $|\mathcal{N}_i| \le \Delta, \forall v_i \in \mathcal{V}$. NEI checks any combination of three-neighbors for finding $\mathcal{BFG}$, which has complexity $O(\Delta^3)$. If an edge is successfully inferred, it needs to inform corresponding neighbors to update the local distances. At most Δ nodes need to be informed. Thus, the time complexity of NEI is $O(\Delta^3)$, which is highly efficient.

Overall, NEI is fully distributed, easy to be implemented, effective, and reliable.

D. NEI in More Realistic Scenarios

Although NEI is designed based on the ideal disc model, it can be applied in more realistic scenarios, including: 1) noisy ranging, 2) communication link failure, 3) and error decoding of neighbor states.

• **Noisy distance measurements.** Denote the noisy measurement by $d_{ij} = \|\mathbf{p}_i - \mathbf{p}_j\|_2 + \sigma_{ij}\|\mathbf{p}_i - \mathbf{p}_j\|_2$, where $\sigma_{i,j} \sim N(0,\sigma^2)$ is zero mean Gaussian noise. Then the ranging noises have more than 0.95 probability to be within the range of $[-3\sigma, 3\sigma]$. To tolerate ranging noises, the negative edge inference condition in (10) can be modified as:

$$\begin{aligned}&d_{jl} = d_{jl}^{+},\\ &\text{If } d_{jl}^{+} > R + 3\sigma \text{ and } d_{jl}^{-} \le R - 3\sigma.\end{aligned} \tag{11}$$

• **Communication link failure.** A general link failure model is considered for any node v_i. Suppose its communication link to a neighbor v_j fails with probability $1 - q_{ij}$ at each iteration. Link failures cause less $\mathcal{BFG}$ can be constructed. As [42], [44], we suppose that the probabilities $\{q_{ij}(1), \cdots, q_{ij}(t)\}$ are independent. It implies that a certain link is not likely to fail continuously. Since NEI is efficient, v_i can check its neighbors for χ times. When new neighbors are found, v_i re-conducts NEI to decrease the impact of link failure. Experiments under different q_{ij} and χ will show that NEI works well in the presence of link failure.

• **Noisy neighbor states.** Even if a communication link (v_i, v_j) is active, v_i can only receive a corrupt version of $\mathbf{p}_j$ (v_j's location). Let $\tilde{\mathbf{p}}_j$ be the received state, $\tilde{\mathbf{p}}_j(t) = \hat{\mathbf{p}}_j(t) + \mathbf{n}_{ij}(t)$, where $\mathbf{n}_{ij}(t)$ indicates the channel noise. Such noise affects distributed calculation of node locations, which can be handled by averaging historical information [42], [44]. However, NEI only needs connectivity information and distance information. Thus, NEI is not affected in this scenario.

V. Distributed BLL Localizability Detection Algorithms

After strengthening neighborhood connectivity, the next problem is how each node determines its BLL localizability distributively. This section presents two distributed algorithms: (1) a distributed Path Extension and Pruning (PEP) algorithm which guarantees to detect all nodes that satisfy the Recursive-3DP condition, (2) a Fast Path Extension and Pruning (Fast-PEP) algorithm which greatly saves the path calculation and communication costs than PEP but has BLL node localizability detection performance very close to PEP. Hereinafter, the explanation of the algorithms is oriented to a single node v_i to show how the algorithms work in a distributed manner.

A. Path Extension and Pruning (PEP) Algorithm

PEP can be decomposed into constructing $\mathcal{G}^{\mathcal{A}}$ and then finding the Recursive-3DP nodes. Verifying Recursive-3DP needs to check the existence of *three disjoint paths* (3DP), which are inherently multi-hop information. In distributed multi-agent networks, *distributed consensus algorithms* are usually adopted for acquiring multi-hop information [71], [75], [76], where each node only interacts with neighbors.

PEP is designed leveraging the distributed consensus idea. (1) In the extension stage, each node learns whether it has 3DP to anchors in $\mathcal{G}_{\mathcal{A}}$ through distributed consensus. A node is called a *3DP node* if it has 3DP to anchors. Otherwise, it is called a *non-3DP node*; (2) In the pruning stage, since the 3DP to anchors must reside in $\mathcal{G}_{\mathcal{A}}$, each node prunes the paths passing through *non-3DP nodes* and rechecks its 3DP property. After path pruning, if a 3DP node no longer has 3DP to anchors, it is excluded from 3DP nodes. The removal of a 3DP node will affect the paths passing through it. The process repeats until each remaining 3DP node finds 3DP to anchors by only passing 3DP nodes, i.e., the remaining 3DP nodes satisfy Recursive-3DP and are BLL localizable.

1) Path Extension (PE) Stage for 3DP Detection: Each v_i stores a binary flag ι_i to indicate whether v_i has found 3DP to anchors. Its paths to anchors are recorded by a list of paths $\mathcal{LP}_i$. Each element $\mathcal{P} \in \mathcal{LP}_i$ is a path starting from an anchor or a node with $\iota = 1$.

Algorithm 2 Path Extension (PE)

```
   Input: Neighbor Set: $\mathcal{N}_i$
   Output: Indicator for 3DP: $\iota_i$; List of Paths: $\mathcal{LP}_i^{local}$;
   /* Initialization */
 1 $\mathcal{N}_i^{\mathcal{A}} \leftarrow cal\_bary\_nei(\mathcal{N}_i)$, $t \leftarrow 0$;
 2 $\mathcal{LP}_i(t) \leftarrow \{v_i\}$, $\iota_i \leftarrow 1$, if $v_i \in \mathcal{A}$;
 3 $\mathcal{LP}_i(t) \leftarrow \emptyset$, $\iota_i \leftarrow 0$, if $v_i \notin \mathcal{A}$;
   /* Path Extension at Iteration $t$ */
 4 receive path lists from $v_j \in \mathcal{N}_i^{\mathcal{A}}$;
 5 if $\iota_i = 1$ or $\mathcal{LP}_{\mathcal{N}_i^{\mathcal{A}}}(t) = \emptyset$ then
       continue;
 6 $\mathcal{LP}_i^{merge} \leftarrow \mathcal{LP}_{\mathcal{N}_i^{\mathcal{A}}}(t) \cup \mathcal{LP}_i(t)$ ;
 7 if $detect\_3DP(\mathcal{LP}_i^{merge}) = true$ then
 8     $\mathcal{LP}_i^{local} \leftarrow \mathcal{LP}^{merge}$;
 9     $\mathcal{LP}_i(t+1) \leftarrow \{i\}$, $\iota_i \leftarrow 1$ // 3DP detected
   else
10     $\mathcal{LP}_i(t+1) \leftarrow$ Aggregate Path List$(\mathcal{LP}^{merge})$;
   // Transmit Path List
11 if $\mathcal{LP}_i(t+1) \neq \emptyset$ and $\mathcal{LP}_i(t+1) \neq \mathcal{LP}_i(t)$ then
12     send $\mathcal{LP}_i(t+1)$ to $\mathcal{N}_i^{\mathcal{A}}$;
   Function Aggregate Path List($\mathcal{LP}$)
13     add $v_i$ to the end of each path in $\mathcal{LP}$;
14     return $\mathcal{LP}$;
```

Algorithm 2 shows the PE routine at node v_i. For initialization (Line 1-3), ι_i is set as 1 and $\mathcal{LP}_i$ is $\{v_i\}$ if v_i is an anchor; ι_i is set as 0 and $\mathcal{LP}_i$ is $\emptyset$ if v_i is an agent. Each node exchanges paths with its barycentric neighbors $\mathcal{N}_i^{\mathcal{A}}$, which is calculated by the function $cal\_bary\_nei(\mathcal{N}_i)$. This function checks each combination of three nodes from $\mathcal{N}_i$. A combination of three neighbors are added to $\mathcal{N}_i^{\mathcal{A}}$ if they are mutually connected.

At the tth iteration of PE, the local behavior of a node v_i has three steps (Line 4-12):

• **Receive Path List.** v_i receives $\mathcal{LP}_j(t)$ from $v_j \in \mathcal{N}_i^{\mathcal{A}}$ and merges the received path list with its own path list, i.e., $\mathcal{LP}_i^{merge} = \mathcal{LP}_{\mathcal{N}_i^{\mathcal{A}}}(t) \cup \mathcal{LP}_i(t)$.

• **Detect 3DP.** v_i detects whether there are three node disjoint paths to anchors in $\mathcal{LP}_i^{merge}$ by a function $detect\_3DP$. This function checks each combination of three paths until it finds three paths that share no common vertex or there is no more three path combination.

1) If v_i detects 3DP, it sets ι_i to 1. It is now a 3DP node and clears its path list, i.e., $\mathcal{LP}_i(t+1) = \{v_i\}$. This step reduces the complexity of path list maintenance. v_i also stores $\mathcal{LP}_i^{merge}$ as $\mathcal{LP}_i^{local}$ for later use in path pruning;
2) If v_i does not detect 3DP, it adds its own ID to the end of each path in $\mathcal{LP}_i^{merge}$ and updates $\mathcal{LP}_i(t+1)$.

• **Send Path List.** v_i transmits $\mathcal{LP}_i(t+1)$ to barycentric neighbors if its path list changes in this time step and the progress repeats in the next round. PE stops if v_i cannot receive paths from $\mathcal{N}_i^{\mathcal{A}}$ or v_i detects 3DP at a certain t.

After PE, each v_i knows whether it has 3DP to anchors by exchanging $\mathcal{LP}$ with neighbors.

2) Path Pruning Stage and the Design of PEP: To meet the Recursive-3DP condition, a BLL localizable node should ensure its 3DP to anchors pass through only 3DP nodes. But after PE, there may be some 3DP nodes whose disjoint paths to anchors pass through non-3DP nodes. Thus, a pruning stage following PE is designed to exclude the 3DP nodes that have paths passing through non-3DP nodes. The target is to find a generated graph $\mathcal{G}_{\mathcal{A}}$ where all nodes in it have 3DP to anchors.

In the pruning stage, each non-3DP node firstly broadcasts that "*I am not a 3DP node*". When a 3DP node receives the above message, it removes paths containing the non-3DP node from its $\mathcal{LP}^{local}$ and then rechecks whether 3DP still exist in $\mathcal{LP}^{local}$. If it no longer has 3DP, it will also broadcast "*I am not a 3DP node*" to barycentric neighbors. The process repeats to exclude 3DP nodes until the remaining 3DP nodes are not affected by the non-3DP nodes. As a result, the sufficient BLL localizability condition Recursive-3DP is met.

Algorithm 3 shows the complete procedure of PEP at node v_i. The flag ι_i in the pruning stage indicates the Recursive-3DP property. $\iota_i(0)$ is the state after PE terminates (Line 1). A list of non-3DP nodes denoted by $\mathcal{F}$ is transmitted over the network. $\mathcal{F}_i$ is initialized as $\{v_i\}$ for a non-3DP nodes v_i; $\emptyset$ for 3DP nodes (Line 2).

At the tth iteration of the pruning stage, a node v_i performs as follows.

• **Receive Non-3DP List.** v_i receives $\mathcal{F}_j$ from any $v_j \in \mathcal{N}_i^{\mathcal{A}}$. The gathered lists are represented as $\mathcal{F}_{\mathcal{N}_i^{\mathcal{A}}}$.

• **Recheck 3DP.** If $\iota_i = 0$, v_i unites $\mathcal{F}_i$ and $\mathcal{F}_{\mathcal{N}_i^{\mathcal{A}}}$ as the new $\mathcal{F}_i$ (Line 7); If $\iota_i = 1$, it recalculates its barycentric neighbors using only neighbors whose $\iota = 1$ (Line 8). The non-barycentric neighbors are denoted by $\mathcal{N}_i^{excl}$ (Line 9). Then, v_i removes the paths containing any node of $\mathcal{F}_{i \cup \mathcal{N}_i^{\mathcal{A}}}$ or $\mathcal{N}_i^{excl}$ from $\mathcal{LP}_i^{local}$ (Line 10). If there still exists 3DP in $\mathcal{LP}_i^{local}$, v_i updates $\mathcal{F}_i$ as $\mathcal{F}_{i \cup \mathcal{N}_i^{\mathcal{A}}}$ (Line 12-13), otherwise, v_i updates ι_i to 0 and adds itself to $\mathcal{F}_i$ (Line 14-15).

• **Send Non-3DP List.** v_i sends $\mathcal{F}_i$ to neighbors if it is different from the previous iteration (Line 16-17).

Finally, the paths passing through non-3DP nodes are pruned by exchanging $\mathcal{F}$ with neighbors so that the nodes satisfying Recursive-3DP are detected.

3) Properties of PEP: PEP is proved to have some desired properties.

• **Guaranteed BLL localizability detection performance.**

A key advantage of PEP is that it has theoretically guaranteed BLL localizable node detection performance as in Theorem 4. The proof is given in Appendix A of the Supplementary Material.

Theorem 4: Given $\mathcal{G}$, the PEP algorithm can detect all BLL localizable nodes satisfying Recursive-3DP. That is a node v_i satisfies Recursive-3DP if and only if PEP terminates with $\iota_i = 1$.

Algorithm 3 Path Extension and Pruning (PEP)

```
Input: Neighbor Set: $\mathcal{N}_i$;
Output: Indicator of BLL-localizability: $\iota_i$;
/* Path Extension */
1  $\iota_i(0)$, $\mathcal{LP}_i^{local}(0) \leftarrow$ run PE;
2  $\mathcal{F}_i(0) \leftarrow \{v_i\}$ if $\iota_i = 0$, $\mathcal{F}_i(0) \leftarrow \emptyset$ if $\iota_i = 1$;
/* Path Pruning at Iteration $t$ */
3  receive $\mathcal{F}_j(t)$ from $v_j \in \mathcal{N}_i^{\mathcal{A}}$ and record as $\mathcal{F}_{\mathcal{N}_i^{\mathcal{A}}}(t)$;
4  if $\mathcal{F}_{\mathcal{N}_i^{\mathcal{A}}}(t) = \emptyset$ then
5      continue;//no new non-3DP node received
6  if $\iota_i(t) = 0$ then
7      $\iota_i(t+1) \leftarrow \iota_i(t)$; $\mathcal{F}_i(t+1) \leftarrow \mathcal{F}_{i \cup \mathcal{N}_i^{\mathcal{A}}}(t)$;
   else
8      $\mathcal{N}_i^{\mathcal{A}}(t+1) \leftarrow cal\_bary\_nei(\forall v_j \in \mathcal{N}_i \wedge \iota_j(t) = 1)$;
9      $\mathcal{N}_i^{excl} \leftarrow$ nodes in $\mathcal{N}_i$ but not in $\mathcal{N}_i^{\mathcal{A}}(t+1)$;
10     $\mathcal{LP}_i^{local}(t+1) \leftarrow$ remove the paths containing nodes
        in $\mathcal{F}_{i \cup \mathcal{N}_i^{\mathcal{A}}}(t)$ or $\mathcal{N}_i^{excl}$ from $\mathcal{LP}_i^{local}(t)$;
11     if $detect\_3DP(\mathcal{LP}_i^{local}(t+1)) = true$ then
12         $\iota_i(t+1) \leftarrow \iota_i(t)$;
13         $\mathcal{F}_i(t+1) \leftarrow \mathcal{F}_{i \cup \mathcal{N}_i^{\mathcal{A}}}(t)$;
       else
14         $\iota_i(t+1) \leftarrow 0$; // no longer has 3DP
15         $\mathcal{F}_i(t+1) \leftarrow \mathcal{F}_{i \cup \mathcal{N}_i^{\mathcal{A}}}(t) \cup \{v_i\}$;
// Transmit Message
16 if $\mathcal{F}_i(t+1) \neq \emptyset$ and $\mathcal{F}_i(t+1) \neq \mathcal{F}_i(t)$ then
17     send $\mathcal{F}_i(t+1)$ to $\mathcal{N}_i^{\mathcal{A}}(t+1)$;
```

Integrating with NEI, PEP can detect the BLL localizable nodes satisfying the new sufficient condition in Theorem 1.

• **Convergence speed.** Let H be the maximum number of hops of the node disjoint paths in $\mathcal{G}$. Note that $H \leq |\mathcal{V}|$.

Theorem 5: The PEP algorithm discovers BLL localizable nodes within at most $2H$ communication rounds.

Proof: For any node, let $\mathbf{H} = [H_1, H_2, H_3]$ be its minimum hop counts to three anchors. In the extension stage, the node can receive the path information of anchor $\mathcal{A}_k$ at most H_k steps, $k \in \{1, 2, 3\}$. Thus, it collects its path information within $\max(\mathbf{H})$ rounds. Similarly, in the pruning stage, at most $\max(\mathbf{H})$ rounds are needed to collect the non-3DP node list. Since $\max(\mathbf{H}) \leq H$, PEP discovers BLL-localizable nodes within at most $2H$ rounds. □

• **Free of launch condition.** In PEP, new localizable nodes are found by checking their path list to anchors, which is independent of the spatial distribution of anchors. Thus, PEP does not require any special launch condition.

• **Time complexity.** In the extension stage, calculation of barycentric neighbors checks each combination of three neighbors, which has complexity $O(\Delta^3)$; Detection of 3DP selects any three neighbors and checks whether their paths share common vertices. Suppose the number of paths of each node is bounded by κ. Since a node can find at most one disjoint path through one neighbor, thus the complexity is $O(\Delta^3 \kappa^3)$; In the pruning stage, non-3DP nodes only need to receive and update the non-3DP node list, which has complexity $O(\Delta)$.

Meanwhile, 3DP nodes need to re-evaluate their 3DP, which has complexity $O(\Delta^3\kappa^3)$. Overall, the time complexity of PEP is $O(\Delta^3\kappa^3)$.

Since Δ is usually small especially in sparse networks, the complexity is mainly determined by κ. 1) when anchors are closely deployed, agents near the anchors quickly collect 3DP to anchors. Such an agent v_i replaces $\mathcal{LP}_i$ with a single node $\{v_i\}$ (κ reduces to 1) and serves as new anchors to enable their nearby agents to collect 3DP and reduce κ. Thus, κ is small in such setting; 2) when anchors are far apart or the network is sparse, an agent needs to expand its $\mathcal{LP}$ constantly until 3DP is found so that κ can be large as the path info propagates. Thus, the complexity of PEP depends on whether an agent can effectively reduce its $\mathcal{LP}$.

• **Communication cost.** In PEP, each v_i exchanges the $\mathcal{LP}_i$ and $\mathcal{F}_i$ messages. There are two designs in PEP to save the communication cost without affecting correctness. 1) If v_i detects 3DP, it replaces $\mathcal{LP}_i$ simply by $\{v_i\}$ and transmits only $\{v_i\}$. This is because any other node can find at most one DP to anchors through v_i. It is not necessary to record the detailed paths to anchors before v_i; 2) If v_i is not a 3DP node yet, it transmits its path list $\mathcal{LP}_i$ only when its path list changes in this timestamp. $\mathcal{LP}_i$ has at most κ paths and each path contains at most H IDs, where H is the maximum hop number of the network. In the pruning stage, $\mathcal{F}_i$ is transmitted only when it is updated. $\mathcal{F}_i$ is a list of non-3DP nodes, which contains at most n IDs.

A case study of PEP is given in Appendix B of the Supplementary Material, along with the communication cost demonstration.

B. Fast-PEP Algorithm

PEP pursuits the guaranteed BLL localizable node detection performance by paying the cost that each node keeps and negotiates all the paths to anchors with barycentric neighbors. Although path list simplification and event-driven communication are used, the negotiating is still expensive. Thus, Fast-PEP is designed. It keeps only the shortest paths to anchors in each node's path list, greatly reducing the path list negotiating cost while showing attractive properties.

1) Design of Fast-PEP: According to the property analysis, the complexity of PEP is mainly determined by κ, i.e., the magnitude of the path list. To accelerate PEP, an immediate idea is lightening the path list to be stored and transformed. Note that we are concerned about disjoint paths to different anchors, so we consider keeping only one path from a certain node to each anchor. Specifically, we keep the shortest path. If there is more than one shortest path, choose the one with the smaller ID of the second node. Using Algorithm 4 to replace **Function Aggregate Path List** in the extension stage, Fast-PEP is formed. Other procedures are consistent with PEP.

Take Fig. 6 as an example, two anchors and three agents are shown and other network details are omitted to ease the presentation. The lightened $\mathcal{LP}$ in Fast-PEP is shown in Table III. At $t = 1$, v_4 and v_5 only find one path to each anchor, so path lightening is not needed. At $t = 2$, v_4 has two paths to reach anchor v_2, i.e., $v_2 \to v_4$ and $v_2 \to v_5 \to v_4$ (written as $2; 2, 5$ for brevity). Using Algorithm 4, the longer path $2, 5$ is abandoned. Similarly, v_6 has two paths to anchor v_2, i.e., $2, 4; 2, 5$, then only $2, 4$ will be kept; At $t = 3$, the $\mathcal{LP}$ of each node is the same with $t = 2$. The system reaches consensus. For each node, PEP stores $\mathcal{LP}^{merge}$ while Fast-PEP only stores $\mathcal{LSP}$ (the lightened $\mathcal{LP}^{merge}$), thus the path list saved by Fast-PEP is significantly lightened.

Algorithm 4 Aggregate Shortest Path

```
Input: $\mathcal{LP}$: arbitrary list of paths
Output: $\mathcal{LSP}$: the list of shortest paths
1 $\mathcal{LSP} \leftarrow \emptyset$ ;
2 group $\mathcal{LP}$ by starting vertex;
3 add the shortest path in each group to $\mathcal{LSP}$;
4 add $v_i$ to the end of each path in $\mathcal{LSP}$;
5 return $\mathcal{LSP}$;
```

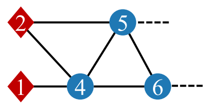


Fig. 6. A schematic diagram of Fast-PEP.

TABLE III
$\mathcal{LP}$ IN FAST-PEP

		node 4	node 5	node 6
t=1	$\mathcal{LP}^{merge}$	1;2	2	$\emptyset$
	$\mathcal{LSP}$	1;2	2	$\emptyset$
t=2	$\mathcal{LP}^{merge}$	1;2;2,5	2;1,4;2,4	1,4;2,4;2,5
	$\mathcal{LSP}$	1;2	1,4;2	1,4;2,4
t=3	$\mathcal{LP}^{merge}$	1;2;2,5	2;1,4;2,4;1,4,6;2,4,6	1,4;2,4;2,5;1,4,5
	$\mathcal{LSP}$	1;2	1,4;2	1,4;2,4

2) Properties of Fast-PEP: An attractive property of Fast-PEP is that although each node uses the lightened $\mathcal{LP}$, in the path expansion phase, Fast-PEP ensures that all 3DP nodes are correctly detected.

Theorem 6: A node v_i has 3 disjoint paths to anchors if and only if Fast-PEP terminates with $\iota_i = 1$ after the extension phase.

The proof of Theorem 6 is given in Appendix C of the Supplementary Material. Then, we show the correctness of Fast-PEP.

Theorem 7: If Fast-PEP terminates with $\iota_i = 1$ after the pruning stage, the node v_i must satisfy the Recursive-3DP condition.

Proof: When rechecking 3DP in the pruning stage, only paths passing through 3DP nodes will be counted. After pruning reaches consensus, all 3DP nodes reach anchors through three disjoint paths containing only 3DP nodes. So they all satisfy the Recursive-3DP condition. □

For complexity analysis, we consider that n^* out of the n agents are BLL localizable. Since only the shortest path to each BLL localizable node is stored, the number of paths of each node is at most n^*. Overall, the complexity of Fast-PEP is $O(\Delta^3 n^{*3})$. As to communication cost, when 3DP can be found, Fast-PEP also only transmits its own ID as PEP. When 3DP cannot be found, the upper bound of path number in $\mathcal{LP}_i$ reduces to n^*.

Fast-PEP uses lightened path list, so it no longer guarantees to detect all Recursive-3DP nodes like PEP. For a 3DP node, it may have multiple combinations of three disjoint paths to anchors. However, Fast-PEP only stores the shortest combination of them. Supposing that a node v_i satisfies Recursive-3DP, Fast-PEP wrongly detects it as unlocalizable if the following conditions hold simultaneously.

1) v_i has at least one combination of three disjoint paths that pass through only 3DP nodes.
2) The 3DP of v_i stored by Fast-PEP pass through at least one non-3DP node.

However, later experiments will show that the two conditions are rarely simultaneously met. Since for any agent, although many paths are not stored and transmitted, the agent can still obtain these paths from neighbors' path list negotiation. For example, in Fig. 6, the path $2, 5$ is abandoned for v_6 at $t = 2$, it can still receive this path from neighbor v_5 at $t = 3$. Even this path is again abandoned due to the existence of path $2, 4$, it can immediately become a 3DP node if a disjoint path to another anchor (except for v_1 and v_2) is received. Thus, the localizable nodes detected by Fast-PEP are almost always the same as PEP. Moreover, Fast-PEP inherits the properties of guaranteed convergence, launch condition free, and limited communication overhead.

C. PEP and Fast-PEP in Realistic Scenarios

In considering the more realistic environments mentioned in Section IV-D, PEP and Fast-PEP are only affected by communication link failure since the distance measurements and neighbor states are not involved in their routine. The reduction of links causes that some BLL localizable nodes may be wrongly detected as unlocalizable. Since NEI can significantly strengthen the neighborhood connectivity, re-conduction of PEP and Fast-PEP is not essential. Experiments will show that the detection performance is still satisfactory compared with existing detection algorithms.

D. Application of NEI and PEP

Section III-A has shown that ignoring node localizability in BLL leads to incorrect convergence of node locations. Since NEI, PEP, and Fast-PEP are all distributed, they can be well integrated with the BLL algorithms. After distributed BLL localizable node detection, each node can select only localizable neighbors to construct its linear equation, so that the impacts of the unlocalizable nodes can be excluded. Then the incorrect convergence problem is avoided. A *Localizability Aware Barycentric linEar Localization framework (LABEL)* is therefore formed, which is detailed in Algorithm 5. Note that PEP can be replaced by Fast-PEP for better efficiency. In addition, the unlocalizable nodes may cause that the iterative system cannot converge. The localizability knowledge help to exclude such effects.

Algorithm 5 LABEL

```
Input: distance measurements to neighbors
Output: $\hat{\mathbf{p}}_i$: localization result
1 run NEI to strengthen neighborhood connectivity;
2 run PEP to obtain BLL node localizability;
3 if $v_i$ is BLL localizable then
4     calculate the barycentric coordinates with localizable
       neighbors;
5     calculate $\hat{\mathbf{p}}_i$ with distributed solvers, e.g., Richardson
       iteration [46] or DCG [72];
  else
6     keep idle; // do not localize BLL unlocalizable nodes
```

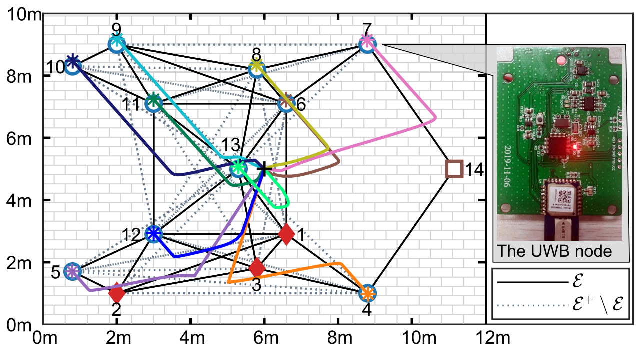


Fig. 7. Testbed in an open area.

VI. Evaluation

Extensive evaluations are conducted to verify the effectiveness of NEI, PEP, Fast-PEP, and LABEL. A testbed validation is firstly given and other parts are large-scale simulations conducted using MATLAB.

A. Testbed Validation

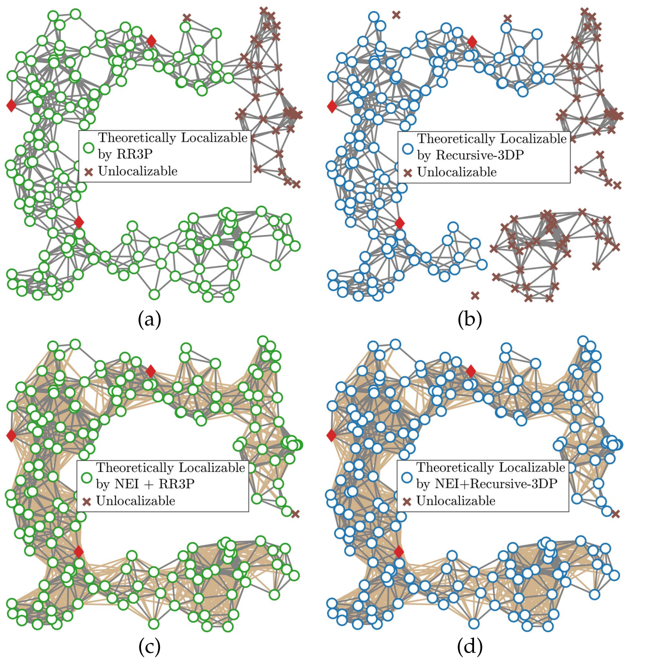
Experiments in a testbed are conducted. Fig. 7 illustrates a network of 14 wireless UWB nodes (3 anchors and 11 agents) in a $12m \times 10m$ open square. The ranging radius R is set to around $5m$ by adjusting the transmission power. The ground truth locations are collected manually with the aid of the bricks ($0.3m \times 0.6m$). The localizable nodes characterized by RR3P and Recursive-3DP are $\{v_6, v_8, v_9, v_{11}, v_{12}, v_{13}\}$ and $\{v_{12}, v_{13}\}$, respectively. After NEI, the detected implicit edges are plotted as dashed lines. The localizable nodes become $\{v_4, v_5, v_6, v_7, v_8, v_9, v_{10}, v_{11}, v_{12}, v_{13}\}$ through both RR3P and Recursive-3DP. Thus NEI helps characterizing truly localizable nodes more correctly.

Then, using PEP, the detected localizable nodes are also $\{v_4, \cdots, v_{13}\}$. In LABEL, only localizable nodes participate in distributed location calculation and the unlocalizable node $\{v_{14}\}$ is excluded. The estimated locations are all initialized from the center of the area, shown as the black cross marker. The converged locations are shown as asterisk markers. The colored lines denote the convergence trails of estimated locations. It can be observed that the estimated locations are close to true locations (hollow circles). Overall, through NEI and PEP, more BLL localizable nodes are found and the BLL localizable nodes are correctly localized.

B. The Effectiveness of NEI

The effectiveness of NEI is evaluated from two aspects, 1) strengthening the neighborhood connectivity, and 2) increasing the number of theoretically localizable nodes in both RRL and BLL. The RR3P nodes are identified by combining network flow and the pebble game algorithm [68]. The Recursive-3DP nodes are identified by the IMF algorithm [57]. They are all centralized algorithms.

1) Visualizing the Effectiveness of NEI: Fig. 8 shows a network with 180 agents and 3 anchors. The RRL localizable nodes satisfying RR3P and BLL localizable nodes satisfying Recursive-3DP are plotted in Fig. 8(a) and Fig. 8(b), the numbers are 147 and 112, respectively. Fig. 8(c) and Fig. 8(d) show that NEI infers lots of implicit edges and the neighborhood



Theoretically Localizable by RR3P
Unlocalizable
(a)
Theoretically Localizable by Recursive-3DP
Unlocalizable
(b)
Theoretically Localizable by NEI + RR3P
Unlocalizable
(c)
Theoretically Localizable by NEI+Recursive-3DP
Unlocalizable
(d)

Fig. 8. Theoretically localizable nodes using different sufficient condition. (a) RR3P nodes in $\mathcal{G}$. (b) Recursive-3DP nodes in $\mathcal{G}^{\mathcal{A}}$. (c) RR3P nodes in $\mathcal{G}^{+}$. (d) Recursive-3DP nodes in $\mathcal{G}^{\mathcal{A}+}$.

connectivity is strengthened for each node. The number of RRL localizable nodes and the number of BLL localizable nodes both increase to 179. It is verified by global rigidity that the detected localizable nodes are truly localizable. So the result shows the remarkable effect of NEI in finding truly localizable nodes for both RRL and BLL.

2) Statistical Results: In statistical results, the neighborhood connectivity is assessed by average node degree $AvgDeg = \frac{1}{|\mathcal{V}|}\sum_{v_i \in \mathcal{V}} |\mathcal{N}_i|$, which can be controlled by varying the maximum ranging radius R. The number of nodes is set to be 103 (100 agents and 3 anchors) and R is varied from $14m$ to $24m$. For each R, 1,000 networks are generated randomly for statistical results. Fig. 9 presents the $AvgDeg$ of $\mathcal{G}$, $\mathcal{G}^{\mathcal{A}}$, $\mathcal{G}^{+}$, and $\mathcal{G}^{\mathcal{A}+}$. The error bar represents the standard deviation. It shows that $AvgDeg$ in $\mathcal{G}^{\mathcal{A}}$ is always smaller than that in $\mathcal{G}$, because only neighbors forming triangles can participate in calculating the barycentric coordinates. By NEI, the $AvgDeg$ in $\mathcal{G}^{+}$ and $\mathcal{G}^{\mathcal{A}+}$ increase significantly. This shows the effectiveness of NEI and the rich implicit edges in nodes' neighborhood.

The proportion of RR3P nodes, Recursive-3DP nodes, RR3P nodes in $\mathcal{G}^{+}$ (written as "NEI+RR3P"), and Recursive-3DP nodes in $\mathcal{G}^{\mathcal{A}+}$ (written as "NEI+Recursive-3DP") are compared in Fig. 10. It shows that: (1) the unlocalizable nodes exist even in highly dense networks when $R = 24$ and after running NEI. This suggests the importance of localizable node detection for running BLL correctly. (2) Without NEI, the number of BLL localizable nodes is always much less than the number of RRL localizable nodes, especially in sparse networks. This suggests the importance of inferring the implicit edges. (3) In all settings, NEI can greatly help to detect more RRL localizable nodes and BLL localizable nodes. (4) The contribution of NEI is more significant in sparse networks since neighbors of many nodes cannot form triangles without the implicit edge information.

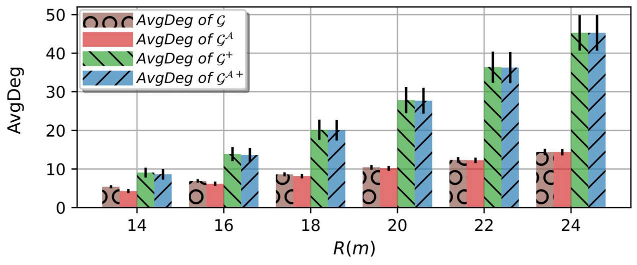


Fig. 9. The $AvgDeg$ in $\mathcal{G}$, $\mathcal{G}^{\mathcal{A}}$, $\mathcal{G}^{+}$, and $\mathcal{G}^{\mathcal{A}+}$.

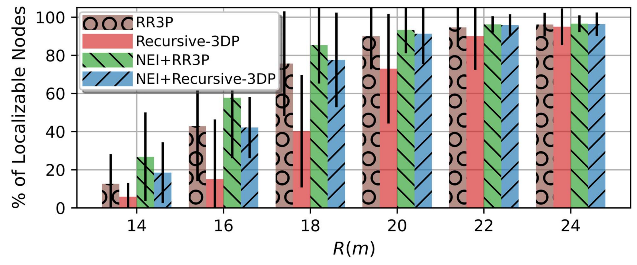


Fig. 10. Proportion of localizable nodes by different sufficient conditions.

TABLE IV
THE NUMBER OF LOCALIZABLE NODES IN FIG. 11, $|\mathcal{S}| = 180$

Launch Condition (LC)	LC of TP	LC of TE	None LC
Recursive-3DP	112	112	112
NEI+Recursive-3DP	179	179	179
TP	84	0	0
TE	104	105	0
PEP	**112**	**112**	**112**
NEI+PEP	**179**	**179**	**179**

C. The Effectiveness of PEP and Fast-PEP

PEP and Fast-PEP are compared with the trilateration protocol (TP) [29] and the triangle extension (TE) [56] algorithms. The number of theoretically BLL localizable nodes satisfying Recursive-3DP in $\mathcal{G}^{\mathcal{A}+}$ is also compared.

1) Visualizing the Effectiveness of PEP: Fig. 11 illustrates the localizable nodes detected by different algorithms using the same network topology as Fig. 8. The specific numbers are shown in Table IV.

• **(Launch Condition (LC) free).** In Fig. 11(a) - Fig. 11(c), "LC of TP" means 3 anchors have common neighbors. "LC of TE" means 2 anchors have common neighbors. "None LC" represents no anchors have common neighbors. TP and TE fail to detect any localizable node if the corresponding LC is not met. By contrast, PEP works stably under any LC.

• **(Much better BLL localizability detection performance).** Fig. 11(a) and Fig. 11(b) show that even when the corresponding LC is met, TP and TE still miss to detect many Recursive-3DP nodes. PEP always detects all the Recursive-3DP nodes as given in Table IV.

• Especially in Fig. 11(d), it can be seen that after integrating with NEI, almost all nodes in the network are successfully detected as localizable by PEP. This shows that NEI+PEP can greatly improve the localizable node detection capability than existing sufficient conditions, i.e., TP and TE. They help BLL successfully discover these localizable nodes to localize them using LABEL.

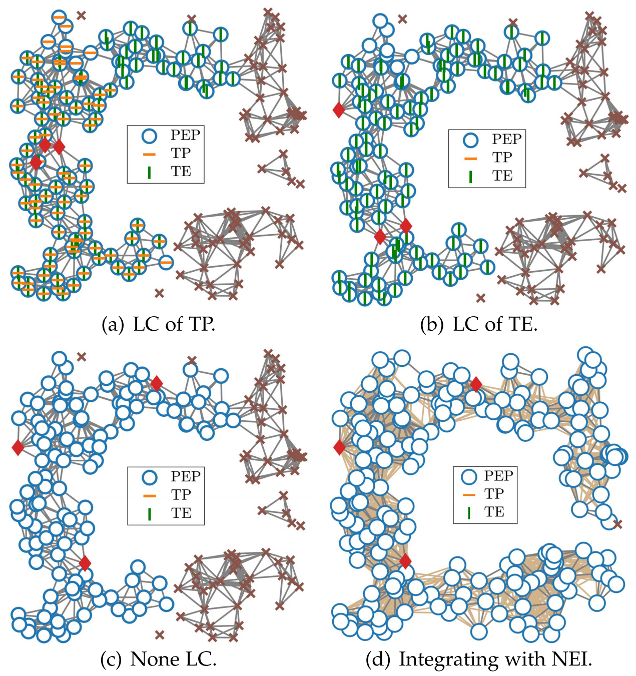


Fig. 11. The detected localizable nodes using different detection algorithms. Edges in (a)-(c) are $\mathcal{E}^{\mathcal{A}}$. Edges in (d) are $\mathcal{E}^{\mathcal{A}+}$.

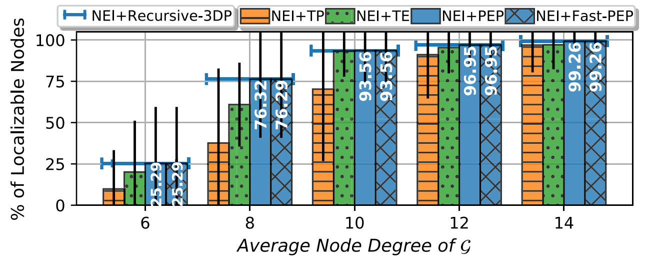


Fig. 12. The localizable nodes detected by different algorithms.

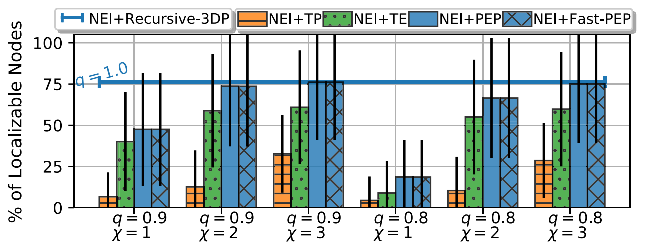


Fig. 13. The statistics of localizable nodes in presence of link failure.

2) Statistical Results of Detection Capacity: Under each $AvgDeg$, 1,000 networks are randomly generated and the statistical results are presented in Fig. 12. The average proportion of Recursive-3DP nodes in $\mathcal{G}^{\mathcal{A}+}$ is plotted with the blue horizontal line. For fairness, all the detection algorithms are integrated with NEI. Results show that TP and TE miss a large portion of nodes satisfying the Recursive-3DP condition. Meanwhile, PEP always detects all of them. Fast-PEP returns the same result as PEP in most networks and misses Recursive-3DP nodes very rarely.

To consider more realistic scenarios, we assume that the communication link between any v_i and a neighbor v_j fails with the probability of $1 - q_{ij}$. In Fig. 13, $AvgDeg$ is 8. The success probability q_{ij} varies in $\{0.9, 0.8\}$ and the number

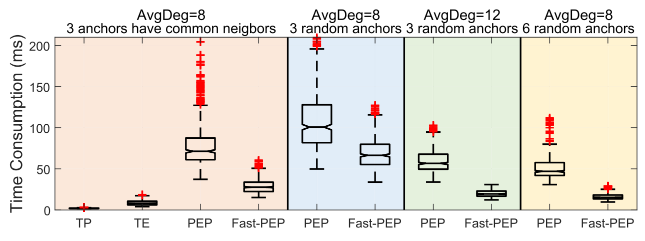


Fig. 14. The algorithm time consumption in different settings.

		Launch Condition	Time Complexity	Space Complexity	Theoretical Ensurance	Distributed
State-of-the-art	TP	3 anchors have common neighbors	Low	Low	×	√
	WE	3 anchors in the same wheel	Low	Low	×	√
	TE	2 anchors have common neighbors	Low	Low	×	√
Proposed	**PEP**	**None**	**High**	**High**	√	√
	Fast-PEP	**None**	**Low**	**Low**	×	√

Fig. 15. A summary of distributed localizable node detection algorithms.

of attempts χ to run NEI varies in $\{1, 2, 3\}$. It is shown that PEP and Fast-PEP are less affected by link failure. Because although some links fail, there are still many paths that can be received. Moreover, the impact of link failure can be alleviated by a very small number of NEI reattempts.

3) Statistical Results of Detection Efficiency: Fig. 14 shows the time consumption of different algorithms under different node degree and anchor distribution settings. Since TP and TE require launch conditions to start, their time consumption is plotted only when 3 anchors have common neighbors. They require much less time than PEP. Fast-PEP runs much more efficiently than PEP. For networks with random anchor distributions, only PEP and Fast-PEP can work. Fast-PEP can save nearly 70% time than PEP in denser networks or when there are more anchors. This is because the agents can quickly collect their 3DP to anchors and serve as new anchors to help other agents finding 3DP.

A summary of these distributed BLL localizable node detection algorithms is given in Fig.15.

- PEP and Fast-PEP are free of launch condition.
- Only PEP can guarantee to detect all BLL localizable nodes satisfying the Recursive-3DP condition.
- Fast-PEP has a very close detection performance as PEP, but it is much more efficient.

D. The Effectiveness of LABEL

The effectiveness of the LABEL framework is evaluated. Comparison algorithms include the state-of-the-art BLL algorithm ECHO [46] and the representative centralized RRL algorithm G2O [14], both of which do not consider node localizability. The evaluation metric uses *the localization error* calculated by $||\hat{\mathbf{p}}_i - \mathbf{p}_i||_2$, where $\hat{\mathbf{p}}_i$ and $\mathbf{p}_i$ are the calculated location and ground truth location of v_i, respectively.

1) Visualizing the Localization Accuracy: Fig. 16 visualizes the localization results of different algorithms in a network of 35 nodes. The ground truth locations of the anchors, localizable nodes, and unlocalizable nodes are plotted as red diamonds, green circles, and brown squares, respectively. The calculated locations are plotted as red asterisk markers. The blue edge connecting the ground truth and the calculated location implies the localization error. The term "localizable"

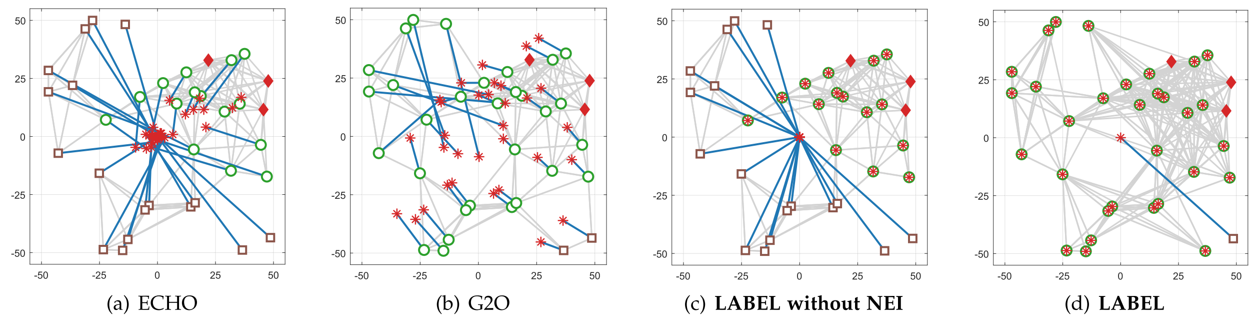


Fig. 16. A visualization of the localization accuracy.

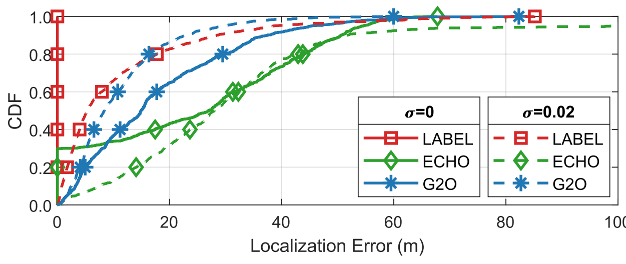


Fig. 17. The cumulative distribution function (CDF) of localization errors.

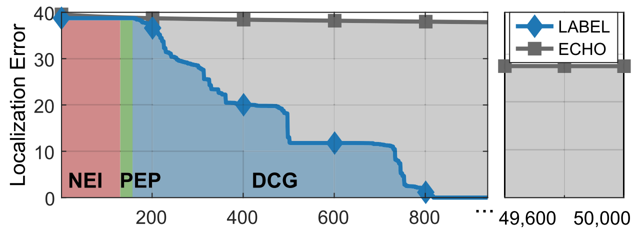


Fig. 18. The evolution of localization errors in different phases.

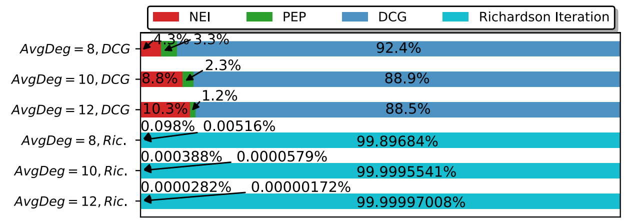


Fig. 19. The percentage of iterations occupied by different phases.

means BLL localizable when ECHO, and LABEL are adopted, and it means RRL localizable when G2O is adopted. Fig. 16 shows that ECHO and G2O indiscriminately localize both localizable and unlocalizable nodes, whose results are not correct when the entire network is not localizable; LABEL only localizes the BLL localizable nodes, but the calculated locations are ensured to be correct and LABEL can tell which of them are localizable. Moreover, LABEL significantly increases the number of localizable nodes through NEI.

2) Statistical Results of LABEL's Accuracy: For statistical results of the accuracy, 1,000 random networks are generated. For fairness, only the localization errors of localizable nodes are counted. The ranging noises are set as the widely-adopted zero-mean Gaussian noises as in [65]. The CDFs of localization errors using different algorithms are shown in Fig. 17. Even when the distances are noiseless, i.e, $\sigma = 0$, only LABEL has zero localization error, because ECHO and G2O are impacted by the unlocalizable nodes. When the distances are noisy, LABEL is still remarkably more accurate than ECHO.

3) On the Overhead of NEI and PEP: Compared with the existing BLL methods, LABEL introduces two additional modules before location iteration, i.e., NEI and PEP. Their overhead in the entire localization process is evaluated. For the network in Fig. 16, the localization error variation progresses in LABEL and in ECHO is compared in Fig. 18. Nodes in ECHO converge to wrong locations after 50,000 rounds. In LABEL, the time consumed by NEI and PEP is equal to 130 rounds and 27 rounds in ECHO, respectively. LABEL uses DCG [72] as the distributed location updating algorithm. Through the NEI, PEP, and DCG routine, LABEL calculates the node locations correctly in 937 rounds. In the LABEL framework, PEP and NEI takes only a very small portion in the whole process.

4) Statistical Results of Overhead: The average percentage of iterations occupied by NEI, PEP, and DCG are shown in Fig. 19. It shows that NEI takes more time in dense networks because a node needs to check more $\mathcal{BFG}s$. For PEP, the most time-consuming scenario is when the network is sparse. The localizable nodes need to wait until 3DP can be found. When edges are dense, PEP terminates rapidly and consumes only 1.2% of the total iterations, because localizable nodes can efficiently collect 3DP information. Moreover, if another distributed solver *Richardson iteration is adopted, the overhead of NEI and PEP is almost negligible.* Overall, NEI and PEP take only a small portion of time in the localization process but provide much better localization accuracy than existing BLL algorithms that ignore node localizability.

VII. Conclusion and Future Work

This paper investigates BLL in realistic and challenging scenarios where networks are not entirely localizable. A key observation is that BLL localizability requires stronger neighborhood connectivity. By inferring implicit edges using NEI, the BLL node localizability can be more accurately characterized. Moreover, PEP and Fast-PEP are proposed for distributed BLL localizable node detection. PEP is free of launch conditions and ensures to find BLL localizable nodes satisfying the Recursive-3DP condition. Fast-PEP can significantly reduce time consumption while showing satisfactory performance. Using NEI and PEP as an initialization procedure, a localizability aware BLL framework, i.e., LABEL is proposed, which guarantees the correct location convergence of localizable nodes. The effectiveness of the proposed algorithms in localizability, accuracy, and efficiency are validated using both real and simulated networks. The understanding of BLL node localizability improves the application scope of BLL.

Many interesting problems arising from this study deserve further research. The BLL localizability condition in 3D can be investigated using the idea of NEI and PEP. The detected node localizability can also give knowledge about the distribution of the network, e.g., the area where a large number of unlocalizable nodes gather indicates the weakness of the network. Such knowledge provides hints on distributively optimizing node deployment or locally determining motion control of agents for better network localization, which will be studied in future work.

REFERENCES

[1] K. Liu and X. Li, "Enhancing localization scalability and accuracy via opportunistic sensing," *IEEE/ACM Trans. Netw.*, vol. 26, no. 3, pp. 1517–1530, Jun. 2018.

[2] L. Chang et al., "FitLoc: Fine-grained and low-cost device-free localization for multiple targets over various areas," *IEEE/ACM Trans. Netw.*, vol. 25, no. 4, pp. 1994–2007, Aug. 2017.

[3] X. Tong, H. Li, X. Tian, and X. Wang, "Wi-Fi localization enabling self-calibration," *IEEE/ACM Trans. Netw.*, vol. 29, no. 2, pp. 904–917, Apr. 2021.

[4] C. Ma, B. Wu, S. Poslad, and D. R. Selviah, "Wi-Fi RTT ranging performance characterization and positioning system design," *IEEE Trans. Mobile Comput.*, vol. 21, no. 2, pp. 740–756, Feb. 2022.

[5] F. Shan, J. Zeng, Z. Li, J. Luo, and W. Wu, "Ultra-wideband swarm ranging," in *Proc. IEEE Conf. Comput. Commun.*, May 2021, pp. 1–10.

[6] F. R. F. Consortium. (2021). *Localization Cases Based on Distance Measurement Using UWB Technology*. [Online]. Available: https://www.firaconsortium.org/discover/use-cases#location

[7] Q. Liu, X. Bai, X. Gan, and S. Yang, "Lora RTT ranging characterization and indoor positioning system," *Wireless Commun. Mobile Comput.*, vol. 2021, pp. 1–10, Mar. 2021.

[8] H. Huang, W. Miao, G. Min, C. Huang, X. Zhang, and C. Wang, "Resilient range-based d-dimensional localization for mobile sensor networks," *IEEE/ACM Trans. Netw.*, vol. 28, no. 5, pp. 2037–2050, Oct. 2020.

[9] Tsingoal. (2021). *Social Distancing and Contact Tracing to Anti COVID-19*. [Online]. Available: https://www.social-distancing-contact-tracing.com/

[10] Wikipedia. (2022). *List of UWB-Enabled Mobile Devices*. [Online]. Available: https://en.wikipedia.org/wiki/List_of_UWB-enabled_mobile_devices

[11] Y. Shang, W. Ruml, Y. Zhang, and M. P. J. Fromherz, "Localization from mere connectivity," in *Proc. 4th ACM Int. Symp. Mobile Ad Hoc Netw. Comput.*, 2003, pp. 201–212.

[12] Y. Shang and W. Ruml, "Improved MDS-based localization," in *Proc. 23rd Annu. Joint Conf. IEEE Comput. Commun. Soc. (INFOCOM)*, vol. 4, Mar. 2004, pp. 2640–2651.

[13] J. Aspnes et al., "A theory of network localization," *IEEE Trans. Mobile Comput.*, vol. 5, no. 12, pp. 1663–1678, Dec. 2006.

[14] R. Kümmerle, G. Grisetti, H. Strasdat, K. Konolige, and W. Burgard, "G2o: A general framework for graph optimization," in *Proc. IEEE Int. Conf. Robot. Autom.*, May 2011, pp. 3607–3613.

[15] D. Shamsi, N. Taheri, Z. Zhu, and Y. Ye, "Conditions for correct sensor network localization using SDP relaxation," in *Discrete Geometry and Optimization*. Cham, Switzerland: Springer, 2013, pp. 279–301.

[16] R. L. Moses, D. Krishnamurthy, and R. M. Patterson, "A self-localization method for wireless sensor networks," *EURASIP J. Adv. Signal Process.*, vol. 2003, no. 4, pp. 1–11, 2003.

[17] G. Destino and G. Abreu, "On the maximum likelihood approach for source and network localization," *IEEE Trans. Signal Process.*, vol. 59, no. 10, pp. 4954–4970, Oct. 2011.

[18] S. Zhao, X.-P. Zhang, X. Cui, and M. Lu, "A new TOA localization and synchronization system with virtually synchronized periodic asymmetric ranging network," *IEEE Internet Things J.*, vol. 8, no. 11, pp. 9030–9044, Jun. 2021.

[19] V. Ramadurai and M. L. Sichitiu, "Localization in wireless sensor networks: A probabilistic approach," in *Proc. Int. Conf. Wireless Netw.*, 2003, pp. 275–281.

[20] R. Peng and M. L. Sichitiu, "Probabilistic localization for outdoor wireless sensor networks," *ACM SIGMOBILE Mobile Comput. Commun. Rev.*, vol. 11, no. 1, pp. 53–64, 2007.

[21] X. Wang, Z. Yang, J. Luo, and C. Shen, "Beyond rigidity: Obtain localisability with noisy ranging measurement," *Int. J. Ad Hoc Ubiquitous Comput.*, vol. 8, nos. 1–2, p. 114, 2011.

[22] W. Liu, E. Dong, and Y. Song, "Analysis of flip ambiguity for robust three-dimensional node localization in wireless sensor networks," *J. Parallel Distrib. Comput.*, vol. 97, pp. 57–68, Nov. 2016.

[23] Q. Guo, Y. Zhang, J. Lloret, B. Kantarci, and W. K. G. Seah, "A localization method avoiding flip ambiguities for micro-UAVs with bounded distance measurement errors," *IEEE Trans. Mobile Comput.*, vol. 18, no. 8, pp. 1718–1730, Aug. 2019.

[24] X. Wang, Y. Liu, Z. Yang, J. Liu, and J. Luo, "ETOC: Obtaining robustness in component-based localization," in *Proc. 18th IEEE Int. Conf. Netw. Protocols*, Oct. 2010, pp. 62–71.

[25] X. Wang, J. Luo, Y. Liu, S. Li, and D. Dong, "Component-based localization in sparse wireless networks," *IEEE/ACM Trans. Netw.*, vol. 19, no. 2, pp. 540–548, Apr. 2011.

[26] L. Zhang, L. Liu, C. Gotsman, and S. J. Gortler, "An as-rigid-as-possible approach to sensor network localization," *ACM Trans. Sensor Netw.*, vol. 6, no. 4, pp. 1–21, Jul. 2010.

[27] T. Sun, Y. Wang, D. Li, Z. Gu, and J. Xu, "WCS: Weighted component stitching for sparse network localization," *IEEE/ACM Trans. Netw.*, vol. 26, no. 5, pp. 2242–2253, Oct. 2018.

[28] Y. Wang, T. Sun, G. Rao, and D. Li, "Formation tracking in sparse airborne networks," *IEEE J. Sel. Areas Commun.*, vol. 36, no. 9, pp. 2000–2014, Sep. 2018.

[29] T. Eren et al., "Rigidity, computation, and randomization in network localization," in *Proc. IEEE INFOCOM*, vol. 4, Mar. 2004, pp. 2673–2684.

[30] D. Moore, J. Leonard, D. Rus, and S. Teller, "Robust distributed network localization with noisy range measurements," in *Proc. 2nd Int. Conf. Embedded Networked Sensor Syst.*, 2004, pp. 50–61.

[31] H. Akcan and C. Evrendilek, "Reducing the number of flips in trilateration with noisy range measurements," in *Proc. 12th Int. ACM Workshop Data Eng. Wireless Mobile Acess*, Jun. 2013, pp. 20–27.

[32] A. Savvides, C.-C. Han, and M. B. Strivastava, "Dynamic fine-grained localization in ad-hoc networks of sensors," in *Proc. 7th Annu. Int. Conf. Mobile Comput. Netw.*, 2001, pp. 166–179.

[33] Y. Zhou, J. Li, and L. Lamont, "Multilateration localization in the presence of anchor location uncertainties," in *Proc. IEEE Global Commun. Conf. (GLOBECOM)*, Dec. 2012, pp. 309–314.

[34] J. Du, C. Yuan, M. Yue, and T. Ma, "A novel localization algorithm based on RSSI and multilateration for indoor environments," *Electronics*, vol. 11, no. 2, p. 289, Jan. 2022.

[35] D. K. Goldenberg et al., "Localization in sparse networks using sweeps," in *Proc. 12th Annu. Int. Conf. Mobile Comput. Netw.*, 2006, pp. 110–121.

[36] G. Oliva, S. Panzieri, F. Pascucci, and R. Setola, "Sensor networks localization: Extending trilateration via shadow edges," *IEEE Trans. Autom. Control*, vol. 60, no. 10, pp. 2752–2755, Oct. 2015.

[37] C. Savarese, J. M. Rabaey, and J. Beutel, "Location in distributed ad-hoc wireless sensor networks," in *Proc. IEEE Int. Conf. Acoust., Speech, Signal Process. (ICASSP)*, vol. 4, May 2001, pp. 2037–2040.

[38] C. Savarese, J. M. Rabaey, and J. Beutel, "Locationing in distributed ad-hoc wireless sensor network," in *Proc. IEEE Int. Conf. Acoustics. Speech Signal*. New York, NY, USA: IEEE, May 2001, pp. 2037–2040.

[39] F. Yin, C. Fritsche, D. Jin, F. Gustafsson, and A. M. Zoubir, "Cooperative localization in WSNs using Gaussian mixture modeling: Distributed ECM algorithms," *IEEE Trans. Signal Process.*, vol. 63, no. 6, pp. 1448–1463, Mar. 2015.

[40] T. V. Nguyen, Y. Jeong, H. Shin, and M. Z. Win, "Least square cooperative localization," *IEEE Trans. Veh. Technol.*, vol. 64, no. 4, pp. 1318–1330, Apr. 2015.

[41] J. Xiong, Z. Xiong, Y. Ding, J. W. Cheong, and A. G. Dempster, "Multihypothesis Gaussian belief propagation for radio ranging-based localization and mapping," *IEEE Trans. Instrum. Meas.*, vol. 71, pp. 1–13, 2022.

[42] U. A. Khan, S. Kar, and J. M. F. Moura, "Distributed sensor localization in random environments using minimal number of anchor nodes," *IEEE Trans. Signal Process.*, vol. 57, no. 5, pp. 2000–2016, May 2009.

[43] U. A. Khan, S. Kar, B. Sinopoli, and J. M. F. Moura, "Distributed sensor localization in Euclidean spaces: Dynamic environments," in *Proc. 46th Annu. Allerton Conf. Commun., Control, Comput.*, Sep. 2008, pp. 361–366.

[44] U. A. Khan, S. Kar, and J. M. F. Moura, "DILAND: An algorithm for distributed sensor localization with noisy distance measurements," *IEEE Trans. Signal Process.*, vol. 58, no. 3, pp. 1940–1947, Mar. 2010.

[45] Y. Diao, Z. Lin, M. Fu, and H. Zhang, "A new distributed localization method for sensor networks," in *Proc. 9th Asian Control Conf. (ASCC)*, Jun. 2013, pp. 1–6.

[46] Y. Diao, Z. Lin, and M. Fu, "A barycentric coordinate based distributed localization algorithm for sensor networks," *IEEE Trans. Signal Process.*, vol. 62, no. 18, pp. 4760–4771, Sep. 2014.

[47] U. A. Khan, S. Kar, and J. M. F. Moura, "Linear theory for self-localization: Convexity, barycentric coordinates, and Cayley–Menger determinants," *IEEE Access*, vol. 3, pp. 1326–1339, 2015.

[48] C. Hou, Y. Hou, and Z. Huang, "A framework based on barycentric coordinates for localization in wireless sensor networks," *Comput. Netw.*, vol. 57, no. 17, pp. 3701–3712, Dec. 2013.

[49] T. Han, Z. Lin, R. Zheng, Z. Han, and H. Zhang, "A barycentric coordinate based approach to three-dimensional distributed localization for wireless sensor networks," in *Proc. 13th IEEE Int. Conf. Control Autom. (ICCA)*, Jul. 2017, pp. 600–605.

[50] P. P. V. Tecchio, "Range-only node localization: The arbitrary anchor case in d-dimensions," Ph.D. dissertation, Dept. Elect. Syst. Eng., Univ. Pennsylvania, Philadelphia, PA, USA, 2019.

[51] Z. Yang and Y. Liu, "Understanding node localizability of wireless ad hoc and sensor networks," *IEEE Trans. Mobile Comput.*, vol. 11, no. 8, pp. 1249–1260, Aug. 2012.

[52] Z. Guo et al., "Perpendicular intersection: Locating wireless sensors with mobile beacon," *IEEE Trans. Veh. Technol.*, vol. 59, no. 7, pp. 3501–3509, Sep. 2010.

[53] L. Mo et al., "Canopy closure estimates with GreenOrbs: Sustainable sensing in the forest," in *Proc. 7th ACM Conf. Embedded Netw. Sensor Syst.*, 2009, pp. 99–112.

[54] B. Jackson and T. Jordán, "Connected rigidity matroids and unique realizations of graphs," *J. Combinat. Theory, B*, vol. 94, no. 1, pp. 1–29, 2005.

[55] Z. Yang, Y. Liu, and X.-Y. Li, "Beyond trilateration: On the localizability of wireless ad-hoc networks," in *Proc. IEEE INFOCOM*, Apr. 2009, pp. 2392–2400.

[56] H. Wu, A. Ding, W. Liu, L. Li, and Z. Yang, "Triangle extension: Efficient localizability detection in wireless sensor networks," *IEEE Trans. Wireless Commun.*, vol. 16, no. 11, pp. 7419–7431, Sep. 2017.

[57] H. Ping, Y. Wang, and D. Li, "IMF: Iterative max-flow for node localizability detection in barycentric linear localization," 2021, *arXiv:2102.07100*.

[58] D. Niculescu and B. Nath, "DV based positioning in ad hoc networks," *Telecommun. Syst.*, vol. 22, no. 1, pp. 267–280, Jan. 2003.

[59] M. Li and Y. Liu, "Rendered path: Range-free localization in anisotropic sensor networks with holes," *IEEE/ACM Trans. Netw.*, vol. 18, no. 1, pp. 320–332, Feb. 2010.

[60] Y. Wang, S. Lederer, and J. Gao, "Connectivity-based sensor network localization with incremental Delaunay refinement method," in *Proc. IEEE INFOCOM*, Apr. 2009, pp. 2401–2409.

[61] G. Tan, H. Jiang, S. Zhang, Z. Yin, and A.-M. Kermarrec, "Connectivity-based and anchor-free localization in large-scale 2D/3D sensor networks," *ACM Trans. Sensor Netw.*, vol. 10, no. 1, pp. 1–21, Nov. 2013.

[62] T. J. S. Chowdhury, C. Elkin, V. Devabhaktuni, D. B. Rawat, and J. Oluoch, "Advances on localization techniques for wireless sensor networks: A survey," *Comput. Netw.*, vol. 110, pp. 284–305, Dec. 2016.

[63] R. C. Shit, S. Sharma, D. Puthal, and A. Y. Zomaya, "Location of things (LoT): A review and taxonomy of sensors localization in IoT infrastructure," *IEEE Commun. Surveys Tuts.*, vol. 20, no. 3, pp. 2028–2061, 3rd Quart., 2018.

[64] S. Safavi, U. A. Khan, S. Kar, and J. M. F. Moura, "Distributed localization: A linear theory," *Proc. IEEE*, vol. 106, no. 7, pp. 1204–1223, Jul. 2018.

[65] H. Ping, Y. Wang, D. Li, and T. Sun, "Flipping free conditions and their application in sparse network localization," *IEEE Trans. Mobile Comput.*, vol. 21, no. 3, pp. 986–1003, Mar. 2022.

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

[67] D. K. Goldenberg et al., "Network localization in partially localizable networks," in *Proc. IEEE 24th Annu. Joint Conf. IEEE Comput. Commun. Societies.*, vol. 1, Mar. 2005, pp. 313–326.

[68] D. J. Jacobs and B. Hendrickson, "An algorithm for two-dimensional rigidity percolation: The pebble game," *J. Comput. Phys.*, vol. 137, no. 2, pp. 346–365, 1997.

[69] A. F. Mobius, *Der Barycentrische Calcul*. Leipzig, Germany: Georg Olms Verlag, 1827.

[70] P. P. V. Tecchio, N. Atanasov, S. Shahrampour, and G. J. Pappas, "N-dimensional distributed network localization with noisy range measurements and arbitrary anchor placement," in *Proc. Amer. Control Conf. (ACC)*, Jul. 2019, pp. 1195–1201.

[71] Z. Lin, Z. Han, and M. Cao, "Distributed localization for multi-robot systems in presence of unlocalizable robots," in *Proc. 59th IEEE Conf. Decis. Control (CDC)*, Dec. 2020, pp. 1550–1555.

[72] H. Ping, Y. Wang, and D. Li, "DCG: Distributed conjugate gradient for efficient linear equations solving," 2021, *arXiv:2107.13814*.

[73] U. A. Khan, S. Kar, and J. M. F. Moura, "Distributed localization and tracking with coordinated and uncoordinated motion models," in *Proc. 47th Annu. Allerton Conf. Commun., Control, Comput. (Allerton)*, 2009, pp. 202–208.

[74] F. Thomas and L. Ros, "Revisiting trilateration for robot localization," *IEEE Trans. Robot.*, vol. 21, no. 1, pp. 93–101, Feb. 2005.

[75] J. Alonso-Mora, E. Montijano, T. Nägeli, O. Hilliges, M. Schwager, and D. Rus, "Distributed multi-robot formation control in dynamic environments," *Auto. Robots*, vol. 43, no. 5, pp. 1079–1100, 2019.

[76] Z. Lin, M. Fu, and Y. Diao, "Distributed self localization for relative position sensing networks in 2D space," *IEEE Trans. Signal Process.*, vol. 63, no. 14, pp. 3751–3761, Jul. 2015.

Haodi Ping (Graduate Student Member, IEEE) received the B.S. and M.S. degrees from the Department of Computer Science and Technology, China University of Geosciences, Beijing, in 2015 and 2018, respectively. 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, graph optimization, and applications.

Yongcai Wang (Member, IEEE) received the B.S. and Ph.D. degrees from the Department of Automation Sciences and Engineering, Tsinghua University, in 2001 and 2006, respectively. From 2007 to 2009, he worked as an Associate Researcher at NEC Laboratories China. From 2009 to 2015, he was a Research Scientist at the Institute for Interdisciplinary Information Sciences (IIIS), Tsinghua University. In 2015, he was a Visiting Scholar at Cornell University. He is currently an Associate Professor with the Department of Computer Sciences, Renmin University of China. His research interests include network localization and combinatorial optimization and applications.

Deying Li (Member, IEEE) 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.

Wenping Chen received the B.E. and M.E. degrees from the Department of Computer Science and Technology, Xi'an Jiaotong University, in 1997 and 2000, respectively, and the Ph.D. degree from the Department of Computer Science and Technology, Tsinghua University, in 2003. She is currently working as a Lecturer with the Department of Computer Science, Renmin University of China. Her current research interests include mobile computing, wireless ad hoc networks, and social computing.