ARL-TR-8909 • Feb 2020

DEVCOM
ARMY RESEARCH
LABORATORY

Rigid Neighborhood Discovery and Decentralized Localization for Multi-Agent Mobile Networks

by Moshe Hamaoui

NOTICES

Disclaimers

The findings in this report are not to be construed as an official Department of the Army position unless so designated by other authorized documents.

Citation of manufacturer's or trade names does not constitute an official endorsement or approval of the use thereof.

Destroy this report when it is no longer needed. Do not return it to the originator.

ARL-TR-8909 • FEB 2020

Rigid Neighborhood Discovery and Decentralized Localization for Multi-Agent Mobile Networks

by Moshe Hamaoui
Weapons and Materials Research Directorate, CCDC Army Research Laboratory

REPORT DOCUMENTATION PAGE	*Form Approved* *OMB No. 0704-0188*

1. REPORT DATE *(DD-MM-YYYY)*
February 2020

2. REPORT TYPE
Technical Report

3. DATES COVERED (From - To)
1 March 2019–16 September 2019

4. TITLE AND SUBTITLE
Rigid Neighborhood Discovery and Decentralized Localization for Multi-Agent Mobile Networks

5a. CONTRACT NUMBER

5b. GRANT NUMBER

5c. PROGRAM ELEMENT NUMBER

6. AUTHOR(S)
Moshe Hamaoui

5d. PROJECT NUMBER

5e. TASK NUMBER

5f. WORK UNIT NUMBER

7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES)
CCDC Army Research Laboratory
ATTN: FCDD-RLW-LF
Aberdeen Proving Ground, MD 21005-5066

8. PERFORMING ORGANIZATION REPORT NUMBER
ARL-TR-8909

9. SPONSORING/MONITORING AGENCY NAME(S) AND ADDRESS(ES)

10. SPONSOR/MONITOR'S ACRONYM(S)

11. SPONSOR/MONITOR'S REPORT NUMBER(S)

12. DISTRIBUTION/AVAILABILITY STATEMENT
Approved for public release; distribution is unlimited.

13. SUPPLEMENTARY NOTES
ORCID: https://orcid.org/0000-0002-4851-4823
primary author's email: <moshe.hamaoui.civ@mail.mil>.

14. ABSTRACT

Location awareness is crucial for many mobile-network applications. While commercial applications rely heavily on the convenience and ubiquity of GPS, military applications must remain robust across the spectrum of denied and contested battlespaces. The use of interagent RF ranging measurements provides one means of reconstructing the relative network geometry. If all pairwise range measurements are always available to all agents, each agent can then separately solve for the network geometry. For dynamic mobile networks with constrained communications, and particularly for extended networks with many agents, the available range measurements may not uniquely specify the entire network geometry. Instead, each agent must discover and localize a solvable subset of the network. This report presents a decentralized method of rigid-neighborhood discovery and localization. The method is implemented in simulation under conditions of range-limited measurement and communication. Results suggest that rigid-neighborhood selection can improve relative localization compared to full-network or random-neighborhood selection.

15. SUBJECT TERMS
multi-agent localization, rigid graphs, graph theory

16. SECURITY CLASSIFICATION OF:			17. LIMITATION OF ABSTRACT	18. NUMBER OF PAGES	19a. NAME OF RESPONSIBLE PERSON
					Moshe Hamaoui
a. REPORT	b. ABSTRACT	c. THIS PAGE			19b. TELEPHONE NUMBER (Include area code)
Unclassified	Unclassified	Unclassified	UU	28	410-306-0968

Contents

List of Figures

List of Tables

1. Introduction/Background

Location awareness is crucial for many mobile-network applications. While commercial applications rely heavily on the convenience and ubiquity of GPS, military applications must remain robust across the spectrum of denied and contested battlespaces. The use of interagent RF ranging measurements provides one means of reconstructing the unanchored (relative) network geometry. If all pairwise range measurements are always available to all agents, each agent can then separately solve for the network geometry. For dynamic mobile networks with constrained communications, and particularly for extended networks with many agents, the available range measurements may not uniquely specify the entire network geometry. Instead, each agent must discover and localize a solvable subset of the network.

As range measurements and communications are often range-limited, agents will typically have more information about nearby agents than those farther away. An agent's solvable subset therefore typically comprises its neighbors. For many applications, this neighborhood solution will suffice to facilitate collaborative behaviors, including formation control, cooperative tracking, and weapon–target assignment.

In general, the various unanchored neighborhood solutions are not expressed in a common reference frame. However, with sufficient membership overlap between two sets, the frame transformation can be estimated and the neighborhoods can be merged. With sufficient overlap and careful consensus protocols, this procedure can continue hierarchically until all agents agree on a consistent coordinate assignment for all members. This global solution is all-inclusive but remains unanchored since the agreed-upon reference frame has no known relation to the "absolute" world frame.

For some applications, absolute positioning may be available to a (possibly changing) subset of so-called "anchor" agents. The anchors then define an absolute frame, allowing the relative unanchored solutions to be transformed to the common absolute frame. The anchored solution allows agents to interact not only with each other, but also with the environment.

1.1 Problem Statement

In a decentralized multi-agent architecture, inter-agent distances are observed, communicated, and received in an ongoing, bidirectional flow of information across the network. Agents maintain and update a record of most recently received measurements. It is from this local record of asynchronous measurements that an agent develops a view of the network beyond its immediately adjacent neighbors. At any instant t, then, we may imagine two edge-weighted graphs that characterize the network. The first is the interaction graph $G(t)$ that describes which direct peer-to-peer measurements and communications may occur at time t. The vertices $V = v_1, v_2, ..., v_n$ correspond to all n member agents of the network, and the weights on edges $E(t)$ encode the corresponding inter-agent range measurements. The second graph $G_i(t)$ is the ith agent's best estimate of the edge-weighted network. This is usually an estimate (rather than a subset of $G(t)$) because the aggregated measurements are generally asynchronous—requiring some extrapolation to synchronize measurements. By the same token, it is also possible for edges to exist in $G_i(t)$ that may not be present in $G(t)$.

The decentralized *full* localization problem asks the ith agent to estimate the positions of all n members, based on $G_i(t)$ and, more particularly, based on the associated weighted adjacency matrix $\mathbf{W}_i \in \mathbb{R}^{n \times n}$. As is discussed in Section 1, the solution is uniquely specified if and only if $G_i(t)$ is globally rigid. Furthermore, even if an agent has instantaneous access to all direct measurements encoded in W (the weighted adjacency matrix associated with $G(t)$), there is certainly no guarantee that $G(t)$ is itself globally rigid—or even connected.

In contrast, the decentralized *neighborhood* localization problem demands only that the ith agent estimate the positions of some vertex subset $V_i^\kappa \subseteq V_i$. For notational consistency, we likewise designate the corresponding subgraph with the κ superscript, G_i^κ. The motivation for this relaxed problem formulation comes from the observation that graphs can often be partitioned into more rigid components. One trivial but very practical example is the disconnected graph whose components are entirely uncoupled from one another—but may themselves exhibit some degree of rigidity. In any event, while the solution space is expanded, it is not at all obvious how to choose G_i^κ. In other words, how do we methodically identify an optimally rigid subgraph?

1.2 Global Rigidity

Let $\mathbf{X} \in \mathbb{R}^{n\times\eta}$ be the stacked coordinates of n points in $\mathbb{R}^{\eta}$, and let $G(\mathbf{X})$ be a *bar framework* on $\mathbf{X}$, where each point $\mathbf{x}_i \in \mathbb{R}^{\eta}$ corresponds to a vertex in G and the edges of G correspond to fixed-length bars joining adjacent vertices. $G(\mathbf{X})$ is then *globally* rigid in $\mathbb{R}^{\eta}$, if for every framework $G(\mathbf{Y}), \mathbf{Y} \in \mathbb{R}^{n\times\eta}$; $\mathbf{Y}$ is identical to $\mathbf{X}$ up to translation, rotation, and reflection. Similarly, a graph G is said to be globally rigid in $\mathbb{R}^{\eta}$, if for any realizable set of edge lengths, there is only one possible realization (again, up to congruence).

Global rigidity is closely related to the graph theoretic notion of vertex connectivity. G is vertex-k connected if a minimum of k vertices must be deleted in order to disconnect G. Hendrickson[1] showed that global rigidity in $\mathbb{R}^{\eta}$ requires that G must be vertex $\eta + 1$ connected, as well as redundantly rigid. The condition of redundant rigidity further demands that G must remain rigid (flex free) with the removal of any one edge. These necessary conditions are also sufficient for dimension $\eta = 2$,[2] though not for $\eta \geq 3$.[3] No combinatorial characterization of global rigidity for $\eta \geq 3$ has been found,[4] but an algebraic characterization based on the associated stress matrix has been shown to be sufficient[5] and necessary.[6]

Based on the rigidity and stress matrices, graphs can be readily classified as flexible, rigid, globally rigid, or complete. This classification also characterizes the relative degeneracy, of the solution space. Flexible configurations admit an infinite number of solutions, and locally rigid configurations allow only a finite solution set.

2. Methods

2.1 k-vertex Connected Components

We return now to the question of how to methodically identify solvable subgraphs of $G_i(t)$. In light of the previous discussion on rigidity, we may state more precisely that we seek globally rigid substructures. Interestingly, simulations show that the vertex connectivity κ generally correlates with rigidity class, with a higher κ corresponding to an elevated rigidity class. Most importantly for the present discussion, although $\eta + 1$ connectivity is necessary but *insufficient* to guarantee global rigidity in $\mathbb{R}^3$, it turns out to be a highly probable indicator, as shown in Fig. 1. This observation suggests the idea that for practical applications we may identify globally rigid subgraphs in $\mathbb{R}^3$ by searching for k-vertex connected components (k-VCCs) of degree $k = 4$, or higher.

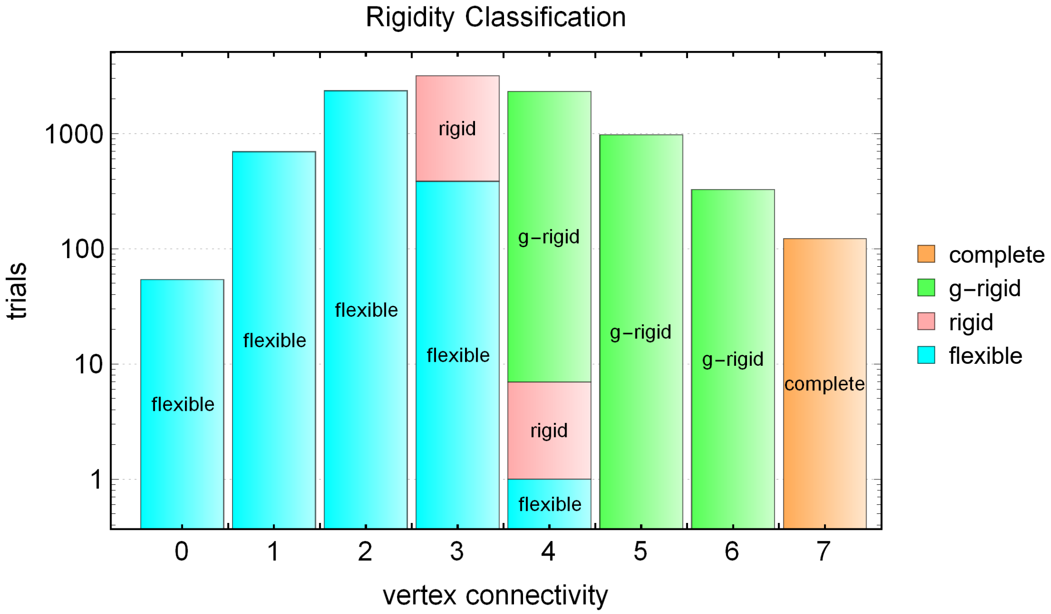

Fig. 1 Stacked histogram of rigidity classes binned by vertex connectivity (log scale) for 10,000 random graphs on $n = 8$ vertices

There is another practical benefit to ranking solvability by connectivity. Whereas a globally rigid graph (with specified edge lengths) is theoretically always solvable, existing localization algorithms tend to do better with higher connectivity. Thus, even if we could identify a marginally globally rigid substructure, we may well do better to choose an alternate substructure with higher connectivity.

2.2 Maximal k-VCC Identification

Various algorithms exist to measure the connectivity of a given graph, as well as coded implementations in various languages.[7] It has been more difficult to find algorithms and code to implement maximal k-VCC identification. To fill this gap, an original recursive algorithm has been developed for maximal k-VCC identification. The pseudocode descriptions are shown in Algorithms 1 and 2, and coded implementations in *Mathematica*[7] and MATLAB are available upon request.

Algorithm 1 Hierarchical k-VCC discovery

```
input: $G$, $k_{\text{thresh}}$
output: All $k$-VCC's and associated $k$.
procedure kvccComps($G, k_{\text{thresh}}$)
    $block = \{\{G, k_{\text{thresh}}\}\}$
    declare global variable $kvcc$            ▷array of $k$-VCC's and connectivities
    $kvcc = \varnothing$
    $i = 0$
    while $block \neq \varnothing$ do
        $block \leftarrow \text{kvccStep}(block)$                    ▷see Algorithm 2
        $i = i + 1$
    end while
    return $kvcc$
end procedure
```

The intuition behind this algorithm is the fact that for any k-vertex connected component, the pairwise connectivities must satisfy $k_{ij} \geq k$. Also, candidate components must themselves form maximal cliques on a graph with adjacency matrix $\mathbf{A}$ with elements

$$a_{ij} = \begin{cases} 1 & k_{ij} \geq k \\ 0 & \text{otherwise} \end{cases}. \tag{1}$$

Algorithm 2 k-VCC discovery at each iteration

```
input: $\{\{G_1, k_{\text{thresh},1}\}, \{G_2, k_{\text{thresh},2}\} \ldots\}$
output: $\{\{G_1, k_{\text{thresh},1}\}, \{G_2, k_{\text{thresh},2}\} \ldots\}$          ▷updated, for next iteration
procedure KVCCSTEP($\{\{G_1, k_{\text{thresh},1}\}, \{G_2, k_{\text{thresh},2}\} \ldots\}$)
    for all $\{G_i, k_{\text{thresh,i}}\}$ do
        $\mathbf{C} \leftarrow$ matrix of pairwise vertex connectivities of $G_i$
        $k_g \leftarrow \min(\mathbf{C})$        ▷element-wise minimum, excluding main diagonal
        $k_{\max} \leftarrow \max(\mathbf{C})$        ▷highest possible component connectivity
        $k_0 \leftarrow \min(\mathbf{C} > k_g \wedge \mathbf{C} \geq k_{\text{thresh,i}})$
        if $k_g \geq k_{\text{thresh}}$ then
            $kvcc \leftarrow$ append $\{\mathrm{V}(G_i), k_{\text{thresh,i}}\}$ to $kvcc$    ▷update global variable
        end if
        if $k_g = k_{\max}$ then        ▷test for complete graph
            $block_i = \varnothing$
            return
        end if
        if $k_0 = \varnothing$ then        ▷no more cliques
            $block_i = \varnothing$
            return
        end if
        $\mathbf{A} \leftarrow$ binary matrix$(C \geq k_0)$        ▷adjacency matrix
        $cliques \leftarrow$ findCliques$(\mathbf{A})$        ▷maximal cliques
        if $cliques = \varnothing$ then
            $block_i = \varnothing$
            return
        else
            $k_{\text{next}} \leftarrow \max(k_g + 1, k_{\text{thresh,i}})$
            $\{S_1, S_2, \ldots\} \leftarrow$ set of induced subgraphs of $G_i$ by $cliques$
            $block_i \leftarrow \{\{S_1, k_{\text{next}}\}, \{S_2, k_{\text{next}}\} \ldots\}$
        end if
    end for
    return $\bigcup\{block_1, block_2, \ldots\}$        ▷concatenate and flatten to form 2D array
end procedure
```

Thus, the algorithm proceeds by successively calculating the k_{ij}, dropping edges that do not satisfy the pairwise connectivity requirement, taking the resulting subgraph, searching for maximal cliques, and investigating the connectivity k of each clique (i.e., the connectivity of the subgraph induced by the vertices in this clique). If $k \geq k_{\text{thresh}}$, the corresponding vertex set is recorded (along with its connectivity). k_{thresh} is then raised, and the procedure repeats recursively on each clique (where the associated induced subgraph becomes the next input graph).

This constitutes a generic algorithm to hierarchically decompose any graph into its k-VCCs. For this application, an agent is only interested in k-VCCs of which it is a member (since the agent is seeking to localize *itself* within a neighborhood). To do this, the algorithm simply ignores cliques of which it is not a member, which has the added benefit of also speeding up k-VCC identification. One may also choose to set the input connectivity threshold to $k_{\text{thresh}} = 4$, since this is a necessary condition for global rigidity, as described in Section 1.

Finally, we note that the algorithm can be further accelerated by first identifying the agent's k-core component and searching only in the k-core's induced subgraph. An agent's k-core component is a maximal weakly connected subgraph in which all vertices have degree at least k. The idea here is that any maximal k-VCC of which the agent is a member must be a subset of its k-core. By definition, any two vertices in k-vertex connected component are connected by at least k vertex-disjoint paths. Then each vertex in the k-VCC must have degree at least k.

2.3 Communication Lag

The essential information that the ith agent transmits and records is $\mathbf{W}_i \in \mathbb{R}^{n\times n}$, the weighted adjacency matrix which encodes distance measurements, and $\mathbf{T}_i \in \mathbb{R}^{n\times n}$, which carries the corresponding timestamps. The mobile network is dynamic both in terms of its geometric configuration and communication links, so that received measurements are generally asynchronous. With each received communication, the agent updates these matrix entries to reflect the most recent measurement available for distance w_{ij}. The approach taken here is then to separately synchronize the measurements and then localize. Each agent maintains a parametrically defined number of past measurements for each entry in $\mathbf{W}_i$ and $\mathbf{T}_i$, and spline extrapolation is used to estimate the current $\hat{\mathbf{W}}_i$. While spline-based synchronization is adequate, it is by no means optimal. For practical applications, particularly where motion and noise models are available, a Kalman filter would offer more accurate estimates of current range values, as described in Allik et al.[8]

2.4 Localization

Having identified a vertex subset $V_i^\kappa \in V_i$ belonging to a maximal k-VCC, the last step is to localize the agents belonging to this closed neighborhood. The available ranges are collected in the ith agent's weighted adjacency matrix $\hat{\mathbf{W}}_i^\kappa$, where the over-hat accent denotes that fact that the matrix is an estimate. In particular, we seek to find a coordinate assignment $\mathbf{X}_i^\kappa \in \mathbb{R}^{m \times 3}$, where the pairwise Euclidean distances are consistent with the pairwise distances encoded in $\mathbf{W}_i^\kappa$ and $m = |V_i^\kappa|$.

The problem can be addressed with a family of techniques known as multidimensional scaling (MDS).[9] Several flavors of MDS exist, including faster noniterative subspace methods (so-called "classical MDS"), as well as slower but more accurate and robust iterative methods (e.g., Scaling by Majorizing a Complicated Function [SMACOF],[10] which guarantees monotonic convergence[9,11]). Aside from speed considerations, there are other differences that may be significant. Classical MDS requires all pairwise measurements. In contrast, iterative methods can accommodate missing measurements and weighting, but are inherently vulnerable to false local min convergence and therefore are sensitive to the choice of starting configuration upon which to iterate.

One viable approach is to use the last known estimate of $\mathbf{X}_i^\kappa$ to initialize iterative MDS in the current time-step. This tends to work well in a dynamic setting if $V_i^\kappa \in V_i$ is relatively constant in time so that complete coordinate estimates are available for these agents. For the k-VCC approach described in this report, however, $\mathbf{X}_i^\kappa$ is generally not constant. Instead, therefore, we have adopted a hybrid approach initializing with the last solution when available and classical MDS otherwise. Because classical MDS requires the complete distance matrix, we must somehow estimate the missing entries.

Some authors[12,13] have proposed using an all-pairs shortest path algorithm (e.g., Dijkstra or Floyd–Warshall) to complete the distance matrix. In this approach, the edge weights on G_i^κ are taken to be the corresponding distances. The shortest weighted-graph path is calculated, and the sum of path weights is then taken to be the missing distance. Note that, by definition, a k-VCC will be connected unless the agent in question is completely disconnected from the network. This guarantees that there will always be a path between any two agents in G_i^κ so that all entries can be estimated. Finally, if the approximate maximum sensor range is known, this estimate can be improved upon by replacing multi-hop estimates with the average of the maximum range and the path length (which is approach implemented here).

3. Simulation

3.1 Simulation Model

To investigate the performance of the proposed decentralized k-VCC localization scheme, a multi-agent network was modeled in Simulink. The agents each follow a prescribed trajectory while ranging, communicating, and localizing. At each step, the interaction matrix $G(t)$ is recalculated based on the agent positions $\mathbf{X}(t)$ and maximum sensor range. $G(t)$ defines which range measurements and communications take place. We assume a simple communication scheme where all agents can range and communicate simultaneously. Details of the communication package and the agent's internal recording protocols were described in Section 2. User-defined simulation parameters are described in Table 1, and the Simulink model is shown in Fig. 2. As shown in Fig. 3, the agents are initially clustered together in close proximity. Agents then briefly split into two groups before reassembling. At maximum separation, the groups cannot communicate with each other, and agents are forced to rely on local information to maintain position awareness.

Table 1 Parameters in multi-agent simulation model

Parameter	Value	Description
n	12	number of agents
$delta$	3.3 km	max sensor range
$XWayPoints$	...	network geometry waypoints determine trajectory
$tWayPoints$	...	time vector for $XWayPoints$
f_full	1 Hz	update frequency of full network localization
$f\_neighborhood$	1 Hz	update frequency of k-VCC neighborhood localization
$f\_transceiver$	20 Hz	update frequency of ranging and communication
$lagThresh$	4 s	discard measurements older than $lagThresh$
$nHistory$	4	number of unique past range measurements to retain for interpolation and synchronization

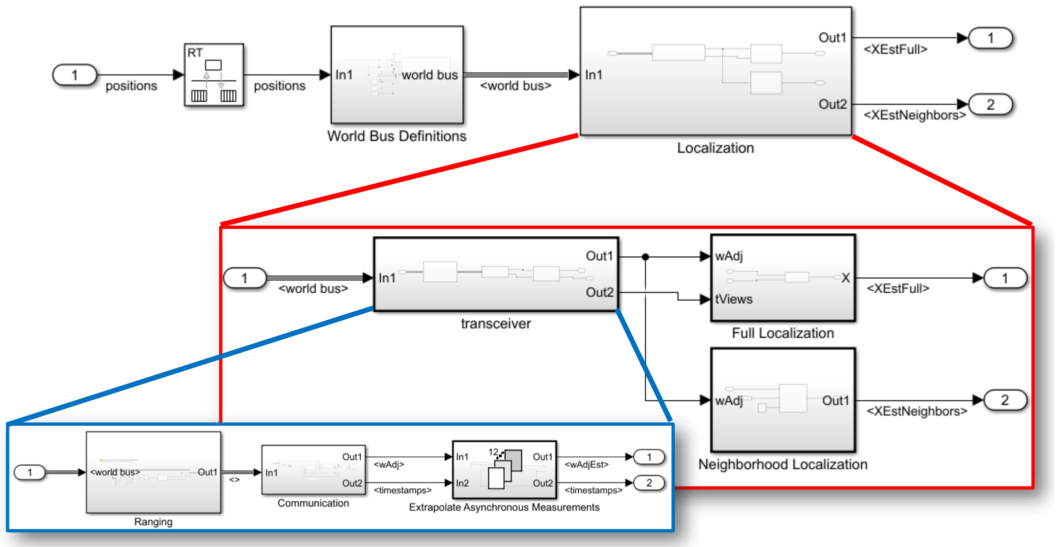

Fig. 2 Multi-agent Simulink model with full network localization and k-VCC localization running side by side

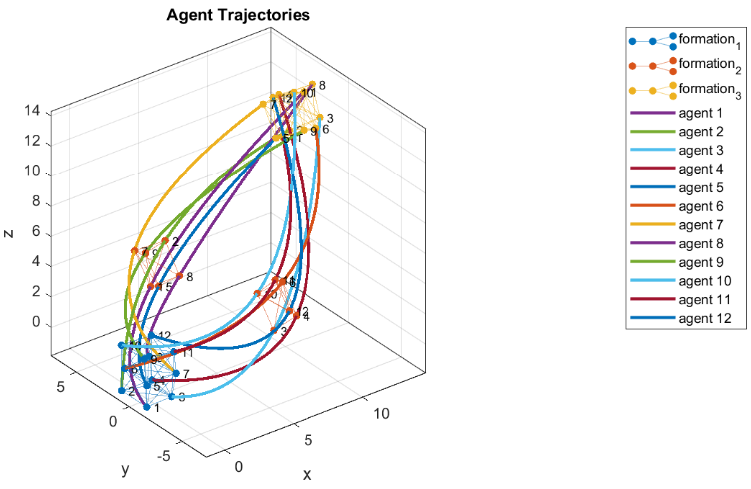

Fig. 3 Example multi-agent trajectory transitioning across waypoint configurations

3.2 Metrics

The choice of metric by which to judge relative (unanchored) localization is complicated by the fact that the solution is, at best, only unique up to rigid transformation. Geometric misalignment of the estimated embedding to the true configuration, while intuitive, would therefore rely on first finding an optimal transformation between the coordinates sets. While this is easily calculated for two morphologically identical embeddings, it becomes less meaningful as the morphologies diverge—due to missed or noisy measurements. Instead, we use a variation of the normalized Kruskal stress (Borg and Groenen,[9] p. 42), which we define as

$$s = \sum_{ij} \sqrt{\frac{\left[(K_{ij})_e - (K_{ij})_0\right]^2}{[(K_{ij})_0]^2}}, \tag{2}$$

where edge kernels $(\mathbf{K})_e$ and $(\mathbf{K})_0$ are constructed from estimated and true position configurations, respectively. The edge kernel is so-named because it collects inner products of all pairwise displacement vectors in the geometric configuration. Construction of $\mathbf{K}$ is detailed in Algorithm 3. Normalization ensures that the scale of the configuration volume does not skew results. Figure 4 shows an overlay of absolute misalignment (arbitrary units) and stress for a sample data set using a classical MDS localization scheme. The example is intended only to reassure the reader that stress correlates well with misalignment.

Algorithm 3 Constructing $\mathbf{K}$

```
input: $\mathbf{X} \in \mathbb{R}^{N \times 3}$ for $N$ agents
output: $\mathbf{K} \in \mathbb{R}^{M \times M}$ where $M = \binom{N}{2}$
procedure EDGEDOT($\mathbf{X}$)
    $N \leftarrow \text{length}(\mathbf{X})$
    $k \leftarrow 1$
    for $i \leftarrow 1, N-1$ do
        for $j \leftarrow i+1, N$ do
            $\mathbf{V}_k \leftarrow \mathbf{X}_j - \mathbf{X}_i$
            $k \leftarrow k+1$
        end for
    end for
    $\mathbf{K} \leftarrow \mathbf{V}\mathbf{V}'$
    return $\mathbf{K}$
end procedure
```

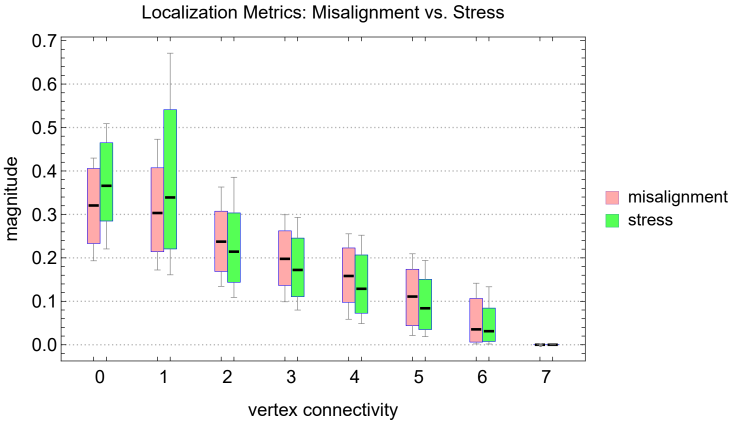

Fig. 4 Average position misalignment is an intuitive metric by which to assess ensemble localization performance but depends on first estimating an optimal rigid transformation to the true point configuration. Stress provides a more direct performance metric because it is invariant under rigid transformation.

4. Results and Discussion

Decentralized models for k-VCC rigid neighborhood localization, as well as full-network localization, are run side by side. The former technique works by identifying neighborhoods with maximal vertex connectivity—which are naturally more rigid, and hence, localizable. Thus, it is illuminating to compare $\kappa(G(t))$ against each agent's $\kappa(G(t)_i^\kappa)$, where the shorthand $\kappa(.)$ is the vertex connectivity of the indicated graph. The connectivity histories for $G(t)$ and $G(t)_i^\kappa, i = 1, 2, ..., n$ are shown in Figs. 5 and 6, respectively. After the initial startup, the general trend is $\kappa(G(t)_i^\kappa) \geq \kappa(G(t))$ meaning that agents are often able to discover smaller neighborhoods with higher connectivity compared to the full network structure. Figure 7 shows a 2-D overlay of the two connectivity histories. This disparity becomes particularly significant during the mid-flight interval when $\kappa(G(t))$ tends to zero while $\kappa(G(t)_i^\kappa) \geq 3$—so that agents can localize to a neighborhood even though the full network may remain unsolvable.

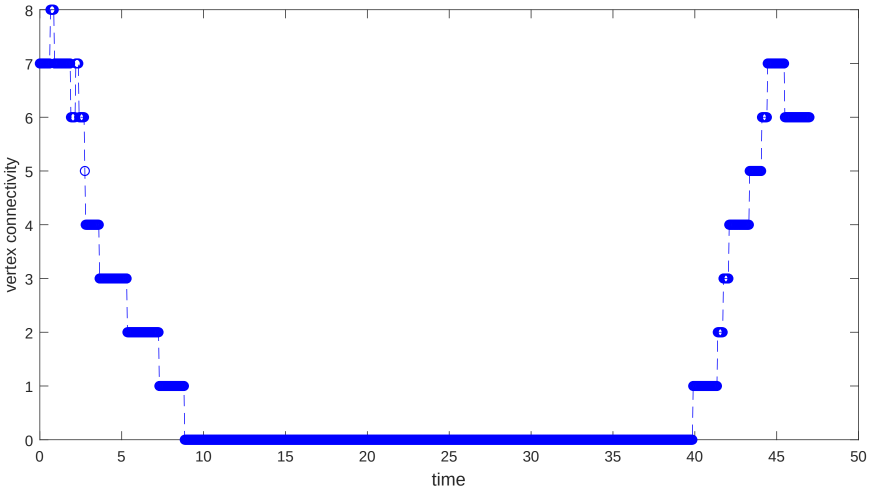

Fig. 5 Vertex connectivity history of the complete mobile network $G(t)$

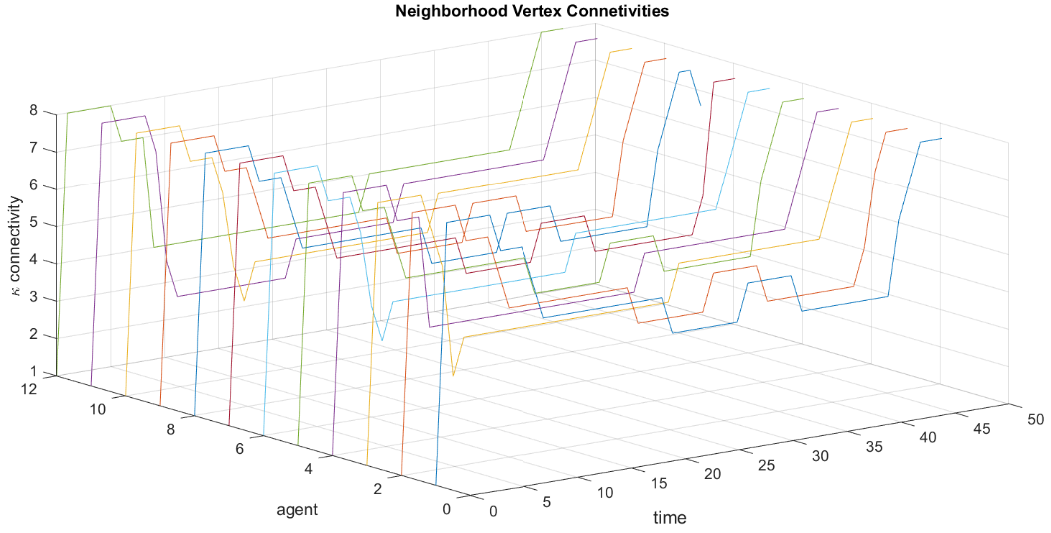

Fig. 6 Vertex connectivities for each agent's discovered neighborhood as a function of time

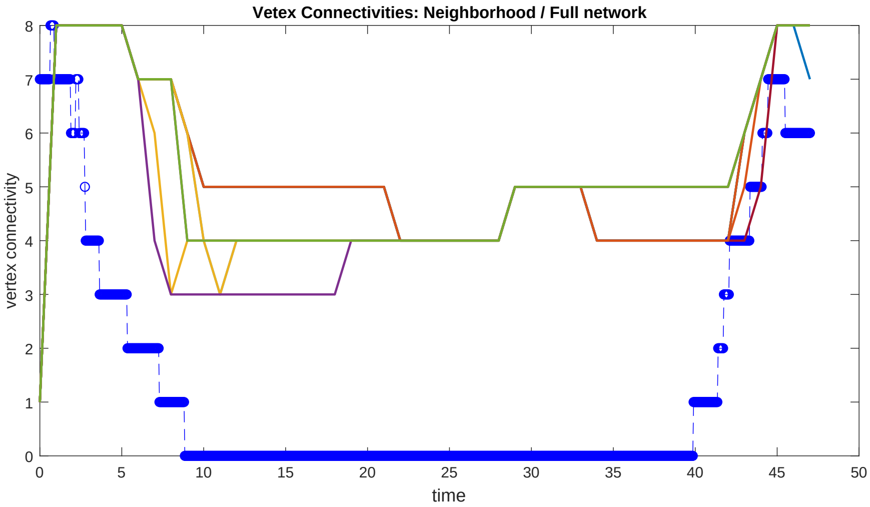

Fig. 7 Overlay of vertex connectivities of full network (blue bubbles) and agents' maximal k-VCC neighborhood

Neighborhood localization performance (stress) across the example trajectory is shown in Fig. 8, and corresponding results for full-network (distributed) localization are shown in Fig. 9. As expected, based on the connectivity results presented earlier, neighborhood localization is particularly advantageous during periods of degraded interaction. Although the threshold connectivity for solvability is $\kappa \geq 4$, these results (for example, agent 10 between 10 and 17 s) show that agents are in fact able to successfully localize at $\kappa = 3$. This is only possible because agents leverage knowledge of the previous solution when possible.

In general, a measurement graph of $\kappa = 3$ carries an ambiguity due to possible reflection so that the two solutions are consistent with the available range measurements. To visualize this, recall that $\kappa = 3$ implies that there are at least two vertices that cannot be connected by more than three vertex-independent paths. Somewhere in the graph there must be three vertices whose removal would sever any connection between the first two vertices. Now, suppose we find a set of coordinates consistent with the available range measurements such that the triplet now defines a plane. If we were to then reflect all positions on one side of the plane with respect to the plane, the pairwise distances along *existing edges* must remain the same, leading to a second possible solution. By the same argument, if there exists another such triplet whose members also share the same side of the plane, the number of solutions will again double. This type of construction can be carried on indefinitely, as long as additional triplets can be found. Importantly, however, these reflections correspond to a discontinuous set of embeddings with (typically) radically different geometries. While it is also possible for a three-vertex connected system to exhibit continuous flex, simulations suggest that rigid configurations are more likely, as seen in Fig. 1—implying discrete, widely separated solutions.

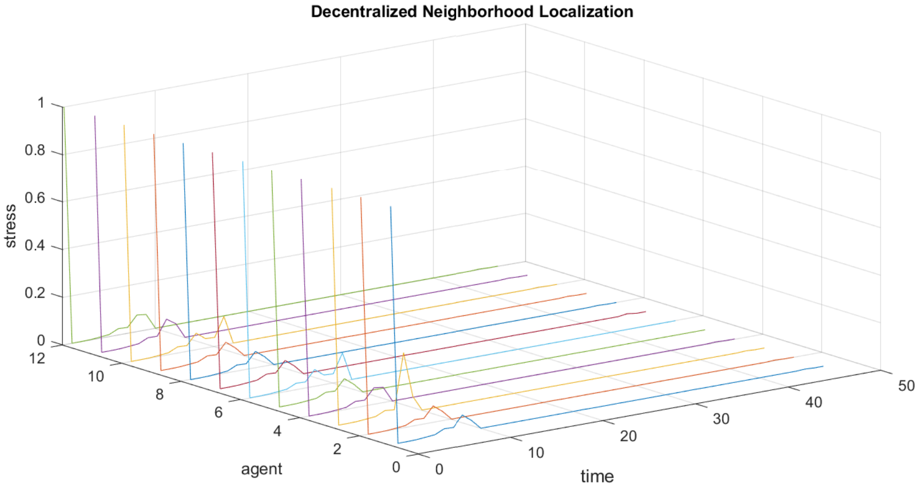

Fig. 8 Neighborhood localization performance across the trajectory

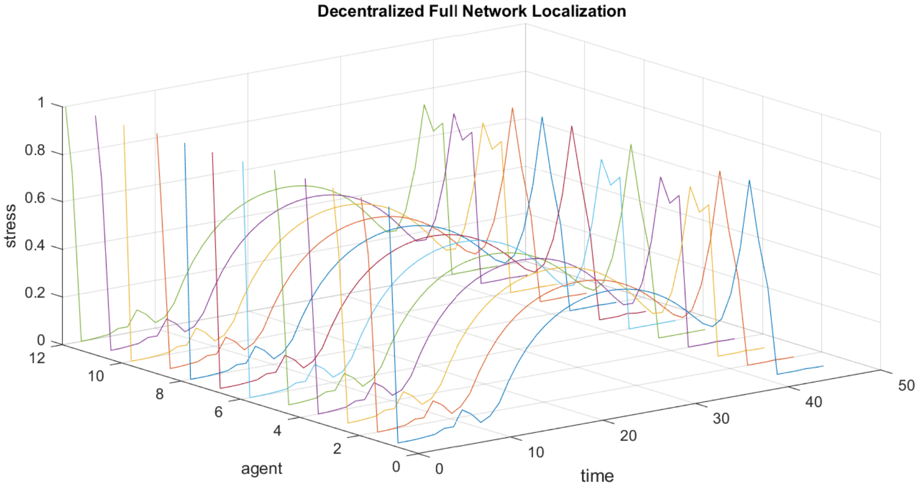

Fig. 9 Decentralized localization performance for full-network awareness

Under such circumstances, and without additional information, there is still no way to decide which solution is correct. For dynamic mobile networks, however, past history can be leveraged to better estimate the current state. In particular, agents retain both the previous solution and the associated metric stress (which indicates goodness of solution). If the last estimate achieved low stress, then this solution becomes the initial guess for the iterative optimization strategy at the current step. Furthermore, if the agent was navigating at some point under conditions of $\kappa \geq 4$, then the system was (almost definitely) uniquely solvable. If the agent then passes briefly into periods of lower connectivity and the geometry has not since changed dramatically, then the optimization routine will naturally converge to the closest solution, thus preserving the correct state estimate. This explains how agents can maintain near-perfect localization for intermittent periods of $k = 3$ connectivity.

Of course, flexible graphs ($\kappa \leq 2$) will also benefit from a good initial guess, but because continuous flex implies an infinite number of solutions in the vicinity of the true configuration, the solution will generally not snap to the correct one (as it appears to for $k = 3$). Instead, localization performance smoothly degrades with time as the geometry deforms (as seen, for example, in Fig. 9 during mid-flight).

5. Conclusion

The main contribution of this report is a robust method of decentralized neighborhood localization for mobile networks with dynamic topologies. Agents must carefully decide which subset of the observed network to localize to assure uniqueness of solution and solvability. It was shown that four-vertex connectivity is a highly probable indicator of global rigidity, and that an agent's maximal k-VCC therefore represents an optimally rigid neighborhood. A recursive algorithm identifying all k-VCCs of a given graph and, specifically, those belonging to a particular vertex (or agent), has been described. An MDS technique was also outlined, which is used to estimate all neighbors' positions based on available (asynchronous) inter-agent range measurements.

To validate this decentralized approach for switching mobile networks, a Simulink framework was constructed to model the temporal flow of information across the agent network. The network itself was modeled as a dynamic proximity graph evolving over time as a function of inter-agent spacings along a prescribed trajectory. This captures the essential features of range-limited communication and measurement. Agent-level protocols then implement ranging, transmission, asynchronous data consolidation, and localization.

An example 12-agent trajectory was chosen for this simulation comparing performance of two decentralized relative localization schemes: 1) k-VCC neighborhood and 2) full network localization. For tightly cohesive networks, the two solutions tend to converge, but results show that neighborhood localization becomes especially advantageous under conditions of clustering subdivision (see Fig. 3).

Other situations which could benefit from neighborhood detection and localization include the following:

1. The network is well connected, with the exception of just a few agents. Even if the rigid component was well estimated under full localization, it is difficult to distinguish between good and bad estimates without resorting to neighborhood detection.

2. The network diameter is large. Even if the complete network is globally rigid, the delays associated with multi-hop communications may render measurements stale—particularly for rapidly deforming mobile networks.

3. The network size is very large, unknown, or changing due to member addition/attrition. Full localization may then be too time-consuming or simply impossible.

Rigid neighborhood detection and localization is a general approach that is not limited to swarming munitions. The technique is applicable to any networked assets or personnel that require relative position awareness, including dismounted Soldiers, search and rescue teams, and autonomous unmanned aerial vechicles. The network itself may comprise any dynamic set of heterogeneous “agents of opportunity”—other participating Soldiers, ground/air vehicles, or beacons. *Unanchored* localization is useful on its own for tasks such as troop coordination, formation control, and patterned weapon delivery. If some of the participating agents also have access to absolute position measurements (i.e., “anchor agents”), then the unanchored solution can be transformed to the world frame, granting all members absolute position awareness, as described in Allik et al.[8]

6. References

1. Hendrickson B. Conditions for unique graph realizations. SIAM Journal on Computing. 1992;21(1):65–84.

2. Jackson B, Jordán T. Connected rigidity matroids and unique realizations of graphs. Journal of Combinatorial Theory, Series B. 2005;94(1):1–29.

3. Connelly R. On generic global rigidity. In: Applied geometry and discrete mathematics; Vol. 4; Providence (RI): American Mathematical Society; 1991. p. 147–155.

4. Tóth C, O'Rourke J, Goodman JE. Handbook of discrete and computational geometry. Third edition. ed. Boca Raton (FL): CRC Press; 2017. Tóth C, O'Rourke J, Goodman J, editors.

5. Connelly R. Generic global rigidity. Discrete & Computational Geometry. 2005;33(4):549–563.

6. Gortler SJ, Healy AD, Thurston DP. Characterizing generic global rigidity. American Journal of Mathematics. 2010;132(4):897–939.

7. Mathematica. Ver. 12.0. Champaign (IL): Wolfram Research, Inc.;. 2019.

8. Allik BL, Hamaoui M, Don M, Miller C. Kalman filter aiding mds for projectile localization. In: AIAA Scitech 2019 Forum; p. 1159.

9. Borg I, Groenen PJF. Modern multidimensional scaling: theory and applications. 2nd ed. New York (NY): Springer; 2005. (Springer series in statistics) Borg I, Groenen PJF, editors.

10. de Leeuw J, Heiser WJ. Convergence of correction matrix algorithms for multidimensional scaling. Ann Arbor (MI): Mathesis Press; 1977.

11. de Leeuw J, Mair P. Multidimensional scaling using majorization: Smacof in r. Journal of Statistical Software. 2009;31(3):30.

12. Shang Y, Ruml W, Zhang Y, Fromherz MPJ. Localization from mere connectivity. In: Proceedings of the 4th ACM international symposium on Mobile ad hoc networking & computing; 2003 June 1–3; Annapolis, MD.; p. 201–212.

13. Ash JN, Potter LC. Robust system multiangulation using subspace methods. In: 2007 6th International Symposium on Information Processing in Sensor Networks; 2007 Apr 25–27; Cambridge, MA.; p. 61–68.

List of Symbols, Abbreviations, and Acronyms

2-D –	two-dimensional
GPS –	global positioning system
k-VCC –	k-vertex connected component
MDS –	multidimensional scaling
RF –	radio frequency
SMACOF –	Scaling by Majorizing a Complicated Function

1 DEFENSE TECHNICAL
(PDF) INFORMATION CTR
DTIC OCA

1 CCDC ARL
(PDF) FCDD RLD CL
TECH LIB

1 CCDC ARL
(PDF) FCDD RLW LF
M HAMAOUI