Graphical properties of easily localizable sensor networks

Brian D. O. Anderson · Peter N. Belhumeur ·
Tolga Eren · David K. Goldenberg · A. Stephen Morse ·
Walter Whiteley · Y. Richard Yang

Published online: 6 April 2007
© Springer Science + Business Media, LLC 2007

Abstract The sensor network localization problem is one of determining the Euclidean positions of all sensors in a network given knowledge of the Euclidean positions of some, and knowledge of a number of inter-sensor distances. This paper identifies graphical properties which can ensure unique localizability, and further sets of properties which can ensure not only unique localizability but also provide guarantees on the associated computational complexity, which can even be linear in the number of sensors on occasions. Sensor networks with minimal connectedness properties in which sensor transmit powers can be increased to increase the sensing radius lend themselves to the acquiring of the needed graphical properties. Results are presented for networks in both two and three dimensions.

Keywords Localization · Sensor networks · Global rigidity · Graph theory

1 Introduction

An important problem in the area of sensor networks is that of sensor network localization. Broadly speaking, a planar or possibly three-dimensional array of sensors exists, and a collection of inter-sensor distances are known. Additionally, the Euclidean coordinates of a small number of sensors (beacon or anchor sensors) are known. The localization problem is then one of determining the Euclidean coordinates of all the sensors.

1.1 Fitting this paper in the taxonomy of sensor network localization

For background papers dealing with various aspects of sensor network localization, see e.g. [1–8]. To grasp the context in which the results of this paper are applicable, we shall give a brief taxonomy of methods of sensor network localization. One classification derives from the fact that different quantities may be sensed in order to perform localization. Methods that in some way determine a distance may use received signal strength (but then require a ‘path loss exponent’ to convert power to distance), one- or two-way
propagation time, or time-difference-of-arrival, and below these are commented upon further. Methods based on determining angles (Angle Of Arrival or AOA) are also well known. See [9–12] for examples of such methods and the recent interesting paper [13] which sets out to define a fundamental framework for localizability and localization using angle data. Within the framework of methods using distance, the broadest difference is between those methods which use neighbor type information, i.e. information for each sensor about the number of other sensors within a particular distance of that sensor, as opposed to actual distance values to those neighbors (albeit noisy values). Examples of methods using neighbor type information can be found in [14–18]. Prominent within these schemes is Multidimensional Scaling (MDS), concerning which it is relevant to include the following quote from [18]: “MDS can provide a very good starting point for local optimisation. MDS is good at finding the right topology of the network, but not the precise locations of nodes, because MDS uses shortest path distances to approximate the distance between nodes more than 1 hop away and the approximation may not be accurate.”

It is naturally of interest to make comparisons among the various methods for localization. However, this is difficult, if not generally impossible, in part because the data sets to which each method can be applied are different in kind, making fair comparisons hard to achieve.

This paper is concerned with localization in networks where actual distance information is available. Among a number of the key questions that can be asked are the following:

(a) How much distance data needs to be collected to localize a network, at least if the data is free of noise?
(b) What is the computational complexity of localization?
(c) Can localization be carried out sequentially, sensor by sensor, or in some kind of distributed fashion, or are central calculations required?
(d) What is the effect of noise (i.e. errors in distance measurements) in the various algorithms that might be advanced?

(Of course, a number of these questions apply to other methods, and it is relevant to note for MDS that it is a centralized algorithm in its raw form, though recent work has attempted to break away from this restriction [18].) Semidefinite programming [19, 20] and stochastic annealing [21, 22] underpin two further classes of centralized algorithms.

The key messages of this paper are that graph theory provides tight answers to (a) and (b); and that by collecting more data, but nevertheless an amount of data that scales linearly with the number of sensors, a helpful answer to (c) can be provided. While little can at this stage be said about (d), we note two recent works that tie together noise with sequential localization based on trilateration [23, 24] and a further work using trilateration ideas to improve an estimate of path loss exponent in the received signal strength approach to distance determination, see [25].

What is primarily being advanced in this paper are theoretical underpinnings for the applicability of a number of methods already advanced by others. However, we do present some simulation data later in the paper that illustrates these theoretical underpinnings, and note that further simulation data relevant to a number of the ideas in this paper can be found in [26, 35]. Before clarifying in more detail the contribution of this paper, we offer a number of remarks concerning the localization problem based on distance measurements. The problem can be split up into an existence or solvability problem and an algorithmic problem. The existence problem is: what are the properties of a sensor network which ensure unique solvability of the localization problem? The algorithmic problem is: how can one go about solving the localization problem, and what is the computational complexity involved in a solution? A more refined question of an algorithmic character is: how can one deal with the presence of errors in the inter-sensor measurements, and how do such errors translate into errors in the algorithm’s output of sensor coordinates?

1.2 The role of graph theory

Many of these problems can be studied in the framework of graph theory, and we will cast the ideas of this paper using this perspective. Let the set of sensor nodes be $S$, let distances $d_{ij}$ between certain pairs of nodes $s_i, s_j$ be given, and suppose the coordinates $p_i$ of certain nodes (the anchor nodes) $s_i$ are known. The localization problem is one of finding a map $p : S \rightarrow R^d$ (where $d$ is 2 or 3) which assigns coordinates $p_i \in R^d$ to each node $s_i$ such that $\|p(i) - p(j)\| = d_{ij}$ holds for all pairs $i, j$ for which $d_{ij}$ is given, and the assignment is consistent with any node coordinate assignments provided in the problem statement.

We can associate a graph $G = (V, E)$ with a sensor network by associating a vertex of the graph with each sensor (the vertex set is $V$), and an edge of the graph with each sensor pair for which the inter-sensor distance is known (the edge set is $E$). Let $|V|$ denote the number of vertices and $|E|$ the number of edges. A d-dimensional framework $(G, p)$ is a graph $G = (V, E)$ together with a map $p : V \rightarrow R^d$. The framework is a realization if it results in $\|p(i) - p(j)\| = d_{ij}$ for all pairs $i, j$ where $i, j \in E$. One can form a mental picture of such a framework as a physical structure of bars and joints, with the bar lengths equal to the prescribed distances. Two frameworks $(G, p)$ and $(G, q)$ are equivalent if $\|p(i) - p(j)\| = \|q(i) - q(j)\|$ holds for all pairs $i, j$ with $i, j \in E$. The two frameworks $(G, p)$ and $(G, q)$ are congruent if $\|p(i) - p(j)\| = \|q(i) - q(j)\|$ holds for all pairs $i, j$ with $i, j \in V$. This is the same as saying...
that \((G,q)\) can be obtained from \((G,p)\) by an isometry of \(\mathbb{R}^d\), i.e. a combination of translations, rotations and reflection.

A framework \((G,p)\) is rigid if there exists a sufficiently small positive \(\varepsilon\) such that if \((G,q)\) is equivalent to \((G,p)\) and \(\|p(i) - q(i)\| < \varepsilon\) for all \(i \in V\) then \((G,q)\) is congruent to \((G,p)\). Intuitively, a rigid framework cannot flex. We remark that there exist rigid frameworks \((G,p)\) and \((G,q)\) which are equivalent but not congruent, see [27]. A framework \((G,p)\) is globally rigid if every framework which is equivalent to \((G,p)\) is congruent to \((G,p)\). Obviously, if \(G\) is a complete graph then the framework \((G,p)\) is necessarily globally rigid.

Given the graph and distance set of a globally rigid framework, there is not enough information to position the framework absolutely in \(\mathbb{R}^d\). To do this requires the absolute position of at least three vertices \((d = 2)\) or four vertices \((d = 3)\), and in fact they must be generically positioned. In \(\mathbb{R}^2\) for example, knowledge of the absolute position of three vertices would not be sufficient if they were collinear, as the absolute position of all other vertices would only be determined up to reflection about the line passing through the collinear vertices.

The existence/solvability problem for sensor networks can be thought of as follows. Suppose a framework is constructed which is a realization, i.e. the edge lengths corresponding to the collection of inter-sensor distances. The framework may or may not be rigid; and even if it is rigid, there may be a second and differently shaped framework which is a realization (constructable with the same vertex, edge and length assignments). If up to congruence there is a unique rigid framework consistent with the lengths, i.e. the framework is globally rigid, then the sensor network can be thought of as like a rigid entity of known structure, and one then only needs to know the Euclidean position of several sensors in it to locate the whole framework in two or three dimensional space as the case may be, see [6]. So the existence/solvability problem is describing how one can decide if a prescribed framework is globally rigid. Consider a rigid framework \((G,p)\) in \(\mathbb{R}^d\). It is said to be generic if the set containing the coordinates of all its points is algebraically independent over the rationals. It is known that rigidity of frameworks in \(\mathbb{R}^d\) is a generic property. In other words, provided \(p\) is generic, the graph \(G\) alone determines the rigidity of the framework, and so we can speak of rigidity of a graph. Note that there may be a thin set of vertex positions (in general defined by one or more equalities which are polynomial in the vertex positions and have rational coefficients) such that rigidity does not hold on this set. We remark that there is a test for rigidity involving the rank of a matrix with entries formed from the coordinates of the vertices, and in two dimensions there is a combinatorial (essentially graph theoretic) necessary and sufficient condition for rigidity, termed Laman's theorem [28]. Although the matrix rank condition generalizes to three dimensions, no necessary and sufficient combinatorial condition is available for three dimensions; the obvious generalization of the Laman conditions are only necessary in three dimensions, but not sufficient [29, 30]. What of the global rigidity property? In two dimensions, there is an elegant necessary and sufficient condition for generic global rigidity of a framework (i.e. global rigidity of a generic framework), and it is of a graph theoretic nature: either the associated graph is the complete graph on three vertices or it must have two properties which we unpack immediately below, viz. it must be 3-connected and it must be generically redundantly rigid [27, 31]. The definition of 3-connectedness is standard: between any two vertices of the graph, there must exist at least three paths which have no edge or vertex in common (apart from the end vertices), or equivalently, it is not possible to find two vertices whose removal (together with the removal of the edges incident on them) would render the graph unconnected. A graph is termed generically redundantly rigid if with the removal of any edge, it remains generically rigid. In two dimensions, there is a variant of Laman’s theorem for checking generic redundant rigidity.

In three-dimensional space, it is necessary that a graph be 4-connected and generically redundantly rigid for the graph to be generically globally rigid. However, these conditions are known to be insufficient [32, 33]. No necessary and sufficient conditions for generic global rigidity are known, and it is not clear that such conditions have to exist, in contrast to the two dimensional case. That is, there may be examples of three dimensional graphs for which specification of a set of lengths confined to certain intervals for each length always guarantees global rigidity, while specification of the lengths for the same sensor pair but confined to other intervals for each length results in lack of global rigidity. On the other hand, it is clear that there do exist graphs which are generically globally rigid in \(\mathbb{R}^3\), for example, any complete graph with 4 or more vertices.

The computational complexity of the algorithmic localization problem has been dealt with in the literature. The general answer is that the computational complexity of localization is NP-hard and probably exponential in the number of vertices [34]; this continues to be true for an important subclass of sensor network graphs, those which are unit disk graphs (which capture the common practical constraint that two vertices have an edge joining them in the graphical representation of the sensor network if and only if the sensors are closer than a pre-specified constant distance, say, often termed the sensing radius [35]).

Not surprisingly however, exceptions to the general computational complexity conclusion can be found by imposing more conditions on the underlying graph. In particular, one might expect that with more data, i.e. more intersensor distances being specified than the minimum number.
required to secure generic global rigidity of the underlying graph, there might be an opportunity to cut computational costs. Indeed this is so. There is an important class of graphs in two dimensions, called trilateration graphs and defined in detail in a later section, in which the computational complexity of localization is polynomial, and on occasions linear, in the number of vertices [6]. (The linear result applies if a so-called seed of the trilateration graph is known; if it is unknown, it takes in general polynomial time to find.) Now while many graph theory results available for two dimensions do not generalize, or do not generalize straightforwardly, to three dimensions, the trilateration conclusion is a happy exception: a generalization, which we can term quadrilateralization, allows a three-dimensional sensor network with the quadrilateral property to be localized in polynomial (or linear, given a seed) time.

We come now to the key contribution of this paper. It is to explain how to systematically construct generically globally rigid, trilateration and quadrilateralization graphs from graphs without these properties. When we say ‘construct’, we imply ‘by adding extra edges in the graph’. However what is also important is how the addition of extra edges can be implemented in the underlying sensor network. Roughly speaking, it involves sensors determining distances not just to their immediate neighbors, but also to their two-hop, three-hop and even four-hop distant neighbours. This may be a natural thing to do in a sensor network: for a network for which the underlying graphical model is a unit disk graph, it corresponds to upping the sensing radius (presumably by adjusting transmit powers) for each sensor. In the case of determining distances to two-hop neighbours, doubling of the sensing radius will suffice. Indeed, an advantage of this construction-based approach is that during deployment, sensors can first perform simple topology control (using for example measurement power control) to construct a topology with simple connectivity-based properties. Topology control for connectivity is well studied (e.g., [23, 40, 42–44]). Then using our construction-based operation (e.g. power control), the network constructs a localizable topology. Note the important property that upward adjustment of the sensing radius may only be required for a localization step, which may only need to be performed once, or at least occasionally. For handling of data collected by the sensor network (apart from that used in localization), a lower degree of connectivity may well suffice.

There are, actually, other alternatives to sensor radius adjustment, however that is achieved. First, if a sensor can determine the angle between two of its neighbours in addition to the distances to those neighbours, then the cosine law allows determination of the distance between those neighbours. Any pair of neighbours of a given sensor are either neighbours of each other or at a two-hop distance from each other, and so all direct distances between two-hop neighbours can be determined given applicability of the cosine law idea. Second, especially if the network is a random network, i.e. one where sensors are positioned in accord with some prescribed distribution, often uniform or Poisson, doubling of the sensor radius can be replaced by using 4 times as many sensors with the same sensor radius as before.

Two further points should be noted. First, in order that a sensor sense and be sensed by its two-hop distant neighbors, a doubling of the sensing radius may be excessively great. Suppose a particular sensor \( j \) has \( n_j \) neighbors. Let every sensor pass to its neighbors the list of its own neighbors. Each sensor in this way can learn the list of its two-hop neighbors. If sensors increase their powers synchronously, they only need to do so until the correct set of two-hop neighbors are seen. Second, in order to communicate with two-hop neighbors, the communication may not need to be as frequent as that with the immediate neighbors (and thus a saving of power can be achieved). In fact, it might only be required once. The point of communicating with two-hop neighbors is often to eliminate a binary ambiguity (known as flip ambiguity). Once this is eliminated, even for a moving sensor network, it may be enough to remain within range only of the original neighbors.

1.3 Structure of the paper

Sections 2 and 3 deal with the construction of globally rigid graphs in two and three dimensions respectively, Section 4 deals with the construction of trilateration graphs in two dimensions and quadrilateralization graphs in three dimensions. Section 5 presents evaluation results. Section 6 contains concluding remarks. The result of Section 2 was announced in [6] without proof. The three-dimensional results of Section 3, the results of Section 4, and the evaluations in Section 5 are new.

2 Generating globally rigid two-dimensional graphs

Before stating the main result of the section, we need to introduce some notation. Let \( G = (V, E) \) be a graph. Then the graph \( G^2 \) is defined as \( (V, E \cup E^2) \) where \((v_a, v_b) \in E^2\) just when \( v_a \neq v_b \) and there exists \( v_c \) with \((v_a, v_c) \in E \) and \((v_b, v_c) \in E \). Thus \( G^2 \) is obtained from \( G \) by adding edges between the vertex pairs of \( G \) which are separated by precisely one intermediate vertex, i.e. by adding edges between the two-hop vertex pairs of \( G \). The concept of the power of a graph can be found in the literature, e.g. [36], see page 74.

Second, we say that a graph is edge \( k \)-connected if between any two vertices, there exist \( k \) paths with no two paths sharing an edge in common (though two paths may have a vertex
in common). Since $k$-connectedness requires the existence of $k$ paths between any two vertices with no edge or vertex pairwise common, it is evident that $k$-connectedness implies edge $k$-connectedness, but not the converse.

Now we have the main result of the section.

**Theorem 2.1.** Let $G = (V, E)$ be an edge 2-connected graph in $\mathbb{R}^2$. Then $G^2$ is generically globally rigid.

Note that if $G$ is an abstraction of a sensor network, and if an edge occurs in $G$ just when the two corresponding sensors are within a common sensing radius $r$ of one another, then a doubling of the sensing radius will produce a new graph which has $G^2$ as a proper (though not necessarily strictly proper) subgraph. Consequently, the new graph will be generically globally rigid, and the sensor network localizable for generic sensor locations, given three or more anchor nodes.

In Section 1, we outlined a simple procedure indicating how sensors might adjust their power levels in order to replace $G$ by $G^2$. It turns out that the consequence localization task is not especially complicated. The reader not interested in the proof of Theorem 2.1 should proceed straight to Section 2.3 to read how localization can be performed.

2.1 The special case of a cycle

In order to prove this main result, we shall first establish the result for a graph $G$ which is precisely a cycle. Refer to Fig. 1

**Lemma 2.1.** Let $C$ be a cycle in $\mathbb{R}^2$; then $C^2$ is generically globally rigid.

**Proof:** Suppose that $C$ has vertices $v_1$, $v_2$, ..., $v_k$ and edges $v_1v_2$, $v_2v_3$, ..., $v_{k-1}v_k$, $v_kv_1$. If $k = 3$ the result is trivial (as the complete graph on three vertices is generically globally rigid). So assume $k > 3$. We shall show that $C^2$ is 3-connected and then generically redundantly rigid. As noted in the introduction, these properties are necessary and sufficient to establish generic global rigidity.

Consider the existence of paths in $C^2$ between $v_i$ and $v_{2m+1}$ for any $m$ with $2m + 1$ not exceeding $k$. Three paths which have no common vertices other than end vertices are: $v_1v_kv_{k-1}...v_{2m+1}$, $v_1v_3v_5...v_{2m+1}$ and $v_1v_2v_4...v_{2m}v_{2m+1}$. Likewise if we consider paths between $v_i$ and $v_{2m}$, then the following paths have no common vertices other than end vertices: $v_1v_kv_{k-1}...v_{2m}$, $v_1v_3v_5...v_{2m-1}v_{2m}$ and $v_1v_3v_5...v_{2m}$. This establishes the 3-connectedness of $C^2$.

It remains to show that if we remove an edge from $C^2$ then it remains rigid. Suppose an edge is removed which is an edge of $C$, without loss of generality $v_1v_2$. Consider the sequence of triangles, the edges of which are all in $C^2$: $v_1v_2v_3$, $v_1v_3v_4$, $v_1v_4v_5$, ..., $v_{k-2}v_{k-1}v_1$, and the corresponding subgraphs spanned by the vertices and edges as each triangle is added. A Henneberg sequence of vertex additions results, with each member of the sequence differing from the previous one by the addition of one vertex of degree 2. By a standard result in rigid graphs the resulting graph is generically rigid.

It is also a subgraph of $C^2$ which contains all vertices of $C^2$ but does not contain the edge $v_kv_1$. Hence $C^2 \setminus v_kv_1$ is generically rigid. If instead of the edge $v_kv_1$ the edge $v_{k-1}v_1$ is removed, the same argument applies. Hence generic redundant rigidity of $C^2$ is established.

While the above lemma establishes the generic global rigidity property of $C^2$, it contains no indication of any algorithm by which $C^2$ might be realized, or how a globally rigid framework realizing $C^2$ might be found. We shall interrupt the flow of the proof of Theorem 2.1 to establish this, in the process providing an alternative proof to Lemma 2.1.

Suppose $C$ has vertices $v_1$, $v_2$, ..., $v_k$ and edges $v_1v_2$, $v_2v_3$, ..., $v_{k-1}v_k$, $v_kv_1$. Then $C^2$ has edges $v_1v_2$, $v_1v_3$, $v_2v_3$, $v_2v_4$, ..., $v_{k-1}v_k$, $v_{k-1}v_1$, $v_kv_1$ and $v_kv_2$. Consider the realization of a framework $F$ corresponding to $C^2$, where we fix the coordinates of $v_1$ and $v_2$ so that $v_1 = (0, 0)$, $v_2 = (a, c)$ for some $a, c > 0$. Knowledge of the lengths of $v_2v_4$ and $v_3v_4$ establishes the position of $v_4$ with a binary ambiguity. For each of these possible locations for $u_4$, knowledge of the lengths $v_3u_5$ and $u_4v_5$ will establish the position of $v_5$ with a binary ambiguity, making four possibilities in all. Successively, we obtain the positions of $v_6$, $v_7$, ..., $v_k$ with $2^3$, $2^6$, $2^{k-3}$ ambiguities. However, $v_k$ is also connected to $v_1$ and $v_2$. Knowledge of the associated lengths resolves the ambiguity in the position of $v_k$. (In fact, knowledge of the length between $v_k$ and $v_1$ alone will be sufficient; further, there is yet another edge, viz., that joining $v_{k-1}$ to $v_1$, the use of which would resolve all ambiguities save that associated with $v_k$.) Resolving the ambiguity in $v_k$ then sequentially allows resolution of the ambiguity in
v_{k-1}, v_{k-2}, \ldots, v_4$, and in this way the unique realization of the framework (up to congruence) is established.

The idea is depicted in Fig. 2.

As noted above, this constructive procedure provides an alternative proof of Lemma 2.1. A further proof again has been suggested in a communication of Cheung and Whiteley [38], based on iterating a Henneberg edge-splitting construction starting with the complete graph on four vertices. Such a procedure establishes that when $C$ contains $n$ vertices, then $C^2$ less the edges $v_1v_n$ and $v_2v_n$ is globally rigid, for $n = 5, 6, \ldots$.

2.2 Generic edge 2-connected graphs

We return now to the proof of Theorem 2.1. A further two Lemmas will be used to prove the main result. The second especially is intuitively obvious, and no proof will be given here.

**Lemma 2.2.** Let $H_0$ be a generically globally rigid graph in $\mathbb{R}^2$ with at least three vertices, and let a further graph $H_1$ be defined by adjoining one vertex to the vertex set of $H_0$, and three edges, each connecting the new vertex to three different vertices of $H_0$. Then $H_1$ is generically globally rigid.

**Proof:** Let $a$ be the additional vertex adjoining to $H_0$. Let $F_0$ be a generic framework realizing the graph $H_0$. Consider the realization of a framework $F_1$ corresponding to the graph $H_1$, where the vertices other than $a$ are located as in the framework $F_0$. The one vertex of $F_1$ for which coordinates have to be determined is the vertex $a$. Consider any two of the three edges incident on $a$; knowledge of each length position $a$ on the circumference of each of two circles, and thus generically there are two possible points for $a$ to lie at, relative to $F_0$. Knowledge of the length of the third edge linking $a$ to $F_0$ then eliminates the ambiguity. Thus the global rigidity of $F_0$ implies the same property for $F_1$. Hence $H_1$ is generically globally rigid.

**Lemma 2.3.** Let $H_1 = (V_1, E_1)$ and $H_2 = (V_2, E_2)$ be two generically globally rigid graphs in $\mathbb{R}^2$ with at least three vertices in common. Then $H_1 \cup H_2 = (V_1 \cup V_2, E_1 \cup E_2)$ is generically globally rigid.

Note that the qualification implied by the word generically in the preceding two lemmas is mild overkill. Whereas genericity demands of a set of vertices that their coordinates do not satisfy a polynomial equation involving rational coefficients, all that is actually required in the hypotheses of the lemmas is that the points be in general position, i.e. that there are no three vertices on a line (a property which is implied by genericity). If the three vertices in a realization of the graph $H_0$ of Lemma 2.2 happen to be collinear for example, then the realization of the graph $H_1$ will not be globally rigid, no matter where the extra vertex is located.

**Proof of Theorem 2.1:** Because $G$ is edge 2-connected, it necessarily contains at least one cycle. If $G$ contains just one cycle we are done. Therefore, suppose that $G$ contains more than one cycle and that one cycle is $C_1 = v_1 v_2 \ldots v_k$. If the vertex set of $C_1$ is identical with that of $G$ it is clear we are done by Lemma 2.1. Suppose then it is not identical; because $G$ is connected, every vertex in $G \setminus C_1$ is joined by a path to $C_1$ and therefore there is a vertex of $G \setminus C_1$ that is connected by a single edge to a vertex of $C_1$. Call this vertex $v_{L}$, and without loss of generality let the edge be $v_1v_{L}$.

Now consider the graph $G_1 = (V_1, E_1)$ with vertex set that of $C_1$ together with $v_L$ and with edge set that of $C_1$ together with $v_1v_L$. Then $G_1^2$ has as its edge set the edges of $C^2$ and three more edges, viz $v_1v_{L}, v_2v_{L}$, and $v_kv_{L}$. By Lemma 2.2, and identifying $C^2$ with $H_0$, we see that $G_1^2$ is generically globally rigid.

Now because $G$ is edge 2-connected, there is necessarily a second path other than the single edge $v_1v_L$ linking the two vertices $v_1$ and $v_{L}$, i.e. there is a cycle, call it $C_2$, containing

![Graph](image-url)
Fig. 4 An algorithm to localize $G^2$ when $G$ is 2-edge-connected

- $G(V, E)$: the input 2-connected graph
- $L$: the set of already localized sensor nodes

identify a cycle $C = v_1v_2...v_k$ from $G$
$v_1 \leftarrow (0, 0)$; $v_2 \leftarrow (a, 0)$; $v_3 \leftarrow (b, c)$
localize the sensor nodes in $C$
$L \leftarrow \{\text{nodes in } C\}$

while $(L$ does not contain all nodes$)$

- construct graph $G_m(V, E_m)$ from $G$ with node set $V$ and edge set $E_m$:
  $E_m = \{(u, v) : \exists u' \in L \land (u, u') \in E \} \cup \{(v_1, v_2) \in E : v_1, v_2 \notin L\}$
- identify a cycle $C = v_1v_2...v_k$ in $G_m$ starting from $v$
- localize nodes $v_1$ to $v_k$
- $L \leftarrow L \cup \{v_1, ..., v_k\}$

compute sensor true positions using a transformation based on anchor positions

these two vertices as successors. The cycle clearly cannot contain both $v_2$ and $v_k$ as a successor of $v_1$. Without loss of generality, suppose it does not contain $v_2$ as a successor of $v_1$. Consider the graph $G_2 = (V_2, E_2)$ with vertex set that of $C_2$ together with $v_2$ and with edge set that of $C_2$ together with $v_1v_2$. Then arguing as in the previous paragraph, we have that $G_2^3$ is generically globally rigid. This graph also contains the three vertices $v_1$, $v_2$, $v_k$, together with the three edges joining them. The graph $G_2^3$ has the same property; since both are globally rigid, the graph formed from the union of the vertex sets and the edge sets of $G_1^3$ and $G_2^3$ is generically globally rigid, by Lemma 2.3. This graph is obviously a subgraph of $(G_1 \cup G_2)^3$ with the same vertex set. Accordingly, $(G_1 \cup G_2)^3$ must then be globally rigid.

If the vertex set of this graph is a strictly proper subset of the vertex set of $G$, then one must find a further vertex joined by a single edge to $C_1 \cup C_2$, and then argue as in the immediately preceding paragraph. This procedure is continued, until the set of vertices of $G_1 \cup G_2 \cup \ldots \cup G_r$, for some $r$ is identical with the vertex set of $G$, and $(G_1 \cup G_2 \cup \ldots \cup G_r)^3$ is generically globally rigid and a subgraph of $G^3$. This establishes that $G^2$ is generically globally rigid. $\Box$

2.3 An algorithm to localize $G^2$

We can describe fairly easily how localization occurs for a sensor network with a graph which is of the form $G^2$, where the underlying graph $G$ is 2-edge-connected. (If there is a subgraph of the graph of the sensor network containing all vertices and of the form $G^2$, the result is equally true). We identify a cycle $C_1$ in $G$, with vertices $v_1, v_2, \ldots, v_k$ and edges $v_1v_2, v_2v_3, \ldots, v_{k-1}v_k$. In the associated framework, temporarily suppose that $v_1$ is located at (0, 0), $v_2$ at $(a, 0)$ and $v_3$ at $(b, c)$ for some $a, c > 0$. Following the procedure given just above Fig. 2, which is used to illustrate the procedure, the distances associated with the edges of $C^2$ allow localization of $v_4, v_5, \ldots, v_k$ (retaining the temporary coordinate basis). We locate a second cycle $C_2$ of $G$ intersecting $C_1$ but with at least one distinct vertex. In the associated framework, we localize the vertices of $C_1 \cup C_2$ using the edges of $(C_1 \cup C_2)^3$; this is straightforward and is described in the theorem proof. The procedure is continued until all vertices are localized. Using anchor positions, an isometry based on translation, rotation and possible reflection of the initially localized framework (the location of which depended on making a temporary assumption about the positions of $v_1, v_2$ and $v_3$) can be determined, which yields new and correct localized values. Figure 4 formally specifies the algorithm.

Note that in order to improve algorithm efficiency, when we search for a new cycle $G_m$ intersecting previous cycles and a new first vertex, call it $v$, in $G_m$, we try to identify a short cycle. This can be achieved by modifying the standard depth-first search algorithm [37]; at each node, before we take the recursion, we first check if any of the neighbors of the node is directly connected back to $v$. Also note that we can reduce the complexity of the algorithm by using ear-decomposition to identify the cycles at the beginning of the algorithm, but this may be less effective compared with the algorithm in Fig. 4.

In a preprint of Cheung and Whiteley [38], it is noted that the 2-edge connected condition in the theorem statement can tolerate a minor relaxation, and in the process establish a necessary and sufficient condition for $G^2$ to be globally rigid: $G$ must be connected, and such that if the removal of any edge $e$ disconnects $G$, then one of the two components is a single vertex. The proof is a simple extension of that given above.

3 Generating globally rigid three-dimensional graphs

While sensor networks in two dimensions appear much more common than those in three dimensions, it is apparent that there should be no inherent limitation of interest to two
dimensions. In this section, we prove an extension of the two dimensional result of the previous section. Again, our starting point will be the properties of a cycle. However, we cannot proceed by working with the three-dimensional generalisations of 3-connectivity and redundant rigidity, and we need an alternative procedure in \( \mathbb{R}^3 \) to establish generic global rigidity of a certain graph derived from a cycle. The procedure will be like that mentioned in Section 2 after the proof of Lemma 2.1, where we indicated how a realization of \( C^2 \) (for a two-dimensional cycle \( C \)) could be found.

We need to introduce some notation. Let \( G = (V, E) \) be a graph in \( \mathbb{R}^3 \). Then the graph \( G^3 \) is defined as \((V, E \cup E^2 \cup E^3)\) where \((v_i, v_j) \in E^3 \) when there exists \( v_k \) and \( v_l \) with \((v_k, v_l) \in E, (v_i, v_l) \in E \) and \((v_k, v_i) \in E \). Thus \( G^3 \) is obtained from \( G \) by adding edges between those vertex pairs of \( G \) which are separated by precisely one or two intermediate vertices, i.e. by adding edges between the two-hop and three-hop vertex pairs of \( G \).

The result we shall in fact prove is the following.

**Theorem 3.1.** Let \( G = (V, E) \) be an edge 2-connected graph in \( \mathbb{R}^3 \). Then \( G^3 \) is generically globally rigid.

When \( G \) is associated with a sensor network in which every sensor has a common sensing radius \( r \), a tripling of the radius will induce a graph of which \( G^3 \) is a (not necessarily strictly proper) subgraph. (If vertices joined by a path with three or more intermediate vertices correspond to sensors closer than \( 3r \), then \( G^3 \) will be a strictly proper subgraph.) Radius tripling provides a (potentially expensive) way of securing the level of connectivity required to achieve sensor network localization, always provided of course that one can postulate an edge 2-connected graph to start with. Incidentally, based on the two-dimensional result, one might have conjectured a result like that of the theorem, but with the stronger hypothesis of edge 3-connectivity rather than edge 2-connectivity.

### 3.1 The special case of a cycle

As for the two-dimensional case, we shall first consider a graph which is exactly a cycle; then we shall build out to a general graph. Accordingly, as the starting point for proving the theorem, we shall prove:

**Lemma 3.1.** Let \( C \) be a cycle in \( \mathbb{R}^3 \); then \( C^3 \) is generically globally rigid.

**Proof:** The case \( k = 4 \) is trivial, since then \( C^3 \) is the complete graph on 4 vertices. So suppose there are more than 4 vertices. For \( k = 5 \) it is easy to see that \( C^3 \) is also a complete graph. As such, it is globally rigid.

Suppose then that \( k > 5 \). Now \( v_1, v_2, v_3 \) and \( v_4 \) are vertices of a complete tetrahedral subgraph of \( C^3 \). Also, in \( C^3 \) edges join \( v_3 \) to each of \( v_2, v_3 \) and \( v_4 \)—one can think of this as a kind of Henneberg extension. Hence the subgraph of \( C^3 \) defined by the vertices \( v_1 \) through \( v_3 \) and the edges joining them in \( C^3 \) is rigid; there are two possible non-congruent frameworks corresponding to the specified distances, being distinguished by the two partial reflections of the tetrahedron defined by \( v_2 \) through \( v_3 \) relative to the tetrahedron defined by \( v_1 \) through \( v_4 \). If \( k < 8 \), this ambiguity is however not present since \( v_1, v_2 \) is an edge whose length in the framework determines which of the two possibilities applies.

Next, \( v_6 \) is connected to \( v_3, v_4 \) and \( v_5 \) in \( C^3 \). (Again, one might think of this as Henneberg extension.) Hence, discounting for the moment the existence of any edges linking \( v_5 \) to \( v_1 \) or \( v_6 \) to \( v_1 \) or \( v_2 \), the subgraph of \( C^3 \) defined by the vertices \( v_1 \) through \( v_6 \) is rigid, with four possible non-congruent frameworks corresponding to the distances; the frameworks are distinguished by the two partial reflections of the tetrahedron defined by \( v_2 \) through \( v_3 \) and a further two partial reflections of the tetrahedron defined by \( v_4 \) through \( v_6 \). In the event that \( k < 9 \), there will be present in \( C^3 \) an edge connecting \( v_5 \) to \( v_1 \) or \( v_6 \) to \( v_1 \) or \( v_2 \), and then the ambiguity will be resolved.

The argument continues in this way, considering the addition of \( v_7, v_8 \), etc., with each additional vertex and its three connecting edges to earlier indexed vertices defining a rigid subgraph of \( C^3 \) with a number of binary ambiguities of noncongruent frameworks defined by partial reflections. The overall ambiguity associated with the aggregate of these partial reflections will be resolved when, if \( m \) is the number of vertices in the cycle \( C \), the vertex \( v_{m-2} \) is introduced, since it is also connected in \( C^3 \) to \( v_1 \). This means that \( C^3 \) is generically globally rigid, as claimed.

### 3.2 General edge 2-connected graphs

To build out the result for a cycle to one applicable to more general graphs than a cycle, we shall rely on the following two Lemmas, which are intuitively obvious variants on two lemmas in the previous section and for which no proof will be given.

**Lemma 3.2.** Let \( H_0 \) be a generically globally rigid graph in \( \mathbb{R}^3 \) with at least four vertices, and let a further graph \( H_1 \) be defined by adjoining one vertex to the vertex set of \( H_0 \), and four edges, each connecting the new vertex to four different vertices of \( H_0 \). Then \( H_1 \) is generically globally rigid.

**Lemma 3.3.** Let \( H_1 \) and \( H_2 \) be two generically globally rigid graphs in \( \mathbb{R}^3 \) with at least four vertices in common. Then \( H_1 \cup H_2 \) is generically globally rigid.
Once again, we note that the qualifier 'generically' in the lemma hypotheses can be replaced by requiring that no four points are in general position, i.e. coplanar. The result of Lemma 3.1 on cycles will not be valid in a realization where four successive vertices of \( C \) are coplanar. And likewise, if in Lemma 3.2 the four vertices of \( H_0 \) to which the new vertex of \( H_1 \) is connected are coplanar in a particular realization, the associated realization of \( H_1 \) cannot be globally rigid.

**Proof of Theorem 3.1:** If \( G \) contains just one cycle we are done. Therefore, suppose that \( G \) contains more than one cycle and that one cycle is \( C_1 = v_1v_2 \ldots v_k \). If the vertex set of \( C_1 \) is identical with that of \( G \) it is clear we are done by the first lemma. If not, then we can choose a vertex of \( G \) that is connected by a single edge to a vertex of \( C_1 \). (Since \( G \) is connected, there must be such a vertex). Call this vertex \( v_L \), and without loss of generality let the edge be \( v_Lv_2 \).

Now because \( G \) is edge 2-connected, there is necessarily a second path other than the single edge \( v_Lv_2 \) linking the two vertices \( v_1 \) and \( v_L \), i.e. there is a cycle, call it \( C_2 \), containing these two vertices as successors. The cycle clearly cannot contain both \( v_2 \) and \( v_3 \) as the other successor node of \( v_L \). In addition, suppose it does not contain \( v_2 \).

Consider the graph \( G_1 = (V_1, E_1) \) with vertex set of \( C_2 \) together with \( v_L \) and with edge set that of \( C_1 \) together with \( v_1v_L \). Then \( G_1 \) has as its edge set the edges of \( C_1 \) and five more edges, viz. \( v_1v_L, v_2v_L, v_3v_L, v_{k-1}v_L, \) and \( v_kv_L \). By Lemma 3.2, and identifying \( C_3 \) and \( H_2 \), we see that \( G_1 \) is globally rigid. Consider also the graph \( G_2 = (V_2, E_2) \) with vertex set that of \( C_2 \) together with \( v_2 \) and \( v_3 \) if this vertex is not in \( C_2 \), and with edge set that of \( C_2 \) together with \( v_1v_2 \) and \( v_1v_3 \) if \( v_3 \) is not in \( C_2 \). Then arguing much as in the previous paragraph, but appealing twice to Lemma 3.2, we have that \( G_2 \) is globally rigid.

The two graphs \( G_1 \) and \( G_2 \) are both globally rigid and have a common set of at least four vertices, viz. \( v_k, v_1, v_2, v_3 \). Hence the graph formed from the union of the vertex sets and the edge sets of \( G_1 \) and \( G_2 \) is globally rigid, by Lemma 3.3. This graph is obviously a subgraph of \( (G_1 \cup G_2)^3 \) with the same vertex set. Accordingly, \( (G_1 \cup G_2)^3 \) must then be globally rigid.

If there are any vertices of \( G \) which are not vertices of \( G_1 \cup G_2 \), then the above line of argument must be repeated, by determining such a vertex which is connected by a single edge to \( G_1 \cup G_2 \), then determining a cycle containing that edge, and so on. As for the two dimensional case, the process can obviously be repeated until the set of vertices of \( G_1 \cup G_2 \cup \ldots \cup G_r \) for some \( r \) is identical with the vertex set of \( G \), and \( (G_1 \cup G_2 \cup \ldots \cup G_r)^3 \) is generically globally rigid a subgraph of \( G^3 \). This establishes that \( G^3 \) is generically globally rigid.

In [38] it is pointed out that a necessary and sufficient condition for \( G^3 \) to be generically globally rigid is that \( G \) is connected, and if the removal of any 2-valent vertex \( x \) should disconnect \( G \), then one of the resulting two components is a single vertex.

It is straightforward to extend the algorithm in Fig. 4 to the three-dimensional case.

4 Generating two-dimensional trilateration and three-dimensional quadrilateration graphs

4.1 Trilateration graphs

We begin by recalling the notion of a trilateration graph, [6]. While this is of principal relevance just in \( \mathbb{R}^2 \), the definition remains valid in \( \mathbb{R}^3 \). Let \( G = (V, E) \) be a graph. Then \( G \) is a trilateration graph if there are \( (a) \) three vertices, \( v_1, v_2, \) and \( v_3 \) say, for which \( v_1v_2, v_2v_3, \) and \( v_3v_1 \) are all edges of \( G \) and \( (b) \) an ordering (actually, a partial ordering suffices) of the remaining vertices as \( v_4, v_5, v_6, \ldots \) such that any \( v_i \) is joined by (at least) three edges to three earlier vertices in the sequence. The three vertices \( v_1, v_2, \) and \( v_3 \) are known as a seed of the trilateration graph. Given a graph that is somehow known to be a trilateration graph, there can be more than one seed and more than one vertex ordering consistent with the trilateration property.

Trilateration graphs are important in \( \mathbb{R}^2 \), because if a sensor network has a trilateration graph, and at least three of the sensors are anchor nodes, i.e. have known Euclidean coordinates, then the whole network can be easily localized, as we now argue. To see this, assume first that one knows the seed and the ordering. Temporarily locate the seed vertices consistently with the edge lengths by requiring one to be at the origin, one to be on the positive \( x \) axis and the remaining one in the positive \( y \) half-plane. Then evidently all vertices can be localized relative to these vertices sequentially, in a single sweep and in time \( O(|V|) \). Then knowledge of the anchor node true positions will define a translation, rotation and possible reflection of the initially determined position of the whole graph to align the anchor nodes with their correct positions, and new positions follow for the rest of the nodes through application of the same translation, rotation and possible reflection. If one knows the seed but does not know the ordering, the time is \( O(|V| + |E|) \). If one does not know the seed, one must experiment with different choices of three nodes as a trial seed from which trilateration-type localization is attempted. There are \( (1/6)|V||(|V| - 1)|(|V| - 2) \) different choices of three nodes from \( |V| \). So the complexity of localization, requiring the identification of a seed followed by the sequential localization of all the vertices, is at worst quartic in the number of vertices, \( O(|E||V|^3 + |V|^4) \) in fact. [Actually, if an upper bound, \( c \) say, on the valency of

\[ \square \]
every vertex is known, then the maximum number of seeding triangles is of order $c^2 |V|$, and the complexity becomes quadratic. These remarks on complexity do not cover any procedure which nodes might use to discover a sensor’s two-hop neighbors or three-hop neighbors. This was described in Section 1 for two-hop neighbors and the complexity is linear in the number of nodes. It will remain linear for three-hop neighbors too.

As will be seen in the main result of this section, there is a simple way to order the vertices of a graph prior to exploiting a trilateration property, and in particular to obtain a seed.

The steps involved in localization, assuming that an increase of sensing radius can be achieved to reach three-hop neighbors are: order the graph vertices as described below (and this can be by propagation through the network, as will be seen, and does not require central computation); increasing the sensing radius (or equivalent measure) to secure trilateration structure, with identified seed; localization of all vertices, relative to the seed; use of anchor positions to obtain absolute position information, differing from the relative localization by a translation and rotation.

Before presenting the main result for $\mathbb{R}^2$, we require the following lemma.

**Lemma 4.1.** Let $G = (V, E)$ be a connected graph with $N$ vertices. Then there exists an ordering $v_1, v_2, \ldots, v_N$ of the vertices of $G$ such that for all $p > 1$, the subgraph of $G$ induced by the set $\{v_1, v_2, \ldots, v_p\}$, denoted $G_p$, is connected.

Note that the above lemma is not restricted to $\mathbb{R}^2, \mathbb{R}^3$, etc. [The proof is straightforward. Pick a vertex, and call it $v_1$. Pick a vertex connected to $v_1$, call it $v_2$. Then pick a vertex connected to either of $v_1$ or $v_2$, etc. One can do this until all vertices have been picked, because of the connectivity condition on $G$.] Now with the above lemma in hand, we can state the main result:

**Theorem 4.1.** Let $G = (V, E)$ be a connected graph with $N$ vertices, and let $v_1, v_2, \ldots, v_N$ be an ordering of the vertices of $G$ such that for all $p > 1$, the subgraph of $G$ induced by the set $\{v_1, v_2, \ldots, v_p\}$, denoted $G_p$, is connected. Then $G^3$ is a trilateration graph, with the same vertex sequence $v_1, v_2, \ldots, v_N$.

Note that the theorem, while valid in both $\mathbb{R}^2$ and $\mathbb{R}^3$, is of principal relevance for $\mathbb{R}^2$, because of the application to localization: tripling the sensing radius of a sensor network containing at least three anchor nodes in $\mathbb{R}^2$ renders it localizable, with attractive computational complexity. Figure 5 illustrates a trilateration graph obtained by the technique of Theorem 4.1.

The proof will be assisted by the following lemma:

**Lemma 4.2.** Let $G = (V, E)$ be a connected graph with $N$ vertices, and let $v_1, v_2, \ldots, v_N$ be an ordering of the vertices of $G$ such that for all $p > 1$, the subgraph of $G$ induced by the set $\{v_1, v_2, \ldots, v_p\}$, denoted $G_p$, is connected. Then in $G^3$, for all $p > 2$, $v_p$ is a neighbour of two distinct vertices in the set $\{v_1, v_2, \ldots, v_{p-1}\}$.

**Proof:** Since $G_p$ is connected, $v_p$ is a neighbour in $G_p$ of some vertex in the set $v_1, v_2, \ldots, v_{p-1}$, say $v_i$. Also, if $i > 1$, the same argument implies $v_i$ is a neighbour in $G_i$ of some vertex in $v_1, v_2, \ldots, v_{i-1}$; as $G_i \subseteq G_p$, $v_i$ is also a neighbour in $G_p$ of some vertex in $v_1, v_2, \ldots, v_{i-1}$, call it $v_j$. If $i = 1$, then $v_i$ is connected to $v_2$. Then in $G^3$, $v_p$ is connected to $v_i$ and $v_j$, where $1 \leq i < j \leq p - 1$, $1 \leq i < j \leq p - 1$, $i \neq j$.

With this lemma in hand, the proof of the theorem is relatively straightforward. The main idea is to extend the result of the lemma from $G^2$ to $G^3$.

**Proof of Theorem 4.1:** Since the subgraph of $G$ induced by the three vertices $v_1$, $v_2$, and $v_3$ is connected, it is obvious that in $G^3$, three edges connect the three vertices. To establish the trilateration property then, all we need to prove is that in $G^3$, for all $p > 3$, $v_p$ is a neighbour of three distinct vertices in the set $\{v_1, v_2, \ldots, v_{p-1}\}$. Regard $v_p$ as a vertex of $G^2_p$. By the lemma immediately above, it is a neighbour of two vertices, say $v_i$ and $v_j$ with $1 \leq i < j \leq p - 1$. Consider now $G_j$. Then $j$ is a neighbour in $G_j$ of some $v_k$ with $1 \leq k < j$. We now consider several cases.

**Case 1:** Suppose $v_k \neq v_i$. Then $v_i$, $v_j$ and $v_k$ are neighbours of $v_p$ in $G^2_p$.

**Case 2:** Suppose $v_k = v_i$ is a neighbour of $v_j$ in $G_j$. Three subcases occur.
Case 2a: If \( j = 2 \), then \( i = 1 \) and \( v_3 \) is a neighbour of either \( v_1 \) or \( v_2 \) in \( G_3 \subset G_p \); then \( v_1, v_2 \) and \( v_3 \) will be neighbours of \( v_p \) in \( G_p \).

Case 2b: If \( j > 2 \) and \( i > 1 \), then \( v_i \) has a neighbour in \( G_3 \subset G_1 \subset G_p \), call it \( v_k \), with \( k < i \); it follows that \( v_k, v_i \) and \( v_j \) are neighbours of \( v_p \) in \( G_p^3 \).

Case 2c: If \( j > 2 \) and \( i = 1 \), then \( v_3 \) is a neighbour of \( v_1 = v_i \), and \( v_1, v_2 \) and \( v_j \) are neighbours of \( v_p \) in \( G_p^3 \).

The hypothesis for the theorem here is less demanding than that for the theorem of Section 2. On the other hand, seen from the viewpoint of adjusting a sensing radius to achieve a particular type of connectivity, the requirement here to assure the trilateration property is to treble rather than double the sensing radius. For the three-dimensional case, which we now treat, having a trilateration property is not enough, and the requirement is to quadruple the sensing radius, to assure a quadrilateration property. Localization can then be achieved in linear time.

### 4.2 Quadrilateration graphs

The quadrilateration property is a simple extension of the trilateration property, and is principally useful in \( \mathbb{R}^3 \). Let \( G = (V, E) \) be a graph. Then \( G \) is a \textit{quadrilateration graph} if there are (a) four vertices, \( v_1, v_2, v_3 \) and \( v_4 \) say, for which \( v_1 v_2, v_1 v_3, v_2 v_3, v_2 v_4 \) and \( v_3 v_4 \) are all edges of \( G \) and (b) an ordering of the remaining vertices as \( v_5, v_6, v_7, \ldots \) such that any \( v_i \) is joined by (at least) four edges to four earlier vertices in the sequence. The four vertices \( v_1, v_2, v_3 \) and \( v_4 \) are known as a seed of the quadrilateration graph.

The key to the main result is the following lemma, generalizing both Lemma 4.2 above, and the key idea of the theorem on trilateration graphs.

**Lemma 4.3.** Let \( G = (V, E) \) be a connected graph with \( N \) vertices, and let \( v_1, v_2, \ldots, v_N \) be an ordering of the vertices of \( G \) such that for all \( p > 1 \), the subgraph of \( G \) induced by the set \( \{v_1, v_2, \ldots, v_p\} \), denoted \( G_p \), is connected. Then in \( G^4 \), for all \( p > 4 \), \( v_p \) is a neighbour of four distinct vertices in the set \( \{v_1, v_2, \ldots, v_{p-1}\} \).

**Proof:** Regarding \( v_p \) as a vertex of \( G^3 \), it is a neighbour of \( v_j, v_k \) and \( v_l \) for some \( 1 \leq i < j < k < p \). By considering a limited number of particular cases similar to the proof of Theorem 4.1, it follows that there exists a vertex \( v_m, m < p \), \( m \neq i, j, k \) such that in \( G_p \), \( v_m \) is a neighbour of one of \( v_i, v_j \) or \( v_k \). Then \( v_i, v_j, v_k \) and \( v_m \) are neighbours of \( v_p \) in \( G_p^4 \).

The proof of the following theorem is now immediate from this lemma.

**Theorem 4.2.** Let \( G = (V, E) \) be a connected graph with \( N \) vertices, and let \( v_1, v_2, \ldots, v_N \) be an ordering of the vertices of \( G \) such that for all \( p > 1 \), the subgraph of \( G \) induced by the set \( \{v_1, v_2, \ldots, v_p\} \), denoted \( G_p \), is connected. Then \( G^4 \) is a quadrilateration graph, with the same vertex sequence \( v_1, v_2, \ldots, v_N \).

### 5 Evaluation of localization in random networks

We generate 100 instances of test networks each with \( N \) nodes by uniformly distributing the nodes in an area of \( 760 \times 787 \). We do not consider anchors, as we are interested here in how many nodes we can localize.

For each instance of the test networks, we compute the following performance metrics:

- **\( r_1 \):** We raise the sensing radius of the network gradually until the largest connected component of the network contains all of the \( N \) nodes. We refer to this radius as \( r_1 \).
- **\( r_3 \):** One way to achieve \( G^3 \), as required in Theorem 4.1, when \( G \) is the connected network at radius \( r_1 \), is to just treble \( r_1 \). We denote \( 3r_1 = 3 \times r_1 \).
- **\( r_2 \):** Another way to achieve \( G^3 \) when \( G \) is the connected network at radius \( r_1 \) is to start with radius \( r_1 \) at each node, and then raise the sensing radius of each node individually so that it connects to all of its neighbours’ neighbours’ neighbours. We compute the average of the radii of all nodes and denote it by \( r_2^3 \).
- **\( r_2 \):** We raise the sensing radius of the network gradually until the largest 2-connected component of the network contains all of the \( N \) nodes. We refer to this radius as \( r_2 \). Note that at \( r_2 \) we achieve node 2-connectivity, which implies edge 2-connectivity and thus is a stronger condition than required in Theorem 2.1.
- **\( r_2 \):** One way to achieve \( G^3 \) when \( G \) is the 2-connected network at radius \( r_2 \) is to just double \( r_1 \). We denote \( 2r_2 = 2 \times r_2 \).
- **\( r_2^2 \):** Another way to achieve \( G^3 \) is to start with radius \( r_2 \) at each node, and then raise the sensing radius of each node individually so that it connects to all of its neighbours’ neighbours. We compute the average of the radii of all nodes and denote it by \( r_2^2 \).
- **\( r_{GR} \):** We raise the sensing radius of the network gradually until the largest globally rigid component contains all \( N \) nodes. We refer to this radius as \( r_{GR} \).

Figure 6 reports the results for the first 30 instances when \( N = 100 \). We make the following observations. First, controlling the sensing radii of the nodes individually to increase connectivity, e.g., from \( G \) to \( G^2 \) or \( G \) to \( G^3 \), requires lower radius compared to doubling or trebling the network-wide sensing radius. This result is somehow intuitive and is verified by observing that \( 2r_2 > r_2^2 \) and \( 3r_1 > r_1^2 \).
for all test instances, as foreshadowed in the discussion at the end of Section 1.2. It should be noted, however, that connectivity control on individual nodes may have larger overhead than the simpler operation of doubling or trebling the network sensing radius. Second, it is clear that \( 2r_2 \) is higher than \( r_{GR} \). This is expected since the network at the network-wide sensing radius \( 2r_2 \) is globally rigid, and \( r_{GR} \) is the minimum network sensing radius to be globally rigid. The average sensing radius \( r_2^2 \) is also higher than \( r_{GR} \) for all instances of networks. Third, it is clear that \( 3r_1 \) is higher than \( r_{GR} \). This is again expected since the network at the network-wide sensing radius \( 3r_1 \) is a trilateration network which is globally rigid, and \( r_{GR} \) is the minimum network sensing radius to be globally rigid. Fourth, we observe that in general, the sensing radius to achieve trilateration (i.e. \( 3r_1 \) and \( r_2^2 \)) is higher than that to be localizable using the algorithm in Fig. 4.

Figure 7 normalizes the sensing radii with respect to \( r_{GR} \). We observe that to be localizable using the algorithm in Fig. 4, the sensing radius is on average 1.48 times for \( r_2^2 \) and 1.75 times for \( 2r_2 \) of the global rigidity radius. To be able to trilaterate, the sensing radius is on average 1.78 times for \( r_1^3 \) and 2.3 times for \( 3r_1 \) of the global rigidity radius. Although the sensing radii for these algorithms are higher than the minimum global rigidity radius, as we will see below, using the minimum radius may cause many nodes to be uniquely localizable only in theory but not in known algorithms.

To start, we first show the number of nodes that are uniquely localizable at \( r_1 \) and \( r_2 \). The result is shown Fig. 8. We observe that although there are instances in which a large number of nodes are uniquely localizable at \( r_2 \), there are also instances in which a large number of nodes cannot be uniquely localized even in theory. Even when a node can be uniquely localizable in theory, it may not be localizable by an efficient algorithm. Fig. 9 shows the number of uniquely localizable nodes that can be localized by the algorithm in Fig. 4. We observe that the algorithm can localize a large number of nodes in most instances. However, even at \( r_{GR} \), there are instances (e.g., instance 6) in which many nodes...
are localizable in theory but not by the algorithm. We have also conducted experiments with 200 nodes, and observed instances in which even at $r_{GR}$ less than 5% of the nodes can be localized by the algorithm in Fig. 4. This shows the importance of topology control for easily localizable networks. Figure 10 shows the number of nodes which can be localized by trilateration. Again, we observe large variations in different test cases. At radius $r_{GR}$, although there are many instances in which above 90% of the nodes can be localized by trilateration, there are also many instances in which less than 20% of the nodes can be trilaterated. This again shows the importance of applying graph-theoretic techniques to construct networks for easily localizable networks.

6 Conclusions

This paper has shown how, by an operation that can be likened to increasing the sensing radius of the sensors in a sensor network, the localization problem can be made solvable, and indeed solvable in linear time, if the network has certain limited connectivity properties before any adjustment of the sensing radius.

Certain variants on the ideas can easily be contemplated. It is known for example that in $\mathbb{R}^2$, six-connectivity of a graph guarantees generic global rigidity, [31]. Whether or not such a graph is a trilateration graph is not known to the authors, or whether it could be made a trilateration graph by a simpler maneuver than that contemplated in this paper is not known. Again, it is possible to contemplate sensors with directional properties. Suppose each sensor in a two-dimensional sensor network is guaranteed to have one neighbour in every 120 degree sector (or 90 or 60 degree sector). Could one expect the trilateration property?

One can also envisage that sensors are laid down by a random process, as discussed for example in [6]. Consider for example a graph obtained by laying down $n$ sensors in a unit area, independently and with uniform distribution. Suppose the sensing radius is $r$ for all sensors. If $r = (1/\sqrt{\pi} + \epsilon)\sqrt{\log n}/n$ for arbitrary positive $\epsilon$ and all $n$, so that $r$ is allowed to depend on $n$, then as $n \to \infty$ the graph is connected, see [39]. Thus with probability very close to 1, and with a large value of $n$, trilateration will be achieved when $r$ is chosen to exceed the lower threshold of $3(1/\sqrt{\pi})\sqrt{\log n}/n$. One could still contemplate graphs however where the trilateration property 'just' failed; one might suspect that such graphs could at least still be globally rigid, with parts of them in trilateration 'islands', linked by a certain number of edges. If the number of islands is small, one might conjecture that the computational complexity of localizing such a globally rigid graph could be exponential in the number of islands, but not the number of vertices.

As noted in Section 1, from the graph theory point of view, instead of increasing the sensing radius $r$ to acquire some graphical property, one can increase the areal density of sensors; this is particularly apparent in the random case.

References


© Springer


Brian Anderson is a Distinguished Professor at the Research School of Information Sciences and Engineering, The Australian National University, Australia. Professor Anderson took his undergraduate degrees in Mathematics and Electrical Engineering at Sydney University, and his doctoral degree in Electrical Engineering at Stanford University. He worked in industry in the United States and at Stanford University before serving as Professor of Electrical Engineering at the University of Newcastle, Australia from 1967 through 1981. At that time, he took up a post as Professor and Head of the Department of Systems Engineering at the Australian National University in Canberra, where he was Director of the Research School of Information Sciences and Engineering from 1994 to 2002. For
approximately one year to May 2003, he was the inaugural CEO of the newly formed National ICT Australia, established by the Australian Government through the Department of Communications, Information Technology and the Arts and the Australian Research Council under the Information and Communication Technologies Centre of Excellence program. Professor Anderson has served as a member of a number of government bodies, including the Australian Science and Technology Council and the Prime Minister’s Science, Engineering and Innovation Council. He was a member of the Board of Cochlear Limited, the world’s major supplier of cochlear implants from its listing until 2005. He is a Fellow of the Australian Academy of Science and Academy of Technological Sciences and Engineering, the Institute of Electrical and Electronic Engineers, and an Honorary Fellow of the Institution of Engineers, Australia. In 1989, he became a Fellow of the Royal Society, London, and in 2002 a Foreign Associate of the US National Academy of Engineering. He holds honorary doctorates of the Catholic University of Louvain in Belgium, the Swiss Federal Institute of Technology, and the Universities of Sydney, Melbourne and New South Wales. He was appointed an Officer of the Order of Australia in 1993. He was President of the International Federation of Automatic Control for the triennium 1990 to 1993, and served as President of the Australian Academy of Science for four years from 1998 to 2002. Professor Anderson became the Chief Scientist of National ICT Australia in May 2001 and served in that role till September 2006.

Walter Whiteley (B.Sc. 66, Queen’s University at Kingston, Canada) received his Ph.D. in Mathematics from MIT, Cambridge Mass in 1971. He is currently the Director of Applied Mathematics at York University, and a member of the graduate programs in Mathematics, in Computer Science, and in Education. His research focuses on the rigidity and flexibility of systems of geometric constraints (distances, angles, directions, projections, ...). Recent work has included applications of this theory to location in networks, control of formations of autonomous agents, built structures in structural engineering, linkages in mechanical engineering, geometric constraints in computational geometry and CAD, and algorithms for protein flexibility in biochemistry. He is also active in geometry education and development of visual reasoning at all levels of mathematics education and in applications of mathematics.

A. Stephen Morse was born in Mt. Vernon, New York. He received a B.S.E.E degree from Cornell University, MS degree from the University of Arizona, and a Ph.D. degree from Purdue University. From 1967 to 1970 he was associated with the Office of Control Theory and Application OCTA at the NASA Electronics Research Center in Cambridge, Mass. Since 1970 he has been with Yale University where he is presently the Dudley Professor of Engineering and a Professor of Computer Science. His main interest is in system theory and he has done research in network synthesis, optimal control, multivariable control, adaptive control, urban transportation, vision-based control, hybrid and nonlinear systems, sensor networks, and coordination and control of large groups of mobile autonomous agents. He is a Fellow of the IEEE, a Distinguished Lecturer of the IEEE Control System Society, and a co-recipient of the Society’s 1993 and 2005 George S. Axelby Outstanding Paper Awards. He has twice received the American Automatic Control Council’s Best Paper Award and is a co-recipient of the Automatica Theory/Methodology Prize. He is the 1999 recipient of the IEEE Technical Field Award for Control Systems. He is a member of the National Academy of Engineering and the Connecticut Academy of Science and Engineering.

Yang Richard Yang received the B.E. degree in Computer Science and Technology from Tsinghua University, Beijing, China, in 1993, and the M.S. and Ph.D. degrees in Computer Science from the University of Texas at Austin in 1998 and 2001, respectively. Since 2001, he has been with the Department of Computer Science, Yale University, New Haven, CT, where currently he is an Associate Professor of Computer Science and Electrical Engineering. His current research interests are in computer networks, mobile computing, and sensor networks. He leads the Laboratory of Networked Systems (LANS) at Yale University.