A novel geometric approach towards a linear theory for sensor localization

Usman A. Khan†, Soummya Kar‡, and José M. F. Moura†

Abstract

Localization, finding the coordinates of a point with respect to other points with known coordinates—referred to as anchors, is essentially a nonlinear problem, since, for example, it involves solving equations of a circle when relating distances to Cartesian coordinates, or computing Cartesian coordinates from angles using the law of sines to find sensor-anchor distances. This paper describes a notable alternative, a linear-convex method, to the nonlinear localization problem. This linear approach is based on barycentric coordinates that correspond, on the plane, to ratios of areas of triangles. To compute these areas from the distances among the point and the anchors, Cayley-Menger determinants are employed. The aforementioned linear solution takes advantage of the structural convexity, found in many practical applications, that the point of interest falls inside the convex hull of the anchors. Under this structural condition, this paper describes how to use convexity, barycentric coordinates, and the Cayley-Menger determinant to localize by a distributed, linear, iterative algorithm, a set of sensors with respect to a minimal number of anchors. Specifically, in an $m$-dimensional Euclidean space, a set of $m + 1$ anchors is sufficient (and necessary) to localize an arbitrary collection of points (sensors) in the convex hull of the anchors; moreover, such a localization is achievable in a fully distributed manner with iterative inter-sensor communication over a sparse communication graph—in the absence of any centralized computing-communicating entity. The paper further explores natural extensions of the proposed linear solution to mobile sensors (agents) and to uncertainty in the intermittent wireless communication.

†UAK is with the Department of Electrical and Computer Engineering at Tufts University, 161 College Ave., Medford, MA 02155, khan@ece.tufts.edu.‡SK and JMFM are with the Department of Electrical and Computer Engineering at Carnegie Mellon University, 5000 Forbes Ave., Pittsburgh, PA, 15213, {soummyak,moura}@ece.cmu.edu. The work of the third author was partially supported by NSF grant CCF1011903.

This paper has supplementary downloadable material available at http://ieeexplore.ieee.org, provided by the authors. This includes four multimedia AVI format videos, which show the sensor deployments and location estimates in various contexts—appropriate footnotes: 12, 13, 14, 16, and 17, within the manuscript, describe these videos.
I. INTRODUCTION

Localization is a term often referred to as finding the position of a point in a Euclidean space, \( \mathbb{R}^m \), given a certain number of anchors\(^1\) and some measurements, for example, distances and/or angles, between the point and the anchors. Localization is essentially a nonlinear problem that requires either: (i) solving circle equations when the sensor-anchor distances are given; or, (ii) using law of sines to find the corresponding sensor-anchor distances (and then solving circle equations) when the sensor-anchor angles are given. This paper describes a linear solution to the nonlinear localization problem, which exploits a structural convexity condition, relevant to most practical situations, i.e., the points to be localized lie in a convex set and anchors are placed on the corners of this convex set. As an example of this structural condition, consider the points to be localized as objects (packages/assets) in a facility (factory, warehouse, or cell), where the objects remain in the confinement of the facility—the anchors are placed at the appropriate corners of the corresponding facility. In such scenarios, it is quite straightforward to visualize the deployment, and the fact that the objects always lie in the convex hull of the anchors.

Key to our linear solution is to exploit the aforementioned structural condition. Taking advantage of the resulting convexity in the deployment, we reexpress the localization problem in terms of barycentric coordinates that represent a point with unknown location in \( \mathbb{R}^m \) as a linear-convex combination of exactly \( m + 1 \) anchors. To see the power and significance of this representation, recall that a point, \( c_2 \), in an interval of the real line, \([c_1, c_3]\), can be written as a linear-convex combination of the end points \( c_1 \) and \( c_3 \) of the interval as

\[
c_2 = \frac{d_{23}}{d_{13}} c_1 + \frac{d_{21}}{d_{13}} c_3,
\]

where: \( d_{ij} = |c_i - c_j|; c_2, c_1 < c_3 \in \mathbb{R}; \) and \([c_1, c_3]\) is the convex hull\(^2\) of \( c_1 \) and \( c_3 \). Likewise, on the plane, any point \( c_2 \in \mathbb{R}^2 \) can be written as a linear-convex combination,

\[
c_2 = a_1 c_1 + a_3 c_3 + a_4 c_4,
\]

of now three points, \( c_1, c_3, c_4 \in \mathbb{R}^2 \), when \( c_2 \) lies inside the triangle defined by \( c_1, c_3, c_4 \), and we choose the coefficients, \( a_1, a_3, a_4 \), as the ratios of areas of corresponding triangles, details in the next section. Of course, a triangle is the convex hull in \( \mathbb{R}^2 \) of three non-collinear points. This convex-linear representation extends to arbitrary Euclidean spaces \( \mathbb{R}^m \), i.e., any point \( c \in \mathbb{R}^m \) can be written as a linear-convex combination of exactly \( m + 1 \) points in \( \mathbb{R}^m \), provided that \( c \) lies inside the corresponding

---

1 Anchors, or sometimes also referred to as beacons, are such that their exact location is known.

2 Note that \( c_1 < c_3 \) implies that the measure of the convex hull, i.e., \( |c_1 - c_3| \), is strictly positive.
convex hull. The convex hull in $\mathbb{R}^m$ is the $m$-simplex (e.g., tetrahedron in $\mathbb{R}^3$) and the coefficients come from the ratios of hypervolumes—generalization of area (measure in $\mathbb{R}^2$) and volume (measure in $\mathbb{R}^3$) to arbitrary $\mathbb{R}^m$. What is also very significant is that there is an explicit straightforward way to compute these hypervolumes in arbitrary Euclidean spaces only with distance information, called the Cayley-Menger determinant.

The linear-convex representation with barycentric-coordinates is specially useful when locating an arbitrary number of sensors in $\mathbb{R}^m$ with respect to $m+1$ anchors. We describe a linear iterative distributed algorithm to localize an arbitrary number of sensors in a given network with only local communication, i.e., sensors have direct (single-hop) communication with only their immediate neighbors, and there is no fusion center in the network. As we will detail, in most practical situations of interest, only a very few sensors are able to communicate directly with at most one anchor, and most sensors communicate with no anchor at all. In this stringent setting, the distributed localization algorithm we describe expresses the coordinates of each sensor as a linear-convex combination of the state of $m+1$ carefully chosen neighbors; each sensor starts with a random guess of its location and continues to update its location using the linear-convex representation, recursively. Over a few iterations of this recursive procedure—the convergence rate to the true sensor locations is in fact geometric, the state of each sensor converges to the sensor’s true location. The linearity of the proposed localization approach and its innovative extension to a linear iterative process was we believe presented for the first time in [1].

Below we provide a brief historical account on the evolution of localization and an overview of the traditional approaches.

**Localization: A brief history**—Traditionally, localization based on distance measurements has been referred to as trilateration, whereas angle-based methods are referred to as triangulation. Trilateration is the process of finding the location of a sensor in $\mathbb{R}^m$, given only the distance measurements to a certain number of anchors. It can be easily verified that at lest $m+1$ anchors are necessary to find the unknown location without ambiguity, see Fig. 1 (Left) for a sketch of this problem in $\mathbb{R}^2$ where three anchors are needed. With $m+1$ sensor-to-anchor distances, the nonlinear trilateration problem is to find the intersection of three circles centered at each anchor with radius as the corresponding sensor-to-anchor distance. Triangulation, on the other hand, employs the angular measurements based on

---

3Because the algorithm is iterative, each sensor has a ‘current’ guess of its coordinates; this current guess is the state of the sensor that is iteratively updated by the distributed localization algorithm.

4Hereinafter, the term sensor refers to a point with unknown location.
some fundamental trigonometric concepts in order to resolve the location of a sensor with two known angles and one distance. Clearly, since the angles are non-absolute measurements, at least one distance measurement is required to introduce a scale to the coordinate-system, we show the triangulation setup in Fig. 1 (Right)–note that the procedure is still nonlinear.

![Diagram](Fig. 1. Anchors with known locations shown as red triangles, the unknown location is the blue filled circle. (Left) **Trilateration**–With the knowledge of three distances, $d_{c1}$, $d_{c2}$, and $d_{c3}$, the unknown location is at the intersection of three circles shown with dashed lines; the location marked as $\star$ is the additional solution if only two distances, $d_{c1}$ and $d_{c2}$, are known. (Right) **Triangulation**–With the knowledge of $d_{12}$, $\theta_{c1}$, and $\theta_{c2}$, all the dashed and dotted line segments, $h$, $d_{c1}$, $d_{c2}$, $d_{h1}$, and $d_{h2}$, can be computed; for example, $d_{c2} = \frac{d_{12} \sin \theta_{c1}}{\sin(\theta_{c1} + \theta_{c2})}$, $d_{c1} = \frac{d_{12} \sin \theta_{c2}}{\sin(\theta_{c1} + \theta_{c2})}$, which are the distances between $c$ and the anchors. The ambiguity (to which side of the line segment, $1 \leftrightarrow 2$, does $c$ lie) can be removed with a third anchor, or by pre-specifying that $c$ lies above $1 \leftrightarrow 2$, which could be the case if mapping points on earth.

The problems of **trilateration** and **triangulation** have been of significant interest for several thousand years with applications in mapping (land surveying) and celestial navigation. The history of localization, thus, can be divided into the progress made in these two areas (mapping and celestial navigation). Following is a brief historical perspective on the uses of trilateration and triangulation in related localization applications borrowing liberally from the References [2]–[9].

Mapping, or land-surveying, has been around for many centuries and dates back as far as the ancient Egyptians (3000 BC) and Indus civilization (2500 BC). The problem in mapping is to find the location of a set of points with respect to other points in order to generate a map. It is evident that the Great Pyramids, built around 2700 BC, demonstrates some familiarity regarding the mapping and surveying techniques. Recently, an archaeologist, Anthony Johnson, has assessed that Stonehenge (2500 BC) was built with peg and rope geometry used by surveyors [10]. Similar evidence can be found in historical documents on the use of surveying in Greek, Babylon, Mesopotamia, and Rome, among other ancient civilizations.
The initial technological advancement in modern-day mapping is tied to triangulation, see Fig. 1 (Right), which is attributed to Dutch mathematician and cartographer, Gemma Frisius (1508–1555), and Dutch astronomer and mathematician, Willebrord Snellius (1580–1626), see [4] for biographies and related historical accounts. Based on triangulation, much of the Europe was surveyed: (i) the French astronomer and cartographer, Cassini de Thury (1714–1784), built a complete map of France in 1747, and further provided insights into the Paris-Greenwich triangulation [2], [6], [7], [11]; and (ii) the work of Scottish engineer and surveyor, William Roy (1726–1790), led to an accurate geodetic mapping of Great Britain; the Ordnance Survey of Britain, Britain’s national mapping agency, was founded in 1791 partly as a consequence of a geodetic measurement led by William Roy and the French astronomer and cartographer, Jacques Dominique Cassini (1748–1845) son of Cassini de Thury, see [7] for a detailed historical account.

Another fundamental application where localization has deep historical roots is celestial navigation5. Before the invention of modern-day satellites, navigation was performed with respect to natural objects, for examples, celestial bodies (the sun, the moon, a planet or a star). It is believed, from Homer’s *Odyssey*, that primitive attempts to navigation using astronomical principles were made by Mediterranean seaman around 1000 BC. However, significant development in this area has been attributed to maritime explorers in Atlantic ocean lead by the Portuguese under the sponsorship of Prince Henry the Navigator (1394–1460), [5]. Portuguese mathematician, Pedro Núñez, is attributed to finding latitude from simultaneous star altitude observations using a celestial globe, [5].

The principle of celestial navigation is much the same as *trilateration* as described in Fig. 1 (Left), however, the method has to be adjusted for the motion of the vessel, time at which the measurements are taken, and the fact that the distances may not be straight lines but, for example, measured along the geodesics. A slight difference is that instead of finding the distance between the vessel and the celestial body, the distance between the vessel and the point on earth directly below the celestial body—known as its geographic position (GP), is found; the location of the GP is then determined from the tables in the *The Nautical Almanac*6 of the corresponding year, see Fig. 2 for a pictorial representation. The angle from a vessel to a celestial body is measured using a navigational instrument, which is then converted to a distance between the vessel and the GP using a simple principle [12] that adjusts for Earth’s curved

5Several accounts are available on the history of navigation and a good starting point may be [8].

6A nautical almanac describes the positions of certain celestial bodies for the purpose of celestial navigation. The Nautical Almanac has been published annually in the United States of America by the US Naval Observatory since 1852.
surface. A circle on earth is now drawn where the vessel may be located given this distance. Adding another celestial object results into two circles whose one of the two intersections is the actual vessel position; a third celestial object removes this ambiguity.

![Celestial Navigation Diagram](image)

**Fig. 2.** Celestial navigation: An observer, sea vessel, measures its angle, $\theta_o$, to the sun with, for example, a sextant. The sun’s geographical position (GP), the point on earth directly below the sun, is determined for the time and day from, for example, The Nautical Almanac. The GP and the observed angle, $\theta_o$, is then converted to a distance between the observer and GP after accounting for the Earth’s curved surface, which leads to the circle centered at GP. Clearly, adding two more celestial bodies gives the location using trilateration, see Fig. 1 (Left).

Progress in celestial navigation is attributed to an American Captain from Boston, Thomas Hubbard Sumner (1807–1876), during a North Atlantic voyage in 1837 [5], whose work can be found in [13]. Forty years later, a more general method was developed French navigator, Marcq Saint Hilaire, in 1875, known as the Altitude-Intercept method [14], [15]. It is worth mentioning that the precision of any celestial navigation method heavily depends on the accuracy of the navigational instrument used for the angle measurement. This instrumentation accuracy has led to the creation of increasingly accurate measurements, details can be found in [9] among others: kamal–associated to Arab navigators in the 9th century; astrolabe–variations can be found as early as in the Hellenistic world in 150 BC and through the Byzantine period; octant–attributed to the principle of doubly reflecting navigation invented by Sir Isaac Newton (1643–1727), the invention is attributed to an English mathematician, John Hadley (1682–1744), and to a Philadelphia glazier, Thomas Godfrey (1704-1749); and sextant–invented in 1757 by the English mathematical instrument maker, John Bird (1709–1776).

**Modern-day localization**–In recent years, advances in two particular areas, namely, satellites and wireless communication, have facilitated a tremendous growth in the applicability and wide-spread use of localization. With the help of satellites, man-made objects orbiting the earth, the celestial navigation has been replaced by what is referred to as satellite navigation–Global Positioning System (GPS) is a prime example [16]. Clearly, the availability, accuracy, and timing of the satellites can be precisely
controlled as opposed to those of the celestial objects. The principle of GPS-based navigation is based heavily on the timing of the wireless signals coming from the satellites, see [16] for more details.

Wireless communication has further enabled the use of localization technology in indoor (household and industrial) applications, where satellites (or a celestial body) are not available as the open sky is not visible. The distance and angle measurements for trilateration and triangulation have been replaced by related measurements of wireless signals, namely, Received Signal Strength (RSS), Time-of-Arrival (ToA), and Angle-of-Arrival (AoA), see [17] for details on these measurements. Due to wireless communication, it is now possible to conceive localization applications indoors, for example, assets in an industrial or a cargo facility, indoor moving robots [18], and objects in buildings, offices, or hospitals. Similarly, localization of randomly deployed sensors, robots in a remote area, and military units in a battleground where satellite signals may be jammed is further possible.

Wireless sensor networks and multi-agent systems have further lead to localization research that is completely distributed. For example, instead of locating a single sensor–(point with unknown location), in $\mathbb{R}^m$, in the presence of at least $m + 1$ anchors, consider now that there are several sensors, $M >> 1$, whose locations are to be determined. In a Wireless Sensor Network (WSN), the sensors can be deployed randomly over a region of interest, with the ability to monitor certain phenomena in their vicinity. Localization of the sensors is thus an important step in order to infer meaningful information from the data collected by the sensor network. However, it may not be possible for the entire sensor network to transmit the localization-pertinent measurements to a central location, where trilateration or triangulation may be implemented, nor it may be practical for every sensor to have direct access to the anchor measurements. It is, thus, of significant practical value to study and develop algorithms where the sensors are able to exchange information only locally (with some neighboring sensors) in order to find their locations in a completely distributed (decentralized) manner.

**Brief review of the literature**—The literature on modern-day localization is heavily based on the traditional triangulation and trilateration principles, or in some cases, a combination of both. Recent work may also be broadly characterized into centralized and distributed algorithms\(^7\), see [19] where a comprehensive coverage of cooperative and noncooperative strategies is provided.

Illustrative centralized localization algorithms include: maximum likelihood estimators that are formulated when the data is known to be described by a statistical model, [20]–[22]; multi-dimensional

---

\(^7\)In the remainder of this paper, we will use *sensor* to denote a node whose location is to be determined, and *anchor* as the node whose location is known a priori.
scaling (MDS) algorithms that formulate the localization problem as a least squares problem at a centralized location, [23], [24]; work that exploits the geometry of the Euclidean space, like when locating a single robot using trilateration in $m = 3$-dimensional space, see [25] where a geometric interpretation is given to the traditional algebraic distance constraint equations; localization algorithms with imprecise distance information, see [26] where the authors exploit the geometric relations among the distances in the optimization procedure; for additional work, see, e.g., [27]–[31]. Optimization based techniques can be found in [32], [33] and references therein, whereas, polynomial methods for localization are described in [34]. In short, centralized algorithms are fine in small or tethered network environments; but in large untethered networks, they incur high communication cost and may not be scalable; they depend on the availability and robustness of a central processor and have a single point of failure.

Distributed localization algorithms can be characterized into two classes: multilateration and successive refinements. In multilateration algorithms, [35]–[38], each sensor estimates its distance from the anchors and then calculates its location via trilateration, [39]–the term multilateration implies that the distance computation may require a multi-hop communication from the sensor to the anchor. The location estimates obtained from multilateration schemes are subject to large errors because the estimated sensor-anchor distance in large networks, where the anchors are far apart, is noisy. To overcome this problem, a high density of anchors is required. A distributed multidimensional scaling algorithm is presented in [40]. Successive refinement algorithms that perform an iterative minimization of a cost function are presented in [41]–[43]. Reference [41] discusses an iterative scheme where they assume 5% of the nodes as anchors. Reference [43] discusses a Self-Positioning Algorithm (SPA) that provides a GPS-free positioning and builds a relative coordinate system.

Localization literature also consists of a large body of work based on graph theory [44]. Reference [45] discusses the graphical properties of the underlying communication network and considers cases where sequential localization [46] is possible. Reference [47] provides a brief discussion on the relevant graphical concepts on congruency, equivalence, rigidity, and global rigidity; for more details additional references within [47] can be found. Another direction is the completion of partially specified distance matrices, considered in [48], [49]; the algorithms complete the unspecified distances under the geometrical constraints of the underlying network. Both of the above approaches are relevant when the (entire) network communication graph or the (entire) partially specified distance matrix are available at a central location.

Another formulation to solve localization problems in a distributed fashion is the probabilistic approach. Nonparametric belief propagation on graphical models and sequential Monte Carlo methods for mobile localization are considered in [50] and [51], respectively. Particle filtering is studied in [52], where each
sensor stores respective particles for its location that are weighted according to a likelihood. Probabilistic methods are also employed to track and locate mobile robots, see [53] for details. Work on scarce distance information can be found in [54] and references therein.

In the context of localization for mobile networks, reference [55] discusses a tracking algorithm that tracks objects using wireless devices sensing the objects. The wireless devices already know their locations and tracking is achieved by aggregating the sensed information. Another interesting tracking algorithm in [56] tracks humans on a tiled floor by using pressure sensors to study human gait patterns. Reference [57] uses trilateration to solve the localization/tracking problem that requires a large number of close by anchors to have a reasonable location estimate. Some other relevant references in this direction include [58]–[61].

In this paper, we describe D1stributed LOCalization (DILOC) [1]–an algorithm that uses a convexity argument to express the nonlinear localization problem, i.e., trilateration and triangulation shown in Fig. 1, as a linear equation. The convexity assumption we place on the sensor deployment is that all of the sensors (with unknown locations) lie in the convex hull of the anchors. This assumption exploits the convexity structure in localization often encountered in several practical scenarios, i.e., the points to be localized lie in the convex hull of the available anchors. This particular structure enables us to switch from the common Cartesian or polar coordinates (in $\mathbb{R}^m$) to a barycentric-coordinate representation whereby we use the barycentric coordinates [62], [63] to write the coordinates of an unknown location as a linear-convex combination of the anchors’ coordinates. This linear representation leads to an iterative localization algorithm. Since the iterations are linear-convex, they converge to a unique fixed-point regardless of the initial conditions. We show that the fixed-point is indeed the true sensor locations.

Convergence of DILOC is closely related to the convergence of absorbing Markov chains–we apply the absorbing and transient states arguments to show the convergence and the existence of a fixed-point of DILOC. It is noteworthy that DILOC only requires $m+1$ anchors even if most of the sensors are unable to communicate to any anchor. If more than the minimum $m+1$ anchors are available, the algorithm uses of course the additional anchors. A simple straightforward strategy is to divide the sensor deployment into the concatenation of convex polygons where anchors are placed at the corners of a polygon. However, a more general framework, where no such concatenation is employed, is possible, see Section V-B for details. This paper further discusses several generalizations of DILOC, extending it to dynamic network topologies, uncertain wireless environments, and mobile networks [1], [64]–[68].

---

8 A convex hull of a set of points, $\kappa \in \mathbb{R}^m$, is the intersection of all of the convex sets containing $\kappa$. 
We now describe the rest of the paper. Section II presents the typical nonlinear sensor localization problem and contrasts it with our linear-convex approach. Section III formally describes the DIstributed LOCalization (DILOC) algorithm with an illustrative example. Section IV provides analysis and insights into the convergence and related characteristics of DILOC. In Section V, we cast several generalizations to DILOC and provide a brief discussion and results on several variations and extensions to DILOC in a deterministic and uncertain framework. Finally, Section VI concludes the paper. Illustrations and examples are provided throughout the text and several historical remarks and insights have been added during the exposition.

II. SENSOR LOCALIZATION: BARYCENTRIC COORDINATES

Before presenting the distributed sensor localization algorithm, we consider the simple problem of locating one sensor, \( c \in \mathbb{R}^2 \), given three anchors, 1, 2, 3, whose locations are known precisely, see Fig. 1 (Left). The sensor is to compute its location from the knowledge of the three distances, \( d_{c1}, d_{c2}, d_{c3} \), to the anchors. Assume for now that only one distance, \( d_{c1} \), is known to the sensor; with this information alone, the sensor’s location can be anywhere on the circle centered at anchor 1 with radius \( d_{c1} \). When two distances are known, e.g., \( d_{c1} \) and \( d_{c2} \), the sensor’s location is still ambiguous since there are two possible choices given by the intersection of two circles, the aforementioned circle and the circle centered at anchor 2 with radius \( d_{c2} \); the two possible locations for \( c \) are shown as \( \star \) and the dark filled circle in Fig. 1 (Left). This ambiguity is only resolved when three distances are known and the true location becomes the intersection of the three circles shown as the dark filled circle in Fig. 1 (Left).

It can be easily verified that the above setup requires the simultaneous solution of three nonlinear (circle) equations. However, what is more challenging is that the above procedure is only applicable to a network of sensors when it is applied serially. In other words, a first sensor finds three anchors to determine its location; it then becomes an anchor. A second sensor chooses three out of the four currently available anchors resulting now into a total of five anchors. The process continues till all of the sensors locate themselves, see [46] for details on sequential localization. If, in a network, no node can communicate with any three of the currently available anchors (in order to find the sensor-anchor distances), the above procedure cannot continue, and thus fails. Attempting to develop an iterative implementation of the procedure described in Fig. 1 (Left) would not be successful since the iterations, being nonlinear, are sensitive to initial conditions, and convergence is not guaranteed.
A. A convex approach

In contrast to the nonlinear procedure of solving three circle equations simultaneously, we propose an alternate approach to localization. We use the barycentric coordinates of sensor \( c \) with respect to the anchors and exploit the fact that this representation is unique [1], [69], provided that the sensor lies inside the convex hull of the anchors. Convexity, however, is not required in the nonlinear version of the problem and, no matter where the sensor lies, its location can be found, without ambiguity, as the solution of the underlying \((m+1)\) nonlinear equations. In contrast, to guarantee a unique location when we employ the barycentric representation, we need convexity.

![Fig. 3. (Left) Barycentric representation of \( c \in \mathbb{R}^2 \) shown by the blue filled circle, with \( m+1 = 3 \) anchors shown by red triangles. (Middle and Right) Convex Hull Inclusion Test in \( \mathbb{R}^3 \) with \( m+1 = 4 \) anchors: The sensor \( l \) is shown by a blue filled circle, whereas, the anchors in \( \kappa \) are shown by red triangles. (Middle) \( l \in \mathcal{C}(\kappa) \Rightarrow A_\kappa = A_{\kappa \cup \{l\}} \), (Right) \( l \notin \mathcal{C}(\kappa) \Rightarrow A_\kappa < A_{\kappa \cup \{l\}} \).](image)

**Remark 1.** The barycentric representation was introduced by August F. Möbius (1790-1868) well-known for the famous characterization of Möbius strips, see, for example, [70] and [71] for a detailed account and applications.

We explain the barycentric representation with the help of Fig. 3 (Left). Given that \( c \in \mathcal{C}(1,2,3) \), where \( \mathcal{C}(\cdot) \) denotes the convex hull of the points in the argument, the sensor with unknown location needs now to use six distances: the three distances, \( d_{c1}, d_{c2}, d_{c3} \), already introduced from the sensor to the anchors, and also the three inter-anchors distances, \( d_{12}, d_{23}, d_{13} \). From these six distances, the sensor, \( c \), can determine the areas, \( A_{c12}, A_{c13}, A_{c23} \), marked in Fig. 3 (Left) and also \( A_{123} \). Now the
unknown location of \( c \) is uniquely determined by the following equation,

\[
\begin{align*}
\mathbf{c} &= \frac{A_{c23}}{A_{123}} \mathbf{c}_1 + \frac{A_{c13}}{A_{123}} \mathbf{c}_2 + \frac{A_{c12}}{A_{123}} \mathbf{c}_3, \\
&\triangleq a_{c,1} \\
&\triangleq a_{c,2} \\
&\triangleq a_{c,3}
\end{align*}
\]

where \( \mathbf{c}_i, i = 1, 2, 3 \) are the (row) vectors of the x- and y-coordinate of the anchors, and \( a_{c,i} \) are defined as the barycentric coordinates of sensor \( c \) with respect to the neighboring node \( i \). It can be immediately realized that: (i) the barycentric representation is decoupled in the coordinates, i.e., the equation for the x-coordinate is decoupled from the equation for the y-coordinate in Eq. (1); (ii) the representation of a sensor’s location with respect to the three corresponding neighbors is unique [1], [69], i.e., distinct points inside the convex hull of the three given points cannot lead to the same barycentric coordinates, since the numerators change while the denominator is constant; (iii) the representation is linear; and (iv) the representation is convex: \( a_{c,i} \geq 0, a_{c1} + a_{c2} + a_{c3} = 1 \), with the inequality being strict when the sensor does not lie on any line connecting two anchors. Furthermore, since the computation of the barycentric coordinates involves a division by \( A_{123} \), it is required that \( A_{123} \neq 0 \). This is guaranteed by a non-degeneracy condition that all of the anchors do not lie on a straight line (low-dimensional subspace) in \( \mathbb{R}^2 \). The barycentric representation, being linear-convex, leads to well-behaved iterative algorithms because if a linear iterative algorithm converges, its fixed-point is unique. Furthermore, the fixed-point of linear iterations is independent of the initial conditions.

The barycentric representation extends to arbitrary Euclidean spaces, \( \mathbb{R}^m \). For example, in \( \mathbb{R}^3 \), the convex hull becomes the tetrahedron and the areas are volumes. In general, the convex hulls are \( m \)-simplices and the corresponding volume is the \( m \)-dimensional hypervolume\(^9\). Of course, for sensor localization, we may have \( m = 1 \) (sensors in a straight line), \( m = 2 \) (plane), or \( m = 3 \) (3d-space). The generic case \( m > 3 \) is of interest, for example, when the graph nodes represent \( m \)-dimensional feature vectors in classification problems, and the goal is to find in a distributed fashion their global coordinates (with respect to a reference frame.) Since our results are general, we deal with \( m \)-dimensional ‘localization,’ but, for easier accessibility, \( m = 2 \) or \( m = 3 \) may be considered.

To carry out the above iterative, linear-convex process, we need to accomplish the following: (i) given the inter-node distances in \( \mathbb{R}^m \), compute the corresponding hypervolumes (area when \( m = 2 \), or volume when \( m = 3 \)); and (ii) given \( m + 1 \) arbitrarily chosen neighbors, decide that if a sensor lies inside or outside their convex hull. We explain these below.

\(^9\) A 2-simplex is a triangle, a 3-simplex is a tetrahedron, and so on. Similarly, the hypervolume in \( \mathbb{R}^2 \) is area, the hypervolume in \( \mathbb{R}^3 \) is volume, and so on.
B. The Cayley-Menger determinant

The Cayley-Menger determinant is a very useful mathematical construct [72]–[76]. It is the determinant of an \((m+2)\times (m+2)\) symmetric matrix that relates to the hypervolume, \(A_{\Theta_l}\), of the convex hull, \(C(\Theta_l)\), of \(m+1\) points in \(\Theta_l \in \mathbb{R}^m\), via an integer sequence, \(s_{m+1}\). Let \(1_{m+1}\) denote an \(m+1\)-dimensional column vector of 1s. The Cayley-Menger determinant is given by

\[
A_{\Theta_l}^2 = \frac{1}{s_{m+1}} \begin{vmatrix}
0 & 1_{m+1}^T \\
1_{m+1} & Y
\end{vmatrix},
\]

where \(Y = \{d_{lj}^2\}, l, j \in \Theta_l\), is the matrix of squared distances, \(d_{lj}\), among the \(m+1\) points in \(\Theta_l\) and

\[
s_m = \frac{2^m (m!)^2}{(-1)^{m+1}}, \quad m = \{0, 1, 2, \ldots\}.
\]

The first few coefficients of the integer sequence \(s_m\), are \(-1, 2, -16, 288, -9216, 460800, \ldots\). Although the elements of this sequence grow large very rapidly, in most applications we are only interested in the first few terms.

Remark 2. The Cayley-Menger determinant is named after the British mathematician, Arthur Cayley (1821-1895), and the Austrian-American mathematician, Karl Menger (1902-1985). A simpler form of the corresponding determinant in Eq. (2) is associated to Lagrange, whereas the general form was established by A. Cayley in his first published paper [72] and dates back to year 1841. Combined with Menger’s seminal work on Euclidean geometry [77] in year 1931–characterization of the Euclidean metric among all semi-metrics, the relation in Eq. (2) is referred to as the Cayley-Menger determinant. See [73] for a brief historical account and other related references to this topic.

Remark 3. The sequence, \(s_m\), in the Cayley-Menger determinant given in Eq. (3) is available as integer sequence number A055546 in The online encyclopedia of integer sequences compiled by N. J. A. Sloane.

C. Convex hull inclusion test

Per Fig. 3 (Left), Eqs. (1) and (2) give the barycentric representation of a point, \(l\), with respect to the given points, \(\{1, 2, 3\}\), when the original point, \(l\), is in the convex hull of the points \(\{1, 2, 3\}\). We now give an algorithm to test if a sensor, \(l \in \mathbb{R}^m\), lies in the convex hull of \(m+1\) nodes in a set, \(\kappa\), using only the distance information among these \(m+2\) nodes \((\kappa \cup \{l\})\) and the Cayley-Menger determinant. Let \(C(\kappa)\) denote the convex hull formed by the nodes in \(\kappa\). Clearly, if \(l \in C(\kappa)\), the convex hull formed by the nodes in \(\kappa\) is the same as the convex hull formed by the nodes in \(\kappa \cup \{l\}\), i.e.,

\[
C(\kappa) = C(\kappa \cup \{l\}), \quad \text{if } l \in C(\kappa),
\]

March 31, 2013 DRAFT
see Fig. 3 (Middle) on this test in \( \mathbb{R}^3 \). This equation says that, if \( l \in C(\kappa) \), then the hypervolumes of the two convex sets, \( C(\kappa) \) and \( C(\kappa \cup \{ l \}) \), are equal. Let \( A_\kappa \) denote the hypervolume of \( C(\kappa) \) and let \( A_{\kappa \cup \{ l \}} \) denote the hypervolume of \( C(\kappa \cup \{ l \}) \), we have

\[
A_\kappa = A_{\kappa \cup \{ l \}} = \sum_{k \in \kappa} A_{\kappa \cup \{ l \} \setminus \{ k \}} \quad \text{if} \ l \in C(\kappa). \tag{5}
\]

Hence, the test becomes

\[
\begin{align*}
& l \in C(\kappa), \quad \text{if} \ \sum_{k \in \kappa} A_{\kappa \cup \{ l \} \setminus \{ k \}} = A_\kappa, \tag{6} \\
& l \notin C(\kappa), \quad \text{if} \ \sum_{k \in \kappa} A_{\kappa \cup \{ l \} \setminus \{ k \}} > A_\kappa. \tag{7}
\end{align*}
\]

This is shown in Figure 3 (Middle and Right). This inclusion test is based entirely on the hypervolumes, which can be calculated using only the distance information in the Cayley-Menger determinant.

**Remark 4.** In \( \mathbb{R}^2 \), a simple test that uses the angles subtended by \( c \) and each of the two anchors may test the convex hull inclusion. This test can be explained in terms of Fig. 3 (Left), and it simply states that the three angles, \( \angle (c_{12}), \angle (c_{13}), \angle (c_{23}) \), sum to \( 2\pi \) if \( c \in C(1, 2, 3) \). This angle test, however, does not have a straightforward generalization in higher dimensions. The test given in Section II-C was introduced in [1], based only on distances and the Cayley-Menger determinant that further applies to arbitrary Euclidean spaces, \( \mathbb{R}^m \).

### III. DISTRIBUTED SENSOR LOCALIZATION

We now describe the Distributed sensor LOCALizaion (DILOC) algorithm [1] in \( \mathbb{R}^m (m \geq 1) \), an iterative, linear-convex algorithm that uses the barycentric coordinates and the Cayley-Menger determinant, presented in the previous section. To this aim, let \( \Theta \) be the set of all of the nodes in the network decomposed as \( \Theta = \kappa \cup \Omega \), where \( \kappa \) is the set of anchors, i.e., the nodes whose locations in \( \mathbb{R}^m \) are known, and \( \Omega \) is the set of sensors whose locations are to be determined. By \( |\cdot| \), we mean the cardinality of the set in the argument, and we let \( |\Theta| = N \), \( |\kappa| = m + 1 \), and \( |\Omega| = M \), i.e., \( N = m + 1 + M \).

For a set \( \Psi \subseteq \Theta \) of nodes, we denote its convex hull\(^{10} \) by \( C(\Psi) \). For example, if \( \Psi \) is a set of three non-collinear nodes in a plane, then \( C(\Psi) \) is a triangle. Let \( A_\Psi \) be the hypervolume (area when \( m = 2 \), volume when \( m = 3 \), and their generalization in higher dimensions) of \( C(\Psi) \). Let \( d_{lk} \) be the distance between two nodes \( l, k \in \Theta \); the neighborhood of node \( l \) in a given radius, \( r_l \), is

\[
\mathcal{K}(l, r_l) = \{ k \in \Theta : d_{lk} < r_l \}. \tag{8}
\]

\(^{10}\)The convex hull, \( C(\Psi) \), of a set of points in \( \Psi \) is the minimal convex set containing \( \Psi \).
We denote by $c_l \in \mathbb{R}^m$, the $m$-dimensional coordinate (row) vector for node $l \in \Theta$, with respect to a global coordinate system, written as

$$c_l = [c_{l,1}, c_{l,2}, \ldots, c_{l,m}]. \quad (9)$$

The true (possibly unknown) location of node $l$ is represented by the row vector, $c^*_l$. Because the distributed localization algorithm DILOC is iterative, the row vector at time $t$, $c_l(t)$, represents the location estimates, or state, for node $l$ at iteration $t$. We make the following assumptions.

**B0** Convexity. All the sensors lie inside the convex hull of the anchors $C(\Omega) \subseteq C(\kappa)$.

**B1** Anchor nodes. The anchors’ locations, $c_q^*, q \in \kappa$, are known.

**B2** Non-degeneracy. The hypervolume for the set of anchors, $\kappa$, is non-zero, i.e., $A_\kappa \neq 0$.

**B3** Known distances. The distances among any pair of nodes in $\{l\} \cup K(l,r_l)$, $\forall \ l \in \Omega$ are known.

**Remark 5.** Assumption B2 simply states that the anchors do not lie on a hyperplane. If this were the case, the localization problem reduces to a lower dimensional problem, i.e., locating the sensors in $\mathbb{R}^{m-1}$ instead of $\mathbb{R}^m$. For instance, if all of the anchors lie on a plane in $\mathbb{R}^3$, by B0, the sensors also lie in the same plane, and the localization problem can be thought of as localization in $\mathbb{R}^2$.

**Remark 6.** Assumption B3 can also be stated as the existence of radio links among all of the nodes in the set $\{l\} \cup K(l,r_l)$, $\forall \ l \in \Omega$. If $l$ and $j$ have a radio link, $l$ and $j$ can both estimate the inter-node distance, $d_{lj}$, between them. This distance can be found from the Received Signal Strength (RSS), Time of Arrival (ToA), or Angle of Arrival (AoA), of the radio signals, see [17] for details.

From B0, the next Lemma follows easily.

**Lemma 1.** Under B0, for every sensor $l \in \Omega$, there exists a non-degenerate set, $\Theta_l(r_l)$, containing exactly $m + 1$ nodes such that $l$ lies inside the convex hull, $C(\Theta_l(r_l))$, of the nodes in $\Theta_l(r_l)$, for some $r_l > 0$.

The proof is straightforward and can be found in [1]. Intuitively, by B0, the set of anchors, $\kappa$, satisfies the conditions in Lemma 1 and is a valid choice for $\Theta(r_l)$ with $r_l = \max_{l,k} d_{lk}$, $(l \in \Omega, k \in \kappa)$. It is noteworthy that Lemma 1 is only an existence proof; it is important to find such a set within a small communication radius, $r_l$, as long-distance communication may not be possible or too costly. We refer
to such $\Theta_l$ as the *triangulation set*\(^{11}\) at node $l$. For each sensor, a triangulation set can be found by the *convex hull inclusion test* presented in Section II-C. Finding $\Theta_l$ is an important step in DILOC that we refer to as *triangulation*.

**Remark 7.** Because DILOC is based on inter-sensor distances, it may be appropriate to refer to the triangulation sets as the trilateration sets. However, an avid reader may note that in $\mathbb{R}^2$, the barycentric computation and the convex hull inclusion test can be equivalently represented in terms of the angles subtended by a sensor with respect to the three (corresponding) neighbors, see Remark 4. For these reasons, we use the triangulation terminology.

With the above assumptions and notations, we can now formally define the barycentric coordinates of a sensor, $l \in \Omega$, with respect to each of the $m + 1$ neighbors in its triangulation set, $k \in \Theta_l$. The barycentric coordinates, $a_{lk}$, are unique [69], and are given by, see also [62], [63],

$$a_{lk} = \frac{A_{\{l\} \cup \Theta_l \setminus \{k\}}}{A_{\Theta_l}}, \quad (10)$$

with $A_{\Theta_l} \neq 0$ (ensured by B2), where ‘\’ denotes the set difference, $A_{\{l\} \cup \Theta_l \setminus \{k\}}$ is the hypervolume of the set $\{l\} \cup \Theta_l \setminus \{k\}$, i.e., the set $\Theta_l$ with node $l$ added and node $k$ removed, see Fig. 3 (Left) for a graphical representation. The barycentric coordinates can be computed from the inter-node distances $d_{lk}$ (known by B3) using the Cayley-Menger determinant. From Eq. (10), and the facts that the hypervolumes are non-negative and

$$\sum_{k \in \Theta_l} A_{\Theta_l \cup \{l\} \setminus \{k\}} = A_{\Theta_l}, \quad l \in C(\Theta_l), \quad (11)$$

it follows that, for each $l \in \Omega$, $k \in \Theta_l$,

$$a_{lk} \in [0, 1], \quad \sum_{k \in \Theta_l} a_{lk} = 1. \quad (12)$$

We summarize DILOC by the following two steps, initial setup and DILOC iterations.

**A. DILOC: Initial setup**

The initial setup consists of the following two steps.

(i) **Triangulation**: Each sensor $l$ triangulates itself. At the end of this step, every $l \in \Omega$ is paired with its triangulation set, $\Theta_l$, with exactly $m + 1$ neighbors.

\(^{11}\)Reference [1] discusses the probability of finding one such $\Theta_l$ with $r_l \ll \max_{l,k} d_{lk}$, ($l \in \Omega, k \in \Theta$). For further clarification on the existence of a triangulation set within a short-distance, a sufficient condition for triangulation can be found in [1].
The following is a practical approach for triangulation. Sensor $l$ starts with a communication radius, $r_l$, that guarantees triangulation with high probability for a given density of deployment, details can be found in [1]. Sensor $l$ finds all of the sensors that it can reach within this communication radius, $r_l$, and then chooses arbitrarily $m+1$ nodes. Subsequently, sensor $l$ tests if it lies in the convex hull of these $m + 1$ neighbors, see Section II-C. If this procedure succeeds, triangulation stops. If not, sensor $l$ picks another set of $m + 1$ neighbors within the communication radius, $r_l$. Sensor $l$ repeats this step till success is declared. If all attempts fail, the sensor adaptively increases, in small increments, its communication radius, $r_l$, and repeats the process. By B0, success is eventually achieved, and each sensor is triangulated by finding $\Theta_l$ with properties in Lemma 1. If a sensor can measure the direction of the incoming radio signals, a much simpler algorithm can be readily implemented, where each sensor performs the convex hull inclusion test on exactly two appropriately chosen sets of $m + 1$ neighbors, see Lemma 2 in [1] for details.

(ii) Barycentric computation: After a triangulation set is established for each $l \in \Omega$, each sensor, $l$, computes its barycentric coordinates, $a_{lk}$, with respect to each of the $m + 1$ neighbors, $k \in \Theta_l$.

B. DILOC: State updating

Once the setup phase is complete, at time $t+1$, each sensor $l \in \Omega$, iteratively updates its state, i.e., its current location estimate, by a convex combination of the states at time $t$ of the nodes in $\Theta_l$. The anchors do not update their state, since they know their locations (see assumption B1). The updating is explicitly given by

$$c_l(t+1) = \begin{cases} 
c_l(t), & l \in \kappa, \\
\sum_{k \in \Theta_l} a_{lk} c_k(t), & l \in \Omega,
\end{cases} \quad (13)$$

where $a_{lk}$ are the barycentric coordinates of $l$ with respect to $k \in \Theta_l$.

C. DILOC: Example

To illustrate DILOC, we consider a simple example network in $\mathbb{R}^2, m = 2$, with $m + 1 = 3$ anchors and $M = 4$ sensors. The nodes are indexed such that the anchor set is $\kappa = \{1, 2, 3\}, |\kappa| = m + 1 = 3$, and the sensor set is $\Omega = \{4, 5, 6, 7\}, |\Omega| = M = 4$. The set of all of the nodes in the network is, thus, $\Theta = \kappa \cup \Omega = \{1, \ldots, 7\}, |\Theta| = N = 7$. The deployment is shown in Fig. 4. This particular case

---

12A video clip demonstrating the available triangulation sets in a given communication radius is available as supplementary download material. The corresponding file is DILOC_dynamic_topology.avi and is 32 MB.
is chosen to illustrate the details of the algorithm. In particular, it can be verified that sensor 5 does not communicate to any anchor, while every other sensor, 4, 6, 7, can communicate with exactly one anchor. Clearly, if only the communication links (and subsequently, the distances) shown as lines connecting the nodes in each of the figures in Fig. 4 are known, none of the sensors can find its location in a single step. Similarly, it can be easily realized that neither trilateration, Fig. 1 (Left), nor triangulation, Fig. 1 (Right), can be employed in this general scenario.

The triangulation sets, \( \Theta_l, l \in \Omega \), identified by the convex hull inclusion test are

\[
\Theta_4 = \{1, 5, 7\}, \quad \Theta_5 = \{4, 6, 7\}, \quad \Theta_6 = \{2, 5, 7\}, \quad \Theta_7 = \{3, 4, 6\},
\]

as highlighted in Figs. 4. These triangulation sets satisfy the properties in Lemma 1. At each sensor, \( l \in \Omega \), the barycentric coordinates, \( a_{lk}, k \in \Theta_l \), are computed using the inter-node distances (among the nodes in the set \( \{l\} \cup \Theta_l \)) in the Cayley-Menger determinant. It is noteworthy that the inter-node distances that need to be known at each sensor \( l \) to compute \( a_{lk} \) are only the distances among the \( m + 2 \) nodes in the set \( \{l\} \cup \Theta_l \). For instance, the distances in the Cayley-Menger determinant needed by sensor 7 to compute \( a_{73}, a_{74}, a_{76} \) are among the nodes in the set, \( \{7\} \cup \Theta_7 \), i.e., \( d_{73}, d_{74}, d_{76}, d_{34}, d_{36}, d_{46} \). These distances are known at sensor 7 due to B3. Sensor 7, for example, then computes the four areas, \( A_{347}, A_{367}, A_{467}, \) and \( A_{346} \); subsequently the barycentric coordinate of sensor 7, with respect to sensor 3 is \( a_{73} = \frac{A_{467}}{A_{346}} \).

Once the barycentric coordinates, \( a_{lk} \), are computed, DILOC for the sensors in \( \Omega \) is

\[
c_l(t + 1) = \sum_{k \in \Theta_l} a_{lk} c_k(t), \quad l \in \Omega = \{4, 5, 6, 7\}. \tag{14}
\]

The above constitutes the completely local and distributed implementation of DILOC at each sensor. We now show the time evolution of the (sensors’) location estimates in the above example. Fig. 5 (Left) shows the estimated coordinates of each sensor over the first 10 DILOC iterations with random initial
conditions. Fig. 5 (Right) shows the intermediate location estimates for each sensor as DILOC iterates and how each estimate becomes closer to the true location. It is important to note that, while the true location of the sensors need to be inside the convex hull of the anchors, the initial guesses to start DILOC and the intermediate iterates of DILOC can be arbitrarily positioned; in particular, DILOC still converges to the true locations even if the initial/intermediate location estimates are outside the convex hull of the anchors, as illustrated in Fig. 5 (Right).

![Fig. 5](image)

Fig. 5. (Left) Estimated coordinates over the first 10 DILOC iterations. (b) Intermediate location estimates at each sensor.

**Arbitrary deployment:** We now consider an arbitrary network with \( N = 50 \) nodes shown in Fig. 6 (Left) after triangulation. Each sensor randomly picks \( m + 1 \) neighbors in its communication radius and attempts a triangulation. If not successful, the communication radius is increased until each sensor finds a triangulation set. These sets lead to the barycentric coordinates using the Cayley-Menger determinant; DILOC is implemented with zero initial conditions and Fig. 6 (Middle) shows the intermediate locations estimates over 50 DILOC iterations. Finally, Fig. 6 (Right) shows the corresponding errors in the position estimates. The \( x \)- and \( y \)-coordinate mean-squared errors are computed as

\[
e_x(t) = \frac{1}{M} \sum_{l \in \Omega} (c_{l,1}(t) - c_{l,1}^*)^2, \quad e_y(t) = \frac{1}{M} \sum_{l \in \Omega} (c_{l,2}(t) - c_{l,2}^*)^2,
\]

(15)

whereas the total mean-squared error is \((e_x(t) + e_y(t))/2\). For visual clarity, the errors in Fig. 6 (Right) are normalized to have maximum value of 1. A larger example with 500 nodes is available in [1].

---

13 A video clip demonstrating the time evolution of sensor locations similar to Fig. 5 (Right) is available as supplementary download material. The corresponding file is DILOC_iterates1.avi and is 17 MB.

14 A video clip demonstrating the time evolution of sensor locations similar (with random initial conditions) to Fig. 6 (Middle) is available as supplementary download material. The corresponding file is DILOC_iterates2.avi and is 25 MB.
IV. DILOC: Analysis

For compactness of notation and analysis purposes, we write DILOC, given in Eq. (13), in matrix form. Let the anchors in $\kappa$ be indexed by $1, \ldots, m + 1$ and let the sensors in $\Omega$ be indexed by $m + 2, \ldots, N$ where $N = m + 1 + M$; note that this indexing is without loss of generality. We group the nodal coordinates, $m$-dimensional row vectors, $c_i$ in Eq. (9), in an $N \times m$-dimensional coordinate matrix,

$$C = \begin{bmatrix} c_1^T, \ldots, c_N^T \end{bmatrix}^T. \quad (16)$$

Each row of this matrix is the vector of coordinates of the corresponding sensor. The matrix representation of the DILOC equations, Eq. (13), now becomes

$$C(t + 1) = \Upsilon C(t). \quad (17)$$

The structure of the $N \times N$ iteration matrix $\Upsilon$ is more apparent if we partition it as

$$\begin{bmatrix} C_\kappa(t + 1) \\ C_\Omega(t + 1) \end{bmatrix} = \begin{bmatrix} I_{m+1} & 0 \\ B & P \end{bmatrix} \begin{bmatrix} C_\kappa(t) \\ C_\Omega(t) \end{bmatrix}, \quad (18)$$

where $C_\kappa(t)$ are the anchors’ (in the set $\kappa$) coordinates at time $t$, and $C_\Omega(t)$ are the sensors’ (in the set $\Omega$) coordinates at time $t$. Clearly, from Assumption B1, we have $C_\kappa(t) = \ldots = C_\kappa^*$, where $C_\kappa^*$ denotes the exact locations of the anchors. Some key characteristics of Eq. (18) are noted below.

(i) In the matrix $\Upsilon$, we note that the first $m + 1$ rows correspond to the update equations for the anchors in $\kappa$. Recall that by Assumption B1, the anchors’ states are constant; they do not change

Fig. 6. (Left) Randomly deployed $N = 50$ node network and the respective triangulation sets. (Middle) Intermediate location estimates over 50 DILOC iterations. (Right) Mean-squared x-coordinate error (dashed blue line) and mean-squared y-coordinate error (dash-dotted red line) averaged over the $M = 47$ sensors; total mean-squared error (dotted black line).
with time. So, the first \( m + 1 \) rows of \( \Upsilon \) are zero except for a 1 at the diagonal entry \((q,q), q \in \kappa = \{1, \ldots, m + 1\} \).

(ii) Each of the \( M \) remaining rows in \( \Upsilon \), indexed by \( l \in \Omega = \{m+2, m+3, \ldots, N\} \), has at most \( m + 1 \) non-zero elements corresponding to the nodes in the corresponding triangulation set, \( \Theta_l \); these non-zero elements are the barycentric coordinates, \( a_{lk} \), of sensor \( l \) with respect to the nodes in \( \Theta_l \) and add up to 1.

(iii) The \( M \times (m + 1) \) block matrix, \( B = \{b_{lj}\} \), has non-zeros only at those \( b_{lj} \)'s that represent the barycentric coordinates of sensor \( l \) with respect to the anchors \((j \in \kappa) \) in sensor \( l \)'s triangulation set, \( \Theta_l \). Because anchors are not within the communication radius of most sensors, most rows in \( B \) are zero rows. Note however that not all rows of \( B \) can be zero.

(iv) The \( M \times M \) block matrix, \( P = \{p_{lj}\} \), has non-zeros entries in each of the \( l \)th row corresponding to the barycentric coordinates of sensor \( l \) with respect to the sensors in its triangulation set \( \Theta_l \). Note that each row of \( P \) can have at most \( m + 1 \) non-zeros.

A. Convergence

With the matrix format of DILOC in Eq. (17), we now analyze its convergence. To formally state the convergence result, we provide a few additional facts derived from the Assumptions B0–B3.

The entries of the rows of the iteration matrix \( \Upsilon \), in Eq. (18), are either zero or the barycentric coordinates, \( a_{lk} \), which are non-negative. The corresponding matrix, \( \Upsilon \), is thus non-negative. Non-negative matrices have very special properties established by the Perron-Frobenius theorem [78], [79]. By Eq. (12), each row of \( \Upsilon \) adds to 1 making it a stochastic matrix. The matrix, \( \Upsilon \), thus, can be interpreted as the transition probability matrix of a Markov chain, leading to a description of the localization problem and DILOC in terms of the evolution of a Markov chain. Under the assumptions described in B0–B3, the \( N \) nodes in the sensor network correspond to the states of a Markov chain where the \((ij)\)-th element of the iteration matrix, \( \Upsilon = \{v_{ij}\} \), defines the transition probability, i.e., the probability that the \( i \)th state goes to the \( j \)th state. Because of the structure of \( \Upsilon \), the corresponding Markov chain has a very special structure that we describe below.

Remark 8. Note a subtle distinction that \( v_{ij} \) implies that the state \( j \) can reach the state \( i \) in one-step with probability \( v_{ij} \), i.e., \( j \rightarrow i \) in the Markov chain sense, and, that the node \( i \) can transmit information to the node \( j \), i.e., \( i \rightarrow j \) in the communication graph sense.

Absorbing Markov chain. Let an \( N \times N \) matrix, \( \Upsilon = \{v_{ij}\} \), denote the transition probability matrix
of a Markov chain with $N$ states, $s_{i, i=1,...,N}$. A state $s_i$ is called absorbing if the probability of leaving that state is 0 (i.e., $v_{ij} = 0$, $i \neq j$, in other words, $v_{ii} = 1$). A Markov chain is said to be absorbing if it has at least one absorbing state, and if from every state it is possible to go with a non-zero probability to an absorbing state (not necessarily in one step). In an absorbing Markov chain, a state that is not absorbing is called transient. For additional background, see, for example, [80].

Absorbing states–anchors. Note that anchors are absorbing since $\{Y\}_{ii} \triangleq v_{ii} = 1$, $i \in \kappa$, i.e., there is zero probability of escape once trapped in one of these states. To show that the underlying Markov chain is absorbing, it then suffices to show that the sensor states are transient, i.e., from each sensor state it is possible (with strictly positive probability) to reach an absorbing state–anchor, in finite time. Given that the Markov chain in this context has finite state-space, the desired transience of the sensor states is equivalent to showing that from each sensor there exists a directed path, possibly multi-hop, to an anchor. It can be noted that due to $B_0$, each sensor has a directed (Markov chain) path to (or a multi-hop communication link from) all of the $m + 1$ anchors.

Transience of sensor states. Consider the partitioning of the iteration matrix, $Y$, in Eq. (18). With the Markov chain interpretation, the $M \times (m + 1)$ block $B = \{b_{lj}\}$ is a transition probability matrix for the transient states to reach the absorbing states (anchors) in one-step, and the $M \times M$ block $P = \{p_{lj}\}$ is a transition probability matrix for the transient states (sensors) to reach other transient states. Now note that the non-negative matrix $P$ is substochastic since at least one of its row sum is strictly less than one. This row belongs to the sensor that uses anchors for triangulation and at least one such row will always exist by Lemma 1 and consequently, by Assumption $B_0$; the existence is ensured by the fact that at least one sensor has to use anchors to triangulate. Clearly, the row sums for at least one row of $P$ is strictly less than 1 making $P$ a substochastic matrix.

Finally, we show that $P$ is irreducible, i.e., it cannot be rearranged (by row-column permutations) into a block lower-triangular matrix such that the lower diagonal block is stochastic–with rows summing to 1. Let us assume on the contrary that $P$ can be rearranged after a row-column permutation into a block lower-triangular matrix,

$$P = \begin{bmatrix} P_1 & P_{12} \\ 0 & P_2 \end{bmatrix},$$

such that $P_2$ is stochastic, i.e., each row sum of $P_2$ is 1. For the sake of argument, let us partition the set of sensors, $\Theta$, into $\Theta_1$ and $\Theta_2$, where $\Theta_1$ collects all of the sensors corresponding to $P_1$, and $\Theta_2$ collects all of the sensors corresponding to $P_2$. Assuming that $P_2$ is stochastic, implies that the sensors in $\Theta_2$ have triangulated themselves without any anchor and without any sensor from $\Theta_1$. Clearly, this
is not possible because those sensors that lie on the periphery (boundary of $C(\Theta_2)$) of all of the sensors in $\Theta_2$, cannot triangulate without either using a sensor from $\Theta_1$ or using an anchor. Combining the aforementioned arguments that: (i) $P$ is substochastic; and, (ii) $P$ is irreducible, we conclude that $P$ is uniformly substochastic. For uniformly substochastic matrices, it is known that their (non-negative) spectral radius is strictly less than 1, see\textsuperscript{15} [82] for details.

The following lemma is apparent given that $P$ is a uniformly substochastic matrix, and from the aforementioned Markov chain arguments. To summarize, we have: (i) the anchors correspond to the absorbing states; (ii) the sensors correspond to the transient states; and (iii) each transient state has a strictly positive probability to reach an absorbing state, possibly, in multiple steps.

**Lemma 2.** The underlying Markov chain with transition probability matrix given by the iteration matrix $\Upsilon$ is absorbing. The $m+1$ anchors are the absorbing states. For an absorbing Markov chain, the transition probability matrix, $P$, defined only on the transient states is such that

$$
\rho(P) < 1,
$$

where $\rho(\cdot)$ denotes the spectral radius of the matrix in its argument.

A detailed proof is provided in [1] and follows the arguments described before on the absorbing and transient nature of the anchors’ and sensors’ states. With Eq. (18), $\Upsilon^{t+1}$ can be written as

$$
\Upsilon^{t+1} = \begin{bmatrix}
I_{m+1} & 0 \\
\sum_{k=0}^{t} P^k B & P^{t+1}
\end{bmatrix} \Rightarrow \lim_{t \to \infty} \Upsilon^{t+1} = \begin{bmatrix}
I_{m+1} & 0 \\
(I_M - P)^{-1} B & 0
\end{bmatrix},
$$

under Eq. (20), i.e., DILOC in Eq. (13) converges to

$$
\lim_{t \to \infty} \begin{bmatrix}
C_\kappa(t + 1) \\
C_\Omega(t + 1)
\end{bmatrix} = \begin{bmatrix}
I_{m+1} & 0 \\
(I_M - P)^{-1} B & 0
\end{bmatrix} \begin{bmatrix}
C_\kappa^* \\
C_\Omega(0)
\end{bmatrix}.
$$

From Eq. (22), we note that the coordinates, $C_\Omega(t)$, of the $M$ sensors in $\Omega$ converge as $t \to \infty$ to functions of the $m + 1$ anchors’ coordinates, $C_\kappa^*$, in $\kappa$. The limiting coordinates are given by

$$
\lim_{t \to \infty} C_\Omega(t + 1) = (I_M - P)^{-1} BC_\kappa^*.
$$

The above equation further shows that DILOC forgets the sensor initial conditions, $C_\Omega(0)$. In other words, DILOC converges to Eq. (23) regardless of what coordinates are chosen as the sensor initial conditions. The following lemma now shows that the true sensor locations are indeed given by Eq. (23).

\textsuperscript{15}The original paper [81] is in Russian and [82] is an English translation.
Lemma 3. Let $c_l^*$ be the exact coordinates of a node, $l \in \Theta$. Let the $M \times (m+1)$ matrix, $D = \{d_{lj}\}, l \in \Omega, j \in \kappa$, be the matrix of the barycentric coordinates of the $M$ sensors (in $\Omega$) in terms of the $m + 1$ anchors in $\kappa$, relating the coordinates of the sensors to the coordinates of the anchors by

$$c_l^* = \sum_{j \in \kappa} d_{lj} c_j^*, \quad l \in \Omega. \tag{24}$$

Then, we have

$$D = (I_M - P)^{-1} B. \tag{25}$$

A proof of Lemma 3 is provided in [1]. We present an intuitive justification of the fact that $d_{lj}$ are indeed the entries of the matrix defined in the R.H.S of Eq. (25). To this end, note that, by construction and the definition of the barycentric coordinates, the true location vector, $C^*$, is a fixed-point of the matrix $Y$, i.e., $C^* = YC^*$. From the block lower-triangular structure of $Y$, Eq. (18), it then follows that

$$C_{\Omega}^* = BC_{\kappa}^* + PC_{\Omega}^* \Rightarrow C_{\Omega}^* = (I_M - P)^{-1} BC_{\kappa}^*. \tag{26}$$

Finally, we argue that the inverse, $(I_M - P)^{-1}$, always exists under our assumptions, B0-B3. To show this, note that the eigenvalues of the matrix, $I_M - P$, are $1 - \lambda_i(P)$, where $\lambda_i(P)$ is the $i$th eigenvalue of $P$. Since, by Eq. (20) and the Perron Frobenius theorem [78], the maximum eigenvalue of $P$ is non-negative and strictly less than 1, the eigenvalues, $1 - \lambda_i(P)$, of the matrix, $I_M - P$, can never be zero, and thus, $I_M - P$ is invertible. It then follows that $C_{\Omega}^*$ is uniquely determined by Eq. (26), which are the limiting coordinates given by DILOC, Eq. (23). We now recapitulate these results in a theorem.

Theorem 1. Under B0-B3, the DIstributed sensor LOCalization algorithm–DILOC in Eq. (13), converges to the exact sensor coordinates, $c_l^*$, $l \in \Omega$, i.e.,

$$\lim_{t \to \infty} c_l(t + 1) = c_l^*, \quad \forall \ l \in \Omega. \tag{27}$$

Proof: The proof is a consequence of Lemmas 2 and 3.

Remark 9. The convergence rate of DILOC depends on the spectral radius, $\rho(P)$, of the matrix $P$, which as argued before is strictly less than one. In fact, it can be shown that DILOC is characterized by a geometric convergence rate with exponent $\rho(P)$. The convergence is slow if the spectral radius, $\rho(P)$, is close to 1. This can happen if the matrix $B$ is close to a zero matrix, for example, when the sensors cluster in a region of very small area inside the convex hull of the anchors with very far-away anchors, in which case the sensor-anchor barycentric coordinates in the matrix $B$–for the sensors with anchors
in their respective $\Theta_l$, will be very small. When, as in practical wireless sensor applications, the nodes are assumed to be deployed in a geometric or a Poisson fashion—see details in [1], the sensors are uniformly distributed and the event that sensors cluster in a small region away from the anchors is highly improbable.

V. GENERALIZATIONS TO DILOC

A series of papers, [1], [64]–[68], have extended and generalized DILOC. We group these extensions into deterministic and uncertain environments. We briefly explore them below.

A. Deterministic: More than $m + 1$ neighbors

It is not unnatural to assume that a sensor in a typical Wireless Sensor Network (WSN) has more than $m + 1$ neighbors in its communication radius, and thus, it is possible for a sensor to find multiple triangulation sets. Let $\hat{\Theta}_l$ denote the set of sensors or anchors that lie in the communication radius, $r_l$, of sensor $l$, i.e., $\hat{\Theta}_l = \{j \in \Theta \mid d_{lj} < r_l\}$, and let $\Theta^i_l \subseteq \hat{\Theta}_l$ be the collection of subsets of $\hat{\Theta}_l$ such that each element, $\Theta^i_l \in \Theta^i_l$, is a triangulation set for sensor $l$ (i.e., the conditions in Lemma 1 hold for each $\Theta^i_l$). Each sensor $l$ can now express its coordinates as a convex combination of the coordinates of all of the nodes in $\Theta^i_l$, i.e.,

$$c_l = \sum_i w^i_l \sum_{j \in \Theta^i_l} a^i_{lj} c_j, \quad l \in \Omega,$$

where $w^i_l \geq 0 \forall i$ and $\sum_i w^i_l = 1$. Eq. (28) leads to the following iterative procedure

$$c_l(t + 1) = \begin{cases} c_l(t), & l \in \kappa, \\ \sum_i w^i_l \sum_{j \in \Theta^i_l} a^i_{lj} c_j(t), & l \in \Omega, \end{cases}$$

which is characterized below.

**Lemma 4.** The localization algorithm in Eq. (29) converges to the exact sensor locations, $c^*_l$, $\forall l \in \Omega$, i.e., $\lim_{t \to \infty} c_l(t + 1) = c^*_l$, $l \in \Omega$.

The proof can be found in [64]. As an illustration, consider Fig. 7 where for an arbitrary sensor—shown as a blue square, we show the neighbors available in an arbitrarily chosen communication radius—shown as the dashed circle. In the first three figures (from the left), we show three different triangulation sets that can be formed given the neighbors of the node in question; clearly there are more than three choices available. The result of DILOC for each different choice is shown in Fig. 7 (Rightmost) as the solid curve. DILOC modified to incorporate all of the available triangulation sets, from Eq. (29), is shown as...
the dashed line. The corresponding weight, $w_i$, on each triangulation set is chosen to be the inverse of the total number of sets, i.e., $1/|\Theta_l|$.

![Diagram of triangulation sets](image_url)

Fig. 7. (First three from left-to-right) A few different triangulation sets for the sensor marked as blue square. (Rightmost) Solid lines—normalized network localization error with each of the triangulation set picked independently; Dashed line—normalized network localization error for a combination of all of the available triangulation sets merged according to Eq. (28).

**Remark 10.** Fig. 7 (Rightmost) shows that, in an arbitrary deployment, using all available neighbors in every communication radius can lead to a faster convergence rate. However, it is noteworthy that the particular choice of $w_i$’s plays a key role in characterizing this convergence. The optimal choice of such $w_i$’s remains to be investigated; in this context, the characterization in [69] may be helpful.

**B. Deterministic: More than $m + 1$ anchors**

Consider the case where the number of anchors is greater than $m + 1$, i.e., $|\kappa| = K > m + 1$. The iterative procedure has the same form as Eq. (13), however, the total number of sensors plus anchors becomes $K + M$. We have the following result.

**Lemma 5.** The iterative localization algorithm with $K > m + 1$ anchors converges to the exact sensor locations, $c^*_l \forall l \in \Omega$, i.e., $\lim_{t \to \infty} c_l(t + 1) = c^*_l, l \in \Omega$.

The proof can be found in [64] and uses very similar arguments as made in Section IV-A following Lemma 3 on the fixed-point of linear iterations. As an illustration, we show an $N = 50$ node randomly deployed network with $K = 4$ anchors in Fig. 8 (Left) after triangulation. The triangulation in Fig. 8 (Left)\(^\text{16}\) is generated with all of the sensors in the convex hull of the 4 anchors. On the other hand, Fig. 8 (Middle) shows the same network divided into two localization algorithms with 3 anchors each.

\(^{16}\)A video clip demonstrating the time evolution of sensor locations corresponding to a deployment similar to Fig. 8 (Left) is available as supplementary download material. The corresponding file is DILOC_more_anchors.avi and is 18 MB.
The two different triangulations with respect to different sets of 3 anchors is shown as solid vs. dashed lines. Finally, Fig. 8 (Right) plots the normalized total mean-squared error (MSE, see Eqs. (15)) for the entire network and the two different configurations over 20 DILOC iterations. In Fig. 8 (Right), the solid line represents the network localization error for the 4 anchors case, whereas the dashed line shows the network error for the configuration in Fig. 8 (Middle). It can be readily verified that although the configuration with more than three anchors in $\mathbb{R}^2$ can be solved by splitting the deployment into distinct localization problems with three anchors each, DILOC can be implemented without any such partitioning.

**Remark 11.** By increasing the number of anchors, a sensor deployment within any convex polygon in $\mathbb{R}^2$ (or appropriate generalizations in higher dimensions) can be realized where the anchors are placed at the corners of the underlying polygon. In most practical scenarios, e.g., indoor localization of objects (packages/assets) in a facility (warehouse/factory), it is true that the objects (packages/assets) lie within the confines of the facility. The anchors are, thus, to be placed at the appropriate corners of the corresponding facility—the anchor deployment in this scenario is a rather benign task.

Fig. 8. (Left) A network with $M = 46$ sensors in the convex hull of $K = 4$ anchors. (Middle) Two distinct localization problems with 3 anchors each. (Right) Corresponding network localization errors: Red dashed—total error when the deployment is partitioned into two separate problems; Blue solid—error without partitioning.

**C. Deterministic: DILOC with relaxation**

We observe that in DILOC, Eq. (13), at time $t + 1$, the expression for $c_l(t + 1)$, $l \in \Omega$, does not involve its own coordinates, $c_l(t)$, at time $t$. To add a weight on the self-estimate at time $t$, we introduce a relaxation parameter, $\alpha \in (0, 1]$, in the iterations, such that the expression of $c_l(t + 1)$ is a convex combination of $c_l(t)$ and Eq. (13). We refer to this version as DILOC with relaxation, DILOC-REL, is
given by
\[
c_l(t + 1) = \begin{cases} 
(1 - \alpha)c_l(t) + \alpha c_l(t) = c_l(t), & l \in \kappa, \\
(1 - \alpha)c_l(t) + \alpha \sum_{k \in \Theta_l} a_{lk} c_k(t), & l \in \Omega.
\end{cases}
\] (30)

DILOC is the special case of DILOC-REL with $\alpha = 1$. Clearly, DILOC-REL is also distributed as the sensor updates now have additional terms corresponding to their own past states. The following lemma establishes convergence of DILOC-REL; the proof can be found in [1].

**Lemma 6.** The iterative localization algorithm in Eq. (30) converges to the exact sensor coordinates, $c_l^*, l \in \Omega$, i.e., $\lim_{t \to \infty} c_l(t + 1) = c_l^*, \forall l \in \Omega$.

**Remark 12.** A very interesting extension to DILOC with relaxation is to find the fastest convergence rate over all possible values of the relaxation parameter, i.e., $\alpha \in [0, 1]$. Since the convergence rate of DILOC depends on the largest eigenvalue, i.e., the spectral radius of $P$, where the largest eigenvalue is known to be always non-negative by the Perron Frobenius theorem [78], one can show that DILOC with relaxation results into $P_\alpha = (1 - \alpha)I_M - \alpha P$ that has the eigenvalues, $1 - \alpha \lambda_i(P)$, where $\lambda_i(P)$ is the $i$th eigenvalue of $P$. With this linear-convex realignment of $P_\alpha$’s eigenvalues, it may be possible to choose $\alpha$ in order to improve the convergence rate, i.e., decrease the spectral radius. However, such an optimization requires the knowledge of the matrix $P$ or equivalently, all of the inter-sensor barycentric coordinates. Although having this information may be impractical, special structures in sensor deployment may be exploited in order to obtain a pre-deployment spectral characterization of $P$.

The following subsections enumerate the extensions of DILOC to environments with uncertainties.

**D. Uncertain: Dynamic network topology**

Consider the case with dynamic network topology, where each sensor, $l$, chooses a different triangulation set, $\Theta_l(t)$, at each iteration $t$ of the iterative algorithm, such that the conditions in Lemma 1 hold\(^{17}\).

In this case, the coordinates of the $l$th sensor can be written as
\[
c_l(t + 1) = \begin{cases} 
c_l(t), & l \in \kappa, \\
\sum_{j \in \Theta_l(t)} a_{lj}(t)c_j(t), & l \in \Omega.
\end{cases}
\] (31)

The following lemma establishes the convergence of the above algorithm; the proof can be found in [64].

\(^{17}\)A video clip demonstrating the randomly chosen triangulation set is available as supplementary download material. The corresponding file is DILOC_dynamic_topology.avi and is 32 MB.
Lemma 7. The localization algorithm Eq. (31) with different triangulation sets at each iteration, converges to the exact sensor locations, $c_l^* \forall l \in \Omega$, i.e., $\lim_{t \to \infty} c_l(t + 1) = c_l^*, l \in \Omega$. 

Remark 13. The protocol established in the dynamic topology scenario can be either deterministic or random. For example, each sensor may cycle through all of the available triangulation sets in a specific order, or at every iteration may choose a triangulation set randomly. It can be shown that the proposed solution is applicable to both deterministic and random scenarios. Furthermore, the random sampling does not have to be uniform, for example, if a sensor knows that certain triangulation sets may lead to a faster convergence (or equivalently, lower spectral radius of $P$), it may assign high probability of selection to such favorable triangulations. A favorable triangulation is, for example, when the three barycentric coordinates of the sensor in question are relatively close.

E. Uncertain: Random environment

A significant challenge in wireless sensor networks is randomness in the environment. In particular, wireless communication is often subject to data packet drops, communication noise, uncertainty in the barycentric coordinates, and imprecise distance measurements. Our precise modeling of these uncertainties in the context of noisy Received Signal Strength (RSS) and/or noisy Time-of-Arrival (ToA) measurements for the inter-node distance computation can be found in [1], [68]. Briefly, References [1], [68] provide two algorithms, Distributed Localization in Random Environments (DLRE) and DIstributed LocAlization with Noisy Distance measurements (DILAND), respectively, that address the presence of data packet drops, communication noise, imperfect barycentric computation, and noisy distance measurements. The corresponding localization algorithms employ a stochastic approximation of DILOC and show that the modified algorithm converges to the exact sensor locations almost surely.

Remark 14. We briefly note that the DILOC with relaxation in Section V-C is particularly helpful in DILAND and DLRE. Specifically, we choose the relaxation parameter, $\alpha$, as a time-varying sequence, $\alpha(t)$, that satisfies the persistence conditions: $\alpha(t) > 0, \sum \alpha(t) = \infty$, and $\sum \alpha^2(t) < \infty$; this procedure is adapted from the stochastic approximation literature, see [1], [68] for details and the references therein.

F. Mobile networks

DILOC can be further extended to mobile networks [66]. In this setup, we assume that an arbitrary number of sensors with unknown locations lie in the convex hull of at least $m + 1$ anchors that precisely know their locations and motion (for instance, they may have a GPS device). We consider a broad motion
model that captures several practical scenarios of coordinated and uncoordinated motion of mobile agents. The motion model we consider is as follows.

\[
C^*(t) = \mathcal{A}C^*(t) + z(t) + y(t),
\]  

(32)

where the \( N \times N \) matrix \( \mathcal{A} \) relates the motion of a sensor to its neighbors such that the network may move in a coordinated fashion. The matrix \( z(t) \) is the deterministic drift added to the coordinates, whereas the matrix \( y(t) \) is the random drift with bounded norm. By a careful choice of variables in the motion model, Eq. (32), we can consider two scenarios:

**Uncoordinated motion in a fixed region.** In this scenario, the anchors remain fixed and the sensors move randomly inside their convex hull. This can be thought of as the motion of wireless objects that move randomly inside a given region (or cell).

**Coordinated motion driven by anchors.** Consider another scenario where the motion model is driven by anchors and the conditions on \( \mathcal{A} \), guarantee that the sensors move in a coordinated manner, among themselves, driven by the anchors.

It is shown in [66] that the following algorithm,

\[
C(t + 1) = \Upsilon_{t+1} \begin{pmatrix} \mathcal{A}C(t) + \begin{bmatrix} z_u(t) \\ z_x(t) \end{bmatrix} \end{pmatrix},
\]

(33)

converges to the sensor locations in an \( \varepsilon \)-ball\(^{18}\) around the true locations, under some mild conditions on the network connectivity, where \( \Upsilon_{t+1} \) is the matrix of barycentric coordinates at time \( t + 1 \).

VI. CONCLUSIONS

The problem of localization has been known for centuries and has undergone an extensive treatment in the realm of statistical, optimization, and graph-theoretic frameworks, all of which are inherently nonlinear. In this paper, we provide a linear theory for localization, a linear solution to this nonlinear problem, that is applicable to networks of arbitrary number of sensors (points in \( \mathbb{R}^m \) with unknown locations). Our linear localization algorithm, DILOC, realizes that in most practical localization setups, sensors lie in the convex hull of at least \( m + 1 \) anchors. By exploiting judiciously this convexity structure, we change the nonlinear representation of localization in terms of Cartesian or polar coordinates to a linear representation using the barycentric-coordinates. This barycentric-coordinate representation, which is applicable to arbitrary Euclidean spaces \( \mathbb{R}^m \), leads to DILOC, a linear, local, and distributed iterative

\(^{18}\)The radius of the \( \varepsilon \)-ball depends on the upper bound on the norm of the random drift vector, \( y(t) \).
localization algorithm. With DILOC, sensors iteratively refine their position estimates through peer-to-peer inter-agent (local) communication and through local computation (performed at the sensors). In particular, a sensors’s peer-set does not need to include any anchor and there is no need for any centralized entity. The paper discusses several variants of the basic DILOC scheme that are concerned with mobile agents, general network configurations, and extensions of DILOC to cope with communication and sensing uncertainties.
REFERENCES


