Relax and Recover: Guaranteed Range-Only Continuous Localization

Michalina Pacholska, Frederike Dümbgen, and Adam Scholefield

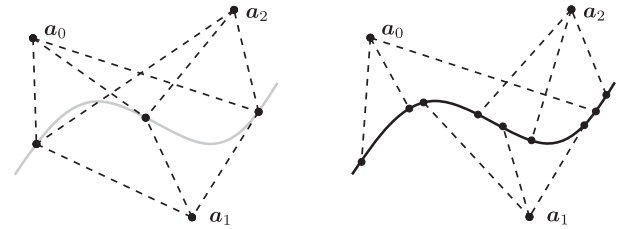
***Abstract*—Range-only localization has applications as diverse as underwater navigation, drone tracking and indoor localization. While the theoretical foundations of lateration—range-only localization for static points—are well understood, there is a lack of understanding when it comes to localizing a moving device. As most interesting applications in robotics involve moving objects, we study the theory of trajectory recovery. This problem has received a lot of attention; however, state-of-the-art methods are of a probabilistic or heuristic nature and not well suited for guaranteeing trajectory recovery. In this letter, we pose trajectory recovery as a quadratic problem and show that we can relax it to a linear form, which admits a closed-form solution. We provide necessary and sufficient recovery conditions and in particular show that trajectory recovery can be guaranteed when the number of measurements is proportional to the trajectory complexity. Finally, we apply our reconstruction algorithm to simulated and real-world data.**

***Index Terms*—Localization, range sensing, optimization and optimal control, trajectory estimation, SLAMpling.**

I. Introduction

A ROBOT'S ability to localize itself accurately is essential for applications such as exploration, rescue and delivery. In unknown or rapidly changing environments, visual (camera-based) simultaneous localization and mapping (SLAM) is a popular solution for reliable localization [1]. However, various settings exist in which visual SLAM is not practical, for example when scanning the environment is impossible (e.g. passive indoor localization) or when the environment does not exhibit enough reliable features (for example under water [2], [3], at high altitudes [4], or in large exhibition-style rooms [5]). In such situations, it is sometimes more feasible to install a few fixed anchors which can provide the robot with distance measurements. Given sparse range measurements from multiple anchors, the robot can calculate its position through multilateration.

While position recovery guarantees exist for traditional lateration in static setups (see Fig. 1(a)), the problem is less understood when the robot is moving. To date, practical systems predominately recover trajectories by coupling partial lateration with filtering techniques [6]. While these approaches lead to good performance, they offer little hope of providing fundamental guarantees for the recovery of the robot's continuous trajectory. In these cases, under which conditions is it possible to uniquely recover the trajectory?



(a) Conventional lateration (b) Proposed trajectory estimation

Fig. 1. Two different approaches for recovering a trajectory $\boldsymbol{r}(t)$ (solid line) from distance measurements (dashed lines) to anchors $\boldsymbol{a}_m$. In conventional lateration (a), we recover single points at which we have at least $D+1$ distance measurements. The proposed method (b) recovers the continuous representation of the trajectory $\boldsymbol{r}(t)$ from non-synchronized measurements.

We answer this question in the particular setting in which a moving robot obtains range measurements from static and known anchors. In particular, we do not require the measurements to be perfectly synchronized, nor to be uniformly distributed in time. We make the realistic assumption that the robot can only measure one range at a time and limit ourselves to smooth trajectories—in particular, we focus on bandlimited and polynomial trajectories. An example setup is sketched in Fig. 1(b). With this setup, it is straight-forward to see that traditional lateration cannot provide us with recovery algorithms, and even less with uniqueness guarantees. One can instead resort to trajectory estimation algorithms, which provide either a probabilistic or deterministic description of the continuous trajectory, but no guarantees for perfect recovery exist.

In this letter, we obtain a novel closed-form solution to the trajectory estimation problem, by relaxing the quadratic constraints. In addition, by studying the obtained linear system, we deduce necessary and sufficient conditions for trajectory recovery of the relaxed problem. This also provides a sufficient condition for trajectory recovery of the original (non-relaxed) problem.

II. Problem Formulation

To enable us to more accurately compare our contribution with existing techniques, we define our problem setup before discussing related work in the next section. Throughout the paper, we use regular lower-case letters for variables (t), regular upper-case letters for constants (K), bold lower-case letters for column vectors ($\boldsymbol{c}$) and bold upper-case for matrices ($\boldsymbol{C}$). By

Manuscript received September 10, 2019; accepted January 4, 2020. Date of publication February 3, 2020; date of current version February 17, 2020. This letter was recommended for publication by Associate Editor Prof. U. Frese and Editor Prof. S. Behnke upon evaluation of the reviewers' comments. *(Michalina Pacholska and Frederike Dümbgen contributed equally to this work.) (Corresponding author: Frederike Duembgen.)*

The authors are with the Faculty of Computer and Communication Sciences, École Polytechnique Fédérale de Lausanne, 1015 Lausanne, Switzerland (e-mail: michalina.pacholska@epfl.ch; frederike.duembgen@epfl.ch; adam.scholefield@epfl.ch).

Digital Object Identifier 10.1109/LRA.2020.2970952

D we denote the robot's embedding dimension, which is fixed but in principle could be arbitrary. In this letter, we usually use $D = 2$.

Our aim is to recover the position, $\boldsymbol{r}(t) \in \mathbb{R}^D$, of a moving device (e.g. a robot), for t in some given interval, $t \in \mathcal{I} \subset \mathbb{R}$. At a set of time instances $\{t_n : n = 0, \ldots, N-1\}$, $t_n \in \mathcal{I}$, we measure the distance from the robot's (unknown) position $\boldsymbol{r}_n := \boldsymbol{r}(t_n)$ to one of M fixed anchors. We denote the anchor positions by $\boldsymbol{a}_m \in \mathbb{R}^D$, $m = 0, \ldots, M-1$, and assume that they are known. The distances are thus $d_n = \|\boldsymbol{r}_n - \boldsymbol{a}_{m_n}\|$, where $\|\cdot\|$ is the Euclidean norm and m_n is the index of the anchor used at time n. In practice, we assume that we can measure distances $\tilde{d}_n$ corrupted by additive zero-mean Gaussian noise: $\tilde{d}_n = d_n + \epsilon_n$, where $\epsilon_n \sim \mathcal{N}(0, \sigma^2)$.[1]

For ease of analysis, we assume that each t_n is different; in fact, this is a strength of our formulation as it means measurements from different anchors are not assumed to be synchronized. However, since the t_n are real numbers, two consecutive t_n can be arbitrarily close.

For noisy measurements, the maximum likelihood estimator (MLE) of the device's position at time instant t_n is given by the solution to the following optimization problem, also denoted Range Least Squares (RLS) [7]:

$$\underset{\hat{\boldsymbol{r}}_n \in \mathbb{R}^D}{\arg\min} \sum_{n=0}^{N-1} \left(\tilde{d}_n - \|\hat{\boldsymbol{r}}_n - \boldsymbol{a}_{m_n}\| \right)^2. \tag{1}$$

It is common to square the two terms inside the brackets, leading to the Squared RLS (SRLS) problem:

$$\underset{\hat{\boldsymbol{r}}_n \in \mathbb{R}^D}{\arg\min} \sum_{n=0}^{N-1} \left(\tilde{d}_n^2 - \|\boldsymbol{a}_{m_n}\|^2 + 2\boldsymbol{a}_{m_n}^\top \hat{\boldsymbol{r}}_n - \|\hat{\boldsymbol{r}}_n\|^2 \right)^2. \tag{2}$$

In the following section, we review some of the techniques that have been devised to tackle these problems.

III. Related Work

Note that although the focus of the paper is localization, we include the SLAM literature, as localization is a core part of SLAM and thus many methods can be transferred from one problem to the other.

A. Basic Concepts for Range-Only Localization

A core concept of many range-based localization algorithms is lateration, or how to estimate an object's location from distances to anchor points of known position. This problem can be tackled from the RLS or SRLS perspective. While both problems are non-convex, they are not equally difficult to solve. In particular, no known algorithm is guaranteed to find the optimal solution of (1) but (2) can be solved optimally [7]. However, the SRLS solution is not the MLE. Therefore, it is common to use a standard non-linear least-squares solver such as Levenberg-Marquardt [8], which can recover a local minimum of (1) [9].

[1]Note that, it is easy to extend the noise model to differing variances for each distance. However, we chose to present the simplest case of additive i.i.d. noise as the alternative results in additional notational overhead for concepts that are well known. In the case of correlated measurements, one can write an analogue of Theorem 1; however, the equations become more complex and the proposed algorithm needs to be changed.

An important variation of the classical lateration problem is when the anchor locations are also unknown. Historically, this problem has often been analyzed through Euclidean Distance Matrices (EDMs), which contain the squared distances between points, and exhibit characteristic properties which can be exploited for denoising, completion and point recovery [10]. Although EDMs by definition are more closely related to the SLAM problem, knowledge of anchor locations can be incorporated through the Procrustes transform, or in a semidefinite program, as proposed in [11].

B. Non-Parametric Trajectory Recovery

Using the above methods, a moving object can only be localized at discrete time instances. This imposes a strong requirement on the number of measurements available at each such time instant, and does not ensure consistency between subsequent position estimates.

Numerous algorithms solve these two issues by combining range measurements with movement estimates from inertial measurement units (IMUs) in standard filtering methods such as particle or Kalman filters [6], [12]–[15]. The obtained accuracy depends strongly on the sampling rate at which position updates can be computed, and problems can arise when IMU measurements are delivered at a much higher frequency than other modalities [16].

Sampling-rate issues can be solved by continuous-time non-parametric trajectory models. A widely used approach [17]–[19] is to impose time consistency using Gaussian processes. Numerous research efforts have been invested to make these computationally expensive methods more efficient, using for example Bayes trees for incremental reordering and just-in-time relinearization [20].

C. Parametric Trajectory Recovery

As opposed to the previously discussed methods, in this letter, we aim to recover a parametric model of the robot's position. A number of other works have been proposed to this end, predominantly using splines. A comprehensive review of this field is given in [16]. Li *et al.* [21] solve the classical SLAM problem, replacing the position update of the usual state-space equations with a continuous, parametric trajectory $x(t) = F(C_k, t)$. The authors consider polynomial basis functions identical to ours, and update the coefficients C_k for sliding time windows at time index k. They use a standard iterative solver which in general converges to a local minimum. Other methods [16], [22], [23] solve the same trajectory estimation problem, parameterizing the trajectory with B-spline basis functions. As B-splines have local support, they automatically offer more flexibility in fitting complex trajectories without recursively updating the coefficients. However, these papers lack in optimality guarantees since Gauss-Newton solvers are used. As opposed to methods solving the more general SLAM framework with arbitrary measurement modalities, we show that, by focusing on range measurements, a closed-form solution and recovery guarantees can be deduced.

Recently, trajectory estimation has been integrated in the traditional EDM framework with so-called *Kinetic EDMs* [24], where all points are considered to move on trajectories. In a similar spirit, this letter extends the traditional lateration framework to a single device moving on a trajectory and measuring ranges from fixed anchors.

IV. Problem Relaxation

In this section, we give an outline of our recovery algorithm. We first introduce the trajectory model and reformulate (2) to include it. Then, we relax the problem by reformulating it into a linear system of equations that can be solved with any linear solver. Finally, we provide intuition of why $O(K)$ measurements are enough to recover $\boldsymbol{C}$.

We assume that the robot trajectory coordinates belong to some K-dimensional linear space of functions $\mathcal{F}$:

$$\boldsymbol{r}(t) = \sum_{k=0}^{K-1} \boldsymbol{c}_k f_k(t), \tag{3}$$

where $\{f_k : k = 0, \dots, K-1\}$ is a basis for $\mathcal{F}$, and the vectors $\boldsymbol{c}_k \in \mathbb{R}^D$ are the multidimensional basis coefficients.

In this work, we focus on bandlimited functions and polynomials. Both these models can approximate naturally occurring trajectories well. For example, bandlimited trajectories describe the oscillatory motion of a body around a stationary point. Polynomials cover constant speed motion ($K = 1$), constant acceleration (e.g. free fall, $K = 2$) and linearly changing acceleration ($K = 3$). For more complex trajectories, polynomials are the essential ingredient to the commonly-used spline approximation.

For the space of bandlimited functions, we define the basis functions for odd K as

$$f_k(t) = \begin{cases} 2\cos(2\pi k t/\tau) & \text{for } k \text{ odd}, \\ 2\sin(2\pi k t/\tau) & \text{for } k \text{ even}, \quad k > 0, \\ 1 & \text{for } k = 0, \end{cases}$$

where τ is the fixed period of the trajectory. For the space of polynomials we simply use the monomial basis $f_k(t) = t^k$.

We can now reformulate (2) in terms of the coefficients $\boldsymbol{c}_k$. By setting $\boldsymbol{f}_n := [f_0(t_n) \dots f_{K-1}(t_n)]^\top$ and $\boldsymbol{C} = [\boldsymbol{c}_0 \dots \boldsymbol{c}_{K-1}] \in \mathbb{R}^{D\times K}$, we can express the sampled positions in matrix form: $\boldsymbol{r}_n = \boldsymbol{C}\boldsymbol{f}_n$. The distances thus become $d_n = \|\boldsymbol{C}\boldsymbol{f}_n - \boldsymbol{a}_{m_n}\|$ and we can reformulate (2) as

$$\begin{aligned} &\underset{\boldsymbol{C}\in\mathbb{R}^{D\times K}}{\arg\min} \sum_{n=0}^{N-1} \left(\tilde{d}_n^2 - \|\boldsymbol{a}_{m_n}\|^2 + 2\boldsymbol{a}_{m_n}^\top \boldsymbol{C}\boldsymbol{f}_n - \boldsymbol{f}_n^\top \boldsymbol{L}\boldsymbol{f}_n\right)^2, \\ &\text{s.t. } \boldsymbol{L} = \boldsymbol{C}^\top\boldsymbol{C}. \end{aligned} \tag{4}$$

Here, $\boldsymbol{L}$ is introduced to separate terms linear in $\boldsymbol{C}$ from those quadratic in $\boldsymbol{C}$. We will refer to (4) as the *SRLS trajectory recovery problem*. It is common to use semidefinite relaxations to make problems like (4) convex [25]. We take a simpler approach and discard the constraint entirely, producing what we refer to as the *relaxed SRLS trajectory recovery problem.*

Solving this relaxed problem is actually equivalent to solving a system of linear equations. To see this, let us introduce the vectorized forms $\mathrm{vec}(\boldsymbol{C})$ and $\mathrm{vec}(\boldsymbol{L})$. Since $\boldsymbol{a}_{m_n}^\top \boldsymbol{C}\boldsymbol{f}_n$ is a scalar, it is equal to its trace and thus

$$\boldsymbol{a}_{m_n}^\top(\boldsymbol{C}\boldsymbol{f}_n) = \mathrm{tr}(\boldsymbol{a}_{m_n}\boldsymbol{f}_n^\top\boldsymbol{C}^\top) = \mathrm{vec}\left(\boldsymbol{a}_{m_n}\boldsymbol{f}_n^\top\right)^\top \mathrm{vec}\left(\boldsymbol{C}\right),$$

where the first equality comes from the commutativity of the trace, $\mathrm{tr}(\boldsymbol{A}^\top\boldsymbol{B}) = \mathrm{tr}(\boldsymbol{A}\boldsymbol{B}^\top)$, and the second from the fact that $\mathrm{tr}(\boldsymbol{A}^\top\boldsymbol{B}) = \mathrm{vec}(\boldsymbol{A})^\top \mathrm{vec}(\boldsymbol{B})$. Similarly,

$$\boldsymbol{f}_n^\top\boldsymbol{L}\boldsymbol{f}_n = \mathrm{vec}\left(\boldsymbol{f}_n\boldsymbol{f}_n^\top\right)^\top \mathrm{vec}\left(\boldsymbol{L}\right).$$

Input: Anchor coordinates $\boldsymbol{a}_m$, distance measurements d_m, times and anchor indices t_n, m_n.
Output: Trajectory coefficients $\hat{\boldsymbol{C}}$, empty if not unique.

$\boldsymbol{f}_n \leftarrow \begin{bmatrix} f_0(t_n) & \dots & f_{K-1}(t_n) \end{bmatrix}$
set up $\boldsymbol{T}_A, \boldsymbol{T_F}, \boldsymbol{b}$ as in (5)
$b_n \leftarrow \left(\|a_{m_n}\|^2 - d_n^2\right)/2$
$\boldsymbol{b} \leftarrow$ concatenate(b_n)
$\boldsymbol{T}_A \leftarrow$ concatenate$(\mathrm{vec}\left(\boldsymbol{a}_{m_n}\boldsymbol{f}_n'\right)')$
$\boldsymbol{T}_F \leftarrow$ concatenate$(\mathrm{vec}\left(\boldsymbol{f}_n\boldsymbol{f}_n'\right)')$
$\hat{\boldsymbol{C}} \leftarrow [\,]$
if conditions (7) **and** (8) are satisfied **then**
  $\boldsymbol{U}, \boldsymbol{\Sigma}, \boldsymbol{V} \leftarrow \mathrm{SVD}\left(\boldsymbol{T}_F\right)$
  $\boldsymbol{A} \leftarrow \mathrm{concatenate}\left(\boldsymbol{T}_A, \boldsymbol{U}\boldsymbol{\Sigma}\right)$
  $\hat{\boldsymbol{x}} \leftarrow \mathrm{linsolve}\left(\boldsymbol{A}, \boldsymbol{b}\right)$
  $\hat{\boldsymbol{C}} \leftarrow \mathrm{reshape}\left(\hat{\boldsymbol{x}}[1\!:\!DK], (D, K)\right)$
end if

Fig. 2. Proposed algorithm for range-only continuous localization.

Using this notation, the cost in (4) is minimized by the $\boldsymbol{C}$ and $\boldsymbol{L}$ satisfying the following system of equations:

$$b_n = \begin{bmatrix} \mathrm{vec}\left(\boldsymbol{a}_{m_n}\boldsymbol{f}_n^\top\right)^\top & \mathrm{vec}\left(\boldsymbol{f}_n\boldsymbol{f}_n^\top\right)^\top \end{bmatrix} \begin{bmatrix} \mathrm{vec}\left(\boldsymbol{C}\right) \\ -\frac{1}{2}\mathrm{vec}\left(\boldsymbol{L}\right) \end{bmatrix},$$

where $b_n := \frac{1}{2}(\|\boldsymbol{a}_{m_n}\|^2 - \tilde{d}_n^2)$. By concatenating the above equations, we obtain

$$\boldsymbol{b} = \begin{bmatrix} \boldsymbol{T}_A & \boldsymbol{T}_F \end{bmatrix} \begin{bmatrix} \mathrm{vec}\left(\boldsymbol{C}\right) \\ -\frac{1}{2}\mathrm{vec}\left(\boldsymbol{L}\right) \end{bmatrix} \Big\}DK+K^2, \tag{5}$$

where the rows of $\boldsymbol{T}_A \in \mathbb{R}^{N\times DK}$ are $\mathrm{vec}(\boldsymbol{a}_{m_n}\boldsymbol{f}_n^\top)^\top$ and the rows of $\boldsymbol{T}_F \in \mathbb{R}^{N\times K^2}$ are $\mathrm{vec}(\boldsymbol{f}_n\boldsymbol{f}_n^\top)^\top$.

At first sight, it appears that, by additionally solving for $\boldsymbol{L}$, the relaxation has introduced K^2 new variables (or $K(K+1)/2$ if we enforce $\boldsymbol{L}$'s symmetry). However, as we will formally prove in Section V, the required effective increase is only of order K.

To see this, consider applying a Singular Value Decomposition (SVD) to $\boldsymbol{T}_F$: $\boldsymbol{T}_F = \boldsymbol{U}\boldsymbol{\Sigma}\boldsymbol{V}$. Here, $\boldsymbol{\Sigma}$ is an $r \times r$ diagonal matrix, with r the rank of $\boldsymbol{T}_F$. Then, (5) becomes

$$\boldsymbol{b} = \begin{bmatrix} \boldsymbol{T}_A & \boldsymbol{U}\boldsymbol{\Sigma} \end{bmatrix} \begin{bmatrix} \mathrm{vec}\left(\boldsymbol{C}\right) \\ -\frac{1}{2}\boldsymbol{V}^\top\mathrm{vec}\left(\boldsymbol{L}\right) \end{bmatrix} \Big\}DK+r. \tag{6}$$

Therefore, if we have at least $DK + r$ measurements such that $\begin{bmatrix} \boldsymbol{T}_A & \boldsymbol{U}\boldsymbol{\Sigma} \end{bmatrix}$ is full-rank, we can recover $\boldsymbol{C}$ with any linear solver. This gives rise to a simple recovery algorithm, which is summarized in Fig. 2. Note that, before solving the linear system, we check invertibility using the conditions derived in the next section.

In the polynomial case, it is easy to see that $r < 2K$. In particular, in this case, each row of $\boldsymbol{T}_F$ contains the vectorized version of samples of the outer product of a polynomial of degree $K-1$ with itself. The columns of $\boldsymbol{T}_F$ are thus samples of polynomials of degrees up to $2K-2$, and so they span (at most) a $2K-1$ dimensional space. Similar reasoning can be applied for bandlimited functions by treating them as elements of a polynomial ring, see Observation 1 from [26]. Therefore, the rank of the whole matrix is $O(K)$ and the computational complexity of the algorithm is $O(N^2\,K + K^4)$.

V. Recovery Guarantees

In this section, we prove conditions for when the linear system corresponding to the relaxed SRLS trajectory recovery problem can be solved for a unique $\boldsymbol{C}$. In the noiseless case, these conditions are sufficient for perfect trajectory recovery. To see this, note that, in the noiseless case, any minimizer of (4) is also a minimizer of (6); therefore, if the the solution of (6) is unique, it is also the unique minimizer of (4) (assuming (4) is solvable). In addition, in the noiseless case, the unique solution of the SRLS problem is also the unique solution of the RLS problem.

When there is noise on the distances, we do not obtain perfect reconstruction but the conditions for the invertibility of the linear system are still valid. This is because the matrix that is inverted does not depend on measured distances.

The main theoretical contribution of this letter is the following theorem.

Theorem 1: Let $\hat{\boldsymbol{C}}$ satisfy the relaxed SRLS trajectory recovery problem. Given N measurements from non-degenerate anchors at random times $t_0, \ldots, t_{N-1}$ sampled from a continuous distribution on $\mathcal{I}^N$, $\hat{\boldsymbol{C}}$ is unique and can be recovered with the proposed algorithm if

$$N \geq K(D+2) - 1, \tag{7}$$

and

$$\sum_{m=0}^{M-1} \min(k_m, K) \geq K(D+1), \tag{8}$$

where k_m is the number of measurements in which the m-th anchor is used. Furthermore, in the noiseless case, $\hat{\boldsymbol{C}} = \boldsymbol{C}$ and perfect trajectory recovery is achieved. Finally, if (8) is not satisfied, then $\hat{\boldsymbol{C}}$ is not unique.

Here, we call a set of anchors *non-generate* if no $D+1$ anchors lie on the same affine subspace. This assumption is only slightly stronger than the common requirement that not all anchors lie on the same affine subspace. Randomly placed anchors will satisfy this condition almost surely.

The assumption that times follow a continuous distribution is in place to ensure that the times are not adversarial, in particular that the functions f_k are unlikely to be zero. Note that the t_n do not have to be independent.

Before proving the theorem, let us try to gain some more intuition of its meaning. Condition (7) formalizes that $O(KD)$ measurements are sufficient to localize. Even without the relaxation, one would expect to need $K(D+1)$ measurements for recovery. Indeed, to recover a trajectory of complexity K, one can independently localize K points along it, and to localize a single point, we need $D+1$ distance measurements.

Condition (8) describes how measurements cannot be arbitrarily distributed between anchors. In particular, if an anchor provides more than K measurements, only the first K have an effect on uniqueness, see the examples in Fig. 3. Moreover, unique recovery is not possible with measurements from less than $D+1$ anchors..

A natural question to ask is, how likely it is to obtain a measurement set sufficient to recover $\boldsymbol{C}$. Unfortunately, the probability of the set of measurements satisfying Theorem 1 does not seem to have a closed form formula. Fortunately, it depends only on the partition of measurements between the M anchors, so it can be easily calculated numerically by counting partitions that satisfy (8), see Fig. 4. Clearly, the probability of recovering $\boldsymbol{C}$ is non decreasing, because adding a new measurement can only increase the rank of $\begin{bmatrix}\boldsymbol{T}_A & \boldsymbol{T}_F\end{bmatrix}$. In practice, we observe that the probability is already large for $K(D+2)-1$ measurements, and grows with the number of measurements to 1.

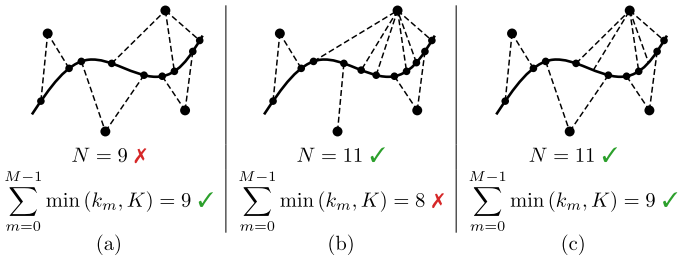


Fig. 3. Examples of sufficient and insufficient measurements, with model degree $K = 3$ and embedding dimension $D = 2$; i.e., $K(D+2) - 1 = 11$ and $K(D+1) = 9$. (a) Condition (8) is satisfied, but Condition (7) is not. (b) Condition (7) is satisfied, but Condition (8) is not because too many measurements involve the same anchor. (c) Both conditions are satisfied and recovery is guaranteed.

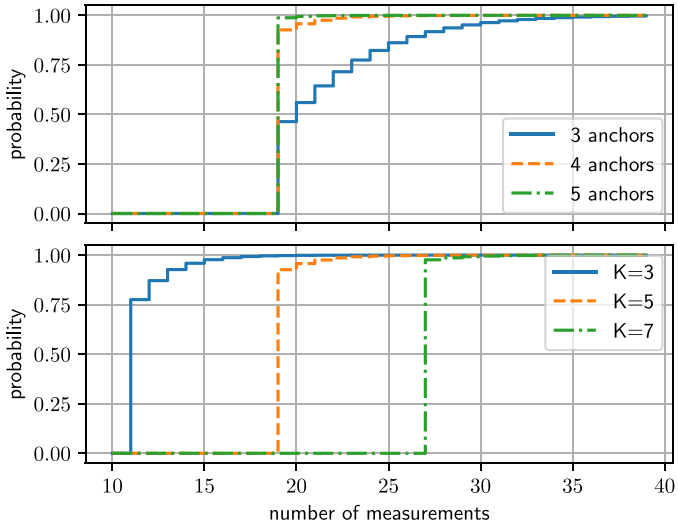


Fig. 4. Probability of recovering $\boldsymbol{C}$, with dimension $D = 2$. Upper plot: the trajectory degree is set to $K = 5$ and the number of anchors vary. We observe that the probability of recovering $\boldsymbol{C}$ grows with the number of anchors. This is because, the more anchors we use the higher the chance that the measurements spread uniformly and satisfy (8). Lower plot: the number of anchors is set to $M = 4$ and the trajectory degree varies. For the proposed algorithm (7) states that the minimum number of required measurements is $11, 19$ and 27 for trajectory complexities of $3, 5$ and 7, respectively.

Note that, in practice, the matrix might be ill conditioned when the anchors are almost co-linear, or many measurements to the same anchor are taken at almost the same time.

To conclude this section, we prove Theorem 1. We first transform the system $\begin{bmatrix}\boldsymbol{T}_A & \boldsymbol{T}_F\end{bmatrix}$ into a form that is simpler to analyse. Instead of applying the SVD, we introduce a matrix $\boldsymbol{\Psi}$ to reduce the dimensions of $\boldsymbol{L}$, thus instead of solving the linear system for $\mathrm{vec}(\boldsymbol{C})$ and $\boldsymbol{V}^\top \mathrm{vec}(\boldsymbol{L})/2$ we solve for $\mathrm{vec}(\boldsymbol{C})$ and $\boldsymbol{\Psi}\,\mathrm{vec}(\boldsymbol{L})/2$. Both formulations are equivalent because $\boldsymbol{V}^\top$ and $\boldsymbol{\Psi}$ are the same size and of full column rank. The difference is that $\boldsymbol{\Psi}$ is constructed to make a theoretical argument in the proof, while the SVD is more numerically stable and so is used in practice.

In the second step of the proof, we use the following lemma to derive (8).

Lemma 2 (Theorem 1 from [26]): Consider the set of $N = KJ$ vectors of the form $\mathrm{vec}(\boldsymbol{g}_n \boldsymbol{f}_n^\top)$. It is a basis in $\mathbb{R}^{KJ}$ if and only if no more than K group vectors $\boldsymbol{g}_n$ are equal.

In the above lemma, it is assumed that $\boldsymbol{f}_n$ and $\mathcal{F}$ are defined as in Section II with times sampled from a continuous distribution, and $\boldsymbol{g}_n$ are vectors in $\mathbb{R}^J$ that are either equal or linearly independent. More formally, the vectors $\boldsymbol{g}_n$ form a collection such that each set $\mathcal{G}$ of $\boldsymbol{g}_n$ spans a $\min(|\mathcal{G}|, J)$-dimensional subspace of $\mathbb{R}^J$.

The final step of the proof uses the following lemma to show that if $\boldsymbol{T}_A$ is invertible and (7) is satisfied then $\hat{\boldsymbol{C}}$ is unique.

Lemma 3 (Lemma 4 from [26]): Let $\boldsymbol{A}_i \in \mathbb{R}^{r \times r}$ be a full rank matrix and let $\boldsymbol{A}_{i+1}$ be a matrix constructed by first appending any column to $\boldsymbol{A}_i$ and then appending a row of the form

$$\begin{bmatrix} p_0(t) & \dots & p_{r-1}(t) & p_r(t) \end{bmatrix},$$

where $p_j \in \mathcal{R}$, $j = 0 \dots r$, is evaluated at a random time t (from a continuous distribution) and the degree of p_r in at least one of the variables is greater than the degree of any other p_j in the same variable. Then, $\boldsymbol{A}_{i+1}$ is full rank with probability one.

In the above lemma, $\mathcal{R}$ is a set of functions $\mathbb{R} \to \mathbb{R}$ with the structure of a polynomial ring, such that if $p \in \mathcal{R}$ and $p \neq 0$, then the Lebesgue measure of zeros of p is zero ($\lambda(\{t : p(t) = 0\}) = 0$).

Observe that both polynomials and bandlimited function satisfy the assumptions on $\mathcal{R}$. Bandlimited functions are associated with elements of the ring of trigonometric polynomials, $R[X, Y]/(X^2 + Y^2 - 1)$. Throughout the proof, when we refer to a *polynomial*, we mean an element of $\mathcal{R}$.

Proof of Theorem 1: The rows of $\boldsymbol{T}_F$ have the form $\mathrm{vec}(\boldsymbol{f}_n \boldsymbol{f}_n^\top)^\top$ and are evaluations of polynomials of degrees $0, \dots, 2K-2$. Thus, there is a matrix $\boldsymbol{\Psi} \in \mathbb{R}^{K^2 \times (2K-1)}$ such that $\mathrm{vec}(\boldsymbol{f}_n \boldsymbol{f}_n^\top)^\top$ equals

$$\begin{bmatrix} f_0(t_n) & \dots & f_{K-1}(t_n) & f_K(t_n) & \dots & f_{2K-2}(t_n) \end{bmatrix} \boldsymbol{\Psi},$$

where the f_k are as defined before for $k < K$, and are some polynomials of degree k for $K \leq k < 2K-1$. Therefore,

$$\mathrm{vec}\left(\boldsymbol{f}_n \boldsymbol{f}_n^\top\right)^\top = \begin{bmatrix} \boldsymbol{f}_n^\top & f_K(t_n) & \dots & f_{2K-2}(t_n) \end{bmatrix} \boldsymbol{\Psi},$$

and the right hand side of (5) can be written as

$$\begin{bmatrix} \mathrm{vec}\left(\boldsymbol{a}_{m_n} \boldsymbol{f}_n^\top\right)^\top & \boldsymbol{f}_n^\top & f_K(t_n) & \dots & f_{2K-2}(t_n) \end{bmatrix} \begin{bmatrix} \mathrm{vec}\,(\boldsymbol{C}) \\ -\frac{1}{2}\boldsymbol{\Psi}\,\mathrm{vec}\,(\boldsymbol{L}) \end{bmatrix}$$

Since $\boldsymbol{f}_n$ can be expressed as $1 \cdot \boldsymbol{f}_n$, the row vector in the equation above can be rearranged into

$$\begin{bmatrix} \mathrm{vec}\left(\begin{bmatrix} \boldsymbol{a}_{m_n} \\ 1 \end{bmatrix} \boldsymbol{f}_n^\top\right)^\top & f_K(t_n) & \dots & f_{2K-2}(t_n) \end{bmatrix}.$$

Let us now call $\tilde{\boldsymbol{T}}_A \in \mathbb{R}^{N \times (D+1)K}$ the matrix consisting of vectors $\mathrm{vec}([\boldsymbol{a}_{m_n}^\top 1]^\top \boldsymbol{f}_n^\top)^\top$ as rows, and $\tilde{\boldsymbol{T}}_F \in \mathbb{R}^{N \times K-1}$ the matrix consisting of the additional polynomials; i.e., the n-th row of $\tilde{\boldsymbol{T}}_F$ is $\begin{bmatrix} f_K(t_n) & \dots & f_{2K-2}(t_n) \end{bmatrix}$.

Observe that if any subset of rows of $\tilde{\boldsymbol{T}}_A$ is a basis in $\mathbb{R}^{K(D+1)}$, then $\tilde{\boldsymbol{T}}_A$ is invertible. We can apply Lemma 2 to any $K(D+1)$ rows of $\tilde{\boldsymbol{T}}_A$, with $\boldsymbol{g}_n^\top = [\boldsymbol{a}_{m_n}^\top\ 1]$ and $J = D+1$, because the assumptions that anchors are non-degenerate translates to the assumptions that different $\boldsymbol{g}_n$ are independent. This way we get that $K(D+1)$ rows form a basis in $\mathbb{R}^{K(D+1)}$ if and only if no more than K vectors $[\boldsymbol{a}_{m_n}^\top\ 1]$ are equal, where index n corresponds to the indexes of those $K(D+1)$ rows.

We would like to know when such a $K(D+1)$ element subset of rows exists—call it a core subset. Note that for each anchor, there can only be K rows containing this anchor in the core subset. If the m-th anchor is contained in k_m rows, it can contribute at most $\min(k_m, K)$ rows to the core subset. Therefore, the total number of core rows is limited by

$$\sum_{m=0}^{M-1} \min(k_m, K).$$

If this number is greater or equal to $K(D+1)$, then the core subset is indeed a basis, and $\tilde{\boldsymbol{T}}_A$ is full column rank.

What remains to be shown is that given $K(D+2)-1$ measurements with $\tilde{\boldsymbol{T}}_A$ full column rank, then $\begin{bmatrix} \tilde{\boldsymbol{T}}_A & \tilde{\boldsymbol{T}}_F \end{bmatrix}$ is full column rank too. To do this, we inductively apply Lemma 3.

First, since $\tilde{\boldsymbol{T}}_A$ is full rank, it has $K(D+1)$ independent rows. Without loss of generality, we can assume that those are its first $K(D+1)$ rows. We can then let $\boldsymbol{A}_0$ be the top left $K(D+1) \times K(D+1)$ submatrix of $\begin{bmatrix} \tilde{\boldsymbol{T}}_A & \tilde{\boldsymbol{T}}_F \end{bmatrix}$. For any $i \in [1 \dots K-1]$, let $\boldsymbol{A}_i$ be the top left $(K(D+1)+i) \times (K(D+1)+i)$ submatrix of $\begin{bmatrix} \tilde{\boldsymbol{T}}_A & \tilde{\boldsymbol{T}}_F \end{bmatrix}$. By Lemma 3, if $\boldsymbol{A}_i$ is full rank, then $\boldsymbol{A}_{i+1}$ is full rank with probability one. The latter is also true if $\boldsymbol{A}_i$ is full rank with probability one, since the probability of two events with probability one is still one. Therefore, since $\boldsymbol{A}_i$ is full rank, by mathematical induction, the matrix $\boldsymbol{A}_{K-1}$, consisting of the first $K(D+2)-1$ rows of $\begin{bmatrix} \tilde{\boldsymbol{T}}_A & \tilde{\boldsymbol{T}}_F \end{bmatrix}$ is full rank with probability one, so the whole matrix $\begin{bmatrix} \tilde{\boldsymbol{T}}_A & \tilde{\boldsymbol{T}}_F \end{bmatrix}$ is full *column* rank with probability one. ■

VI. Results

In the previous section, we established conditions under which unique recovery of $\boldsymbol{C}$ is possible. In this section, we compare the performance of different localization algorithms on simulated and real data.

As mentioned before, we assume additive Gaussian noise on distance measurements $\tilde{d}_n = d_n + \epsilon_n$, where ϵ_n are i.i.d. random variables, $\epsilon_n \sim \mathcal{N}(0, \sigma)$.

An immediate problem comes from the fact that we use SRLS for trajectory recovery and the distribution of *squared* distances is both additive and multiplicative:

$$\tilde{d}_n^2 = d_n^2 + 2 d_n \epsilon_n + \epsilon_n^2,$$

while our recovery method implicitly assumes only centered additive noise. Indeed, in the proposed algorithm we solve a linear system of equations, and we obtain a solution that would be a MLE if the Gaussian noise was added to the squared distances. To alleviate this problem, we propose a Weighted Least Squares (WLS) approach described below.

If the noise is small, the ϵ_n^2 term is negligible. The remaining noise has distribution $2 d_n \epsilon_n \sim \mathcal{N}(0, 2 d_n \sigma)$. If we knew d_n, we could use WLS, with weights $1/d_n$, to bring the system back to an i.i.d. noise model. Since we do not know the distances, we

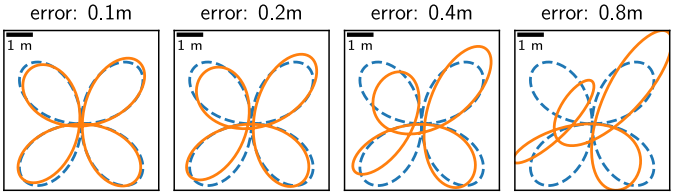


Fig. 5. Visualisation of the distance between trajectories. Original bandlimited trajectory of order $K = 7$ (dashed blue) and a randomly perturbed trajectory (solid orange). The Frobenius distance between the trajectory coefficients is displayed above each subplot.

use the *measured* distances $\tilde{d}_n$, and obtain a noisy version of the n-th row of (5):

$$\frac{d_n}{\tilde{d}_n}\epsilon_n + \frac{1}{\tilde{d}_n}\begin{bmatrix}\boldsymbol{T}_A & \boldsymbol{T}_F\end{bmatrix}_n \begin{bmatrix}\operatorname{vec}(\boldsymbol{C}) \\ -\frac{1}{2}\operatorname{vec}(\boldsymbol{L})\end{bmatrix} = \frac{b_n}{\tilde{d}_n}.$$

Again, assuming that noise is small, $d_n/\tilde{d}_n$ is close to 1, and the noise is approximately i.i.d. In practice, to avoid dividing by extremely small numbers, we can add some small regularisation γ to the distance and divide by $\tilde{d}_n + \gamma$.

A. Simulations

In this section, we report the root squared error E between the estimate $\hat{\boldsymbol{C}}$ and the ground truth $\boldsymbol{C}$:

$$E(\boldsymbol{C},\hat{\boldsymbol{C}}) = \|\boldsymbol{C} - \hat{\boldsymbol{C}}\|_F, \tag{9}$$

where $\|\cdot\|_F$ is the Frobenius norm. For the chosen basis of bandlimited functions, this norm is equivalent to the power of the signal: using power as opposed to energy makes errors comparable between trajectories with different periods. For more intuition, see Fig. 5.

In the simulations, we take samples t_n uniformly in the interval $[0, \tau]$, and at each time we choose anchors uniformly at random. Technically speaking, this violates the assumption of the random measurement times in Theorem 1. However, the assumption mostly serves for mathematical rigour and all typical sampling schemes (random, uniform, etc.) work in practice.

When simulating different values for N, we discard some measurements uniformly at random. We fix $\tau = 2$. Fig. 6 shows the coefficients reconstruction error obtained using the weighted reconstruction. We can see that the error decreases with oversampling, and the fitted slope is roughly -0.6. This means that for $10\times$ oversampling we get more than a $5\times$ reconstruction improvement. The regular (non-weighted) solver performed similarly, with a smaller improvement from oversampling.

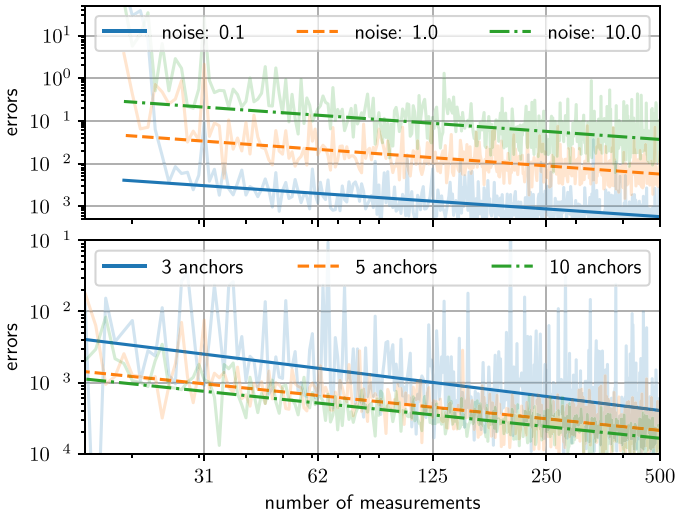


Fig. 6. Reconstruction from noisy measurements using the weighted system of equations. For clarity, slopes (dark lines) were fitted to the averages over 1000 simulations (light lines). The simulated distances were between 0.1 m and 10 m. Upper plot: the trajectory degree was set to $K = 5$ and $M = 4$ anchors were used; the magnitude of the noise is changing. In this setup, the minimum number of measurements required is 19. We can see that the algorithm is robust to noise starting from about $3\times$ oversampling. Lower plot: the trajectory degree was set to $K = 3$ and noise magnitude was set to $\sigma = 1$m; the number of anchors is changing. The reconstruction error does not depend heavily on the number of anchors, but it has much higher variance for $D + 1 = 3$ anchors.

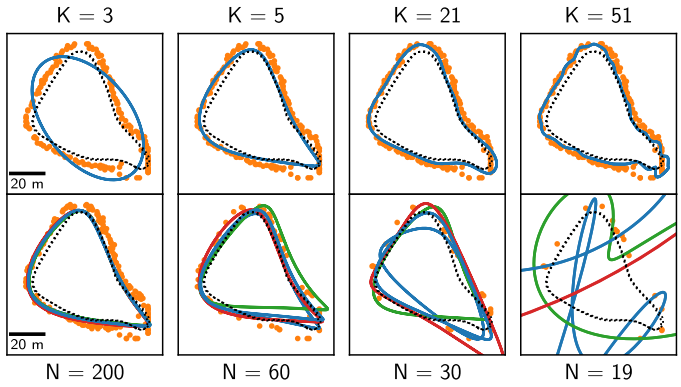


Fig. 7. Reconstruction accuracy of lawnmower trajectory using period $\tau = 54$ s. In dotted black is the ground truth trajectory (from GPS), in solid lines are our reconstructions, and the orange dots show the RLS estimates. The top row shows different complexities K using all available distance measurements. In the bottom row, we fix K to 5 and drop measurements uniformly at random. The different colors correspond to different sets of measurements.

B. Real-World Experiments

We test our trajectory estimation algorithm on two real-world datasets provided by Djugash *et al.* [27]. The datasets consist of an autonomous lawnmower moving on a grass field, using ultra-wideband (UWB) signals to 4 stationary anchors for range measurements, and densely sampled kinematic GPS for ground truth. The distance measurements have an average standard deviation of ca. 0.5m, with a tendency to overestimate [27].

We first evaluate the *Plaza2* dataset. The trajectory completed by the robot does not perfectly fit our models, but we will see that it can be approximated by the bandlimited model. We can estimate its period τ by visual inspection. Using all 499 range measurements, we use the algorithm presented in Fig. 2 to estimate the coefficients for different complexities K, and report the obtained trajectories in the first row of Fig. 7. We see that the trajectory is well approximated with degrees of $K = 5$ or higher.

Next, we fix K to the smallest sensible value for the given trajectory ($K = 5$) and test the performance of our algorithm when dropping measurements uniformly at random, down to the minimum number required (19 measurements). The second row in Fig. 7 shows that the obtained reconstruction quality remains satisfactory down to 30 measurements only, and is not too sensitive to the specific distance measurements selected. As

TABLE I
MSE OF RECOVERED TRAJECTORIES FOR TWO REAL-WORLD UWB-BASED LOCALIZATION DATASETS [27]

# measurements N	**100**			**300**			**499**			**10**	**20**	**30**	**50**
model complexity K	**5**	**11**	**19**	**5**	**11**	**19**	**5**	**11**	**19**	**2**	**2**	**2**	**2**
model mismatch	3.7	1.8	1.5	3.8	2.0	1.7	3.8	2.1	1.8	0.3	0.3	0.4	0.7
SRLS	137.5	141.0	114.5	22.2	22.2	21.8	13.2	13.2	13.2	33.8	15.1	14.2	7.8
SRLS fitted	77.9	100.5	4378.0	18.1	17.6	17.8	15.6	14.1	13.9	50.4	119.0	15.6	8.7
RLS	113.6	134.3	106.1	16.5	17.2	16.2	9.7	9.7	9.7	32.5	14.1	12.7	6.3
RLS fitted	62.1	91.5	5971.6	13.2	12.8	12.9	10.5	9.2	9.4	48.5	198.8	12.9	6.7
LM ellipse/line	12.7	147.6	339.8	11.3	25.9	13.2	11.4	11.5	12.0	253.4	24.6	16.3	6.6
LM ours weighted	12.7	13.0	20.5	11.3	11.9	12.2	11.4	11.5	12.0	18.7	7.2	7.5	6.6
ours	13.9	14.1	69.5	13.2	11.5	11.6	13.2	11.4	11.3	51.7	4.9	4.8	4.1
ours weighted	10.3	10.6	68.2	8.8	7.3	7.3	8.7	7.2	6.9	5639.1	4.9	4.8	4.3
	bandlimited trajectory (*Plaza2*)									polynomial trajectory (*Plaza1*)			

seen already in simulation, the variance of the reconstructions is higher for fewer measurements.

For comparison, we also plot localization results with the point-wise RLS method [7], using the latest distance measurements from $D+1$ anchors. The method uses brute force on a uniform grid initialized in the bounding box of the anchors with a grid size of 0.5 m. The trajectory only starts to be recognizable from $N = 200$ measurements upwards, and the individual estimates are more noisy than ours.

Finally, we compare the provided algorithms quantitatively with standard lateration methods and solvers. Table I shows the trajectory recovery accuracy for the two UWB-based datasets from [27]. As we have seen before, the trajectory in *Plaza2* is approximately bandlimited. The trajectory in *Plaza1*, on the other hand, covers the lawn in a zig-zag manner. In practice, for such a trajectory, an iterative application of our algorithm would be appropriate. For the purpose of this letter, we simply report the average reconstruction accuracy over 20 different linear parts of the trajectory. We compute the error by sampling the parametric model at the times at which we have ground truth measurements, compute the mean squared error (MSE) between the predicted and ground truth points, and average over 20 different realizations.

We evaluate the proposed solution with and without the weighting introduced at the beginning of this Section. For the *RLS* and *SRLS* methods, we compare both the raw laterated points and a parametric trajectory fitted to these points (*RLS/SRLS fitted*). The *LM* method reported for comparison solves (1) using the `scipy.optimize` [28] implementation of Levenberg-Marquardt [8] optimization. We compare two different initializations: a simple ellipse / line in the correct order of magnitude (*LM ellipse/line*) and the result of our weighted algorithm as initialization (*LM ours weighted*). The row *model mismatch* shows the error of the model fitted directly to the ground truth position measurements, and gives an upper bound on how well we can hope to do.

We can draw a number of interesting conclusions from the quantitative evaluation. First of all, we note that weighting distance measurements for real data is improves the reconstruction accuracy significantly, especially for the bandlimited trajectory (left side of Table I). We believe that this is due to a positive bias in the distance measurements; indeed, if the bias is removed by an oracle, the weighted and non-weighted algorithms perform almost identically.

We further observe that the reconstruction behaves poorly when the number of measurements is close to the limit given in Theorem 1. For the bandlimited trajectory of degree $K = 19$ this limit is 75 measurements, and $N = 100$ noisy measurements are too few to accurately estimate the trajectory. For the polynomial trajectories, we observe the same behavior for $N = 10$ and $K = 2$, where $N = 10$ is close to the minimum of 7.

Finally, we note that the accuracy of the *LM* methods depend highly on the initialization, which is a well-known limitation [9]. This shows in a big difference between the results for *LM ellipse/line* and *LM ours weighted*, which is particularly pronounced for high model orders and few measurements. Quite surprisingly, even though *LM* optimizes the RLS cost function and thus aims to recover the MLE estimate for zero-mean Gaussian noise on measurements, it does not compare favorable with our proposed algorithm. We believe that both the fact that the solution is only suboptimal and the bias in the distance measurements play against this method. However, we suggest that if *LM* is the preferred solution in a complete system for different reasons, our proposed method is an attractive candidate for an accurate initialization.

VII. CONCLUSION AND FUTURE WORK

In this work we proposed a closed-form trajectory estimation method. Even though the method is based on a relaxation, the number of required measurements stays modest. Furthermore, Theorem 1 provides recovery guarantees and the framework eliminates the impractical assumption of perfectly synchronized measurements and dense anchors deployment. We demonstrated the performance of our method both on simulated and real data, showing in particular the advantage over point-wise lateration and traditional solvers.

To the best of our knowledge, this letter presents the first recovery guarantees for trajectory estimation from range measurements. As such, we have focused on the theoretical aspects. In future, we believe that the model can be extended to build more sophisticated and practically relevant localization algorithms. One natural extension would be to apply the proposed method in spline-based approximations, and to allow for dynamically changing trajectory models. A second direction would be to incorporate measurements of different modalities (e.g. IMU) and estimate the full pose. Finally, it would be interesting to investigate if the reconstruction could be improved with standard linear regression "tricks", such as different regularization methods, different noise models, (e.g. correlated noise described by Mahalanobis distance) or different optimisation methods (e.g. random projections).

ACKNOWLEDGMENT

The authors would like to thank Ivan Dokmanić for many fruitful discussions and Martin Vetterli for his suggestions and guidance. We also thank Karen Adam and Sepand Kashani for their constructive comments on the manuscript.

Detailed contributions: Adam Scholefield designed research. Michalina Pacholska, Frederike Dümbgen, and Adam Scholefield designed the algorithm, Frederike Dümbgen and Michalina Pacholska implemented the algorithm and simulations. Frederike Dümbgen performed literature review and worked on real data. Michalina Pacholska proved algorithm properties, and analyzed simulations. Frederike Dümbgen, Michalina Pacholska, and Adam Scholefield wrote the paper. All the code used to produce the results of this letter is available at https://github.com/lcav/continuous-localization.

REFERENCES

[1] C. Cadena *et al.*, "Past, present, and future of simultaneous localization and mapping: Toward the robust-perception age," *IEEE Trans. Robot.*, vol. 32, no. 6, pp. 1309–1332, Dec. 2016.

[2] A. Bahr, J. J. Leonard, and M. F. Fallon, "Cooperative localization for autonomous underwater vehicles," *Int. J. Robot. Res.*, vol. 28, no. 6, pp. 714–728, 2009.

[3] G. Vallicrosa, P. Ridao, D. Ribas, and A. Palomer, "Active Range-Only Beacon Localization for AUV Homing," in *Proc. Int. Conf. Int. Robot. Syst.*, 2014, pp. 2286–2291.

[4] F. R. Fabresse, F. Caballero, L. Merino, and A. Ollero, "Active perception for 3D range-only simultaneous localization and mapping with UAVs," in *Proc. Int. Conf. Unm Air Syst.*, 2016, pp. 394–398.

[5] F. Dümbgen *et al.*, "Multi-Modal Probabilistic Indoor Localization on a Smartphone," in *Proc. Int. Conf. Indoor Positioning Indoor Navigation*, 2019, pp. 1–8.

[6] S. Thrun, W. Burgard, and D. Fox, *Probabilistic Robotics*. Cambridge, MA, USA: MIT press, 2005.

[7] A. Beck, P. Stoica, and J. Li, "Exact and approximate solutions of source localization problems," *IEEE Trans. Signal Process.*, vol. 56, no. 5, pp. 1770–1778, May 2008.

[8] J. J. Moré, "The Levenberg-Marquardt algorithm: Implementation and theory," in *Proc. Numer. Anal.*, 1978, pp. 105–116.

[9] D. M. Rosen, C. DuHadway, and J. J. Leonard, "A convex relaxation for approximate global optimization in simultaneous localization and mapping," in *Proc. IEEE Int. Conf. Robot. Aut*, 2015, pp. 5822–5829.

[10] I. Dokmanic, R. Parhizkar, J. Ranieri, and M. Vetterli, "Euclidean Distance Matrices: Essential theory, algorithms, and applications," *IEEE Signal Process. Mag.*, vol. 32, no. 6, pp. 12–30, Nov. 2015.

[11] S. Sremac, F. Wang, H. Wolkowicz, and L. Pettersson, "Noisy euclidean distance matrix completion with a single missing node," *J. Global Optim.*, 2019, pp. 973–1002.

[12] J. L. Blanco, J. A. Fernández-Madrigal, and J. González, "Efficient probabilistic range-only SLAM," in *Proc. Int. Conf. Int. Robot. Syst.*, 2008, pp. 1017–1022.

[13] E. Menegatti, A. Zanella, S. Zilli, F. Zorzi, and E. Pagello, "Range-only SLAM with a mobile robot and a Wireless Sensor Networks," in *Proc. IEEE Int. Conf. Robot. Autom.*, 2009, pp. 8–14.

[14] Y. Song, M. Guan, W. P. Tay, C. L. Law, and C. Wen, "UWB/LiDAR Fusion for cooperative range-only SLAM," in *Proc. IEEE Int. Conf. Robot. Autom.*, 2019, pp. 6568–6574.

[15] A. Torres-González, J. R. Martinezdedios, and A. Ollero, "Range-only SLAM for robot-sensor network cooperation," *Auton. Robots*, vol. 42, no. 3, pp. 649–663, 2018.

[16] P. Furgale, C. H. Tong, T. D. Barfoot, and G. Sibley, "Continuous-time batch trajectory estimation using temporal basis functions," *Int. J. Robot. Res.*, vol. 34, no. 14, pp. 1688–1710, 2015.

[17] C. H. Tong, P. Furgale, and T. D. Barfoot, "Gaussian Process Gauss-Newton for non-parametric simultaneous localization and mapping," *Int. J. Robot. Res.*, vol. 32, no. 5, pp. 507–525, 2013.

[18] J. Dong, M. Mukadam, B. Boots, and F. Dellaert, "Sparse Gaussian processes on matrix lie groups: A unified framework for optimizing continuous-time trajectories," *Proc. IEEE Int. Conf. Robot. Autom.*, 2018, pp. 6497–6504.

[19] T. D. Barfoot, C. H. Tong, and S. Särkkä, "Sparse Gaussian Processes on Matrix Lie Groups: A Unified Framework for Optimizing Continuous-Time Trajectories," *Rob: Sci. Syst.*, 2014.

[20] M. Kaess, H. Johannsson, R. Roberts, V. Ila, J. J. Leonard, and F. Dellaert, "iSAM2: Incremental smoothing and mapping using the bayes tree," *Int. J. Robot. Res.*, vol. 31, no. 2, pp. 216–235, 2012.

[21] T. Li, H. Chen, S. Sun, and J. M. Corchado, "Joint smoothing and tracking based on continuous-time target trajectory function fitting," *IEEE Trans Autom. Sci. Eng.*, vol. 16, no. 3, pp. 1476–1483, Jul. 2019.

[22] E. Mueggler, G. Gallego, and D. Scaramuzza, "Continuous-time trajectory estimation for event-based vision sensors," *Rob: Sci. Syst.*, 2015.

[23] D. Droeschel and S. Behnke, "Efficient continuous-time SLAM for 3D lidar-based online mapping," in *Proc. IEEE Int. Conf. Robot. Autom.*, 2018, pp. 5000–5007.

[24] P. Tabaghi, I. Dokmanić, and M. Vetterli, "Kinetic Euclidean Distance Matrices," *IEEE Trans. Signal Process.*, vol. 68, pp. 452–465, 2020.

[25] Z. Q. Luo, W. K. Ma, A. So, Y. Ye, and S. Zhang, "Semidefinite relaxation of quadratic optimization problems," *IEEE Signal Process. Mag.*, vol. 27, no. 3, pp. 20–34, May. 2010.

[26] M. Pacholska, K. Adam, A. Scholefield, and M. Vetterli, "Matrix recovery from bilinear and quadratic measurements," Feb. 2020, *arXiv:2001.04933 [eess.SP]*.

[27] J. Djugash, B. Hamner, and S. Roth, "Navigating with Ranging Radios: Five Data Sets with Ground Truth," *J. Field Robot.*, vol. 26, no. 9, pp. 689–695, 2009.

[28] P. Virtanen *et al.*, "SciPy 1.0-fundamental algorithms for scientific computing in python," Jul. 2019, *arXiv:1907.10121 [cs.MS]*.