Personal use of this material is permitted. Permission from IEEE must be obtained for all other uses, in any current or future media, including reprinting/republishing this material for advertising or promotional purposes, creating new collective works, for resale or redistribution to servers or lists, or reuse of any copyrighted component of this work in other works.
Exploration With Massive Sensor Swarms

Stephan Schlupkothen, Guido Dartmann, and Gerd Ascheid
Chair for Integrated Signal Processing Systems, RWTH Aachen University, Germany
{schlupkothen,dartmann,ascheid}@ice.rwth-aachen.de

Abstract—In this paper algorithms for wireless sensor networks (WSN) localization and tracking are presented. They are optimized for the exploration of inaccessible environments with unknown shape and dynamics. The algorithms are based on semidefinite programming (SDP) and unscented Kalman filtering (UKF). The SDP approach is derived from a recent solution for WSN localization with anchor node uncertainty, but has been formulated iteratively and allows, therefore, the utilization of massive sensors swarms, which has not been investigated so far.

Index Terms—Localization, sensors, tracking, low power, exploration

I. INTRODUCTION

Wireless Sensor Networks (WSN) have recently attracted a lot of researchers because of their flexibility and applicability in various fields. A crucial property of these sensors is their size, from which constraints on battery consumption, communication range and provided measurements are deduced. WSNs are for example very important in the field of exploration of inaccessible environments such as groundwater and other underground systems. Because of the sensor size and underground application a direct measurement via GPS or other satellite demanding techniques is often impossible. Therefore, the WSN has to have the ability to localize itself by limited information, such as range measurements. The localization and tracking problem results hence in a high-dimensional complex optimization problem and is in general non-convex.

Related Work: The problem of WSN localization has already been studied for many years but the focus lay mainly on theoretical question under which conditions a unique solution exists, e.g. [1], [2]. In [3]–[5] the first convex formulations using SDPs were proposed. Later other also analyzed the solvability with numerical optimization algorithms solvers, [6]. None of these investigated the special problem of huge moving sensor networks.

Contribution: The application scenario of our paper is the exploration of inaccessible environments, which requires a huge amount of moving sensors. Hence, this paper addresses the problem of localization and tracking of sensors, that are traveling through a pipe or tunnel like system. This further complicates the estimation since nodes with known absolute positions exist only at the in- and outflow. For this problem, we evaluated the most promising already existing SDP localization algorithm, [7], and extended it to an iterative formulation which accounts for the exponentially increasing complexity and resource demands when using huge sensors networks, and therefore also exponentially growing constraint sets. In addition, we propose a new unscented Kalman filter (UKF) tracking algorithm for this problem.

II. THE ENVIRONMENT MODEL

The model of our evaluation scenario is closely related to the underground water system. For simplicity, we assume to have a system with one inflow and only one outflow, see Figure 1a. The sensors are inserted constantly into the system to enable at one point a “steady-state” where sensors are distributed all over the system. Measurements are performed every period. Simultaneously, also two sensors are released into the system.

To evaluate the performance of the algorithms, we assume a model which is easier to represent in terms of fluid dynamics, simple to reproduce and more challenging concerning the tracking performance. It is based on a so called sinusoidal path. This means that the center of the path has a sinusoidal shape with a certain amplitude, frequency, phase offset and thickness $d_{\text{max}}$, see Figure 1b. For the sake of simplicity, the fluid is assumed to have a constant velocity profile.

III. PROBLEM FORMULATION

The mathematical problem of sensor localization with range measurements only is well known and can be written as follows, [3], [5], [7]. The problem is in general non-convex and $NP$-hard, [6]. We assume to have $k$ anchor nodes whose absolute position, $a_1, a_2, \ldots, a_k \in \mathbb{R}^2$, with $a_i^T = [x_i, y_i]$, is known. Furthermore, we want to localize $n$ moving sensors with the coordinates $x_{k+1}, x_{k+2}, \ldots, x_{k+n} \in \mathbb{R}^2$, with $x_i^T = [x_i, y_i]$. Then, the Euclidean distance between the sensor $i$ and $j$ will be denoted as $r_{ij}$ and can be written as:

$$r_{ij} = r_{ji} = \sqrt{(x_i - x_j)^2 + (y_i - y_j)^2} = \|x_i - x_j\|_2;$$

$$i, j = k + 1, k + 2, \ldots, n; (i, j) \in \mathcal{E}$$

(1)

and similar between anchor node $i$ and sensor node $j$: $r_{ij} = r_{ji} = \|a_i - x_j\|_2$, where $m = n + k$ denotes the total amount of sensors and anchor nodes. $\mathcal{E}$ is the set of all anchor to sensor and sensor to sensor distance measurements, which may also be referred to as the set of edges when viewing this as a graph-theoretic problem, i.e. $\mathcal{G} = (\mathcal{E}, \mathcal{V})$, where $\mathcal{V}$ denotes the set of vertices. The measurements in presence of noise are:

$$\hat{r}_{ij} = r_{ij} + n_{ij}.$$ 

Furthermore, $n_{ij}$ is assumed to be a Gaussian noise process. The general optimization problem can then be written as:

$$\arg\min_{\{x_i\}} \sum_{(i,j) \in \mathcal{E}} \left( \hat{r}_{ij} - \sqrt{(x_i - x_j)^2 + (y_i - y_j)^2} \right)^2$$

(2)

IV. THE SEMIDEFINITE PROGRAMMING (SDP) APPROACH

The general SDP relaxation of Problem (2) is already well known and has been derived in several manners, see [3], [4], [8]. Moreover, they are known to produce accurate results.
behavior of the ISDP. In each iteration \( (1) \), the inner loop over variable \( i \)
This modified SDP will be denoted as MSDP. In Algorithm \( (25) \), we have chosen one of the most recent and advanced formulations of the SDP relaxation that also considers the presence of noisy measurements. Here, we only want to refer the reader to the SDP formulation \( (27) \) in [7], which we have chosen for a performance evaluation which is comparable to many other formulations that have been published so far.

V. THE ITERATIVE SEMIDEFINITE PROGRAMMING (ISDP) APPROACH

Our iterative SDP algorithm is based on formulation \( (25) \) in [7], which considers the case of anchor node uncertainty, i.e. when the absolute positions of the anchor nodes are disturbed by (white) Gaussian noise. We have introduced two further constraints, that preserve the convexity of the problem and reflect the disc graph property: \( R^2 \leq \gamma_{ij} \) for \( (i, j) \notin E \) and \( r_{ij}^2 \leq \gamma_{ij} \leq R^2 \) for \( (i, j) \in E \). This means a connection between two sensors or anchor nodes exists only if their distance is less than the sensors sensing radius \( R \) and vice versa. Furthermore, the noise variances have been generalized to parameters, \( 1/\alpha_{ij} \) and \( 1/\beta_i \), that can be freely selected and allow further optimization. This is reasonable since in the case of an iterative approach the anchor node noise cannot be assumed to be Gaussian anymore. The modified objective is:

\[
\sum_{i=k+1}^{m} \sum_{j=1}^{k} \alpha_{ij} [\gamma_{ij} - 2\gamma_{ij} \delta_{ij} r_{ij}] + \sum_{i=1}^{k} \beta_i [y_{ii} - 2\alpha_i^T x_i], \quad \text{with:}
\]

\[
\alpha_{ij} := \begin{cases} 
\delta_{ij}\alpha_{ij} & \text{if } i > k, j > k \\
\delta_{ij}\alpha_{ij}/2 & \text{otherwise}
\end{cases}
\]

This modified SDP will be denoted as MSDP. In Algorithm \( (1) \), the inner loop over variable \( l \) represents the iterative behavior of the ISDP. In each iteration \( l \) we select a set of nodes \( V_{est} \) which will be estimated and the corresponding set of nodes from \( V_{det} \) that will help to translate the relative measurements into absolute positions. Based on these sets we will run the above described MSDP in every iteration \( l \) with the parameters \( \alpha_{ij}, \beta_i, q \) and \( E_k \); \( \bar{x}_{\text{est}} = \text{MSDP}(V_{\text{est}}, V_{\text{det}}, \alpha_{ij}, \beta_i, q, E_k) \). This procedure is continued until all sensor coordinates have been estimated, i.e. \( |V_{\text{det}}| = 0 \) and repeated for every time instant \( k \). Parameter \( q \geq 0 \) influences how many iterations are necessary to determine all sensor coordinates. For \( q = 0 \), only a single iteration will be run which is equivalent to the non-iterative SDP formulation. For the highest amount of iterations, it is advised to set \( q \) to the maximum value of edges between each node in \( V_{\text{est}} \) and \( V_{\text{det}} \). A reasonable guideline for the choice of \( q \) may be the minimum amount of distance measurements \( q_{\text{min}} \) that is needed in the noise free case for triangulation, i.e. in the two-dimensional case \( q_{\text{min}} = 3 \). Since the size of the SDP is determined by the size of \( V_{\text{est}} \) and \( V_{\text{det}} \), parameter \( q \) also gives a trade-off between complexity, accuracy and needed iterations. The function \( f_\alpha(l) \) and \( f_\beta(l) \) give a further degree of freedom, and allow an iteration dependent weighting between anchor nodes and sensors. Since in this approach the reliability of the estimate decreases with every iteration due to numerical inaccuracy, noise and other factors, we have chosen an declining function: \( f_\alpha(l) = 1/\ell \) and \( f_\beta(l) = 1 \). This is equivalent to the assumption that the variance of the noise process that disturbs all distance measurements increases in each iteration.

VI. UNSCENTED KALMAN FILTER TRACKING (UKF) APPROACH

Since the sensors are moving over time, a tracking approach is reasonable, despite the fact that due to increased amount of sensors the complexity rises exponentially with the size of the state vector. Furthermore, this tracking approach reduces - by
velocity estimation and prediction - flip ambiguities, i.e. non-unique solutions, which occur for example in these SDP-based algorithms. Regarding highly non-linear systems, the UKF is a promising compromise between performance, accuracy and complexity, [9]. An extended Kalman filter (EKF) is known to lack precision in this case, [10], and has, therefore, not been chosen. Because of the assumed fluid dynamics, see Section II, we have selected the following movement model, where the absolute value of the velocity \( v \) is estimated separately. It is assumed to be constant and estimated independent of the velocity angle \( \phi = \arctan(y/x) \), [11]. The state vector \( x \) is given as: \( x_k = \begin{bmatrix} x & y & v & \phi & \omega \end{bmatrix} \). Where \( x \) and \( y \) are the Cartesian coordinates of the sensors, \( v \) is the absolute value of the velocity, i.e. \( v = \sqrt{x^2 + y^2} \), and \( \omega \) is the turning rate. The state evolution and measurements are given by

\[
x_{k+1} = f(x_k) + w_k
\]

\[
= \begin{bmatrix} x + \frac{2}{3} v \sin\left(\frac{\omega T}{2}\right) \cos\left(\phi + \frac{\omega T}{2}\right) \\
y + \frac{2}{3} v \sin\left(\frac{\omega T}{2}\right) \sin\left(\phi + \frac{\omega T}{2}\right) \\
v \\
\phi + \omega T \\
\omega \end{bmatrix} + w_k
\]

\[
y_k = h(x_k) + v_k
\]

\[
= \begin{bmatrix} \sqrt{(x_{1,k} - x_{2,k})^2 + (y_{1,k} - y_{2,k})^2} \\
\vdots \\
\sqrt{(x_{i,k} - x_{j,k})^2 + (y_{i,k} - y_{j,k})^2} \\
\vdots \\
\end{bmatrix} + v_k
\]

where \( w_k \) models the process noise and is assumed to be sampled from a Gaussian noise process with zero mean and covariance \( Q \in \mathbb{R}^{5n \times 5n} \). The measurement noise is denoted as \( v_k \) and is also sampled from a Gaussian noise process with zero mean and covariance \( R \in \mathbb{R}^{|E| \times |E|} \). The measurement function \( h(x) \) provides the Euclidean distances between different sensors or between sensors and anchor nodes, i.e. for all \((i,j) \in E \), and stacks them into a single vector. Since the dimensionality of the problem changes over time - sensors may enter and/or exit at any measurement instance or simply a connection between two sensors has been lost - also the size of the covariance matrices \( Q \) and \( R \) changes. In Algorithm (2), \( n_k \) and \( |E_k| \) refers to the number of sensors and distance measurements at time instant \( k \). At this point in time \( k \), the current graph is called \( G_k = (E_k, V_k) \). The functions \( \text{ukfPredict()} \) and \( \text{ukfUpdate()} \) perform the UKF time update and measurement updates respectively, [9].

### VII. Evaluation Criteria

For detecting and exploring the sensor environment, we are mainly interested in the trajectory estimate accuracy. Therefore, we propose the following criteria, where \( \hat{x} \) denotes the estimate of \( x \) and \( x^* \) its true value:

a) The root mean square error (RMSE) is first computed for every sensor \( i \) over all time instances \( k = \{k_{\text{start}}, k_{\text{end}} + 1, \ldots, k_{\text{end}}\} \), \( K = k_{\text{end}} - k_{\text{start}} \), and then averaged over all sensors \( n \): \( \text{RMSE}_i = \frac{1}{n} \sum_{i=1}^{n} \text{RMSE}_i \), where

\[
\text{RMSE}_i = \sqrt{\frac{1}{K} \sum_{k=k_{\text{start}}}^{k_{\text{end}}} (\hat{x}_{i,k} - x^*_{i,k})^2 + (\hat{y}_{i,k} - y^*_{i,k})^2}
\]

b) After the first sensors have reached the outflow, a more accurate estimate is possible since more sensors are directly connected to anchor nodes and, thus, the amount of needed ISDP iterations is reduced. Hence, we also examine the RMSE after the first two sensors have reached the outflow, \( \text{RMSE}_{\text{out}} \).

c) In addition, we consider the distance between the estimated position and the true sensor trajectory. This is comparable to a time invariant RMSE and is independent of the current sensor position: \( \text{RMSE}_{\text{traj}} = \frac{1}{n} \sum_{i=1}^{n} \text{RMSE}_{\text{traj}}^i \), with \( l \in \{k_{\text{start}}, k_{\text{end}}\} \):

\[
\text{RMSE}_{\text{traj}}^i = \sqrt{\frac{1}{K} \sum_{k=k_{\text{start}}}^{k_{\text{end}}} \min_{l} \left\| (\hat{x}_{i,k}) - (x^*_{i,l}) \right\|_2}
\]

Where \( x^*_{i,l}, y^*_{i,l} \) denotes the sensors real, time continuous trajectory.

### VIII. Evaluation

We provide evaluation results for the sinusoidal path, see Figure 1b, with a radial extend of \( d = 35 \), amplitude \( A = 150 \), frequency \( f = 1/600 \), phase offset \( \Delta \phi = -\pi/4 \), maximal velocity \( v_{\text{max}} = 30 \) and variable sensing radius \( R \) for 1.25 periods. The simulation starts after the insertion of the first sensor pair and has been terminated after all sensors have reached the outflow and, thus, have been collected. In total

---

**Algorithm 2: The Unscented Kalman Filter Algorithm**

1. Initialize mean and covariance estimates \( \mathbf{M}_{k_{\text{start}}} \in \mathbb{R}^{5n_{\text{start}}} \) and \( \mathbf{P}_{k_{\text{start}}} \in \mathbb{R}^{5n_{\text{start}} \times 5n_{\text{start}}} \).
2. for \((k = k_{\text{start}}; k \leq k_{\text{end}}; k++)\) do
3. \( \mathbf{M}_k := \mathbf{M}_{k-1}; \mathbf{P}_k := \mathbf{P}_{k-1} \)
4. if \( G_{k-1} \neq G_k \) then
5. Determine set of removed and added sensors
6. Remove line in \( \mathbf{M}_k \) corresponding to \( v_i^- \).
7. Remove columns and rows in \( \mathbf{P}_k \)
8. Corresponding to \( v_i^- \).
9. for \((i = 1; i \leq |\mathbf{V}^+|; i++)\) do
10. Insert and initialize line in \( \mathbf{M}_k \) corresponding to \( v_i^+ \)
11. Insert and initialize columns and rows in \( \mathbf{P}_k \)
12. Corresponding to \( v_i^- \) with values from \( \mathbf{P}_0 \).
13. end
14. end
15. \( [\mathbf{M}_k, \mathbf{P}_k] := \text{ukfPredict}(\mathbf{M}_k, \mathbf{P}_k, \mathbf{Q}, f(x)) \)
16. \( [\mathbf{M}_k, \mathbf{P}_k] := \text{ukfUpdate}(\mathbf{M}_k, \mathbf{P}_k, \mathbf{H}, h(x)) \)
17. end
18. return \( \mathbf{M}_k \).
200 sensors have been simulated. For the ISDP algorithm, we chose $q = q_{\text{max}}$. In the figure legends, the following abbreviations are used:

- ISDP refers to Algorithm 1
- UKF corresponds to Algorithm 2
- SDP refers to equation (27) in [7], see also Section IV

Figure 2 and 3 show that throughout the simulations the UKF method outperforms all SDP-based ones. This is due to the fact that the UKF considers the velocity information, in contrast to the SDP-based algorithms. Moreover, especially the ISDP method suffers from error propagation because of its iterative structure. However, for large sensing radii - i.e. when the connectivity is high and less ISDP iterations are executed - the ISDP algorithm performs better than the SDP.

In our scenario, the computational complexity and resource requirements of the ISDP algorithms is significant lower compared to the SDP algorithm, see Table I. This is not only because less sensors are used in each iteration, but also because the amount of constraints scales exponentially with this. Especially linear matrix inequalities (LMI) are complex because the amount of constraints scales exponentially with its iterative structure. However, for large sensing radii - i.e. when the connectivity is high and less ISDP iterations are executed - the ISDP algorithm performs better than the SDP.

In our scenario, the computational complexity and resource requirements of the ISDP algorithms is significant lower compared to the SDP algorithm, see Table I. This is not only because less sensors are used in each iteration, but also because the amount of constraints scales exponentially with this. Especially linear matrix inequalities (LMI) are complex because the amount of constraints scales exponentially with its iterative structure. However, for large sensing radii - i.e. when the connectivity is high and less ISDP iterations are executed - the ISDP algorithm performs better than the SDP.

In our scenario, the computational complexity and resource requirements of the ISDP algorithms is significant lower compared to the SDP algorithm, see Table I. This is not only because less sensors are used in each iteration, but also because the amount of constraints scales exponentially with this. Especially linear matrix inequalities (LMI) are complex because the amount of constraints scales exponentially with its iterative structure. However, for large sensing radii - i.e. when the connectivity is high and less ISDP iterations are executed - the ISDP algorithm performs better than the SDP.

In our scenario, the computational complexity and resource requirements of the ISDP algorithms is significant lower compared to the SDP algorithm, see Table I. This is not only because less sensors are used in each iteration, but also because the amount of constraints scales exponentially with this. Especially linear matrix inequalities (LMI) are complex because the amount of constraints scales exponentially with its iterative structure. However, for large sensing radii - i.e. when the connectivity is high and less ISDP iterations are executed - the ISDP algorithm performs better than the SDP.

In our scenario, the computational complexity and resource requirements of the ISDP algorithms is significant lower compared to the SDP algorithm, see Table I. This is not only because less sensors are used in each iteration, but also because the amount of constraints scales exponentially with this. Especially linear matrix inequalities (LMI) are complex because the amount of constraints scales exponentially with its iterative structure. However, for large sensing radii - i.e. when the connectivity is high and less ISDP iterations are executed - the ISDP algorithm performs better than the SDP.

In our scenario, the computational complexity and resource requirements of the ISDP algorithms is significant lower compared to the SDP algorithm, see Table I. This is not only because less sensors are used in each iteration, but also because the amount of constraints scales exponentially with this. Especially linear matrix inequalities (LMI) are complex because the amount of constraints scales exponentially with its iterative structure. However, for large sensing radii - i.e. when the connectivity is high and less ISDP iterations are executed - the ISDP algorithm performs better than the SDP.

In our scenario, the computational complexity and resource requirements of the ISDP algorithms is significant lower compared to the SDP algorithm, see Table I. This is not only because less sensors are used in each iteration, but also because the amount of constraints scales exponentially with this. Especially linear matrix inequalities (LMI) are complex because the amount of constraints scales exponentially with its iterative structure. However, for large sensing radii - i.e. when the connectivity is high and less ISDP iterations are executed - the ISDP algorithm performs better than the SDP.

In our scenario, the computational complexity and resource requirements of the ISDP algorithms is significant lower compared to the SDP algorithm, see Table I. This is not only because less sensors are used in each iteration, but also because the amount of constraints scales exponentially with this. Especially linear matrix inequalities (LMI) are complex because the amount of constraints scales exponentially with its iterative structure. However, for large sensing radii - i.e. when the connectivity is high and less ISDP iterations are executed - the ISDP algorithm performs better than the SDP.

In our scenario, the computational complexity and resource requirements of the ISDP algorithms is significant lower compared to the SDP algorithm, see Table I. This is not only because less sensors are used in each iteration, but also because the amount of constraints scales exponentially with this. Especially linear matrix inequalities (LMI) are complex because the amount of constraints scales exponentially with its iterative structure. However, for large sensing radii - i.e. when the connectivity is high and less ISDP iterations are executed - the ISDP algorithm performs better than the SDP.

In our scenario, the computational complexity and resource requirements of the ISDP algorithms is significant lower compared to the SDP algorithm, see Table I. This is not only because less sensors are used in each iteration, but also because the amount of constraints scales exponentially with this. Especially linear matrix inequalities (LMI) are complex because the amount of constraints scales exponentially with its iterative structure. However, for large sensing radii - i.e. when the connectivity is high and less ISDP iterations are executed - the ISDP algorithm performs better than the SDP.

In our scenario, the computational complexity and resource requirements of the ISDP algorithms is significant lower compared to the SDP algorithm, see Table I. This is not only because less sensors are used in each iteration, but also because the amount of constraints scales exponentially with this. Especially linear matrix inequalities (LMI) are complex because the amount of constraints scales exponentially with its iterative structure. However, for large sensing radii - i.e. when the connectivity is high and less ISDP iterations are executed - the ISDP algorithm performs better than the SDP.