Robust MIMO Radar Target Localization based on Lagrange Programming Neural Network

Hao Wang, Chi-Sing Leung, Senior Member, IEEE, Hing Cheung So, Fellow, IEEE, Junli Liang, Senior Member, IEEE, Ruibin Feng, and Zifa Han.

Abstract—This paper focuses on target localization in a widely distributed multiple-input-multiple-output (MIMO) radar system. In this system, range measurements, which include the sum of distances between transmitter and target and the distances from the target to receivers, are used. We can obtain an accurate estimated position of the target by minimizing the measurement errors. In order to make our model closer to reality, we introduce two kinds of noises, namely, Gaussian noise and outliers. When we evaluate a target localization algorithm, its localization accuracy and robustness against noise. Hence, in the distributed variant, has many improvements in the aspect of diversity to improve its localization accuracy [2]. Compared to the distributed MIMO radar, its transmitters and receivers use waveform diversity for performance improvement. While for the two categories, namely, colocated and distributed antennas.

In this paper, we focus on the target localization problem under distributed MIMO radar system. For distributed MIMO radar system, the target localization can be obtained directly or indirectly. For direct approaches, including the maximum likelihood (ML) estimators [1], [3], the position of target is directly calculated according to the measurements collected by the antennas. These methods are based on two-dimensional (2-D) search, which requires enormous computation. On the other hand, the indirect approaches first detect the time-of-arrival (TOA) measurements, then estimate the target position according to TOAs. In this case, the ML methods can also be employed to estimate the target position. Generally speaking, ML methods solve a non-convex optimization problem [4]. Hence, the ML methods are usually transformed into the least squares (LS) method to reduce their complexity [5], [6]. The LS solutions are usually obtained in an iterative manner, and their accuracy is largely dependent on the initial position estimate. Besides, due to the property of $l_2$-norm, this method is highly sensitive to outliers. Our proposed algorithms are also based on the LS method, but we provide modifications to avoid the above mentioned disadvantages.

In this paper, we develop a robust target localization algorithms for distributed MIMO radar system based on the Lagrange programming neural network (LPNN) [2], [7]–[11]. And the $l_1$-norm or $l_0$-norm term is applied as objective function to achieve robustness against outliers. In particular, we focus on the LPNN solver to handle optimization problems with $l_1$-norm or $l_0$-norm term. However, the LPNN framework requires that its objective function and constraints should be twice differentiable. In the first proposed method, we introduce a differentiable proximate function to replace $l_1$-norm term in the objective function. While, in the second method, the internal state concept of the LCA is utilized to convert the non-differentiable components due to the $l_1$-norm or $l_0$-norm as differentiable expressions.

The rest of this paper is organized as follows. Background of MIMO radar target localization, LPNN and LCA are described in Section II. In Section III two target localization algorithms are devised. The local stability of the two approaches is proved in Section IV. Numerical results for algorithm evaluation and comparison are provided in Section V. Finally, conclusions are drawn in Section VI.

I. INTRODUCTION

Generally speaking, multiple-input multiple-output (MIMO) radar systems, use multiple antennas to transmit multiple signals and employ multiple receivers to receive the echoes from the target [1]. MIMO radar systems can be grouped into two categories, namely, colocated and distributed antennas. The former positions its antennas closely, and utilizes the waveform diversity for performance improvement. While for the distributed MIMO radar, its transmitters and receivers are widely separated with each other. It employs the spatial diversity to improve its localization accuracy [2]. Compared with traditional radar systems, the MIMO radar, especially the distributed variant, has many improvements in the aspect of localization accuracy and robustness against noise. Hence, in this paper, we focus on the target localization problem under distributed MIMO radar system.

II. BACKGROUND

A. Notation

We use a lower-case or upper-case letter to represent a scalar while vectors and matrices are denoted by bold lower-case and upper-case letters, respectively. The transpose operator is
denoted as $(\cdot)^T$. Other mathematical symbols are defined in their first appearance.

B. MIMO Radar Localization

A MIMO radar localization system [2], [12] normally includes $M$ transmitters and $N$ receivers in a 2-D space. The positions of these transmitters, receivers, and the target to be detected are expressed as $\mathbf{t}_i = [x_i, y_i]^T, i = 1, \cdots, M$, $\mathbf{r}_j = [x_j, y_j]^T, j = 1, \cdots, N$, and $\mathbf{p} = [x, y]^T$, respectively. Assume that each transmitter sends out a distinct electromagnetic wave. All these electromagnetic waves are reflected by the target, and then collected by receivers. The propagation time from the transmitter $\mathbf{t}_i$ to the target is $\tau_i$, while the propagation time between the target and the receiver $\mathbf{r}_j$ is $\tau_j$. Thus, the distance from the transmitter $\mathbf{t}_i$ to target, and that from the target to receiver $\mathbf{r}_j$ can be respectively defined as

$$
d_i = \|\mathbf{p} - \mathbf{t}_i\|_2 = \sqrt{(x'_i - x)^2 + (y'_i - y)^2}, \quad d_j = \|\mathbf{p} - \mathbf{r}_j\|_2 = \sqrt{(x'_j - x)^2 + (y'_j - y)^2}. \quad (1)
$$

The total propagation distances are

$$
d_{ij} = d_i + d_j, \quad i = 1, \cdots, M, \quad j = 1, \cdots, N. \quad (2)
$$

We see that this system needs to measure $M \times N$ distances. However, in practice, noise is almost inevitable. Therefore, the expressions of observed propagation distances are

$$
d_{ij} = c(\tau_i + \tau_j) = d_{ij} + \epsilon_{ij}, \quad (4)
$$

where $i = 1, \cdots, M, \quad j = 1, \cdots, N, \quad \epsilon_{ij}$ denotes the noise, $c$ is the speed of light. The aim of this system is to estimate the position of target $\mathbf{p}$ from $\{\mathbf{t}_i\}, \{\mathbf{r}_j\}$ and $\{d_{ij}\}$. For simplicity, most off-the-shelf algorithms directly assume that $\epsilon_{ij}$ obeys a zero-mean Gaussian distribution. While, in fact, the impulsive noise even outliers cannot be avoided in this system. For this reason, in this paper, we consider that $\epsilon_{ij}$ includes the zero-mean Gaussian white noise and some outliers.

C. Lagrange Programming Neural Network

The LPNN, introduced in [7], is an analog neural network which is able to solve the general nonlinear constrained optimization problem, given by

$$
\min_{\mathbf{z}} \quad f(\mathbf{z}) \quad \text{s.t.} \quad \mathbf{h}(\mathbf{z}) = 0, \quad (5a, 5b)
$$

where $\mathbf{z} = [z_1, \cdots, z_n]^T$ is the variable vector being optimized, $f : \mathbb{R}^n \rightarrow \mathbb{R}$ is the objective function, and $\mathbf{h} : \mathbb{R}^n \rightarrow \mathbb{R}^m (m < n)$ represents $m$ equality constraints. Both $f$ and $\mathbf{h}$ should be twice differentiable in LPNN framework. First, we set up the Lagrangian of (5):

$$
L(\mathbf{z}, \boldsymbol{\zeta}) = f(\mathbf{z}) + \boldsymbol{\zeta}^T \mathbf{h}(\mathbf{z}), \quad (6)
$$

where $\boldsymbol{\zeta} = [\xi_1, \cdots, \xi_m]^T$ is the Lagrange multiplier vector. In LPNN framework, there are $n$ variable neurons and $m$ Lagrangian neurons, which are used to hold the state variable vector $\mathbf{z}$ and the Lagrange multiplier vector $\boldsymbol{\zeta}$, respectively. The dynamics of the neurons can be expressed as

$$
\frac{d\mathbf{z}}{dt} = -\frac{\partial L(\mathbf{z}, \boldsymbol{\zeta})}{\partial \mathbf{z}}, \quad (7a)
$$

$$
\frac{d\boldsymbol{\zeta}}{dt} = \frac{\partial L(\mathbf{z}, \boldsymbol{\zeta})}{\partial \boldsymbol{\zeta}}. \quad (7b)
$$

After the neurons settle down at an equilibrium point, the output of the neurons is the solution we want. The purpose of dynamic in (7b) is to seek for a state with the minimum objective value, while (7a) aims to constrain its outputs into the feasible region. The network will settle down at a stable state if several conditions are satisfied [2], [7], [11]. Obviously, $f$ and $\mathbf{h}$ should be differentiable. Otherwise, the dynamics in (7) cannot be defined.

D. Locally Competitive Algorithm

The LCA [13] is also an analog neural network which is designed for solving the following unconstrained optimization problem:

$$
L_{\text{lca}} = \frac{1}{2} \|\mathbf{b} - \Phi \mathbf{z}\|^2_2 + \lambda \|\mathbf{z}\|_1 \quad (8)
$$

where $\mathbf{z} \in \mathbb{R}^n, \quad \mathbf{b} \in \mathbb{R}^m, \quad \Phi \in \mathbb{R}^{m \times n} (m < n)$. To construct the dynamics of LCA, we need to calculate the gradient of $L_{\text{lca}}$ with respect to $\mathbf{z}$. However, $\lambda \|\mathbf{z}\|_1$ is non-differentiable at zero point. Thus the gradient of (8) is

$$
\partial \|\mathbf{z}\|_1 = -\Phi (\mathbf{b} - \Phi \mathbf{z}) + \lambda \partial \|\mathbf{z}\|_1, \quad (9)
$$

where $\partial \|\mathbf{z}\|_1$ denotes the sub-differential of $\|\mathbf{z}\|_1$. According to the definition of sub-differential, we know that at the non-differentiable point the sub-differential is equal to a set. To handle this issue, LCA introduces an internal state vector $\mathbf{u} = [u_1, \cdots, u_n]^T$ and defines a relationship between $\mathbf{u}$ and $\mathbf{z}$.

$$
z_i = T_{\lambda}(u_i) = \begin{cases} 0, & |u_i| \leq \lambda, \\ u_i - \lambda \text{sign}(u_i), & |u_i| > \lambda. \end{cases} \quad (10)
$$

In the LCA, $\mathbf{z}$ and $\mathbf{u}$ are known as the output state variable and internal state variable vectors, respectively. $\lambda$ denotes the threshold of the function. Furthermore, from (10), we can deduce that

$$
\mathbf{u} - \mathbf{z} \in \lambda \partial \|\mathbf{z}\|_1. \quad (11)
$$

Hence, LCA defines its dynamics on $\mathbf{u}$ rather than $\mathbf{z}$ as

$$
\frac{d\mathbf{u}}{dt} = -\partial \|\mathbf{z}\|_1 = -\mathbf{u} + \mathbf{z} + \Phi^T \Phi (\mathbf{b} - \Phi \mathbf{z}). \quad (12)
$$

It is worth noting that if the dynamics of $\mathbf{z}$ are used directly, we need to calculate $\partial \|\mathbf{z}\|_1$ which is equal to a set at the zero point. Therefore, LCA uses $d\mathbf{u}/dt$ rather than $d\mathbf{z}/dt$.

In [13], a general element-wise threshold function is also proposed, which is described as

$$
z_i = T_{(\eta, \delta, \lambda)}(u_i) = \text{sign}(u_i) \frac{|u_i| - \delta \lambda}{1 + e^{-\eta (|u_i| - \lambda)}}, \quad (13)
$$

where $\lambda$ still denotes the threshold, $\eta$ is a parameter used to control the threshold transition rate, and $\delta \in [0, 1]$ indicates $\mathbb{I} = [-1, 1]$.

---

1For the absolute function $|z|$, the sub-differential $\partial |z|$ at $z = 0$ is equal to $[-1, 1]$.
Some examples of this general threshold function are given in Fig. 1. The general threshold function in (13) is used for solving the unconstrained optimization problem given by

$$L_{\text{eca}} = \frac{1}{2} \| b - \Phi z \|^2 + \lambda S_{\eta, \delta, \lambda}(z).$$

(14)

where $S_{\eta, \delta, \lambda}(z)$ is a proximate function of $L_p$-norm ($0 \leq p \leq 1$), and it has an important property:

$$\lambda \frac{\partial S_{\eta, \delta, \lambda}(z)}{\partial z} \equiv u - z.$$  

(15)

The exact form of $S_{\eta, \delta, \lambda}(z)$ cannot be obtained. However, it does not influence the application of it due to the fact that the neural dynamics are defined in terms of its gradient function rather than the penalty term itself. According to the discussion in [13], when $\eta \rightarrow \infty$ and $\delta = 0$, we obtain an ideal hard threshold function given by

$$z_i = T_{(\infty, 0, \lambda)}(u_i) = \begin{cases} 0, & |u_i| < \lambda, \\ u_i, & |u_i| > \lambda. \end{cases}$$

(16)

The corresponding penalty term is

$$\lambda S_{\infty, 0, \lambda}(z) = \frac{1}{2} \sum_{i=1}^{n} I(|z_i| > \lambda),$$

(17)

where $I(\cdot)$ denotes an indicator function. Obviously, $S_{\infty, 0, \lambda}(z)$ is a proximate function of $l_0$-norm. It is worth noting that the variables $z_i$ produced by the ideal threshold function (16) cannot take values in the range of $[-1, 0)$ and $(0, 1]$.

While, if we let $\eta \rightarrow \infty$ and $\delta = 1$, then the general threshold function is reduced to the soft threshold function, given by

$$z_i = T_{(\infty, 1, \lambda)}(u_i) = T_\lambda(u_i)$$

(18)

and the penalty term

$$\lambda S_{\infty, 1, \lambda}(z) = \lambda \| z \|_1.$$  

(19)

More details about parameter setting of the threshold function can be found in [13]. Besides, the behavior of the dynamics under different settings has been studied in [13–15].

### III. DEVELOPMENT OF PROPOSED METHOD

#### A. Problem Formulation

In traditional TOA system, it is generally assumed that the noise $e_{i,j}$ ($i = 1, \cdots, M; j = 1, \cdots, N$) follows Gaussian distribution. And according to LS method [2], [4], the problem can be formulated as

$$\min_{p} \sum_{i=1}^{M} \sum_{j=1}^{N} \left( \hat{d}_{i,j} - \| p - t_i \|_2 - \| p - r_j \|_2 \right)^2.$$  

(20)

Combining with the definitions given in (1) and (2), the problem can be rewritten as

$$\min_{p, d_i, d_j} \sum_{i=1}^{M} \sum_{j=1}^{N} \left( \hat{d}_{i,j} - d_i - d_j \right)^2,$$  

(21a)

s.t. $d_i^2 = \| p - t_i \|_2^2, \quad i = 1, \cdots, M,$  

(21b)

$d_j^2 = \| p - r_j \|_2^2, \quad j = 1, \cdots, N.$  

(21c)

It is well known that, compared with $l_2$-norm, $l_1$-norm is less sensitive to outliers, hence in our proposed model, the problem is modified as the following form:

$$\min_{p, d_i, d_j} \left\| \hat{d} - d_i - d_j \right\|_1,$$  

(23a)

s.t. $d_i^2 = \| p - t_i \|_2^2, \quad i = 1, \cdots, M,$  

(23b)

$d_j^2 = \| p - r_j \|_2^2, \quad j = 1, \cdots, N.$  

(23c)

(23) includes $M+N$ equality constraints and $M+N$ inequality constraints. Since LPNN can handle the problem with equality constraint only, the inequality constraints in (23) should be removed. To achieve this purpose, we use the following proposition.

**Proposition 1:** The optimization problem in (23) is equivalent to

$$\min_{p, d_i, d_j} \left\| \hat{d} - d_i - d_j \right\|_1,$$  

(24a)

s.t. $d_i^2 = \| p - t_i \|_2^2, \quad i = 1, \cdots, M,$  

(24b)

$d_j^2 = \| p - r_j \|_2^2, \quad j = 1, \cdots, N.$  

(24c)
Proof: Obviously, to simplify the optimization problem to , we need to prove that the inequality constraints in are unnecessary. Suppose that \((p^0, d_1^0, ..., d_M^0, d_1^-, ..., d_N^-)\) is the optimal solution of . According to the reverse triangle inequality, we see that

\[
\sum_{i=1}^{M} \sum_{j=1}^{N} |\hat{d}_{i,j} - d_i^* - d_j^*| \geq \sum_{i=1}^{M} \sum_{j=1}^{N} |\hat{d}_{i,j} - |d_i^*| - |d_j^*||, \quad (25)
\]

As \(\hat{d}_{i,j}\) is distance, all of them must be nonnegative. Thus, (25) can be rewritten as

\[
\sum_{i=1}^{M} \sum_{j=1}^{N} |\hat{d}_{i,j} - d_i^* - d_j^*| \geq \sum_{i=1}^{M} \sum_{j=1}^{N} \hat{d}_{i,j} - |d_i^*| - |d_j^*|, \quad (26)
\]

The inequality in (26) implies that the optimal solution \((p^*, d_1^*, ..., d_M^*, d_1^-, ..., d_N^-)\) of is greater or equal to the solution achieved by the feasible point \((p^0, d_1^0, ..., d_M^0, d_1^-, ..., d_N^-)\). Since \((p^*, d_1^*, ..., d_M^*, d_1^-, ..., d_N^-)\) is the optimal solution, the equality of (26) must be held. Thus \(d_i^* = |d_i^*|\) for \(\forall i \in [1, ..., M]\) and \(d_j^* = |d_j^*|\) for \(j \in [1, ..., N]\). In other words, \(d_i^* \geq 0\) for \(i \in [1, ..., M]\) and \(d_j^* \geq 0\) for \(j \in [1, ..., N]\). Hence, we can remove the inequality constraints in .}

In order to facilitate the analysis of this paper, we introduce a dummy variable \(z\) and rewrite (24) as

\[
\min_{p, z, d_1^-, d_j^+} \|z\|_1, \quad (27a)
\]

s.t.

\[
z = \bar{d} - d^+ - d', \quad (27b)
\]

\[
d_i^2 = \|p - t_i\|_2^2, \quad i = 1, ..., M, \quad (27c)
\]

\[
d_j^2 = \|p - r_j\|_2^2, \quad j = 1, ..., N. \quad (27d)
\]

where \(z = [z_1, ..., z_{M,1}, z_1, ..., z_{M,2}, ..., z_1, ..., z_{M,N}]^T\), is a vector with \(M \times N\) elements.

B. LPNN for MIMO Radar Localization

To obtain a real-time solution of the problem given in (27), we consider using the LPNN framework. However, LPNN requires its objective function and constraints are all twice differentiable. Obviously, due to the \(l_1\)-norm term, the objective function in (27) does not satisfy this requirement. Hence, for calculating the gradient of \(l_1\)-norm at the non-differentiable point, we propose two approaches.

**Method 1:** Intuitively, we consider using a differentiable \(l_1\)-norm proximate function [16]:

\[
g(x) = \frac{\ln(\cosh(ax))}{a}, \quad (28)
\]

where \(a > 1\) is a scalar. Fig 2 shows the shapes of \(g(x)\) with different \(a\). It is observed that the shape of the proximate function is quite similar with \(l_1\)-norm when \(a\) is large.

With this \(l_1\)-norm proximate function, the problem in can be rewritten as

\[
\min_{p, z, d_1^-, d_j^+} \sum_{k=1}^{MN} \frac{1}{a} \ln(\cosh(az_k)), \quad (29a)
\]

s.t.

\[
z = \bar{d} - d^+ - d', \quad (29b)
\]

\[
d_i^2 = \|p - t_i\|_2^2, \quad i = 1, ..., M, \quad (29c)
\]

\[
d_j^2 = \|p - r_j\|_2^2, \quad j = 1, ..., N. \quad (29d)
\]

After the modification, the objective function is differentiable. It is worth noting that the gradient of \((1/a) \sum_{k=1}^{MN} \ln(\cosh(az_k))\) with respect to \(z_k\) is equal to \(\tanh(az_k)\) (the hyperbolic tangent function) which is frequently used as an activation function in artificial neural networks.

Theoretically, we can directly use (29) to deduce the LPNN dynamics. While, our preliminary simulation results show that the neural dynamics may not be stable. Hence, we introduce several augmented terms into its objective (29a), after that we have

\[
\min_{p, z, d_1^-, d_j^+} \sum_{k=1}^{MN} \frac{1}{a} \ln(\cosh(az_k)) + C\|z - \bar{d} + d^+ + d'\|_2^2
\]

s.t.

\[
z = \bar{d} - d^+ - d', \quad (30a)
\]

\[
d_i^2 = \|p - t_i\|_2^2, \quad i = 1, ..., M, \quad (30b)
\]

\[
d_j^2 = \|p - r_j\|_2^2, \quad j = 1, ..., N. \quad (30c)
\]

where \(C\) is a scalar used for regulating the magnitude of these augmented terms. For any point in feasible region, the constraints must be satisfied. Hence the augmented terms equal 0 at an equilibrium point. They can improve the stability and convexity of the dynamics and do not influence the optimal
solution. The Lagrangian of (30) is

\[
L(p, z, d_i^t, d_j^t, \alpha, \beta, \lambda) = \sum_{k=1}^{MN} \frac{1}{2k} \ln(\cosh(az_k)) + \alpha^T(z - \hat{d} + d' + d'') + \sum_{i=1}^{M} \beta_i(d_i^t - \|p - t_i\|^2) + \sum_{j=1}^{N} \lambda_j(d_j^t - \|p - r_j\|^2) + \frac{C}{2} \|z - \hat{d} + d' + d''\|^2 + \frac{C}{2} \sum_{i=1}^{M} (d_i^t - \|p - t_i\|^2)^2 + \frac{C}{2} \sum_{j=1}^{N} (d_j^t - \|p - r_j\|^2)^2 
\]

(31)

where \(p, z, d_i^t(i = 1, \cdots, M), d_j^t(j = 1, \cdots, N)\) are state variable vectors, \(\alpha = [\alpha_1, \ldots, \alpha_{MN}]^T, \beta = [\beta_1, \ldots, \beta_M]^T\) and \(\lambda = [\lambda_1, \ldots, \lambda_N]^T\) are Lagrangian variable vectors. According to this Lagrangian and the concepts of LPNN given in (7), its dynamics are defined as:

\[
\frac{dz}{dt} = - \frac{\partial L(p, z, d_i^t, d_j^t, \alpha, \beta, \lambda)}{\partial z} = - \tanh(az) - \alpha - C(z - \hat{d} + d' + d''), \quad (32)
\]

\[
\frac{dd_i^t}{dt} = - \frac{\partial L(p, z, d_i^t, d_j^t, \alpha, \beta, \lambda)}{\partial d_i^t} = - \sum_{j=1}^{N-1} \alpha_{i+jM} - 2\beta_i d_i^t - C \sum_{j=1}^{N} (z_{i,j} - \hat{d}_{i,j} + d_i^t + d_j^t) - 2C \frac{d_i^t}{2} \|p - t_i\|^2, \quad (33)
\]

\[
\frac{dd_j^t}{dt} = - \frac{\partial L(p, z, d_i^t, d_j^t, \alpha, \beta, \lambda)}{\partial d_j^t} = - \sum_{i=1}^{M} \alpha_{i+(j-1)M} - 2\lambda_j d_j^t - C \sum_{i=1}^{M} (z_{i,j} - \hat{d}_{i,j} + d_i^t + d_j^t) - 2C \frac{d_j^t}{2} \|p - r_j\|^2, \quad (34)
\]

\[
\frac{dp}{dt} = - \frac{\partial L(p, z, d_i^t, d_j^t, \alpha, \beta, \lambda)}{\partial p} = \sum_{i=1}^{M} 2\beta_i(p - t_i) + \sum_{j=1}^{N} 2\lambda_j(p - r_j) + 2C \sum_{i=1}^{M} (p - t_i)(d_i^t - \|p - t_i\|^2) + 2C \sum_{j=1}^{N} (p - r_j)(d_j^t - \|p - r_j\|^2), \quad (35)
\]

In the above dynamics, \(i = [1, \cdots, M]\) and \(j = [1, \cdots, N]\). Equations (32)-(35) are used for seeking the minimum objective value, while (36)-(38) can restrict the equilibrium point into the feasible region.

Method 2: In this method, we introduce LCA into LPNN framework to solve sub-differentiable problem in (27). First, we also introduce several augmented terms into the objective function to make the system more stable. Thus, (27) can be modified as:

\[
\min_{p, z, d_i^t, d_j^t} \|z\|_1 + \frac{C}{2} \|z - \hat{d} + d' + d''\|^2 + \frac{C}{2} \sum_{i=1}^{M} (d_i^t - \|p - t_i\|^2)^2 + \frac{C}{2} \sum_{j=1}^{N} (d_j^t - \|p - r_j\|^2)^2, \quad (39a)
\]

s.t. \(z = \hat{d} - d' - d'', \quad (39b)\)

\[
d_i^t = \|p - t_i\|^2, \quad i = 1, \cdots, M, \quad (39c)\]

\[
d_j^t = \|p - r_j\|^2, \quad j = 1, \cdots, N. \quad (39d)\]

Its Lagrangian is given by

\[
L(p, z, d_i^t, d_j^t, \alpha, \beta, \lambda) = \|z\|_1 + \alpha^T(z - \hat{d} + d' + d'') + \sum_{i=1}^{M} \beta_i(d_i^t - \|p - t_i\|^2) + \sum_{j=1}^{N} \lambda_j(d_j^t - \|p - r_j\|^2) + \frac{C}{2} \|z - \hat{d} + d' + d''\|^2 + \frac{C}{2} \sum_{i=1}^{M} (d_i^t - \|p - t_i\|^2)^2 + \frac{C}{2} \sum_{j=1}^{N} (d_j^t - \|p - r_j\|^2)^2 \quad (40)
\]

According the concept of LCA, we introduce an internal variables \(u = [u_1, \ldots, u_{MN}]^T\). Combining the dynamics of LPNN in (7) with the concept of LCA in (12), we can deduce
that:

\[
\begin{align*}
\frac{du}{dt} &= -\frac{\partial L(p, z, d_i, d_j, \alpha, \beta, \lambda)}{\partial z} \\
&= \frac{u + z - \alpha - C(z - \hat{d} + d' + \lambda' - \hat{d})}{\hat{d}}. \quad (41)
\end{align*}
\]

\[
\begin{align*}
\frac{dd_t^i}{dt} &= -\frac{\partial L(p, z, d_i, d_j, \alpha, \beta, \lambda)}{\partial d_t^i} \\
&= -\sum_{j=1}^{N-1} \alpha_{i+j, M} - 2\beta_t d_t^i \\
&= -C \sum_{j=1}^{N} (z_{i,j} - \hat{d}_{i,j} + d_t^i + d_j^i) \\
&= -2Cd_t^i(d_t^i - \|p - t\|_2^2). \quad (42)
\end{align*}
\]

\[
\begin{align*}
\frac{dd_j^i}{dt} &= -\frac{\partial L(p, z, d_i, d_j, \alpha, \beta, \lambda)}{\partial d_j^i} \\
&= -\sum_{i=1}^{M} \alpha_{i+(j-1), M} - 2\lambda_t d_j^i \\
&= -C \sum_{j=1}^{N} (z_{i,j} - \hat{d}_{i,j} + d_t^i + d_j^i) \\
&= -2Cd_j^i(d_j^i - \|p - r_j\|_2^2). \quad (43)
\end{align*}
\]

\[
\begin{align*}
\frac{dp}{dt} &= -\frac{\partial L(p, z, d_i, d_j, \alpha, \beta, \lambda)}{\partial p} \\
&= \sum_{i=1}^{M} 2\beta_t (p - t_i) + \sum_{j=1}^{N} 2\lambda_t (p - r_j) \\
&= +2C \sum_{i=1}^{M} (p - t_i)(d_t^i - \|p - t_i\|_2^2) \\
&= +2C \sum_{i=1}^{M} (p - r_j)(d_j^i - \|p - r_j\|_2^2). \quad (44)
\end{align*}
\]

\[
\begin{align*}
\frac{d\alpha}{dt} &= -\frac{\partial L(p, z, d_i, d_j, \alpha, \beta, \lambda)}{\partial \alpha} \\
&= z - \hat{d} + d' + \lambda' - \hat{d'} \\
&= \frac{\frac{d\alpha}{dt}}{\hat{d}}. \quad (45)
\end{align*}
\]

\[
\begin{align*}
\frac{d\beta_t}{dt} &= -\frac{\partial L(p, z, d_i, d_j, \alpha, \beta, \lambda)}{\partial \beta_t} \\
&= \frac{d_t^i - \|p - t\|_2^2}{\hat{d}}. \quad (46)
\end{align*}
\]

\[
\begin{align*}
\frac{d\lambda_j}{dt} &= -\frac{\partial L(p, z, d_i, d_j, \alpha, \beta, \lambda)}{\partial \lambda_j} \\
&= \frac{d_j^i - \|p - r_j\|_2^2}{\hat{d}}. \quad (47)
\end{align*}
\]

In above dynamics, \(i = \{1, \cdots, M\}\) and \(j = \{1, \cdots, N\}\). It is worth noting that the relationship between \(u\) and \(z\) is given by (13). When we set \(\eta = \alpha, \delta = 1\) and \(\lambda = 1\), the mapping given by (13) becomes the soft threshold and \(u - z \in \partial \|z\|_1\). However, in order to further reduce the influence of outliers, we replace the \(l_1\)-norm term in the objective function of (39) by \(S_{10000,0,1}(x)\), thus the function can be modified as

\[
\begin{align*}
\min_{u, z, d_t^i, d_j^i} S_{10000,0,1}(z) + \frac{C}{2}\|z - \hat{d} + d' + \lambda' - \hat{d}'\|^2 \\
&+ \n frac{\frac{1}{2}}{2} \sum_{i=1}^{M} \left(\|d_t^i - \|p - t\|_2^2\|^2\right) \\
&+ \frac{\frac{1}{2}}{2} \sum_{j=1}^{N} \left(\|d_j^i - \|p - r_j\|_2^2\|^2\right). \quad (48a)
\end{align*}
\]

s.t.

\[
\begin{align*}
&\|z - \hat{d} + d' + \lambda' - \hat{d}'\|_2 = \|d_t^i - \|p - t\|_2^2\|_2, \quad i = 1, \cdots, M, \quad (48b)
\end{align*}
\]

\[
\begin{align*}
&\|d_j^i - \|p - r_j\|_2^2\|_2, \quad j = 1, \cdots, N. \quad (48d)
\end{align*}
\]

For this problem we can also solve it with LPNN and LCA. The dynamics are same as (41)-(47). The relationship between \(u\) and \(z\) is still given by (13). While in this case, we set \(\eta = 10000\), \(\delta = 0\) and \(\lambda = 1\). The threshold function given in (13) becomes a proximate hard threshold and \(\partial S_{10000,0,1}(x) = u - z\).

For both two methods mentioned above, the dynamics are updated with the following rule,

\[
\begin{align*}
\frac{du}{dt} &= \frac{u^{t+1}}{u^{t}} + \mu_1 \frac{du^{(t)}}{dt}, \quad (49)
\end{align*}
\]

\[
\begin{align*}
\frac{dd_t^i}{dt} &= \frac{d_t^{i+1}}{d_t^{i}} + \mu_2 \frac{dd_t^{i}}{dt}, \quad i = 1, \cdots, M, \quad (50)
\end{align*}
\]

\[
\begin{align*}
\frac{dd_j^i}{dt} &= \frac{d_j^{i+1}}{d_j^{i}} + \mu_3 \frac{dd_j^{i}}{dt}, \quad j = 1, \cdots, N, \quad (51)
\end{align*}
\]

\[
\begin{align*}
\frac{dp}{dt} &= \frac{p^{t+1}}{p^{t}} + \mu_4 \frac{dp^{t}}{dt}, \quad (52)
\end{align*}
\]

\[
\begin{align*}
\frac{d\alpha}{dt} &= \frac{\alpha^{t+1}}{\alpha^{t}} + \mu_5 \frac{d\alpha^{t}}{dt}, \quad (53)
\end{align*}
\]

\[
\begin{align*}
\frac{d\beta_j^i}{dt} &= \frac{\beta_i^{t+1}}{\beta_i^{t}} + \mu_6 \frac{d\beta_i^{t}}{dt}, \quad i = 1, \cdots, M, \quad (54)
\end{align*}
\]

\[
\begin{align*}
\frac{d\lambda_j}{dt} &= \frac{\lambda_j^{t+1}}{\lambda_j^{t}} + \mu_7 \frac{d\lambda_j^{t}}{dt}, \quad j = 1, \cdots, N. \quad (55)
\end{align*}
\]

where \(^{(t)}\) corresponds to the estimate at the \(t\)th iteration and \(\mu_1, \mu_2, \mu_3, \mu_4, \mu_5, \mu_6, \mu_7\) are the step sizes which should be positive and not too large to avoid divergence.

A typical example of the dynamics of the second method is given in Fig. [3] The settings are described in the second experiment of Section [VI] From Fig. [3] we see that the network can settle down within around 50 characteristic times.

**IV. LOCAL STABILITY OF PROPOSED ALGORITHMS**

In this section, we prove the local stability of the proposed methods. Local stability means that a minimum point should be stable. Otherwise, the network never converges to the minimum.

For method 1, we let \(x = [x^T, p^T, d_1^T, \cdots, d_M^T, d_1^T, \cdots, d_N^T]^T\) be the decision variable vector and \([x^*, \alpha^*, \beta^*, \lambda^*] \) be a minimum point of the dynamics given by (39)-(48). According to Theorem 1 in [7], there are two sufficient conditions for local stability in the LPNN approach. The first one is convexity, i.e., the Hessian matrix of the Lagrangian at \([x^*, \alpha^*, \beta^*, \lambda^*] \) should be positive definite. It has been achieved by introducing the augmented terms. When \(C\) is large enough, at the minimum
vectors of the constraints with respect to the decision variable $\alpha$. In our case, we have $\alpha = \beta_1, \ldots, \beta_4$; (b) $\lambda_1, \ldots, \lambda_4$.

point the Hessian matrix is positive definite under mild conditions [2, 7], [9–11].

The second one is that at the minimum point, the gradient vectors of the constraints with respect to the decision variable vector should be linearly independent. In our case, we have $MN + M + N$ constraints, namely,

\[
\begin{align*}
 h_{M(j-1)+i}(p, z, d_1, d_2) &= z_{M(j-1)+i} - d_{i,j} - d_1 - d_2 \\
 h_{MN+i}(p, z, d_1, d_2) &= d_1^2 - \|p - t\|^2, \\
 h_{MN+M+j}(p, z, d_1, d_2) &= d_2^2 - \|p - r_j\|^2,
\end{align*}
\]

where $i = 1, \ldots, M$ and $j = 1, \ldots, N$. The gradient vectors of these constraints at the minimum point are given by

\[
\begin{bmatrix}
 \frac{\partial h_1(x^*)}{\partial p} \\
 \frac{\partial h_1(x^*)}{\partial z} \\
 \frac{\partial h_1(x^*)}{\partial d_1} \\
 \vdots \\
 \frac{\partial h_M(x^*)}{\partial d_1} \\
 \frac{\partial h_1(x^*)}{\partial d_M} \\
 \vdots \\
 \frac{\partial h_M(x^*)}{\partial d_M} \\
 \frac{\partial h_{MN+M+N}(x^*)}{\partial d_1} \\
 \vdots \\
 \frac{\partial h_{MN+M+N}(x^*)}{\partial d_M} \\
 \frac{\partial h_{MN+M+N}(x^*)}{\partial d_N}
\end{bmatrix}
\]

\[
\begin{bmatrix}
 \frac{\partial h_1(x^*)}{\partial p} \\
 \frac{\partial h_1(x^*)}{\partial z} \\
 \frac{\partial h_1(x^*)}{\partial d_1} \\
 \vdots \\
 \frac{\partial h_M(x^*)}{\partial d_1} \\
 \frac{\partial h_1(x^*)}{\partial d_M} \\
 \vdots \\
 \frac{\partial h_M(x^*)}{\partial d_M} \\
 \frac{\partial h_{MN+M+N}(x^*)}{\partial d_1} \\
 \vdots \\
 \frac{\partial h_{MN+M+N}(x^*)}{\partial d_M} \\
 \frac{\partial h_{MN+M+N}(x^*)}{\partial d_N}
\end{bmatrix}
\]

where $0 = [0, 0]^T$. We can see there are $MN + M + N$ gradient vectors, each has $MN + M + N + 2$ elements. The first $MN$ columns are the gradient vectors of equality (59). Obviously, these columns are independent with each other. Similarly, we can see that the first $M$ middle columns are the gradient vectors of equality (59), and the last $N$ columns are the gradient vectors of equality (59). Both of them are independent within themselves. For the constraints given in (56), their gradients with respect to $p^*$ are all equal to 0. While for the constraints given in (57) and (58), as long as the position of the target to be detected does not coincide with any transmitter or receiver, their gradients with respect to $p^*$ cannot be zero. Hence the first $MN$ columns are independent with the last $M + N$ columns. Besides, it is easy to notice that the middle $M$ columns are independent with the last $N$ columns. Thus, the gradients of the constraints are linearly independent, and the dynamics around a minimum point are stable.

For method 2, $x = [u^T, p^T, d_1, \ldots, d_M, d_1^*, \ldots, d_M^*, \alpha^*, \beta^*, \lambda^*]^T$. We need to show that, at a local minimum point $[x^*, \alpha^*, \beta^*, \lambda^*]^T$, the gradient vectors of constraints are linearly independent. The proof process is basically similar with the first method, except that the decision variable $z$ is replaced by $u$. Thus, the
If we assume

$$\begin{bmatrix}
\frac{\partial h_1(x^*)}{\partial d_1^1} & \cdots & \frac{\partial h_1(x^*)}{\partial d_M^1} \\
\frac{\partial h_1(x^*)}{\partial d_1^2} & \cdots & \frac{\partial h_1(x^*)}{\partial d_M^2} \\
\vdots & \ddots & \vdots \\
\frac{\partial h_1(x^*)}{\partial d_1^N} & \cdots & \frac{\partial h_1(x^*)}{\partial d_M^N}
\end{bmatrix}
= \begin{bmatrix}
\frac{\partial h_{MN+N}(x^*)}{\partial d_1^1} & \cdots & \frac{\partial h_{MN+N}(x^*)}{\partial d_M^1} \\
\frac{\partial h_{MN+N}(x^*)}{\partial d_1^2} & \cdots & \frac{\partial h_{MN+N}(x^*)}{\partial d_M^2} \\
\vdots & \ddots & \vdots \\
\frac{\partial h_{MN+N}(x^*)}{\partial d_1^N} & \cdots & \frac{\partial h_{MN+N}(x^*)}{\partial d_M^N}
\end{bmatrix}$$

where $k = M(j-1) + i$.

$$g_k = \frac{\partial h_k(x^*)}{\partial z_k} \frac{\partial z_k}{\partial u_k} = \frac{1}{1 + \exp(-\eta(|u_k| - 1))} + \frac{\eta(|u_k| - \delta) \exp(-\eta(|u_k| - 1))}{(1 + \exp(-\eta(|u_k| - 1)))^2}.$$ 

For the case with $L_1$-norm objective function, $\eta \to \infty$, $\delta = 1$. If we assume $z_i \neq 0$, in other words, all data points are influenced by noise, thus we have $g_k = 1$ for $\forall k = 1, \ldots, MN$. When the proximate $L_0$-norm objective function is used, we let $\eta$ be a large positive number, $\delta = 0$. Without any assumption, we can deduce that, for $\forall k = 1, \ldots, MN$, $g_k$ is a positive constant. Next, similar with the proof process of method 1, we can prove that a minimum point of the second method has local stability.

V. Numerical Examples

In this section, we conduct several simulations and experiments to test the performance of our proposed algorithms. First, we discuss the parameter settings and the initialization. In our MIMO radar localization system, 4 transmitters and 4 receivers are used, i.e., $M = N = 4$. Their positions are $t_1 = [-5000, 6000]^T m, t_2 = [0, 7500]^T m, t_3 = [10500, 0]^T m, t_4 = [6000, 4000]^T m$ and $r_1 = [-10000, -6000]^T m, r_2 = [-9000, 5000]^T m, r_3 = [0, 4200]^T m, r_4 = [6400, -8000]^T m$, respectively. The true position of the target is $p = [-2000, 1000]^T m$. The geometry of transmitters, receivers and target are shown in Fig. 4.

For the proposed approaches, we set $C = 20$ and step sizes $\mu_1 = \mu_2 = \mu_5 = \mu_6 = \mu_7 = 10^{-3}, \mu_2 = (\mu_0) = 10^{-5}$. The initial values of variables $p, \epsilon, d_i^j (i = 1, \ldots, M), d_j^i (j = 1, \ldots, N), \alpha = [\alpha_1, \ldots, \alpha_{MN}], \beta = [\beta_1, \ldots, \beta_M], \lambda = [\lambda_1, \ldots, \lambda_N]$ are some small random values. And we set $\alpha = 50$ in the first method. Two-state-of-the-art algorithms are implemented for performance comparison. They are the target localization algorithm described in [2] and the robust target localization algorithm given by [12]. The former is also based on LPNN framework, but it assumes that the noise follows Gaussian distribution and uses $l_2$-norm in its objective function. While, the latter is a robust target localization algorithm for MIMO radar system. It introduces the maximum correntropy criterion (MCC) into the conventional ML method to deal with outliers, and apply half-quadratic optimization technique to handle the corresponding nonconvex nonlinear function.

A. Experiment 1: Target Localization in Gaussian Noise

In the first experiment we test the root mean squared error (RMSE) performance of the proposed algorithms under Gaussian noise environment without introducing any outliers. The standard deviation of the Gaussian noise varies from 1 to 10. For each noise level, we repeat the experiment 100 times. The results are shown in Fig. 5.

In Fig. 5 CRLB is short for Cramér-Rao lower bound, which denotes a lower bound on the variance of any unbiased estimator [2]: $l_2$-norm LPNN denotes the algorithm given by [2]; $l_1$-norm-like LPNN is our proposed method 1; $l_1$-norm LPNN LCA and $l_0$-norm LPNN LCA represent our proposed method 2 with $l_1$-norm objective function and $l_0$-norm objective function respectively; MCC is the robust algorithm given in [12]. In Fig. 5 we see that the performance of our proposed...
two robust algorithms is closed to the CRLB in Gaussian noise environment. They are better than the robust algorithm given by [12] and are slightly inferior to the algorithm given by [2], which is based on the Gaussian noise model.

B. Experiment 2: Target Localization in Gaussian Noise with NLOS Outliers

The basic setting of the second experiment is similar with the first one, but we fix the variance of Gaussian noise to 100, and introduce outliers into the measurement matrix. We assume that there exists non-line-of-sight (NLOS) propagation between one transmitter and the target or between the target and one receiver. Thus all measurements associated with this transmitter or receiver include NLOS outliers [17], [18]. In this experiment, we randomly choose one of the transmitters or receives and add NLOS outliers into its relevant measured values. In other words, the measurements can be seen as a 4 x 4 matrix, one of columns or rows is influenced by NLOS outliers. The outliers are generated by exponential distribution. Then we conduct the experiments under different outlier levels. For each different outlier level, we also repeat the experiment 100 times. The results are shown in Fig. 6.

From Fig. 6 we see that our first method and second method with $l_1$-norm objective have decent performance when the outlier level (standard deviation of the exponential distribution) is less than $10^{4.5}$. Both of them break down when the outlier level is $10^5$. While the second method with $l_0$-norm objective function may not handle the low level outliers very well, but it is very effective to reduce the influence high level outliers. Conversely, the algorithm given by [2] is very sensitive with outliers. For the robust target localization algorithm [12], even though it can also effectively reduce the influence of outliers, its performance is worse than our proposed methods.

In the following experiment, we randomly choose one transmitter and one receiver, and then add NLOS outliers into their relevant measurements. Thus 7 elements in the 4 x 4 measurement matrix include outliers. The results are shown in Fig. 7.

From Fig. 7 we see that the robust target localization algorithm given by [12] and our proposed algorithms can reduce the influence of outliers in this case. However, due to the high proportion of outliers, all mentioned algorithms cannot give satisfied results. But among them, the performance of $l_0$-norm LPNN LCA is still the best. Then, we further add one transmitter $t_5 = [−6000, −5000]^T m$ and one receiver $r_5 = [8000, 600]^T m$. After that, the geometry of transmitters and receivers is depicted in Fig. 8.

Then, we still randomly choose one transmitter and one receiver, add NLOS outliers into their relevant measurements. Thus 9 elements in the 5 x 5 measurement matrix are with outliers. The results are shown in Fig. 9.

From Fig. 9 we see the performance of our proposed algorithms is better than others. And, generally speaking, $l_0$-norm LPNN LCA is the best.
The performance of our proposed algorithms is also superior. The results are shown in Fig. 10. At each outlier level, we still repeat the experiment 100 times. The Laplace distribution ranges from $-\lambda$ to $\lambda$, where $\lambda$ is the standard deviation of the Laplace distribution in this experiment. The standard deviation of the exponential distribution (outlier level) is varied from $10^{-2}$ m to $2 \times 10^{5}$ m.

C. Experiment 3: Target Localization in Gaussian Noise with Some SINR Outliers

In the third experiment, we evaluate the performance of our proposed algorithms under low signal-to-interference-noise ratio (SINR) environment. This experiment is implemented based on the MIMO radar system given by Fig. 4. First, we set the variance of Gaussian noise to 100 and introduce outliers, which are generated to model the low SINR environment. Assume that 5 measurement values $d_{1,2}$, $d_{2,3}$, $d_{3,4}$, $d_{4,1}$, and $d_{1,4}$, are influenced by SINR outliers. The SINR outliers can be both negative and positive, they are generated by Laplace distribution in this experiment. The standard deviation of Laplace distribution ranges from $2 \times 10^{2}$ m to $2 \times 10^{5}$ m. At each outlier level, we still repeat the experiment 100 times. The results are shown in Fig. 10.

From Fig. 10 we see that under low SINR environment, the performance of our proposed algorithms is also superior to others. The performance of $l_{0}$-norm LPNN LCA is still the best.

VI. CONCLUSION

In this paper, two algorithms for solving target localization problem in a distributed MIMO radar system were developed. To alleviate the influence of outliers, we utilize the property of $l_{p}$-norm ($p = 1$, or $p = 0$) and redesign the objective function of the original optimization problem. To achieve a real-time solution, both two proposed algorithms are based on the concept of LPNN. Since the objective function of the proposed mathematic model is non-differentiable. Two approaches are devised for solving this problem. In first method, we use a differentiable function to approximate the $l_{1}$-norm in objective function. While, in the second method, we combine the LCA with LPNN framework, and propose an algorithm which not only solves the non-differentiable problem of $l_{1}$-norm but also $l_{0}$-norm. The experiments demonstrate that the proposed algorithms can effectively reduce the influence of outliers and they are superior to several state-of-the-art MIMO radar target localization methods.

REFERENCES

[1] H. Godrich, A. M. Haimovich, and R. S. Blum, “Target localization accuracy gain in MIMO radar-based systems,” IEEE Transactions on Information Theory, vol. 56, no. 6, pp. 2783–2803, 2010.
[2] J. Liang, C. S. Leung, and H. C. So, “Lagrange programming neural network approach for target localization in distributed MIMO radar,” IEEE Trans. Signal Process., vol. 64, no. 6, pp. 1574–1585, Mar. 2016.
[3] O. Bar-Shalom and A. J. Weiss, “Direct positioning of stationary targets using MIMO radar,” Signal Processing, vol. 91, no. 10, pp. 2345–2358, 2011.
[4] B. K. Chalise, Y. D. Zhang, M. G. Amin, and B. Himed, “Target localization in a multi-static passive radar system through convex optimization,” Signal Processing, vol. 102, pp. 207–215, 2014.
[5] Y. Huang, J. Benesty, G. W. Elko, and R. M. Mersereau, “Real-time passive source localization: A practical linear-correction least-squares approach,” IEEE transactions on Speech and Audio Processing, vol. 9, no. 8, pp. 943–956, 2001.
[6] D. Li and Y. H. Hu, “Least square solutions of energy based acoustic source localization problems,” in Parallel Processing Workshops, 2004. ICPP 2004 Workshops. Proceedings. 2004 International Conference on. IEEE, 2004, pp. 443–446.

[7] S. Zhang and A. Constantinides, “Lagrange programming neural networks,” IEEE Transactions on Circuits and Systems II: Analog and Digital Signal Processing, vol. 39, no. 7, pp. 441–452, 1992.

[8] M. Nagamatu and T. Yanaru, “On the stability of Lagrange programming neural networks for satisfiability problems of prepositional calculus,” Neurocomputing, vol. 13, no. 2, pp. 119–133, 1996.

[9] X. Zhu, S.-W. Zhang, and A. G. Constantinides, “Lagrange neural networks for linear programming,” J. Parallel Distrib. Comput., vol. 14, no. 3, pp. 354–360, Mar. 1992.

[10] V. Sharma, R. Jha, and R. Naresh, “An augmented Lagrange programming optimization neural network for short term hydroelectric generation scheduling,” Engineering Optimization, vol. 37, pp. 479–497, Jul. 2005.

[11] J. Liang, H. C. So, C. S. Leung, J. Li, and A. Farina, “Waveform design with unit modulus and spectral shape constraints via Lagrange programming neural network,” IEEE Journal of Selected Topics in Signal Processing, vol. 9, no. 8, pp. 1377–1386, 2015.

[12] J. Liang, D. Wang, L. Su, B. Chen, H. Chen, and H. So, “Robust mimo radar target localization via nonconvex optimization,” Signal Processing, vol. 122, pp. 33–38, 2016.

[13] C. J. Rozell, D. H. Johnson, R. G. Baraniuk, and B. A. Olshausen, “Sparse coding via thresholding and local competition in neural circuits,” Neural Computation, vol. 20, no. 10, pp. 2526–2563, 2008.

[14] A. Balavoine, C. J. Rozell, and J. Romberg, “Global convergence of the locally competitive algorithm,” in Proc. IEEE Digital Signal Processing Workshop and IEEE Signal Processing Education Workshop (DSP/SPE), 2011, pp. 431–436.

[15] A. Balavoine, J. Romberg, and C. J. Rozell, “Convergence and rate analysis of neural networks for sparse approximation,” IEEE Transactions on Neural Networks and Learning Systems, vol. 23, no. 9, pp. 1377–1389, 2012.

[16] C.-S. Leung, J. Sum, and A. G. Constantinides, “Recurrent networks for compressive sampling,” Neurocomputing, vol. 129, pp. 298–305, 2014.

[17] J.-F. Nouvel and M. Lesturgie, “Study of NLOS detection over urban area at Ka band through multipath exploitation,” in 2014 International Radar Conference. IEEE, 2014, pp. 1–5.

[18] P. Setlur, T. Negishi, N. Devroye, and D. Erricolo, “Multipath exploitation in non-LOS urban synthetic aperture radar,” IEEE Journal of Selected Topics in Signal Processing, vol. 8, no. 1, pp. 137–152, 2014.
Standard deviation of the Laplace noise (m)

RMSE(m)

- $l_2$-norm LPNN
- $l_1$-like LPNN
- $l_1$-norm LPNN LCA
- $l_0$-norm LPNN LCA
- MCC

Standard deviation of the Laplace noise (m)
Standard deviation of the Laplace noise (m)

RMSE(m)

\( l^2 \)-norm LPNN

\( l^1 \)-norm–like LPNN

\( l^1 \)-norm LPNN LCA

\( l^0 \)-norm LPNN LCA

MCC

Standard deviation of the Laplace noise (m)
Standard deviation of the Laplace noise (m)

RMSE (m)

- $l_2$-norm LPNN
- $l_1$-norm–like LPNN
- $l_1$-norm LPNN LCA
- $l_0$-norm LPNN LCA
- MCC

RMSE (m)

Standard deviation of the Laplace noise (m)