DISTRIBUTED LOCALIZATION IN WIRELESS SENSOR NETWORKS AS A PRE-IMAGE PROBLEM IN A REPRODUCING KERNEL HILBERT SPACE

Mehdi ESSOLOH, Cédric RICHARD, Hichem SNOUSSI, Paul HONEINE

Institute Charles Delaunay (FRE CNRS 2848)
University of Technology of Troyes
12 rue Marie Curie, BP 2060, 10010 Troyes cedex
Phone: 03.25.71.58.88 - Fax: 03.25.71.56.99 - mehdi.essoloh@utt.fr

ABSTRACT

In this paper, we introduce a distributed strategy for localization in a wireless sensor network composed of limited range sensors. The proposed distributed algorithm provides sensor position estimation from local similarity measurements. Incremental *Kernel Principal Component Analysis* techniques are used to build the nonlinear manifold linking anchor nodes. Non-anchor nodes positions are estimated by the pre-image of their nonlinear projection onto this manifold. This non-linear strategy provides a great accuracy when data of interest are highly corrupted by noise and when sensors are not able to estimate their Euclidean inter-distances.

1. INTRODUCTION

Recent technological advances in electronics and wireless communications have led to the development of tiny, low-power and low-cost sensors for physical observations purpose. Deployed randomly and densely in the environment of interest, and designed with efficient distributed algorithm, sensor networks seem to offer several opportunities, specially in monitoring and tracking applications [1]. The first step of estimating the sensor-location after deployment is thus a crucial issue. GPS system may solve in practice localization problem for each node of the embedded network. However, GPS receivers at each device may be too expensive and too power-intensive for the desired application, whose low energy consumption is the main constraint to respect. As a consequence, we consider only few sensors, called *anchor nodes*, which have a perfect a priori knowledge of their coordinates thanks to GPS receivers.

The most famous techniques used for localization applications are based upon either semidefinite programming (SDP) or multidimensional scaling (MDS) algorithms (see [2] and references therein). In the SDP framework, proximity measurements between sensors are expressed as geometric constraints, leading to a convex optimization problem. Unfortunately, this approach is unable to accommodate precise range data and unadapted to large scale sensor networks. The MDS algorithm [3], which is strongly related to the linear *Principal component Analysis* (PCA), has shown its effectiveness to estimate unknown sensors location by applying an orthogonal basis transformation. However, if the data are not inter-sensor distances or are linked to coordinates by an unknown non-linear function, linear techniques such as MDS and PCA fail to accurately estimate the node positions.

In this paper, we propose a distributed strategy for solving the localization problem, by borrowing state-of-art methods from machine learning. Each step of the proposed distributed scheme is built in the context of wireless sensor networks application, i.e keeping in mind energy reserve and computational limitations. We assume that each device determines non-Euclidean similarity measurements with other sensors, from some measurements such as the received signal strength indication (RSSI) or estimated covariance sensor data [4]. Thus, we propose to use these similarities $[\delta_{ij}]_{i,j=1}^{N}$ between neighbor nodes for location estimation purpose. The main idea is to design a nonlinear manifold via a high-dimensional *Reproducing Kernel Hilbert Space* $\mathcal{H}$ (RKHS) thanks to similarities between anchor-node measurements. Next, non-anchor sensors coordinates are estimated by the pre-image of their projection onto the manifold.

This paper is organized as follows: Section 2 is devoted to the main contribution of this work which consists of a kernel-based non-linear method for self-localization. In section 3, simulation results confirming the algorithm efficiency are shown.

Problem statement

Consider a network of N sensor nodes, of m anchors of known positions and $N-m$ unaware-position sensors, living in a p-dimensional space ($p=2$ for localization in plane, with $N-m \gg m > p$). Let $\boldsymbol{x}_i \in \mathbb{R}^p$ be the i-th sensor coordinates, $\{\boldsymbol{x}_i\}_{i=1}^{m}$ is the set of anchors nodes coordinates whose positions are known, $\{\boldsymbol{x}_i\}_{i=m+1}^{N}$ fit the unknown remaining sensor coordinates. If we assume that maximum spotting sensor range is equal to a distance r, sensor i will consider sensor j as a neighbor when the distance $\|\boldsymbol{x}_i - \boldsymbol{x}_j\|$ is lower than r. The i-th sensor neighborhood is denoted by $\mathcal{V}(i)$.

2. DISTRIBUTED LOCALIZATION ALGORITHM

The proposed algorithm is implemented in three steps. The first step is devoted to build the Hilbert space $\mathcal{H}$ associated to the reproducing positive definite kernel $\kappa(\boldsymbol{x}_i, \boldsymbol{x}_j)$ which best approximates estimated anchor pairwise similarities δ_{ij}. The second step is dedicated to map the data into a high-dimensional feature space, obtained from anchor informations by a Kernel-PCA technique. The third part is aimed to reconstruct the $N-m$ unknown sensors positions by a pre-image optimization

Inputs: $\{[\delta_{ij}]_{j\in\mathcal{V}(i)}\}_{i=1}^{m}, \{\mathbf{x}_i\}_{i=1}^{m}$
Initialization: $\sigma^* \leftarrow 0$
for $i = 1$ **to** m
- Compute σ_i^* maximum of $\Omega_i(\boldsymbol{x}_1, \ldots, \boldsymbol{x}_m, \boldsymbol{K}^*)$ at anchor number i
- $\sigma^* \leftarrow \sigma^* + \frac{1}{m}\sigma_i^*$
- Communicate σ^* to anchor number $i+1$

end for
Communicate σ^* to the N sensors

Table 1: Pseudo-code of distributed alignment maximization implemented on the m anchor-nodes, where each anchor knows only nearby anchor positions

scheme. The choice of *Kernel-PCA* [5] method is supported by its non-linearity property, its flexibility with a large variety of kernel functions and its distributed capabilities.

2.1 Kernel selection from anchor similarities

In order to build a valid RKHS, a kernel function respecting the anchor similarities should be selected thanks to the alignment method. The alignment criterion is a measure of similarity between two reproducing kernels or between a kernel and a target function [6]. Let $\boldsymbol{K}^*$ be a target matrix, and $\boldsymbol{K}_\sigma$ the Gram matrix associated to the reproducing kernel κ_σ, with a tuning parameter σ, for a given training set, i.e. with entries $\kappa_\sigma(\boldsymbol{x}_i, \boldsymbol{x}_j)$ for $i, j \in \{1, \ldots, m\}$. The alignment between these two matrices is defined by

$$\mathcal{A}(\boldsymbol{K}_\sigma, \boldsymbol{K}^*) = \frac{\langle \boldsymbol{K}_\sigma, \boldsymbol{K}^* \rangle_F}{\sqrt{\langle \boldsymbol{K}_\sigma, \boldsymbol{K}_\sigma \rangle_F \langle \boldsymbol{K}^*, \boldsymbol{K}^* \rangle_F}} \qquad (1)$$

where $\langle ., . \rangle_F$ is the Frobenius inner product between two matrices. For our purpose, the target matrix $\boldsymbol{K}^*$ is given by our similarity matrix, with

$$\boldsymbol{K}^*(i, j) = \delta_{ij}, \qquad (2)$$

for all $i, j \in \{1, \ldots m\}$, in which case $\langle \boldsymbol{K}^*, \boldsymbol{K}^* \rangle_F$ is constant. We have restricted potential kernels to Gaussian kernels defined as: $\kappa_\sigma(\boldsymbol{x}_i, \boldsymbol{x}_j) = \exp(-\|\boldsymbol{x}_i - \boldsymbol{x}_j\|^2/2\sigma^2)$ where σ is a positive scalar to be determined and $\boldsymbol{x}_i$, $\boldsymbol{x}_j$ are anchor coordinates. Therefore, the aim is to maximize the alignment criterion (1) with respect to σ.

Thanks to Lagrange method [7], the optimization problem is equivalent to:

$$\sigma^* = \arg\max_{\sigma} \sum_{i=1}^{m} \Big[\sum_{j=1}^{m} (\delta_{ij} \kappa_\sigma(\boldsymbol{x}_i, \boldsymbol{x}_j)) - \lambda \sum_{i,j=1}^{m} \kappa_\sigma(\boldsymbol{x}_i, \boldsymbol{x}_j)^2 \Big],$$

where the term between brackets will be denoted by $\Omega_i(\boldsymbol{x}_1, \ldots, \boldsymbol{x}_m, \boldsymbol{K}^*)$.

In practice, if the anchor j is a neighbor of anchor i, δ_{ij} is computed, otherwise it is set to zero. This formulation respects the hypothesis of spatial correlation [4] and helps to preserve energy by limiting communications to neighbors. Thus, we obtain

$$\Omega_i(\boldsymbol{x}_1, \ldots, \boldsymbol{x}_m, \boldsymbol{K}^*) = \sum_{j\in\mathcal{V}(i)} \delta_{ij} \kappa_\sigma(\boldsymbol{x}_i, \boldsymbol{x}_j) - \lambda \sum_{j\in\mathcal{V}(i)} \kappa_\sigma(\boldsymbol{x}_i, \boldsymbol{x}_j)^2$$

Pre-processing: learn σ^* as in Table 1
Inputs: $\mathbf{K} = [\kappa_{\sigma^*}(\mathbf{x}_i, \mathbf{x}_j)]_{i,j=1}^{m}$
Initialization: Randomly set $\mathbf{A}(0)$
for $k = 1$ **to** m
- Compute $\mathbf{A}(k)$ from (4) at anchor k
- Communicate $\mathbf{A}(k)$ to anchor $k+1$

end for
Communicate $\mathbf{A}^*$ to the non-anchor nodes

Table 2: Pseudo-code of incremental Kernel-PCA algorithm implemented on the m anchor-nodes.

where λ is the Lagrange coefficient and $\mathcal{V}(i)$ is the set of anchor-neighbors of anchor i. A distributed gradient descent algorithm is performed from anchor to anchor according to the scheme illustrated in Table 1.

After computing σ^* as the best alignment, the second step consists in building the nonlinear manifold related to a subspace in the RKHS $\mathcal{H}$.

2.2 Kernel-PCA upon anchor nodes

As a nonlinear extension of PCA, *Kernel-PCA* determines the principal axes in a RKHS, which can be constructed explicitly from the input space by a nonlinear map $\phi(.)$, or implicitly by considering the corresponding reproducing kernel $\kappa(\cdot, \cdot)$. In other words, for an m input data problem, $[\boldsymbol{x_k}]_{k=1}^{m}$, a PCA is performed in $\mathcal{H}$ yielding a set of $m-1$ orthogonal axes[1] $\{\boldsymbol{v}^1, \ldots, \boldsymbol{v}^{m-1}\}$, and thus defining a subspace $\mathcal{P}$ of $\mathcal{H}$. Since the latter lies in the span of the ϕ-images of the input data, *Kernel-PCA* [5] is computed by diagonalizing the dot-product matrix $\boldsymbol{K}$ in $\mathcal{H}$, by solving

$$m\lambda_k \boldsymbol{a}_k = \boldsymbol{K}\boldsymbol{a}_k \quad 1 \leq k \leq m-1 \qquad (3)$$

for the column vectors $\boldsymbol{a}_k$, with $k = 1, \ldots, m-1$, of feature coefficients, and $\boldsymbol{K} = [\langle \phi(\boldsymbol{x}_i), \phi(\boldsymbol{x}_j) \rangle]_{i,j=1}^{m}$ is the dot-product matrix in $\mathcal{H}$, known as the *Gram matrix* or learning matrix. This is done without the need to carry out the map ϕ explicitly. Only the kernel values $\kappa(\boldsymbol{x}_i, \boldsymbol{x}_j)$ are needed.

For the choice of the reproducing kernel, there is no restrictions with Kernel-PCA. We propose to use the one obtained from section 2.1, with the maximum alignment criterion. Therefore, we consider the Gram matrix $[\kappa_{\sigma^*}(\boldsymbol{x}_i, \boldsymbol{x}_j)]_{i,j=1}^{m}$. The mapping $\phi : \mathbb{R}^p \rightarrow \mathcal{H}$ corresponds to a RKHS $\mathcal{H}$ where dot products $\langle \phi(\boldsymbol{x}_i), \phi(\boldsymbol{x}_j) \rangle_{\mathcal{H}}$ are as close as possible to similarity measurements δ_{ij}.

We assume that each anchor-sensor has at least one anchor as a neighbor node which means that all anchor nodes respect a connected graph. As considered in section 2.1 and illustrated in Table 1, each anchor already knows the positions of its neighbor anchors, which for sensor i is given by $\{\boldsymbol{x}_j\}_{j\in\mathcal{V}(i)}$. Thus, it can compute $\kappa_{\sigma^*}(\boldsymbol{x}_i, \boldsymbol{x}_j)$ for $j \in \mathcal{V}(i)$ or nullify it otherwise.

[1] For the sake of clarity, we assume that all mapped training pattern are centered in the feature space. Therefore, we count $m-1$ eigenvectors associated to positive eigenvalues non-null. Otherwise one should substitute $\boldsymbol{K}$ in (3) with $\boldsymbol{K} - \frac{1}{m}\mathbf{1}_m\boldsymbol{K} - \frac{1}{m}\boldsymbol{K}\mathbf{1}_m + \frac{1}{m^2}\mathbf{1}_m\boldsymbol{K}\mathbf{1}_m$ where $(\mathbf{1}_m)_{ij} = 1$

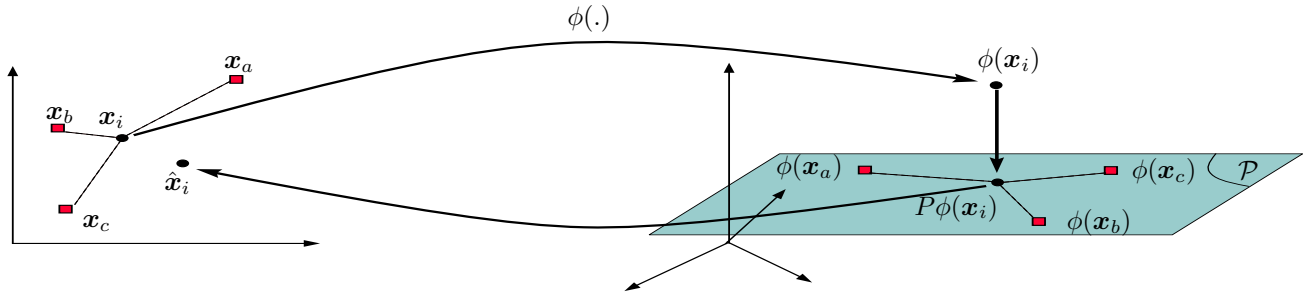

Figure 1: Illustration of the idea behind Kernel-PCA and pre-image techniques for localization in sensor networks. The left-hand-side frame corresponds to the input space (plane) and the right-hand-side to the feature space. The latter is reduced to the subspace $\mathcal{P}$ defined by the anchor nodes (red squares). The position of any non-anchor node (black circle) is estimated from (7) by a pre-image technique, after projecting its image onto $\mathcal{P}$.

After evaluating the anchor-node pairwise similarities, an incremental Kernel-PCA algorithm is run over the m anchors of the network by considering the kernel values $[\kappa_{\sigma^*}(\boldsymbol{x}_i, \boldsymbol{x}_j)]_{i,j=1}^m$. The *Kernel Hebbian* algorithm [8], presented as a direct application of the *Generalized Hebbian* algorithm in an RKHS, is dedicated to solving a kernel eigen-problem in a distributed way. The matrix of feature coefficients $\boldsymbol{A} = [\boldsymbol{a}_1\ \boldsymbol{a}_2\ \cdots\ \boldsymbol{a}_{m-1}] \in \mathbb{R}^{m\times m-1}$ is updated by each anchor node as follows:

$$\boldsymbol{A}(k+1) = \boldsymbol{A}(k) + \eta\Big(\boldsymbol{y}(k)\boldsymbol{b}(k)^T - LT\big(\boldsymbol{y}(k)\boldsymbol{y}(k)^T\big)\boldsymbol{A}(k)\Big), \tag{4}$$

where η is a predefined learning rate, $\boldsymbol{y}(k) = \sum_{j=1}^m \boldsymbol{a}^j(k)\kappa_{\sigma^*}(\boldsymbol{x}_k, \boldsymbol{x}_j)$, $\boldsymbol{b}(k) = [0\cdots\ 1\ \cdots 0]$ a canonical unit vector whose $k-th$ element is 1, and $LT(\cdot)$ the lower triangular operator.

At the initialization stage, $\boldsymbol{A}(0)$ is assigned a random value[2] by the anchor number 1. Then it is communicated between anchors and updated with respect to the recursive expression (4). After this learning stage, $\boldsymbol{A}^*$ is communicated to the remaining $N-m$ sensors of the network. The pseudo-code is described in Table 2. Next, for each sensor i, we represent its image $\phi(\boldsymbol{x}_i)$ in the space defined by the anchors, and obtained from the feature vectors defined by $\boldsymbol{A}^*$. The problem of estimating $\hat{\boldsymbol{x}}_i$ from that representation is the classical pre-image reconstruction problem.

2.3 Pre-image for location estimation

As we consider a reproducing kernel associated to a nonlinear map $\phi(\cdot)$, the induced RKHS is of higher dimensionality, and infinite for some kernels such as the Gaussian kernel. Thus, the image $\phi(\boldsymbol{x}_i)$ of any sensor i has redundant information. We consider representing it with the finite-number features in $\boldsymbol{A}^*$ obtained with anchors, as we hope that the remaining dimensions contain only noise information. Representing the data in an eigen-space is known in machine learning literature as the *empirical kernel map* [9]. Next we study the problem of getting back to the input space, with a pre-image technique, in order to estimate the $N-m$ coordinates of non-anchor sensors. But before, we recall that $\mathcal{P}$ is spanned by the $m-1$ eigenvectors, or less if some eigenvalues are null, and define P as the projection operator onto it. Thus, $P\phi(\boldsymbol{x})$ is the projection of $\phi(\boldsymbol{x})$ onto $\mathcal{P}$, as illustrated in Fig.1.

The projection satisfies the minimal reconstruction error $\|P\phi(\boldsymbol{x}) - \phi(\boldsymbol{x})\|^2$. While we have no access to $\boldsymbol{x}$, we want to determine $\hat{\boldsymbol{x}} \in \mathbb{R}^p$ whose image $\phi(\hat{\boldsymbol{x}})$ best approximates $P\phi(\boldsymbol{x})$, in the least square sense. Since $P\phi(\boldsymbol{x})$ is already known given only the anchor/non-anchor similarities, the optimized problem could then be put in the following form [10]:

$$\hat{\boldsymbol{x}} = \arg\min_{\boldsymbol{z}} \|P\phi(\boldsymbol{x}) - \phi(\boldsymbol{z})\|^2 \tag{5}$$

This optimization problem should be solved at each of the $N-m$ non-anchor sensors, in order to estimate their unknown coordinates. Each non-anchor sensor j has to evaluate m kernel values $\kappa_{\sigma^*}(\boldsymbol{x}_i, \boldsymbol{x}_j)$ for $i \in \{1, \ldots, m\}$ (i.e. with the m anchors). However, this computational cost is reduced in practice, as a small number of anchors are in the neighborhood of each sensor, i.e if anchor i is inside the visibility range of sensor j, we have $\kappa_{\sigma^*}(\boldsymbol{x}_i, \boldsymbol{x}_j) = \delta_{ij}$, otherwise $\kappa_{\sigma^*}(\boldsymbol{x}_i, \boldsymbol{x}_j) = 0$.

We remind that non-anchor sensors have already received the last update of expansion matrix $\boldsymbol{A}^*$. Each sensor can extract its nonlinear principal components. Assuming that β_j^k is the projection onto the k-th component for the non-anchor sensor j, it reads:

$$\beta_j^k = \sum_{i=1}^m a_k^i \kappa_{\sigma^*}(\boldsymbol{x}_i, \boldsymbol{x}_j) \tag{6}$$

Therefore, we can write $P\phi(\boldsymbol{x}_j) = \sum_{k=1}^{m-1} \beta_j^k \boldsymbol{v}^k$ where $\boldsymbol{x}_j$ is the sensor j unknown coordinates, β_j^k is the pro-

Pre-processing: `learn` $\mathbf{A}^*$ `as in Table 2`
Inputs: $\mathbf{A}^*, \sigma^*, \{[\delta_{ij}]_{l\in\mathcal{V}(j)}\}_{j=m+1}^N, \{\mathbf{x}_i\}_{i=1}^m$
Initialization: `randomly set` $\mathbf{x}_{m+1}, \ldots, \mathbf{x}_N$
for $j = m+1$ **to** N
- `At node number` j`, compute` $\hat{\mathbf{x}}_j$ `from (7)`

end for

Table 3: Pseudo-code of pre-image algorithm for localization estimation.

[2] $\boldsymbol{A}(0)$ must neither be the zero vector nor orthogonal to the eigenvectors.

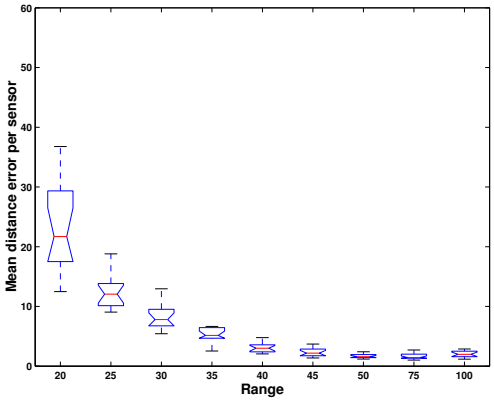

Figure 2: Evolution of mean distance error per sensor with a gaussian generative covariance function $\tau = 20$ (**Exp.I**).

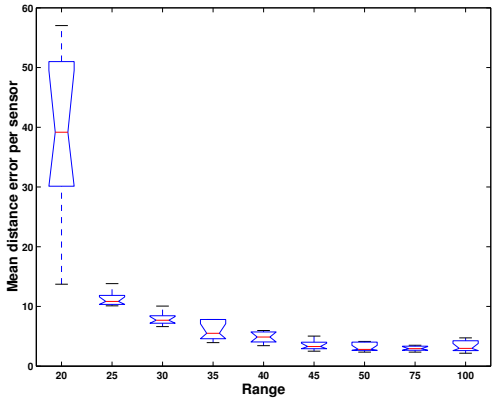

Figure 3: Evolution of mean distance error per sensor with a third degree polynomial generative covariance function $d = 60$ (**Exp.II**).

jection on the k^{th} component and $\boldsymbol{v}^k$ is the k-th eigenvector. Solution from (5) can be simplified in terms of dot products in $\mathcal{H}$:

$$\hat{\boldsymbol{x}}_j \quad = \quad \arg\min_{\boldsymbol{z}} \kappa_{\sigma^*}(\boldsymbol{z},\boldsymbol{z}) - 2P\phi(\boldsymbol{x}_j)^T.\phi(\boldsymbol{z})$$

where we neglected a term independent of $\boldsymbol{z}$. If the considered kernel is radial such as the Gaussian kernel, thus $\kappa(\boldsymbol{z},\boldsymbol{z})$ is constant for all $\boldsymbol{z}$, we get the following optimization problem (see [10] for more details)

$$\begin{aligned}\hat{\boldsymbol{x}}_j &= \arg\max_{\boldsymbol{z}} P\phi(\boldsymbol{x}_j)^T\phi(\boldsymbol{z}) \\ &= \arg\max_{\boldsymbol{z}} \sum_{k=1}^{m-1}\beta_j^k \sum_{i=1}^{m} a_k^i \kappa_{\sigma^*}(\boldsymbol{z},\boldsymbol{x}_i), \qquad (7)\end{aligned}$$

Thus, a gradient descent algorithm is performed by non-anchor sensor j in order to minimize the criterion (7). Initially, all non-anchors nodes have a random position in the network and the optimization is executed locally with no synchronization constraints. The pre-image technique for localization in sensor networks is illustrated in Table 3.

3. EXPERIMENTS

The algorithm proposed in this paper is independent of the type of similarities considered for localization purpose. In our simulation scenario, we make the assumption that data inputs are jointly distributed with a covariance being function of the distance. Thus, estimated covariance relations are used as local similarity measurements between neighbor nodes [4]. For this, let a network consists of sensors measuring the same physical phenomena, such as temperature, pressure or luminance measurements. Here, $\boldsymbol{w}_i$ represents the vector data recorded by sensor i on a given time interval. By considering a static field, we assume that data readings $\{\boldsymbol{w}_i\}_{i=1}^N$ are jointly generated from a normal distribution of mean $\mu = [\mu_1 \cdots \mu_N]$ and covariance matrix $\boldsymbol{C}$ shaped as

$$\mathbf{C} = [\psi(\|\boldsymbol{x}_i - \boldsymbol{x}_j\|)]_{i,j=1}^N \qquad (8)$$

where $\psi : [0,\infty[\rightarrow [0,1]$ is a non-negative decreasing function. The experimental setting consists of 20 anchor-sensors and 80 non-anchor sensors randomly spread over a square surface 100×100. The data readings for each sensor consists of 200 measurements. We consider the Gaussian kernel, while its bandwidth σ^* is given by maximizing the alignment.

In the first experience (noted **Exp.I**), data are generated according to the unknown covariance $\psi_1(\mathbf{z}) = \exp(-\frac{\mathbf{z}^2}{2\tau^2})$ where $\tau = 20$. Fifty simulations were run for each range value. Localization results are presented in Fig. 2 for different sensor range value, the Lagrange coefficient for alignment λ is fixed to 0.3. The second experiment (noted **Exp.II**) is dedicated to test the robustness of our algorithm by using a generative covariance ψ different from the Gaussian kernel. The spherical model [11], which is commonly applied in environmental and geological sciences, is used with

$$\psi_2(\mathbf{z}) = \begin{cases} 1 - \frac{3}{2d}\mathbf{z} + \frac{1}{2d^3}\mathbf{z}^3 & \text{for} \quad 0 \leq \mathbf{z} \leq d \\ 0 & \text{for} \quad d < \mathbf{z} \end{cases}$$

where d is a cut-off distance. In **Exp.II**, the parameter d is fixed to 60 in order to obtain a decreasing speed

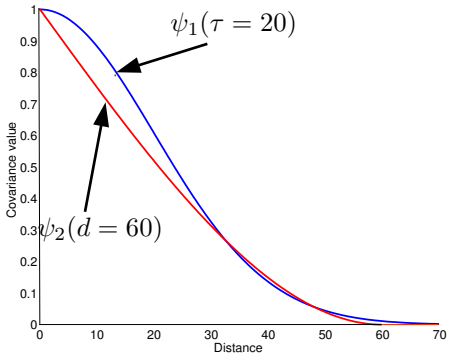

Figure 4: Graph comparing curves of functions ψ_1 with parameter $\tau = 20$ and ψ_2 with $d = 60$.

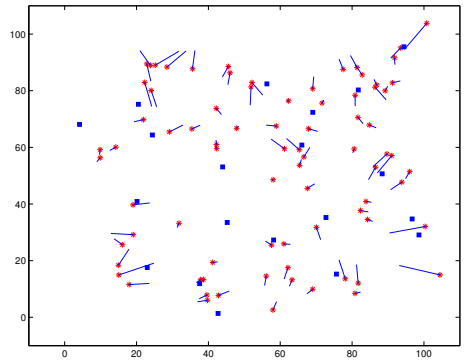

Figure 5: The 20 anchors are represented by blue squares, estimated location of the 80 nodes by a red cross, distance error in location to true position by segment.

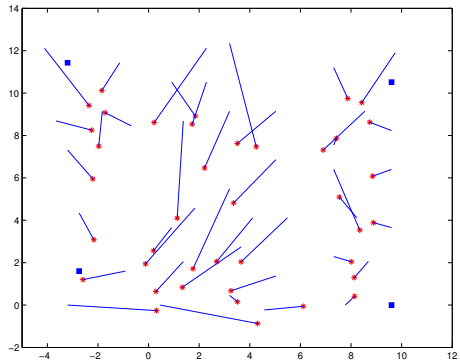

Figure 6: Localization results with our algorithm from the indoor campaign data [12].

of covariance with distance close to that of **Exp.I** with $\tau = 20$ (see Fig. 4). An example of localization results is shown in Fig. 5 with $r = 45$. Note that some similarity measurements are missed when the sensor range r is fixed to 45 and $d = 60$ (see Fig. 5), since remote sensor pairwise are ignored in each algorithm step. Results of fifty simulations with $\lambda = 0.3$, are shown in Fig. 3 for different sensor range values. Results of **Exp.II** show that our algorithm is still robust when no similarity exists between the chosen Gaussian kernel and the underlying generative covariance.

We also confront our algorithm to a well-known state-of-the-art algorithm (the MLE algorithm) with real RSS readings. In [12], the wireless sensor network configuration consists of 44 sensors placed in a 14-by-13 meters office area, whose 4 anchors near the corner. Localization is executed in a centralized framework and no neighborhood constraint is considered (i.e each sensor detects all others remaining). The MLE algorithm [12] yields a mean location error over the 40 non-anchors equal to 2.30 meters. For performance comparison, we use the same RSS pairwise measurements related to this indoor campaign and publicly available at http://www.eecs.umich.edu/∼hero/localize/. Input similarities δ_{ij} are computed as follows:

$$\delta_{ij} = \exp\left(- \frac{\|\mathbf{s}_{x_i} - \mathbf{s}_{x_j}\|^2}{2\sigma^2} \right) \qquad (9)$$

where $\mathbf{s}_{x_i}$ is the 44-dimensional RSS vector read at sensor i. Our algorithm is tested and yields a mean location error over the 40 non-anchors equal to 2.13 meters ($\lambda = 1.1$, $\sigma = 10$). The localization results are depicted in Fig. 6.

4. CONCLUSION AND PERSPECTIVES

In this article, we have shown that Kernel-PCA, coupled with reconstruction techniques, computes unknown coordinates of the network sensors thanks to a non-linear transformation ϕ of the input space. This localization problem is solved on simulated and real data. In both centralized and distributed framework, we have shown how pre-image techniques can achieve accurate positioning results.

REFERENCES

[1] D. Estrin, L. Girod, G. Pottie, and M. Srivastava, "Instrumenting the world with wireless sensor networks," *IEEE Int. Conf. on Acoustics, Speech and Signal Processing (ICASSP)*, vol. 4, pp. 2033–2036, 2001.

[2] J. Bachrach and C. Taylor, "Localization in sensor networks," in *Handbook of Sensor Networks* (I. Stojmenovic, ed.), 2005.

[3] T. Cox and M. Cox, *Multidimensional Scaling, Second Edition*. Chapman & Hall/CRC, 2000.

[4] N. Patwari and A. Hero, "Manifold learning algorithms for localization in wireless sensor networks," *IEEE Int. Conf. on Acoustics, Speech, and Signal Processing (ICASSP)*, vol. 3, pp. 857–860, 2004.

[5] B. Schölkopf, A. Smola, and K.-R. Müller, "Nonlinear component analysis as a kernel eigenvalue problem," *Neural Computation*, vol. 10, no. 5, pp. 1299–1319, 1998.

[6] N. Cristianini, A. Elisseeff, J. Shawe-Taylor, and J. Kandola, "On kernel target alignment," *Proceedings of the Neural Information Processing Systems*, pp. 367–373, 2002.

[7] D. Luenberger, "Linear and nonlinear programming," *Mass.: Addison-Wesley*, 1984.

[8] K. Kim, M. Franz, and B. Schölkopf, "Iterative kernel principal component analysis for image modeling," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 27, no. 9, pp. 1351–1366, 2005.

[9] B. Schölkopf and A. J. Smola, *Learning with Kernels*. Cambridge, MA, USA: MIT Press, 2002.

[10] S. Mika, B. Schölkopf, A. Smola, K.-R. Müller, M. Scholz, and G. Rätsch, "Kernel PCA and de-noising in feature spaces," *Proceedings of the Neural Information Processing Systems*, pp. 536–542, 1999.

[11] T. Gneiting, "Compactly supported correlation functions," Technical report NRCSE-TRS No. 045, Environmental Protection Agency, May 2000.

[12] N. Patwari, A. O. Hero, M. Perkins, N. Correal, and R. O'Dea, "Relative location estimation in wireless sensor networks," *IEEE Trans. on Sig. Proc.*, vol. 51, no. 8, pp. 2137–2148, 2003.