LOCALIZATION IN SENSOR NETWORKS – A MATRIX REGRESSION APPROACH

Paul Honeine, Cédric Richard, Mehdi Essoloh, Hichem Snoussi

Institut Charles Delaunay (FRE CNRS 2848)- LM2S - Université de technologie de Troyes
12 rue Marie Curie, BP 2060, 10010 Troyes cedex, France - fax. +33.3.25.71.56.99

ABSTRACT

In this paper, we propose a new approach to sensor localization problems, based on recent developments in machine leaning. The main idea behind it is to consider a matrix regression method between the ranging matrix and the matrix of inner products between positions of sensors, in order to complete the latter. Once we have learnt this regression from information between sensors of known positions (beacons), we apply it to sensors of unknown positions. Retrieving the estimated positions of the latter can be done by solving a linear system. We propose a distributed algorithm, where each sensor positions itself with information available from its nearby beacons. The proposed method is validated by experimentations.

1. INTRODUCTION

In ad hoc wireless sensor networks, a large number of applications require location awareness of the sensors, including tracking, environmental monitoring and many military applications. Without the knowledge of its position, the information captured by a sensor becomes obsolete. The main building block of these networks is a low-cost sensor, with low power resources, leaving no room to (absolute) self-positioning device. To overcome this drawback, one includes in the network a small number of sensors with known positions (and sometimes high power and communication capabilities). These sensors, often known by anchors or beacons and designated hereafter by the latter, communicate to other sensors information allowing the latter to estimate their positions. For this, each sensor determines ranging (distance) measurements with other sensor, from some measurements such as the received signal strength indication (RSSI), the connectivity, the hop count, the time difference of arrival, ... Most work on localization in sensor networks considers either multidimensional scaling (MDS) techniques or semidefinite programming (see [1, 2], and references therein), in order to determine a function that links the ranging of the sensors to their positions, based on the known positions of some beacons.

{FirstName.SecondName}@utt.fr

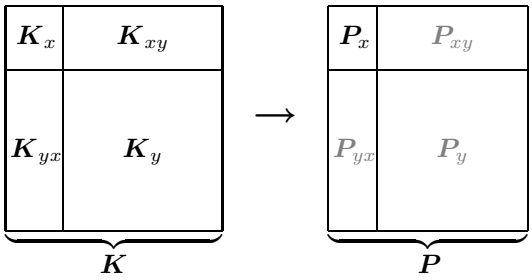
Introduced by Aronszajn in the mid 50s, the usefulness of reproducing kernels has been demonstrated in the last 15 years in the field of pattern recognition with the statistical learning theory and the so called kernel machines, such as support vector machines (SVM) and kernel principal component analysis (kernel-PCA) [3]. Reproducing kernels provide new insights in sensor networks. This has been known for a while, as many researchers in sensor networks focus on detection, tracking and classification using kernel machines. In recent years, there has been an increasing interest in this framework for localizing sensors, with kernel-PCA [4, 5], SVM [6], and manifold regularization [7].

In this paper, we derive a two-stage strategy. First we seek a mapping function between the ranging and the inner products between positions, of a given sensor with beacons. Learned with data available from beacons, it is then applied to any sensor, leading to an estimate on the inner products between its (unknown) position and the (known) positions of the beacons. In the second stage, we determine the position of the sensor from these estimated inner products. It turns out that the first stage is a matrix completion problem, where the inner-product matrix is completed from the (entirely available) ranging matrix, and thus can be solved with the recently introduced matrix regression method [8]. By learning the regression from available data in both matrices simultaneously, thus from inter-beacon information, we show that this reduces to a linear optimization problem. The second stage can be solved by considering the Nyström method, a technique for approximating kernel matrices in the machine leaning community (see [9] for a link to different MDS techniques). We investigate a distributed version of the method, by solving locally the optimization problem. We emphasize that the proposed method is independent of the ranging type, thus can be applied to RSSI, hop count, or any other ranging information.

This paper is organized as follows. We begin by presenting the matrix regression method of sensor localization, then in section 3 we derive a technique for determining the position of the sensors. We propose in section 4 a distributed algorithm, taking into account only nearby beacons for the considered sensor. Finally, computer simulations are carried out to validate the proposed approach.

2. THE MATRIX REGRESSION METHOD

Consider a network of m sensors of unknown positions and n beacons of known positions, living in a d dimension (2D or 3D). Let $\boldsymbol{X}$ and $\boldsymbol{Y}$ be the coordinate matrices of beacons and sensors, respectively, of size n-by-d and m-by-d, and $[\boldsymbol{X}^\top\ \boldsymbol{Y}^\top]^\top$ the overall coordinate matrix. The inner product between their coordinates is given by $\boldsymbol{P} = [\boldsymbol{X}\ \boldsymbol{Y}]\,[\boldsymbol{X}^\top\ \boldsymbol{Y}^\top]$, which can be decomposed into four block submatrices, $\boldsymbol{P}_x = \boldsymbol{X}\boldsymbol{X}^\top$, $\boldsymbol{P}_{yx} = \boldsymbol{Y}\boldsymbol{X}^\top$, $\boldsymbol{P}_{xy} = \boldsymbol{P}_{yx}^\top$, and $\boldsymbol{P}_y = \boldsymbol{Y}\boldsymbol{Y}^\top$, as illustrated in (1) with unknown submatrices set to gray-color. On the other hand, we have the overall ranging matrix, denoted by $\boldsymbol{K}$ with entries $\kappa(\boldsymbol{x}_i, \boldsymbol{x}_j)$, similarly decomposed into $\boldsymbol{K}_x$, $\boldsymbol{K}_{yx}$, $\boldsymbol{K}_{xy}$, and $\boldsymbol{K}_y$, as given in (1).



(1)

In a conventional regression problem, one seeks a function $\phi(\cdot)$ that links an input variable $\boldsymbol{x}$ into a response (output) variable z, under the constraints $\phi(\boldsymbol{x}_i) = z_i$ for all available training data $\{(\boldsymbol{x}_1, z_1), \ldots, (\boldsymbol{x}_n, z_n)\}$. While there exists an infinite number of functions verifying such constraints, one considers functions with some regularizing properties (such as smoothness). This can be done by restricting the hypothesis space to the RKHS of a given reproducing kernel, say $\kappa(\cdot, \cdot)$. Moreover, from the Representer Theorem [10], the optimal function has the form

$$\phi(\cdot) = \sum_{k=1}^{n} \alpha_k \kappa(\boldsymbol{x}_k, \cdot). \tag{2}$$

For instance, this is true for kernel-PCA, where each principal axe $\phi(\cdot)$ (principal function to be more precise, since it belongs to the RKHS) is determined by its n coefficients α_k, obtained by an eigen-decomposition of the n-by-n matrix of entities $\kappa(\boldsymbol{x}_i, \boldsymbol{x}_j)$, thus $\boldsymbol{K}_x$. Since $\phi(\boldsymbol{x})$ corresponds to the principal coordinate of $\boldsymbol{x}$, the latter can be represented into a low-dimensional space by considering only a couple of principal coordinates. Since this is the essence of both MDS and kernel-PCA techniques, localization in sensor networks using kernel-PCA is proposed in [4], or more recently [5] (see [11] for a connection to MDS).

In what follows, we consider the general case of determining a set of optimal functions, fully described by their coefficients, which identifies the mapping between the two matrices described in (1). This is known as a matrix regression problem [8], between the input data $\kappa(\boldsymbol{x}_i, \boldsymbol{x}_j)$ of matrix $\boldsymbol{K}$ and the output $\boldsymbol{x}_i\boldsymbol{x}_j^\top$ of $\boldsymbol{P}$, and learnt from the available input-output couples, i.e. $\boldsymbol{K}_x$ and $\boldsymbol{P}_x$. For this, we consider a model of the form

$$\Psi(\boldsymbol{x}_i, \boldsymbol{x}_j) = \boldsymbol{x}_i\boldsymbol{x}_j^\top, \tag{3}$$

and determine the it from inter-beacon information. As above, we consider a particular form of Ψ, by representing each $\boldsymbol{x}$ in some coordinates obtained with a set of functions, $[\phi_1(\boldsymbol{x})\ \ \phi_2(\boldsymbol{x})\ \ \cdots]^\top$, leading to $\Psi(\boldsymbol{x}_i, \boldsymbol{x}_j) = [\phi_1(\boldsymbol{x}_i)\ \phi_2(\boldsymbol{x}_i)\ \cdots][\phi_1(\boldsymbol{x}_j)\ \phi_2(\boldsymbol{x}_j)\ \cdots]^\top = \sum_h \phi_h(\boldsymbol{x}_i)\phi_h(\boldsymbol{x}_j)$. In analogy to kernel-PCA where theses are principal coordinates obtained from principal axes constructed from available data $\kappa(\boldsymbol{x}_k, \boldsymbol{x}_\ell)$, we consider the same form as (2) for all ϕ_h's, constructed from $\boldsymbol{K}_x$. Therefore one should determine for each ϕ_h the optimal coefficient vector $\boldsymbol{\alpha} = [\alpha_1\ \alpha_2\ \cdots\ \alpha_n]^\top$, with

$$\begin{aligned} \phi_h(\boldsymbol{x}_i)\,\phi_h(\boldsymbol{x}_j) &= \Big(\sum_{k=1}^{n} \alpha_k \kappa(\boldsymbol{x}_k, \boldsymbol{x}_i)\Big)\Big(\sum_{\ell=1}^{n} \alpha_\ell \kappa(\boldsymbol{x}_\ell, \boldsymbol{x}_j)\Big) \\ &= \boldsymbol{\kappa}_i^\top \boldsymbol{\alpha}\,\boldsymbol{\alpha}^\top \boldsymbol{\kappa}_j \end{aligned}$$

where $\boldsymbol{\kappa}_i$ is the i-th column vector of $\boldsymbol{K}_x$. From the sum of these terms and (3), we get $\boldsymbol{x}_i\boldsymbol{x}_j^\top = \Psi(\boldsymbol{x}_i, \boldsymbol{x}_j) = \boldsymbol{\kappa}_i^\top \boldsymbol{A}\,\boldsymbol{\kappa}_j$, where $\boldsymbol{A}$ is a coefficient matrix. Since this should be satisfied for all beacons, i.e. $i, j = 1, \ldots, n$, we can write the optimization problem in matrix form, with

$$\min_{\boldsymbol{A}} \|\boldsymbol{P}_x - \boldsymbol{K}_x^\top \boldsymbol{A}\,\boldsymbol{K}_x\|_F^2, \tag{4}$$

where $\|\cdot\|_F$ is Frobenius norm. Once we obtain the optimal coefficient matrix $\boldsymbol{A}$, we can apply the resulting map to sensors with unknown positions, with $\boldsymbol{x}_i\boldsymbol{y}_j^\top = \Psi(\boldsymbol{x}_i, \boldsymbol{y}_j)$. From expressions above, we obtain the matrix expression

$$\boldsymbol{P}_{xy} = \boldsymbol{K}_x^\top \boldsymbol{A}\,\boldsymbol{K}_{xy}. \tag{5}$$

The optimization problem (4)-(5) can be solved by writing (4) as a generalized eigen-decomposition problem and injecting the resulting matrix in (5). As we notice that both expressions contain the matrix $\boldsymbol{T} = \boldsymbol{K}_x^\top \boldsymbol{A}$, we propose to solve the following equivalent optimization problem :

$$\min_{\boldsymbol{T}} \|\boldsymbol{P}_x - \boldsymbol{T}\boldsymbol{K}_x\|_F^2, \quad \text{and} \quad \boldsymbol{P}_{xy} = \boldsymbol{T}\boldsymbol{K}_{xy}. \tag{6}$$

This is a linear problem yields $\boldsymbol{T} = \boldsymbol{P}_x\boldsymbol{K}_x^{-1}$, and thus

$$\boldsymbol{P}_{xy} = \boldsymbol{P}_x\,\boldsymbol{K}_x^{-1}\boldsymbol{K}_{xy}. \tag{7}$$

3. SENSOR POSITION ESTIMATION

After estimating the matrix $\boldsymbol{P}_{xy}$ of inner products of positions between beacons and sensors, one has to find the coordinates of the latter. For this purpose, we take advantage

of the Nyström method, initially developed in the machine learning community to approximate a matrix by another matrix of lower rank [12]. In our case, on the one hand we recall that by construction we have $\boldsymbol{P}_x = \boldsymbol{X}\boldsymbol{X}^\top$, thus a d-rank matrix. From its eigen-decomposition, we have

$$\boldsymbol{P}_x = \boldsymbol{U}_d \, \boldsymbol{\Lambda}_d \, \boldsymbol{U}_d^\top ,$$

where $\boldsymbol{\Lambda}_d$ is a diagonal matrix of the d nonzero eigenvalues of $\boldsymbol{P}_x$, and $\boldsymbol{U}_d$ the matrix whose columns are the corresponding eigenvectors. By identification, we get

$$\boldsymbol{X} = \boldsymbol{U}_d \, (\boldsymbol{\Lambda}_d)^{1/2} \tag{8}$$

On the other hand, we can write $\boldsymbol{P}_{xy} = \boldsymbol{X}\boldsymbol{Y}^\top$ where $\boldsymbol{Y}$ is the coordinate matrix to be identified. By injecting (8) into this definition, we get the coordinates of the m sensors from

$$\boldsymbol{Y}^\top = (\boldsymbol{\Lambda}_d)^{-1/2} \, \boldsymbol{U}_d^\top \, \boldsymbol{P}_{xy}. \tag{9}$$

Since the resulting coordinates are determined in the space defined by the eigenvectors, one must conduct a final step of mapping them, with a linear (or affine to be more precise) transformation, into the initial space of beacons. Such step is commun to MDS techniques.

4. DISTRIBUTED ALGORITHM

In section 2, we considered completing the inner-product matrix by inverting the ranging matrix of the beacons. This stipulates that the beacons communicate to each other, in a peer-to-peer fashion or a multi-hop strategy. However, in practice, beacons may not be in the range of each other. Moreover, the matrices may become too cumbersome to invert and manipulate for a large scale sensor network. To overcome this, we propose a distributed algorithm, where each sensor gets information from nearby beacons in order to find its own position. In other words, any sensor i defines a set of neighboring[1] beacons with a submatrix of $\boldsymbol{K}_x$ and its counterpart in $\boldsymbol{P}_x$, denoted respectively $\boldsymbol{K}_i$ and $\boldsymbol{P}_i$, and $\boldsymbol{X}_i$ the corresponding coordinates. While the global optimization problem (6) leads to expression (7), by considering the distributed approach, we get

$$\boldsymbol{p}_i = \boldsymbol{P}_i \, \boldsymbol{K}_i^{-1} \boldsymbol{\kappa}_i, \tag{10}$$

where $\boldsymbol{p}_i$ is the inner-product column vector of positions between sensor i and its nearby beacons, and $\boldsymbol{\kappa}_i$ the column vector of ranging between them.

The corresponding coordinates of this sensor can be revealed by writing *locally* the expression (9), obtained from an eigen-decomposition problem. Then, a mapping transformation must be carried out as presented above, by considering this time only neighboring beacons. While this becomes fairly cumbersome for each sensor, we propose an alternative approach to find the coordinates, based on the pseudoinverse operator. For this purpose, we rewrite the problem as the following optimization problem

$$\min_{\boldsymbol{y}} \|\boldsymbol{p}_i - \boldsymbol{X}_i \, \boldsymbol{y}^\top\|_F^2.$$

It is well known that the solution of this linear system is given by the left pseudoinverse of the matrix $\boldsymbol{X}_i$, with

$$\boldsymbol{y} = \left(\boldsymbol{X}_i^\top \boldsymbol{X}_i\right)^{-1} \boldsymbol{X}_i^\top \boldsymbol{p}_i. \tag{11}$$

We emphasize on the fact that we don't need to apply a mapping to reposition the sensors with respect to beacons. The simplicity of the algorithm is illustrated in Table 1.

for each sensor i

Find the nearby beacons	`[dump,ind]=sort(K(i,1:n))`
Consider the closes nc beacons	`ind=ind(end:-1:end-nc+1)`
Get ranging between these beacons	`Ki=K(ind,ind)`
Get inner products between them	`Pi=Xn(ind,:)*Xn(ind,:)'`
Consider ranging with them	`ki=K(i,ind)`
Compute inner products with (10)	`pi=Pi*inv(Ki)*ki`
Determine position with (11)	`y=pi/(Xn(ind,:)')`

Table 1. Pseudocode of the distributed algorithm.

5. SIMULATIONS

To illustrate our method, we consider a configuration similar to the one proposed in [13], with ranging between two sensors is only a function of the distance between them, with

$$\kappa(\boldsymbol{x}_i, \boldsymbol{x}_j) = \exp\left(-\|\boldsymbol{x}_i - \boldsymbol{x}_j\|^2 / 2\sigma^2\right),$$

where σ is parameter corresponding to the range of the sensors. Next, sensors are randomly spread on a 1-by-1 square.

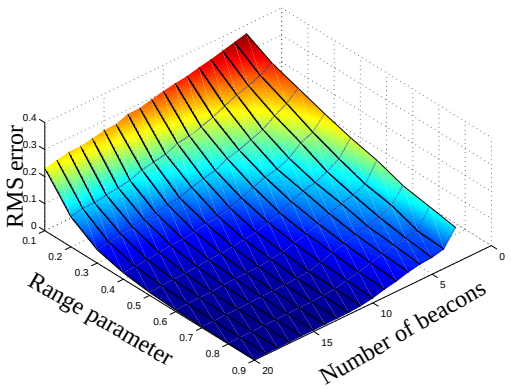


Fig. 1. Root-mean-square error on positions with the centralized algorithm, as a function of n and σ

[1] Different strategies can be proposed to define the neighborhood of a sensor. This can be done by examining the ranging values, where high values correspond to neighbors. We fix their number in simulations.

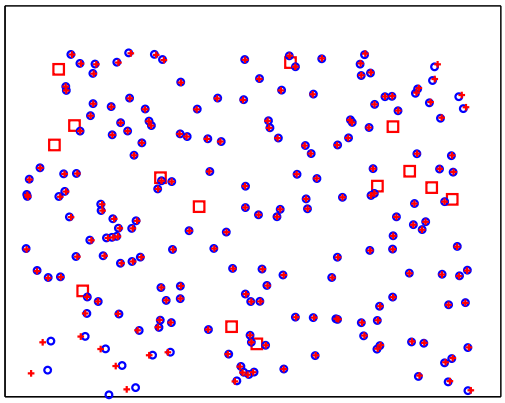


Fig. 2. Topology constructed by the centralized algorithm. The beacons are represented by □, sensors positions by +, and their estimations by ○.

In a first series of experiments, we apply the centralized algorithm to a network of 200 sensors, and study the influence of both the number of beacons and the range parameter. For this, we take $n = 3, 4, \ldots, 20$ and $\sigma = 0.1, 0.2, \ldots, 0.9$. In Fig. 1, we plot the resulting root-mean-square error, averaged over 100 trials. As expected, the localization error decreases as the number of beacons increases, and the visibility between sensors is high. By taking for instance 15 beacons (almost 7% of the sensors), and $\sigma = 0.75$, we get the topology illustrated in Fig. 2.

In a second experimentation, we consider a large scale network of 1000 sensors of low range, with $\sigma = 0.3$. To these, we include 50 beacons with the same characteristics. This setting results in inverting and manipulating large sparse matrices, since there is low visibility between these entities. For these reasons, we consider the distributed algorithm, each sensor determines its coordinates with information from the 5 most nearby beacons. Fig. 3 illustrates the resulting topology, with a root-mean-square error of 0.018.

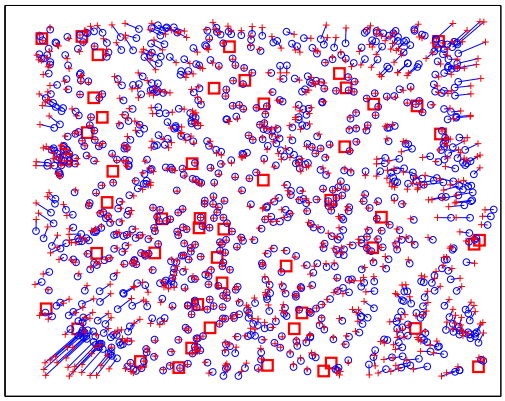


Fig. 3. Topology constructed by the distributed algorithm for a large scale network (same legend as Fig. 2).

6. CONCLUSION

In this paper, we took advantage of recent works in kernel machines for solving the localization problem in sensor networks. We showed that the matrix regression method allows us to estimate unknown positions of sensors. We derived a distributed algorithm, based on information from local neighborhood of each sensor. There are several directions for further research, including mobile ad hoc network (MANet), with an iterative update of the coordinates.

7. REFERENCES

[1] J. Bachrach and C. Taylor, "Localization in sensor networks," in *Handbook of Sensor Networks*, Ivan Stojmenovic, Ed., 2005.

[2] G. Mao, B. Fidan, and B.D.O. Anderson, "Wireless sensor network localization techniques," *Comput. Networks*, vol. 51, no. 10, pp. 2529–2553, 2007.

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

[4] O. C. Jenkins, "Relative localization from pairwise distance relationships using kernel PCA," Tech. Rep. CRES-03-010, University of Southern California, 2003.

[5] M. Essoloh, C. Richard, and H. Snoussi, "Distributed localization in wireless sensor networks from covariance measurements," in *Subm. to EUSIPCO*, Lausanne, Switzerland, August 2008.

[6] X. Nguyen, *Learning in decentralized systems: A nonparametric approach*, Ph.D. thesis, EECS Department, University of California, Berkeley, Aug 2007.

[7] S.J. Pan, J. T. Kwok, Q. Yang, and J. J. Pan, "Adaptive localization in a dynamic wifi environment through multi-view learning," in *AAAI*. 2007, pp. 1108–1113, AAAI Press.

[8] Y. Yamanishi and J.-P. Vert, "Kernel matrix regression," Tech. Rep. http://arxiv.org/abs/q-bio/0702054v1, 2007.

[9] J. Platt, "FastMap, MetricMap, and Landmark MDS are all Nystr om algorithms," in *Proc. 10th International Workshop on Artificial Intelligence and Statistics*, 2005, pp. 261–268.

[10] B. Schölkopf, R. Herbrich, and R. Williamson, "A generalized representer theorem," Tech. Rep. NC2-TR-2000-81, Royal Holloway College, Univ. of London, UK, 2000.

[11] C.K.I. Williams, "On a connection between kernel PCA and metric multidimensional scaling," *Mach. Learn.*, vol. 46, no. 1-3, pp. 11–19, 2002.

[12] C.K.I. Williams and M. Seeger, "Using the Nyström method to speed up kernel machines," in *NIPS 13*, T. Leen, T. Dietterich, and V. Tresp, Eds. 2001, pp. 682–688, MIT Press.

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