

Indoor Positioning in Peer-to-Peer Networks

Rogier Brussee, Mirela Dărău, Marta Dworczyńska, Yabin Fan, Paulien Koeleman, Piotr Kowalczyk, Jaron Samson, Nico Schlömer, Tomasz Swist, Sandra van Wijk

Abstract—We present two algorithms for indoor positioning in peer-to-peer networks. The setup is a network of two types of devices: reference devices with a known location, and user devices that can determine distances to reference devices and each other. We present two algorithms for positioning estimation in such a peer-to-peer network. The first one is purely geometric and reduces determining the best estimate for the position of a device to a least squares problem. The second approach can be considered as a probabilistic version of the geometric approach. We estimate the probability density function that a device is located at a position, given a probability density function for the positions of the other devices in the network, and a probability density function of the measured distances.

Keywords—Navigation algorithm, positioning estimate, peer-to-peer network, least squares, probability density function

I. INTRODUCTION

While Global Navigation Satellite Systems (GNSS) have many applications, in the indoor environment, their use is problematic. Even for an “open sky user” on the earth’s surface, the GNSS-signal is weak (e.g. for Galileo, the “Total Received Minimum Power on ground” is -155 dBW for E5/E6, resp. -157 dBW for E1 [1]). Indoors, the signal is even weaker, as it is attenuated by building material. As a result, the acquisition of GNSS-signals indoors is difficult, or may even be impossible.

Several approaches to indoor positioning exist. First, the sensitivity of GNSS receivers may be improved by advanced signal processing techniques. So far, in deep-indoor environments, their sensitivity remains insufficient. Second, indoor users may use navigation sensors which require no external signal such as an Inertial Navigation System (INS) or an electronic compass. Unfortunately, this also necessitates dead reckoning, whose accuracy deteriorates over time. Third, indoor users may use signals from fixed terrestrial systems e.g. WiFi or GSM base-stations. Although of major interest, the method is limited by its low accuracy of ranging, especially without a line-of-sight between user and base-station.

R. Brussee, University of Applied Sciences Utrecht, The Netherlands.

M. Dărău, Eindhoven University of Technology, The Netherlands.

Wroclaw University of Technology, Poland.

Y. Fan, Eindhoven University of Technology, The Netherlands.

P. Koeleman, Free University Amsterdam, The Netherlands.

P. Kowalczyk, University of the West of Scotland, Great Britain.

J. Samson, European Space Agency/ESTEC, The Netherlands.

N. Schlömer, University of Antwerpen, Belgium.

T. Swist, Warsaw University, Poland.

A.C.C. van Wijk, Eindhoven University of Technology, The Netherlands.

We thank Rashid Mirzavand Boroujeni (Eindhoven University of Technology, The Netherlands), Valentina Masarotto (Delft University of Technology, The Netherlands), Rob De Staelen (University of Gent, Belgium) for discussions in the workshop Mathematics and industry Centre for Mathematics and Computer science Amsterdam, January 2010 (<http://swi2010.cwi.nl/>), where most of the work in this paper was done.

The “Peer-to-peer (P2P) positioning”, approach of this paper is complementary to the methods above. Indoor users transmit and receive signals from each other to improve the navigation performance of all users involved. Examples of systems for which P2P positioning could be applied are Ultra Wide Band (UWB) [2], Wireless Access Vehicular Environment (WAVE) [3], and Bluetooth [4]. The current paper does not depend on a specific indoor positioning technology however. We only assume that devices can exchange distance and position information, and determine the relative distance to each other and a small number of reference devices. Reference devices are assumed to have a known location by having established their position using the P2P network previously, through GNSS, by being fixed to a building, or other means. We then estimate the location of the user device and the positioning error, based on error estimates for the distance measurement and the position of the reference devices.

We develop two algorithms to compute location estimates. One is based on geometrical methods and least square solutions. In the approximation that errors are small, and assuming that the reference devices have a precisely determined position, this gives a precisely and efficiently computable location estimate together with an estimate for the covariance. Equivalently, we compute a Gaussian probability density function (pdf) for the position. The other method is based on computing general pdfs. It requires neither precise positioning of the reference devices nor Gaussian localization and distance measurement errors. Unfortunately the pdf can only be computed approximately, and even that is costly.

We make several simplifying assumptions. First, we ignore all real life distance measurement problems such as systematic measuring errors (including clock bias, but see remark III.1), signals reflected from the walls, or wavefront distortion by the ionosphere. In addition, we assume that all devices are static, although this is not essential. Also, we only take into account that signals die out, by not assuming that all devices in the network are in range of each other.

The organization of this paper is as follows. In section II we set up the mathematical problem and introduce notation. In section III, we reduce the problem to an equivalent “mostly linear” problem. Under extra conditions we efficiently determine the position of a user device together with an estimate for the accuracy. In Section IV, we compute probability density functions for the position of a device, taking into account the uncertainty in the position of the other devices in range. We use it iteratively, adding user devices whose position has been determined with finite precision to the list of reference devices. We give some numerical simulations in section V. Finally, in section VI we give conclusions and suggestions for future research.

II. NOTATION AND SETUP

We model physical space as an Euclidean space with N dimensions (in practice $N = 2$ or 3). We assume further, that we have defined on physical space a fixed Euclidean coordinate system, e.g. latitude, longitude, height in city sized areas. If we speak of the coordinates of a position as a vector in \mathbb{R}^N , we always mean the coordinates with respect to this fixed coordinate system. To simplify notation, denote the inner product on \mathbb{R}^N by \cdot defined in the standard way i.e. $\mathbf{x} \cdot \mathbf{y} = \sum_{k=1}^N x^{(k)} y^{(k)}$. This defines the norm $\|\mathbf{x}\| = (\mathbf{x} \cdot \mathbf{x})^{1/2}$ and the distance $d(\mathbf{x}, \mathbf{y}) = \|\mathbf{x} - \mathbf{y}\|$. The dot product, and hence norm and the distance, are independent of the chosen Euclidean coordinate system.

1) *Setup*: Suppose now that we are given n reference devices with coordinates $X_1, \dots, X_n \in \mathbb{R}^N$, and that for each device i we have an estimate $x_i \in \mathbb{R}^N$ of X_i . Furthermore, we are given estimates d_i for the distance $d(X, X_i)$ between the position X of a user device and the position X_i of the reference device i . Our goal is to given an estimate $x \in \mathbb{R}^N$ of the position $X \in \mathbb{R}^N$. We further assume that estimation errors of the positions of the reference devices $\delta X_i = X_i - x_i$ are distributed in some known way e.g. Gaussian with mean zero and given covariance matrix. We also assume that the error in the distance estimate ε_i is distributed in a known way. Our additional goal is to estimate the positioning error $X - x$ e.g. by computing the covariance matrix.

III. TRIANGULATION

With triangulation we mean positioning using distance measurements to reference devices, and the use of Euclidean geometry. In this generality, GNSS positioning is the canonical example. In fact, we could use the methods of this paper using GNSS satellites as reference devices. However, the crucial difference in the indoor case is that we do not have an *a priori* estimate of the position with an uncertainty that is small compared to the distance to reference devices. Indoors, a user device can be anywhere between the reference devices or somewhat beyond. The corresponding problem in the GNSS case, where the satellites are in orbit with a 26000 km radius, would be to position in a 50000 km radius sphere around the earth. From this point of view, knowing that a user device is on the earth surface in western Europe, is a rather useful and good *a priori* estimate. Technically, an *a priori* estimate allows us to linearise the problem around it. Without it, the problem becomes nonlinear in an essential way. A key part of the geometric methods is therefore a simple trick to reduce the positioning problem with n quadratic equations, to an equivalent problem with n linear equations summing to zero together with a single quadratic equation. The solution of the linear sub problem then gives a good *a priori* estimate to iteratively solve the full nonlinear problem by linearization.

A. Reduction to a mostly linear problem

In this subsection we assume that measuring and positioning errors are roughly Gaussian distributed. We also assume that

the position errors of different devices are independent¹. Thus we write

$$X_i = x_i + \xi_i \quad (1)$$

where the $\xi_i = \delta X_i$ are small, independent Gaussian noise terms with mean 0 and covariance matrix C_i .

It is natural to also assume that distance measurement errors are independent and Gaussian distributed. Thus

$$\|X - X_i\| = d_i + \varepsilon_i \quad (2)$$

where ε_i is Gaussian with mean zero and standard deviation σ (here we ignore that distances are positive). We assume that the standard deviation σ , and the standard deviation of ξ ($\text{var}(\|\xi\|^2)^{1/2} = \text{tr}(C_i)^{1/2}$) are small compared to the distance d_i , and we only keep the lowest order in the noise terms. Using \doteq for the first order approximation we have

$$\|X - x_i\|^2 \doteq d_i^2 + 2\xi_i \cdot (X - x_i) + 2d_i\varepsilon_i \quad (3)$$

The problem is now to estimate the mean x , and a covariance matrix $C = \text{cov}(X)$ of X , at least approximately. In general, there is no closed or algorithmic solution for a system of coupled quadratic equations. In fact a brute force numerical approach minimising the total error with only 4 circles already caused trouble for MATLAB. Because in this case we are intersecting only spheres, the system can fortunately be simplified to a “mostly linear” system.

For a function $f(y)$ on \mathbb{R}^N , denote by

$$\overline{f(x)} = \overline{f(x)} = \frac{1}{n} \sum_i f(x_i)$$

the average of quantities depending on the estimated position of reference device i . Averaging the identity

$$\|X - x_i\|^2 = \|X - \bar{x}\|^2 - 2(X - \bar{x}) \cdot (x_i - \bar{x}) + \|x_i - \bar{x}\|^2$$

we get the identity

$$\overline{\|X - x\|^2} = \|X - \bar{x}\|^2 + \overline{\|x_i - \bar{x}\|^2}. \quad (4)$$

By subtracting identity (4) from the n equations (3) we see that they are equivalent (up to second order terms in the errors) to a system of n linear equations that sum to zero, together with a single quadratic equation

$$\|X - (\bar{x} + \bar{\xi})\|^2 \doteq \overline{d^2} - \overline{\|x_i - \bar{x}\|^2} + 2\bar{\xi} \cdot (x - \bar{x}) + 2\overline{d\varepsilon}, \quad (5)$$

$$\begin{aligned} & 2(X - (\bar{x} + \bar{\xi})) \cdot ((x_i - \bar{x}) + (\xi_i - \bar{\xi})) \doteq \\ & -(d_i^2 - \overline{d^2}) + (\|x_i - \bar{x}\|^2 - \overline{\|x_i - \bar{x}\|^2}) \\ & - 2(d_i\varepsilon_i - \overline{d\varepsilon}) + 2(x_i - \bar{x}) \cdot (\xi_i - \bar{\xi}) - 2(x - \bar{x}) \cdot \bar{\xi}. \end{aligned} \quad (6)$$

The system of linear equations (6) has the standard form $\mathbf{A}(X - \mathbf{a}) = \mathbf{b}$ with \mathbf{A} , \mathbf{a} and \mathbf{b} coefficients with a Gaussian noise term. Because there is a noise term in \mathbf{A} , and not just in \mathbf{b} , finding the best estimate for $X - \mathbf{a}$ (and hence X), is a so called total least squares problem [5], [6]. Such problems can be completely analyzed using singular value

¹Note The independence assumption is restrictive, as we want to raise the status of a device to that of a reference device. However, we do this by determining its position which depends on all the other positions.

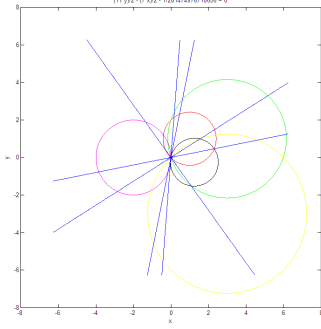


Fig. 1: The position problem for 4 devices in 2D can be formulated as intersection 4 circles or as 4 lines and a single circle. A fifth line is the linearization of the mean distances circle in the intersection point

decompositions. Unfortunately such decompositions are highly nonlinear, expensive to compute and make the dependence of errors and positions unclear.

B. Least square solutions

We now make the further simplifying *assumption* that the errors in the positions of the reference devices ξ_i can be neglected. We then get the much simpler equations

$$\|X - \bar{x}\|^2 \doteq \bar{d}^2 - \|x_i - \bar{x}\|^2 + 2\bar{d}\varepsilon_i, \quad (7)$$

$$2(x_i - \bar{x}) \cdot (X - \bar{x}) \doteq -(d_i^2 - \bar{d}^2) + \|x_i - \bar{x}\|^2 - \|x_i - \bar{x}\|^2 - 2(d_i\varepsilon_i - \bar{d}\varepsilon_i). \quad (8)$$

The linear part (8) of these equations has the standard least square problem form

$$\mathbf{A}(X - \mathbf{a}) = \mathbf{b} + \delta\mathbf{b},$$

with the error term $\delta\mathbf{b}$ on the constant part \mathbf{b} . We can now take the least square solution of the linear equations (8) only. If the rank of the system is maximal (i.e. if the position of the devices is generic and $n - 1 \geq N$), then the least square solution x_{ls} of the least square problem will be a good estimator for X . Still assuming $n - 1 \geq N$ and the reference devices are in general position, the well-known explicit formula for the least square solution gives it as

$$x_{\text{ls}} = \mathbf{a} + (\mathbf{A}^t \mathbf{A})^{-1} \mathbf{A}^t \mathbf{b}. \quad (9)$$

Remark III.1. We can relax the condition that ε_i has mean 0, by assuming it suffers from a systematic overestimation e , i.e. that the errors ε_i are of the form

$$\varepsilon_i = e + \theta_i$$

where only θ_i has mean 0 and is small (as in the case of GNSS positioning). If we only keep terms linear in the θ_i the term $-2(d_i\varepsilon_i - \bar{d}\varepsilon_i)$ gets replaced by

$$-2((d_i - \bar{d})e + (d_i\theta_i - \bar{d}\theta_i) + e(\theta_i - \bar{\theta})),$$

with the last term coming from a ε^2 term which we can no longer neglect. Likewise the $2\bar{d}\varepsilon_i$ term in (7) becomes

$$e^2 + 2\bar{d}e + 2e\bar{\theta}_i + 2\bar{d}\bar{\theta}_i.$$

It follows that e can be estimated together with X , by the methods below. If $n \geq N + 2$ we can either directly use (8) which is linear in e , or if $n = N + 1$, we linearly express the coordinates X in terms of e and solve the quadratic equation in e resulting from (7).

However, we can do better than (9) as we could, and should, take the covariance of the \mathbf{b} term into account. While the error terms $\delta\mathbf{b}$ in the right hand side of this extended system are assumed (approximately) Gaussian, they are neither independent, nor identically distributed. Fortunately, with our independence assumptions on the ε_i 's, the covariance matrix Δ of \mathbf{b} is easy to compute explicitly:

$$\Delta_{ij} = \begin{cases} 4\sigma^2((1 - 2/n)d_i^2 + (1/n)\bar{d}^2) & \text{if } i = j, \\ (4/n)\sigma^2(\bar{d}^2 - d_i^2 - d_j^2) & \text{if } i \neq j. \end{cases} \quad (10)$$

The standard least square solution x_{ls} maximizes the likelihood of a solution, assuming that the error term $\delta\mathbf{b}$ is *standard* Gaussian distributed. By the explicit form of the Gaussian, this is equivalent to minimising the norm of the error term $\|\mathbf{A}(x - \mathbf{a}) - \mathbf{b}\|^2$. In the general case, setting

$$\Delta = \text{cov}(\delta\mathbf{b}),$$

the maximum likelihood solution is the minimiser x_{ml} of

$$E = (\mathbf{A}(x - \mathbf{a}) - \mathbf{b})^t \Delta^{-1} (\mathbf{A}(x - \mathbf{a}) - \mathbf{b}) = \|\Delta^{-1/2} (\mathbf{A}(x - \mathbf{a}) - \mathbf{b})\|^2. \quad (11)$$

Therefore, the maximum likelihood solution is the standard least square solution of the system where both sides of the equation are multiplied with $\Delta^{-1/2}$. By the explicit form of the least square solution (9) and some matrix algebra, the minimiser is given by

$$x_{\text{ml}} = \mathbf{a} + (\mathbf{A}^t \Delta^{-1} \mathbf{A})^{-1} \mathbf{A}^t \Delta^{-1} \mathbf{b}. \quad (12)$$

Conveniently, (12) contains no matrix square root.

On closer examination, this expression must be modified however. We cannot simply take the inverse of the covariance matrix, because Δ is singular. The reason is, that because the linear equations sum to zero, the noise term $\delta\mathbf{b}$ is contained in the hyperplane

$$H : \sum_{j=1}^n y_j = \sqrt{n} w^t y = 0.$$

Here $w = n^{-1/2}(1, 1, \dots, 1)^t$ is the normal of H normalised to length 1. Hence the image of Δ is contained in H , and $w^t \Delta = (\Delta w)^t = 0$. For the same reason, the image of \mathbf{A} is also contained in H . Abstractly, this solves the problem: we simply reformulate the least squares problem as taking values in the Euclidean space H , rather than in \mathbb{R}^n . We then choose an orthonormal basis of H , and compute both the covariance matrix $\Delta|_H$ and its inverse with respect to this basis. Instead of (11), we minimise

$$\tilde{E} = \|\Delta|_H^{-1/2} \text{Pr}_H (\mathbf{A}(X - \mathbf{a}) - \mathbf{b})\|_H^2, \quad (13)$$

where the orthogonal projection on H is defined by

$$\text{Pr}_H = (1 - ww^t) \quad (14)$$

This abstract point of view is inconvenient computationally. However, we can use the following trick. Choose $\lambda > 0$ and define the matrix

$$\tilde{\Delta} = \Delta + \lambda ww^t. \quad (15)$$

Using equation (14), definition (15) can be rewritten as $\tilde{\Delta} = \text{Pr}_H \Delta \text{Pr}_H + \lambda \text{Pr}_{H^\perp}$. Equivalently, with respect the basis of \mathbb{R}^n consisting of an orthonormal basis of H together with w , the matrix of $\tilde{\Delta}$ is of the form $\tilde{\Delta} = \begin{pmatrix} \Delta|_H & 0 \\ 0 & \lambda \end{pmatrix}$, hence $\tilde{\Delta}^s = \begin{pmatrix} \Delta|_H^s & 0 \\ 0 & \lambda^s \end{pmatrix}$. In particular, $\tilde{\Delta}$ is invertible if and only if $\Delta|_H$ is. Now the norm (13) can be rewritten as

$$\tilde{E} = \|\tilde{\Delta}^{-1/2} (\mathbf{A}(X - \mathbf{a}) - \mathbf{b})\|^2. \quad (16)$$

This is because $\tilde{\Delta}^{-1/2}$ restricts to $\Delta|_H^{-1/2}$ on H , the norm on H is the restriction of the norm on \mathbb{R}^n , and the image of \mathbf{A} and the vector \mathbf{b} are contained in H . In particular, \tilde{E} does not depend² on λ . The minimiser of \tilde{E} is therefore only slightly modified compared to (12) and becomes

$$x_{\text{ml}} = \mathbf{a} + \left(\mathbf{A}^t \tilde{\Delta}^{-1} \mathbf{A} \right)^{-1} \mathbf{A}^t \tilde{\Delta}^{-1} \mathbf{b}. \quad (17)$$

We have also thrown away information from the quadratic term which we should not. However we now have a good 0th order estimate x_{ml} for the location of X . We can therefore linearise the remaining equations around this estimate to get a linear system (denoted by using \approx).

$$2(x_{\text{ml}} - \bar{x}) \cdot (X - \bar{x}) \approx \overline{d^2} - \|x_{\text{ml}} - \bar{x}\|^2 + 2\overline{d\varepsilon} \quad \text{for } i = 0 \quad (18)$$

$$2(x_i - \bar{x}) \cdot (X - \bar{x}) \doteq -(d_i^2 - \overline{d^2}) + (\|x_i - \bar{x}\|^2 - \|x_{\text{ml}} - \bar{x}\|^2) - 2(d_i \varepsilon_i - \overline{d\varepsilon}) \quad \text{for } i = 1, \dots, n \quad (19)$$

Compared to (8), this is an extension with one extra row of the linear system (8), which is also of the form $\hat{\mathbf{A}}(X - \hat{\mathbf{a}}) = \hat{\mathbf{b}} + \delta\hat{\mathbf{b}}$. Again we can compute the covariance matrix of $\delta\hat{\mathbf{b}}$:

$$\hat{\Delta}_{ij} = \begin{cases} (4/n)\sigma^2 \overline{d^2} & \text{if } i = j = 0, \\ (4/n)\sigma^2 (\overline{d^2} - d_j^2) & \text{if } i = 0, j \neq 0, \\ 4\sigma^2 ((1 - 2/n)d_i^2 + (1/n)\overline{d^2}) & \text{if } i = j \geq 1, \\ (4/n)\sigma^2 (\overline{d^2} - d_i^2 - d_j^2) & \text{if } i \neq j, i, j \geq 1. \end{cases} \quad (20)$$

The matrix $\hat{\Delta}$ is not invertible because the image is contained in the hyperplane

$$\hat{H} : \sum_{j=1}^n y_j = \sqrt{n} \hat{w}^t y \subseteq \mathbb{R}^{n+1}$$

where $\hat{w}^t = n^{-1/2}(0, 1, \dots, 1)$. However like in (13) we can formulate the maximum likelihood solution as the minimiser of a problem on \hat{H} . However, like in equation (15), we

²For numerical stability one should choose λ small to insure that the relation $w^t(\mathbf{A}(x - \mathbf{a}) - \mathbf{b}) = 0$ is satisfied with high precision. On the other hand to make the inversion not too poorly conditioned it should be of the order of the coefficients of Δ (e.g. $\lambda = (1/10)\sigma^2/n$).

can define $\tilde{\hat{\Delta}} = \hat{\Delta} + \lambda \hat{w} \hat{w}^t$, and as in (16), minimise $\tilde{E} = \|\tilde{\hat{\Delta}}^{-1/2} ((\hat{\mathbf{A}}X - \hat{\mathbf{a}}) - \hat{\mathbf{b}})\|^2$ which does not depend λ . As in (17) its minimiser is given by

$$\hat{x}_{\text{ml}} = \hat{\mathbf{a}} + \left(\hat{\mathbf{A}}^t \tilde{\hat{\Delta}}^{-1} \hat{\mathbf{A}} \right)^{-1} \hat{\mathbf{A}}^t \tilde{\hat{\Delta}}^{-1} \hat{\mathbf{b}} \quad (21)$$

If we go further and *define* X_{ml} as a random variable through

$$X_{\text{ml}} = \mathbf{a} + \left(\hat{\mathbf{A}}^t \tilde{\hat{\Delta}}^{-1} \hat{\mathbf{A}} \right)^{-1} \hat{\mathbf{A}}^t \tilde{\hat{\Delta}}^{-1} (\hat{\mathbf{b}} + \delta\hat{\mathbf{b}})$$

then X_{ml} is Gaussian distributed with mean \hat{x}_{ml} . Moreover setting $\mathbf{C} = \left(\hat{\mathbf{A}}^t \tilde{\hat{\Delta}}^{-1} \hat{\mathbf{A}} \right)^{-1}$, we compute the covariance as

$$\text{cov}(X_{\text{ml}}) = \mathbf{C} \hat{\mathbf{A}}^t \tilde{\hat{\Delta}}^{-1} \text{cov}(\delta\hat{\mathbf{b}}) \tilde{\hat{\Delta}}^{-1} \hat{\mathbf{A}} \mathbf{C} = \mathbf{C}. \quad (22)$$

The probability density function of X_{ml} can thus be completely explicitly given as

$$p_{\text{ml}}(x) = \det(2\pi\mathbf{C})^{-\frac{1}{2}} \exp\left(-\frac{1}{2}(x - \hat{x}_{\text{ml}})^t \mathbf{C}^{-1} (x - \hat{x}_{\text{ml}})\right) \quad (23)$$

IV. PROBABILITY DENSITY FUNCTIONS ALGORITHM

Until now, even though a stochastic approach has been in the background, we could have presented everything in terms of error estimation. We now strengthen the stochastic aspect. We compute the probability distribution of the position of a device, based on measurements with known accuracy of the distance to neighboring devices, and the probability distribution of the position of the reference devices. We then use an iterative algorithm, where in each step a device in a P2P network can alter, and hopefully improve, its localization pdf.

A. Model and notation

There are two kinds of uncertainties involved in determining the position of a user device: uncertainty in the distance measurement, and the uncertainty in the location of the other devices. We capture these uncertainties in two random variables. As before, assume space has N dimensions and assume that an Euclidean coordinate system is fixed beforehand.

We denote by X_k the \mathbb{R}^N valued random variable of the coordinates of the unknown position of reference device k . We assume X_k has a probability density function (pdf) $p_k(x)$ which can be *any* non negative function on \mathbb{R}^N with integral 1. We have already found the pdf of a device when using the least squares method of III in equation (23).

Suppose we are given the pdf $p_D(r)$ of the random variable $D = d(X, a)$, the distance of X to a fixed point a . For example, if we measure a distance r_0 with a relatively small Gaussian error σ , we could use a Gaussian with mean r_0 and standard deviation σ . On the other hand, if the errors are due to quantization, we would use a uniform distribution. We want to determine the pdf of X assuming $p_D(r)$ is all the information we have (Figure 2a). Clearly $X = a + \Delta X$ where $\Delta X = X - a$. Then $p_X(x) = p_{\Delta X}(x - a)$. The pdf $p_{\Delta X}(x')$ on \mathbb{R}^N only depends on the radius $r = \|x'\|$, and so, by abuse

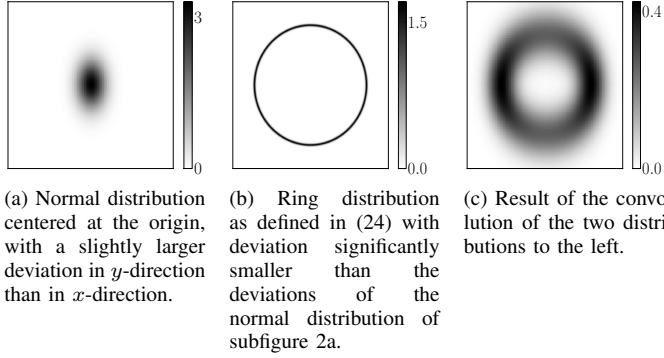


Fig. 2: Illustration of the convolution process (25). Subfigure 2a shows the convolution data, subfigure 2b the convolution kernel, and (2c) the actual convolution of the two. Note how smaller deviation in x -direction of the bivariate Gaussian distribution increases the values of the convolution at the x -extreme ends of the blurred circle.

of notation, can be written as $p_{\Delta X}(r)$. This function of r is not just p_D however. Instead we have

$$\begin{aligned} p_D(r)dr &= \int_{r \leq \|\Delta X\| \leq r+dr} p_{\Delta X}(\|x'\|) d^N x' \\ &= \text{Vol}(S^{N-1}) p_{\Delta X}(r) r^{N-1} dr \end{aligned}$$

where $\text{Vol}(S^{N-1})$ is the volume of the $N-1$ dimensional sphere (i.e. 2π for $N=2$ and 4π for $N=3$). Hence

$$p_{\Delta_1 X}(x') = p_D(\|x'\|) \|x'\|^{-(N-1)} \text{Vol}(S^{N-1})^{-1}. \quad (24)$$

If p_D is sharply peaked around r_0 we can replace r^{N-1} by r_0^{N-1} and absorb r_0^{N-1} in the normalization constant. The difference between p_D and p_{Δ} can then be mostly ignored.

B. Distance measurements to a single reference device

We estimate the pdf for the location of a device X , given the pdf $p_{X_1}(x)$ of a single reference device X_1 , and the pdf $p_{D_1}(r)$ for the distance $d(X, X_1)$ between X and X_1 . Clearly, setting $\Delta_1 X = X - X_1$, we can write

$$X = X_1 + \Delta_1 X.$$

We have just determined the pdf of $\Delta_1 X$ in equation (24). Since we *assume* the errors in the distance measurement and the location of X_1 are independent, the random variables X_1 and $\Delta_1 X$ are independent as well. It is then standard that the pdf p of X is given by the convolution

$$\begin{aligned} p(x) &= (p_1 * p_{\Delta_1 X})(x), \\ &= \int p_1(x - x') p_{\Delta_1 X}(x') d^N x'. \end{aligned} \quad (25)$$

For numerical computations (as in Figure 2), we use the well known fact that convolution can be computed using the Fourier transform \mathcal{F} .

$$p(x) = (\mathcal{F}^{-1}(\mathcal{F}(p_1)\mathcal{F}(p_{\Delta_1 X}))) (x). \quad (26)$$

C. Distance measurements to multiple reference devices

We now change notation slightly. We denote the position random variable determined by doing a measurement to reference device k by $X^{(k)}$ with pdf $p^{(k)}$. For each device k , we think of $X^{(k)}$ as the position of a different observer, with a functionally identical user device. In particular, observers of different devices are allowed to be at different positions. We can then *assume* that the locations $X^{(k)}$ are independent random variables, i.e. that

$$P(X^{(1)} \in \Omega_1 \wedge \dots \wedge X^{(n)} \in \Omega_n) = \prod_{k=1}^n P(X^{(k)} \in \Omega_k) \quad (27)$$

for spatial regions $\Omega_1, \dots, \Omega_n \subseteq \mathbb{R}^N$. We can determine the pdf $p^{(k)}$ as in (25). In fact, we can allow more general pdfs, as long as their determination does not involve the other devices $l \neq k$ since this would break independence. For example, we could formally define a “collective” device, $k = \infty$ say. Through distance measurements to all accurately known reference devices, it gives us a Gaussian distribution $p^{(\infty)}$ as in (23).

What we want to determine however, is the pdf p of the location X of a single device, from n distance measurements, given that these measurements are made at the same location³. With the above independence assumption, one can show (c.f. appendix A) that the pdf is proportional to the product of the pdf’s of the $X^{(k)}$ i.e. $p(x)$ is of the form

$$p(x) = I^{-1} \prod_k p^{(k)}(x). \quad (28)$$

The normalization constant I is uniquely determined by the normalization condition $\int p(x) d^N x = 1$. This is satisfying, as taking the product of pdf’s is the analog of intersecting subsets.

D. Positioning in peer to peer to peer networks

The analysis in the previous section IV-C leads to a natural iterative algorithm to determine the pdf of a device in a network of communicating peer devices. Starting from a small number of reference devices with a well-known location, the position information is propagated through the network. This effectively creates new reference devices, although typically with a less accurately known position. As in any iterative algorithm we need a good starting point for the iteration. Therefore we have to find a good initial pdf. If reference devices with small position errors (compared to distance errors) are in range, equation (23) gives us such a distribution. Otherwise we have to use some coarse initial guess for the initial distribution, e.g. anywhere in a GSM cell or anywhere in a shopping mall. Here, we simply assume that such a dumb guess can be modeled by a uniform distribution in some region Ω . Since distributions are normalised at the end, we can ignore normalising constants in the computation.

³The “same location” is based on the assumption of static devices. For example, if only the user device is moving with velocity $v(t)$ and the measurement to reference device 2 is done T seconds after that to 1 we would want to have $X^{(2)} = X^{(1)} + \int_0^T v(t) dt$.

Algorithm 1 One update step for the pdf of a client. Information about previous updates is entirely discarded, a user device could store data of the previous measurements to decrease the uncertainty of the distance to its neighbors' positions. In the simulation, the mean of all previous computations is stored in each step and updated according to the new measurement. This assumes that none of the devices change their position or that movement is measured and compensated for.

```

for all devices within reach do
  request accuracy
end for
if # accurate reference devices  $\geq N + 1$  then
  for all accurate devices  $k$  do
    measure the distance to  $k$  with some accuracy
  end for
   $p \leftarrow$  Gaussian determined through least squares see (23)
else
   $p \leftarrow$  uniform distribution over  $\Omega$ 
end if
for all devices  $k$  within reach not already used do
  measure the distance  $d_k$  to  $k$  with some accuracy  $\delta d_k$ 
   $p_{D_k} \leftarrow p(d_k, \delta d_k)$  (e.g. Gaussian distribution)
   $p_{\Delta_k X} \leftarrow r^{-(N-1)} p_{D_k}$ , see (24)
  request the pdf  $p_k$  of client  $k$ 
   $p^{(k)} \leftarrow p_k * p_{\Delta_k X}$ , using FFT see (26)
   $p \leftarrow p \cdot p^{(k)}$ 
end for
normalize  $p$ 
broadcast  $p$  on request

```

In the algorithm we assume that positions and measurements are independent. However, clearly this is no longer the case once the algorithm is run, as the position of one device is determined by all other devices. The simulation seems to suggest that this is not a serious problem however.

V. SIMULATION

In this section, a few example constellations are set up. We iterate over a number of time steps, to see how the pdfs of the individual users evolve when more and more accurate data of the other users becomes available. Each of the following example setups consists of a number of user devices \otimes positioning themselves by distance measurements to reference devices \odot (for satellite). These are located in a two dimensional square-shaped domain Ω with edge length normalized to 1. All lengths, including standard deviations are in these units.

Since we were mainly interested in testing whether the algorithm works and converges, we always start from the dumb initial guess for the pdf i.e. the uniform distribution over the domain, even when the number of reference devices is 3 or more. There is always at least one reference device in the domain, for otherwise the result is not interesting. It is assumed that the pdf of reference device is the normal distribution centered at its true position x_{sat} with $\sigma_{\text{sat}} = 0.05$ independently for all reference devices.

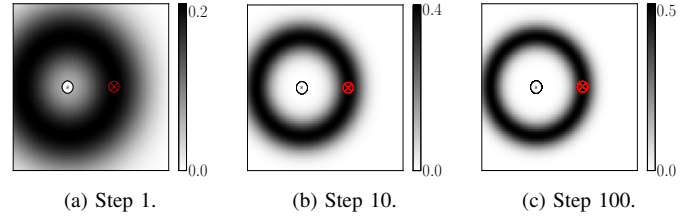


Fig. 3: Experiment with one user device (the red \otimes) and one reference device (\odot) after multiple executions of algorithm 1. The pdf of the user device is shown. After the first measurement, the user device can only determine its position up to rotation around the reference device. In this experiment we average the distances measured so after more steps, the width of the rings grows smaller as the deviation of the distance estimation is diminished. After 100 steps, the fuzziness of the distribution mainly depends on σ_{sat} rather than the uncertainty in the distance measurements.

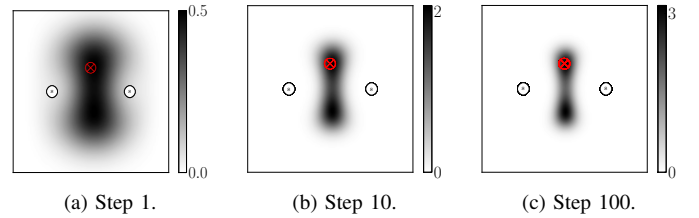


Fig. 4: Experiment with one user device (the red \otimes) and two reference devices (\odot) after multiple executions of algorithm 1. After distance measurements to both of the reference devices, the user devices are localised in the regions where the two fuzzy rings overlap, i.e., where their product is locally maximal. The fact that there are two “hot spots” cannot be overcome because the positioning problem is symmetric in the line through the reference devices. Hence there is not enough information to prefer one over the other. Again the uncertainty about the distance measurements is filtered out by sampling over 100 steps and averaging the distance.

With this setup, each of the user devices now iteratively executes Algorithm 1 to update its pdf given pdfs of its direct neighbours. The simulation is run for six settings, see Figures 3 to 8 We depict the pdfs of one user device after 1, 10, and 100 steps. Typically, the pdf becomes more and more concentrated in one or multiple locations (when the data is insufficient to determine a unique location). For each setting, we discuss the setup and the results.

VI. CONCLUSION AND DISCUSSION

In this paper we presented two promising methods for solving the problem of computing device locations from information about neighbour locations.

The first method uses least square techniques and Euclidean geometry. It is comparatively simple to analyse, implement and execute on a device but in the simple form presented here it is limited to the case where reference devices are well localised

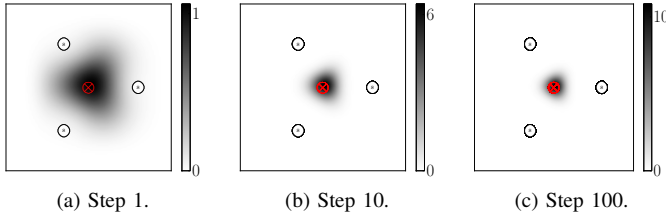


Fig. 5: Experiment with one user device (the red \otimes) and three reference devices (\odot) after multiple executions of the algorithm 1. As opposed to figures 3 and 4. After one measurement one “hot spot” is found that likely contains the devices position. As more samples in distance measurement are taken, the uncertainty σ_{sat} dominates for the user device as well.

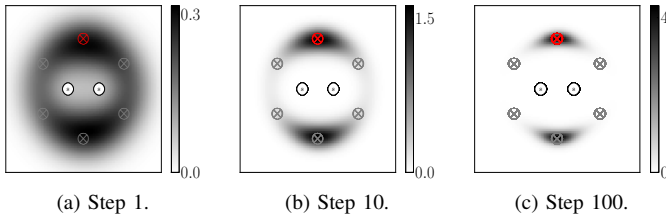


Fig. 6: Experiment with six user devices (\otimes) and two reference devices (\odot) after multiple executions of algorithm 1. No single user device has contact with both the user devices, but the localization information propagates through the network to each of the user devices, such that they all know they have pdfs similar to the one in figure 4 rather than figure 3.

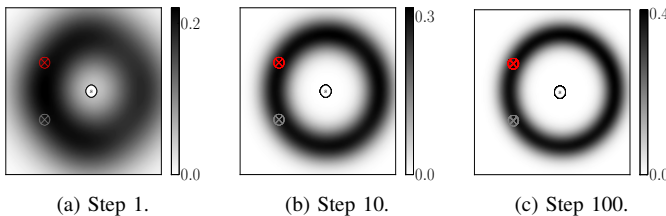


Fig. 7: Experiment with two user devices (\otimes) and one reference device (\odot) after multiple executions of algorithm 1. The situation is similar to figure 3, except that the two clients with (initially) uncertain position, improve their radial localization by exchanging position information relative to each other. For the collection of devices as a whole, the problem is invariant under rotations so this extra information does not improve an estimate of the users angular position. This experiment shows that adding clients without information on their absolute positions does not alter the uncertainty of absolute localizations of user devices.

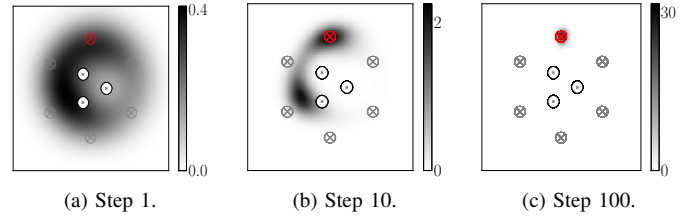


Fig. 8: Experiment with six user devices (\otimes) and three reference devices (\odot) after multiple executions of algorithm 1. No device ever has contact with more than one reference device. Curiously, the pdf of the user device has a curved shape with a preferred direction in space even after one step, likely related to the particular (not fully rotational symmetric) arrangement of the reference devices within the hexagon of user devices. After a few steps, two “hot spots” are formed which is rather similar to the much simpler situation of figure 4. One of the “hot spots” is indeed the true position of the user device. After a few more steps, this location is preferred over the other local maximum, and the user device eventually gets a good estimate for its position.

compared to typical errors in the distance measurement. We also had to assume that distance errors were Gaussian, and that there are enough ($N + 1$ i.e. 3 in 2d, resp. 4 in 3d) reference devices with absolutely determined position. The output of the algorithm is an estimate for the location of the user device and a covariance matrix for the errors in the positioning. Equivalently, its output is a Gaussian probability density function (pdf) for the location of the device.

The second method uses complete information in the form of probability density functions. In theory, it is more flexible and gives more information than the first method. However, this comes at the price of having to exchange far more information between devices and using a far more computation-intensive algorithm. Also we have to rely on an iterative algorithm which must converge. This makes it rather less clear what the precise performance of the algorithm is. It would in fact be good to compare the results of the pdf method and the least square method in the case both are applicable. The pdf approach does give reasonable outcomes in multiple settings of the simulation however: usually the peak of the pdf coincides with the true position of the client. If it was impossible to get any more specific information than a localization in two hotspots or a ring (see for example Figure 6 or 5 there is a clear symmetry which forces this result upon us. This means that, all algorithm that uses the same information have this “problem”.

In both cases, information has to be transmitted between peers and computation has to be done. However, for the second method, far more information has to be transmitted to represent a complete pdf, and the amount of computation that has to be done should probably be reduced to make it more suitable for practical use, ideally with minimal loss in performance. This holds in particular for very fine-grained discretizations of the clients environment. An alternative for communicating

an entire pdf is approximating it, where appropriate, by a multivariate Gaussian distribution, which is determined by a few parameters. As only a few real-valued parameters have to be exchanged in this case, this would also dramatically reduce the amount of data to be transferred. One would then need a clear conditions to fill in the “where appropriate” (e.g. in terms of the Jensen Shannon divergence of the true distribution with respect to the Gaussian with the same mean and variance). Clearly, a bivariate Gaussian distribution might not always be suitable see for example Figures 3 and 6. However, these are the simple cases which can be approximated by extending the arsenal of distributions somewhat (e.g by a ring shaped distribution and a product of a degenerate Gaussian and a ring-shaped distribution). It is left to further research to investigate how much decline in performance this gives and which approximation of the pdf is appropriate.

There are many possibilities for further research. An important first point for the pdf method is whether the algorithm converges, and if so in what sense. From our experiments we are confident that it converges to a location or more precisely a probability distribution of a location for every device in the network. Unfortunately that does not necessarily mean that this pdf has a peak at roughly the correct position, or that the peak is unique if enough information has been fed in the system that it should, i.e. that $N+1$ ($= 3$ (resp. 4) in 2d (resp. 3d)) reference devices with a reasonable estimate for the absolute positions are known in the network. The conditions under which it actually does, would be interesting, as would be the rate of convergence.

An option for further improvement might also be taking previous measurements into account by weighing them with the current one, and using this for the updating of the pdf. Currently, only the most recent measurement is taken into account, which might, by chance, have a large error. In an early stage of the updating process, this might cause the pdf to concentrate about an erroneous location, far off the true location, from which it needs many extra measurements to get back to the true location. On the other hand, putting too much weights on previous measurements makes it last longer before the pdf concentrates around a location.

Finally, these algorithms should be extended to include moving devices, that is, people walking around in a building. Such an extension should not be difficult simply by treating a relative velocity measurement as the addition of another distance measurement. This makes the questions on speed of convergence and ease of computation even more relevant.

REFERENCES

- [1] OD SIS ICD, “European open service signal in space interface control document,” EC, Tech. Rep., February 2010.
- [2] Wymeersch, H. et al., “Cooperative localization in wireless networks,” *Proceeding of the IEEE*, pp. 97–427, February 2009.
- [3] Cozetti, H. A. et al., “Comparative analysis of IEEE802.11p and MS-Aloha in vanet scenarios,” in *Proceedings of Second IEEE International Workshop on Vehicular Networking, VON-09*. IEEE, December 2009.
- [4] F. Forno, G. Malnati, and G. Portelli, “Design and implementation of a Bluetooth ad hoc network for indoor positioning,” in *Software, IEE Proceedings-*, vol. 152, no. 5. IET, 2005, pp. 223–228.
- [5] S. Van Huffel and J. Vandewalle, *The total least squares problem: computational aspects and analysis*. Society for Industrial Mathematics, 1991.

- [6] P. d. Groen, “Introduction to total least square,” *Nieuw Archief voor Wiskunde*, pp. 237–253, 1996.

APPENDIX

Proofsketch of (28). We have to compute

$$\int_{\Omega} p(x) = P(X \in \Omega \mid X = X^{(1)} = X^{(2)} = \dots = X^{(n)})$$

for all sufficiently small Ω . It is technically convenient to relax the condition “all locations equal”, to “all locations within distance ϵ of each other” and take the limit $\epsilon \rightarrow 0$. We therefore want to compute

$$P(X^{(1)} \in \Omega \wedge \dots \wedge X^{(n)} \in \Omega \mid \|X^{(k)} - X^{(l)}\| < \epsilon).$$

for sufficiently small Ω . Things are clearer if more generally we compute

$$P(X^{(j)} \in \Omega_j \mid \|X^{(k)} - X^{(l)}\| < \epsilon) = \quad (29)$$

$$P(\|X^{(k)} - X^{(l)}\| < \epsilon \mid X^{(j)} \in \Omega_j) \quad (30)$$

$$\times P(X^{(j)} \in \Omega_j) \quad (31)$$

$$/P(\|X^{(k)} - X^{(l)}\| < \epsilon). \quad (32)$$

where the indices j, k and l run over all devices $1, \dots, n$. Here we assume that (31) and (32) are non zero, since otherwise (29) is zero. Now (32) does not depend on the Ω_k and can therefore be absorbed in the normalization constant. Expression (30) can be written as

$$\frac{\int_{\|x^{(k)} - x^{(l)}\| < \epsilon, x^{(k)} \in \Omega_k} p^{(1)}(x^{(1)}) \dots p^{(n)}(x^{(n)})}{\int_{x^{(k)} \in \Omega_k} p^{(1)}(x^{(1)}) \dots p^{(n)}(x^{(n)})} \quad (33)$$

The pdf’s $p^{(k)}$ can be approximated (in L_1) by staircase functions, so for the Ω_k sufficiently small, by constants on $\Omega_1 \times \dots \times \Omega_n$. The quotient (33) then simplifies to

$$\frac{\text{Vol}(\{\|x^{(k)} - x^{(l)}\| < \epsilon\} \cap \Omega_1 \times \dots \times \Omega_n)}{\text{Vol}(\Omega_1 \times \dots \times \Omega_n)}$$

which to lowest order in ϵ is given by

$$\frac{\text{Vol}(B^{N(n-1)}(0, \epsilon)) \text{Vol}(\Omega_1 \cap \dots \cap \Omega_n)}{\prod_{k=1}^n \text{Vol}(\Omega_k)}.$$

Hence combining, we conclude that

$$\frac{P(X \in \Omega)}{\text{Vol}(\Omega)} \propto \prod_k \frac{P(X^{(k)})}{\text{Vol}(\Omega)}.$$

Shrinking Ω to an infinitesimal we find (28).