Joint Coordinate Optimization in Fingerprint-Based Indoor Positioning

Mohammad Nabati, Seyed Ali Ghorashi, and Reza Shahbazian

***Abstract*—Fingerprint-based indoor positioning uses pattern recognition algorithms (PRAs) to estimate the users' locations in wireless local area network environments, where satellite-based positioning methods cannot work properly. Traditionally, the training phase of PRA is separately conducted for x and y coordinates. However, the received signal strength from access points is a unique fingerprint for each measured point, not for x and y coordinates separately. In this letter, we propose a method to jointly employ the x and y coordinates during the training phase using a novel PRA-based Gaussian process regression (GPR), named 2D-GPR. Experimental results show that the proposed 2D-GPR improves the accuracy of positioning more than $40cm$ in limited data samples and has a lower calculation cost compared with conventional GPR.**

***Index Terms*—wireless local area network, fingerprint-based positioning, machine learning, pattern recognition.**

I. Introduction

SATELLITE signals penetration is not enough for an accurate positioning process [1], [2]. The satellites use ranging information, which suffers from non-line-of-sight error and does not provide the desired accuracy for indoor areas [3]. The fingerprint-based wireless local area network (WLAN) positioning is proposed for indoor environments to reach a high accuracy positioning [3]. In the training (offline) phase of this method, the received signal strength (RSS) vectors from several access points (APs) are captured at reference points (RPs) in the WLAN environment. Then, a pattern recognition algorithm (PRA) is optimized to convert the RSS vectors to the Cartesian coordinates.

The conventional PRAs have been constructed for single output scenarios, and cannot be used to jointly estimate the x and y coordinates. These algorithms, such as conventional Gaussian process regression (CGPR) [4], support vector regression (CSVR) [5], and random forest (CRF) [6] have been used for positioning purposes, whose optimizations are performed for x and y coordinates separately. Some GPR-based algorithms have been developed for multi-task learning problems [7] to jointly extract different tasks' information. In the context of multi-task learning, the input features are different for each task, and the complexity of these techniques is at a high level, which is not suitable for a positioning system due to the battery consumptions and delays. Also, GPR-based multi-task learning algorithms have been designed for correlated outputs. However, the x and y are not necessarily correlated in the WLAN environment. Besides, the uniqueness of the RSS vector as a fingerprint corresponds to the RP location, not its x and y coordinates separately, and the PRAs can be modified by considering this fact. It is possible to jointly estimate the x and y coordinates via the k-nearest neighbors-based algorithms [2]. However, they work only based on input features and do not consider the physical locations of data within an optimization process. Also, deep neural networks (DNNs) can be adapted to jointly employ the x and y coordinates [8]. However, they need a massive number of data [1].

Most of the existing works have separately performed the optimization process for x and y coordinates. Also, to the best of our knowledge, the previous GPR-based positioning algorithms did not consider a joint optimization/estimation process for x and y coordinates. In this letter, we introduce a novel fingerprint-based GPR positioning algorithm via a *joint* coordinate optimization within the offline phase. The proposed method, named 2D-GPR, maximizes the multivariate probability of x and y coordinates with a shared covariance matrix, which means that it is optimized with respect to RPs' locations. It is crucial for a PRA to have the desired accuracy using limited data samples, considering the time-consumption and labor-intensity of data collection [3]. The proposed 2D-GPR algorithm achieves better accuracy compared with CGPR when a small set of data is available. Besides, 2D-GPR has a lower computation complexity compared with CGPR due to its one-time optimization accomplishment in the training phase and calculation of matrix elements in the test phase.

II. Conventional GPR Based Positioning

Data collection is the initial step for fingerprint-based positioning, in which RSS vectors captured from APs are stored in a database, as depicted in Fig. 1. A PRA is fit to the collected data, and then the trained PRA converts the online RSS vectors of the test points (TPs) to the location coordinates. This section describes the structure of conventional Gaussian process regression (CGPR) as a PRA, which is needed to apprehend the proposed 2D-GPR algorithm. First, we consider a training dataset consists of RPs' fingerprints and corresponding locations as follows

$$\mathbf{S} = [\mathbf{s}_1, \mathbf{s}_2, \cdots, \mathbf{s}_N]^T, \quad \begin{matrix} \mathbf{x} = [x_1, x_2, \cdots, x_N]^T \\ \mathbf{y} = [y_1, y_2, \cdots, y_N]^T, \end{matrix} \tag{1}$$

where $\mathbf{s}_i \in \mathbb{R}^{M\times 1}$ is the fingerprint of the i^{th} RP, M is the number of APs, and N is the number of RPs. In CGPR the aim is to obtain two functions that can map any RSS vector $\forall \mathbf{s}_i \in \mathbb{R}^{M\times 1}$ to the 2D Cartesian coordinates as follows

$$x_i = \omega_x(\mathbf{s}_i) + \epsilon_x, \quad y_i = \omega_y(\mathbf{s}_i) + \epsilon_y \ \text{ and } \ \epsilon_x, \epsilon_y \sim \mathcal{N}(0, \sigma_n^2), \tag{2}$$

M. Nabati and R. Shahbazian are with the Cognitive Telecommunication Research Group, Department of Telecommunications, Faculty of Electrical Engineering, Shahid Beheshti University, Tehran, Iran (e-mail:mo.nabati@mail.sbu.ac.ir; r_shahbazian@sbu.ac.ir).

S. A. Ghorashi is with the School of Architecture, Computing and Engineering, University of East London, E16 2RD London, U.K. and also with the Cognitive Telecommunication Research Group, Department of Telecommunications, Faculty of Electrical Engineering, Shahid Beheshti University, Tehran, Iran (e-mail:s.a.ghorashi@uel.ac.uk).

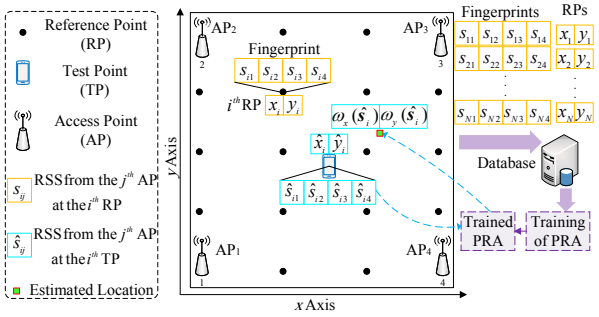


Fig. 1. The schematic of fingerprint-based indoor positioning process.

where ω_x and ω_y are two functions that can convert RSS vectors to the x and y coordinates, respectively. In the training phase, it is assumed that the measurements are noisy and we do not have access to the function values obtained by the noise-free RSS observations, and $\omega_x(\mathbf{s}_i)$ and $\omega_y(\mathbf{s}_i)$ are unknown. However, we have access to the x_i and y_i variables with noisy RSS values, where ϵ_x and ϵ_y are zero-mean Gaussian noise with σ_n^2 variance. The ω_x and ω_y should be optimized by the training dataset in (1). Since the optimization process of ω_x and ω_y are similar, we only describe this procedure for ω_x. In the Gaussian processes perspective [9], all collections of x_i variables have a joint Gaussian distribution. Here, the joint distribution can be interpreted as the effect of noise on the APs signals in radio channels. Initially, we can assume a zero-mean Gaussian process, and therefore, the vector $\mathbf{x}$ has a multivariate probability distribution as follows

$$\mathbf{x} \sim \mathcal{GP}(\mathbf{0}, \mathbf{C}_x), \tag{3}$$

where $\mathbf{C}_x \in \mathbb{R}^{N\times N}$ is the covariance matrix, and each element of this matrix is calculated by a user-defined kernel function. The kernel functions capture the similarity between the two pairs of RSS vectors. Here, we use a combination of three kernels as follows

$$c_{ij}^x = \alpha_1^2 \exp\left(-\frac{(\mathbf{s}_i - \mathbf{s}_j)^T(\mathbf{s}_i - \mathbf{s}_j)}{\gamma^2}\right) + \alpha_2^2 \mathbf{s}_i^T \mathbf{s}_j + \sigma_n^2 \delta_{ij}, \tag{4}$$

where c_{ij}^x represents the i^{th} row and j^{th} column of matrix $\mathbf{C}_x$, $\mathbf{s}_i$ is the i^{th} row of $\mathbf{S}$, and $\delta_{ij} = \{1 \text{ if } i = j,\ 0 \text{ o.w}\}$. In (4), the first and second terms are "squared exponential" and "linear" kernels that elicit non-linear and linear dependencies of $\mathbf{s}_i^{th}$, respectively. Finally, the third term models the variance of ϵ_x in (2). The linear kernel can smooth the GPR model for better fitting to the data. The vector $\boldsymbol{\theta} = [\alpha_1, \alpha_2, \gamma, \sigma_n]^T$ contains the hyperparameters and should be optimized in the training phase to maximize the log-likelihood of multivariate probability density function (PDF)

$$\tilde{\boldsymbol{\theta}} = \arg\max_{\boldsymbol{\theta}} \log(p(\mathbf{x})) = \arg\min_{\boldsymbol{\theta}} (-\log(p(\mathbf{x})), \tag{5}$$

where $p(\mathbf{x})$ is the multivariate PDF defined as follows

$$p(\mathbf{x}) = \frac{1}{(2\pi)^{N/2}|\mathbf{C}_x|^{1/2}} \exp(-\frac{1}{2}\mathbf{x}^T \mathbf{C}_x^{-1}\mathbf{x}). \tag{6}$$

Therefore, the objective function of (5) is as follows

$$\mathcal{L}(\boldsymbol{\theta}) = -\log p(\mathbf{x}) = \frac{1}{2}\log|\mathbf{C}_x| + \frac{N}{2}\log(2\pi) + \frac{1}{2}\mathbf{x}^T\mathbf{C}_x^{-1}\mathbf{x}. \tag{7}$$

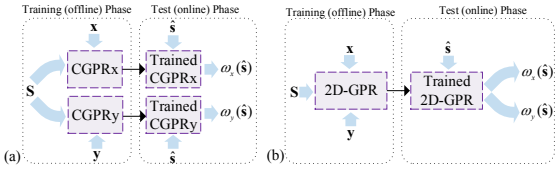


Fig. 2. Difference between (a) Conventional GPR (CGPR) and (b) Proposed 2D-GPR algorithms.

The optimization problem in (5) is non-convex. However, it can be solved for a locally optimum point via gradient-based algorithms such as conjugate gradient [10]. The conjugate gradient algorithm (CGA) needs the first-order gradient of $\mathcal{L}(\boldsymbol{\theta})$ to optimize the hyperparameters and for the j^{th} hyperparameter can be calculated as follows

$$\nabla\mathcal{L}(\theta_j) = -\frac{1}{2}\text{tr}((\mathbf{q}_x\mathbf{q}_x^T - \mathbf{C}_x^{-1})\frac{\partial \mathbf{C}_x}{\partial \theta_j}) \text{ where } \mathbf{q}_x = \mathbf{C}_x^{-1}\mathbf{x}. \tag{8}$$

The CGA is iterated till convergence, and then it is needed to derive the posterior distribution from the joint distribution to estimate the users' locations. Assume that the joint distribution of test and train samples is as follows

$$\begin{bmatrix} \mathbf{x} \\ \hat{\mathbf{x}} \end{bmatrix} \sim \mathcal{N}\left[\begin{pmatrix} \mathbf{0} \\ \mathbf{0} \end{pmatrix}, \begin{pmatrix} \mathbf{C}_x & \mathbf{C}_x(\mathbf{s}, \hat{\mathbf{s}}) \\ \mathbf{C}_x(\hat{\mathbf{s}}, \mathbf{s}) & \mathbf{C}_x(\hat{\mathbf{s}}, \hat{\mathbf{s}}) \end{pmatrix}\right], \tag{9}$$

where $\hat{\mathbf{x}} \in \mathbb{R}^{\hat{N}\times 1}$ is the vector of x coordinates at TPs that should be estimated by conditioning over the x coordinates of RPs, $\hat{N}$ is the number of TPs (users), $\mathbf{C}_x \in \mathbb{R}^{N\times N}$ is the covariance matrix between the x coordinates of RPs, $\mathbf{C}_x^T(\hat{\mathbf{s}}, \mathbf{s}) = \mathbf{C}_x(\mathbf{s}, \hat{\mathbf{s}}) \in \mathbb{R}^{N\times\hat{N}}$ is the covariance matrix between the x coordinates of RPs and TPs, and $\mathbf{C}_x(\hat{\mathbf{s}}, \hat{\mathbf{s}}) \in \mathbb{R}^{\hat{N}\times\hat{N}}$ is the covariance matrix between the x coordinates of TPs. The elements of these matrices are calculated by the optimized hyperparameters, and then the posterior distribution can be derived as follows

$$\begin{aligned} &\hat{\mathbf{x}}|\mathbf{x} \sim \mathcal{N}(\boldsymbol{\omega}_x, \boldsymbol{\Phi}_x) \\ &\boldsymbol{\omega}_x = \mathbf{C}_x(\hat{\mathbf{s}}, \mathbf{s})\,\mathbf{C}_x^{-1}\mathbf{x} \\ &\boldsymbol{\Phi}_x = \mathbf{C}_x(\hat{\mathbf{s}}, \hat{\mathbf{s}}) - \mathbf{C}_x(\hat{\mathbf{s}}, \mathbf{s})\mathbf{C}_x^{-1}\mathbf{C}_x(\mathbf{s}, \hat{\mathbf{s}}), \end{aligned} \tag{10}$$

where $\boldsymbol{\omega}_x$ is the pattern recognition function whose outputs equal to the estimated coordinates, and $\boldsymbol{\Phi}_x$ is the covariance matrix whose diagonal elements are the estimated variances.

III. 2D-GPR Based Positioning

In this section, we present our proposed 2D-GPR algorithm. Fig. 2 shows the difference between the proposed 2D-GPR and CGPR algorithms. Since the radio channels' noise has a similar effect for the pairs of x_i and y_i at the i^{th} RP, a shared covariance matrix can model the effect of noise for the pairs of x_i and y_i, simultaneously. Therefore, we consider a shared covariance matrix for both coordinate vectors ($\mathbf{x}$ and $\mathbf{y}$) as follows

$$(\mathbf{x}, \mathbf{y}) \sim \mathcal{GP}((\mathbf{0}, \mathbf{0}), \mathbf{C}_r), \tag{11}$$

where $\mathbf{C}_r \in \mathbb{R}^{N\times N}$ is the shared covariance matrix for x_i and y_i variables, and each element of this matrix is calculated by a kernel function (e.g., Eq. (4)). Considering a shared covariance matrix for both coordinates means that the similarities of

RSS vectors are captured for the RPs in the optimization process, not for x_i and y_i coordinates separately. The proposed optimization problem employs both coordinates during the training phase as follows

$$\tilde{\boldsymbol{\theta}} = \arg\max_{\boldsymbol{\theta}} \log(p(\mathbf{x},\mathbf{y})) = \arg\min_{\boldsymbol{\theta}}(-\log(p(\mathbf{x},\mathbf{y}))), \quad (12)$$

where $p(\mathbf{x},\mathbf{y}) = p(\mathbf{x}|\mathbf{y})p(\mathbf{y})$. Each observation ($x_i$ or y_i) is conditionally independent of other observations given the corresponding latent variables ($\omega_x(\mathbf{s}_i)$ or $\omega_y(\mathbf{s}_i)$) [9]. The latent variables are affected by the noise in radio channels which is modeled by covariance matrix. The *shared* covariance matrix makes the *joint* distribution for the pairs of x_i and y_i at the same time. Thus, we can assume that the observations are independent $p(\mathbf{x}|\mathbf{y}) = p(\mathbf{x})$, because we have already modeled the dependency of x_i and y_i by a shared covariance matrix $\mathbf{C}_r$ for the latent variables. Besides, considering a shared covariance matrix does not allow us to derive the $p(\mathbf{x}|\mathbf{y})$. Let us derive the $p(\mathbf{x}|\mathbf{y})$ in more detail with the following joint distribution

$$\begin{bmatrix} \mathbf{y} \\ \mathbf{x} \end{bmatrix} \sim \mathcal{N}\left[\begin{pmatrix} \mathbf{0} \\ \mathbf{0} \end{pmatrix}, \quad \boldsymbol{\Sigma}_{x,y} = \begin{pmatrix} \mathbf{C}_y(\mathbf{s},\mathbf{s}) & \mathbf{C}_{x,y}(\mathbf{s},\mathbf{s}) \\ \mathbf{C}_{y,x}(\mathbf{s},\mathbf{s}) & \mathbf{C}_x(\mathbf{s},\mathbf{s}) \end{pmatrix}\right], \quad (13)$$

where the joint covariance matrix $\boldsymbol{\Sigma}_{x,y}$ is a singular matrix and is not invertible, because the input features and hyperparameters are the same for both coordinates. In other words, in (13) we have $\mathbf{C}_x(\mathbf{s},\mathbf{s}) = \mathbf{C}_y(\mathbf{s},\mathbf{s}) = \mathbf{C}_{x,y}(\mathbf{s},\mathbf{s}) = \mathbf{C}_{y,x}(\mathbf{s},\mathbf{s})$. As can be seen, the $p(\mathbf{x}|\mathbf{y})$ cannot be derived. Therefore, the objective function can be calculated as follows

$$\begin{aligned} \mathcal{L}(\boldsymbol{\theta}) &= -\log(p(\mathbf{x},\mathbf{y})) \\ &= \frac{1}{2}\log|\mathbf{C}_r| + \frac{N}{2}\log(2\pi) + \frac{1}{2}\mathbf{x}^T\mathbf{C}_r^{-1}\mathbf{x} + \\ &\frac{1}{2}\log|\mathbf{C}_r| + \frac{N}{2}\log(2\pi) + \frac{1}{2}\mathbf{y}^T\mathbf{C}_r^{-1}\mathbf{y}. \end{aligned} \quad (14)$$

The gradient of (14) for the j^{th} hyperparameter is derived as follows

$$\nabla\mathcal{L}(\theta_j) = \ = -\frac{1}{2}\mathrm{tr}((\mathbf{q}_x\mathbf{q}_x^T + \mathbf{q}_y\mathbf{q}_y^T - 2\mathbf{C}_r^{-1})\frac{\partial\mathbf{C}_r}{\partial\theta_j}), \quad (15)$$

where $\mathbf{q}_x = \mathbf{C}_r^{-1}\mathbf{x}$ and $\mathbf{q}_y = \mathbf{C}_r^{-1}\mathbf{y}$. As explained in the previous section, CGA can be applied to update the hyperparameters till convergence. The joint distribution of TPs and RPs for this 2D scenario can be written as follows

$$\begin{bmatrix} (\mathbf{x},\mathbf{y}) \\ (\hat{\mathbf{x}},\hat{\mathbf{y}}) \end{bmatrix} \sim \mathcal{N}\left[\begin{pmatrix} (\mathbf{0},\mathbf{0}) \\ (\mathbf{0},\mathbf{0}) \end{pmatrix}, \begin{pmatrix} \mathbf{C}_r & \mathbf{C}_r(\mathbf{s},\hat{\mathbf{s}}) \\ \mathbf{C}_r(\hat{\mathbf{s}},\mathbf{s}) & \mathbf{C}_r(\hat{\mathbf{s}},\hat{\mathbf{s}}) \end{pmatrix}\right], \quad (16)$$

where $\hat{\mathbf{x}},\hat{\mathbf{y}} \in \mathbb{R}^{\hat{N}\times 1}$ are locations of TPs, $\mathbf{x},\mathbf{y}$ are locations of RPs, $\hat{N}$ is the number of TPs (users), $\mathbf{C}_r \in \mathbb{R}^{N\times N}$ is the covariance matrix of RPs, $\mathbf{C}_r^T(\hat{\mathbf{s}},\mathbf{s}) = \mathbf{C}_r(\mathbf{s},\hat{\mathbf{s}}) \in \mathbb{R}^{N\times\hat{N}}$ is the covariance matrix of TPs and RPs, and $\mathbf{C}_r(\hat{\mathbf{s}},\hat{\mathbf{s}}) \in \mathbb{R}^{\hat{N}\times\hat{N}}$ is the covariance matrix of TPs.

Proposition: The posterior distribution of 2D-GPR can be derived as follows

$$\begin{aligned} &(\hat{\mathbf{x}},\hat{\mathbf{y}})|(\mathbf{x},\mathbf{y}) \sim \mathcal{N}((\omega_x,\omega_y),\ \boldsymbol{\Phi}_r) \\ &\boldsymbol{\omega}_x = \mathbf{C}_r(\hat{\mathbf{s}},\mathbf{s})\,\mathbf{C}_r^{-1}\mathbf{x} \\ &\boldsymbol{\omega}_y = \mathbf{C}_r(\hat{\mathbf{s}},\mathbf{s})\,\mathbf{C}_r^{-1}\mathbf{y} \\ &\boldsymbol{\Phi}_r = \mathbf{C}_r(\hat{\mathbf{s}},\hat{\mathbf{s}}) - \mathbf{C}_r(\hat{\mathbf{s}},\mathbf{s})\mathbf{C}_r^{-1}\mathbf{C}_r(\mathbf{s},\hat{\mathbf{s}}). \end{aligned} \quad (17)$$

Proof: To derive the posterior distribution in (17), we explain the process for the $\boldsymbol{\omega}_x$ with respect to the shared covariance matrix that can similarly be employed for $\boldsymbol{\omega}_y$. Initially, from the Bayes rule, we have

$$\begin{aligned} p(\hat{\mathbf{x}}|\mathbf{x}) &= \frac{p(\hat{\mathbf{x}},\mathbf{x})}{p(\mathbf{x})} \sim \frac{\exp(-\frac{1}{2}\bar{\mathbf{x}}^T\Sigma_r^{-1}\bar{\mathbf{x}})}{\exp(-\frac{1}{2}\mathbf{x}^T\mathbf{C}_r^{-1}\mathbf{x})} \\ &= \exp(-\frac{1}{2}(\bar{\mathbf{x}}^T\Sigma_r^{-1}\bar{\mathbf{x}} - \mathbf{x}^T\mathbf{C}_r^{-1}\mathbf{x})), \end{aligned} \quad (18)$$

where,

$$\bar{\mathbf{x}} = \begin{bmatrix} \mathbf{x} \\ \hat{\mathbf{x}} \end{bmatrix}, \ \Sigma_r = \begin{bmatrix} \mathbf{C}_r & \mathbf{C}_r(\mathbf{s},\hat{\mathbf{s}}) \\ \mathbf{C}_r(\hat{\mathbf{s}},\mathbf{s}) & \mathbf{C}_r(\hat{\mathbf{s}},\hat{\mathbf{s}}) \end{bmatrix}. \quad (19)$$

In (18), $\boldsymbol{\Sigma}_r^{-1}$ is the inverse of 2×2 block matrix that can be calculated as below

$$\boldsymbol{\Sigma}_r^{-1} = \begin{bmatrix} \mathbf{C}_r^{-1} + \mathbf{C}_r^{-1}\mathbf{C}_r(\mathbf{s},\hat{\mathbf{s}})\mathbf{H}\mathbf{C}_r(\hat{\mathbf{s}},\mathbf{s})\mathbf{C}_r^{-1} & -\mathbf{C}_r^{-1}\mathbf{C}_r(\mathbf{s},\hat{\mathbf{s}})\mathbf{H} \\ -\mathbf{H}\mathbf{C}_r(\hat{\mathbf{s}},\mathbf{s})\mathbf{C}_r^{-1} & \mathbf{H} \end{bmatrix}, \quad (20)$$

where $\mathbf{H} = (\mathbf{C}_r(\hat{\mathbf{s}},\hat{\mathbf{s}}) - \mathbf{C}_r(\hat{\mathbf{s}},\mathbf{s})\mathbf{C}_r^{-1}\mathbf{C}_r(\mathbf{s},\hat{\mathbf{s}}))^{-1}$. The above equation can be substituted to inner term in (18) and can be simplified as follows

$$\begin{aligned} &\bar{\mathbf{x}}^T\boldsymbol{\Sigma}_r^{-1}\bar{\mathbf{x}} - \mathbf{x}^T\mathbf{C}_r^{-1}\mathbf{x} = \begin{bmatrix} \mathbf{x}^T & \hat{\mathbf{x}}^T \end{bmatrix}\boldsymbol{\Sigma}_r^{-1}\begin{bmatrix} \mathbf{x} \\ \hat{\mathbf{x}} \end{bmatrix} - \mathbf{x}^T\mathbf{C}_r^{-1}\mathbf{x} \\ &= (\hat{\mathbf{x}} - \mathbf{C}_r(\hat{\mathbf{s}},\mathbf{s})\mathbf{C}_r^{-1}\mathbf{x})^T\mathbf{H}(\hat{\mathbf{x}} - \mathbf{C}_r(\hat{\mathbf{s}},\mathbf{s})\mathbf{C}_r^{-1}\mathbf{x}). \end{aligned} \quad (21)$$

It can be seen that the posterior mean vector and covariance matrix are as follows

$$\begin{cases} \boldsymbol{\omega}_x = \mathbf{C}_r(\hat{\mathbf{s}},\mathbf{s})\mathbf{C}_r^{-1}\mathbf{x} \\ \boldsymbol{\Phi}_r = \mathbf{H}^{-1} = \mathbf{C}_r(\hat{\mathbf{s}},\hat{\mathbf{s}}) - \mathbf{C}_r(\hat{\mathbf{s}},\mathbf{s})\mathbf{C}_r^{-1}\mathbf{C}_r(\mathbf{s},\hat{\mathbf{s}}). \end{cases} \quad (22)$$

Note that $\boldsymbol{\Phi}_r$ is equal for both coordinates because it does not depend on x and y.

Complexity Analysis: In the training phase, 2D-GPR needs to calculate the $\mathbf{C}_r^{-1}$ in (15) with the complexity order of $O(N^3)$, which is the highest order complexity among the other operations. This process is done twice to calculate the $\mathbf{C}_x^{-1}$ and $\mathbf{C}_y^{-1}$ for CGPR. In the test phase, 2D-GPR needs to calculate the $\mathbf{C}_r^{-1}$ in (17) which is equal for $\boldsymbol{\omega}_x$ and $\boldsymbol{\omega}_y$, whereas CGPR performs this process twice to calculate the $\mathbf{C}_x^{-1}$ and $\mathbf{C}_y^{-1}$. Therefore, 2D-GPR has lower complexity compared with that of CGPR in both training and test phases. Although there are other operations in the training and test phases, they have lower order of complexities.

IV. Experimental Results

The proposed method has been implemented using R programming language, and 75 RSS samples from 27 Wi-Fi APs have been recorded at each of 250 points which are depicted in Fig. 3. The Monte-Carlo cross-validation method [11] is used to reduce the error bias for evaluations, in which the points in Fig. 3 are used either as RPs or TPs. These points are randomly selected as RPs and TPs for K times, and the mean average error (MAE) over these K times for prediction of TPs' locations is calculated as follows

$$\mathrm{MAE} = \frac{1}{K\hat{N}}\sum_{k=1}^{K}\sum_{i=1}^{\hat{N}}\sqrt{(\hat{\mathbf{x}}_i^k - [\boldsymbol{\omega}_x(\hat{\mathbf{s}}_i)]^k)^2 + (\hat{\mathbf{y}}_i^k - [\boldsymbol{\omega}_y(\hat{\mathbf{s}}_i)]^k)^2} \quad (23)$$

TABLE I
THE MAE (m) OF DIFFERENT ALGORITHMS VIA DIFFERENT EXPERIMENT SETUPS FOR DIFFERENT NUMBER OF RPS AND APS.

RP (N) ⇒	40				80				150			
AP (M) ⇒	3	5	9	27	3	5	9	27	3	5	9	27
2D-GPR	3.28	2.93	2.91	2.62	2.78	2.61	2.48	2.12	2.67	2.40	2.21	1.94
CGPR [4]	3.76	3.34	3.07	2.78	2.97	2.64	2.63	2.19	2.77	2.42	2.25	1.97
CRF [6]	4.35	3.84	3.35	3.31	3.67	3.26	2.89	2.86	3.29	2.90	2.49	2.43
DNNR [8]	3.70	3.43	3.02	3.90	3.66	3.30	2.81	3.36	3.28	2.98	2.65	2.90

where $\hat{N}$ is the number of TPs, $K = 10$ is the number of iterations, $\hat{\mathbf{x}}_i^k$ and $\hat{\mathbf{y}}_i^k$ are the coordinates of the i^{th} TP within the k^{th} iteration, and $[\omega_x(\hat{\mathbf{s}}_i)]^k$ and $[\omega_y(\hat{\mathbf{s}}_i)]^k$ are the estimated coordinates of the i^{th} TP within the k^{th} iteration. We average out all 75 RSS samples for the RPs obtained from multiple times to reduce the small-scale fading effect [4]. However, in the online phase, the users cannot get high samples, and we assume that each user has 3 samples. Therefore, the number of TPs ($\hat{N}$) would be $(250 - N) \times (75/3)$.

We have conducted 12 experiments and compared the proposed 2D-GPR algorithm with three PRAs including, CGPR [4], CRF [6], and deep neural network regression (DNNR) [8]. The MAE of different experiment setups are reported in Table I. During the data collection phase, signals of all detected APs are recorded. Most of these APs belong to other floors or buildings, and we perform a preprocessing to select the APs according to their reliability in the environment. If we define w to be the number of reliable samples of an AP (RSS $>$ -70 dBm) on all 250 points, and W to be the number of all samples in the environment (75×250), the APs with $\frac{w}{W} > \rho$ are selected. The value of $\frac{w}{W}$ for an AP is between 0 and 1, and the high ρ selects more reliable APs. $\rho = 0.6$, 0.5 and 0.4 correspond to the selection of 3, 5 and 9 APs, respectively.

As can be seen in Table I, when the number of RPs and APs are limited ($N = 40$ and $M \leq 5$), 2D-GPR is more effective than other PRAs such that the improvement is more than $40cm$ compared with CGPR. Also, in ideal scenarios where the number of RPs and APs are enough (e.g., $N = 150$ and $M = 27$), 2D-GPR performs a little better than CGPR and outperforms the CRF and DNNR considerably. Considering the fact that PRAs will be saturated by increasing the number of data [1], [12], we can see that the performance of 2D-GPR algorithm is much better than its counterparts in practice. Also note that adding poor APs may have a destructive effect on some algorithms, for instance, the accuracy of DNNR will diminish when all 27 APs are used.

In Fig. 4(a) and Fig. 4(b), we have illustrated the boxplot of $K = 10$ iterations for a low RPs and APs density scenario (e.g., $N = 40$, $M = 5$), and a high RPs and APs density scenario (e.g., $N = 150$, $M = 27$), respectively, in which the horizontal lines show the median error. As can be seen, 2D-GPR has less median error and variance in the low-density scenario. We can conclude that our proposed method is more effective, especially for large-scale areas with low densities of RPs and APs. Besides, the proposed method is preferable due to its lower calculation cost compared with CGPR.

V. CONCLUSION

We proposed a novel fingerprint-based positioning method, named 2D-GPR, which is an evolved version of CGPR.

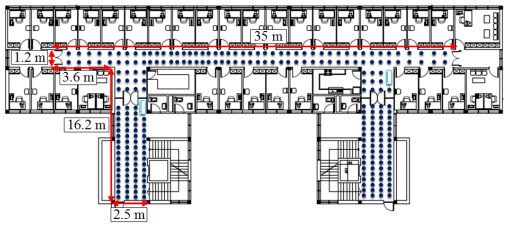
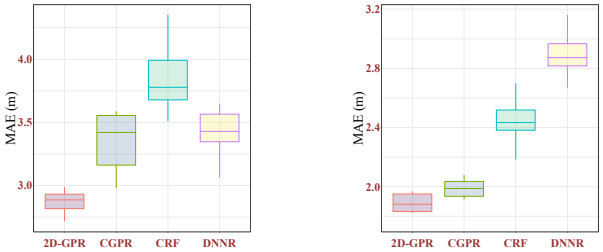


Fig. 3. Floor plan of the indoor environment, where RSS samples have been collected at 250 points (indicated by dots). These points are randomly selected as RPs and TPs.



(a) ($N = 40, M = 5$)

(b) ($N = 150, M = 27$)

Fig. 4. Boxplot for two scenarios of Table I where (a) is a low RPs and APs density scenario and (b) is a high RPs and APs density scenario.

We conducted several experiments with different experiment setups, and the results proved the superiority of this algorithm over baseline PRAs, especially when the densities of RPs and APs are limited. The proposed method also has less complexity compared with CGPR. For future work, it can be extended to a 3D positioning algorithm.

REFERENCES

[1] M. Nabati, H. Navidan, R. Shahbazian, S. A. Ghorashi, and D. Windridge, "Using synthetic data to enhance the accuracy of fingerprint-based localization: A deep learning approach," *IEEE Sensors Letters*, vol. 4, no. 4, pp. 1–4, 2020.

[2] Y. Tao and L. Zhao, "Fingerprint localization with adaptive area search," *IEEE Communications Letters*, vol. 24, no. 7, pp. 1446–1450, 2020.

[3] A. Khalajmehrabadi, N. Gatsis, and D. Akopian, "Modern WLAN fingerprinting indoor positioning methods and deployment challenges," *IEEE Communications Surveys & Tutorials*, vol. 19, no. 3, pp. 1974–2002, 2017.

[4] K. N. R. S. V. Prasad, E. Hossain, and V. K. Bhargava, "Machine learning methods for RSS-based user positioning in distributed massive MIMO," *IEEE Transactions on Wireless Communications*, vol. 17, no. 12, pp. 8402–8417, 2018.

[5] W. Kim, J. Park, J. Yoo, H. J. Kim, and C. G. Park, "Target localization using ensemble support vector regression in wireless sensor networks," *IEEE Transactions on Cybernetics*, vol. 43, no. 4, pp. 1189–1198, 2013.

[6] X. Guo, N. Ansari, L. Li, and H. Li, "Indoor localization by fusing a group of fingerprints based on random forests," *IEEE Internet of Things Journal*, vol. 5, no. 6, pp. 4686–4698, 2018.

[7] H. Liu, J. Cai, and Y.-S. Ong, "Remarks on multi-output Gaussian process regression," *Knowledge-Based Systems*, vol. 144, pp. 102–121, 2018.

[8] R. Zhou, M. Hao, X. Lu, M. Tang, and Y. Fu, "Device-free localization based on CSI fingerprints and deep neural networks," in *IEEE International Conference on SECON*, 2018, pp. 1–9.

[9] C. K. Williams and C. E. Rasmussen, *Gaussian processes for machine learning*. MIT press Cambridge, MA, 2006.

[10] J. Nocedal and S. Wright, *Numerical optimization*. Springer Science & Business Media, 2006.

[11] Q.-S. Xu and Y.-Z. Liang, "Monte Carlo cross validation," *Chemometrics and Intelligent Laboratory Systems*, vol. 56, no. 1, pp. 1–11, 2001.

[12] X. Zhu, C. Vondrick, C. C. Fowlkes, and D. Ramanan, "Do we need more training data?" *International Journal of Computer Vision*, vol. 119, no. 1, pp. 76–92, 2016.