Robotics and Autonomous Systems 103 (2018) 134–150

Contents lists available at ScienceDirect

Robotics and Autonomous Systems

journal homepage: www.elsevier.com/locate/robot

# Precise and accurate wireless signal strength mappings using Gaussian processes and path loss models

Renato Miyagusuku *, Atsushi Yamashita, Hajime Asama

*Department of Precision Engineering, Graduate School of Engineering, The University of Tokyo, 113-8656 Tokyo, Japan*

## HIGHLIGHTS

- Novel probabilistic wireless signal strength mappings for robot localization.
- Mappings generated by combining physics-based and data-driven generative models.
- Main improvements result from using physics-based models to reduce prediction uncertainties.
- All datasets and software available online for the community.

## ARTICLE INFO

*Article history:*
Received 8 June 2017
Received in revised form 19 January 2018
Accepted 19 February 2018
Available online 27 February 2018

*Keywords:*
Robot localization
Wireless sensor model
Signal strength mapping
Gaussian processes

## ABSTRACT

In this work, we present a new modeling approach that generates precise (low variance) and accurate (low mean error) wireless signal strength mappings. In robot localization, these mappings are used to compute the likelihood of locations conditioned to new sensor measurements. Therefore, both mean and variance predictions are required. Gaussian processes have been successfully used for learning highly accurate mappings. However, they generalize poorly at locations far from their training inputs, making those predictions have high variance (low precision). In this work, we address this issue by incorporating path loss models, which are parametric functions that although lacking in accuracy, generalize well. Path loss models are used together with Gaussian processes to compute mean predictions and most importantly, to bound Gaussian processes' predicted variances. Through extensive testing done with our open source framework, we demonstrate the ability of our approach to generating precise and accurate mappings, and the increased localization accuracy of Monte Carlo localization algorithms when using them; with all our datasets and software been made readily available online for the community.



## 1. Introduction

The usage of wireless signals for robot localization in indoor, GPS-denied, locations has gained popularity in recent years, partly due to the almost ubiquitous presence of wireless local area networks (WLANs) in most buildings. Although wireless signals-based localization systems do not achieve as high accuracy as those based on sensors such as laser rangefinders or RGB-D cameras, they possess certain characteristics that make their usage appealing, which are listed below:

**Signals' uniqueness** IEEE802.11 compliant packets transmit its source's own unique identifier (i.e., the access point's macaddress) as part of its header. By extracting this identifier, its source can be identified unequivocally. This makes wireless signal-based systems never suffer from data association problems, which refers to those that arise from incorrectly identifying two or more similar features or landmarks (e.g., failing to distinguish between two similar doors when using a camera, or two similar rooms using laser scans). This makes wireless signals-based systems inherently robust.

**Lower computational requirements** The amount of data that needs to be processed for wireless signals-based localization is significantly lower than that produced by more accurate sensors like RGB-D cameras or laser rangefinders, which implies lower computational requirements.

**Readily available hardware** WLANs being almost ubiquitously present in most buildings means no hardware infrastructure is required. Furthermore, most robots already possess wireless capabilities, in which case the required additional hardware would be none. Even if the robot does not possess wireless capabilities, a wireless network interface controller (WNIC) can be added for a

* Corresponding author.
*E-mail addresses:* miyagusuku@robot.t.u-tokyo.ac.jp (R. Miyagusuku), yamashita@robot.t.u-tokyo.ac.jp (A. Yamashita), asama@robot.t.u-tokyo.ac.jp (H. Asama).

https://doi.org/10.1016/j.robot.2018.02.011
0921-8890/

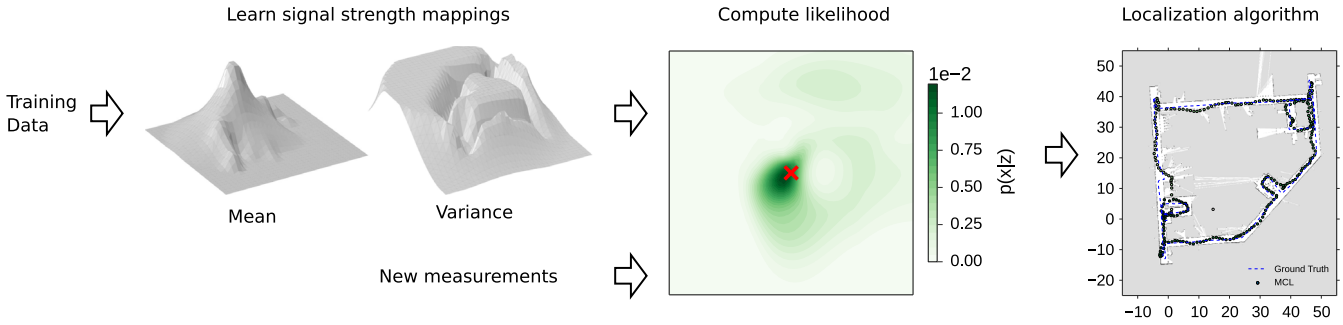



**Fig. 1.** Overview of wireless-based localization using signal strength mappings. For robot localization, given training data, this approach learns location-signal strength mappings. Using the mean and variance of signal strength predictions generated with these mappings, it is possible to compute the likelihood of locations conditioned to new sensor measurements. These likelihoods can then be used as the perceptual model of a wide variety of localization algorithms, such as the Monte Carlo Localization algorithm. In this work we propose a new modeling approach for generating precise and accurate mappings.

really low price. Thus making the additional hardware cost almost nonexistent.

Therefore, wireless-based localization is inherently robust, computationally fast, and does not require any type of hardware deployment or modifications on the environment, making it ideal to use with other sensors, or by low-cost robots. Its main drawback, which limits these potential applications, is its accuracy. To increase this localization accuracy, precise and accurate signal strength mappings are needed. In wireless-based localization, these mappings are used to generate signal strength predictions, necessary for computing the likelihood of locations given acquired measurements. Fig. 1 shows an overview of wireless-based localization and illustrates the essential role these mappings play. Therefore, by improving these mappings, better localization accuracies can be obtained. It is important to notice, that in a Bayesian approach, for computing the likelihood of locations, not only predicted signal strength mean values are required, but also their predicted variances.

When propagating through space, signals interact with all objects in the environment, either being reflected, scattered or diffracted. Therefore, the challenges for generating signal strength mappings arise due to the difficulty of modeling these phenomena, as well as the noisy nature of the signals themselves.

Signal modeling considers only the core behavior of signal propagation and learns it using a parametric function — considering the more complex ones as model noise. These models generalize well (as they are based on the physical phenomenon itself), but lack accuracy — even when learned directly from training data of the environment. On the other hand, signal mapping relinquishes the idea of directly modeling this complex phenomenon, and instead surveys the environment in advance, acquiring training datasets. These datasets are then used explicitly when making predictions; therefore it can be considered as a data-driven approach. Signal mapping generates models that are considerably more accurate than those generated by signal modeling, resulting in most previous research focusing on them. Unfortunately, as data-driven approaches, these models generalize poorly for locations far from their training inputs, causing prediction at such locations to have high variance and yielding imprecise mappings.

The main purpose of this work is the generation of precise and accurate signal strength mappings. For this, we propose a modeling approach which employs both a signal mapping technique, Gaussian processes (GPs), as well as a signal modeling one, path loss models. GPs are used to generate accurate mappings, while path loss models are used to improve their generalization beyond training data. Interestingly, when using both models under our approach, the main improvement is not the generation of more accurate mappings, in terms of lower root mean square errors with respect to training data, but of more precise ones, in terms of lower predicted variances. These more precise mappings, when used by the perceptual model of a Monte Carlo Localization (MCL) algorithm, lead to higher localization accuracy. Effectively improving the state of the art wireless-based localization systems.

This paper is organized as follows. In Section 2 we discuss related work in wireless based localization, and formally define our problem in Section 3. In Section 4 we present our proposed WiFi model, mainly composed of a path loss and a GP component. In Section 5 we describe our experiments and analyze the posteriors generated by our approach — also comparing it to WiFi models generated using only GPs. This analysis and comparisons are extended in Section 6 by using these model as the perceptual model of an MCL algorithm, allowing us to quantify the accuracy enhancements obtained when using our model. This work is concluded in Section 7, where potential future work is also presented.

## 2. Related work

Previous work using wireless signal strength for robot localization can be classified based on geometric and fingerprinting techniques. Geometric techniques calculate the robot's location using simple geometry and "anchors" (access points whose locations are known). Given at least three anchors, if the angle or distance to these anchors can be inferred from new measurements, by means of simple geometry it becomes easy to calculate the location of the robot — several techniques are presented in [1]. Angle information can be reliably extracted from wireless signals but requires specialized hardware, such as rotatory antennas [2], antenna arrays [3] or access points implementing orthogonal frequency division multiplexing (OFDM) [4] - although OFDM networks are commercially available, they are still not ubiquitous. As we do not wish to impose hardware constraints on our system, we focus on techniques employing distance instead of angle information. Distance can be estimated using either propagation time or signal strength. Although propagation time is a good metric to estimate distance, it requires precise synchronization between the network nodes, and is highly sensitive to communication latencies [5], posing heavy assumptions regarding network synchronization. Most techniques using signal strength model them as a function of the distance between anchors and receivers, with some work as the one presented in [6] also considering their relative angle. These models are often parametric functions, with parameters chosen empirically or learned from data directly [7]. Once these parameters have been obtained, predictions can be made for any location in the environment. Furthermore, signal strength is easy to acquire, requires no additional hardware and makes no assumptions regarding the wireless networks. However, due to the difficulty to model signal propagation, it is a poor estimator of distance [8], with applications solely relying on it estimating locations with errors between 3 to 9 meters on average [9].

Fingerprinting, also called scene analysis or profiling, refers to techniques which first obtain samples of the measurements in known locations in the environment (training points), to then use this dataset to predict the location of new measurements. Compared to geometric approaches, fingerprinting techniques have demonstrated higher accuracies in practice [10] and do not require anchors. However, they need extensive training datasets, whose acquisition are more time consuming and laborious than obtaining the location of the network's nodes. Nonetheless, because of their higher accuracy, previous work has almost exclusively centered on them [11–13]. For localization, fingerprinting either matches new measurements to the most similar samples in its training dataset, becoming a classification problem, or learns wireless signal strength mappings, making it a regression problem (these mappings are then used to generate predictions which are used to compute the likelihood of new measurements to have been originated at any candidate location).

Work with classification techniques include the usage of k-Nearest Neighbors [14], support vector machines [15], random forests [16], among others. When more than one location is likely to be the true one, some of these approaches deal with it by tracking these locations over some distance to find the true location [17], or exploit it by transforming the locations' classification scores into a perceptual likelihood [16] and use it in an MCL based approach. The main disadvantage of these approaches is that localization accuracy significantly drops when the training dataset is not dense - i.e., the distance between consecutive points in the training dataset is large.

Work with regression techniques include linear interpolation in graphs [18], smoothing functions [17], GPs [19,20], among others. Among these techniques, GPs have the advantage of not only being able to model the complex behaviors of signal strengths but also directly calculating the prediction variances. Both necessary to construct perceptual likelihoods that can be employed by MCL-based algorithms, such as the MCL we use for testing. An additional advantage is the extensive work already done to extend WiFi GPs to handle heteroscedastic noise [21], multi-floor environments [22], and to address a self-localization and mapping (SLAM) problem (i.e., when the training set does not have labeled locations) [23]. Although to a lesser degree, these techniques' performances also degrade when making predictions far from their training dataset inputs.

Our aim is to develop an approach for generating wireless signal strength mappings that

1. do not place hardware nor network limitation,
2. generate accurate mappings,
3. generalize well outside training data.

In order to comply with (1), we do not consider approaches that require specialized hardware or synchronized networks, leaving out geometric techniques not using signal strength. For achieving (2) we focus our attention on fingerprinting techniques. (3) is the most difficult one, as no previous work has demonstrated both good generalization and high accuracy. For achieving this, we propose the usage of path loss models, a type of signal modeling technique, together with Gaussian processes, a type of regression technique used in fingerprinting. By combining both approaches' strengths while overcoming their limitations, mappings that are both precise and accurate can be obtained.

In our previous work [24], we proposed the idea of including path loss models as informative priors over Gaussian processes. In this work, we extend this idea and propose a different optimization method for path loss models, which yields more consistent and accurate results, and allows for real-time implementations of our system. Furthermore, we delve more into the localization algorithm required for taking advantage of the mappings we obtain and perform extensive experiments using data collected in several indoor environments in order to validate it and compare its localization accuracy and robustness to state of the art wireless-based robot localization systems. It is important to note that, although results in both works are similar, all training datasets in this work have been acquired while the robot is actively exploring the environment, without stopping, and for all testing real-time constraints were applied to the system. Additionally, all datasets acquired and software developed in this work have been made readily available online for the community.[1]

## 3. Problem definition

This work addresses indoor robot localization. Regarding the indoor environment, the existence of several WLANs is assumed. No assumption regarding the WLAN's access points spatial distribution is made. There is no need for the access points to be homogeneously distributed, and there is no prior knowledge of their locations. However, they are assumed to remain in the same location at all times.

Regarding the robot to be localized. It is assumed that it has wireless capabilities, specifically an 802.11-compliant WNIC, and that its motion is planar. Localization is considered to be performed using only odometry and measurements from the robot's WNIC. Odometry is assumed to be readily available from the robot's sensors such as encoders. Wireless signal strengths are assumed to be sensed using its WNIC's built-in Received Signal Strength (RSS) indicator. The heading direction of the robot is assumed not to affect measurements, which is equivalent to assuming that both the robot and the access points have antennas with fairly homogeneous radiation patterns - e.g., omnidirectional antennas.

It is also assumed that a training dataset composed of data pairs of RSS measurements means and the locations where they were taken is available. Formally, given an arbitrary number $m$ of access points. The training dataset is $(\mathbf{X}, \mathbf{Z})$, with $\mathbf{X} \in \mathbb{R}^{n\times 2}$ being the matrix of $n$ training input samples $\mathbf{x}_i$ that correspond to the $x - y$ Cartesian coordinates where the samples were taken and $\mathbf{Z} \in \mathbb{R}^{n\times m}$, the matrix created from the sampled mean of RSS measurements taken from each of the $m$ access points at each of the $n$ different positions. Whenever no RSS measurements from an access point $j$ are sensed at a certain position $i$, the value of $z_{i,j}$ is set to the WNIC's lower sensing limit — thorough discussion on this particular issue is addressed in the next section.

## 4. WGPPL: WiFi fingerprinting using path loss models as informative priors of Gaussian processes

The physical phenomenon we are trying to model is that of electromagnetic wave propagation through space. While propagating, electromagnetic waves can be reflected, scattered and diffracted by walls, furniture, and moving objects. This phenomenon can be accurately described by Maxwell's equations. However, these are rarely used in practice because it requires knowledge of all objects, and their electromagnetic characteristics, in the environment. Instead, it is more suitable to decompose wireless signals propagation phenomena into a path loss, shadowing, multipath, and fast fading components. The path loss component models the dissipation of the signal's power radiated by the transmitter when traveling through open space — hence it is a function of the distance between the transmitter and the receiver. The shadowing component models the result of power absorption by obstacles and will persist as long as the obstacle remains — we consider this effects to be constant in time and specific to each location on the map. The multipath component models the effects caused by

[1] http://www.robot.t.u-tokyo.ac.jp/~miyagusuku/software.

signals reaching the receiver by several paths, and like the shadowing component, it is location dependent (in our work, we consider shadowing and multipath together). Finally, the fast fading component models the effect related to signal strength variations in time even at the same location.

It is important to notice that in our approach we do not wish to model wireless signal strength propagation phenomena, but how WNICs sense it. Received signal strength (RSS) is measured by WNICs in dBm (decibel-milliwatt), which is ten times the logarithm of the measured power in milliwatts, i.e., Power[dBm] = $10\log_{10}$(Power[mW])). Most WNICs are able to sense signal strengths only in the range of −10 to −90 dBm (0.1 to 1e-9 mW). If a measurement is higher than −10 dBm, it is assigned this value regardless, and if it is lower than −90 dBm, it is not registered at all. Although it varies per country, for the 2.4 GHz bands, the maximum transmission power is commonly 20 dBm (100 mW). Considering signal strength propagation through free space, WNICs can theoretically correctly sense signals as close as 0.31 m and as far as 31 m from their source. However, in practice, this range can drastically change, due to occlusions, propagation medium, etc.

Such sensing limitations are, of course, not an exclusive problem of WNICs. For example, most commercial laser rangefinders can only sense within 30 m, and cameras have a predetermined resolution and field of view. For wireless signal strength modeling, the upper limit (−10 dBm) can be ignored, as it is rarely exceeded. Out of all our testing, the maximum signal strength we sensed was −27 dBm, although we drove very near to several access points located in hallways. On the contrary, the lower sensing limit is quite problematic and should be considered. The lower limit causes most of the surveyed locations to have no measurements for any specific access point. If these locations are ignored, information would be lost, as not sensing signals in a particular location does encode information about the propagation phenomenon. This would also cause training datasets to have few training points, making data-driven approaches have large variances for most locations, which is highly undesirable. Instead, we prefer to assign an arbitrarily low signal strength (specifically the WNIC's low sensing limit) to these locations and add them to the dataset. However, in this case, it is important to remember that not sensing a signal can also be caused by large dynamic obstacles occluding them (which should not be considered when learning location-signal strength mappings), WNICs dropping measurements due to checksum errors, sensor noise, etc. This makes these data points less reliable, hence, special care must be taken when learning using them.

For convenience, from this point onwards we will scale and add an offset to measured signal strengths so an RSS value of 1 is equal to −10 dBm and an RSS value of 0 is 90 dBm, which we also consider to be the lower sensing limit. Making RSS measurements always non-negative and in the range from 0 to 1.

An important consideration of our work is our differentiation between the models aiming to represent signal strength propagation through space and those aiming to represent RSS as observed by WNICs. We name the former "propagation models" and the latter "sensor models".

*(1) Propagation models:.* Represent the signal strength propagation phenomenon without any limitations imposed by sensing hardware. Although these models cannot be properly observed due to WNICs sensing limitations, simplified models can be learned based on physics. Specifically, our propagation model only explicitly considers signal's path loss component, leaving shadowing, multipath and fast fading components as system noise. The path loss component is modeled using a parametric function whose parameters are learned from data. A detailed explanation is given in Section 4.1. Although our model provides good generalization and can make predictions of arbitrarily low signal strengths, it has low accuracy due to the shadowing and multipath effects it ignores.

*(2) Sensor models:.* These models represent RSS as sensed by WNICs, including the sensing limitations previously mentioned. Our sensor model considers path loss, shadowing and multipath components, leaving fast fading as system noise. For the path loss component, we use the same parametric function learned for the propagation model, while for both shadowing and multipath components we use a GP. This GP learns the details that the simple parametric function ignores, increasing the model's accuracy. A detailed description of GPs and how we use them in the context of WiFi modeling is given in Section 4.2. Two additional considerations are made for our RSS sensor model. First, to account for WNICs sensing characteristics, predictions are bounded by WNICs' sensing limits and variances are modified. Second, a point mass distribution is added at the lower sensing limit. This point mass distribution is used to compensate for unmodeled issues which could lower signal strength (such as large dynamic obstacles occluding signals), as well as WNICs' hardware failures. The value of this point mass distribution is obtained by comparing the predictions generated from the models and the training data points. Section 4.3 describes these considerations in more detail.

### 4.1. Path loss component

Path loss models aim to capture the essence of signal propagation without resorting to complex models or other geometric considerations, directly estimating signal strength as a function of the location where the signals are sensed ($\mathbf{x}$) and their source — in our case the access point's location ($(x_{ap}, y_{ap})$). For our implementation we use the simplified loss model described in [25]:

$$PL(\mathbf{x}) = a - b\log_{10}(d) + \epsilon_{pl}, \tag{1}$$

where $PL(\mathbf{x})$ is the predicted signal strength, as computed by the model, at location $\mathbf{x}$. $d$ is the Euclidean distance between the location $\mathbf{x}$ and the estimated position of the access point $(x_{ap}, y_{ap})$, $a$ and $b$ being positive constants and $\epsilon_{pl}$ a Gaussian noise with variance $\sigma_{pl}^2$.

In our approach, a path loss model is learned independently for each access point in the environment. For each access point $j$, the set of parameters $(x_{ap}, y_{ap}, a, b)_j$ that characterize its model are learned directly from training data corresponding to its observed signal strengths $(\mathbf{X}, \mathbf{z}_j)$. In order to do so, we define an error function and minimize it. The most common approach is to use the quadratic error function and optimize it using least squares. However, because the training dataset $(\mathbf{X}, \mathbf{z}_j)$ corresponds to signals as sensed by a WNIC, signal strengths below its lower sensing limit are set to zero. The quadratic error function would compute incorrect errors for models predicting negative RSS values when training inputs are zero. To account for this, for training data points equal to zero, we multiplied the squared error by a sigmoid function over the model's prediction. This sigmoid function is a differentiable function which outputs zero for negative values and one for positive ones, making the error function describe the desired behavior for our problem and allowing explicit calculation of the error function derivatives for faster computations. As described, we define our error function as:

$$Error = \sum_{i \in \{1:n\}} e_i^2, \tag{2}$$

where

$$e_i^2 = \begin{cases} (PL_j(\mathbf{x}_i) - z_{i,j})^2 & z_{i,j} > 0, \\ \text{sig}(PL_j(\mathbf{x}_i))(PL_j(\mathbf{x}_i))^2 & z_{i,j} = 0, \end{cases} \tag{3}$$

with sig$(\cdot)$ being a sigmoid function added so negative predictions generated by the path loss model are not penalized when the

sampled sensor readings are zero. For our implementation, the sigmoid function selected was

$$\text{sig}(z) = \frac{1}{1 + e^{-k_s z}}, \tag{4}$$

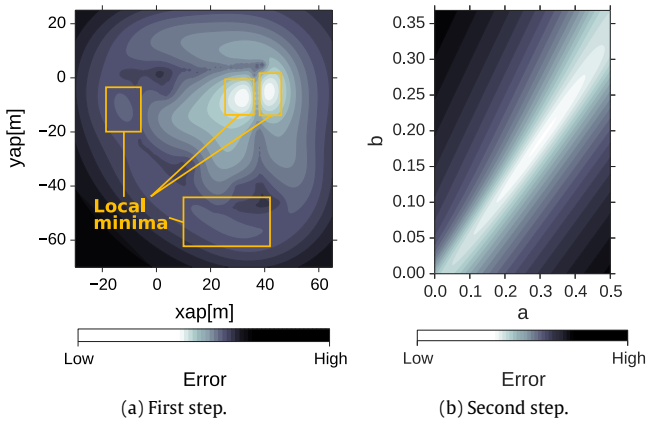
with $k_s$ set to 50 - $k_s$ being a constant that determines the sharpness of the sigmoid around zero, higher values generate sharper functions. This sigmoid function was chosen as its derivative can be computed efficiently ($\text{sig}'(z) = k_s \text{sig}(z)(1 - \text{sig}(z))$), but any other sigmoid function is equally viable.

For minimizing this error function we propose a two-step optimization, where the parameters set ($x_{ap,j}$, $y_{ap,j}$) and the set ($a_j$, $b_j$) are optimized alternately until convergence. We use this approach as optimizing all the parameters simultaneously causes oscillations, demanding longer computation times.

Initial values for all parameters need to be set at the beginning of the optimization procedure. From Eq. (1) it can be observed that $a_j$ determines the value of RSS measurements at the source. The maximum value $a_j$ can take is 1.25 (20 dBm) as this is the maximum allowed transmission power for an access point, while its minimum value is equal to the maximum observed RSS measurement for access point $j$ in our training dataset. Although any value in this range is suitable, in practice we have observed better performance when choosing $a_j$ to be the maximum observed RSS scaled by a factor between 1.5 and 2, as long as the resulting value is lower than 1.25, if higher 1.25 should be assigned. Parameter $b_j$ controls the decay of RSS with respect to $d$. The ratio of $a_j$ to $b_j$ can be used to compute the furthest distance at which RSS measurements from access point $j$ can still be sensed, $max\_d_j = 10^{a_j/b_j}$. We found experimentally that $b_j$ values between $0.75a_j$ and $a_j$ provided suitable initializations, with the former yielding a maximum range of 21 m and the latter 10 m. Either of these ratios can be chosen, with us choosing $b = 0.75a$. It is important to note that while larger ratios are still viable, smaller ratios, like 0.5, should be avoided, as they tend to destabilize the optimization, resulting in optimized access point locations very far from all training data points. Fig. 2(a) shows an example of the error function when performing the optimization of the first step.

As it can be observed the error function has several local minima, hence initialization is important. Given the training dataset, the location corresponding to the highest measured signal strength is a suitable candidate for initializing ($x_{ap,j}$, $y_{ap,j}$) as signal strengths are stronger the closer they are to its source. However, in order to avoid local minima, it is recommended to use more than one initial location.

Considering the training dataset's trajectory to describe a polygon, local minima can exist both inside and outside it. In order to find the global minimum at least two different initial locations should be used, one that favors locations inside the polygon, and another that favors those outside. A straightforward approach to generating these two initial locations is to use the location corresponding to the highest measured signal strength, $\mathbf{x}_{max}$, and the average of the trajectory's locations, $\mathbf{x}_{mean}$. As $\mathbf{x}_{max}$ is necessarily located in either one of the polygon's sides or vertices and $\mathbf{x}_{mean}$ is located inside the polygon (for all convex polygons and most non-convex ones), the initial points: $2\mathbf{x}_{max} - \mathbf{x}_{mean}$ and $\mathbf{x}_{mean}$ are valid choices. By replacing $\mathbf{x}_{mean}$, by a weighted average, $\mathbf{x}_{wmean}$, that uses RSS measurements as weights, convergence speeds were slightly improved as initial locations were closer to the final estimated access points locations. While it is possible to define multiple other valid strategies and increment the number of initial candidates. Even when assigning multiple initial locations randomly distributed around $\mathbf{x}_{max}$, access points locations kept converging to the same locations as when only using these two. Therefore, our approach only uses these two locations, as to avoid unnecessary additional computations.




(a) First step. (b) Second step.

**Fig. 2.** Example of the outputs of Eq. (2) for fitting path loss model parameters using the proposed two-step optimization approach. The first step optimizes parameters $x_{ap}$ and $y_{ap}$, while the second optimizes parameters $a$, $b$. Initialization is specially important for the first step — shown in (a), as the error function has several local minima.

The second step of the optimization considers ($x_{ap,j}$, $y_{ap,j}$) fixed. Given this assumption, Eq. (1) can be reformulated as $PL = a - bL$, with $L$ being $\log_{10}(|\mathbf{x} - xap|)$, thus becoming a linear model with respect to $L$. The linear least squares solution for this model has a closed-form solution and, if its Hessian matrix $2[1\ L]^T[1\ L]$ is positive definite, also a global minima, as its error function would be strictly convex. Our error function can be understood as the summation of two functions: (i) the least squares error function over the non-zero set of data points $z_{i,j} > 0$, and (ii) a modified least squares error function over the remaining data points $z_{i,j} = 0$. Regarding the convexity of our error function, if the Hessian of our non-zero set is positive definite, (i) is strictly convex. For (ii), if it is assumed that the sigmoid in Eq. (3) is extremely sharp, the error function becomes the least squares error function for all parameter combinations $(a, b)$ at or above the line $a = bL$, and zero for those below it. Being equal to the least squares error function, ours is convex for the domain ($a \geq bL$), with its minimum values being equal to zero for all $a = bL$. As this minimum value is the same for the remaining of the function's domain, any segment between $((a_0, b_0), e_i(a_0, b_0))$ and $((a_1, b_1), e_i(a_1, b_1))$ will always lie above or at the error function; which is the definition of a convex function. Therefore, our error function is strictly convex, as it is obtained by adding a strictly convex function (i) and a convex function (ii). Hence, optimization is far easier in this step as the error function has a global minimum, as seen in Fig. 2(b).

For both optimization steps, the Levenberg–Marquardt algorithm was used, yielding more accurate and consistent estimations than in [24] where learning was performed by minimizing the negative log likelihood of training data given path loss models predictions where only a penalty, in the form of higher variance, was added to all zero values, regardless of path loss predictions being positive or negative.

Then the model's noise variance $\sigma^2_{pl,j}$ can be estimated using maximum likelihood estimation. Assuming that the learned model $PL_j(\cdot)$ has additive white Gaussian noise, the uncertainty of a signal strength measurement $z_{*,j}$ at an arbitrary location $\mathbf{x}_*$ can be computed as:

$$p(z_{*,j}|\mathbf{x}_*, PL_j, \sigma_{pl,j}) = \mathcal{N}(z_{*,j}|PL_j(\mathbf{x}_*), \sigma^2_{pl,j}). \tag{5}$$

Considering that training data ($\mathbf{X}$, $\mathbf{z}_j$) was drawn independently from the distribution in Eq. (5), the likelihood function of the training data should be computed as the product of the likelihood of each individual training point. As training data corresponds to

signal strengths as measured by WNICs, no information is gained from training points where the training data is zero ($z_{i,j} = 0$) and the path loss model prediction is negative ($PL_j(\mathbf{x}_i) < 0$). Therefore, such points are excluded from the computation of the likelihood function of the training data, resulting in:

$$p(\mathbf{z}_j|\mathbf{X}, PL_j, \sigma_{pl,j}) = \prod_i \mathcal{N}(z_{i,j}|PL_j(\mathbf{x}_i), \sigma^2_{pl,j}), \tag{6}$$

for $i \in \{1 : n\}$ if $z_{i,j} > 0$ or $PL(\mathbf{x}_i) \geq 0$.

The model's noise variance can then be computed by maximizing the log of this likelihood, which is:

$$\begin{aligned}\ln\left(p(\mathbf{z}_j|\mathbf{X}, PL_j, \sigma_{pl,j})\right) = &-\frac{n_j}{2}\ln(2\pi) - n_j\ln(\sigma_{pl,j}) \\ &- \frac{1}{2\sigma^2_{pl,j}}\sum_i (z_{i,j} - PL_j(\mathbf{x}_i))^2,\end{aligned} \tag{7}$$

with $n_j$ being the total number of valid training data points $i$, as defined for the previous equation.

Finally, the value of $\sigma_{pl,j}$ which maximizes Eq. (7) can be computed by setting its derivative with respect to $\sigma_{pl,j}$ to zero. Solving this equation results in:

$$\sigma^2_{pl,j} = \frac{1}{n_j}\sum_i (z_{i,j} - PL_j(\mathbf{x}_i))^2. \tag{8}$$

### 4.2. GP for modeling mismatches

For modeling signal strength, we employ GPs, which were selected for its previous success in modeling wireless signal strength for localization in [19,20,23], as well for its ease to incorporate priors [24]. A more general treatment of GPs can be found in [26]. In this section, we showcase the main equations used for our specific approach.

In summary, GPs are generalizations of normal distributions to functions, describing functions of finite-dimensional random variables. Given some training points, a GP generalizes these points into a continuous function where each point is considered to have a normal distribution, hence a mean and a variance. The essence of the method resides in assuming a correlation between values at different points, this correlation is characterized by a covariance function or a kernel.

As we desire to model the shadowing and multipath components only, we first calculate a new training dataset ($\mathbf{X}$, $\mathbf{S}$) where $\mathbf{X}$ is the matrix of input samples $\mathbf{x}_i$, and $\mathbf{S}$ is the matrix of mismatches between RSS measurements ($\mathbf{Z}$) and the predictions obtained by the path loss models ($PL(\mathbf{X})$), i.e., $\mathbf{S} = \mathbf{Z} - PL(\mathbf{X})$. Then, under the GP approach, two assumptions are made. First, each data pair ($\mathbf{x}_i$, $\mathbf{s}_i$) is assumed to be drawn from a noisy process:

$$\mathbf{s}_i = f(\mathbf{x}_i) + \epsilon, \tag{9}$$

where $\epsilon$ is the noise generated from a Gaussian distribution with known variance $\sigma^2_n$.

Second, any two output values, $\mathbf{s}_p$ and $\mathbf{s}_q$, are assumed to be correlated by a covariance function based on their input values $\mathbf{x}_p$ and $\mathbf{x}_q$:

$$\text{cov}(\mathbf{s}_p, \mathbf{s}_q) = k(\mathbf{x}_p, \mathbf{x}_q) + \sigma^2_n\delta_{pq}, \tag{10}$$

where $k(\mathbf{x}_p, \mathbf{x}_q)$ is a kernel, $\sigma^2_n$ the variance of $\epsilon$ and $\delta_{pq}$ is one only if $p = q$ and zero otherwise.

Given these assumptions, for any finite number of data points, the GP can be considered to have a multivariate Gaussian distribution:

$$\mathbf{s} \sim \mathcal{N}(m(\mathbf{x}),\ k(\mathbf{x}_p, \mathbf{x}_q) + \sigma^2_n\delta_{pq}), \tag{11}$$

and therefore be fully defined by a mean function $m(\mathbf{x})$ and a kernel function $k(\mathbf{x}_p, \mathbf{x}_q)$.

For our approach we set $m(\mathbf{x})$ as the zero function, i.e., $m(\mathbf{x}) = 0$ for all possible $\mathbf{x}$. This makes the shadowing and multipath components tend to zero when no data is available, and the whole sensor model to the path loss component, which is the desired behavior.

Regarding the kernel function, we select the widely used squared exponential kernel, defined as:

$$k_{se}(\mathbf{x}_p, \mathbf{x}_q) = \sigma^2_{se}\exp\left(-\frac{(\mathbf{x}_p - \mathbf{x}_q)^2}{l^2_{se}}\right), \tag{12}$$

with hyperparameters $\sigma^2_{se}$ (known as the signal variance), and $l_{se}$ (known as the length-scale).

Although previous work [19] recommends the use of Matérn kernels instead of squared exponential ones in wireless signal strength-based localization as to avoid squared exponential kernels' tendency to output unreasonably low predicted variances, in practice, the use of Matérn kernels did not noticeably improve the localization accuracy in our approach. Therefore, squared exponential kernels were selected for convenience.

Now, that we have fully defined our system, kernel hyperparameters $\theta_{se} = (\sigma_{se},\ l_{se})$ and system noise $\sigma_n$ have to be learned for each access point $j$. This can be done using maximum a posteriori estimation of its set of parameters $(\theta_{se,j}, \sigma_{n,j})$, which occurs when $p(\theta_{se,j}, \sigma_{n,j}|\mathbf{X}, \mathbf{s}_j)$ is maximized.

Using the Bayes' rule and assuming an uninformative prior distribution $p(\theta_{se,j}, \sigma_{n,j}|\mathbf{X})$, we have:

$$p(\theta_{se,j}, \sigma_{n,j}|\mathbf{X}, \mathbf{s}_j) = p(\mathbf{s}_j|\mathbf{X}, \theta_{se,j}, \sigma_{n,j}); \tag{13}$$

therefore, the problem of finding the maximum a posteriori estimation of $p(\theta_{se,j}, \sigma_{n,j}|\mathbf{X}, \mathbf{s}_j)$ becomes equivalent to minimizing the negative log likelihood ($nll_{GP,j}$) of $p(\mathbf{s}_j|\mathbf{X}, \theta_{se,j}, \sigma_{n,j})$, where:

$$\begin{aligned}nll_{GP,j} = &\ 0.5n\ln(2\pi) + 0.5\ln(\det(\mathbf{K} + \sigma^2_{n,j}\mathbf{I}_n)) \\ &+ 0.5(\mathbf{s}^T_j(\mathbf{K} + \sigma^2_{n,j}\mathbf{I}_n)^{-1}\mathbf{s}_j),\end{aligned} \tag{14}$$

with $\mathbf{K} = k(\mathbf{X}, \mathbf{X})$, and $\mathbf{K} + \sigma^2_{n,j}\mathbf{I}_n \in \mathbb{R}^{n\times n}$ the covariance matrix between all training points $\mathbf{X}$, usually called Gram Matrix.

Instead of learning a different set of parameters for each access point, it is also possible to learn a single set of parameters $(\theta_{se}, \sigma_n)$ for all. It is important to note that while these parameters characterize how model outputs and inputs are correlated, model outputs (predictions) are still mostly dependent on their individual training data. This will becomes obvious once we derive the equation that characterizes model predictions (Eq. (18)) later in this section. The negative log likelihood ($nll_{GP}$) for this case can be easily derived from Eq. (14) by noting that in this case, all models have the same Gram matrix. Thus we can rewrite the summation of all individual log likelihoods as:

$$\begin{aligned}nll_{GP} = &\ 0.5nm\ln(2\pi) + 0.5m\ln(\det(\mathbf{K} + \sigma^2_n\mathbf{I}_n)) \\ &+ 0.5\text{tr}(\mathbf{S}^T(\mathbf{K} + \sigma^2_n\mathbf{I}_n)^{-1}\mathbf{S}).\end{aligned} \tag{15}$$

The advantage of using a different set of parameters per access point is better model fitting. However, in our testing, only a 1.24 dBm root mean square difference between generated predictions was observed. The main advantage of using a single set of parameters is faster computations. Furthermore, for modeling shadowing and multipath components, assuming that the correlations between inputs and outputs of all models are similar is sensible. As both phenomena are location dependent, and all locations are common to all access points. Nevertheless, it was tested if using a different set of parameters would yield improvements in localization accuracy. And as expected, no noticeable difference in

performance was obtained. This ratified our choice of using a single set of parameters to model the covariance of the shadowing and multipath components for all access points.

Using this approach, once parameters have been optimized, it is possible to make predictions $\mathbf{s}_*$ for an unknown data point $\mathbf{x}_*$, by rewriting Eq. (11) considering the known and unknown data points as:

$$\begin{bmatrix} \mathbf{S} \\ \mathbf{s}_* \end{bmatrix} \sim \mathcal{N}\left(\mathbf{0}, \begin{bmatrix} \mathbf{K} + \sigma_n^2 \mathbf{I}_n & \mathbf{k}_* \\ \mathbf{k}_*^T & k_{**} + \sigma_n^2 \end{bmatrix}\right), \tag{16}$$

where $\mathbf{k}_* = k(\mathbf{X}, \mathbf{x}_*) \in \mathbb{R}^{n\times 1}$ is the vector that relates training points $\mathbf{X}$ with the test point $\mathbf{x}_*$, and $k_{**} = k(\mathbf{x}_*, \mathbf{x}_*)$ is the variance of the test point, which for the squared exponential kernel is equal to $\sigma_{se,j}^2$. By conditioning $\mathbf{s}_*$ to $\mathbf{x}_*$, $\mathbf{X}$ and $\mathbf{S}$, it is finally obtained that:

$$p(\mathbf{s}_*|\mathbf{x}_*, \mathbf{X}, \mathbf{S}) \sim \mathcal{N}(\mathbb{E}[\mathbf{s}_*], \text{var}[\mathbf{s}_*]), \tag{17}$$

where,

$$\mathbb{E}[\mathbf{s}_*] = \mathbf{k}_*^T(\mathbf{K} + \sigma_n^2 \mathbf{I}_n)^{-1}\mathbf{S}, \tag{18}$$

$$\text{var}[\mathbf{s}_*] = (\sigma_{se}^2 + \sigma_n^2 - \mathbf{k}_*^T(\mathbf{K} + \sigma_n^2 \mathbf{I}_n)^{-1}\mathbf{k}_*)\mathbf{1}_m, \tag{19}$$

with $\mathbf{1}_m \in \mathbb{R}^{1\times m}$ being a vector whose elements are all ones. Variance predictions for all access point models are the same as they entirely depend on their covariance function, which is the same for all. As all points are considered to be Gaussian distributions, in order to make predictions, only Eqs. (18) and (19) need to be computed.

Fig. 3(b) shows an example of the predicted means generated by this approach. As it can be seen, GPs have no problem learning the complex mappings required to model signal strength's shadowing and multipath components. From Fig. 3(e), which shows the model's predicted variances, it is important to note that predicted variances naturally grow for locations farther from training data. This is the desired behavior, as the uncertainty of predictions increases for farther locations, and is the result of the second term of Eq. (19), $\mathbf{k}_*^T(\mathbf{K}+\sigma_n^2\mathbf{I}_n)^{-1}\mathbf{k}_*$, tending to zero, as correlations between predicted locations and training data inputs weaken, making the variance tend to $k_{**}$ - its maximum value.

### 4.3. Sensor model

Now that all components (path loss, shadowing and multipath) have been learned, our sensor model needs to integrate them and output the predicted means and variances of the signal propagation phenomena, under WNIC constraints.

The signal strength propagation model is the result of adding path loss, shadowing and multipath; therefore, for computing its predicted means it is reasonable to simply add the predictions generated by both the path loss and the GP models. In order to account for WNICs sensing limitations, we bound this sum below by zero. Therefore, for any location $\mathbf{x}_*$, the predicted mean ($\mathbb{E}[\mathbf{z}_*]$) of our sensor model is computed as:

$$\mathbb{E}[\mathbf{z}_*] = \max(PL(\mathbf{x}_*) + \mathbb{E}[\mathbf{s}_*], 0), \tag{20}$$

with $PL$ being computed by Eq. (1) and $\mathbb{E}[s_*]$ by Eq. (18).

Computing predicted variances for our sensor model is a more interesting problem. From our approach assumptions, we have that path loss models learn the path loss component, considering the shadowing and multipath components as model noise, making path loss variances $\sigma_{PL}^2$ at least as large as these components. As the GP model learns shadowing and multipath components, assuming path loss components were properly learned, it is a fair assumption that GP predicted variances var[$\mathbf{s}_*$] are always equal or lower than path loss variances — which we have verified experimentally. If only this is considered, then the adequate variances of the sensor model should be the GP model predicted variances, as path loss predictions uncertainties were refined by using the GP model, and the model variances are the result of this procedure.

However, this changes once WNICs sensing limitations are considered, as we argue that for negative values path loss predicted variances have to be adjusted. Because, when path loss models predict a negative signal strength whose absolute value is large, the certainty of WNICs output to be zero is high, as even if the prediction is off by several standard deviations, it would still remain negative, making the WNICs output still zero. For example, given a propagation model with variance equal to 4, if the model makes a prediction of $-6$, the probability of the WNIC to output something other than zero would be 0.1349%; therefore computing the model's prediction as a zero-mean Gaussian with a variance of 4, would not be correct.

Considering an adjusted path loss model, which remains Gaussian with mean values equal to the path loss predictions bounded below by zero, $\overline{PL}(\mathbf{x})$. Our problem is now to compute an adequate adjusted variance $\sigma_s^2$ which increases model confidence for negative path loss predictions. Although lower variances are desirable, as they would yield more precise mappings, we must be careful not to generate unreasonably low ones. In order to lessen underestimation of adjusted variances, we compute them so for all propagation models we have at least a 99.7% confidence that signal strength measurements sampled from them will lie within three standard deviations of their corresponding sensor models.

Due to having a Gaussian distributions, the probability of sampling a measurement equal or lower than $PL(\mathbf{x}) + 3\sigma_{pl}$ is 0.9986 for all path loss models, as:

$$\int_{-\infty}^{PL(\mathbf{x})+3\sigma_{pl}} \mathcal{N}\left(z; PL(\mathbf{x}), \sigma_{pl}^2\right) dz = 0.9986. \tag{21}$$

For the case where $PL(\mathbf{x}) + 3\sigma_{pl} \leq 0$, once negative samples are set to zero due to the lower sensing bound, we have that 99.86% of the samples are expected be zero. As $\overline{PL}(\mathbf{x}) = 0$, adjusted path loss models will comply with our condition regardless of the value of $\sigma_s$. For $PL(\mathbf{x}) + 3\sigma_{pl} > 0$, our condition will also comply as long as $\overline{PL}(\mathbf{x}) + 3\sigma_s$ is at least $PL(\mathbf{x}) + 3\sigma_{pl}$. Using this inequality and the previous stated case, we have that the smallest viable $\sigma_s$ can be computed as:

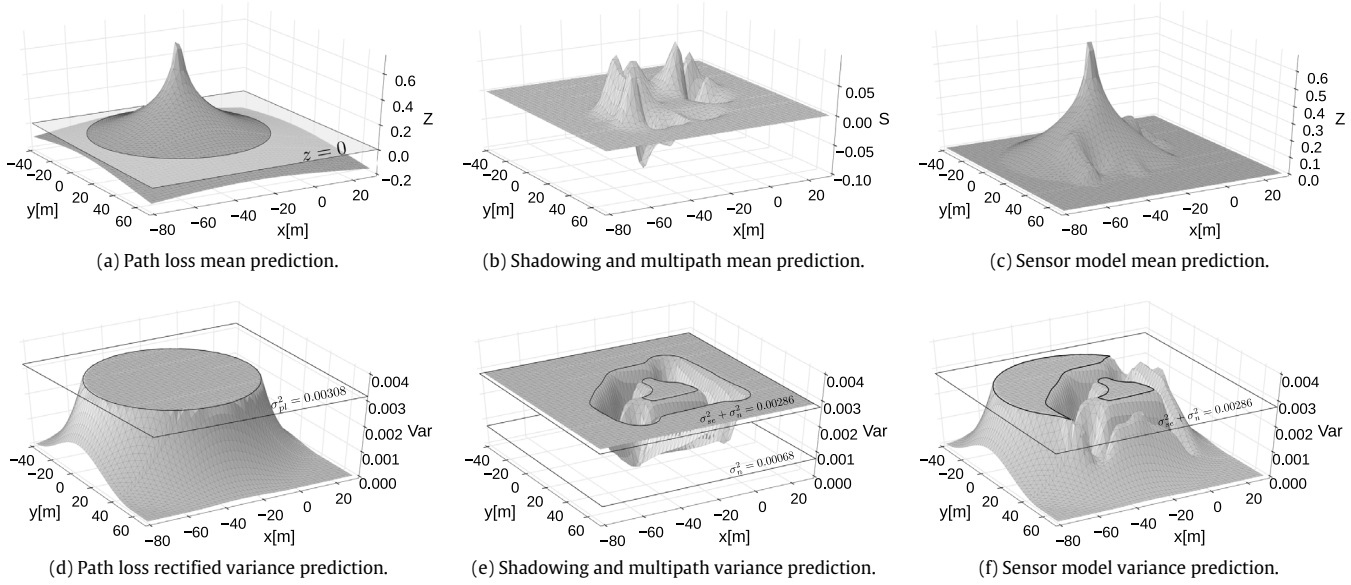
$$\sigma_s = \frac{1}{3}\max(PL(\mathbf{x}) + 3\sigma_{pl}, 0) - \max(PL(\mathbf{x}), 0) \tag{22}$$

This equation also provides variances which comply for $PL(\mathbf{x}) > 0$ as $\overline{PL}(\mathbf{x}) = PL(\mathbf{x})$ and $\sigma_s = \sigma_{pl}$, making the probability of a sample from the propagation model to lie within 3 standard deviations equal to 99.7% as $p(PL(\mathbf{x}) - 3\sigma_{pl} < \mathcal{N}\left(z; PL, \sigma_{pl}\right) < PL(\mathbf{x}) + 3\sigma_{pl}) = 0.997$

With this adjusted path loss variance $\sigma_s$, we now compute the sensor model variance as the minimum value between the path loss models and the GP model predicted variances:

$$\text{var}[\mathbf{z}_*] = \min(\sigma_s^2, \text{var}[\mathbf{s}_*]). \tag{23}$$

Fig. 3(f) shows an example of the generated predicted variances by this equation, where we can observe the result of the interaction




(a) Path loss mean prediction. (b) Shadowing and multipath mean prediction. (c) Sensor model mean prediction.

(d) Path loss rectified variance prediction. (e) Shadowing and multipath variance prediction. (f) Sensor model variance prediction.

**Fig. 3.** Example of mappings generated by our approach. Figures (a), (b) and (c) show our models mean predictions, generated using Eqs. (1), (18) and (20), respectively. Our path loss model (a) is a parametric function, which has good generalization but cannot capture all the complexities of signal strengths. For capturing these complexities a Gaussian Process (b) is used. This is a data-driven model which can learn the complex mappings necessary to model the complexities our first model fails to capture. Finally, our sensor model (c) is the aggregation of the previous two, with additional consideration to compensate for hardware limitations when sensing signal strengths. Figures (d), (e) and (f) show our models' variance predictions, generated using Eqs. (22) and (18) and (19), respectively. (d) shows variance predictions when path loss models variance ($\sigma^2_{pl} = 3.08\text{e}{-3}$ in this example) is adjusted to account for the change in the certainty of negative means predictions (refer to Section 4.3 for a detailed explanation). This results in a variance mapping, where for all non-negative mean predictions a maximum $\sigma^2_{pl}$ value is computed, and for negative mean predictions it is reduced depending on its magnitude until reaching 0. Gaussian Process variances (e) depend entirely on its learned covariance function and the spatial location of its training data. As a data-driven approach, predicted variances are minimum and equal to the model's signal noise variance ($\sigma^2_n = 6.8\text{e}{-4}$ in this example) for locations in its training data and naturally grow for those farther from it (until reaching $\sigma^2_{se} + \sigma^2_n = 2.86\text{e}{-3}$ in this example). Sensor model variance predictions (f) are the result of the interaction of these two models, which make variances increase for locations farther from training points (GP model) and decrease for locations far from the access point's predicted location (path loss model).

of path loss models and GPs' different behaviors: (1) GPs' increase in variance for locations farther from training points, and (2) path loss models' decrease for negative mean predictions whose absolute values are large, which occur for locations far from the access point's predicted location.

The issue with this approach is that for $PL(\mathbf{x}) + 3\sigma_{pl} \leq 0$ it generates variances equal to zero. Although such variances do comply with our condition (99.86% of samples being within 3 standard deviations of the model's mean), they are undesirable as they generate zero value probabilities. To avoid zero value probabilities we add a uniform distribution over the entire RSS range to the sensor model. This distribution is scaled by a small value, $p_u$, which is equal to 1e-3 for all our testing. This value is introduced to account for random measurements, and most importantly, so sensor models generate non-zero probabilities regardless of the value of $PL(\mathbf{x})$.

Finally, as previously mentioned, we also consider a point mass distribution at zero, used to compensate for unmodeled issues which could lower signal strength below WNICs sensing limits (such as large dynamic obstacles occluding signals), as well as WNICs' hardware failures (such as WNICs dropping measurements due to checksum errors).

The value of this point mass distribution is computed directly from training data, by computing the probability of measuring a zero value at a location where $PL(\mathbf{x})$ predicts a non-zero value. In our implementation we calculated it as:

$$p_{zero} = p_0 + \frac{1}{n * m} \sum_{i,j} \text{sgn}(PL_j(\mathbf{x}_i)) \quad \forall i, j : z_{i,j} = 0, \tag{24}$$

where $p_0$ is a small value added to compensate for possible unobserved phenomena.

Given all these considerations, the total probability for any incoming signal strength for access point $j$ is computed as:

$$p(z_{new,k}|\mathbf{x}_*) = (1 - p_u - p_{zero})\varphi\left(\frac{z_{new,j} - \mathbb{E}[z_{*,j}]}{\sqrt{\text{var}[z_{*,j}]}}\right) + p_u + p_{zero}\delta_{new,j}, \tag{25}$$

with $\delta_{new,j}$ being equal to 1 when $z_{new,j}$ is zero, and 0 otherwise; and $\varphi(\cdot)$ the standard normal distribution.

This equation computes the likelihood for each individual access point. As each access point can be considered independent given the location $\mathbf{x}_*$, the integrated likelihood $p(\mathbf{z}_{new}|\mathbf{x}_*)$ should be obtained by multiplying the individual likelihoods for all access points. However, it has been observed in practice that this leads to overconfident estimates, yielding suboptimal results [20]. A simple remedy is to replace each individual likelihood by a "weaker" version of it $p(\mathbf{z}_{new}|\mathbf{x}_*)^\alpha$ with $\alpha < 1$ - similar to what is suggested in [27] regarding beam models for rangefinders.

For this work, the value of $\alpha$ selected is the inverse of the number of access points $m$, making integration of individual likelihoods not its product but its geometric average:

$$p(\mathbf{z}_{new}|\mathbf{x}_*) = \left(\prod_{k=1}^{m} p(z_{new,k}|\mathbf{x}_*)\right)^{1/m}. \tag{26}$$

## 5. Experimental evaluations

In this section, we test and discuss the advantages of our model (WGPPL) with respect to a standard GP formulation (WGP) using sensor data acquired in office-like environments. WGP considers the same formulation described in Section 4.2, with the same mean

and kernel functions (a zero function and the squared exponential kernel), but using different training data. So, instead of only learning the shadowing and multipath component, it learns the whole sensor model by using the training dataset ($\mathbf{X}$, $\mathbf{Z}$) instead of using ($\mathbf{X}$, $\mathbf{S}$). Predictions for this model are done directly using Eqs. (18) and (19).

First, we describe in detail our signal strength acquisition protocol and then assess the previously mentioned models based on the predictions and likelihoods they generate. Then, although it is difficult to discern which model is better for localization purposes without actually testing it on a localization algorithm, we analyze the mappings and posteriors obtained and provide some metrics that give a good idea of the performance of the different approaches. Discussion and quantitative analysis of localization accuracies are deferred to Section 6 where the models are used in conjunction with an MCL algorithm.

### 5.1. Datasets and data acquisition

For our work, wireless signal strength measurements were acquired passively by listening to access points' beacon frames without actively interacting with them. Therefore, samples were acquired at the fixed time intervals set by each access point, who would typically broadcast one beacon frame every 100 ms.

Three different buildings at the University of Tokyo were surveyed for testing. We did not modify the existing WLAN infrastructure of any of these buildings. No additional access points were added, nor existing ones were relocated. We also did not have any prior knowledge regarding the number of access points nor their locations. Occupancy grid maps of the buildings, as generated by a laser rangefinder, are shown in Fig. 4. For each building, several datasets were taken by surveying signals in 2 different channels of the 2.4 GHz band. Surveying 2 channels out of the 14 available was a practical choice, and our approach can work with any number of channels. Although surveying more channels is equivalent to having more access points from which extract information, it important to consider that WNICs cannot sense signals several channels at the same time, hence it is necessary to either keep your WNIC switching between channels (which reduces the effective time you can listen to each individual channel), or use multiple WNICs. In our experiments, signals were acquired using only one Dual Band Wireless-AC 8260 WNIC equipped with a Panasonic Let's Note CZ-SZ5 laptop.

For all tests, this laptop was placed on top of a Pioneer 3 DX mobile robot which provided mobility. The Pioneer also recorded odometry information which is used for the MCL algorithm that will be introduced in Section 6. It was also equipped with a laser rangefinder only used for building occupancy maps and obtaining the ground truth position of the datasets. For all tests, the robot was operated constantly, without stopping to acquire data points. Fig. 5 shows the hardware configuration for data acquisition.

Regarding software configuration, data acquisition was run on a 64 bits Ubuntu 14.04 LTS OS using *tcpdump* version 4.5.1 as wireless packet analyzer. All sensor and odometry information was recorded with timestamps in a rosbag format. For each building, three runs were performed, each recorded in a separate dataset. Table 1 shows a summary of the data collected on each dataset.

### 5.2. Mean prediction

First, we assess RSS mean prediction is enhanced when using path loss models (WGPPL) over the classic formulation (WGP), as well as learning a different set of parameters ($\sigma_{se}$, $l_{se}$, $\sigma_n$) per access point (multiple sets) over a single set of parameters for all access points (single set). We also compare these approaches with other state of the art approaches. In particular with Gaussian

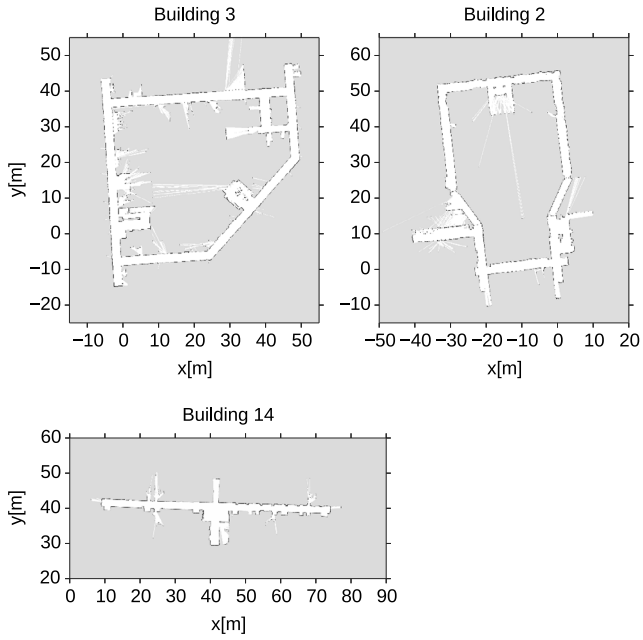



**Fig. 4.** Occupancy grid maps of the environment used for testing.

**Fig. 5.** Hardware configuration for data acquisition: CZ-SZ5 laptop and a Pioneer 3 DX.

Process Regression with Polynomial Surface Fitting Mean (PSFM-GPRS) [28] which uses a second degree polynomial function instead of the path loss models we employ, a graph-based approach (GraphWiFi) [18] that uses interpolations in a graph for making predictions, and Vector Field Maps [17] which employ bilinear interpolations for making predictions.

For each building, we randomly divided *dataset1* into 10 equal sized subsets, and trained both models using 9 of the subsets, to later test prediction accuracy on the remaining one (10-fold cross-validation). We also used all *dataset1* to train both models and tested prediction accuracy on *dataset2* and *dataset3*. In order to decrease the effect of fast fading and sensor noise, the data corresponding to 10 data points was combined and the mean RSS values used. Fig. 6 shows the root of the mean squared error between the testing datasets and the recorded RSS values for the mean predictions of both approaches. From the figure, it becomes evident that mean predictions either WGP or WGPPL are essentially the same, with less than 0.25 dB difference between predictions (we attribute this to the ability of GPs to successfully learn complex mappings from data). Also, both methods generate comparable mapping accuracies to all other approaches for *dataset2* and *dataset3*, with the higher accuracy of GraphWiFi and VFM in *dataset1* due to these approaches directly interpolating training data, which is equal to 90% of the ground truths in our 10-fold-cross-validation scheme.

**Table 1**
Detailed information of datasets employed in the different environments shown in Fig. 4.

| | Duration | Traversed distance | Average speed | #Access Points[a] | #Locations | #Sensed RSS |
|---|---|---|---|---|---|---|
| B2 dataset1 | 20 : 18 s | 225.13 m | 0.18 m/s | 144 | 1204 | 237268 |
| B2 dataset2 | 12 : 28 s | 159.53 m | 0.21 m/s | 142 | 732 | 145661 |
| B2 dataset3 | 11 : 30 s | 160.07 m | 0.23 m/s | 108 | 683 | 127497 |
| B3 dataset1 | 27 : 37 s | 284.06 m | 0.17 m/s | 88 | 1638 | 251040 |
| B3 dataset2 | 16 : 31 s | 184.24 m | 0.19 m/s | 80 | 979 | 143447 |
| B3 dataset3 | 9 : 10 s | 190.47 m | 0.35 m/s | 73 | 510 | 82138 |
| B14 datatest1 | 12 : 11 s | 141.19 m | 0.19 m/s | 36 | 722 | 60894 |
| B14 datatest2 | 8 : 16 s | 95.92 m | 0.19 m/s | 31 | 489 | 42956 |
| B14 datatest3 | 5 : 27 s | 96.63 m | 0.30 m/s | 26 | 305 | 29193 |

[a] Difference in the number of access points among datasets belonging to the same building is due to some datasets surveying smaller areas (hence being exposed to signals from fewer access points) and by signals from some access points not being available when the second or third survey of the environment was performed (due to access points being off or transmitting on a different channel which was not surveyed).

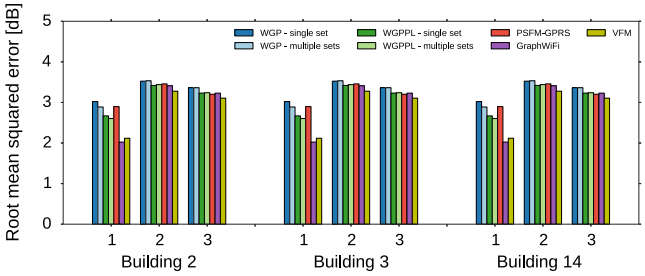



**Fig. 6.** Root mean square errors (RMSE) when predicting data samples from available datasets show no significant difference between all models' mean predictions.

### 5.3. Variance predictions

As WGP predicted variances are computed using the same equation than WGPPL's shadowing and multipath components, the generated mappings share the same characteristics - i.e., variances are at the lowest for locations near its training inputs, smoothly growing for locations farther from them. It also makes the difference between predicted variances generated by both models the same as those generated by the WGPPL's shadowing and multipath component (shown in Fig. 3(e)) and its sensor model (shown in Fig. 3(f)). These differences originate from WGPPL's bounding of variances using Eq. (23), which allows WGPPL predicted variances to be low for locations far from its path loss model's predicted access point location. Although intuitively, having lower variances represents more certain predictions, it is difficult to directly assess if this is an improvement or not, as it mostly depends on the accuracy of the predicted means and how it fares with respect to other points. Comparisons with GraphWiFi and VFM are not possible as predicted variances in GraphWiFi variance predictions represent the expected uncertainty of the signal strength predictions without considering locations. Instead of increasing the uncertainty of predictions at locations far from training data locations, they propose a localization algorithm which restricts the possible robot locations to areas in close proximity to training data locations. While this restriction allows for successful robot localization, their model loses generalization and requires this special localization algorithm, which also makes this model incompatible for comparisons with our classic MCL. Regarding VFM, predicted variances are not even computed. Instead, possible robot locations are searched by matching measured signal strengths to mean mappings, and then all tentative solutions are tracked using a multi-hypothesis Kalman filter.

### 5.4. Posterior distributions

In the context of robot localization, the most important application of our mappings is its role in computing the posterior distribution of locations given sensor information - $p(\mathbf{x}|\mathbf{z})$. To compute these posteriors, we first compute the joint likelihood distributions given sensor measurements for any tentative location ($p(\mathbf{z}|\mathbf{x})$) using our models (Eq. (26)), and then apply Bayes rule assuming noninformative priors, obtaining:

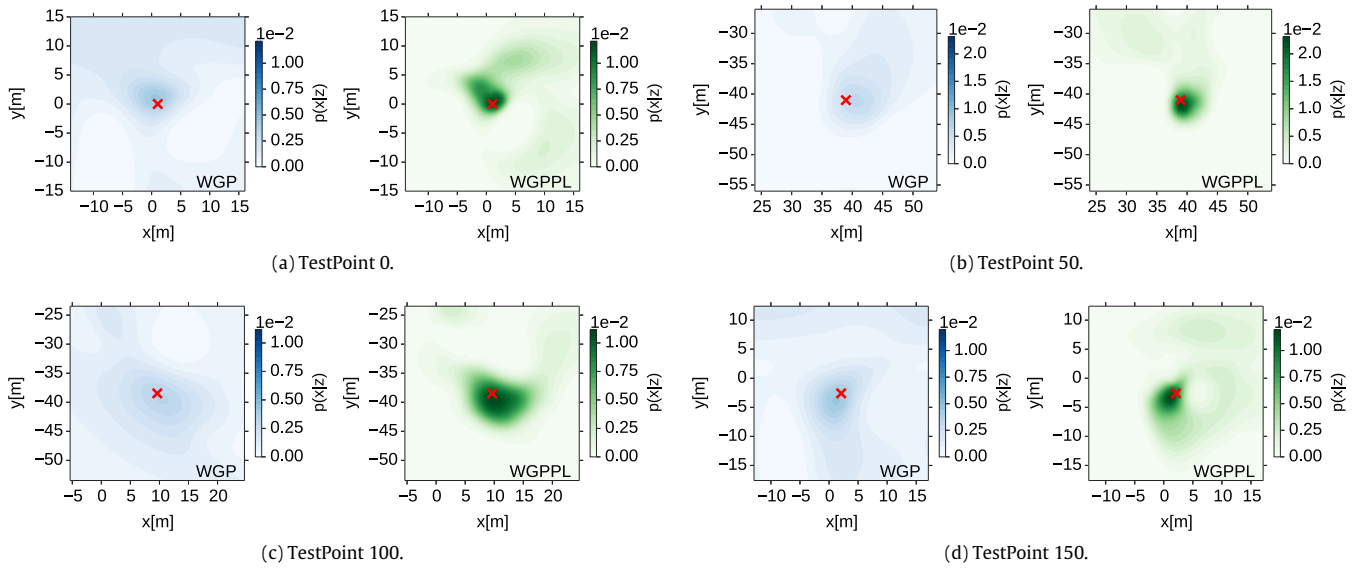
$$p(\mathbf{x}|\mathbf{z}) = cp(\mathbf{z}|\mathbf{x}), \tag{27}$$

where $c$ is a normalization constant. In order to compare the posteriors generated between WGP and WGPPL models, we do need to compute this normalization constant; however, its computation is not necessary for its use in conjunction with Bayes-based localization algorithms — like the MCL algorithm we will introduce in the following section. As, in such algorithms, it is only necessary to compute the posterior distributions up to a proportionality constant.

In general, good posterior distributions should allocate high probabilities to areas surrounding the true location and as low as possible to farther locations. Fig. 7 shows several examples of posteriors generated by each approach at different testing points where WGPPL posteriors are evidently better. While both models generate posteriors centered around the robot's true location, i.e., assign higher probabilities to locations near the robot's true location; WGPPL posteriors are considerably more peaked than WGP ones, which are rather flat in comparison — this can be easily noticed from the figures as WGPPL posteriors have very dark shades (representing high probabilities) near the robot's true location, marked with an X, and light shades at locations farther from it, while WGP posteriors have a more uniform shade for all locations (representing more similar probabilities). This is a consistent result obtained for all testing points, for all datasets.

From Eqs. (26) and (27) it can be easily proved that posteriors solely depend on the models' predicted means and variances. In addition, from our previous discussions, we have already established that both models generate considerably similar predicted means, mainly differentiating by their predicted variances. Therefore, differences in the generated posteriors should be mostly attributed to the difference in the predicted variances - i.e., on the precision of the mappings. The reason why WGPPL predicted variances are capable of enhancing posteriors to this degree lies on the model's capability of unequivocally assigning low probabilities to locations far from the robot's true location, even when such locations are far from its training datasets. WGP fails to generate precise predictions for such cases as it must consider its lack of generalization capabilities, or risk generating inaccurate predictions. Different from WGP models, WGPPL can use its path loss components to provide confident predictions even for those locations.

In order to quantify the effect of these distributions we sampled the posteriors and computed the normalized probability of a sample being generated at a certain distance with respect to the true position — the higher the probability is close to zero, the better. The probability is computed as the number of samples located within concentric circles around the true location,



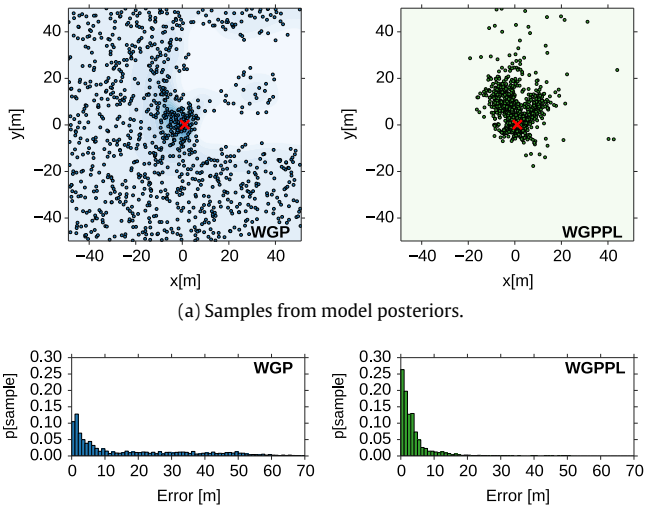

(a) TestPoint 0.

(b) TestPoint 50.

(c) TestPoint 100.

(d) TestPoint 150.

**Fig. 7.** Comparison of probability distributions generated by WGP and WGPPL models for several testing points. Light shaded areas represent low posterior probability, while darker ones, high. Test points' true locations are marked by an **X**. While both models consistently generate posteriors centered around the robot's true location, WGPPL posteriors are considerably more peaked than WGP ones, which are rather flat in comparison. This can be easily noticed by WGPPL posteriors darker shades near the robot's true location, and light shades at locations farther from it, while WGP posteriors have a more uniform shade of color for all locations.




(a) Samples from model posteriors.

(b) Probability of samples with respect to their distance to true location.

**Fig. 8.** To quantitatively assess WGP and WGPPL, we sample their posteriors in order to compute the probability distribution of the samples' distance to the true location. Models with higher probabilities close to zero, as WGPPL, being preferable. Figure (a) shows 1000 samples taken from the same posteriors shown in Fig. 7(a), while (b) shows a histogram of its computed probability.

normalized by the area considered of such ring. For sampling, we used an accept–reject algorithm, that although computationally inefficient provides unbiased samples. Fig. 8(a) shows an example of samples taken from the distributions both models and Fig. 8(b) the histogram of their corresponding normalized probabilities. As it can be observed WGGPL samples are more concentrated around robot's true position than WGP samples.

For this example, samples from WGPPL are distributed closer to the true location than those from WGP. This results in the WGPPL sample probability density to be peaked towards zero, steadily decreasing until around 15 m, where they become extremely low. For WGP samples, probabilities are not centered around zero, and

**Table 2**
Average probability of locations of samples with respect to true location obtained from sampling WGP and WGPPL posteriors for all buildings and datasets.

| | Building 2 | | Building 3 | | Building 14 | |
|---|---|---|---|---|---|---|
| | WGP | WGPPL | WGP | WGPPL | WGP | WGPPL |
| d1 | 0.079 | 0.174 | 0.085 | 0.218 | 0.078 | 0.237 |
| d2 | 0.067 | 0.144 | 0.074 | 0.175 | 0.086 | 0.217 |
| d3 | 0.075 | 0.150 | 0.074 | 0.149 | 0.085 | 0.190 |

at 15 m they are low but not negligible. In order to verify that the same tendencies occurred for all cases, the same test was performed for all data points in all datasets available. Table 2 presents the average probabilities around ground truth obtained. From the table, it can be observed that for all cases our approach generates higher probabilities at locations close to the true location.

Although having a higher concentration of particles around the true location is intuitively a desired characteristic, it is not evident how much, quantitatively, this aids in localization. The next section addresses this issue by using the posteriors compared here as the basis of a localization algorithm.

## 6. MCL for wireless-based localization

This section shows localization accuracies when using either the WGP or WGPPL models in conjunction with an MCL algorithm. MCL algorithms are a family of algorithms widely used for localization in robotics and are often the default choice given its ease of implementation and good performance across a broad range of localization problems.

In general MCLs are an implementation of Bayesian Filters, which estimate the posterior distribution of robot's poses $\mathbf{p}$ conditioned on sensor information — with the robot's pose defined as its location and heading angle. This posterior, called belief ($Bel(\mathbf{p})$), is computed in a recursive manner given sensor information $\mathbf{z}$ and actuation commands $\mathbf{a}$:

$$Bel(\mathbf{p}_t) \propto p(\mathbf{z}_t|\mathbf{p}_t) \int p(\mathbf{p}_t|\mathbf{p}_{t-1}, \mathbf{a}_{t-1}) Bel(\mathbf{p}_{t-1}) d\mathbf{p}_{t-1}, \quad (28)$$

with $Bel(\mathbf{p})$ being represented by a set of possible pose samples, referred as particles, and the likelihood of them being correct, represented by weights.

In order to compute Eq. (28), two models are necessary: $p(\mathbf{p}_t|\mathbf{p}_{t-1}, \mathbf{a}_{t-1})$ and $p(\mathbf{z}_t|\mathbf{p}_t)$. The first, called robot motion model, computes the likelihood of a pose given the robot's previous believed poses and actuation inputs — though often odometry information is used instead of actuation inputs, as it tends to be more reliable. The second, called perceptual model, computes the likelihood of a sensor reading with respect to the robots believed pose. For all testing presented in this section the robot motion model used was the commonly used odometry model and the perceptual model was either the WGP or WGPPL posteriors as the model's joint likelihood $p(\mathbf{z}|\mathbf{x})$ is equivalent to the required perceptual model given our assumption of robot's heading not affecting sensor measurements.

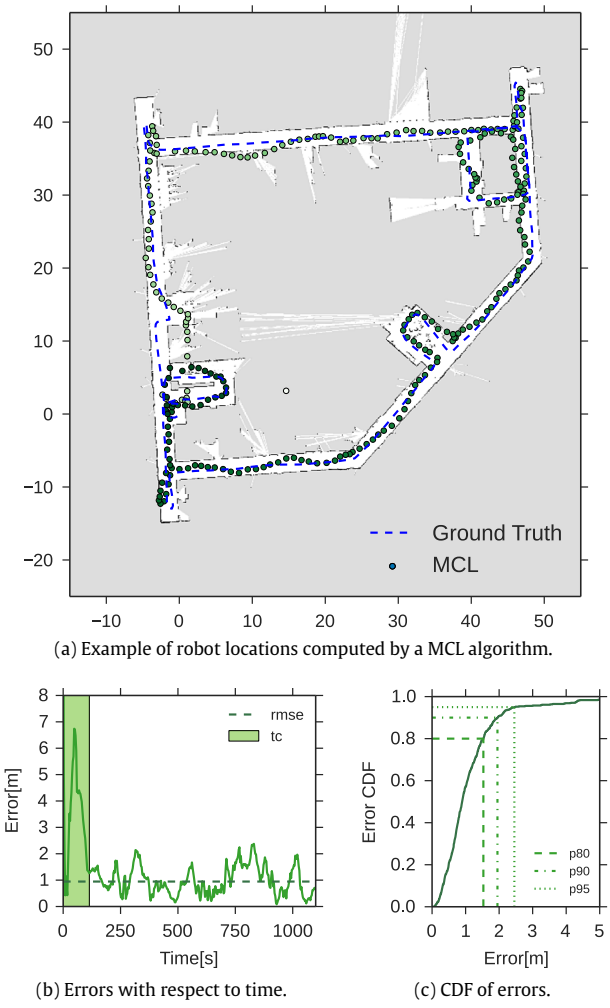
MCL implements Eq. (28) in two steps: a motion update, and a sensor update and an importance sampling. In the first step, the motion update takes previous particles and updates them considering its motion model and actuation inputs. The updated samples become distributed according to $p(\mathbf{p}_t|\mathbf{p}_{t-1}, \mathbf{a}_{t-1}) \times Bel(\mathbf{p}_{t-1})$, which is called the proposal distribution. This step can be run whenever new odometry information becomes available. In the second step, weights $w^{(i)}$ are calculated using the perceptual model and sensor information. These weights are used for importance sampling, where new particles are sampled using the pairs $\langle \mathbf{p}_{t-1}, w_{t-1} \rangle$. This resampling compensates for the mismatches between the proposal distribution and Eq. (28). This step can be performed at a different rate than the motion update, whenever sensor information becomes available.

For testing the performance of all our proposed algorithms we employed the open source framework ROS (Robot Operating System). All implementations are used in conjunction with the time-stamped sensor information previously acquired. Ground truth locations are computed by an MCL using the laser rangefinder measurements collected, and errors are defined as the X-Y Cartesian distance between these locations and those estimated by our wireless based MCL.

In order to quantify the performance of the localization outputs, several metrics are considered: convergence time (tc), root mean square errors (RMSE), and precision at 80 percent (p80). Tc is computed as the time required for localization errors to attain stability, RMSE is computed as the mean value of errors after tc is achieved. The precision of the localization outputs for a fixed percentile is defined as the value for which that percentile of all errors is lower or equal. Therefore, if the localization precision at 80% is 1m, it means that 80% of the localization outputs are within 1m of the true location. Fig. 9 shows an example of the localization outputs for one implemented MCL algorithm, as well as an illustration of the different metrics for such run.

Testing using time-stamped logs allows us to freely modify the algorithms' key parameters and thoroughly examine their impact. This, while using real data and in a real-time manner, without needing to re-run experiments for each parameter configuration. Furthermore, it also allows us to run each parameter configuration several times. In this work, for each configuration of parameters tested we ran 25 simulations unless otherwise specified. Due to the randomized nature of MCL algorithms, this is necessary to guarantee that the results presented are the average expected localization errors and not just outliers with good/bad performances.

The most important parameter to evaluate when implementing any MCL algorithm is the number of particles to be used. A higher number of particles, allows the algorithm to better represent any arbitrary distribution; moreover, it tends to increases the algorithm's accuracy (till a certain degree) and reduces the chances of failure in convergence. However, the usage of a high number of particles can strain the hardware capabilities of our system. For our implementation using wireless signals, other than the impact of the number of particles and our main discussion regarding the usage of WGPPL over WGP as perceptual models, we also discuss two other parameters: the sensor update rate, and the initialization of particles.




(a) Example of robot locations computed by a MCL algorithm.

(b) Errors with respect to time.

(c) CDF of errors.

**Fig. 9.** Example of localization outputs using MCL with 100 particles and the WGPPL model as perceptual model are shown in (a). Dashed lines indicate the robot's true path while the dots show MCLs predicted locations. Lighter dots represent the beginning of the test and darker ones the end. Corresponding localization errors for the same run, are shown in (b). Where the shaded area shows the time prior to convergence (tc = 114 s), and the dashed line shows its root mean square error (RMSE = 0.95 m). Cumulative density function (CDF) of the errors are shown in (c). Precision of the run at 80% (precisions at other percentages are also shown in the figure — p80 = 1.53 m, p90 = 1.95, p95 = 2.45 m for 80, 90 and 95% respectively).

### 6.1. Sensor update rate

Sensor update rate is particularly important to be discussed in our case due to the particularities of our sensor. For MCL, in general, having higher rates is desirable as it allows for faster convergence and more accurate results. The advantage of lower rates is the possibility of allowing the sensor to acquire more measurements, which can then be filtered and combined in order to obtain more accurate measurements. For several sensors in localization, measurements can be acquired fast and even a single measurement is precise enough, making the fusion of several measurements either possible even at high MCL rates or unnecessary.

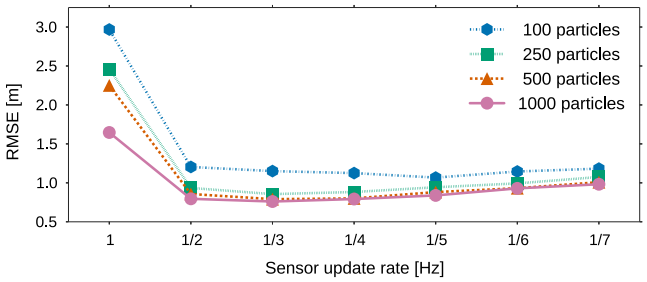


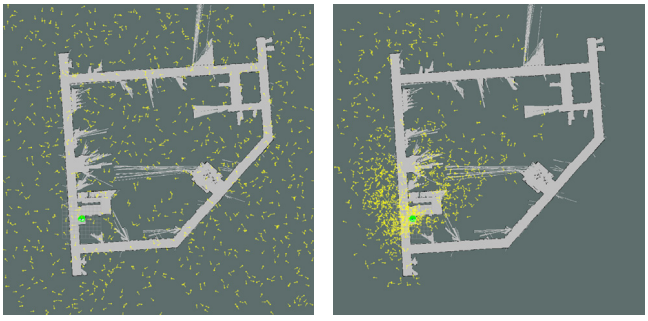

**Fig. 10.** Comparison of root mean square errors for different sensor sampling rates and different number of particles. Sampling rates of 1 Hz (highest sampling rate) having the worst overall performances, and sampling rates of 1/3 the best.

Therefore, it is common to choose MCL rate as the highest possible under computational power constraints. In our implementation, such constraints are not a major issue, as the time required for evaluating our models is relatively low. For example, only 0.053 s were required for evaluating an MCL with 1000 particles, using the same computer described in Section 5.1, which has two cores running at 2.3 GHz. Comfortably allowing sensor update rates of 10–15 Hz. However, for wireless sensors, it is not possible nor recommendable to use such high rates. The reasons are twofold: data acquisition is *slow* and considerably noisy. First, data acquisition is slow because signal strength measurements are acquired passively by listening to access points' beacons without actively interacting with them. Access points broadcast these beacons at fixed intervals set by them, typically of 100 ms. Second, measurements are noisy not only due to sensor noise but also due to the wireless signal propagation phenomenon itself. Specifically, due to its fast fading component. To reduce this noise, which would impair the accuracy of our localization system, it is preferable to use lower sensor update rates so the average of several measurements over time can be used instead of individual, noisy, measurements.

Fig. 10 shows localization accuracies when using several sensor update rates and different number of MCL particles. All results were obtained using sensor initialization (discussed in the following section) and WGPPL as perceptual model. Sensor sampling times were varied in multiples of 1 s as experimental data was originally sampled at 1 Hz. For each sensor update, the average value of all RSS measurements acquired during its sampling time was used. With the average number of RSS measurements per second being 5.7 in our dataset. This number depends on the frequency each access point broadcasts its beacon frames (once every 88 ms on average for our case) and the number of channels being surveyed (2 for our case). As it can be observed from the figure, the highest sensor update rate has the worst performance, opposite to most sensors. And overall, a sensor update rate of 1/3 Hz was found to have the best performance in terms of lower RMSE.

### 6.2. Initialization

Being a recursive implementation, it is necessary to provide a way to initialize the algorithm for time step 0. Whenever global localization is intended, it is common to initialize $Bel(\mathbf{p})$ using a uniform distribution, which results in particles randomly scatter all around the map. Sampling from this uniform distribution is a practical choice, as for most sensors it is not possible to sample the posterior distribution of sensor data; however, this is not the case for either WGP or WGPPL models. Nonetheless, given our initial problem assumptions (robot orientation does not affect wireless measurements) it is not possible to sample 3D poses — as our models do not have information regarding robots' orientation. Although it is not possible to sample 3D poses, the posteriors



(a) Uniform sampling. (b) Sensor sampling.

**Fig. 11.** Particle initialization can be performed uniformly at random (a) or from wireless models posteriors (b). For each case 1000 particles, represented by the yellow arrows, are shown. For recursive localization algorithms, such as MCL, high density of particles around the true location at initialization is desired.

**Table 3**
Comparison between random and sensor initialization.

| Random initialization | | | | |
|---|---|---|---|---|
| Number of particles | 100 | 250 | 500 | 1000 |
| Time for convergence [s] | 280 | 248 | 156 | 84 |
| Root mean square error [m] | 1.60 | 1.21 | 1.13 | 1.04 |
| Failures [%] | 36.5 | 5.5 | 0 | 0 |
| **Sensor initialization** | | | | |
| Number of particles | 100 | 250 | 500 | 1000 |
| Time for convergence [s] | 248 | 76 | 76 | 72 |
| Root mean square error [m] | 1.36 | 1.13 | 1.07 | 1.05 |
| Failures [%] | 3.5 | 0 | 0 | 0 |

provide valuable x-y locations, to take advantage of it we sample x-y locations from the model posterior while orientations are sampled uniformly at random. Fig. 11 shows an example of particle initialization using sensor sampling with respect to sampling from the uniform distribution.

We analyze the effect of these two initializations in terms of time for convergence, root mean square localization errors and convergence failure rate for different numbers of particles by running each parameter configuration 50 times. Fig. 12 shows localization errors and convergence time obtained for MCLs with 100, 250, 500 and 1000 particles for one dataset; while Table 3 summarizes the results obtained from all others. Results correspond to using WGPPL as MCL's perceptual model.

As it can be noticed from the table, for MCLs with high number of particles (500 and 1000) convergence speed is only slightly increased when using sensor sampling, while localization accuracy is similar throughout the experiment. Therefore, the usage of this initialization may not be appealing enough for such cases. For low number of particles (100 and 250), this scenario changes, as convergence speed noticeably increases, as well as localization accuracy before convergence. After convergence, if achieved, similar localization accuracy is obtained as it no longer depends on the initialization method but rather on the quality of the information provided by the observations and sensor models. Notably, for low number of particles, sensor initialization greatly reduces convergence failure, with only 3.5% failure against 36.5% for 100 particles and no observed failure for 250 particles against 5.5% for 250 particles. This is a crucial improvement, as the most sought property is convergence.

### 6.3. WGP vs WGPPL

In this section we continue the discussion presented in Section 5 regarding WGP and WGPPL by quantifying the localization accuracies of both models when used as MCL's perceptual likelihood.

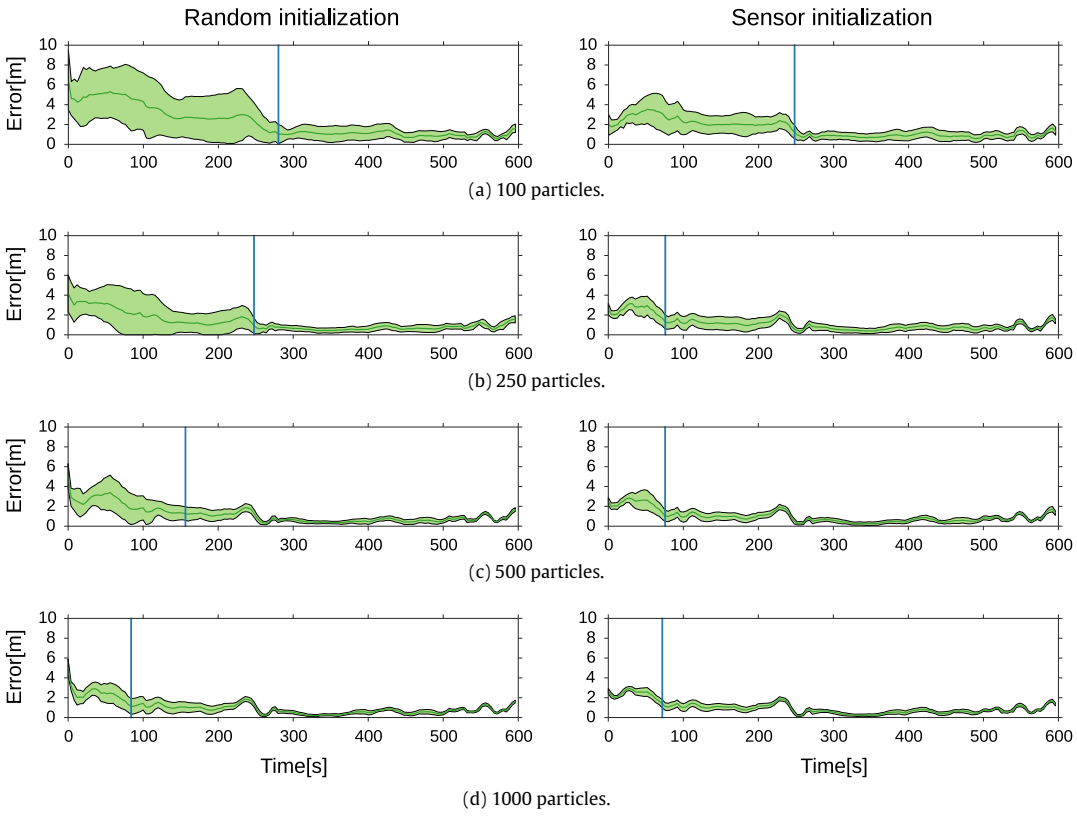



**Fig. 12.** Localization errors obtained when performing random and sensor initializations. Solid green lines show the means for all tests while the shaded areas ± one standard deviation. Blue lines indicate point of convergence, defined as the time when the mean plus one standard deviation of the localization errors stabilize under 2 m.

The metrics chosen for this analysis are all but those related to convergence, as both models allow for extremely fast convergence. Table 4 shows the root mean square errors for all datasets employed. The failure rate is also shown in parenthesis for the 100 and 250 particles cases, for the others the rate was consistently zero, so they were omitted.

From the table, it can be observed that for all datasets, our approach consistently outperformed the classic GP formulation. This difference was more noticeable with low number of particles but persisted even with high number of particles. Furthermore, the failure rate is greatly improved, with no failure for as low as 250 particles for all models with our approach — and for even 100 particles for all B3 datasets, while for the WGP formulation up to 44% of failures were obtained.

As can be observed from our testing, localization accuracy was similar for all tested scenarios, although the obvious discrepancies among them (such as the different number of access points, the layout of the environments, etc.). For all scenarios, Test1 showed the best performance, which was expected as models were trained using the same dataset used for them (although at a different sensor rate). Therefore, higher localization accuracy in these tests was expected. These tests will be used as reference when assessing the localization performance of the other two.

A major factor that affects signal strength measurements is the presence or absence of obstacles. For example, in our testing, doors were commonly found large objects whose position constantly changed (as they could either be opened or closed). The effect of these objects should have been present in all tests as all testing was performed during working hours without controlling nor monitoring the state of doors. However, localization degradation was minimum for overall, especially on tests 2 which had very similar localization accuracies with respect to tests 1. This leads us to believe that the presence or absence of these obstacles did not cause major degradations in localization accuracies.

**Table 4**
MCL RMSE and failure rate. Failure rates for 500 and 1000 particles was 0% for both models.

| B2 datasets | | | | |
|---|---|---|---|---|
| Particles | 100 | 250 | 500 | 1000 |
| Test1:WGP | 1.57 (16%) | 1.11 (0%) | 1.05 | 0.97 |
| Test1:WGPPL | 1.29 (4%) | 0.92 (0%) | 0.86 | 0.83 |
| Test2:WGP | 2.62 (28%) | 1.98 (0%) | 1.68 | 1.63 |
| Test2:WGPPL | 1.50 (4%) | 1.13 (0%) | 1.08 | 1.07 |
| Test3:WGP | 2.59 (40%) | 2.08 (0%) | 1.94 | 1.88 |
| Test3:WGPPL | 1.80 (0%) | 1.41 (0%) | 1.37 | 1.38 |
| B3 datasets | | | | |
| Test1:WGP | 1.40 (32%) | 1.06 (0%) | 0.94 | 0.89 |
| Test1:WGPPL | 1.01 (0%) | 0.83 (0%) | 0.81 | 0.77 |
| Test2:WGP | 1.94 (24%) | 1.26 (0%) | 1.17 | 1.09 |
| Test2:WGPPL | 1.38 (0%) | 1.20 (0%) | 1.05 | 1.02 |
| Test3:WGP | 1.72 (44%) | 1.59 (0%) | 1.53 | 1.48 |
| Test3:WGPPL | 1.58 (0%) | 1.45 (0%) | 1.41 | 1.40 |
| B14 datasets | | | | |
| Test1:WGP | 1.55 (36%) | 1.39 (20%) | 0.97 | 0.91 |
| Test1:WGPPL | 0.92 (4%) | 0.80 (0%) | 0.76 | 0.75 |
| Test2:WGP | 1.84 (32%) | 1.32 (24%) | 0.93 | 0.89 |
| Test2:WGPPL | 0.99 (4%) | 0.91 (0%) | 0.85 | 0.82 |
| Test3:WGP | 2.51 (32%) | 2.08 (0%) | 1.98 | 1.93 |
| Test3:WGPPL | 1.74 (16%) | 1.49 (0%) | 1.40 | 1.42 |

The highest degradation in performance was found in tests B3:Test3 and B14:Test3. Where, for MCLs with 1000 particles, localization errors increased by 0.65 m on average, when the average

localization error for the other tests was only 0.28 m. As can be noted from Table 1, the main difference between these tests and all others is the higher average speed of the robot that needed to be localized. Robot speeds for these tests were 0.30 and 0.35 m/s respectively, opposed to around 0.20 m/s for all other tests. This apparent drop in localization accuracy at higher robot speeds can be explained by the less frequent (with respect to distance) sensor updates. A possible way to allow our system to handle higher robot speeds is to change from passive signal acquisition to active acquisition. Active acquisition can be performed by sending probe requests, which are special frames sent by WNICs requesting information from all access points in its surroundings. Though, continuously probing networks may be considered intrusive.

Another possible cause for localization degradation is missing access points, as they would cause zero measurements when sensor models predict non-zero values. This was commonly observed during testing. However, we consider that as our localization system relies on several tens to few hundreds of access points, whose likelihoods are integrated, the overall effect of few faulty access points was considerably lessened. Furthermore, we explicitly added a point mass distribution at zero ($p_{zero}$) in our sensor model (Eq. (25) in order to compensate for these occurrences, as not measuring signals even when our sensor models predict non-zero values is not heavily penalized thanks to this point mass distribution. On the other hand, non-zero measurements when our sensor models predict zero values would have a much stronger effect on our system, as our models do not account for unexpected non-zero signals. This should only occur if access points are moved from their original location, which was assumed not to occur in our work (no constraints nor prior knowledge on access points locations is assumed; however, they are assumed to remain in the same location at all times). Although this was also not observed during testing, it is of considerable concern. A possible way to address this issue is to adaptively update training data and models to account for displaced access points. Readers interested in such approaches are referred to [29] where RSS measurements and their predicted locations were continuously added to a database, which was used to periodically update its models by computing RSS statistics, filtering abnormal RSS measurements, and computing new mappings, and [30] where training points were also annotated with the time when they were collected, as their approach employs this information to prioritize recent RSS measurements when making predictions. The main drawback of continuously increasing training datasets is the corresponding increase in computation required. To reduce this issue, works like [31] proposed fusing training points via clustering and [32] proposed eliminating redundant ones using some heuristics. As well as [13] which contains a more thorough review of several approaches like the previously mentioned.

Regarding localization precision at 80%, Table 5 shows the results obtained from all testing. As previously mentioned a localization precision at 80% of 1 m, means that 80% of the localization outputs are within 1 m of the true location.

Similarly to RMSE, our approach outperformed the classic formulation for localization precision at 80%, guaranteeing localization between 1 and around 2 m for all cases at 80% confidence.

### 6.4. Performance with respect to training data size

Finally, we address a common issue that affects the performance of fingerprinting techniques, such as ours: its strong correlation with training dataset quality and size. As previously mentioned, to avoid poor quality training data points, for every 10 acquired measurements, our system generates a single data point. Making the total number of data points in our training datasets for buildings 2, 3 and 14 equal to 121, 164 and 73, and the average distance between training points 1.82, 1.69 and 1.89 m, respectively. If this number of training data points is reduced, the localization performance of our system is expected to deteriorate. In order to quantify this effect, we ran simulations with varying training data sizes. We did this by using only one training data point for every 2, 3 and 4 data points available. Fig. 13 shows the localization accuracy obtained using an MCL with 1000 particles and our approach trained using these smaller datasets.

From the figure, we can observe that localization accuracy deteriorate when the training data size is reduced. As MCLs trained using the original training datasets had a localization error of 1.05 m on average, while those using the modified datasets had a localization error of 1.19, 1.42 and 1.63 m respectively. It is important to notice that even when using the smallest datasets there is only a 55% increase in localization errors on average (the results for these cases are labeled as 1/4 in the figure). This is remarkable considering that the smallest datasets we tested are only one-fourth of the original ones, having on average only 29.7 training points, with one training point every 7.24 m. This shows the ability of our approach to generalize location-wireless signal strength mappings even from scarce training data points.

**Table 5**
MCL localization precision at 80%.

| | | | | |
|---|---|---|---|---|
| B2 datasets | | | | |
| Particles | 100 | 250 | 500 | 1000 |
| Test1:WGP | 2.16 | 1.62 | 1.44 | 1.32 |
| Test1:WGPPL | 1.68 | 1.26 | 1.18 | 1.12 |
| Test2:WGP | 3.78 | 2.84 | 2.38 | 2.30 |
| Test2:WGPPL | 2.08 | 1.70 | 1.64 | 1.62 |
| Test3:WGP | 3.58 | 2.86 | 2.64 | 2.54 |
| Test3:WGPPL | 2.54 | 2.04 | 1.94 | 1.94 |
| B3 datasets | | | | |
| Test1:WGP | 2.00 | 1.48 | 1.36 | 1.28 |
| Test1:WGPPL | 1.44 | 1.18 | 1.16 | 1.08 |
| Test2:WGP | 2.50 | 1.74 | 1.56 | 1.46 |
| Test2:WGPPL | 2.02 | 1.60 | 1.32 | 1.32 |
| Test3:WGP | 2.34 | 2.18 | 2.04 | 1.94 |
| Test3:WGPPL | 2.12 | 1.90 | 1.86 | 1.86 |
| B14 datasets | | | | |
| Test1:WGP | 1.92 | 1.54 | 1.30 | 1.20 |
| Test1:WGPPL | 1.32 | 1.10 | 1.04 | 1.00 |
| Test2:WGP | 2.34 | 1.54 | 1.20 | 1.14 |
| Test2:WGPPL | 1.36 | 1.22 | 1.08 | 1.04 |
| Test3:WGP | 3.80 | 2.86 | 2.68 | 2.56 |
| Test3:WGPPL | 2.34 | 2.12 | 2.02 | 2.04 |

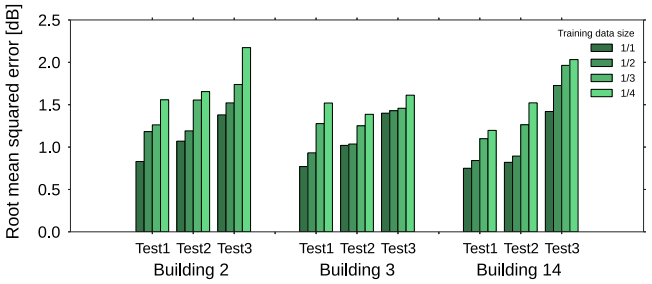



**Fig. 13.** Localization root mean square error performance using different training dataset sizes. Results correspond to localization accuracies obtained using an MCL with 1000 particles and our models trained using our original training datasets (labeled as 1/1), half of the training data points (labeled as 1/2), one third (labeled as 1/3) and one fourth (labeled as 1/4). As expected, MCLs using smaller training data sizes show a degradation in localization performance; however, this degradation is small when compared to the reduction in their training data sizes.

## 7. Conclusions

In this work we have presented a novel modeling approach for generating precise and accurate wireless signal strength mappings by using Gaussian processes and path loss models. These mappings are important for robot localization as they can be used to compute the likelihood of locations conditioned to sensor data — core component of Bayes based localization algorithms.

Our testing using several datasets in three different indoor environments has also demonstrated that signal strength mappings generated by our approach consistently output better likelihood distributions than previous GP-based approaches. These distributions output lower probabilities for points far from the robot's true position and distributions peaked closer to the robot's true position — both desirable features. Interestingly, the main enhancement on the likelihood functions was not due to an increased accuracy in RSS predictions by our models, but rather by their better handling of variances. Moreover, using the likelihoods as the perceptual model of an MCL algorithm, we have also demonstrated that these likelihoods enhance localization accuracy, especially when a low number of particles are used. Using our proposed approach we have successfully performed localization with as little as 100 particles for several datasets and 250 particles for all.

Our future work will focus on the extension to other sensors and/or applications of the concept we have developed in this work — that of using the physics of the phenomena being modeled not only for the generation of predictions but also for bounding the predicted variances. As well as the adaptation and fashioning of the methodology developed here for localization systems using exclusively wireless sensors, for its use in conjunction with other sensors commonly used, such as rangefinders and cameras.

## Acknowledgment

This work was in part funded by ImPACT Program of Council for Science, Technology and Innovation (grant no. 2015-PM07-02-01) (Cabinet Office, Government of Japan).

## References

[1] D. Koks, Numerical calculations for passive geolocation scenarios, DSTO-RR-0319, Defence Science and Technology Organisation Edinburgh (Australia) Electronic Warfare and Radar Division, 2007.
[2] L. Brás, N.B. Carvalho, P. Pinho, L. Kulas, K. Nyka, A review of antennas for indoor positioning systems, Int. J. Antennas Propag. (2012) 1–14. http://dx.doi.org/10.1155/2012/953269.
[3] R. Schmidt, Multiple emitter location and signal parameter estimation, IEEE Trans. Antennas and Propagation 34 (3) (1986) 276–280.
[4] M. Kotaru, K. Joshi, D. Bharadia, S. Katti, Spotfi: Decimeter level localization using wifi, ACM SIGCOMM Comput. Commun. Rev. 45 (4) (2015) 269–282.
[5] X. Wang, Z. Wang, B. O Dea, A TOA-based location algorithm reducing the errors due to non-line-of-sight (NLOS) propagation, IEEE Trans. Veh. Technol. 52 (1) (2003) 112–116.
[6] D. Jeong, K. Lee, Directional rss-based localization for multi-robot applications, in: Proceedings of the 12th WSEAS International Conference on Signal Processing, Robotics, and Automation, 2013.
[7] L. Bruno, P. Robertson, Observability of path loss parameters in WLAN-based simultaneous localization and mapping, in: International Conference on Indoor Positioning and Indoor Navigation, 2013, pp. 1–10.
[8] K. Benkič, M. Malajner, P. Planinšič, Ž. Čučej, Using RSSI value for distance estimation in wireless sensor networks based on ZigBee, in: 15th International Conference on Systems, signals and image processing, 2008, pp. 303–306.
[9] E. Elnahrawy, X. Li, R.P. Martin, The limits of localization using signal strength: A comparative study, in: First Annual IEEE Communications Society Conference on Sensor and Ad Hoc Communications and Networks, 2004, pp. 406–414.
[10] F. Lemic, A. Behboodi, V. Handziski, A. Wolisz, Experimental decomposition of the performance of fingerprinting-based localization algorithms, in: International Conference on Indoor Positioning and Indoor Navigation, 2014, pp. 355–364.
[11] H. Liu, H. Darabi, P. Banerjee, J. Liu, Survey of wireless indoor positioning techniques and systems, IEEE Trans. Syst. Man Cybern. C 37 (6) (2007) 1067–1080.
[12] S. He, S.-H.G. Chan, Wi-Fi fingerprint-based indoor positioning: Recent advances and comparisons, IEEE Commun. Surveys Tutor. 18 (1) (2016) 466–490.
[13] A. Khalajmehrabadi, N. Gatsis, D. Akopian, Modern WLAN fingerprinting indoor positioning methods and deployment challenges, IEEE Commun. Surveys Tutor. (2017).
[14] M. Brunato, R. Battiti, Statistical learning theory for location fingerprinting in wireless LANs, Comput. Netw. 47 (6) (2005) 825–845.
[15] L.B. Del Mundo, R.L.D. Ansay, C.A. M. Festin, R.M. Ocampo, A comparison of wireless fidelity (wi-fi) fingerprinting techniques, in: International Conference on Convergence, 2011, pp. 20–25.
[16] B. Benjamin, G. Erinc, S. Carpin, Real-time WiFi localization of heterogeneous robot teams using an online random forest, Auton. Robots 39 (2) (2015) 155–167.
[17] J.S. Gutmann, E. Eade, P. Fong, M.E. Munich, Vector field SLAM - Localization by learning the spatial variation of continuous signals, IEEE Trans. Robot. 28 (3) (2012) 650–667.
[18] J. Biswas, M. Veloso, WiFi localization and navigation for autonomous indoor mobile robots, in: IEEE International Conference on Robotics and Automation, 2010, pp. 4379–4384.
[19] A. Schwaighofer, M. Grigoras, V. Tresp, C. Hoffmann, GPPS: A Gaussian process positioning system for cellular networks, in: Advances in Neural Information Processing Systems 16, 2003, pp. 579–586.
[20] B. Ferris, D. Haehnel, D. Fox, Gaussian processes for signal strength-based location estimation, in: Proceedings of Robotics Science and Systems, 2006, pp. 1–8.
[21] R. Miyagusuku, A. Yamashita, H. Asama, Gaussian processes with input-dependent noise variance for wireless signal strength-based localization, in: IEEE International Symposium on Safety, Security, and Rescue Robotics, 2015, pp. 1–6.
[22] M. Jirků, V. Kubelka, M. Reinstein, WiFi localization in 3D, in: IEEE/RSJ International Conference on Intelligent Robots and Systems, 2016, pp. 4551–4557.
[23] B. Ferris, D. Fox, N.D. Lawrence, WiFi-SLAM using Gaussian process latent variable models, in: International Joint Conference on Artificial Intelligence, Vol. 7, 2007, pp. 2480–2485.
[24] R. Miyagusuku, A. Yamashita, H. Asama, Improving gaussian processes based mapping of wireless signals using path loss models, in: IEEE/RSJ International Conference on Intelligent Robots and Systems, 2016, pp. 4610–4615.
[25] A. Goldsmith, Wireless Communications, Cambridge University Press, Cambridge, United Kingdom, 2005.
[26] C.K. Williams, C.E. Rasmussen, Gaussian Processes for Machine Learning, MIT press, Cambridge, MA, 2006.
[27] S. Thrun, W. Burgard, D. Fox, Probabilistic Robotics, MIT press, Cambridge, MA, 2005.
[28] H. Zou, M. Jin, H. Jiang, L. Xie, C.J. Spanos, WinIPS: WiFi-based non-intrusive indoor positioning system with online radio map construction and adaptation, IEEE Trans. Wirel. Commun. 16 (12) (2017) 8118–8130.
[29] J.Y. Zhu, J. Xu, A.X. Zheng, J. He, C. Wu, V.O. Li, WIFI fingerprinting indoor localization system based on spatio-temporal (ST) metrics, in: International Conference on Indoor Positioning and Indoor Navigation, 2014, pp. 611–614.
[30] V. Radu, M.K. Marina, Himloc: indoor smartphone localization via activity aware pedestrian dead reckoning with selective crowdsourced wifi fingerprinting, in: International Conference on Indoor Positioning and Indoor Navigation, 2013, pp. 1–10.
[31] Y. Kim, Y. Chon, M. Ji, S. Park, H. Cha, Scalable and consistent radio map management scheme for participatory sensing-based wi-fi fingerprinting, in: Proceedings of the 3rd International Workshop on Mobile Sensing, 2013, pp. 14–18.
[32] S. Eisa, J. Peixoto, F. Meneses, A. Moreira, Removing useless aps and fingerprints from wifi indoor positioning radio maps, in: International Conference on Indoor Positioning and Indoor Navigation, 2013, pp. 1–7.

**Renato Miyagusuku** received his B.S. degree in Mechatronics from the Department of Mechanical Engineering, National University of Engineering, Peru, in 2011. Later he received his M.S. degree from the Department of Precision Engineering, the University of Tokyo, Japan in 2015; and is currently pursuing a Ph.D. degree at the same university. His main interest is machine learning applied to robotics, in particular, wireless signal strength based robot localization.

**Atsushi Yamashita** received his B.E., M.E., and Ph.D. degrees from the Department of Precision Engineering, the University of Tokyo, Japan, in 1996, 1998, and 2001, respectively. From 1998 to 2001, he was a Junior Research Associate in the RIKEN (Institute of Physical and Chemical Research). From 2001 to 2008, he was an Assistant Professor of Shizuoka University. From 2006 to 2007, he was a Visiting Associate of California Institute of Technology. From 2008 to 2011, he was an Associate Professor of Shizuoka University. From 2011, he is an Associate Professor in the Department of Precision Engineering, the University of Tokyo. His research interests include robot vision, image processing, and motion planning. He is a member of ACM, IEEE, JSPE, RSJ, IEICE, JSAE, JSME, IEEJ, IPSJ, ITE, SICE and Society for Serviceology.

**Hajime Asama** received his M.S., and Dr.Eng degrees from the University of Tokyo, in 1984 and 1989, respectively. He worked at RIKEN in Japan from 1986 to 2002 as a Research Associate, Research Scientist, and Senior Research Scientist. He became a professor of RACE (Research into Artifacts, Center for Engineering), the University of Tokyo in 2002, and a professor of School of Engineering, the University of Tokyo since 2009. He received the RSJ Distinguished Service Award in 2013. He was the vice-president of RSJ in 2011–2012, and an AdCom member of IEEE Robotics and Automation Society in 2007–2009. Currently, he is the president-elect of IFAC since 2017, the president of the International Society for Intelligent Autonomous Systems since 2014, and an associate editor of Control Engineering Practice, Journal of Robotics and Autonomous Systems, Journal of Field Robotics, etc. He is a council member of the Science Council of Japan since 2017. He is a Fellow of IEEE, JSME, and RSJ. His research interests include service robotics, distributed autonomous robotic systems, embodied-brain systems, human interfaces for teleoperated robots, disaster response robots, rehabilitation robots.