UNIVERSITY OF OULU

FACULTY OF INFORMATION TECHNOLOGY AND ELECTRICAL ENGINEERING
DEGREE PROGRAMME IN WIRELESS COMMUNICATIONS ENGINEERING

MASTER'S THESIS

BAYESIAN GRAPHICAL MODELS FOR INDOOR LOCALIZATION IN MTC DEPLOYMENT SCENARIOS

Author	Henrique Hilleshein
Supervisor	Assistant Prof. Carlos Morais de Lima
Second Examiner	Assistant Prof. Hirley Alves

February 2021

Hilleshein H. (2021) Bayesian Graphical Models for Indoor Localization in MTC Deployment Scenarios. University of Oulu, Degree Programme in Wireless Communications Engineering, 44 p.

ABSTRACT

Herein, we propose and assess an iterative Bayesian-based indoor localization system to estimate the position of a target device. We describe the Bayesian network and then build graphical models for various measurement metrics, namely Received Signal Strength (RSS), Time Difference of Arrival (TDOA), and Angle of Arrival (AOA) which are collected by the distributed receivers in the network area. The estimations are carried out by Markov chain Monte Carlo (MCMC) methods which approximates the target's position using the Bayesian network model and measurements collected by the receivers. We employ an iterative method by using previous estimations of the target's position as prior knowledge to improve the accuracy of the subsequent estimations, where the prior knowledge is used as the prior distributions of our Bayesian model. In our results, we observe that the proposed iterative localization system improves the performance of the Bayesian TDOA-based localization system by increasing the respective estimate accuracy. Furthermore, we show that the number of measurements collected by the receivers and the selected prior distribution also affect the performance of the proposed iterative mechanism. In fact, the number of measurements increases the accuracy of the mechanism, while its benefit diminishes with more iterations as the mechanism progresses. Regarding the prior distribution, we show that it can lead to good or bad estimations of the target's position, and therefore, needs to be carefully chosen considering the measurement metric and the mobility of the target node.

Keywords: DAG, Bayesian networks, tracking, IPS, MCMC, TDOA, RSS, AOA.

TABLE OF CONTENTS

FOREWORD

This Master thesis concludes my cycle in the Master's degree in Wireless Communications Engineering at the University of Oulu.

Herein, I present the research I carried out during my Master's studies. It is a challenging topic that required knowledge that I did not have previously. However, with the courses from the Master's program, self-studying, and the support of my supervisor I was able to write this document.

I would like to specially thank my supervisor Assistant Prof. Carlos Morais de Lima and the Assistant Prof. Hirley Alves. Carlos thoroughly supported me in my research. Hirley introduced the Master's programme and the Centre of Wireless Communications to me. They taught me different things and helped me in my scientific paper publications. Also, I would like to thank my family and everyone that helped me in my formation from Brazil and Finland.

Oulu, 3rd February, 2021

Henrique Hilleshein

LIST OF ABBREVIATIONS AND SYMBOLS

AP	Access Point
AOA	Angle Of Arrival
CDF	Cumulative Distribution Function
DAG	Direct Acyclic Graph
eMBB	extreme Mobile Broadband
FG	Full Gaussian
FP	Full Posterior
GPS	Global Positioning Systems
HMC	Hamiltonian/Hybrid Monte Carlo
IIoT	ndustrial Internet of Things
IPS	Indoor Positioning System
KDE	Kernel Density Estimation
KL	Kullback–Leibler
MCMC	Markov Chain Monte Carlo
MD	Mixture Distribution
MH	Metropolis-Hasting
mMIMO	Multiple-Input Multiple-Output
MTC	Machine-Type Communications
mMTC	massive Machine-Type Communications
NOMA	Non-Orthogonal Multiple Access
NUTS	No-U-Turn Sampler
RMSE	Root Mean Square Error
RSS	Received Signal Strength
RSSI	Received Signal Strength Indicator
RV	Random Variable
PGM	Probabilistic graphical Model
TDOA	Time Difference of Arrival
TOF	Time of Flight
URLLC	Ultra-Reliable Low-Latency Communication

1 INTRODUCTION

Recently, the number of devices with Machine-type Communications (MTC) devices has grown exponentially and these devices are used for myriads of diverse applications with different requirements of latency, reliability, throughput and security [1]. In such deployment scenarios, wired connection is often not possible or affordable due to their locations or applications, therefore, we alternatively resort to wireless communications.

In the 5th generation cellular networks, the mobile network will be driven to efficiently serve such MTC devices while considering many applications that do not even exist yet [2]. Actually, there is a paradigm shift because previously the mobile network was only oriented to provide human-centric high quality services for personal devices such as smartphones and tablets [3]. Conversely, the MTC devices have a different data traffic pattern which commonly uses short packets and is also asymmetric concentrating on the uplink direction [4]. Moreover, MTC devices have energy constraints which require allocating resources of the network considering the energy efficiency requirements [5]. Therefore, the cellular network not only needs to handle such data load with strict requirements but also maintain the present-day data traffic and considering the energy efficiency of the network. In fact, such wireless network is challenging to implement, and we can devise three essential types of 5G communications to comply with such requirements: critical MTC, massive Machine-type Communications (mMTC), and extreme Mobile Broadband (eMBB) [2].

In mMTC, the wireless network needs to provide services to a huge number of end low-complexity devices [6], while in critical MTC, also known as Ultra-Reliable Low-Latency Communication (URLLC), the applications require communications with very low latency and high reliability [7]. For eMBB, the network has higher data-rate and coverage than the provided by the 4G [2]. Due to these requirements, we need to create new technologies and algorithms to cope with these new challenges such as millimeter wave [8], massive Multiple-Input Multiple-Output (mMIMO) technique [9], and grant-free Non-Orthogonal Multiple Access (NOMA) [10].

In such scenarios, the position of the MTC devices can be employed, for example, to devise more efficient position-aided radio resource management procedures. By using this information, we can indeed allocate the resources of our network more efficiently as shown in [11], and provide more secure service as addressed in [12]. Moreover, the positioning system is considered a key enabler for many applications [13] such as warehouse management [14], beamforming antennas [15], facility management [16], and automated guided vehicle for smart factories [17]. In fact, positioning systems are important for the Industrial Internet of Things (IIoT) because applications of processes in various verticals sectors including but not limited to manufacturing, logistics, transportation, and mining can require position information of devices [18]. Therefore, we need accurate positioning systems to support 5G location-based services [19].

Nowadays, there are commercial positioning systems that can accurately localize devices in outdoor environments such as the Global Positioning System (GPS) which do not work properly in urban canyon scenarios and fail completely in indoor deployments [20]. Therefore, we should develop efficient Indoor Positioning Systems (IPS) that can localize targets with high accuracy [21]. As MTC devices usually have cost constraints, it becomes interesting to use the same wireless connection to transfer both data and signaling to locate devices as it may reduce hardware costs. In this case, we can use features of wave propagation to help us in estimating a target's position, where the 5G radio features play a strong role increasing the accuracy of IPSs due to its use of high carrier frequency [22], high number of antenna elements [23], and smaller cells [24], for example. We can use distinct measurement metrics to estimate a target's position such as Angle of Arrival (AOA) [25, 26], Received Signal Strength Index (RSSI) [27, 28] and Time of Flight (TOF) [29, 30]. RSSI is a built-in measurement that is usually available in different radio access technologies which makes it a cheaper option [21], while TDOA and AOA techniques give more accurate estimation than RSSI but require extra hardware in the APs [31].

The techniques used for IPS usually require a big dataset of measurements and/or learning of the environment before using the system, and in many cases, these requirements can be prohibitive [21]. Moreover, as the IPS is dynamic it would require routine calibration which could be costly. Author in [21], tackled this problem by proposing a RSSI-based IPS mechanism where we use Bayesian networks and Markov chain Monte Carlo (MCMC) to estimate targets' position. This mechanism does not need any prior knowledge or preliminary measurements regarding to the environment [21]. We use a Bayesian network to describe through graphs all the Random Variables (RVs) of the statistical model, their interdependence and how they are related to our desired event [32]. The quantity of interest or our desired event is the measurement metric that we are using to estimate the target's position. This method employs the Bayes' theorem which allows us to use our most updated knowledge regarding the RVs and our assumptions about the measurement metrics to estimate the target's position [33]. Also, different from a point estimate of the frequentist approach, the Bayesian approach gives the probability density estimation or quantification of uncertainty of the RVs [33] which contains more information regarding the target's position than a point estimate. After designing the Bayesian graphical model, we need to use an estimator to carry out estimations of the respective model, and MCMC is the most used method to make inferences of Bayesian networks [34]. The MCMC method is a sampling technique used to sample arbitrary unknown distributions using concepts of Markov chain and Monte Carlo [34]. In a Bayesian network, the MCMC algorithm samples all the RVs in the graphical statistical model based on our observations regarding the experiment of interest and respective assumptions about the RVs [33]. The samples of a RV are used to move between states of a Markov chain, where its equilibrium distribution is the probability density estimation of the respective RV [33]. In other words, we estimate the target's position with on-the-fly measurements together with our assumptions about radio channel propagation and network infrastructure.

1.1 Related works

Authors in [35], extended the Bayesian-based localization mechanism from [21] by considering a heterogeneous network where APs carry out different measurements metrics, namely RSSI and Time of Arrival (TOA) making it a IPS with hybrid metrics. At the same time, they combined previous and current measurements for the target's position estimation. However, the corresponding estimations still exhibit error over a meter which is too big for indoor applications that require maximum error of a few centimeters. In [36], we propose an iterative method which uses the current estimation of a target's position to improve the position estimate at subsequent iterations of the RSS-based IPS. As aforementioned, this mechanism uses Bayes' rule to estimate the target's position as well as our assumptions or prior beliefs regarding the RVs in our statistical model. After estimating the RVs, we then use such updated knowledge about variates in the model in the subsequent estimations. In fact, these assumptions correspond to the prior distribution of the Bayes' theorem, while the updated knowledge represents the posterior distribution [33]. Therefore, we repeatedly estimate and update the prior distributions so as to obtain a more accurate and reliable Bayesian-based IPS. It is worth noticing that the choice of prior distribution affects the estimation and in [37], we exploit this same mechanism while using different prior distributions to confirm their effects on the performance of the MCMC algorithm as previously addressed in [33].

1.2 Contributions

Herein, we continue working with our proposed iterative mechanism in [36, 37], while detailing how the Bayesian networks and the MCMC sampling methods work. In fact, the main contributions of this work are summarized next:

- investigate the MCMC sampling algorithms and compare their performances. The Hamiltonian Monte Carlo (HMC) algorithm outperforms the Metropolis–Hastings (MH) algorithm and the Gibbs sampling due to their random walk behaviour [33, 38, 39, 40], while No-U-Turn Sampler (NUTS) algorithm which extends the HMC is more efficient than a well-tuned HMC algorithm according to Authors in [41];
- further study the effects of the selected prior distribution to carry out the estimations, while we also discuss how the number of measurements affects the performance of the proposed mechanism;
- employ the proposed mechanism for tracking of moving targets where we learn that the choice of the prior distribution is even more sensitive;
- build the Bayesian graphical models for TDOA, AOA, and RSSI localization systems.

All in all, we show the proposed iterative mechanism improves the performance of a Bayesian TDOA-based IPS method, but the choice of the prior distributions can be tricky depending on the used measurement metric and whether the target is moving or not. Also, a higher number of measurements per iteration/estimation enhances the accuracy of the target's position estimation but its effects reduce through the iterations.

The reminder of this paper is organized as follows. Section 2 presents the fundamentals of Bayesian networks and MCMC sampling algorithm methods. Afterwards, we introduce our system model for AOA, RSSI, and TDOA in Section 3, while demonstrate the implementation of our proposed mechanism for RSSI localization system. In Section 4, we demonstrate and assess our results in terms of Root Mean Square Error (RMSE), Cumulative Distribution Function (CDF) and Kernel Density Estimation (KDE). Lastly, we draw our conlusions and final remarks in Section 5.

2 BAYESIAN INFERENCE AND SAMPLING METHODS

In this section, we first introduce the Bayesian theory and underlying concepts, then explain how to create statistical models using this framework and how to use it to infer quantities of interest. To begin with, the Bayesian network concept is explained, subsequently, we describe the mathematical problem to be solved and how we can make predictions using MCMC sampling methods. Thereafter, an iterative estimation mechanism is devised using the aforementioned statistical framework.

2.1 Bayesian Network

A Bayesian network corresponds to a Probabilistic Graphical Model (PGM) which is a statistical model to graphically represent RVs and their interdependence [33, 42]. It is advantageous to use diagrams to represent probability distributions because we can visualize the probabilistic model via graphs. Moreover, we have better insights into the properties of the statistical model while inspecting the diagram, and we can manipulate complex computations through graphs [33]. Therefore, we visualize and describe an arbitrary statistical model using graphs to obtain augmented analyses of its joint distribution.

PGMs are composed of vertices and edges. The vertices represent the RVs, while the edges describes the relation between the RVs [33]. The Bayesian networks are also known as Directed Acyclic Graphs (DAGs) where loops in the graph are not allowed [33]. It means that the graph does not have paths which leads a vertix/RV to itself as in the Fig. 1 [42]. The directed edges of the graph represent the interdependence of the RVs. For better understanding, we will denote the RVs as children or parents depending on their relationship. The RV at the head side of an edge is a child of the RV at the tail side, while the RV at the tail side is a parent. In a Bayesian network, it is assumed that the children are conditionally dependent on their parents while RVs not direct linked with an edge are considered conditionally independent [43]. Therefore, a RV is conditionally independent on all RVs in the graph aside from its parents.

We use this concept of children and parents interdependence to describe the joint distribution of a Bayesian network. The joint distribution of a generic Bayesian network can be denoted by [43]

$$f(V) = \prod_{v \in V} f(v|\,\mathrm{pa}[v]), \tag{1}$$

where V is the set of all RVs in the statistical model, while $\mathrm{pa}[v]$ is the parents of the RV v. Let's study the graph in the Fig. 1 to exemplify a Bayesian network. As aforementioned, the children are just conditionally dependent on their parents. Therefore, the RV D is conditionally dependent on B and C while it is conditionally independent on A, and C is conditionally dependent on A but conditionally independent from the other RVs in the graph. The RVs A and B are conditionally independent on all RVs in the graph. Taking the relationship between the RVs into account, we can describe their joint distribution as

$$f(A, B, C, D) = f(A)f(B)f(C|A)f(D|B, C), \tag{2}$$

where the conditional probabilities are in accordance with the relation of the RVs depicted in Fig. 1. Indeed, we need to find all the conditional distributions to be able to find

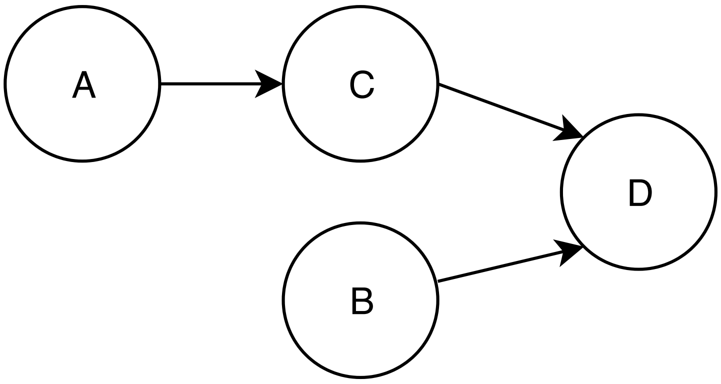

Figure 1. DAG Example. RV D is conditionally dependent on B and C, but it is conditionally independent on A.

this joint distribution. We can solve the conditional distributions by applying the Bayes' theorem [33].

The Bayes' theorem allows us to build an estimator, while incorporates our prior beliefs about the RVs related to the event of interest. Such prior beliefs are based on our knowledge about the RVs before observing any data related to our experiment [34]. Moreover, after observing actual data of an event, we can make inferences about it and verify if our prior beliefs regarding the event reflects the real world [34]. It means that after making inferences, we can update our prior beliefs about the event and use it to have more accurate estimations in the future. The Bayes' theorem is formulated as follows [33]

$$f(\mathcal{H}|\mathcal{D}) = \frac{f(\mathcal{D}|\mathcal{H})f(\mathcal{H})}{f(\mathcal{D})}, \tag{3}$$

where $\mathcal{D}$ is the observed data while $\mathcal{H}$ is our hypothesis/prior beliefs. Moreover, $f(\mathcal{H})$ is the prior distribution where its parameters represent all our knowledge regarding the desired event before observing any data $\mathcal{D}$. $f(\mathcal{D}|\mathcal{H})$ is the likelihood which represents the correctness of our hypothesis regarding the desired event when compared to the observed data $\mathcal{D}$. $f(\mathcal{D})$ is the normalization factor.

This approach was used in IPSs studies [21, 35, 44, 36] where we can use our knowledge of wave propagation to describe RVs and their interdependence related to our desired event. The event of interest could be one these measurement metrics RSSI, AOA and TDOA, for example. Based on one of these metrics, we can describe all we know regarding the RVs that contribute to the chosen metric and their interdependence through graphs. Indeed, the coordinates of the target are part of the RVs that contribute to this event. When we make observations of our desired event, we can estimate all the RVs present in its statistical model based on our prior beliefs of the RVs. In other words, once we make inference of this statistical model, we have the estimation of all RVs in this statistical model which includes the estimated coordinates of the target. The estimated coordinates are our updated beliefs/assumptions regarding the target's position, and we can use these new assumptions to improve the accuracy of the subsequent estimations of the target's position.

We can find the joint distribution analytically or numerically. The analytical method gives an exact solution, but it can be challenging to solve it depending on the number of RVs and conditional probabilities. Moreover, it is common to not have closed form

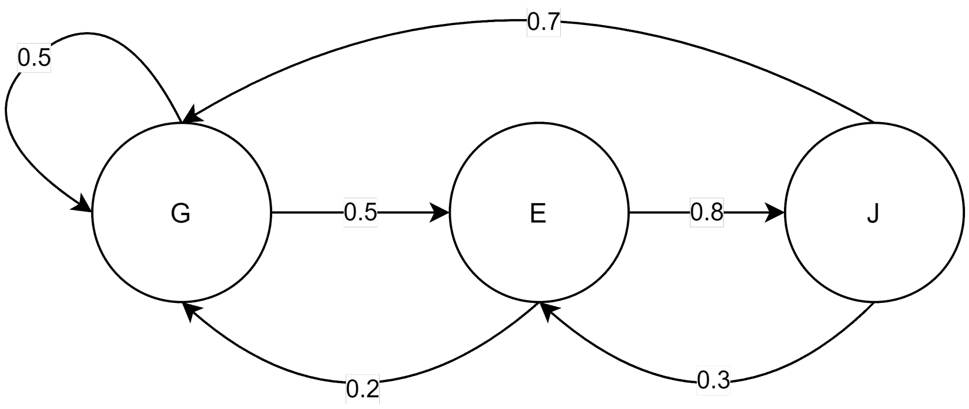

Figure 2. Markov chain example.

solution [33, 34]. Therefore, we decided to use numerical approximation solutions to find results for the statistical model of our positioning system, and the MCMC sampling method is the most used tool to find numerical approximations of Bayesian network models [34].

2.2 Markov Chain Monte Carlo methods

MCMC is a computational approach that uses Markov chain and Monte Carlo concepts to sample desired unknown probabilistic distributions [45]. A Monte Carlo method samples probability distributions by running the same experiment several times to generate numerical quantities of interest which allow us to understand the behavior of an observed event [46]. Due to the Law of the Large Number, if the number of samples is large enough, the outcome of the quantities should approximately converge to the exact correct quantities which would be found by analytical methods. However, many times there are no analytical form solutions, and numerical results are the only option [33].

Markov chain is a stochastic model that represents the set of possible states that an observed event can be while describing the probability of such event to change its current state to another [47]. It is a memoryless method where the next state is just conditionally dependent on the current state [33]. The Fig. 2 is an example of Markov chain. The G, E and J nodes are states and the edges represent the possible transactions between the states. It means that if the current state of the observed event is state G, the next state will be G or E with the same probability. In case the current state is J, the next state is going to be G with 70% of probability or E with 30% of probability. As we can see, the states prior the current state do not affect the transaction to the next state. The change of state in the chain is denoted step where given enough number of steps we reach the equilibrium distribution of the Markov chain. The equilibrium distribution gives the probability of the event to be in any determined state of the chain after a large number of steps independently which was the used initial state [33]. The equilibrium distribution is an important property that we use in MCMC.

The MCMC sampling algorithms create a collection of Markov chains with arbitrarily chosen initial value/state where the transaction between states happens according to a sampling algorithm that tries to wander through the chain to find the states with higher probability [33]. The random samples obtained by the Monte Carlo method are the states of the chain itself, and the more the algorithm pass through a specific state, the higher is the probability of that state [33]. When the MCMC algorithm finishes exploring the sampling space, we have the equilibrium distribution of the Markov chains created by the samples. This equilibrium distribution is the posterior distribution that we are aiming for in (3) [33].

In a Bayesian network, the MCMC sampling algorithm start with some initial values, and then it samples all the RVs in the graph to simulate them based on observations of the desired event and our prior distributions [21]. The observations are distributed according to the joint distribution, and therefore, the MCMC samples every single RV and its respective conditional distributions trying to find the combination of distributions that results in the distribution in which the observations are distributed [33]. In other words, the observations have embedded the information regarding all the RVs in the model by providing the unnormalized joint distribution. The MCMC method exploits this information to estimate all the RVs through a sampling process that depends on the used algorithm, for example, the Metropolis-Hastings (MH). The time the MCMC algorithm takes to find the equilibrium distribution depends on the chosen prior distribution, step size of the sampling algorithm and the sampling algorithm itself as we will explain next.

We choose the prior distributions arbitrarily and they represent all we know about the RVs before observing any data of the event of interest. This distribution affects the accuracy and performance of the sampling algorithm depending on how our prior beliefs reflect the real world [33]. The closer or similar this distribution is to the posterior distribution, the better our MCMC method will converge to the approximated posterior distribution [34]. Studies regarding the similarity of two distributions are out of the scope of this work. However, the similarity of a prior distribution to the actual posterior distribution can intuitively be described using Kullback–Leibler (KL) divergence as defined in [48].

As aforementioned, the next state of the Markov chain depends only on the current state where the states themselves are samples of the MCMC sampling algorithm. Therefore, the sampling algorithm samples the next sample based on last accepted sample. However, the algorithm should consider the distance between the current sample and the next one. This distance is the step size of the sampling algorithm which has a strong influence in its performance [33]. Samples drawn with a large step size have lower correlation with the current state but reduces the probability of acceptance of the sample. On the other hand, small step size gives high probability of acceptance of the sample but causes strong correlation between the new sample and the current one. Therefore, too large step size results in many rejected samples, and a too small step size requires more samples to cover the whole sampling space. Both situations cause slow Markov chain which is a slow convergence to the equilibrium distribution [33].

The sampling algorithm used by the MCMC method also affects strongly the convergence time of the Markov chain to the equilibrium distribution. The common samplings algorithms are the Metropolis–Hastings (MH), Gibbs sampling, Hamiltonian/Hybrid Monte Carlo (HMC) [33] and No-U-Turn Sampler (NUTS) [41]. These algorithms generate the samples to create and to give steps through the states of the Markov chain. We will explain the MH further because it gives the intuition regarding MCMC sampling algorithms and because it is the base for all the other algorithms. This algorithm was first introduced in [49] where the author proposed a Monte Carlo method that was more efficient in sampling high dimensional probability distributions than the other methods available at that time by using Markov chains. He used Markovian concepts and the Metropolis algorithm where the drawn samples candidates were just conditionally dependent on the last accepted sample. Therefore, a sample candidate z^* is sampled from an arbitrary proposal distribution $q(z^*|z^{(\tau)})$ in which $z^{(\tau)}$ is our last accepted sample or the current state of the chain. In [33], the probability of the sample candidate z^* being accepted is defined as

$$A_k(z^*, z^{(\tau)}) = \min\Big(1, \frac{\tilde{p}(z^*)q(z^{(\tau)}|z^*)}{\tilde{p}(z^{(\tau)})q(z^*|z^{(\tau)})}\Big), \tag{4}$$

where $\tilde{p}(z^*)$ and $\tilde{p}(z^{(\tau)})$ are the likelihood of the unnormalized desired distribution at z^* and $z^{(\tau)}$, respectively. The unnormilized desired distribution is obtained with observations that were draw from desired distribution. $q(z^*|z^{(\tau)})$ is the distribution which we take our sample candidates from, and therefore, the step size of our sampling algorithm will depend on the variance of this distribution. It means that if the variance of the proposal distribution is too large or too small, we have a slow Markov chain. If we consider a symmetrical proposal distribution $q(z^*|z^{(\tau)})$ such as a Gaussian distribution, then $q(z^{(\tau)}|z^*) = q(z^*|z^{(\tau)})$ as the probability of the transaction from the current state to the sample candidate is the same than from sample candidate to the current state. Thus, we can rewrite (4) as

$$A_k(z^*, z^{(\tau)}) = \min\Big(1, \frac{\tilde{p}(z^*)}{\tilde{p}(z^{(\tau)})}\Big). \tag{5}$$

As you can see in (5), the next state of the Markov chain depends only on its current state. If the sample candidate has higher probability than the current state, we change the state with probability 1. Alternatively, if the current state has higher probability than the candidate one, we move to this next state with probability $\frac{\tilde{p}(z^*)}{\tilde{p}(z^{(\tau)})}$. However, a symmetrical proposal distribution causes random walk behaviour because the algorithm randomly walk through the sampling space while trying to find places with high probability. This behaviour allows the algorithm to travel the sampling space slowly as it progress with the square root of the step size in average [33]. In this case, if we increase the step size, we can help the algorithm to progress faster but we are limited due to the slow Markov chain caused by the large steps. However, even if the MH has a random walk component, it can sample any distribution and gives a good intuition how MCMC algorithms work.

In Algorithm 1, we provide a pseudo code of a MH algorithm. As we mentioned before, we need to draw enough samples from $q(z^*|z^{(\tau)})$ to find the equilibrium distribution where the sample acceptance is computed using (5). Once we finish sampling the desired distribution, we take the equilibrium distribution from the Markov chain which is the desired distribution itself. Note that we do not change the state in the Markov chain when a sample is rejected. Also, replicas of a state should be deleted and the weight of a repeated state should be increased [33].

It is not possible to draw infinite number of samples, and therefore, our numerical approximation of the desired distribution is affected by the initial state. Hence, we should draw some samples for tuning, where we discard the tuning samples and keep the samples that follow them [50]. This process is also known as burn in and it is used to reduce the correlation between the approximated distribution and the initial state, while we can be sure that our samples are from the high probability region [50]. With

Algorithm 1: MH algorithm - Symmetrical proposal distribution

```
Data: Unnormalized desired distribution
Result: Equilibrium distribution of the Markov chain
MarkovChainStates[0] = initial value;
$i = 1$;
while $i <$ Number of Samples do
    $z^{(\tau)}$ = MarkovChainStates[$i-1$];
    Draw sample candidate: $z^* \sim q(z^*|z^{(\tau)})$;
    Probability of the sample candidate being accepted: $A_k(z^*, z^{(\tau)}) = \frac{\tilde{p}(z^*)}{\tilde{p}(z^{(\tau)})}$;
    if $A_k(z^*, z^{(\tau)}) >= 1$ then
        MarkovChainStates[$i$] = $z^*$;
        $i$++;
    else
        Take a random number: $random_num$ ∼Uniform(0,1);
        if $A_k(z^*, z^{(\tau)}) >= random_num$ then
            Accept sample candidate;
            MarkovChainStates[$i$] = $z^*$;
            $i$++;
        else
            Reject sample candidate;
            MarkovChainStates[$i$] = $z^{(\tau)}$;
        end
    end
end
DesiredDistribution = ComputeEquilibriumDistribution(MarkovChainStates)
```

the intuition of the MH algorithm, we can advance and explain succinctly how the other mentioned MCMC algorithms work.

The Gibbs sampling is a special case of the MH algorithm where we sample one RV at a time [33]. The MH algorithm draws samples of all RVs at the same time and afterwards rejects and accepts all of them based on the observations of the joint distribution. With the Gibbs sampling, on the other hand, we sample one RV at time where the sample is conditionally dependent on the previously sampled RVs [33]. This variation reduces the number of rejected samples as we just reject one sample at a time instead of all of them. However, this algorithm has the random walk component as well which limits the efficiency of the MCMC method [51]. We can avoid the random walk behaviour by using the Gibbs sampling with over-relaxation methods or HMC algorithms, however, over-relaxation methods are inefficient for some distributions [51].

The HMC is an elaborated form of the MH which incorporates Halmitonian dynamics to eliminate the random walk component [51]. This algorithm eliminates the random walk component through finding the trajectory of the next sample in which we will likely draw a sample that contributes to our desired distribution [51]. We find the trajectory with the gradient information of the unnormalized desired distribution likelihood function at the current state of the Markov chain [33]. Let's consider that the desired distribution is a random Gaussian, if we multiply this Gaussian by -1, we obtain the curve in Fig. 3. The ball represents the current state $z^{(\tau)}$ of the Markov chain and now we want to sample

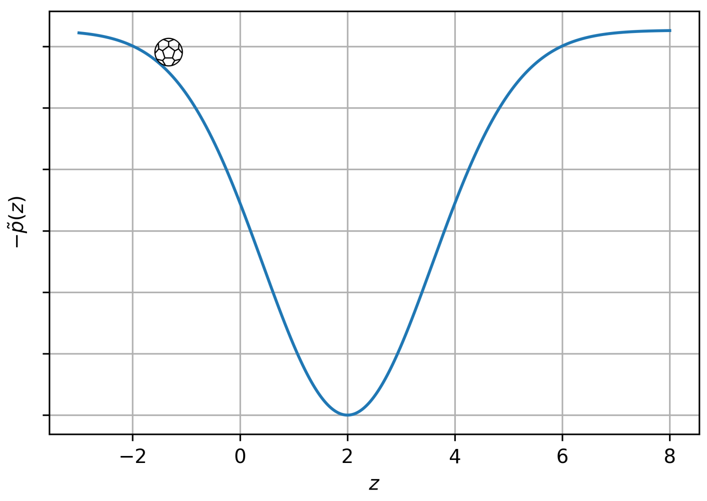

Figure 3. HMC algorithm analogy.

our next state. If we compute the potential energy and the gradient of the curve at the current state $z^{(\tau)}$, we have the momentum which we can use as the direction of the sample candidates z^*. Therefore, we use this information to increase the chances of sampling states that contribute more to the convergence of our Markov chain while using bigger step size with low probability of rejection [33]. Conversely, sampling algorithms with random walk component draw sample candidates towards random directions from the current state[1]. The reader can get more information about this algorithm in [38]. Even if the HMC has a better performance than algorithms with random walk component, it requires tuning where the user should make careful choices regarding the parameters of the length of the trajectories and the step size used in the Hamiltonian dynamics [51]. The NUTS algorithm simplifies the use of this powerful sampling algorithm and avoid U turn as we explain next.

NUTS algorithm extends the HCM algorithm with U turn avoidance and self-tuning properties [41]. The U turn happens in HMC when with the chosen step size, the algorithm always returns to the same state which limits the algorithm to explore the sampling space [52]. The self-tuning property means that we use the HMC algorithm but the algorithm chooses the needed parameter by itself. This algorithm is able to adjust its parameters to sample samples with low correlation (big step size) while keeping a low probability of rejection of the samples [41]. We investigated MCMC sampling algorithms in different studies, we found out that HMC outperforms the Gibbs and MH algorithms [33, 38, 39, 40], while according to Authors in [41], the NUTS algorithm has at least the same performance than a well-tuned HMC . Therefore, we decided to use the NUTS algorithm in our work.

[1]This intuition explains how HMC eliminates random walk behaviour, but we will not go further in the HCM formulation because it is based on differential geometry which is an advanced and challenging field of mathematics that is rarely included in statistics courses [38].

3 SYSTEM MODEL AND EVALUATION FRAMEWORK

Herein, we explain our evaluation framework which is based on the iterative Bayesian mechanism developed in [36] and the use of different prior distributions studied in [37]. In addition, we present the system model for TDOA and AOA while we demonstrate RSSI localization system implementation in Python.

3.1 System Model

In this section, we first detail the deployment scenario, the relevant measurement metrics, namely RSSI, AOA and TDOA, and then propose a Bayesian-based iterative method for accurate indoor positioning.

3.1.1 Deployment scenario

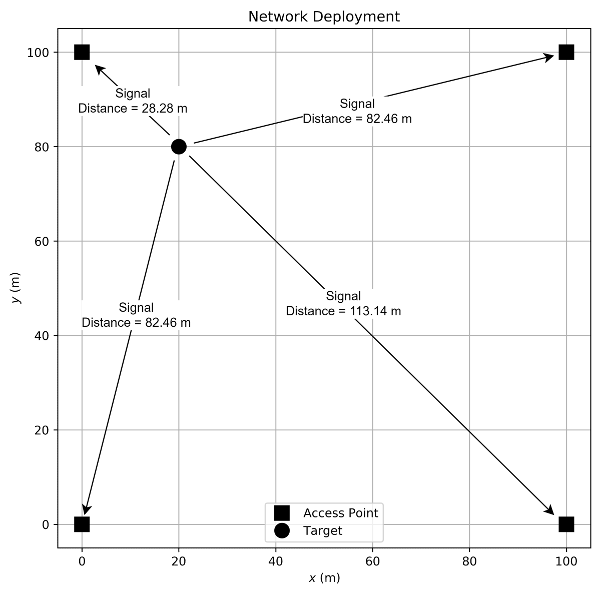

Figure 4. Representation of the deployment scenario.

Fig. 4 depicts our target evaluation scenario which corresponds to a squared warehouse of 100 m and is later used to evaluate the performance of the proposed Bayesian-based iterative positioning mechanism. In this deployment scenario, we consider four APs and one target which are represented as squares and a circle in this figure, respectively. There is one AP at each corner of the warehouse and they use the signals received from the target to collect the respective measurements. In our investigations, we aim to estimate the position of the target node whose actual position is arbitrarily set to (20, 80) m. After collecting the measurements, the AP forward all the relevant data to a common server at the edge of the network that then computes the target position estimate according to our model and considerations. Therefore, we evaluate the proposed mechanism in this deployment scenario.

An IPS needs to use one or more measurement metrics to be able to locate a target in this deployment scenario. Any measurement metric model can be generalized as [53]

$$\mathbf{r} = \mathbf{f}(\mathbf{x}) + \mathbf{n}, \tag{6}$$

where $\mathbf{r}$ is the vector of measurements, $\mathbf{x}$ is a bidimensional variable which is the target's position, $\mathbf{f}(\mathbf{x})$ is respective nonlinear function of the metric in $\mathbf{x}$, and $\mathbf{n}$ is the vector of disturbances following a zero-mean normal distribution with with variance σ^2. We extend this vectorial formulation to each one of the metrics studied in this work.

We can use nonlinear and linear frequentist inference approaches for localization of targets. In the nonlinear approach, we directly employ a optimization problem with (6) to solve for $\mathbf{x}$ by minimizing the cost function of the least squares and weighted least squares which the error function is denoted by [53],

$$\mathbf{e}_{\text{nonlinear}} = \mathbf{r} - \mathbf{f}(\tilde{\mathbf{x}}), \tag{7}$$

where vector $\tilde{\mathbf{x}}$ is the bidimensional optimization variable $[\tilde{x}\ \tilde{y}]^T$ for $\mathbf{x}$. This optimization variable corresponds to the non-linear least Square and maximum likelihood estimators [53]. In the linear approach, on the other hand, we can apply linear techniques to convert (6) into a set of linear equations in $\mathbf{x}$ where the error function is denoted by [53],

$$\mathbf{e}_{\text{linear}} = \mathbf{b} - \mathbf{A}\tilde{\mathbf{x}}. \tag{8}$$

When we employ least squares and weighted least squares on (8), we obtain linear least squares, weighted linear least squares and subspace estimators [53].

In both frequentist perspectives for linear and nonlinear estimator, we are doing a regression of the optimization variable $\tilde{\mathbf{x}}$ to a point estimate of the target's position [33]. The point estimate gives us a poor information regarding the target's position because if the estimation is incorrect, we do not have any other information regarding the target's position. Here, we propose a Bayesian approach where we do not have a point estimate but a distribution that provides us a quantification of uncertainty [33]. In this case, while we do not have a point estimation of the target's position, we have a probabilistic distribution that demarcates the region where most likely the target is located.

As mentioned in Subsection 2.1, we need to provide to the Bayesian statistical model our assumptions regarding the event of interest. In this work, it is assumed that the locations of the APs are known and the target device connects to the APs through line-of-sight links. Other assumptions depend on the event being analyzed such as RSSI, AOA and TDOA measurement metrics as we explain next.

3.1.2 Received Signal Strength-based source localization

RSSI is a typical measurement in present-day wireless communication system and measures the power of a received radio signal [31]. For the ith AP, we assume that the strength of a signal received from a source decays with the log of the distance [53]. Thus, we can derive (6) as

$$\mathbf{r}_{\text{RSS}} = \mathbf{f}_{\text{RSS}}(\mathbf{x}) + \mathbf{n}_{\text{RSS}}, \tag{9}$$

where $\mathbf{f}_{\text{RSS}}(\mathbf{x})$ is formulated as [53]

$$\mathbf{f}_{\text{RSS}}(\mathbf{x}) = \rho_o - \eta \log \mathbf{d}, \tag{10}$$

where ρ_o is the signal strength of the transmitted signal in a referential distance (1 m in our studies), η is the path loss coefficient, $\mathbf{d}$ is a vector of the Euclidean distances between the APs and the target's position ($\mathbf{x}$) [35]. If we estimate the distances in vector $\mathbf{d}$, we can find the target's position with the typical trilateration method by drawing the lines representing the distance in a two dimensional Cartesian plane. If we have at least three APs we can estimate a target's position by using the intersection of these lines [54]. It means that the joint distribution of the vector $\mathbf{d}$ results in the estimation of the target's position.

In the real world, we usually do not know the values of ρ_o and η. Therefore, we need to estimate the RVs ρ_o and η as well. The Bayesian network together with the MCMC sampling algorithm allows us to estimate ρ_o, η and $\mathbf{d}$ based on observations $\mathbf{r}_{\text{RSS}}$ and our assumptions regarding these RVs. In Fig. 5, we represent graphically the interdependencies between the RVs describing the RSS-based localization mechanism by means of a Bayesian network for n APs. The symbols in the spherical and squared vertices are RVs and constants, respectively. D_i is the Euclidean distance between the ith AP and the target at the position $(X,\ Y)$, μ_i follows the equation (9) where χ_i is normally distributed with distribution $\mathcal{N}(0, \mathbf{\Sigma})$, $\mathbf{\Sigma}$ is the covariance matrix of the measurement error with main diagonal σ_i^2, however, we consider that the measurements are independent. The interdependence of the statistical model results in the joint distribution which is denoted by,

$$f(V) = f(X)f(Y)f(\rho_o)f(\eta)\prod_{i=1}^{n} f(\sigma_i^2)f(D_i|X, Y, x_i, y_i)f(\mu_i|\rho_o, \eta, \sigma_i^2), \tag{11}$$

where V is the set of all RVs. The probabilistic graphical model in Fig. 5 is initialized as follows,

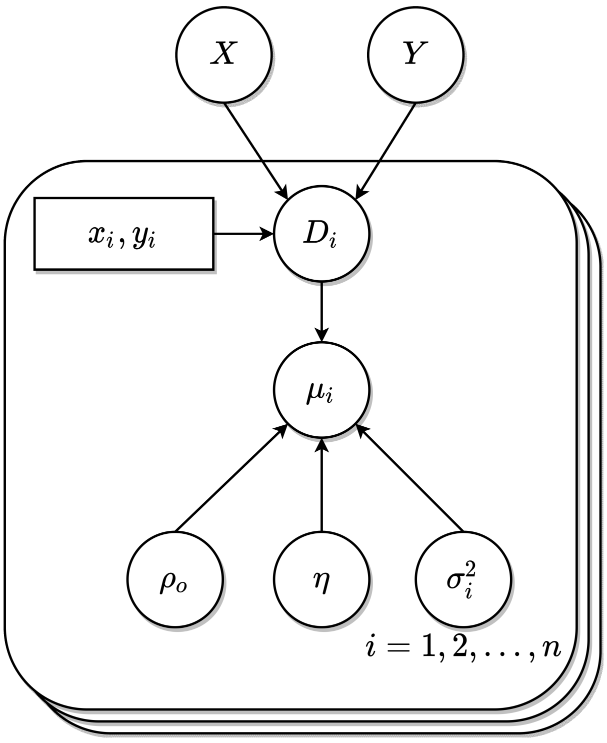

Figure 5. Bayesian probabilistic model of the RSS-based localization mechanism.

$$\begin{aligned}
&X \sim \text{Uniform}(0,\, L),\\
&Y \sim \text{Uniform}(0,\, W),\\
&D_i \sim \sqrt{(X - x_i)^2 + (Y - y_i)^2},\\
&\mu_i \sim \rho_o + \eta log(D_i) + \chi_i,\\
&\rho_o \sim \text{Normal(0,100)},\\
&\eta \sim \text{Normal(0,100)},\\
&\sigma_i^2 \sim \text{HalfNormal(10)}.
\end{aligned} \tag{12}$$

When we do not have any information regarding the RVs, we should use distributions with large variance or flat distributions [34]. The target has the same probability to be at any coordinate of plane, and therefore, in (12) we select a flat distribution (i.e. uniform distribution) to represent our previous knowledge about the target's position. For σ_i^2, η, ρ_o, we chose distributions with large standard deviation as we do not have any prior knowledge about them either. σ_i^2 is a half normal because it can not have negative values.

The MCMC algorithm needs our assumptions as in (12) and RSSI measurements to estimate the target's position. The RSSI measurements are distributed according to the joint distribution (11). Therefore, the MCMC sampling algorithm obtains the unnormalized joint distribution from the measurements to simulate all the RVs in our statistical model. It simulates the RVs through a sampling process where it tries to find a combination of distributions that results in the joint distribution in which the measurements are distributed. In this sampling process, the MCMC estimates the target's coordinates and all the other RVs in our statistical model.

3.1.3 Time Difference of Arrival-based source localization

By employing this method, the signal transmitted by a source is measured in different known receivers locations to calculate the time of arrival. The TDOA is given by the difference between the time of arrival at these receivers with respect to a common reference location [53].[1] In our work, the target corresponds to the transmitting source while the receivers of the target signal represent the APs. This method requires that the APs to be clock synchronized in order to properly compute the respective TDOA metrics. The TDOAs values are stored in a vector of size $L - 1$ where L is the number of APs whose components are given by, [53]

$$r_{\text{TDOA},l} = g_1 + n_{\text{TDOA},l}, \quad l = 1, 2, \cdots, L - 1, \tag{13}$$

where $g_{l+1,1}$ is defined by,

$$g_l = t_{l+1} - t_1. \tag{14}$$

The variable $\mathbf{t}$ is a vector of the time of flight from the target node to the lth AP, $\mathbf{g}$ is the TDOA vector without considering the error, $\mathbf{n}_{TDOA}$ is the intrinsic measurement error drawn from a normal distribution with arbitrary mean and variance. Note that the AP corresponding to t_1 is chosen as the reference receiver to carry out our computations. If we multiply $\mathbf{r}_{\text{TDOA}}$ by the propagation speed, we obtain a vector whose components are

[1]This is a generalization of the time of flight technique where clock synchronization between the target and the APs is not required.

the respective distance differences between the APs and the target. Thus, by combining (13) and (14) above, we can write

$$c \cdot r_{\text{TDOA},l} = d_{l+1} - d_1 + c \cdot n_{\text{TDOA}}, \quad l = 1, 2, \cdots, L-1, \tag{15}$$

where c is the speed of light and the vector $\mathbf{d}$ is the Euclidean distances between the APs and the target. If we estimate the distances in $\mathbf{d}$ based on the propagation speed and the known positions of the APs, we can draw a lines representing the distances from each AP to the target based on the TDOA measurements. The drawn lines are hyperbolic lines in the Cartesian coordinate system, and the intersection of the hyperbolic give us an estimation of the position of the target [53]. Therefore, if we make the Bayesian inference of the joint distribution of $\mathbf{d}$, we will have the distribution that represents the region where the target is located.

In the Fig. 6, we designed our Bayesian network model based on (15). In this model, we consider that the mean of the measurement error is zero. The symbols inside circles and squares represent RVs and constants, respectively. In this graphical mode, (x_i, yi) yields the known coordinates of the APs and c is the wave propagation speed, while D_i, $\boldsymbol{\mu}$ and $\boldsymbol{\sigma^2}$ are respectively the Euclidean distance between the ith AP and the target, the TDOA vector and the variance of the error, and (X,Y) is a bidimensional RV representing the position of the target in a Cartesian coordinate system. The assumptions we used in our model are denoted by,

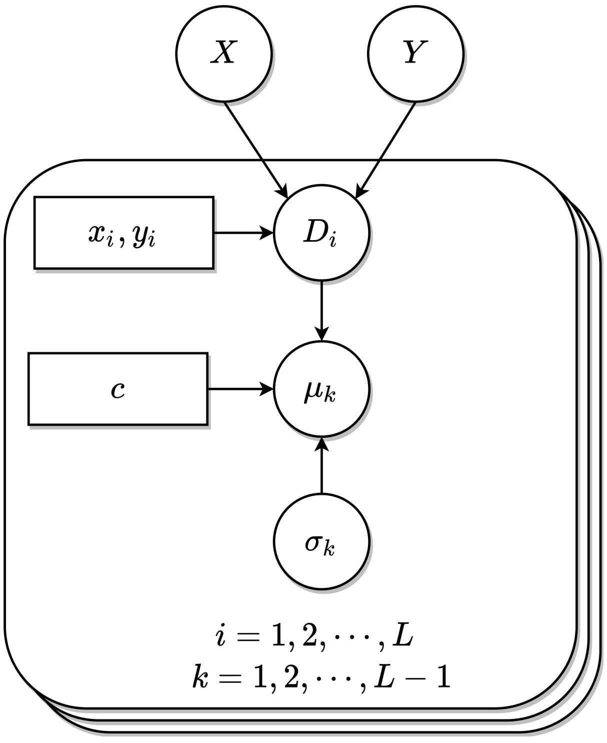

Figure 6. Bayesian probabilistic graphical model of the TDOA-based localization mechanism.

$$\begin{aligned}
&X \sim \text{Uniform}(0,\, L),\\
&Y \sim \text{Uniform}(0,\, W),\\
&D_i \sim \sqrt{(X - x_i)^2 + (Y - y_i)^2},\\
&\mu_k \sim (D_{k+1} - D_1)/c + n\\
&\sigma_k^2 \sim \text{HalfNormal}(10),
\end{aligned} \tag{16}$$

where n is the measurement error following distribution $\mathcal{N}(0, \mathbf{\Sigma})$, $\mathbf{\Sigma}$ is the covariance matrix of the measurement error of the APs in which the main diagonal is the vector $\boldsymbol{\sigma}^{\mathbf{2}}$. In our work, we consider the error to be independent between the APs. The TDOA vector depends on the speed of light, the separation distances between the target and each AP and the variance of the measurement error. Moreover, D_i depends on the location of the target and of the APs. Therefore, the joint distribution can be defined as

$$f(V) = f(X)f(Y)\prod_{i=1}^{L} f(D_i|X, Y, x_i, y_i)\prod_{k=1}^{L-1} f(\sigma_k^2)f(\mu_k|D_{k+1}, D_1, c, \sigma_k^2), \tag{17}$$

where V is the set of RVs of the joint distribution.

3.1.4 Angle of Arrival-based source localization

In order to use this method, the APs need to employ an array of antennas to measure the angle in which they receive the signal from the target [53]. The angle of arrival at the ith access point is denoted by [53]

$$\phi_i = \arctan\frac{y_t - y_i}{x_t - x_i}, \tag{18}$$

where the x_t and y_t are the coordinates of the target, and x_i and y_i are the coordinates of the ith access point. By account for the measurement error, our vectorial formulation in (6) is updated as follows [53]

$$\mathbf{r}_{\text{AOA}} = \boldsymbol{\phi} + \mathbf{n}_{\text{AOA}}, \tag{19}$$

where $\mathbf{r}_{\text{AOA}}$ is the vector of the measured angles, $\mathbf{n}_{\text{AOA}}$ corresponds to the measurement error following zero mean Gaussian distribution and it is independent between the APs. If we estimate ϕ_i with the AOA measurements, we have a line of bearing between the ith AP. We need the intersection of at least two line of bearings to estimate the target's position [53]. In a Bayesian perspective, the joint distribution of $\boldsymbol{\phi}$ provides us the region of the probable target's position. Fig. 7 represents the statistical graphical model of an AOA-based IPS with N APs whose the RVs are defined by

$$\begin{aligned}
&X \sim \text{Uniform}(0,\, L),\\
&Y \sim \text{Uniform}(0,\, W),\\
&\mu_i \sim \arctan\frac{Y - y_i}{X - x_i} + n_{\text{AOA}}\\
&\sigma_i^2 \sim \text{HalfNormal}(10),
\end{aligned} \tag{20}$$

where μ_i follows (19) and σ_i^2 is the variance of n_{AOA}. Considering our prior knowledge about the parameters of the model, we use a bidimensional uniform and half-normal distributions to model the target node coordinate (X, Y) and the respective measurement error standard deviation σ_i^2, respectively. The joint distribution of this statistical model is formulated as

$$f(V) = f(X)f(Y)\prod_{i=1}^{N} f(\sigma_i^2)f(\mu_i|X, Y, x_i, y_i, \sigma_i^2), \tag{21}$$

where V is the set of RVs of the joint distribution.

After estimating the joint distribution of TDOA, RSSI or AOA using MCMC methods, we have an updated knowledge about the RVs in the model and we can use this new knowledge to the subsequent estimations of the respective joint distribution. The use of the updated knowledge regarding the RVs is the base of our iterative Bayesian method as we describe next.

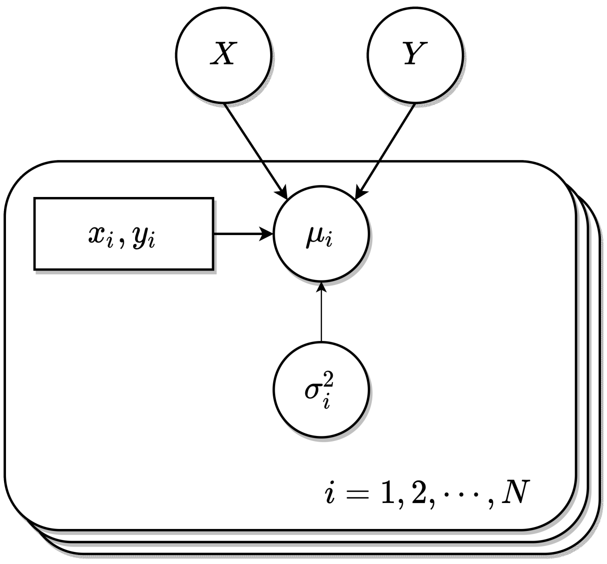

Figure 7. Bayesian probabilistic model of the AOA-based localization mechanism.

3.1.5 Iterative Bayesian method

The iterative Bayesian method sequentially update our prior distribution with our posterior distributions. It means that we repeatedly/iteratively make measurements, estimate the target's position based on our prior distributions and measurements, and update our prior distributions using the estimated posterior distributions.

As mentioned in Section 2.1, the Bayes' theorem enable us to make inferences based on our current knowledge regarding a statistical experiment of interest. Our current knowledge of the RVs in a Bayesian statistical model is represented by the prior distributions of the RVs. The MCMC method samples all the RVs of the Bayesian network model in order to estimate their posterior distribution, and we can use this new knowledge regarding the RVs for the subsequent iteration [55]. In this case, we are not discarding the knowledge that we obtained previously and we are keeping a statistical history of our targeted event. The purpose of updating the prior distributions with the estimated posterior distributions is to find more accurate estimations as the prior distributions affect the performance of the MCMC sampling method [33]. Indeed, when we reuse the posterior distribution to update the next iteration prior distribution, we will carry out biased estimations, which can add error to our proposed mechanism estimation over time [55]. We denoted it as biased estimations because the estimator uses our bias or beliefs regarding the RVs to estimate them.

The Fig. 8 describes the algorithm of our proposed IPS mechanism using a flowchart. The description of the blocks in the flowchart is below:

- **Measurements:** These are the observations used by the MCMC sampling algorithm. The observations are generated according to the metric used such as (10), (15) and (19);
- **Take the first assumptions of prior:** We set the initial assumptions and prior distribution of our proposed mechanism. They could be the assumptions and prior distribution in (12), (16) and (20), for example;
- **MCMC:** We run the MCMC sampling algorithm to estimate the posterior distributions based on our prior distributions and observations;
- **Take the last posterior distribution:** We take our last estimated posterior distribution which is our updated knowledge about the RVs. The current/last estimated posterior distribution provides us information that can be used by the prior distribution in the next iteration;
- **Model the distribution that will be used as prior:** We create the prior distributions and set their parameters according to the current estimated posterior distribution;
- **Update prior:** We updated our statistical graphical models in (12), (16) and (20) with the prior distributions created in the previous block;
- **Estimation/Posterior Distribution:** This is the output of the MCMC sampling algorithm. Here, we can plot and analyze the estimated posterior distributions.

In the first iteration, our mechanism uses the prior distributions defined in (12), (16) or (20) depending what is being measured. From the second iteration onward, we model the RVs' prior distributions based on the current posterior distributions which is our updated information about them. However, the variance of measurement error can change drastically in the real world as the indoor environments are dynamic. It means that from one iteration to another the actual value of $\boldsymbol{\sigma^2}$ in (12), (16) or (20) can be significantly different. In this case, we can not be certain about our updated knowledge regarding this RV and we should use a prior distribution with large variance [34]. Therefore, we arbitrary do not update the prior distribution of $\boldsymbol{\sigma^2}$ by always reusing our initial prior distributions.

In this work, we evaluate three distinct schemes to update prior distributions which are based on the posterior distribution estimate. We denote them as Full Posterior (FP), Full Gaussian (FG) and Mixture Distribution (MD). In FP, we model our prior distribution using the current posterior distribution directly as the prior distribution, while in FG,

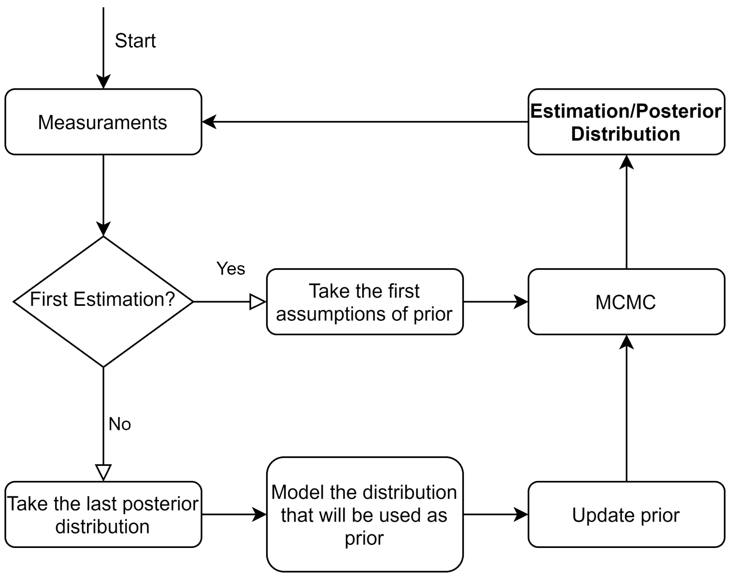
Start
Measuraments
Estimation/Posterior Distribution
First Estimation?
Yes
Take the first assumptions of prior
MCMC
No
Take the last posterior distribution
Model the distribution that will be used as prior
Update prior

Figure 8. Iterative Bayesian method flowchart.

we model the prior distribution as a Gaussian with the same mean and the double of the standard deviation of the current posterior distribution. The MD distribution is a mixture distribution of FP and FG prior distributions with 50% of weight each where the weight value is a arbitrary. These prior distribution schemes are equivalent to the block **"Model the distribution that will be used as prior"** in Fig. 8.

Now that we have all the information of the system model, we can advance for the implementation of our Bayesian-based IPS.

3.2 Implementation

In this section, we describe the implementation details of the proposed iterative mechanism. Firstly, we introduce the PyMC3 framework along with its requirements and, secondly, we show how to describe our Bayesian network models using this framework and how to use the MCMC sampling algorithm. We decided to describe the code for the RSSI because the the graphical model of the RSS-based source localization is more didactic than the others, i.e. easier to follow and understand.

3.2.1 PyMC3 Framework and Requirements

The PyMC3 is a open-source probabilistic programming framework developed using the general purpose Python language [52]. This framework allows to describe statistical models intuitively with syntax and notation very similar to the statistical formulation using mathematical symbols [52]. The PyMC3 framework provides the powerful and modern HMC and NUTS MCMC sampling algorithms [52]. These MCMC algorithm are powerful because they eliminate random walk behaviour as we mentioned in Section 2.2, and have a good performance with high dimensional and complex posterior distributions [52]. MCMC sampling algorithms with random behavior component need more time to converge the Markov chain to the equilibrium distribution [40]. NUTS extends HMC with self tuning whereby the parameters required by the HMC are automatically set and with U-turn avoidance [41] as we mentioned in Section 2.2. It allows inexperienced users to use a powerful sampling algorithm that has at least the same performance of a well-tuned HMC according to the Authors in [41]. In other words, this tool allows the user to focus on the design process and evaluation of statistical models instead of the mathematical and sampling algorithm details [34]. Note that this framework uses the library Theano from Python to make its computations including the gradient computation for HMC and NUTS through automatic differentiation [52]. We can find all the documentation of this framework available in [56]. The code Listing 3.1 shows the initial setup of libraries we need to load to run our scripts.

We carry out our simulation campaign in a cluster running operational system CentOS Linux 7 with Python version 3.6.8.

```python
# The libraries needed
import numpy as nump
import theano.tensor as tens
from scipy import stats
import pymc3
```

Listing 3.1. Loading Python libraries.

3.2.2 Implementation of the RSS-Based IPS

Here, we implement the system and graphical model described in the Section 3.1 using the PyMC3 framework. We start with the deployment scenario detailed in Subsection 3.1.1 where we describe the environment as well as the location of the target and the APs. The code of the deployment scenario is below.

```python
# Number of measurements for each AP
sp_size = 50

# The dimensions of the warehouse
L = 100 # Length
B = 100 # Breadth

# APs position
anchor_pos = nump.array([[0, 0], [0, B], [L, 0], [L, B]])
num_anchors = len(anchor_pos) # Number of APs

# Position of the target
target_pos = nump.array([[20, 80]])
```

Listing 3.2. Warehouse deployment scenario.

We need to simulate the measurements which the APs will carry out. Therefore, we firstly need to compute the real distance between the target and the APs as implemented in Listing 3.3.

```python
# Function that computes the Euclidean distance equation using the
# numpy library
def nump_euclidean(x, y):
    return nump.sqrt(((x - y)**2).sum(axis = 1))
```

```

# The actual distance between the APs and the target.
distance = nump.zeros((num_anchors,1))
for i in range(0, num_anchors):
    distance[i] = nump_euclidean(anchor_pos[i], target_pos)
```

Listing 3.3. The Euclidean distance between the APs and the target.

In Listing 3.4, we generate the measurements of the APs according to (10) based on the distance between the target and APs where the measurements are assumed to be independent.

```
# We firstly create measurement error
var_err = .5 # Variance of the error arbitrary chosen
cov_err = cov*nump.identity(num_anchors) # Covariance matrix

# Error matrix compatible with the sample size and number of APs
error = nump.random.multivariate_normal(mean=nump.zeros(num_anchors),
                                        cov=cov_err, size=sp_size)

# We arbitrary choose the path loss coefficient and the transmission
# power of the target.
bb = 3 # Path loss coefficient
aa = 3 # RSSI in one meter of distance from the target

# We take the real distance and repeat it to be compatible the sample
# size
d_mtx = nump.repeat(distance, sp_size, axis = 1)

# We compute the RSSI in ideal conditions and add error
mu = aa - bb*nump.log(d_mtx) # RSSI ideal condition
ρ_hat = mu + error # RSSI with error
```

Listing 3.4. RSSI measurements.

Now that we have our observed data, we model the Bayesian network following (12). The corresponding Python code is shown below. Note that we consider the joint distribution a multivariate normal distribution because the four APs make RSSI measurements with a Gaussian error. It means that our estimated target's position follows a Bayesian approach where the estimation is probabilistic distribution. It is different from the frequentist inference where we have a point estimate of the target.

```
# PyMC3 uses theano for computation. Therefore, we need to compute the
# Euclidean distance using theano.
def tens_euclidean(x, y):
    return tens.sqrt(tens.sqr(x - y).sum(axis = 0))

with pymc3.Model() as rssi_model:
    x = pymc3.Uniform('x', 0, L)
    y = pymc3.Uniform('y', 0, B)
    φ = tens.stack([x, y], axis = 1)
    α = pymc3.Normal('α', mu = 0, sd = nump.sqrt(10**3)) # ρ_o
    β = pymc3.Normal('β', mu = 0, sd = nump.sqrt(10**3)) # η
    anchors_range = range(num_anchors)
    d = [[] for i in anchors_range]
    μ = [[] for i in anchors_range]
    σ = [[] for i in anchors_range]
    for i in anchors_range:
        d[i] = pymc3.Deterministic('d' + str(i),
                                   tens_euclidean(anchor_vt[i], φ))
        μ[i] = α - β*tens.log(tens.repeat(tens.stacklists([d[i]]),
                                          sp_size, axis=0))
        σ[i] = pymc3.HalfNormal('σ' + str(i), sd = 10)
    covariance = tens.nlinalg.alloc_diag(σ)
    μ_rssi = tens.stack(μ).T
    # joint_dist is joint distribution of our statistical model. It is
    # a multivariate normal distribution.
    joint_dist = pymc3.MvNormal('joint', mu=μ_rssi, cov=covariance,
                                observed=ρ_hat)
```

Listing 3.5. Probabilistic graphical model of the RSS-based source localization mechanism.

The joint distribution implemented in Listing 3.5 incorporates our assumptions regarding path loss, distance, and also our prior knowledge about the RVs. With this probabilistic model, we use the NUTS algorithm to sample the corresponding RVs so as to estimate the target's position. Listing 3.6 has the code which runs the NUTS algorithm 3.6. We use four chains which explore the sampling space starting from different initial positions. We use this together with tuning to reduce the influence of the initial sample to the resultant posterior distribution. For each chain, we draw 2000 samples for the estimation plus 1000 samples used for burn in. We set the target of acceptance of samples to 80% which means that the sampler will try to keep a step size where 80% of the sample candidates are accepted.

```
n_draws, n_jobs, n_cores, n_tunes = 2000, 4, 4, 1000
# n_samples is the number of accepted sample per chain
# n_chain is the number of chains used by the algorithm
# n_cores is the number of process which will run in parallel
# n_burn is the number of samples used as burn in

# Running the NUTS algorithm for our model rssi_model
# The variable traced_samples has the output MCMC sampling algorithm
with rssi_model:
    traced_samples = pymc3.sample(draws = n_samples,
                                  step=pymc3.NUTS(target_accept=.8),
                                  chains = n_chain, cores =
                                  n_cores,  tune = n_burn)
```

Listing 3.6. NUTS algorithm sampling joint distribution.

The sampling function from PyMC3 returns samples of the traced RVs in the model. Fig. 9 shows the estimated position of the target in a 2D plane using a Kernel Density Estimate (KDE) heatmap. We can see that the black circle representing the actual position is indeed within the region where the MCMC method estimated for the target's position. The approximated posterior distributions of the RVs X, Y, ρ_o, η and D_i estimated by the MCMC algorithm are in the Fig. 10, while the RVs σ_i^2 are in the Fig. 11. The MCMC method estimate these distributions based on unnormalized joint distribution taken from the RSSI measurements. Therefore, the algorithm estimates or simulates the RVs by approximating their joint distribution to the unnormalized joint distribution. We can see that the estimated values of the RVs are close to the actual values. The Table 1 compares the values side by side. The estimated position of our target is (19.69, 80.94) m which by applying the Euclidean distance with the actual position results in an error of 98.98 cm.

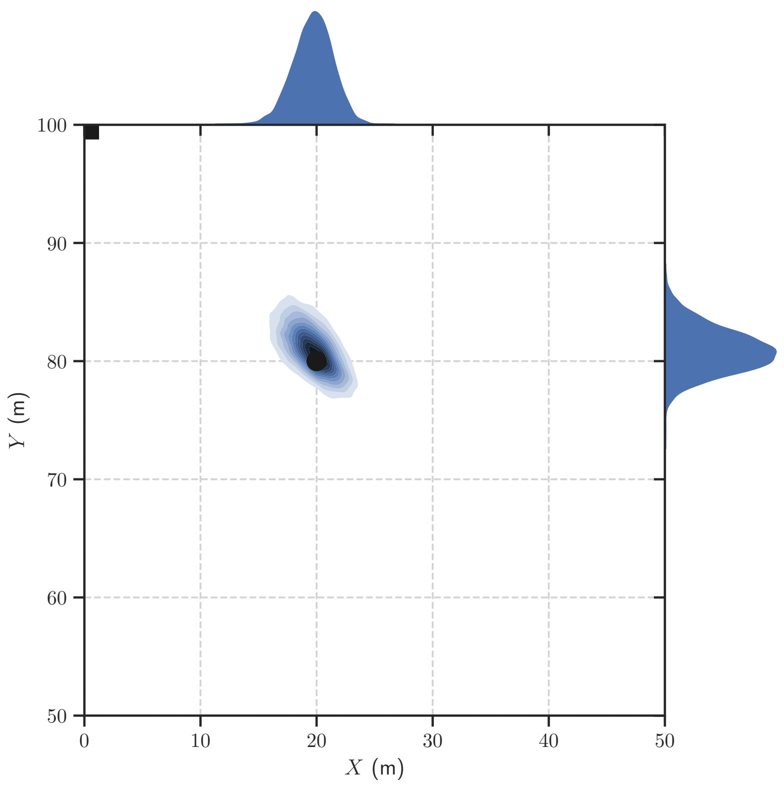

Figure 9. Kernel density estimate of the target's position

Table 1. Actual value vs Estimated value of the RVs in our statistical model.

RV	Actual value	Expected value	Variance
X	20 m	19.69 m	2.77 m
Y	80 m	80.94 m	3.39 m
ρ_0	3 dBm	2.49 dBm	0.93 dBm
η	3	2.88	0.05
D_1	82.46 m	83.31 m	2.51 m
D_2	28.28 m	27.43 m	4.88 m
D_3	113.14 m	114.02 m	4.93 m
D_4	82.46 m	82.56 m	1.98 m
σ_1^2	0.5 dBm	0.54 dBm	0.002 dBm
σ_2^2	0.5 dBm	0.54 dBm	0.002 dBm
σ_3^2	0.5 dBm	0.52 dBm	0.002 dBm
σ_4^2	0.5 dBm	0.46 dBm	0.002 dBm

98.98 cm of error is too big for applications that require a maximum error of a few centimeters. Therefore, we proposed an iterative mechanism to reduce the error and allow this Bayesian method to be used in applications with more strict requirements regarding the estimation of targets' position. The proposed iterative method reuses the output of the the first iteration generated by the MCMC sampling algorithm which contains the information of our posterior distributions as our prior distributions of the next iteration.

The Listing 3.7 has the code used for this purpose. This code is an example of FP prior distribution as defined in the Section 3.1.5. It is worth noting that the PyMC3 requires prior distributions that can be represented analytically, and the output of our MCMC sampling algorithm is a numerical approximation. It means that we can not assign the posterior distribution approximation directly as prior distribution, however, the PyMC3 has a function that interpolates our approximated distribution which then allows to incorporate approximated distributions in our statistical model [56]. An example of this function named "Updating Priors" is available in the documentation of the library [56].

```
def from_posterior(RV, RV_samples):
  min_value, max_value = nump.min(RV_samples), nump.max(RV_samples)
  diff_max_min = max_value - min_value
  x_axis = nump.linspace(minimum_value, maximum_value, 100)
  pdf = stats.gaussian_kde(samples)(x)
  x_axis = nump.concatenate([[x_axis[0] - 3 * diff_max_min], x_axis,
                             [x_axis[-1] + 3 * diff_max_min]])
  pdf = nump.concatenate([[0], pdf, [0]])
  return pm.Interpolated(RV, x_axis, pdf)

all_traces = [traced_samples]
for iterations in range(5):
  # Generate observation for the next iteration
  error = nump.random.multivariate_normal(mean=nump.zeros(num_anchors),
                                          cov=cov_err, size=sp_size)
  ρ_hat = mu + error # RSSI with error
  with pymc3.Model() as rssi_model_iterative:
    x = from_posterior('x', traced_samples['x'])
    y = from_posterior('y', traced_samples['y'])
    φ = tens.stack([x, y], axis = 1)
    α = from_posterior(α, traced_samples[α]) # ρo
    β = from_posterior(β, traced_samples[β]) # η
    anchors_range = range(num_anchors)
    d = [[] for i in anchors_range]
    μ = [[] for i in anchors_range]
    σ = [[] for i in anchors_range]
    for i in anchors_range:
      d[i] = pymc3.Deterministic('d' + str(i),
                                 tens_euclidean(anchor_vt[i], φ))
      μ[i] = α β*tens.log(tens.repeat(tens.stacklists([d[i]]),
                                      sp_size, axis=0))
      σ[i] = pymc3.HalfNormal('σ' + str(i), sd = 10)
    covariance = tens.nlinalg.alloc_diag(σ)
    μ_rssi = tens.stack(μ).T
    # joint_dist is joint distribution of our statistical model. It is
    # a multivariate normal distribution.
    joint_dist = pymc3.MvNormal('joint', mu=μ_rssi, cov=covariance,
                                observed=ρ_hat)
    # Using the NUTS algorithm to sample the RVs of our joint
    # distribution
    traced_samples = pymc3.sample(draws = n_samples,
                                  step=pymc3.NUTS(target_accept=.8),
                                  chains = n_chain, cores = n_cores,
                                  tune = n_burn)
    # Storing our numerical approximations in a vector
    all_traces.append(traced_samples)
```

Listing 3.7. Updating our prior distributions.

The Listing 3.7 updates the prior distributions five times after our initial estimation/iteration. The estimated target's coordinates are in Fig. 12. We can see that our proposed mechanism converged to the wrong coordinate of Y because the small variance given by the FP prior distribution. It means that the bias of the FP scheme provides prior distributions with a too small variance, and therefore, the MCMC algorithm can not explore the sampling space efficiently because of the small variance of the prior distribution. The Table 2 helps us to compare the estimated value with the actual value of the target's coordinates. We can see that our proposed mechanism using FP prior distribution brought the initial error from 98.98 cm to 101.07 cm. Therefore, our proposed mechanism using FP has a poorer estimation than just using the initial prior distributions in (12). We already discussed and solved this problem using FG and MD prior distributions in [36, 37] which provide better results.

Table 2. Actual value vs Estimated value of the RVs in our statistical model after five iterations.

RV	Actual value	Expected value	Variance
X	20 m	19.96 m	0.10 m
Y	80 m	81.01 m	0.13 m

As we aforementioned, we can implement our statistical graphical model using a syntax and notation similar to the statistical formulation with PyMC3. Therefore, we can use the same intuition that we used in the RSS-based IPS for TDOA and AOA by employing the formulation from Subsections 3.1.3 and 3.1.4, respectively. Next, we evaluate our proposed mechanism for static and moving targets using a TDOA-based IPS.

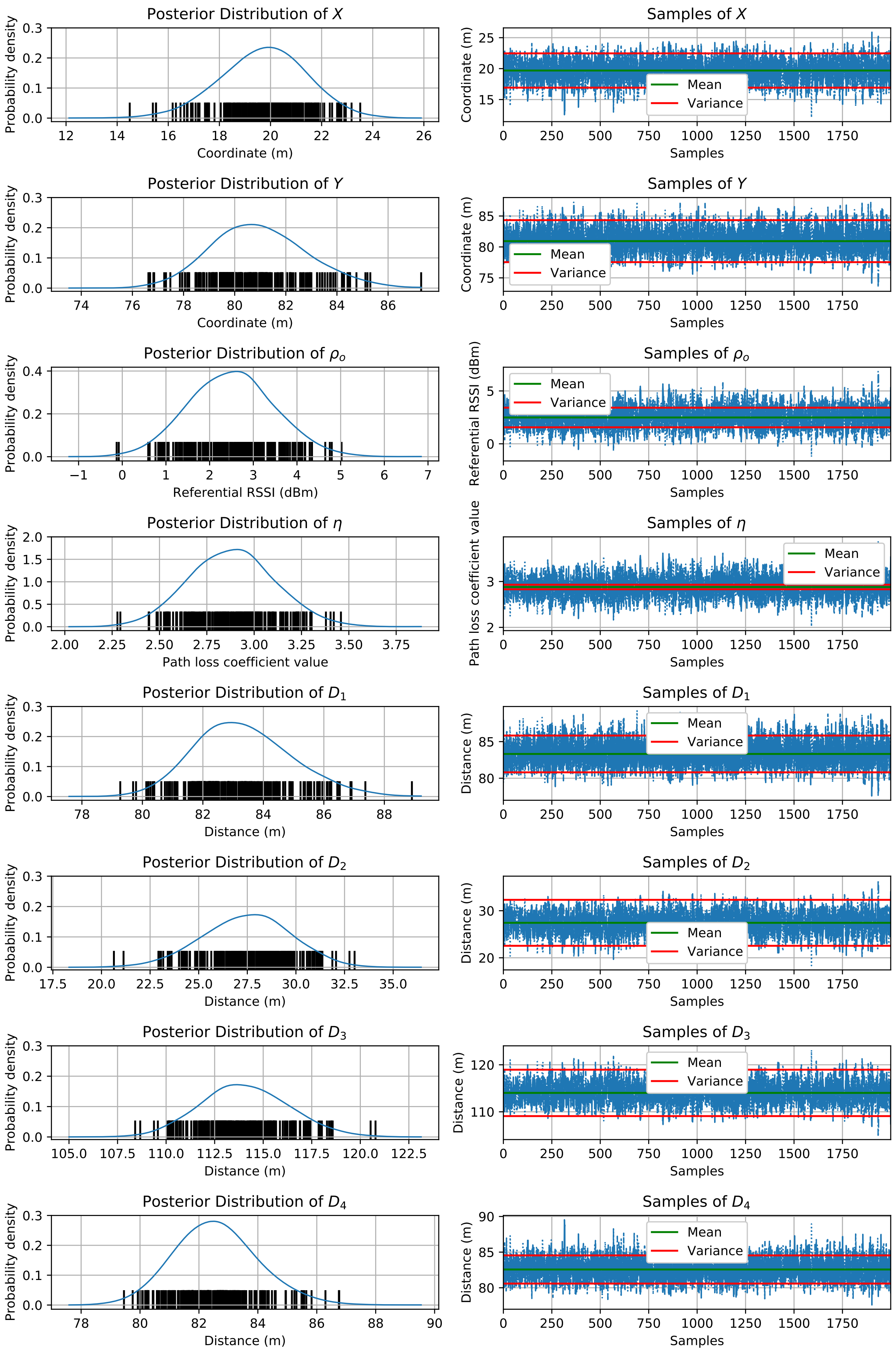

Figure 10. Estimation of the RVs of our model - Part 1.

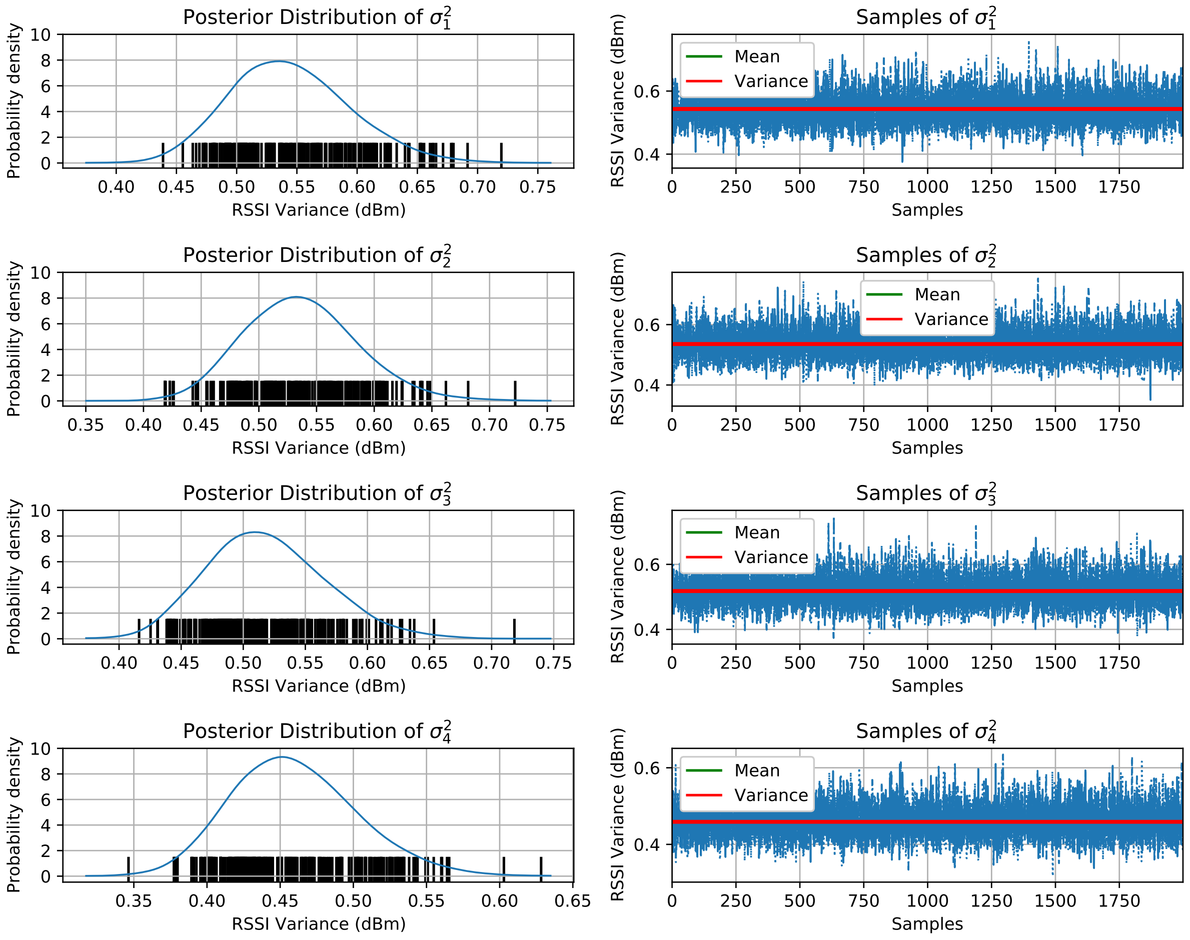

Figure 11. Estimation of the RVs of our model - Part 2.

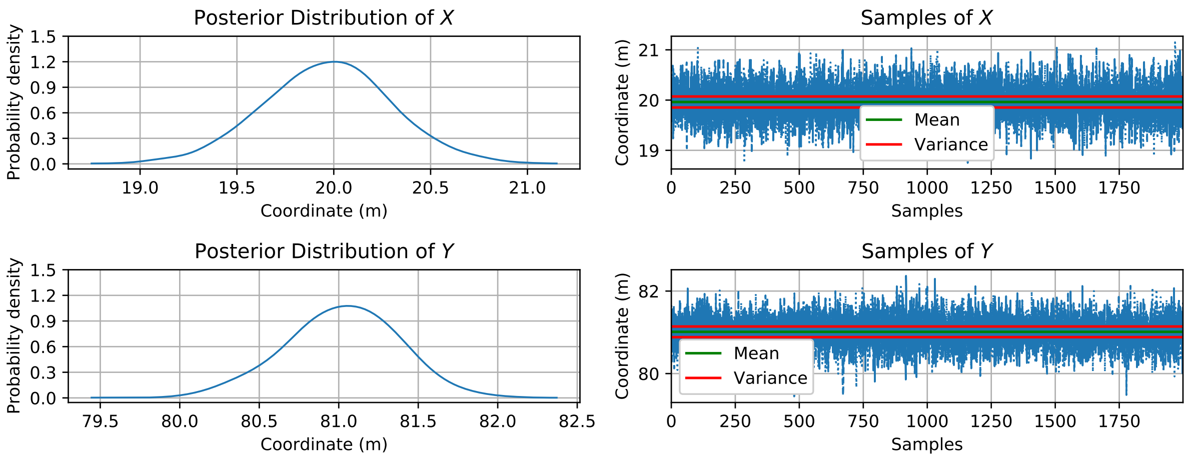

Figure 12. Estimation of the coordinates of our model after five iteration

4 PERFORMANCE EVALUATION

In this Section, we discuss and evaluate the performance of the proposed iterative TDOA-based IPS. An exhaustive simulation campaign is carried out and the results are shown in terms of RMSE and CDF of the mean error. CDF to mean error is the accumulated error between the estimated and the actual target's position where the estimated position is considered to be the mean of the approximated posterior distribution. We consider the variance of the measurement error as $1 \cdot 10^{-15}$ s which corresponds to a variance of 100 m in terms of the distance between the target and the APs. For an increasing number of measured samples per iteration and distinct prior distributions, the evaluation framework is first used to assess how the proposed TODA-based mechanism performs in a baseline scenario where the target node is stationary. Thereafter, we evaluate how the proposed mechanism performs when the target node moves and compare the results against the baseline scenario.

4.1 Evolution of the position estimates with iterations

To begin with, we assess how the iterative operation of the proposed algorithm affects the accuracy of the target node position estimates. In fact, the estimation accuracy improves through the iterations of our proposed mechanism as shown in Fig. 13. Each AP is assumed to collect 50 measurements per iteration and we employ the FP scheme to update the prior distribution. In this figure, the mean of the posterior distribution gets closer to the actual coordinate through the iterations, and our uncertainty about the coordinate reduces as well. It happens because at each new iteration the MCMC sampling algorithm is fed back with a prior distribution that has lower KL divergence to the actual target coordinate distribution which thus provides a better estimation [34]. Fig. 14 shows how the error reduces through the iterations in terms of RMSE. This result asserts the potential of our proposed mechanism which estimates the target node position with an RMSE error of approximately 27.25 cm. However, we observe a diminishing returns effect after around five iterations when the estimate gains in terms of RMSE reduces significantly at each new step. This behaviour is due to the fact that at the beginning we had prior distributions with little or no information regarding the posterior distribution, while later iterations already use prior distributions closer to the actual target coordinates. As our beliefs about the target's coordinates are close to the actual coordinates, we do not gain much new information in the last iterations. It is also worth noticing that, the performance of our mechanism is limited by the measurement errors.

Fig. 15 presents the CDF of the mean error which can be used to assess how the iterative method improves the estimation of the target's position. When comparing the curves of 1 and 20 iterations at the 50th percentile, we observe the former has an error higher than 1 meter while the latter undergo approximately 23 cm only. Different from the traditional frequentist approach which provides a point estimate, these curves give more information about the reliability of our proposed mechanism so that 99% of the estimations have an error lower than 64 cm for 20 and 15 iterations. However, at the 80th percentile, 20 iterations curve exhibits an error lower than 36 cm while the 15 iteration curve has error about 41 cm. We also can see that the accuracy gain per iteration reduces

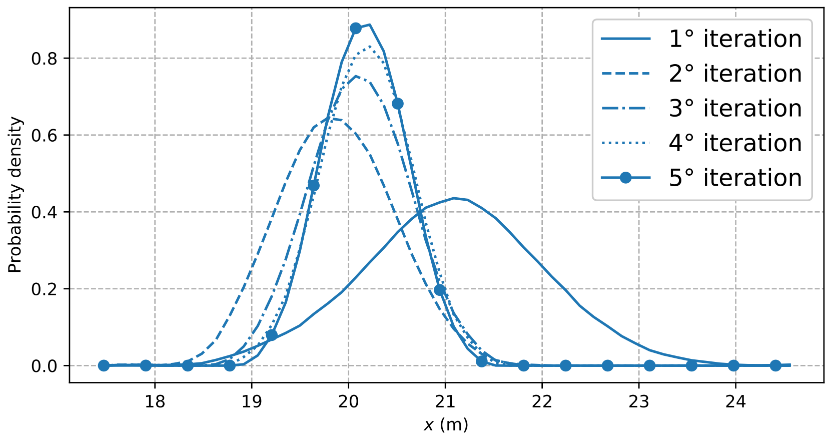

Figure 13. Evolution of the X coordinate posterior distribution.

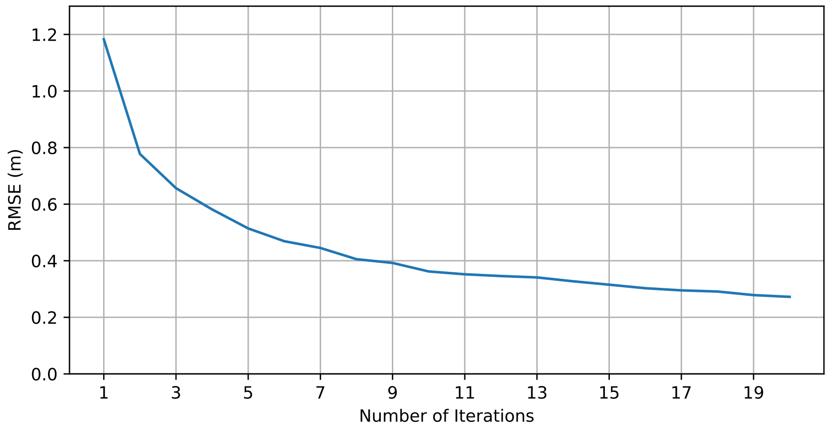

Figure 14. Euclidean Distance - RMSE

with the number of iterations. It means that if we estimate the target's position with b iterations where b tends to infinity, the estimation at the iteration $(b+1)$th is equal to the one at bth.

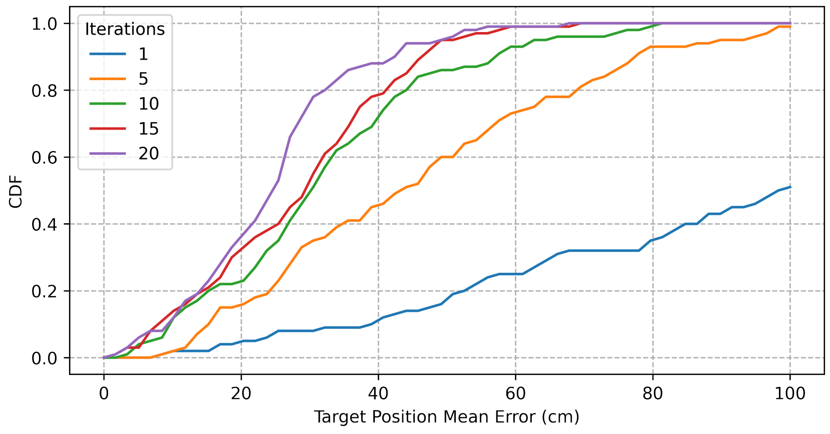

Figure 15. Evolution of the mean error cumulative distribution function.

4.2 Impact of the Prior distribution on the algorithm performance

When considering a static target, we assess the proposed mechanism for different prior distribution models. Fig. 16 shows our results in terms of RMSE using the FP, FG and MD definitions of priors distribution selection schemes as defined in Subsection 3.1.5. In the test scenarios where the target remains static, the FP outperforms both FG and MD for static targets. At the 20th iteration, the FP scheme has a RMSE of approximately 27.25 cm, while FG and MD have 79.96 cm and 68.95 cm, respectively. The FG and MD have higher variance regarding the posterior distribution when compared to the FP prior distribution. Here, we define prior distributions with higher variance or higher uncertainty than the approximated posterior distributions as prior distributions with weaker information regarding the posterior distributions [34]. We use a weaker information when we are not sure if the updated knowledge about an event corresponds to the reality or when we do not want limit our posterior distribution approximation with our bias [34]. We observed the posterior distribution's estimator being limited in Subsection 3.2.2 where after five iterations of our RSS-based IPS using FP scheme, the approximate posterior distribution converged to the wrong target's position because the FP prior distribution scheme limits the sampling space of the MCMC method. We solved this problem for RSS-based IPS in [36, 37], where we obtained a better performance when using FG and MD. Therefore, a weaker information enabled the MCMC method to explore better the sampling space. In our TDOA-based IPS, on the other hand, if we increase our uncertainty regarding our updated knowledge of the target's position by modeling our prior distributions with a weaker information about the posterior distributions, we reduce the performance of our TDOA-based IPS.

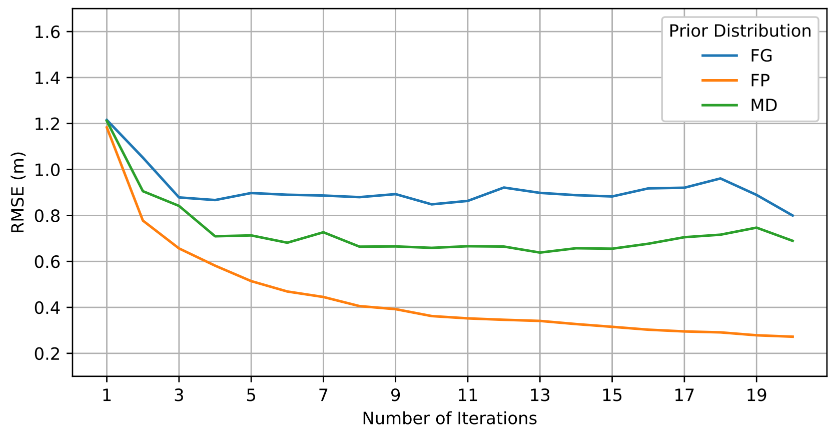

Figure 16. RMSE of the Euclidean distance for different priors.

Fig. 17 depicts the CDF of the mean error at the 20th iteration. The FP scheme shows an average error lower than around 67.8 cm for 99% of the time. Conversely, FG and MD have an error greater than 1m for 24% and 12% of the estimations, respectively. The FP greatly outperforms FG and MD where we can see a significant difference of reliability for estimations with an error lower than 40 cm. This difference happens because the approximate posterior distributions of the target's coordinates using FP scheme has lower KL divergence to the actual posterior distribution than FG and MD.

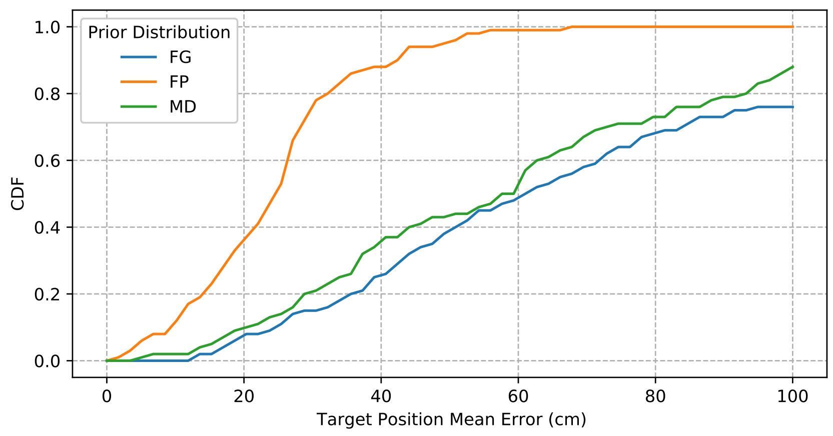

Figure 17. CDF of the mean error for different priors at the 20th iteration.

4.3 Impact of the number of observation per iteration on the algorithm performance

Fig. 18 depicts how the FP-based prior distribution approach and the number of observations or measurements carried out by the APs affects the estimation of stationary target position. Actually, more measurements increase the corresponding evidence and provide more information about all the RVs in the model, and therefore, we can find more precise results. If the number of observations tend to infinity, different prior distributions will asymptotically converge the same posterior distribution [34]. From this figure, we can also observe that the number of observations affects more in the initial iterations where 25 observations in the first iteration resulted in an error of approximately 80 cm higher than 100 observations in terms o RMSE, while at the 20th the error is just approximately 20 cm higher. We observe the same behavior when comparing the CDF of the error mean in the Figs. 19 and 20. Both figures show a better performance when we using 100 observations per iteration but the relative gain of collecting more against fewer measurements reduces with more iterations.

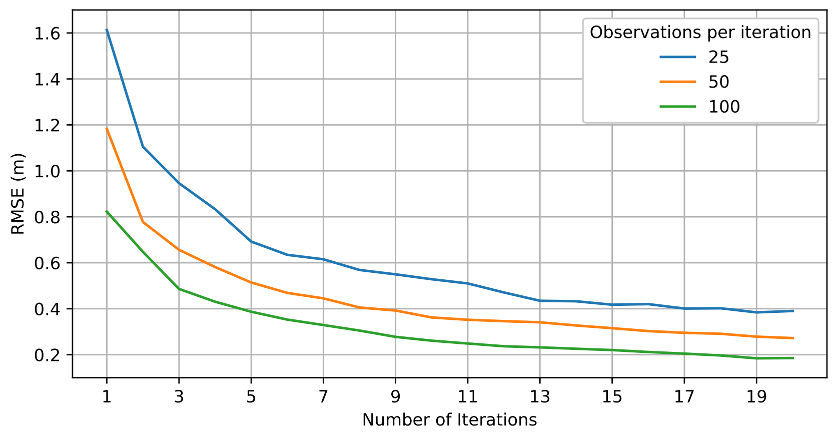

Figure 18. RMSE of the Euclidean Distance for different number of measurements.

Intuitively, the benefit of increasing the number of samples is similar to when we are verifying a histogram of the observations of a random event. The more

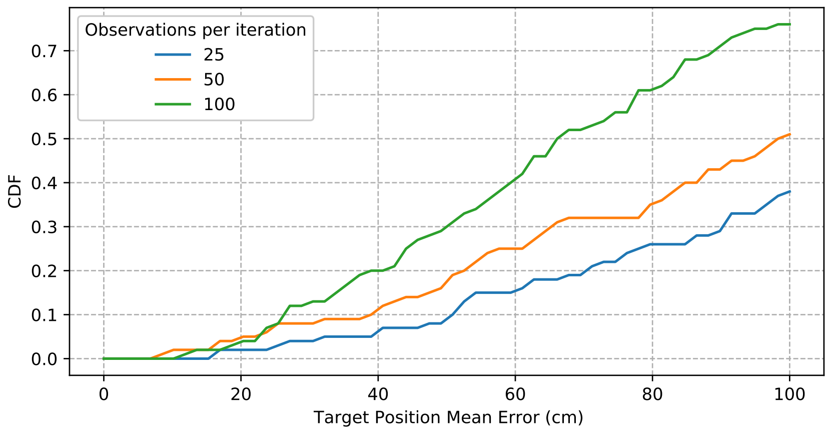

Figure 19. CDF of the mean error for 1 iteration.

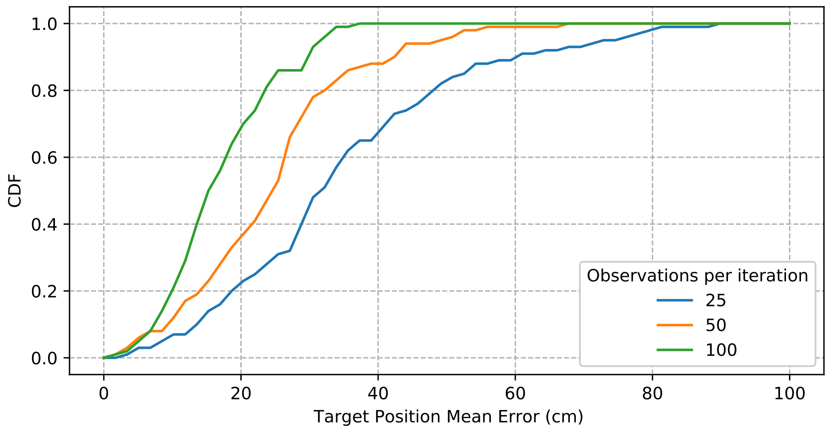

Figure 20. CDF of the mean error for 20 iterations.

observations/evidence we have, the smoother is the curve of the histogram and we can understand how the observations are distributed easier. In our case, it means a better approximation of the unnormalized joint distribution of the model, and the MCMC methods use the unnormalized joint distribution to compute the acceptance probability of the sample candidates of the Markov chain [33]. Therefore, a more accurate unnormalized joint distribution results in a more precise estimation. For HMC and NUTS algorithms, it also means a more accurate computation of the gradient which is used to find the trajectory of the sample candidates [33].

4.4 Bayesian-based tracking mechanism using TDOA measurements

In this section, we extend the Bayesian-based localization mechanism to tracking a moving target. In each iteration, we collect new observations of the target's reference signal, and therefore, we have a new unnormalized joint distribution of the observed event. It means that if we change the target's position during the iterations of our proposed mechanism, the unnormalized joint distribution used by the MCMC method will be based on the current target's position. Fig. 21 illustrates a target moving in circles. The target is moving with constant speed while our proposed mechanism iteratively estimates the target's position to track its trajectory. We assume that the target is slow enough such that the measurements collected in an iteration correspond to the target's reference signal of the same position.

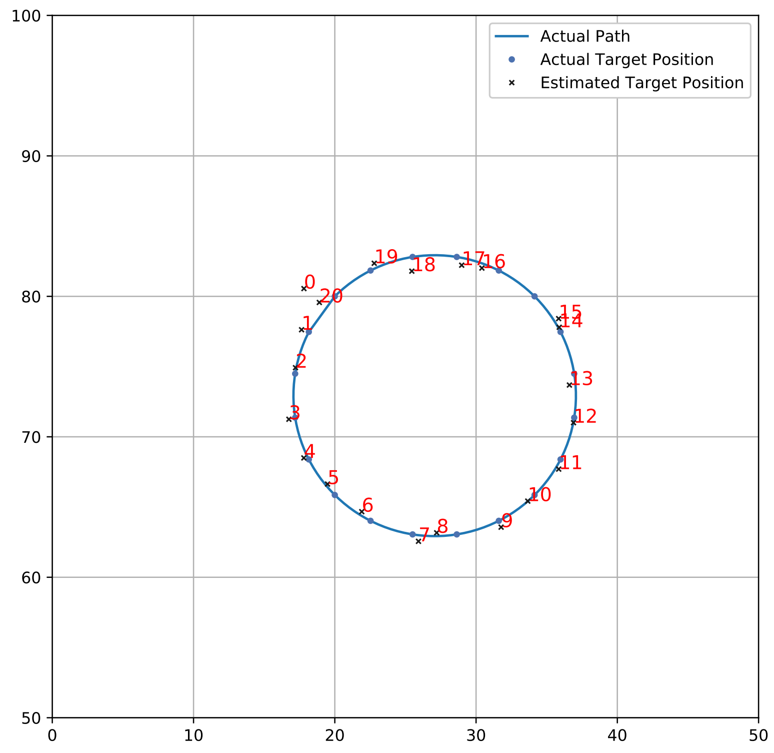

Figure 21. Tracking a target moving in circles.

We evaluate the proposed mechanism by considering just one position update. The initial position of the target is (20, 80) m and the target moves to the position (21, 81) m in two iterations. In the first iteration, we use the flat distributions as priors as in (16), while in the second iteration, the prior distributions are updated based on the posterior distributions of the previous iteration. We consider that the APs collect 50 TDOA measurements in each iteration. Fig. 22 shows the performance of our proposed mechanism in the second iteration for Flat, FG, MD and FP prior distributions. Flat prior distributions are the distributions in (16) where we consider that the target could be in any position of our evaluation scenario. The FP prior distribution has the worse performance so that it would be even better to just use the Flat distribution. Fig. 23 gives intuition on why the FP prior distribution works well for static targets but underperforms for moving ones. We can note the distance between the approximated posterior distribution mean to the actual target's position. The mean and variance of the FP prior distribution does not allow the MCMC sampling algorithm explore the sampling space as it is needed. This mean and variance took from our previous estimation is our bias or beliefs about the RVs which can increase or decrease the estimation error. It is the same problem to the RSS-based IPS for static targets where the bias of the FP prior distribution limits the sampling space of the MCMC method. Therefore, when employing the FP prior distribution, the mean of the approximated posterior distributions of the X and Y coordinates are closer to the initial position (first iteration). The FG and

MD prior distributions have this bias component as well, however, it is less than FP because they have weaker information regarding the posterior distribution. Thus, the MCMC sampling has more freedom to explore the sampling space, while using updated knowledge about the target's position to find more accurate estimations.

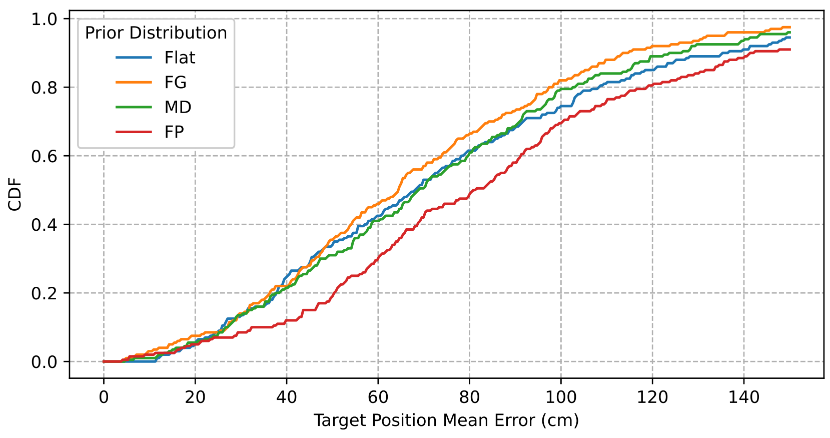

Figure 22. CDF of the mean error of a moving target.

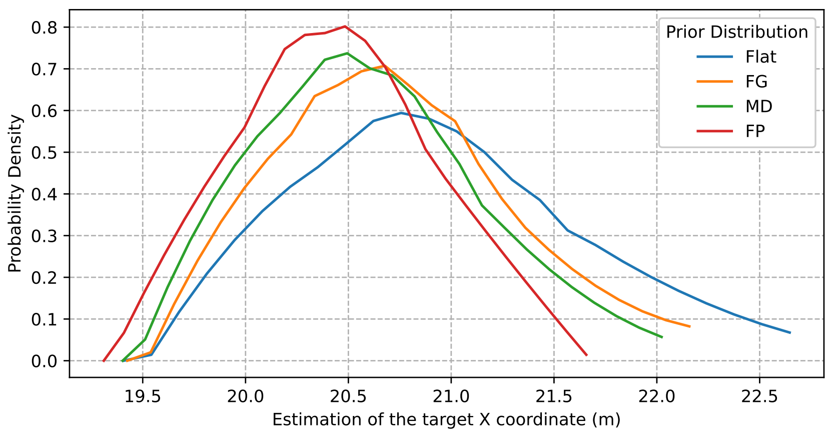

Figure 23. Posterior distributions of the X coordinate of a moving target.

As we aforementioned, we use prior distributions with weaker information when we are not certain how much our current knowledge corresponds to the actual posterior distribution. It means that we should use distributions that are more flat. Therefore, if we update our knowledge about the target's position and it moves afterwards, the correspond prior distribution is not valid anymore. However, we can still use the latest updated knowledge about the target's position and increase uncertainty by using a prior distribution with higher variance. Do to that, FG and MD prior distributions provide better estimations than using completely flat distributions. The problem is to know how uncertain we should be about our previous knowledge because it will depend on the speed of the target, where the faster is the target, the higher should be the variance of the prior distributions.

5 CONCLUSION AND FINAL REMARKS

In this work, we presente an iterative Bayesian-based localization mechanism to accurately estimate the position of the target node in an indoor environment. This method can use different metrics such as TDOA, RSSI and AOA, while exploits any previous knowledge about the experiment by updating the prior distributions in the Bayes' theorem with our latest information regarding the event of interest. During our investigations about MCMC sampling algorithms in our literature review, we found out that the HMC algorithm outperforms the MH and Gibbs sampling algorithms. However, the NUTS algorithm which is an extension of HMC has at least the same performance than the HMC. We use the NUTS algorithm for the TDOA-based localization mechanism and our results show that the proposed iterative method outperforms Bayesian-based localization systems that do not use the latest information about the target's position.

We also observe that the prior distribution greatly affects the estimation of the target's position because the MCMC methods achieve better estimations using prior distributions with lower KL divergence to the actual posterior distribution. However, a prior distribution that limits the MCMC method to travel to the sampling space where the target is actually located will cause erroneous approximated posterior distributions. We observe this behavior by employing the FP distribution with RSS-based IPS or when assessing a moving target. Therefore, the choice of the prior distribution should be done carefully. Moreover, different from RSS-based IPS, the FP prior distribution has the best performance compared to MD and FG in TDOA-based IPS for static targets. It means that a prior distribution model can have different performance for different metrics such as AOA, RSSI and TDOA.

Moreover, the number of measurements collected by the APs per iteration affects the performance of the proposed Bayesian-based positioning system all. Our results show that a higher number of measurements leads to lower estimation errors. For a static target, the TDOA-based localization procedure is more affected by the number of measurements at the first iteration than afterwards. However, it still influences the performance of the last iterations which could be game-changing for many of applications that need IPS with lower estimation errors. Therefore, optimizing the number measurements made by the APs per iteration is a issue that could be addressed in the future.

As shown by our results, the proposed Bayesian-based localization mechanism can also be used for source tracking in indoor deployment scenarios. However, we can not use prior distributions with low variance when the target is moving because our latest knowledge about the target's position does not correspond to the post-movement target's position. Hence, FG and MD prior distributions outperform the FP approach for tracking because their distribution have higher variance than FP scheme. Thus, It would be important to study how uncertainty can be considered when updating our latest knowledge about the target's position as a function of the speed of the target.

6 REFERENCES

[1] Annunziato A. (2015) 5g vision: Ngmn - 5g initiative. In: 2015 IEEE 81st Vehicular Technology Conference (VTC Spring), pp. 1–5.

[2] Ericsson A. (2017) 5g systems enabling the transformation of industry and society. Tech. Rep. .

[3] Durisi G., Koch T. & Popovski P. (2016) Toward massive, ultrareliable, and low-latency wireless communication with short packets. Proceedings of the IEEE 104, pp. 1711–1726.

[4] Shafiq M.Z., Ji L., Liu A.X., Pang J. & Wang J. (2013) Large-scale measurement and characterization of cellular machine-to-machine traffic. IEEE/ACM Transactions on Networking 21, pp. 1960–1973.

[5] Ge X., Yang J., Gharavi H. & Sun Y. (2017) Energy efficiency challenges of 5g small cell networks. IEEE Communications Magazine 55, pp. 184–191.

[6] Dawy Z., Saad W., Ghosh A., Andrews J.G. & Yaacoub E. (2017) Toward massive machine type cellular communications. IEEE Wireless Communications 24, pp. 120–128.

[7] Pokhrel S.R., Ding J., Park J., Park O.S. & Choi J. (2020) Towards enabling critical mmtc: A review of urllc within mmtc. IEEE Access 8, pp. 131796–131813.

[8] Rappaport T.S., Sun S., Mayzus R., Zhao H., Azar Y., Wang K., Wong G.N., Schulz J.K., Samimi M. & Gutierrez F. (2013) Millimeter wave mobile communications for 5g cellular: It will work! IEEE Access 1, pp. 335–349.

[9] Panzner B., Zirwas W., Dierks S., Lauridsen M., Mogensen P., Pajukoski K. & Miao D. (2014) Deployment and implementation strategies for massive mimo in 5g. In: 2014 IEEE Globecom Workshops (GC Wkshps), pp. 346–351.

[10] Abbas R., Shirvanimoghaddam M., Li Y. & Vucetic B. (2017) On the performance of massive grant-free noma. In: 2017 IEEE 28th Annual International Symposium on Personal, Indoor, and Mobile Radio Communications (PIMRC), pp. 1–6.

[11] Yang H., Zhong W., Chen C., Alphones A., Du P., Zhang S. & Xie X. (2020) Coordinated resource allocation-based integrated visible light communication and positioning systems for indoor iot. IEEE Transactions on Wireless Communications 19, pp. 4671–4684.

[12] Kim S., Ha S., Saad A. & Kim J. (2015) Indoor positioning system techniques and security. In: 2015 Forth International Conference on e-Technologies and Networks for Development (ICeND), pp. 1–4.

[13] Macagnano D., Destino G. & Abreu G. (2014) Indoor positioning: A key enabling technology for iot applications. In: 2014 IEEE World Forum on Internet of Things (WF-IoT), pp. 117–118.

[14] Zhao Z., Fang J., Huang G.Q. & Zhang M. (2016) ibeacon enabled indoor positioning for warehouse management. In: 2016 4th International Symposium on Computational and Business Intelligence (ISCBI), pp. 21–26.

[15] Kela P., Costa M., Turkka J., Koivisto M., Werner J., Hakkarainen A., Valkama M., Jantti R. & Leppanen K. (2016) Location based beamforming in 5g ultra-dense networks. In: 2016 IEEE 84th Vehicular Technology Conference (VTC-Fall), pp. 1–7.

[16] Evjen T.Å., Hosseini Raviz S.R., Petersen S.A. & Krogstie J. (2020) Smart facility management: Future healthcare organization through indoor positioning systems in the light of enterprise bim. Smart Cities 3, pp. 793–805. URL: `https://www.mdpi.com/2624-6511/3/3/40`.

[17] Lu S., Xu C., Zhong R.Y. & Wang L. (2017) A rfid-enabled positioning system in automated guided vehicle for smart factories. Journal of Manufacturing Systems 44, pp. 179 – 190. URL: `http://www.sciencedirect.com/science/article/pii/S0278612517300390`.

[18] Lu Y., Richter P. & Lohan E.S. (2018) Opportunities and challenges in the industrial internet of things based on 5g positioning. In: 2018 8th International Conference on Localization and GNSS (ICL-GNSS), pp. 1–6.

[19] Horsmanheimo S., Lembo S., Tuomimaki L., Huilla S., Honkamaa P., Laukkanen M. & Kemppi P. (2019) Indoor positioning platform to support 5g location based services. In: 2019 IEEE International Conference on Communications Workshops (ICC Workshops), pp. 1–6.

[20] Küpper A. (2005) Location-based services: fundamentals and operation. John Wiley & Sons.

[21] Madigan D., Einahrawy E., Martin R.P., Ju W.., Krishnan P. & Krishnakumar A.S. (2005) Bayesian indoor positioning systems. In: Proceedings IEEE 24th Annual Joint Conference of the IEEE Computer and Communications Societies., vol. 2, vol. 2, pp. 1217–1227 vol. 2.

[22] Kanhere O. & Rappaport T.S. (2018) Position locationing for millimeter wave systems. In: 2018 IEEE Global Communications Conference (GLOBECOM), pp. 206–212.

[23] Alamu O., Iyaomolere B. & Abdulrahman A. (2021) An overview of massive mimo localization techniques in wireless cellular networks: Recent advances and outlook. Ad Hoc Networks 111, p. 102353. URL: `http://www.sciencedirect.com/science/article/pii/S1570870520307009`.

[24] Dammann A., Raulefs R. & Zhang S. (2015) On prospects of positioning in 5g. In: 2015 IEEE International Conference on Communication Workshop (ICCW), pp. 1207–1213.

[25] Wong C., Klukas R. & Messier G.G. (2008) Using wlan infrastructure for angle-of-arrival indoor user location. In: 2008 IEEE 68th Vehicular Technology Conference, pp. 1–5.

[26] Khan A., Wang S. & Zhu Z. (2019) Angle-of-arrival estimation using an adaptive machine learning framework. IEEE Communications Letters 23, pp. 294–297.

[27] Yang Z., Zhou Z. & Liu Y. (2013) From rssi to csi: Indoor localization via channel response. ACM Computing Surveys (CSUR) 46, pp. 1–32.

[28] Sugano M., Kawazoe T., Ohta Y. & Murata M. (2006) Indoor localization system using rssi measurement of wireless sensor network based on zigbee standard. Wireless and Optical Communications 538, pp. 1–6.

[29] Bocquet M., Loyez C. & Benlarbi-Delai A. (2005) Using enhanced-tdoa measurement for indoor positioning. IEEE Microwave and wireless components letters 15, pp. 612–614.

[30] Makki A., Siddig A., Saad M., Cavallaro J.R. & Bleakley C.J. (2015) Indoor localization using 802.11 time differences of arrival. IEEE Transactions on Instrumentation and Measurement 65, pp. 614–623.

[31] Mazuelas S., Bahillo A., Lorenzo R.M., Fernandez P., Lago F.A., Garcia E., Blas J. & Abril E.J. (2009) Robust indoor positioning provided by real-time RSSI values in unmodified WLAN networks. IEEE JSTSP 3, pp. 821–831.

[32] Lauritzen S.L. (1996) Graphical models, vol. 17. Clarendon Press.

[33] Bishop C.M. (2006) Pattern recognition and machine learning. springer.

[34] Martin O. (2016) Bayesian Analysis with Python. Packt Publishing Ltd.

[35] de Lima C.H.M., Saloranta J. & Latva-aho M. (2019) Collaborative positioning mechanism using bayesian probabilistic models for industry verticals. In: Proceedings of 2018 16th International Symposium on Wireless Communication Systems (ISWCS).

[36] Hilleshein H., Carlos de Lima H.M., Alves H. & Latva-aho M. (2020) Iterative bayesian-based localization mechanism for industry verticals. In: 2020 IEEE 91st Vehicular Technology Conference (VTC2020-Spring), pp. 1–5.

[37] Hilleshein H., Carlos de Lima H.M., Alves H. & Latva-aho M. (2020) Iterative bayesian-based localization mechanism with mixture distribution. In: 2nd 6G Wireless Summit (6G SUMMIT). 17 March 2020, pp. 1–2. URL: `http://urn.fi/urn:nbn:fi-fe2020052538854`.

[38] Betancourt M. (2018), A conceptual introduction to hamiltonian monte carlo.

[39] MacKay D.J. & Mac Kay D.J. (2003) Information theory, inference and learning algorithms. Cambridge university press.

[40] Chong A. & Lam K.P. (2017) A comparison of mcmc algorithms for the bayesian calibration of building energy models. In: Proceedings of the 15th IBPSA Building Simulation Conference, vol. 4, vol. 4.

[41] Hoffman M.D. & Gelman A. (2014) The no-u-turn sampler: adaptively setting path lengths in hamiltonian monte carlo. J. Mach. Learn. Res. 15, pp. 1593–1623.

[42] Koller D. & Friedman N. (2009) Probabilistic graphical models: principles and techniques. MIT press.

[43] Friedman N., Geiger D. & Goldszmidt M. (1997) Bayesian network classifiers. Machine Learning 29, pp. 131–163.

[44] Alhammadi A., Alraih S., Hashim F. & Rasid M.F.A. (2019) Robust 3d indoor positioning system based on radio map using bayesian network. In: 2019 IEEE 5th World Forum on Internet of Things (WF-IoT), pp. 107–110.

[45] Kroese D.P., Taimre T. & Botev Z.I. (2011) Handbook of Monte Carlo methods / Dirk P. Kroese, Thomas Taimre, Zdravko I. Botev. Wiley New Jersey, xix, 743 p. : p.

[46] Kroese D.P., Brereton T., Taimre T. & Botev Z.I. (2014) Why the monte carlo method is so important today. Wiley Interdisciplinary Reviews: Computational Statistics 6, pp. 386–392.

[47] Ross S.M., Kelly J.J., Sullivan R.J., Perry W.J., Mercer D., Davis R.M., Washburn T.D., Sager E.V., Boyce J.B. & Bristow V.L. (1996) Stochastic processes, vol. 2. Wiley New York.

[48] Kullback S. & Leibler R.A. (1951) On information and sufficiency. Ann. Math. Statist. 22, pp. 79–86. URL: `https://doi.org/10.1214/aoms/1177729694`.

[49] Hastings W. (1970) Monte carlo sampling methods using markov chains and their applications. Biometrika 57, pp. 97–109.

[50] Kass R.E., Carlin B.P., Gelman A. & Neal R.M. (1998) Markov chain monte carlo in practice: a roundtable discussion. The American Statistician 52, pp. 93–100.

[51] Neal R.M. (1998) Suppressing random walks in markov chain monte carlo using ordered overrelaxation. In: Learning in graphical models, Springer, pp. 205–228.

[52] Salvatier J., Wiecki T.V. & Fonnesbeck C. (2016) Probabilistic programming in python using pymc3. PeerJ Computer Science 2, p. e55.

[53] So H.C. (2011) Source Localization: Algorithms and Analysis, John Wiley & Sons, Ltd, chap. 2. pp. 25–66.

[54] Retscher G. & Tatschl T. (2016) Indoor positioning using wi-fi lateration — comparison of two common range conversion models with two novel differential approaches. In: 2016 Fourth International Conference on Ubiquitous Positioning, Indoor Navigation and Location Based Services (UPINLBS), pp. 1–10.

[55] Friedman N. & Goldszmidt M. (1997) Sequential update of bayesian network structure. In: Proceedings of the Thirteenth Conference on Uncertainty in Artificial Intelligence, UAI'97, Morgan Kaufmann Publishers Inc., San Francisco, CA, USA, pp. 165–174.

[56] PyMC Development Team (2018), PyMC3 documentation. URL: `https://docs.pymc.io/`, accessed: January 20th of 2020.