II CONVENCIÓN CIENTÍFICA INTERNACIONAL " CCI UCLV 2019 "

XVIII Simposio de Ingeniería Eléctrica SIE-2019

Simulación de algoritmos de posicionamiento basados en RSSI para redes inalámbricas utilizando ns-3

(Simulation of RSSI-based positioning algorithms for wireless networks using ns-3)

Jannyna Etter-Olguín [a], Cristian Duran-Faundez [a], Jaime Rohten [a], Richard Seguel-Cárdenas [a], Ivan Santana [b]

[a]*Universidad del Bío-Bío, Concepción, Chile*

[b]*Universidad Central de Las Villas, Santa Clara, Cuba*

Abstract

Problematic: Local positioning systems are interesting for applications where the Global Positioning System (GPS) is considered unsuitable. Received Signal Strength Indicator (RSSI)-based systems are desirable as they represent an inexpensive way to provide localization, although their limitations. For validation, network simulators are required, being ns-3 one of the most noted because of its characteristics. However, ns-3 includes no classes specifically developed to simulate positioning algorithms. Objectives: The objective of this paper is to report the development of new classes and methods allowing local positioning systems simulation, discussing various methodological/practical issues present. Methodology: We present a literature study on various localization techniques, briefly presenting architectural and modeling concepts on RSSI-based positioning systems. Then, we describe the implementation of software classes for ns-3. Simulation results are also shown for two implemented algorithms, literature versions of Multilateration and Fingerprinting, in various different scenarios. Results and discussion: Experimental results with the implemented classes are satisfactory. The simulation results confirm the known problems of RSSI for providing precise estimations. On the other side, the obtained results are coherent with those reported in other publications obtained by real experimentation. Conclusion: As ns-3 is widely used, the developments reported in this paper could benefit people working on related subjects. Allowing RSSI-based positioning is not trivial. The simulations allow to realize the existing practical problems, such as those related to the variations of RSSI, coverage, and other. Also, simulation is a useful resource to make further contributions on the subject.

Palabras clave: Network simulation; Positioning; Ad hoc networks; Tracking; RSSI

Email address: `crduran@ubiobio.cl` (Cristian Duran-Faundez).

.

1 Introduction

The automatic positioning of people and assets of the tangible world is a highly desirable capacity for many state-of-the-art systems and services (Bensky, 2008). Location-aware systems have gained interest from science and industry, presenting many open challenges still unsolved (Lymberopoulos et al., 2015). Positioning concerns to multiple application fields, offering new possibilities for cargo traffic management, robotics, security and manufacturing (Taylor et al., 2001; Bischoff et al., 2006; Dobre and Bajic, 2008; Silva-Faundez et al., 2015) (among many others), but it also serves for enhancing network connectivity through location-aware communication protocols (Grover et al., 2014; Molina-Martínez et al., 2018). To that, various technologies provide positioning with highly acceptable precisions, where the Global Positioning System (GPS) is the better known and covers most of the application cases. However, multiple scenarios exclude the GPS as a suitable solution, as the many cases requiring indoor and underground localization. Those scenarios present many nontrivial difficulties, due to the presence of obstacles and other factors in the application environments (Dedes and Dempster, 2005; Lymberopoulos et al., 2015). Current accurate solutions for indoor and local positioning include computer vision localization (Bais et al., 2006), near-field sensors (such as lasers, magnetic or ultrasonic) (Escudero et al., 2007) and, more recently, the use of Ultra-Wide Bands (UWB) technologies (Jiménez Ruiz and Seco Granja, 2017).

This paper focuses on radio-frequency (RF) based positioning, in particular, in the positioning of devices connectible to wireless communication networks. Wireless networks are present in almost every place with human presence. This proliferation is due to their multiple advantages such as ease of installation, flexibility and their capacity to allow mobility without dragging cables. Most implemented wireless networks are oriented to provide Internet connectivity to user-oriented mobile devices, like laptops and smart-phones, but there are also numerous technologies oriented to communicate things from the tangible world to provide remote monitoring or control, which is the case of Wireless Sensor and Actuator Networks (WSAN) (Verdone et al., 2008). These technologies open many possibilities for the industry of the future (Preuveneers and Ilie-Zudor, 2017), smart cities (Nandury and Begum, 2015), and the Internet of Things (Gubbi et al., 2013). In the matter of wireless networks with positioning systems, many works propose the use of the Received Signal Strength Indicator (RSSI) as an input for distance or position estimation. Indeed, even if highly imprecise in comparison with specific-purpose systems designed to use Time-of-Flight (ToF) or Angle-of-Arrival (AoA), RSSI-based systems are still interesting because they represents an inexpensive viable way to provide location information (Duran-Faundez et al., 2010). Indeed, while ToF or AoA require specific and relatively expensive hardware and infrastructure, RSSI is inherent in most wireless modules used in wireless networking. As so, it can be used with almost any device connectible by WiFi, Bluetooth, or other communication technologies. Examples of RSSI-based locations systems are mentioned by Lee and Buehrer (2011).

As local positioning with wireless devices is still an open subject, new protocols and techniques are required. For analysis and validation, practical studies of wireless positioning are possible (e.g., the work of Alkasi et al. (2013)), but with limited scale and fixed environments. For large-scale, multiparameter and repeatable experimentation and validation, simulation tools must be used. In this paper, we tackle the problem of simulating positioning with wireless devices with ns-3 [1] (Riley and Henderson, 2010). ns-3 is

[1] https://www.nsnam.org/

a network simulator highly used by researchers and engineers, because of its computational performance, accuracy, and model libraries availability, and recognized for being the successor of ns-2 (another highly recognized network simulator) (ur Rehman Khan et al., 2013). Despite its potential, ns-3 is not being widely used for simulating location-aware systems, which can be due to the complexity of the coding process (relative to other simulation tools). Some works have reported analysis of positioning algorithms with ns-3, as the works of Ojha and Misra (2013) and Sobehy et al. (2019), but we did not find details about the programming process or the simulation structures.

In this paper, we report the development of two modules for ns-3, able to provide positioning of nodes in wireless networks, implementing two different positioning techniques generally used as a basis for location-aware services: multilateration and fingerprinting. We focus on RSSI-based calculation, but the developments made can be easily extended to other kinds of indicators (like ToF). In particular, the contributions of this works are twofold: (1) First, we describe a complete development chain for simulating RSSI-based location-aware techniques with multilateration and fingerprinting methods in ns-3, discussing methodological and technical issues. (2) Second, we propose a technique for identifying path-loss model parameters, based in the Least Squares approximation. Such a technique can be used in both simulated and real applications.

2 Principles of RSSI-based positioning

Various kinds of location-based sevices can take advantage of some information about the absolute or relative position of a person or an object (Karl and Willig, 2005). Of course, different kinds of location procedures call for particular technologies, as they are implemented in different environments and they provide different levels of precision (Lee and Buehrer, 2011). We focus on local position systems where an absolute or relative bi-dimensional coordinate (i.e., some (x,y) reference to the position of a node) of a static or mobile node must be determined. We consider coordinates calculation in a bi-dimensional space, but an extension to a three-dimensional space is trivial. The considered positioning system considers that the node to be located carries a wireless module capable of executing data packets transmissions or receptions. In general, current location systems include elements performing three kinds of roles:

- Unknown node (U): A wireless node of unknown position whose coordinates are intended to be estimated by the positioning system. Usually, a practical application may include several unknown nodes, but, for simplicity purposes, we consider only a single one in our explanations. Extending the scale of the unknown nodes set is also trivial.
- Collector node (C): The unit in charge of gather the location-related data and calculate the position estimates. In practice, this role could be executed by several physical elements.
- Beacons (B_i): A set of nodes with known positions. Beacons' coordinates are used by the Collector as references to estimate the unknown node's coordinate.

The way in which those nodes are distributed and exchange information can differ. Actually, a determined device could execute more than one role (e.g., a typical GPS navigators consider that the Collector and the Unknown nodes are the same navigation unit, while the beacons are the GPS satellite constellation).

A positioning system could work in several ways. For example, the Collector node could be the unit which desires to calculate its own coordinate (as the case of a GPS navigator), or the Collector could be part of an information infrastructure whose objective is to inform to users about the location of other units (persons or assets) in an environment. The Collector uses information obtained from a series of signals or messages sent between U and a Beacons subset, and exploit some properties of those signals and known information (e.g., the Beacons' coordinates) to make the calculations.

In this paper, we adopt as the basis for the positioning calculation RSSI values measured from packages sent between U and a set of Beacons. RSSI is provided by many wireless modules as an approximation of the Received Signal Strength (RSS), which is a magnitude of the power present in a received radio signal. RSS corresponds to the power transmission of the sender of a signal, affected by many factors such as the the distance of the sender to the receiver. Subsection 2.1 gives some insights about RSS related models, which can be used for simulation and real-world identification, while Subsection 2.2 describes basic techniques for using RSSI to calculate the position of a wireless node, in particular the techniques applied further in this work: multilateration and fingerprinting.

2.1 Modeling path loss

Typically, the received power is represented by the Friis free space propagation model (Friis, 1946), which is a function of the transmitted power, the wavelength, the distance between the transmitter and the receiver and the gains of the transmitter and receiver antennas. Friis model represents an ideal scenario. In practice, many factors can affect the actually perceived RSS, being the main sources of power variations: multipath effects and attenuation by absorption, reflection, scattering, and diffraction (Laaraiedh et al., 2009), due to obstacles and particles in the medium. Of course, the impact of those effects are higher at higher distances. Such a variations are difficult to mathematically represent and identify, thus, most of the models adopted for simulation and analysis are stochastic functions including, at least, one random variable. One very often adopted model is Log Normal Shadowing (LNS) path loss model, which says that the average signal level decreases with the logarithm of the sender-receiver distance, regardless of whether the environment is indoor or outdoor (Rappaport, 2002). This noted model is described by Equation (1):

$$L(dB) = L_0 + 10.\gamma.\log\left(\frac{d}{d_0}\right) + X_\sigma \tag{1}$$

where L is the path loss at a distance d, L_0 is the path loss at a reference distance d_0, γ is the path loss exponent, and X_σ is a zero-mean Gaussian random variable with standard deviation σ. X_σ represents the variations (shadowing) described above.

Besides LNS, other models have been proposed, more adjusted to indoor or outdoor environments. Specific path loss models for indoor environments are the Itu-R 1238 (a recommendation of the International Telecommunication Union (ITU) (2017a) [2]), and the Motley-Keenan model (Motley and Keenan, 1988). For outdoor environments, Itu-R 1411 Line-of-Sight by the International Telecommunication Union (ITU) (2017b), the Okumura model (Okumura et al., 1968; Rappaport, 2002) and the Two-ray model.

[2] www.itu.int

2.2 Positioning Techniques

Many positioning techniques have been proposed in the literature. Even if less traditional techniques exist, most position techniques can be categorized in (1) analytical range estimation-based approximations and (2) pattern database comparison (Bensky, 2008). For the first case, distances between the Unknown node and Beacon nodes are estimated using some mathematical or empirical model (e.g., LNS), and the coordinate is finally calculated by the Collector by applying some geometrical or other kind of mathematical technique. For the second case, data registered at an off-line stage (generally, RSSI statistical values associated with their corresponding Beacon nodes) are compared with the on-line incoming measures, by the Collector, which tries to find the best match of the new situation with a previously stored one (this is generally called fingerprinting or pattern recognition). In this paper, we test two classical techniques representative of those categories: for the first case, a multilateration technique which considers distances estimates errors (Subsection 2.3) and, for the second case, a nearest neighbor fingerprinting algorithm is implemented (Subsection 2.4).

2.3 Multilateration

Multilateration is a technique which allows to calculate the positioning by using the distances between the sender (U) and the receivers (Beacons). Actually, in a real application, those distances are estimations which have always errors, so that it is necessary to count on more than three receivers to get a more precise estimation. We adopt the multilateration technique described by Karl and Willig (2005), which solves the positioning with distance errors, and works as follows: Considering distances estimates to n beacon nodes (with $n \geq 4$), we can form the systems of equations represented in matrix form by Equation (2).

$$2.\begin{bmatrix} x_n - x_1 & y_n - y_1 \\ \vdots & \vdots \\ x_n - x_{n-1} & y_n - y_{n-1} \end{bmatrix} . \begin{bmatrix} x_u \\ y_u \end{bmatrix} = \begin{bmatrix} (r_1^2 - r_n^2) - (x_1^2 - x_n^2) - (y_1^2 - y_n^2) \\ \vdots \\ (r_{n-1}^2 - r_n^2) - (x_{n-1}^2 - x_n^2) - (y_{n-1}^2 - y_n^2) \end{bmatrix} \tag{2}$$

where (x_u, y_u) is the coordinate of the Unknown node U, (x_i, y_i) is the (known) coordinate of the Beacon B_i, and r_i is the estimated Euclidean distance from U to the Beacon B_i. The solution of this equation can be obtained by expressing it as a matrix equation in the form:

$$\mathbf{A}^{\mathrm{T}}.\mathbf{A}.\mathbf{x} = \mathbf{A}^{\mathrm{T}}.\mathbf{b} \tag{3}$$

where $\mathbf{A}$ is the left side matrix of Equation (2) multiplied by 2, $\mathbf{x} = [x_u, y_u]^{\mathrm{T}}$, and $\mathbf{b}$ is the right side matrix of Equation (2). Finally, Equation (3) can be solved by QR or Cholesky factorization (we adopted the last method in our experiments).

2.4 Fingerprinting

Fingerprinting represents a group of algorithms for pattern recognition which require a set of RSSI measures first stored in a database to estimate a position. It is executed hence in two stages: an off-line and an on-line stage. We adopt the fingerprinting technique described by Bensky (2008), which works as follows:

2.4.1 Off-line stage

During the off-line stage, a determined amount of reference points must be selected. The reference points are fixed places at which a transmission unit is located to broadcasts data packets (as the U node should do during the on-line stage). Those data packets are received by Beacon nodes, which send the RSSI-based data to be stored in a database. By doing so, the fingerprinting database contains particular characteristics of the different places where packets were transmitted (reference coordinates and RSSI measures from different Beacons). Bensky (2008)'s procedure says that the transmitter must broadcast a determined number of packets (at a certain interval of time) from a reference point from four different orientations (e.g., "north", "south", "east", "west"), as a way to store more information. Then, the Beacons (capable of receive those messages) calculate and communicate the average of the set of RSSI measures from a particular orientation so the following tuple is registered $V_p = [x_p, y_p, \rho, \widehat{s}_{p,1}, \widehat{s}_{p,2}, \ldots, \widehat{s}_{p,n}]^{\mathrm{T}}$, where (x_p, y_p) is the coordinate of the reference point p, ρ is the orientation, and $\widehat{s}_{p,i}$ are the average of the RSSI values calculated by the Beacon node i from all the captured packets sent by U from position p at orientation ρ. The values of $\widehat{s}_{p,i}$ are normalized by making $\widehat{s}_{p,i} = \widehat{s}_{p,i} - \widehat{s}_{p,1}$. Finally, the normalized tuple is stored in the database.

2.4.2 On-line stage

As in multilateration, in fingerprinting on-line stage (i.e., where the location-based service is working), the U node broadcasts messages which are received by a set of Beacon nodes. Those beacons retrieve the RSSI from the incoming packets and send (at regular intervals) the RSSI average to the Collector node.

Let us denote PT as the vector of average RSSI retrieved values from a set of messages broadcasted by node U, $PT = [s_1, s_2, s_3, \ldots, s_n]^{\mathrm{T}}$, where s_i is the average RSSI calculated by the beacon node i, and PTN as the normalized vector $PTN = [0, s_2 - s_1, s_3 - s_1 \ldots, s_n - s_1]^{\mathrm{T}}$. For each vector V_p previously stored, a distance to the on-line measures D_p can be calculated by doing: $D_p = \sqrt{\sum_{i=1}^{n} (PTN_i - \widehat{s}_{p,i})^2}$, where PTN_i is the ith value stored in PTN.

Finally, the Unknown node coordinate is calculated by averaging the coordinates of the k reference points with the least distance D_p, with k a selectable parameter.

3 ns-3 Implementation

For the case of this paper, we consider a positioning scenario where the nodes execute the following protocol:

(1) The Unknown node broadcasts data packets at specified time intervals,
(2) The packets are received by a set of Beacons (depending on the Beacons topology, the position of the Unknown node and of the environment, some Beacons could be out of the communication range of the Unknown node),
(3) Beacon nodes retrieve RSSI values of the incoming packets, which are retransmitted in the payload of new data packets to the Collector node.

The described scenario is illustrated in Figure 1.

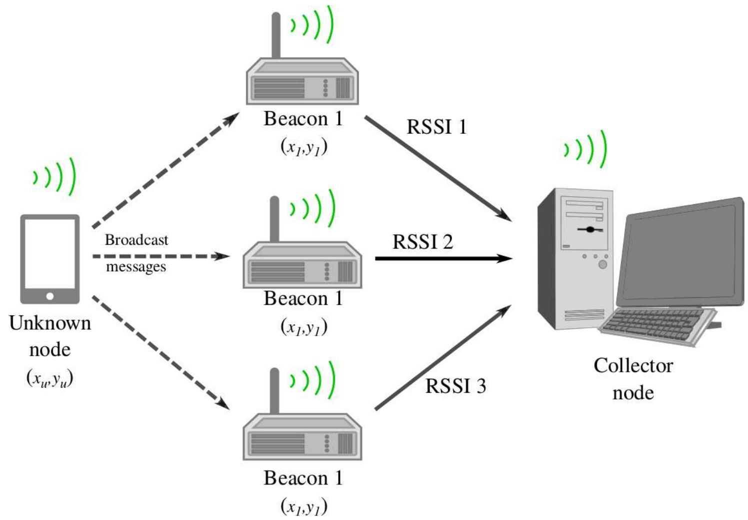

Figure 1. Considered positioning scenario. The Unknown node broadcasts data packets at regular times, Beacon nodes receiving those packets and send the corresponding RSSI values to the Collector node.

Finally, we consider that the Beacon nodes conform an Ad hoc wireless network with the Collector node, and execute the Ad hoc On-Demand Distance Vector Routing (AODV) protocol (Perkins et al., 2003).

Simulating the positioning scenario in ns-3 calls to solve various sub-implementations: (1) the network topology and mobility models, (2) path loss, (3) the creation of events for traffic generation, and (4) the positioning algorithms. From there, points (1)-(3) are solved by ns-3 included modules; point (4) requires for an ad hoc implementation, which is the subject of this paper. The following subsections explain the development of the simulation scenarios.

3.1 General definitions of the simulated scenarios

As usual when simulating with ns-3, general definitions of the scenario are implemented at the beginning of the `main` function. Actually, network layers selection or the physical distribution of the the considered nodes is not relevant for the positioning algorithms implementation, although they can severely impact the system results; many choices in the definition of the simulated scenarios are made, by simplicity, adopting the modules with the most examples available.

Except for the application layer (which depends on the role to be executed) all nodes are created sharing the same layers. We implemented the positioning system as to be implemented with WiFi compliant devices. Physical objects are simulated with the YansWifiPhy and YansWifiPhyHelper classes. The power output for all the nodes is set to the YansWifiPhy maximum available transmission power which is 16.0206 dbm. The WiFi channel is implemented with the YansWifiChannelHelper. The propagation delay is set to be the usual constant (ConstantSpeedPropagationDelayModel); the propagation loss model is also set (we describe this in Section 3.2). For the Medium Access Control (MAC) sublayer, the WifiMacHelper class is adopted; an AdhocWifiMac network is defined as the network type. The WifiHelper class allows (among other options) to set the WiFi standard; we selected the IEEE 802.11b (WIFI_PHY_STANDARD_80211b) as the communication technology. Beacons and unknown nodes are created in independent containers, one for each type, so it is easy to handle them in the code.

With the MobilityHelper class, mobility models can be associated to each node. The "ns3::ConstantPositionMobilityModel" is installed into the beacon nodes and the Collector node, while the "ns3::ConstantVelocityMobilityModel" is installed in the Unknown one.

As said above, the Beacon nodes and the Collector conform an Ad hoc wireless network, capable of routing packets through the AODV protocol. The AodvHelper class is used to implement this function. In addition, IPv4 is selected as the Network Layer protocol (classes Ipv4AddressHelper and Ipv4InterfaceContainer). For data packets transmissions and receptions, User Datagram Protocol (UDP)-based sockets are defined in all the nodes. The network traffic can be programmed by using the Schedule() method of the Simulator class. By doing so, the Unknown node can broadcast packets at determined intervals.

ns-3 has the possibility to define a 3-dimensional space, but (as said) we only work in 2D; all the nodes are defined to be at 1.5m over the ground. The beacon nodes are located in a grid-like topology, with positions starting from coordinate $(0.0, 0.0, 1.5)$ and increasing in regular intervals in the x and y axes until the maximum defined coordinate. Initial positions for the Unknown and the Collector are also given.

3.2 About the loss path model

One main component for simulating RSS-based positioning are the loss path models. ns-3 includes various attenuation models, which are represented for some subclass of the PropagationLossModel base class. Such a class allows to calculate the received power from a transmission power and the path loss provided by one of its subclasses. Examples of those subclasses are: FriisPropagationLossModel, NakagamiPropagationLossModel, JakesPropagationLossModel, and OkumuraHataPropagationLossModel [3]. The selection of the adopted model depends on the characteristics of the scenarios to simulate. In this paper, we use (directly or not) the following propagation models: LogDistancePropagationLossModel, BuildingsPropagationLossModel, HybridBuildingsPropagationLossModel, ItuR1411LoSPropagationLossModel, and ItuR1238PropagationLossModel. Actually, most of our simulations are made with the HybridBuildingsPropagationLossModel, which is composed of various other models, to obtain the path loss of modules communicating at frecuencies from 200 to 2600MHz, in different indoor and outdoor scenarios (including presence of buildings and other). Indeed, different parameters of this model allow to

[3] See the list of available propagation models at: https://www.nsnam.org/doxygen/group__propagation.html

define simulation attributes like the type of environment ("Urban", "SubUrban", "OpenAreas") or the size of a city. HybridBuildingsPropagationLossModel will call to the appropriate model depending on the parameters context.

3.3 Simulating positioning algorithms

As said above, ns-3 does not include a class for simulating positioning algorithms. We developed three classes for such a task: MultilaterationPositioning and FingerprintingPositioning. Both algorithms share a series of similar tasks, which can be included as methods and attributes from a generic superclass. The main common methods and attributes of MultilaterationPositioning and FingerprintingPositioning are summarized in Table 1.

The next subsections describe the events creation and the implementation of the positioning algorithms.

3.3.1 Events Scheduling

ns-3 implements the Simulator class whose functionalities allow, among other multiple things, create events through the time by using the Schedule() method. The events defined (by the CreateEvents() method) are the following:

- Event that generates the traffic between the Unknown node and the Beacons. Invokes the GenerateTraffic() method, which makes the *U* node to broadcast messages at regular intervals.
- Event that calculates the position. Invokes the GenerateCalcPos(). The position calculations are made at certain intervals of time, after the Beacon nodes have the time to receive a certain amount of packets from the Unknown node.
- Event that generates the files with the positions of the Beacons and the Unknown node. It is executed after the last position estimation. It invokes the CreateCommaSeparatedFile() method.

3.3.2 Position Estimation with Multilateration (MultilaterationPositioning)

Multilateration is executed by the implemented MultilaterationPositioning class. The Unknown node broadcasts packets at regular intervals. As said, it can happen that only a subset of nodes receive the messages, for what we define that the Multilateration algorithm considers only the nodes that receive the last broadcasted packets after a series of tries. The method that makes the Unknown node to broadcast the packets is GenerateTraffic(). We arbitrarily define a traffic of 5 packets each 6 seconds.

When the Beacon nodes receive packets from the Unknown node, they calculate the received power. To that, two methods are used: The ReceivePacket() method is executed when a new packet is received by the Beacon node. It then calls the next method. The CalcRxPower() method returns the received power. Is a method from the PropagationLossModel base class, whose subclasses represent some path loss model available.

Table 1
Main common methods and attributes of the developed classes `MultilaterationPositioning` and `FingerprintingPositioning`.

Methods	
Declaration	**Description**
`void SetSocketAddr (Ptr<Socket> socket, Ipv4Address addr);`	Associates the IP address of a beacon node to its socket.
`void SetAddrMobModel (Ipv4Address addr, Ptr<MobilityModel> mm);`	Associates the IP address of a beacon node to its mobility model.
`void SetTxMob (Ptr<ConstantPositionMobilityModel> mmTx)`	Assigns the mobility model from the Unknown node as an attribute.
`void SetSocketRcv (Ptr<Socket> socket);`	Allows each beacon node to receive a packet through its socket.
`void ReceivePacket (Ptr<Socket> socket);`	In this method, the socket of a beacon node receives a packet. After that, the beacon retrieves the RSSI.
`void CreateEvents (Ptr<Socket> socket, uint32_t pktSize, uint32_t nPacks, Time tIntervalPkt, double tStart, double tIntervEv, double tSim, uint32_t nCalcPos, double initCalcPos, double interCalcPos, double tIntervEvSend, double tStartProm, double tIntervProm, uint32_t nPack);`	Method where events are created.
`double CalcDistance (double rss);`	This method returns the estimated distance between the Unknown node (sender) and each beacon node according to a path loss model (ITU-R 1238 if indoor and ITU-R 1411 LoS if outdoor).
`void GenerateCalcPos (uint32_t calcCount, Time calcInterval);`	Function called by events to calculate the position of the Unknown node.
`void CreateCommaSeparatedFile ();`	This method generates a .txt file containing: the time at which a position is calculated, the coordinate of the actual position of the Unknown node, and the estimated coordinate. This method is called by an event only once.

Attributes	
Declaration	**Description**
`struct anchors{ Vector3D pos; double distance; double tiempo; double xTreal; double yTreal; double xTest; double yTest; double distEuclid; };`	Struct that stores information about a beacon node, such as: its position, the distance of the respective beacon and the Unknown node, the power received from the Unknown node, and other.
`std::map<Ptr<Socket>, Ipv4Address> m_socketAddresses;`	A `std::map` container which maps the IP address of a beacon node, using as key the socket handler of the respective beacon.
`std::map<Ipv4Address, Ptr<MobilityModel>> m_addrMobModel;`	A `std::map` container which maps the mobility models of the beacon nodes, using as key their IP addresses.
`std::map<Ipv4Address, anchors> m_addrAnc;`	A `std::map` container which maps the coordinate and the distance to the Unknown node of the beacon nodes, using as key their IP addresses.
`Ptr<ConstantVelocityMobilityModel> txMob;`	Refers to the mobility model of the Unknown node. It is an instance of the class `MobilityModel` which, depending if the Unknown node is mobile or static, indicates the use of the subclasses `ConstantVelocityMobilityModel` or `ConstantPositionMobilityModel`, respectively.
`std::vector<anchors> v_posTx;`	Stores information about each Beacon node.
`anchors ancTx;`	Structure used for storing the time and positions –real and estimated – of the Unknown node in the .txt file.

Multilateration Equations (2) and (3) requires distances estimates between the Beacons and the Unknown node. In a real-world application, a reference path loss model is required to make the estimations of the distances taking as reference RSSI values. We consider `HybridBuildingsPropagationLossModel` as the "real-world"path loss. Estimating the distances with the simulated models is easy, so we choose a different equation as the reference model. We adopt Log Distance model described by Equation (4) which is derived from Equation (1).

$$P_{rx} = p_0 - 10.\gamma.\log\left(\frac{d_{ij}}{d_0}\right) \tag{4}$$

Here, P_{rx} is the received power (dB), and p_0 is the received power at a reference distance d_0. It suggest that, for estimating distances, it is sufficient to express Equation (4) as a function for calculating d_{ij} as a function of P_{rx} which will be the calculated RSSI values. The equation is still easer to calculate if we choose $d_0 = 1$, but we still need a decent estimation for γ. Other works propose to try different values finding some approximation which minimizes the error with a practical benchmark (Alkasi et al., 2013). In this work, we propose a mathematical approximation to identify γ based on the Least Squares method (LSM), whose Equation is defined by:

$$\theta = [M^{\mathrm{T}}.M]^{-1}.M^{\mathrm{T}}.Y \tag{5}$$

Where θ is the vector of parameters to estimate, Y is a vector with experimental measures from where we search the parameters, and M is the model matrix. For obtaining values for the identification, we run a simulation where we take (with `HybridBuildingsPropagationLossModel`) RSSI values at reference distances between two nodes. For Equation (5), we define the following: The parameters to estimate are $\theta = [p_0, \gamma]$, and we make $M = \left[1, -10.\log(d_{ij})\right]$, where $d_i j$ is now the vector of the real distances we consider in the experiment (so M is a $m \times 2$ matrix, where m is the number of measures taken), and Y is the vector with the resulting RSSI values. By solving Equation (5), we obtain useful values for p_0 and γ, so the distances can be properly estimated (at least, as good as possible). Finally, the position calculation is performed by using the following methods: (1) `GenerateCalcPos()` (already mentioned in Table 1), which calls to (2) the `Multilateration()` method, which implements the Multilateration algorithm described in Section 2.3, and (3) the `FactCholesky()` method (which is self-explained).

3.3.3 Position Estimation with Fingerprinting (`FingerprintingPositioning`)

For the off-line stage, a set of reference points is defined. At each reference point, a node broadcasts a set of packets. The positioning technique explained in Section 2.4.2 says that the samples in the off-line stage must be taken from packets transmitted with the sender taking different orientations. Actually, the propagation models of ns-3 do not consider an actual orientation of the node (only its coordinate), but to simulate that, we repeat four times the sampling period, and mark the samples as taken by looking "north", "sourth", "east", "west", as indicated by Bensky (2008). The method `GenerateTraffic_offline`, indicates the broadcasting of the samples from the difference reference points. The Beacon nodes receiving those packets, send the packets to the Collector, which normalize and store them in a data base. We adopt Sqlite3 as database manager. The table `Tmeasure` in the database stores the normalized RSSI. The fields of the table are the following: `exp_id`, which stores the experiment identification, `bal_id`, which indicates the identification of the beacon from where the RSSI is obtained, `pRef_id`, which is the identification of the reference point, `med_orientation`, which indicates the orientation of the broadcasting node, `coordX` and `coordY`, which stores the x and y coordinates, respectively, of the reference point, and `med_prom`, which stores the average RSSI captured by the respective beacon.

At the on-line stage, the following tasks are performed: (1) The generation of the network traffic initiated by the Unknown node. The traffic is generated by the `GenerateTraffic_online()` method, which calls the `Send()` method and schedules the new sending events. (2) The receiving beacons calculate the average RSSI. This is done by the method `CalcProm()`. (3) The Collector calculates the vector of power measures and the normalization (according with the indicated in Section 2.4.1. The method `NorVecPot()` performs this last action. (4) The Collector calculates the Euclidean Distances between the normalized stored power values and the normalized values obtained in the on-line stage. The method in charge of doing that is called `CalcDistEuclidean()`. Finally, the calculation of the k nearest distance and of the coordinate of the Unknown node is performed by the `OrderDist()` and `CalcPosition()` methods, which calculate the position according to the indications in Section 2.4.1.

4 Experiments

Various experiments where performed for validating the implementation, and to observe the resulting precision of the selected methods. Figure 2 shows an example of a positioning estimation case, highlighting the real path followed by the Unknown node, and the estimated path.

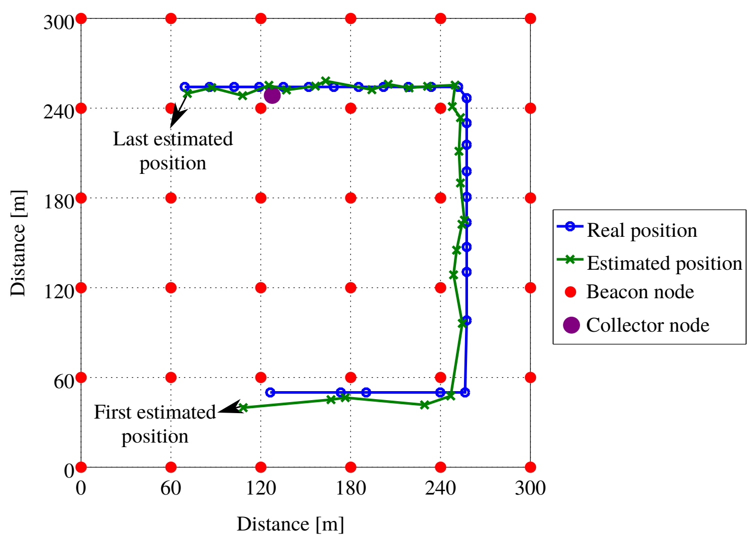

Figure 2. Example of estimated path.

We simulated both an indoor an an outdoor case. The indoor case is a building of 75×15 meters, while for the outdoor case, we simulated an (300×300) meters area. Different scenarios varying the number of beacons (n) and the value of the shadowing sigma (σ) were generated. Actually, the resulting errors are irregular and vary significantly. For generating acceptable results, we filtered the used RSSI values defining a minimum threshold (min RSSI) so that every RSSI inferior to that value is discarded. Each different scenario was simulated 20 times. The results shown are in terms of the average absolute error and the standard deviation of the average results. Table 2 summarizes the resulting average error for indoor and outdoor scenarios.

Table 2
Summarized simulation results for the indoor and outdoor scenarios.

	Parameters			Results Multilateration			Results Fingerprinting		
	Beacons Number (n)	**Shadowing (σ)**	**Min RSSI**	**Error Average**	**Error Std. Deviation**	**Valid Simulations**	**Error Average**	**Error Std. Deviation**	**Valid Simulations**
Indoor Scenario	12	1.0	-70	18.284	1.038	20/20	14.399	1.188	20/20
	12	4.0	-70	19.214	1.651	20/20	17.222	3.952	19/20
	12	8.0	-70	20.707	3.146	20/20	20.893	3.514	19/20
	33	1.0	-70	16.419	1.389	20/20	17.472	1.781	20/20
	33	4.0	-70	16.324	1.451	20/20	22.544	3.192	20/20
	33	8.0	-70	17.320	1.920	20/20	30.307	2.081	20/20
	64	1.0	-60	13.596	1.106	20/20	26.662	1.236	20/20
	64	4.0	-60	16.976	1.557	20/20	21.636	3.328	20/20
	64	8.0	-60	17.688	1.604	20/20	35.570	4.338	20/20
	Parameters			Results Multilateration			Results Fingerprinting		
	Beacons Number (n)	**Shadowing (σ)**	**Min RSSI**	**Error Average**	**Error Std. Deviation**	**Valid Simulations**	**Error Average**	**Error Std. Deviation**	**Valid Simulations**
Outdoor Scenario	9	1.0	-72	74.196	2.917	20/20	143.382	13.196	20/20
	9	3.5	-72	95.940	5.315	20/20	130.706	11.731	20/20
	9	7.0	-72	111.250	5.361	20/20	129.358	11.402	20/20
	25	1.0	-65	7.718	1.226	20/20	135.334	13.485	20/20
	25	3.5	-65	9.961	3.115	20/20	166.935	9.532	20/20
	25	7.0	-65	66.784	12.988	20/20	133.152	9.286	20/20
	36	1.0	-63	11.776	6.139	17/20	196.073	6.346	20/20
	36	3.5	-63	14.442	18.310	20/20	160.666	9.061	20/20
	36	7.0	-63	49.616	16.752	19/20	135.181	9.643	20/20

5 Conclusion

This paper reports the use of ns-3 for simulating RSSI-based local positioning systems. We summarize an explanation on different resources included in ns-3 for simulating positioning scenarios, such as mobility models and propagation models. We report the implementation of different classes for implementing two kinds of positioning algorithms: Multilateration and Fingerprinting. Additionally, we propose a way to identify parameters of the Log Distance equation, based in the Least Squares method. The obtained simulation results evidence the problems of the selected technique for providing precise location information. Those issues are known, and the obtained results are coherent with results reported by other publications, obtained with real experimentation. The developments reported are potentially useful for the networking community, allowing testing scenarios and new proposals validation.

Acknowledgments

This work was financially supported by the University of the Bío-Bío under grants DIUBB 184110 3/R and FDD2011-02.

References

Alkasi, U., Shayokh, M. A., Partal, H. P., Nov 2013. An experimental comparison study on indoor localization: Rf fingerprinting and multilateration methods. In: 2013 International Conference on Electronics, Computer and Computation (ICECCO). pp. 255–259.

Bais, A., Sablatnig, R., Gu, J., June 2006. Single landmark based self-localization of mobile robots. In: The 3rd Canadian Conference on Computer and Robot Vision (CRV'06). pp. 67–67.

Bensky, A., 2008. Wireless positioning technologies and applications. Artech House.

Bischoff, U., Strohbach, M., Hazas, M., Kortuem, G., 2006. Constraint-based distance estimation in ad-hoc wireless sensor networks. In: Römer, K., Karl, H., Mattern, F. (Eds.), Wireless Sensor Networks. Springer Berlin Heidelberg, Berlin, Heidelberg, pp. 54–68.

Dedes, G., Dempster, A. G., Sep. 2005. Indoor gps positioning - challenges and opportunities. In: VTC-2005-Fall. 2005 IEEE 62nd Vehicular Technology Conference, 2005. Vol. 1. pp. 412–415.

Dobre, D., Bajic, E., April 2008. Active product modeling for chemical security management based on smart object concept. In: 7ème Conférence Internationale de Modélisation, Optimisation et Simulation des Systèmes, MOSIM 08. Paris, France.

Duran-Faundez, C., Aguilera-Carrasco, C., Norambuena, A., march 2010. Experimenting with rssi for the perception of moving units in intelligent flexible manufacturing systems. In: Industrial Technology (ICIT), 2010 IEEE International Conference on. pp. 1400 –1405.

Escudero, F., Margalef, J., Luengo, S., Alsina, M., Ribes, J. M., Pérez, J., 2007. Determining position inside non-industrial buildings using ultrasound transducers. Sensors 7 (11), 2579–2598.

Friis, H. T., May 1946. A note on a simple transmission formula. Proceedings of the IRE 34 (5), 254–256.

Grover, J., , Sharma, M., July 2014. Location based protocols in wireless sensor network — a review. In: Fifth International Conference on Computing, Communications and Networking Technologies (ICCCNT). pp. 1–5.

Gubbi, J., Buyya, R., Marusic, S., Palaniswami, M., 2013. Internet of things (iot): A vision, architectural elements, and future directions. Future Generation Computer Systems 29 (7), 1645 – 1660, including Special sections: Cyber-enabled Distributed Computing for Ubiquitous Cloud and Network Services & Cloud Computing and Scientific Applications — Big Data, Scalable Analytics, and Beyond. URL http://www.sciencedirect.com/science/article/pii/S0167739X13000241

International Telecommunication Union (ITU), Jun. 2017a. P.1238 : Propagation data and prediction methods for the planning of indoor radiocommunication systems and radio local area networks in the frequency range 300 mhz to 100 ghz. https://www.itu.int/rec/R-REC-P.1238-9-201706-I/en, recommendation P.1238-9 (06/2017).

International Telecommunication Union (ITU), Jun. 2017b. P.1411 : Propagation data and prediction methods for the planning of short-range outdoor radiocommunication systems and radio local area networks in the frequency range 300 mhz to 100 ghz. https://www.itu.int/rec/R-REC-P.1411-9-201706-I/en, recommendation P.1411-9 (06/2017).

Jiménez Ruiz, A. R., Seco Granja, F., Aug 2017. Comparing ubisense, bespoon, and decawave uwb location systems: Indoor performance analysis. IEEE Transactions on Instrumentation and Measurement 66 (8), 2106–2117.

Karl, H., Willig, A., 2005. Protocols and Architectures for Wireless Sensor Networks. John Wiley & Sons.

Laaraiedh, M., Avrillon, S., Uguen, B., April 2009. Enhancing positioning accuracy through direct position estimators based on hybrid rss data fusion. In: VTC Spring 2009 - IEEE 69th Vehicular Technology Conference. pp. 1–5.

Lee, J. H., Buehrer, R. M., 2011. Handbook of Position Location: Theory, Practice, and Advances. Wiley-IEEE Press, Ch. Fundamentals of received signal strength-based position location, pp. 359–394.

Lymberopoulos, D., Liu, J., Yang, X., Choudhury, R. R., Handziski, V., Sen, S., 2015. A realistic evaluation and comparison of indoor location technologies: Experiences and lessons learned. In: Proceedings of the 14th International Conference on Information Processing in Sensor Networks. IPSN '15. ACM, New York, NY, USA, pp. 178–189.
URL http://doi.acm.org/10.1145/2737095.2737726

Molina-Martínez, C., Galdames, P., Duran-Faundez, C., 2018. A distance bounding protocol for location-cloaked applications. Sensors 18 (5).
URL http://www.mdpi.com/1424-8220/18/5/1337

Motley, A. J., Keenan, J. M. P., June 1988. Personal communication radio coverage in buildings at 900 mhz and 1700 mhz. Electronics Letters 24 (12), 763–764.

Nandury, S. V., Begum, B. A., Aug 2015. Smart wsn-based ubiquitous architecture for smart cities. In: 2015 International Conference on Advances in Computing, Communications and Informatics (ICACCI). pp. 2366–2373.

Ojha, T., Misra, S., Feb 2013. Mobil: A 3-dimensional localization scheme for mobile underwater sensor networks. In: 2013 National Conference on Communications (NCC). pp. 1–5.

Okumura, Y., Ohmori, E., Kawano, T., Fukuda, K., 1968. Field strength and its variability in VHF and UHF land-mobile radio service. Review of the Electrical Communication Lab. 16 (9-10), 825–873.

Perkins, C., Belding-Royer, E., Das, S., Jul. 2003. Ad hoc on-demand distance vector (aodv) routing. RFC 3561, doi:10.17487/RFC3561.

Preuveneers, D., Ilie-Zudor, E., April 2017. The intelligent industry of the future: A survey on emerging trends, research challenges and opportunities in industry 4.0. Journal of Ambient Intelligence and Smart Environments 9 (3), 287–298.

Rappaport, T. S., 2002. Wireless Communications: Principles and Practice, 2nd Edition. Prentice Hall.

Riley, G. F., Henderson, T. R., 2010. The ns-3 Network Simulator. Springer Berlin Heidelberg, Berlin, Heidelberg, pp. 15–34.
URL https://doi.org/10.1007/978-3-642-12331-3_2

Silva-Faundez, J. A., Duran-Faundez, C., Melin, P., Aguilera, C., 2015. An anticipatory control for a flexible manufacturing system based on the perception of mobile units using wsns. International Journal of Computers Communications & Control 10 (5), 702–717.
URL http://univagora.ro/jour/index.php/ijccc/article/view/634

Sobehy, A., Renault, E., Muhlethaler, P., 2019. Position certainty propagation: A localization service for ad-hoc networks. Computers 8 (1).
URL http://www.mdpi.com/2073-431X/8/1/6

Taylor, S. E., McDonald, T. P., Veal, M. W., Grift, T. E., 2001. Using gps to evaluate productivity and performance of forest machine systems. In: Proceedings of the First International Symposium on Precision Forestry. University of Washington, Seattle, WA., pp. 151–155.

ur Rehman Khan, A., Bilal, S. M., Othman, M., 2013. A performance comparison of network simulators for wireless networks.
URL http://arxiv.org/abs/1307.4129

Verdone, R., Dardari, D., Mazzini, G., Conti, A., January 2008. Wireless Sensor and Actuator Networks: Technologies, Analysis and Design, 1st Edition. Academic Press, iSBN: 978-0-12-372539-4.