Robot localization using wireless networks

Oscar Serrano Serrano
e-mail: oserrano@gfi-info.com
Departamento de Informática, Estadística y Telemática
Universidad Rey Juan Carlos (Móstoles, Spain)

15th September 2003

Abstract

This paper describes how IEEE 802.11b wireless networks can be used to determine the position of a robot inside a building. For this purpose we carried on some experiments which have shown that localization using only a wireless network as the only sensorial information (without a motion model) is not possible without a preconstructed sensorial map. But with the use of a motion model (easy to implement if we are using robots), localization can be achieved using theoretical models and MArkov localization techniques and we don't need to bound the localization to the environment. We also believe that even it has been said that orientation affects signal strength [10] the changes are to slight and it's not possible to use this information to obtain a good estimation of the orientation.

1 Introduction

Most successful mobile robot systems to date use localization, as knowledge of the robot's position is essential for a broad range of mobile robot tasks. Robot localization has been recognized as one of the most fundamental problems in mobile robotics [1, 2]. The aim of localization is to estimate the position of a robot in its environment, given a map of the environment and local sensorial data.

Localization is a fundamental task for an autonomous mobile robot, according to Cox: "Using sensory information to locate the robot in its environment is the most fundamental problem to providing a mobile robot with autonomous capabilities" [4, 3, 5, 6]. Thus we will have to locate the robot in a environment for which it have a map.

There are two main approaches to robot localization and both of them are important for the design of autonomous vehicles:

Global localization: To locate the robot current position inside the world. This part is important for robots that are not aware of where they are inside the global map. This problem is generally regarded as the most difficult one as we don't have any prior information about the robot position.

Local localization: To keep on estimating the position of the robot while it's moving. This is important because once the robot knows where it is, we only need to keep track of the possible errors that will be induced by the actuators or the sensors and that will lead the believe that it's in a different positions from the right one.

To solve the localization problem the robot sensors will play a critical role as they will provide the robot with all the needed information. Unfortunately sensors are noisy and provide erroneous measures. Figure 2 shows how the odometry errors accumulate and distorsionate the beliefs of the robot.

. Encoders will be the most natural way to track the position of a robot, but they have been shown to be very inaccurate. Figure 2 shows in the left a map and overlapped to it the path that a robot believes it has followed according to the information of its odometry sensors. On the right side it can be seen how a map builst by a robot looks different from the reality because of sensorial errors. The accumulative odometry errors causes the robot perception to strongly differ from the reality and due to this most of the localization techniques are based in sensorial fusion(probabilistic localization, Markov localization). Another way to estimate a position is using external sensors or marks to triangulate. Triangulation works very good as long as there are reference points and has been used for centuries, being specially important in sailing where the celestial navigation has allowed our ships to sail the oceans without getting lost in an environment where very few natural (stars) or artificial landmarks are available. Figure 1 shows how latitude can be estimated using only the polestar and the horizon as references. To triangulate a position in 2D we need 3 points/lengths or 2 angles and 1 length. There are other localization techniques, some of them require environmental engineering such as the placement of active beacons, passive marks, etc.

The most recent localization techniques at the moment, are using probabilistic localization in which the

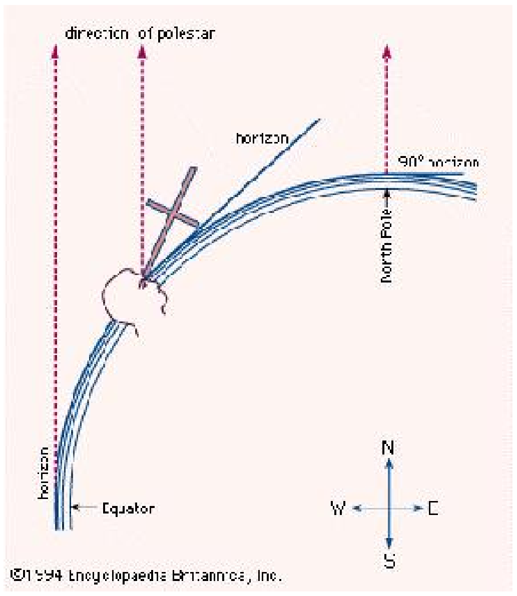


Figure 1: Encyclopedia Britannica: Celestial navigation (latitude estimation)

position is estimated with information from indirect observations. Instead of having sensors or marks that allow us to decide our position directly we will use the history of the data read from diferent kind of sensors to estimate a position that we will be updating with the new incoming readings. This solution suits good to this kind of probems because it allow us to represent ambiguous situations and solve them later when we have received more information. For our work we will use this approach using the signal strengh from a wireless network and information about the robot encoders.

Our interest in the use of wireless networks to locate robots is based in the fact that it will be a cheap, non intrusive method because the infrastructures are already deployed, may cover big areas and can be used easily by the robot only with the aid of a simple network card and it's not needed to add any special mark to the environment. In our work we will try to solve the ilocalization problem using only information from the wireless network and odometrys. we have tested it for different localization method from triangulation to statistical localization but in future works it will be interesting to improve it using also other kinds of sensors.

This approach as been already used by Jason Small [21] who solved the problem using a map with resolution of 1'5 meters and restricting it to a corridor with line of sight which is a quite simple approach, Andrew M. Ladd [10] needs quite complex sensorial maps to obtain a resolution of 1'5 meters, and Sajid M. Siddiqi [24] that is using also a sensorial map with a resolution of 2m and Monte-Carlo localization methods. The main difference of our work with the previous ones is

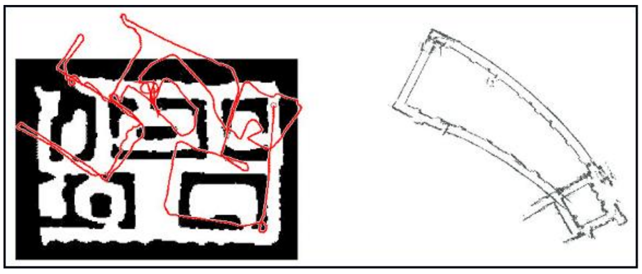


Figure 2: Odometry Errors: heading and distance measurements accumulate errors with time (from Robotic mapping: A survey [22])

that apart from getting a very good estimation of the localization using a map with 2 meter resolution and Markov localization techniques (which is very similar to some of the previous works) we are introducing the possibility of using a theoretical model of indoor radio propagation instead of building a wireless energy map. With this model we are getting worse estimations than with the map but it results on an easier implementation. The resolution we are using to retrieve the data for the simulator or to build the map is 2 meters (because we realized there are no noticeable differences in measures for smaller distances), we make all the calculus to achieve the localization using a 10 cm grid which allow us to give a more precise localization.

The rest of the paper will be organized as follows: Next section will be dedicated to explain the basics of radio signal propagation, the third section will describe the simulator and the tools that we are using, and will introduce the initial work needed (how the result will be visualized and how the initial data will be gathered); In section fourth we will present the experiments carried out and the last section analyzes the results obtained summarizing the conclusions we have reached.

2 RF Signal propagation in Wireless Ethernet

In 1999 the IEEE completed and approved the standard known as 802.11b, this standard uses radio frequencies in the 2.4 GHz band while the 802.11a, which came later on uses the 5 GHz band [8, 9, 13]. This way, computer networks can achieve connectivity with an useable amount of bandwidth (up to 11Mb per Access Point) without being connected to a wall socket. An infrastructure WLAN consists of several clients talking to a central device called an Access Point (AP), which is usually connected to a wired network which offers higher bandwidth. The huge growth of this kind of networks during the last years have inspired us to use them as a extra information to help in robot location.

The radio signals that we will use will be affected by different propagation problems. In outdoor environments (with a clear line sight) the propagation of the signals is perfectly modelled by the free space lost model that can be seen in Figure 3 . In indoor environments [11, 12] the modelling problem became worse since there are obstructions in the form of walls, roofs, furniture and people. These obstructions can affect the propagation of the radio waves in different ways.[7, 14]

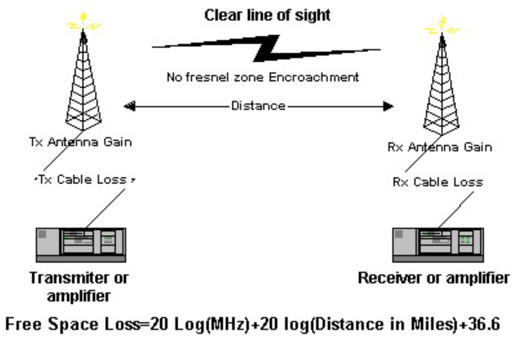


Figure 3: Model for radio signal propagation in outdoor environments

Reflection: Occurs when an electromagnetic wave impinges on an object which is larger than the wave's wavelength. This can cause the propagating wave to loose power while passing through an obstacle such as a wall, but can also cause the reflected wave to propagate along completely different paths.

Diffraction: Occurs when an electromagnetic wave is obstructed by a surface with irregular edges. This can cause a wave to travel around corners and other edges. This effect is very useful in a building environment as it allows the signal to travel a path other than LoS (line of sight).

Scattering: Occurs when an electromagnetic wave is obstructed by an object with dimensions smaller than the wavelength of the wave. Scattering will cause signal dissipation and also an effect similar to that of reflection in which different propagation paths can be followed by the scattered signal. These effects lead to multiple propagation paths for one signal and so lead to the multipath effect.

This obstructions lead to the Multipath problem, which occurs when an RF signal takes different paths when propagating from a source (e.g., a radio NIC) to a destination node (e.g., access point). While the signal is in route, walls, chairs, desks, and other items get in the way and cause the signal to bounce in different directions. A portion of the signal may go directly to the destination, and another part may bounce from a chair to the ceiling, and then to the destination. As a result, some of the signal energy will be delayed.

Multipath delay causes the information symbols represented in an 802.11 signal to overlap. The receiver will make mistakes when demodulating the signal's information. If the delays are great enough the receiver won't be able to distinguish the symbols and interpret the corresponding bits correctly. When multipath strikes in this way, the receiving station will detect the errors through 802.11's error checking process and will not send an 802.11 acknowledgement to the source. The source will then eventually retransmit the signal after regaining access to the medium. Because of retransmissions, users will encounter lower throughput when multipath is significant. The reduction in throughput depends on the environment. Multipath is a typical problem in closed environments and will be the mayor source of error during our experiments.

Another problem that we will have with radio signals is the noise or interferences which are, in general, caused either by radio devices operating in the same bands or by thermal noise, or both. For a single AP, thermal noise is the only source of interference. With multiple ones, however, there is interference from adjacent channels and co-channels. The overall impact of this interference depends upon the number of available frequency channels and cell deployment. Careful cell deployment and management of the number of available channels could mitigate its effects. In our experiments we will have our 3 AP plus 1 unlocalized AP that is not owned by us (so we can not shut it up). This means that we will have also problems with interferences between the Access Points. All this this problems will affect the energy level of the signal that we will get from our sensor and affect the localization algorithm. Temperature, persons, doors, chairs and lots of other factors will introduce changes in the signal levels and this is because this kind of signals are difficult to model in indoor environments.

3 Experimental Setup

In order to realize our experiments we needed to build some modules that can be seen in figure 4. The function of the system is the following: We take the real robot position from ARIA, with this position we compute a odometry position adding an error and we extract the wireless lecture from the table (to which we also add some error). This two datas are the inputs to our algorithm that is giving its output to a graphical module.

The robot used in our experiments is a Pioneer 2-DXE (see figure 5), it is a 44cm x 38cm x 22cm aluminum body robot with 16.5cm diameter drive wheels. The two motors use 19.5:1 gear ratios and contain 500-tick encoders. On flat floor, the P2-DXE can move at speeds of 1.6 m/s. At slower speeds it can carry pay-

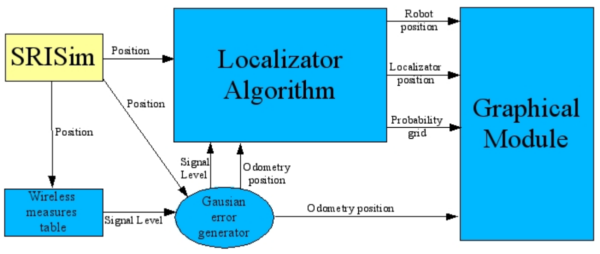


Figure 4: Localizator architecture

loads up to 23 kg. In addition to motor encoders, our robot includes 16 ultrasonic transducer (range-finding sonar) sensors arranged to provide 360-degree coverage in ranges from 15cm to approximately 3m and a set of bumpers around it.

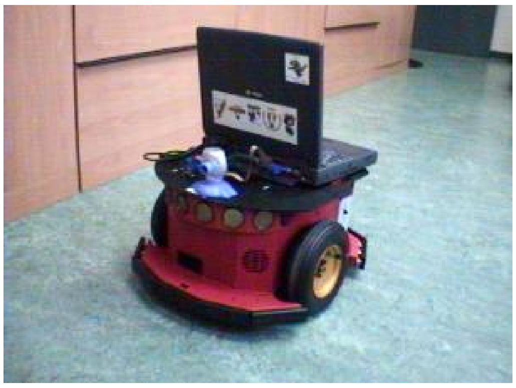


Figure 5: Our pioneer robot with its laptop and camera mounted on (the wireless card is in the laptop)

The standard P2-DX include a Siemens C166-based microcontroller for computing operations, but in our case we are also using a laptop that is carried over the robot and connected through a serial port to the siemens micro-controller and a color camera connected to the laptop using a USB port.

A small proprietary Operating System (P2OS) transfers sonar readings, motor encoder information and other I/O via packets to the PC client and returns control commands. The mobile servers, embodied in the Pioneer 2 Operating System software, manage the low-level tasks of robot control and operation, including motion, heading and odometry, as well as acquiring sensor information (sonar and compass, for example) and driving accessory components. The robot servers do not, however, perform robotic tasks.

For programming purposes we will use the Aria library which is a free software library written in C++. Aria is a client-side software library written in C++ for easy, high-performance access to and management of the robot server, as well as to the many accessory robot sensors and effectors. Aria can be run multi- or

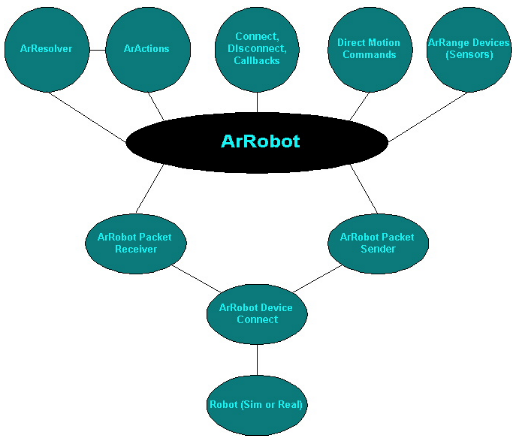
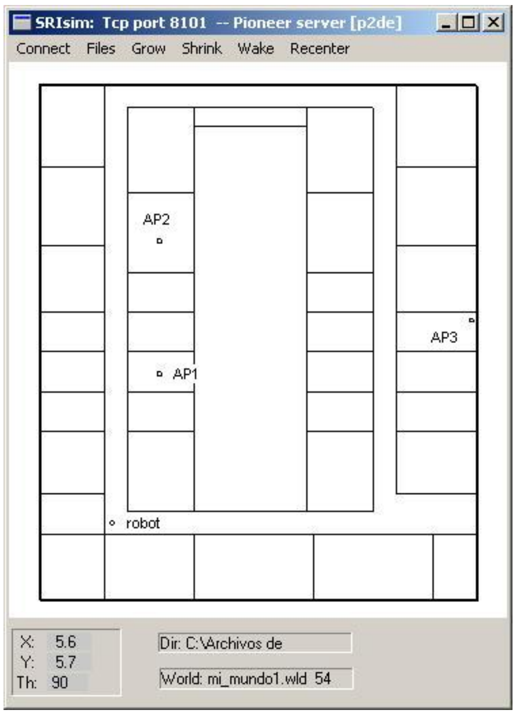


Figure 6: Aria class diagram

single-threaded, using its own wrapper around Linux pthreads and WIN32 threads [15]. In figure 6 can be seen a simplified Aria classes diagram where the most important class is ArRobot which represents our robot, the class ArRangeDevices allows to add sensors like sonars or lasers to the robot, the rest of the classes are intended to facilitate the communication or to add predefined actions or behaviors to the robot.



X: 5.6
Y: 5.7
Th: 90
Dir: C:\Archivos de
World: mi_mundo1.wld 54

Figure 7: SRISim with the map of the dept. building where experiments were carried on

Aria is shipped with a Simulator(SRISim) so that Aria programs can connect either to the real robot or to SRISim working in both systems. SRISim allows to load worlds in a specific format and is useful for the debugging of the programs before testing them on the real robots.

During the development of the work we will use Aria for all the behaviors of the robot and we will use SRISim during the development of the programs.

3.1 Graphical representation

SRISim allows us to have a simulation environment and a graphical tool to see how our robot is behaving. For the visualization of the robot beliefs about its own position we have developed a graphical module showed in figure 15. This module allows the visualization in an X-Windows screen of the world along with the real position (black circle), the encoders position (green circle), the position our algorithm is estimating (red circle) in each moment and the probability cloud using a scale of colors.

For the development of this module we are reading the SRISim world definition archives that contains a list of points that defines the lines that are forming the world. This points are then scaled to the desired size and drew using a digital differential analyzer algorithm(DDA)[23] for the computation of the points that are forming the line. The basis of the DDA method is to take unit steps along one coordinate and compute the corresponding values along the other coordinate. The unit steps are always along the coordinate of greatest change. This algorithm have the drawback of using floating points calculations that are slower than the integer ones but for our simple visualization problem it's working fine.

For the drawing of the circle that is giving shape to the robot we are using a algorithm based in Bresenham's Algorithm [23]. This algorithm is based in reducing the amount of computation required by capitalizing on the symmetry of a circle, we need only to compute the (x, y) values in one part of the circle. For a given point (x, y) on the circle there are seven other points along the circle whose coordinates can be easily found. Simply negating and/or interchanging x and y produces the seven locations, (-x, y), (-x, -y), (x, -y), (y, x), (-y, x), (-y, -x) and (y, -x). This is shown in Figure 8.

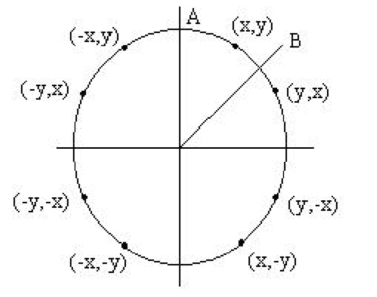


Figure 8: Computing the points of a circle

3.2 Measures gathering

As we will be using widely the simulator in the preliminary versions of our system. We are in the need of collecting measures from the wireless cards all around our world, so that we can use them later on as the real measures while simulating (taking the measures from the wireless cards won't work as the robot is not moving and we will always get the same measures) and also for building a map of the environment.

Modifying the wireless card driver for the supported Linux kernel [25] we took measurements of the signal level, noise level, signal quality and Access Point to which we were connected in all the area that the robot would eventually move. The measurements were took using a laptop equipped with a wireless network card that was moved through the whole area. The measures were taken 4 times (one in each direction) for each spot placing the laptop every 2 meters.

The reason for taking the measures in 4 different directions in each point is based on the multipath effect that we stated before and in the possibility of estimating the orientation of the robot using this information. The decision for doing it each 2 meter was that due to multipath effect we couldn't find significative differences in the measures in shorter distances.

The distribution of the signal levels measured for each AP in each position can be seen in Figures 9, 10, 11 and 12. This figures shows the non-linear, fluctuating nature of indoor radio signal propagation and justifies the need of a motion model to have enough information to achieve a correct localization.In this figures it can be observed that the signal levels aren't very discriminative in short distances, we also noted that are not discriminative at all in relation with orientation. There are areas of up to 6 meters in which the signal is moving around the same level, also the level is usually decreasing with distance but sometimes an increase in the signal is measured when the distance is getting bigger.

For the use of the simulator we will need a continuous source of signal levels measures that we will take from this data. In order to use it as real input measures to our robot we will add a Gaussian error to it (this is not completely realistic as the error in RF signal propagation is not Gaussian but will be enough for simulation purposes). There are many ways to do this [20] and in our case this error will be computed using a transformation function. The most important of these transformation functions is known as the Box-Muller transformation [19]. It allows us to transform uniformly distributed random variables, to a new set of random variables with a Gaussian distribution.

The most basic form of the transformation looks like:

$$y_1 = \sqrt{-2\ln(x_1)}\cos(2\pi x_2)$$
$$y_2 = \sqrt{-2\ln(x_1)}\sin(2\pi x_2)$$

We start with two independent random numbers,

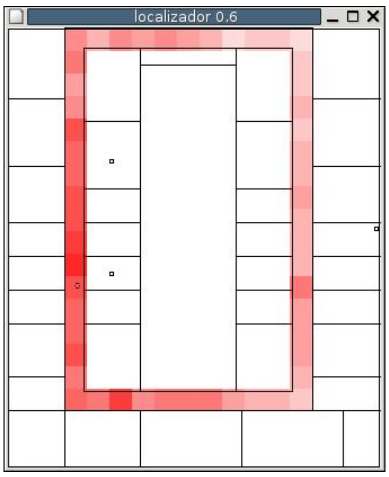


Figure 9: Measures for AP1

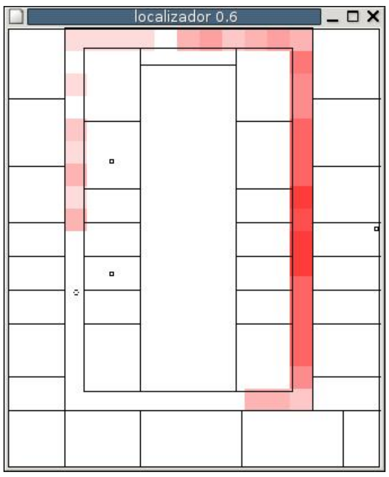


Figure 11: Measures for AP3

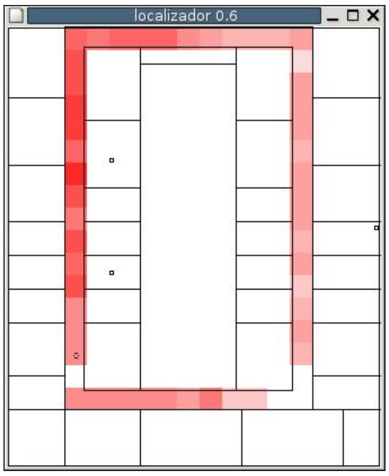


Figure 10: Measures for AP2

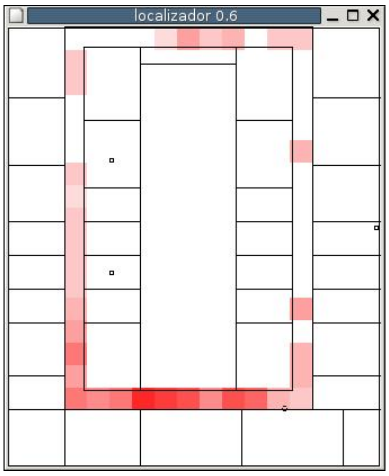


Figure 12: Measures for AP4 (this AP is not controlled by us)

x_1 and x_2, which come from a uniform distribution (in the range from 0 to 1). Then we apply the above transformations to get two new independent random numbers which have a Gaussian distribution with zero mean and a standard deviation of one. This particular form of the transformation has two problems with it, it is slow because of many calls to the math library and it can have numerical stability problems when x_1 is very close to zero. So we are using a polar-form of this transformation that does the equivalent of the sine and cosine geometrically and is giving faster results.

4 The Experiments

As we have explained previously, the goal of this work was to test whether our robot can localizate itself, we are going to limit the problem to the localization of a P2DXE robot in the corridors of the Computer Science department. This department is made up by two 34 meters hallways in the east and west and two of 22 meters in the north and south forming a rectangle sourrounded by several rooms of different shapes and sizes as shown in figure 7. In this area there have been deployed time ago before our experiments the wireless network with three Access Points that can be seen in Figure 7. We can also receive measures from another access point which is not under control.

We let the robot move free by using a wander behaviour and want to be able of locate it at every moment using only the data provided by the wireless network and the odometry.

In the experiments we will try to solve the problem using different methods in order to compare them. In first method we will use a map of the environment. Our second experiment uses a algorithm based on triangulating the position using the information about how much the signal strength decreases with distance. The third and the forth methods use theoretical models of indoor radio propagation.

For all the experiments we will use similar heuristics to compute a distance function between the sensed values and the theoretical value we will expect in each point. With the exception of the first approach in which we will use the additional information that we get when we build the map and that we don't have in the others experiments, like the access point to which we are connected or the number of access points we are getting information from in each point. Also we will use Markov Localization techniques for all of them, we will use a sensorial model and a motion model, and for both of them we will Bayes to add the new evidences to our previous beliefs [16, 17, 18].

The probabilistic data as well as the captured screens that can be found in each experiments where collected from a set of 4 random runs of around 500 seconds each.

4.1 Approach 1

We try to solve the problem using a map built using the measures that the robot could find in the area in which it should move. In the experiments we have discretized the world into 2 square meter cells. As stated before we choose this distance because we couldn't find representative differences in the signal level for smaller distances.

In order to estimate the position of the robot we are just computing the difference between the sensed values (number of AP sensed, AP to which we are connected and quality, error and signal level of the signals for each AP) and the ones stored previously in the map. We use an heuristic function that returns a normalized distance value (0-1) for each position.

The output of our algorithm gives a position with a precision of 10 centimeters even thought the heuristic function will give us the same value for each 2 meters region. The odometry will give as the rest of the information needed to find a more accurate localization as well as other information extracted from the map like walls or the limits of our world through the use of a motion model .

We compute the heuristic value of being in the point (x, y) as the number of times the map measure and the current measure are the same. This value is calculated for each property from: number of AP sensed, AP to which we are connected and quality, error and signal level of the signals for each AP, divided by the total number of checks we are making. For computing the total number of checks the failing checks, that are most of them, are weighted so that they are proportional to the distance to the sensed measure. This is done because even there is a difference between the stored map measure and the sensed masure, for us a big difference should return a smaller value than when there are great differences between them.

This heuristic distance function will give us $d(t)$. Then we use Markov localization [16, 17, 18] and thus we have that:

$Bel(L_t = l)$ is the belief for the position l of being the real position of the robot (L) at time t.

$P(obs(t)/l)$ is the probability to obtain the sensor measures we are observing in time t if the robot were in position l.

$Bel(L_{t-1} = l)$ and $Bel(L_{t-1} = l^{\prime)}$ are the prior distributions.

$P(l/a_t, l')$ is the probability of being in position l if we are in position l' and we have made the action a (in our case a movement) at time t.

α is a normalization factor to make the posterior distribution integrate to 1.

σ is a weight factor to make distances bigger.

In this way we are calculating the sensorial model:

$Bel(L_t = l) = \alpha_t P(obs(t)/l) * Bel(L_{t-1} = l)$

if the most recent sensor data read is a wireless network reading and the motion model:

$Bel(L_t = l) = \int P(l/a_t, l') * Bel(L_{t-1} = l^{\prime)} d'$

if the most recent data read is a odometry lecture.

Where

$P(obs(t)/l) = d(t)\sigma$

and

$P(l/a_t, l^{\prime)} = 1$ if from l' we can move to l with a_t or 0 in any other case.

With this settings we are getting a average error in the localization of 1.08 meters with a standard deviation of 0.70 meters, the error is smaller than 1.5 meters in 82.27% of the times and is smaller than 3 meters in 97.9%.

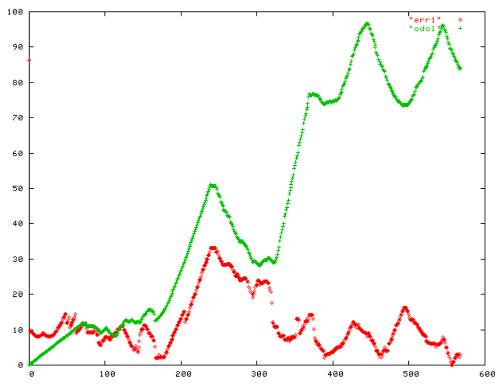


Figure 13: Distribution of errors for approach 1

Figures 13 and 14 show the distribution of the error of our localizator compared to the accumulated odometry error using this method. The X-axis is the time in seconds and the Y-axis is the distance in decimeters. The main difference in both graphics is that the accumulated odometry error is starting at 0 and is not bounded while our algorithm error is starting at higher levels because it needs to update the beliefs and de-

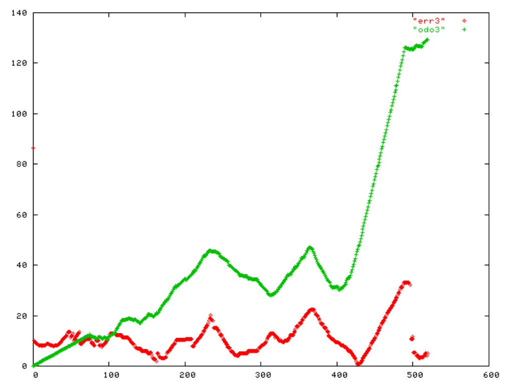


Figure 14: Distribution of errors for approach 1

creasing with time, being bounded around a maximun of 3 meters.

The shape of the error using the wireless network is similar to the shape of the odometry error graph because we are using a motion model that relies on the data provided by odometry sensors.

There can be also observed some discontinuities in the graph of our localizator. This is due to the fact that the error of our algorithm is becoming too large because of invalid odometry readings. And in this cases the information provided by the wireless card is strong enough to change the belief to a position closer than the previous one where the distance function is giving smaller values.

In figures 15 and 16 can be seen our localizator algorithm working. The probability cloud is represented in a scale of reds being the darker reds the position with higher beliefs.

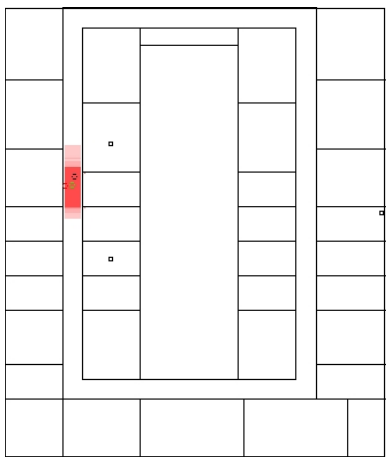


Figure 15: Localizator running using approach 1 after 30 seconds of execution

It can be observed that there is no probability cloud outside the corridor, this is because the goal of the experiment was to locate the robot inside the corridor and thus we don't have any data in the map for this

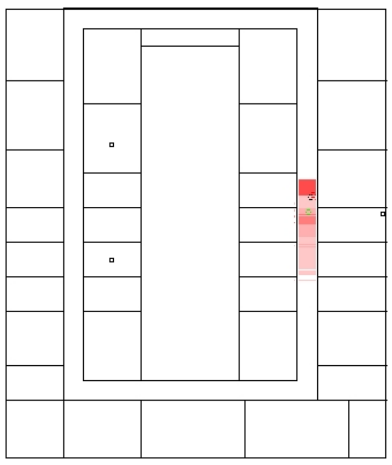


Figure 16: Localizator running using approach 1 after 250 seconds of execution

positions and we are asigning 0 to them. This is making the distance function to compute very high values outside the corridor and to assign very low probabilities for this positions.

4.2 Approach 2

We will try to solve the problem using the signal level as a measurement of the distance to each Access Point. In this case we need an empirical model of how the signal level is affected by the distance, so first of all we have to take measurements for each single access point at different distances. For this measurement we left just one Access Point connected and take one measure each 2 meters. The results can be seen in Figures: 17, 18 and 19.

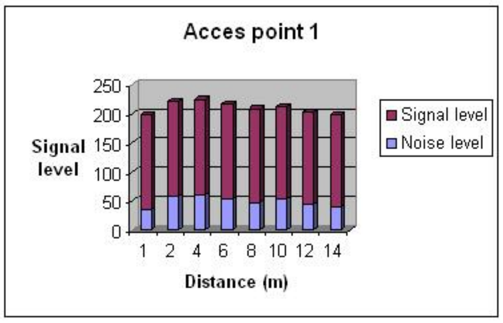


Figure 17: Measures for access point 1

Once we have got measures of how the signal level of each AP is decreasing with the distance we can calculate a distance function just in the same way than in the experiment before.

Using this method we couldn't get useful results. It can be seen how the changes in the signal levels in Figures 17, 18 and 19 aren't significative enough with

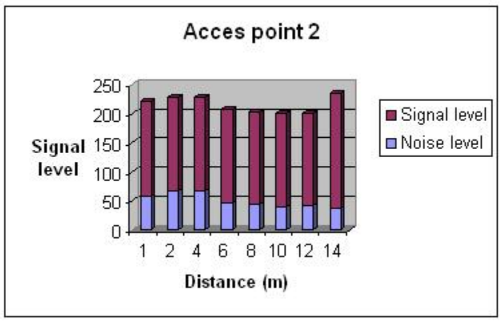


Figure 18: Measures for access point 2

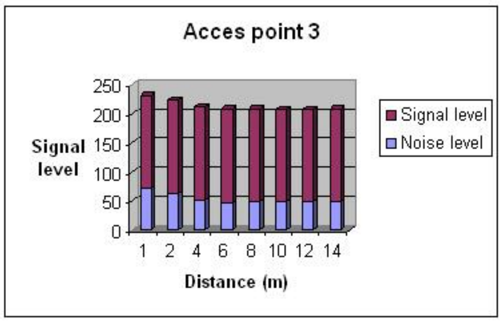


Figure 19: Measures for access point 3

distance. Measures where taken using a clear line of sight and in practice there are walls between the AP's and the robot that makes the signal to differ greatly and makes the localization impossible.

4.3 Approach 3

There have been a lot of research about propagation of radio signals in indoor environments, in this experiment we will try to use one of the theoretical models that have been proposed to get the estimated values of each AP in each point. We have choosen a breakpoint model [11, 12] that is used for indoor propagation analysis.

Our model will compute the signal level for each point as:

$SignalLevel = 240 - (20 + 10 * \log(dist))$ if dist<=5 meters

$SignalLevel = 240 - (34 + 10 * 1.5 * \log(dist/5))$ if dist>5 meters

This is a free space loss model that only takes into account the attenuation produced by the space that the signal is travelling (without concerning about refection, diffraction or scattering) for the first meters but using a higher exponent when the signal is increasing above breakpoint. An example of this model can be seen in Figure 20. In this example the breakpoint is 5 meters and after that distance we are using a different function that is decreasing quickler.

In our experiment as the robot is only moving through the corridor it will rarely stay bellow the breakpoint, that we have set in 5 meters.

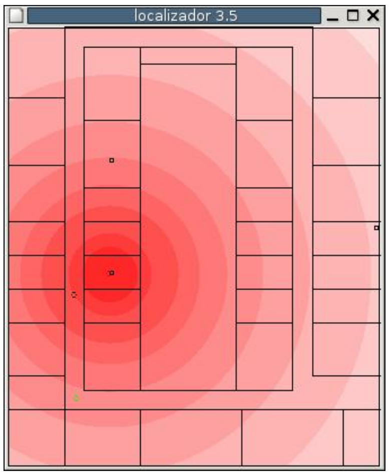


Figure 21: Model for AP1

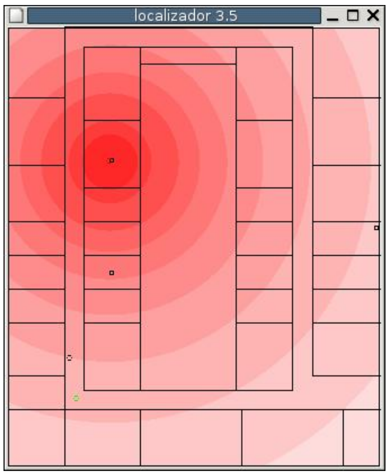


Figure 22: Model for AP2

The use of the model for this experiment allow us to build a map of the theoretical measures we could find in each point with the desired resolution. In our case we will use 10 cm resolution, and thus our distance function will return different measures for each 10 cm region. In figures 21, 22 and 23 it can be seen how we have theoretical signal level measures for every point in the map. This signal levels are different in each 10 cm region. The concentrical circles of the same color that can be observed are due to the the fact that we have chosen to discretize the similar measures in a few colors to make the representation clearer.

The motion model will be the same as in the experiment before, while the formula that we will use to update the sensorial model is given by:

$Bel(L_t = l) = \alpha_t P(obs(t)/l) * Bel(L_{t-1} = l)$

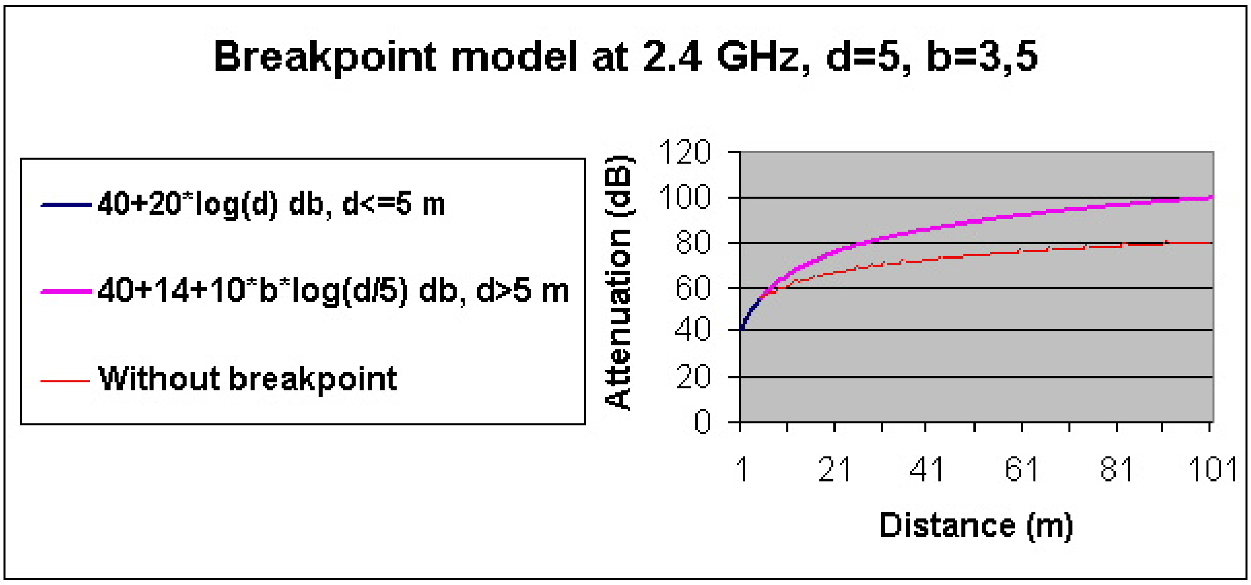


Figure 20: Breakpoint model example

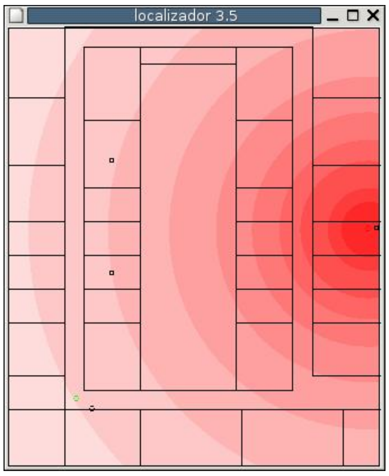


Figure 23: Model for AP3

Where
$P(obs(t)/l) = 1 - d(t)\sigma$
and
$d(t) = \sum_{i=0}^{numap} (\frac{|r^i_{obs} - r^i_l|}{r^i_l})$
r^i_{obs} is the observed signal level for AP i.
r^i_l is the theoretical signal level (using the model) in position l for AP i.

With this configuration we are getting an average error of 2.00 meters with a standard deviation of 1.38 meters, 42% of the times the error is smaller than 1.5 meters and 83% is bellow 3 meters.

In figures 24 and 25 can be seen the distribution of the error of the localizator versus the odometry error in two different runs. After the initial steps in which the robot it still updating its beliefs the error is bounded around 4 meters. There are more discontinuities than in approach 1 because our model is now also giving signal measures to the points outside de corridor, as can be seen in figures 21, 22 and 23. This makes it possible for the localizator to believe that the robot is outside the corridor which forces sudden changes in the beliefs as it can be seen in figure 25. Most of this sudden changes in our localizator are happening when odometry errors are growing too fast which carries the beliefs to positions far away from the right one and in most cases outside of the corridor and is forcing the change. Most of this sudden changes in the beliefs are giving a better approximation to the real position, but sometimes happen that the approximation in particular is worse when the odometry error has moved the beliefs to distant positions and the robot is not passing through reference points, as walls or corners.

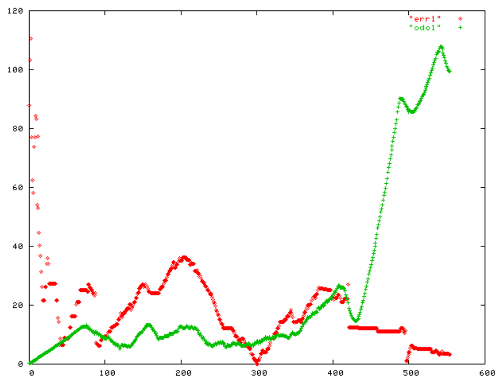


Figure 24: Distribution of errors for approach 3

In figures 26 and 27 can be seen how our localiza-

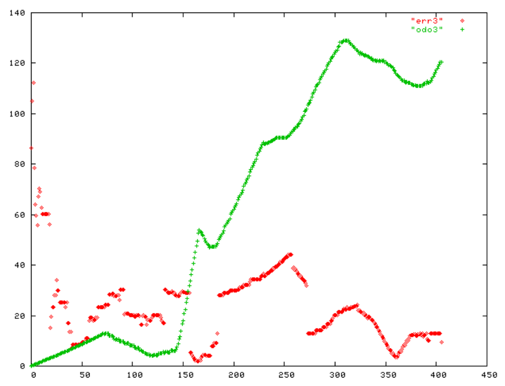


Figure 25: Distribution of errors for approach 3

tor algorithm works. In this case there are probabilities clouds outside the corridor but not in the offices (figure:26) because we are assigning near 0 probabilities to the walls positions and the propagation of evidences that Markov localization uses makes this spaces to have very low probabilities. In figure 27 it can be seen how our algorithm converges to a point while the green circle (accumulated odometry error) is getting totaly lost.

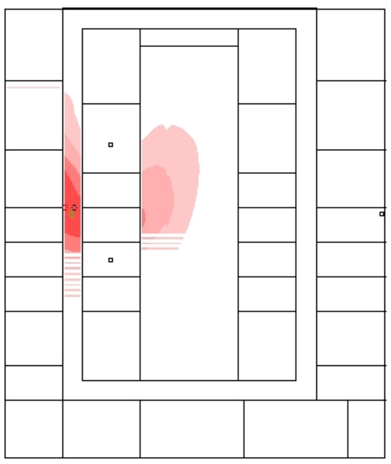


Figure 26: Localizator running using approach 3 after 30 seconds of execution

4.4 Approach 4

The algorithm used in the fourth approach is the same as in the third but we change the sensorial model to follow a exponential distribution to try to reduce the error we had detected in the previous one. The new sensorial model will be:

$d(t) = e^{-(\sum_{i=0}^{numap} (\frac{|r^i_{obs} - r^i_l|}{100})\sigma)^2}$

and

$P(obs(t)/l) = 1 - d(t)$

With this sensorial model we are getting a better

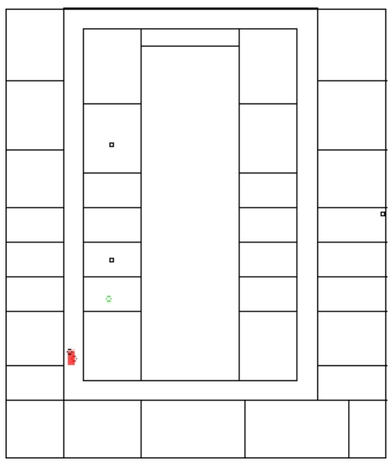


Figure 27: Localizator running using approach 3 algorithm after 300 seconds of execution

localization than with the previous one. The average error is just of 1.53 meters, with a standard deviation of 1.52 meters. 61% of the times the error is under 1.5 meters and 94.1% is smaller than 3 meters.

Figures 28 and 29 present the distribution of the error of our localizator versus the accumulated odometry error in two different robot trajectories around the CS department. After the initial steps in which the robot it still updating the beliefs the error is delimited around 2 meters.

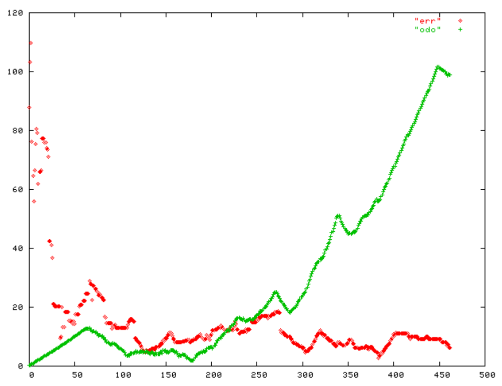


Figure 28: Distribution of errors for approach 4

Figures 30 and 31 show the performance of this algorithm working. As in the previous approach it can be seen the convergence. The stripes that appear in the probability clouds are due to the walls.

5 Conclusions and future work

Even thought the information provided by a wireless network isn't very discriminative and it's not possible to use it to find a position with precision on its

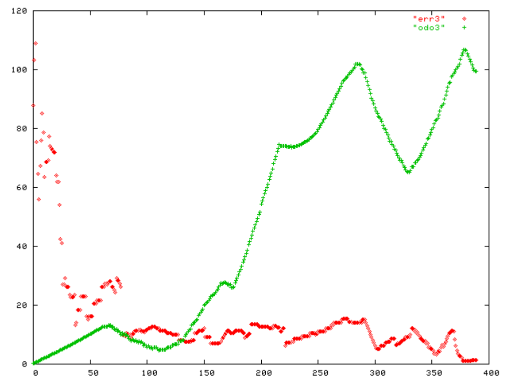


Figure 29: Distribution of errors for approach 4

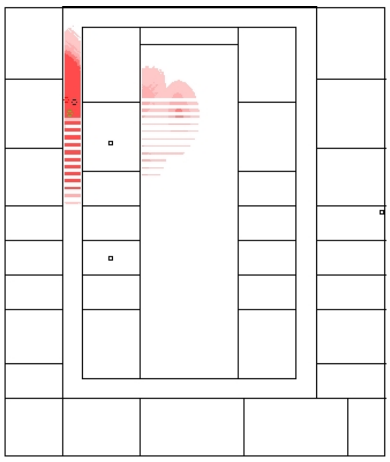


Figure 30: Localizator running using approach 4 after 50 seconds of execution

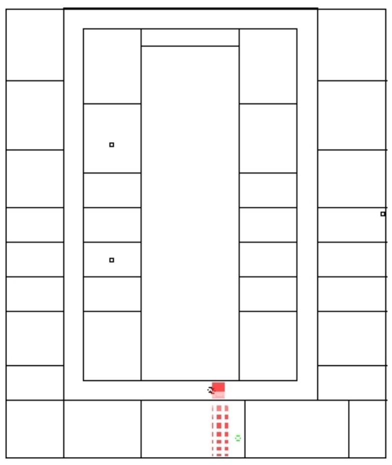


Figure 31: Localizator running using approach 4 after 300 seconds of execution

own. It's clear that wireless information can help in the robot localization task.

It's possible to localize a robot in an indoor environment using only the data provided by a wireless network combined with odometry data [21, 10, 24]. The most accurate method to do it is using a preconstructed map but this has the drawback that will only work as we have an updated map of the environment which is not always possible.

We have shown that it's also possible to realize it with a Breakpoint model of indoor radio propagation with comparable results or even better. This methods are the most promising ones as they are the only ones that doesn't need any additional information about the wireless network to work. The models give higher errors than the map at the beginning but after the initial cycles the measures can be better than using a map. In approach fourth after the initial steps in which the robot it still updating the beliefs the error is bounded around 2 meters which is a better level than using the map. Using a model also has a great advantage at the time of gathering the data as it's only needed to know the position of the AP as well as it is easier to modify at the time of making changes in the wireless network.

We have also seen that the information from the wireless card is not informative enough to estimate the orientation of the robot

In a future we are thinking in realizing these same experiments using the real robot instead of SRISim. It will also be interesting to see if the error levels (that are still high for most of the tasks it could be used in) could be reduced introducing more information from other kind of sensors as for instance sonars as well as to expand the aim of the research to alow the localization of the robot in the whole department including the offices, and or in a 3 dimensional plane including other floors.

References

[1] *I.Cox, G.Wilfong, "Autonomous robot vehicles", Springer Verlag, NW.* 1990.

[2] *J.Borenstein, B.Everett and L.Feng, "Navigating mobile robots: Systems and techniques", A.K. Peters, Ltd. Wesley, MA.* 1996.

[3] *J.M.kleinberg, "The localization problem for mobile robots".* 1996.

[4] *I.Cox, "Blanche - an experiment in guidance and navigation of an autonomous robot vehicle", IEEE transactions on Robotic and automation,.* 7(1991), pp. 193-204

[5] *M. Drumheller, "Mobile robot localization using sonar", IEEE transactions on Pattern Analisys and Machine Intelligence.* 9(1987), pp. 325-331

[6] *C. Ming Wang, "Location estimation and uncertainty analysis for mobile robots", Proc. IEEE International Conference onRobotics and Automations.* 1988, pp. 1230-1235

[7] *S.Shellhammer, "Overview of ITU-R P.1238-1, Propagation Data and Prediction Methods for Planning of Indoor Radiocommunication Systems and Radio LAN in the Frequency Band 900 MHz to 100 GHz",doc.:IEEE802.15-00/294rl.* 2000

[8] *G.Wright, "A House of mirrors: The indoor radio channel and radios for it", Lucent Technologies, Crawford Hill Laboratory and Berkeley Wireless Research Center.* 2000

[9] *A.Tadeusz, Wysocki and Hans-Jürgen Zepernick, "Characterization of the indoor radio propagation channel at 2.4GHz", Journal of telecomunications and information technology.3-4.* 2000

[10] *A.Ladd et al., "Robotics-based Location sensing using Wireless Ethernet",MOBICOM'02, September. 23-26.* 2002

[11] *M.Papadopouli, "Location sensing techniques", Mobile Computing Group, UNC.*

[12] *A.Clarke, "A Reaction Diffusion Model for Wireless Indoor Propagation", Master of Science dissertation, University of Dublin.* 2002

[13] *J.Tamminen, "2.4 GHz WLAN Radio Interface", Radionet Oy.* 2002

[14] *J.Yee, H.Pezeshki-Esfahani, "Understanding Wireless LAN Performance trade-offs", Communication Systems Design, WhitePaper, November.* 2002

[15] *D.van Heesch, "Aria Reference manual", ActivMedia Robotics.* 2002

[16] *M.I.Ribeiro and P.Lima, "Markov Localization", Instituto de Sistemas e Robótica (ISR) Lisboa.* 2002

[17] *M.A.Crespo, J.M.Cañas, "Localización probabilística en un robot con visión local", Trabajo Fin de Carrera, Departamento de Sistemas Inteligentes aplicados, Universidad Politécnica de Madrid.* 2003

[18] *S.Thrun, "Probabilistic Algorithms in Robotics", AIMagazine,21(4):93-109.* April 2000.

[19] *G.E.P.Box, M.E.Muller, "A note on the generation of random normal deviates",Annals Math. Stat, V. 29, pp. 610-611.* 1958

[20] *R.Y.Rubinstein, "Simulation and the Monte Carlo method", John Wiley and Sons, ISBN 0-471-08917-6.* 1981

[21] *J.Small, A.Smailagic, D.P. Siewiorek, "Determining user location for context aware computing through the use of a wireless LAN infrastructure", Institute for Complex Engineered Systems, Carnegie Mellon University.*

[22] *S.Thrun, "Robotic Mapping: A Survey", Technical Report CMU-CS-02-111, Carnegie Mellon University, Computer Science Department, Pittsburgh, PA.* 2003

[23] *A. Iglesias, "computer-aided geometric design and computer graphics: line drawing algorithms", Department of Applied Mathematics and Computational Sciences, University of Cantabria, UC-CAGD Group.* 2001

[24] *S.M.Siddiqi, G.S. Sukhatme, A.HOward, "Experiments in Monte-Carlo localization using WiFi Signal strength", The 11th International conference on advanced robotics, Coimbra, Portugal.* July 2003

[25] *Luis Rodero Merino, Miguel Ángel Ortuño Pérez, Jesús M. González Barahona, Vicente Matellán Olivera, "Monitorización de redes 802.11 con Linux". XIII Telecom I+D 2003.* November 2003