Localization Performance Quantification by Information Theory

Rafael Berkvens* Herbert Peremans† Maarten Weyn*

March 27, 2015

Abstract

We want to quantify the localization performance of exteroceptive sensors to identify optimal combinations. Current methods generally use the localization error, but this is very situation specific and does not let us compare the sensors without extensive domain specific knowledge. We propose to use information theory to quantify the performance of sensor models used in localization algorithms. We demonstrate this approach using a camera, a geomagnetic flux sensor, a radio frequency identification reader, and a Wi-Fi scanner, which we use in three different environments. Our results show that the mean mutual information between the positions in the environment and a set of measurements is a valid localization performance metric. We conclude that information theory can estimate the localization performance of a given sensor model in an environment without making assumptions about the posterior distribution, which is necessary when expressing the localization performance as mean spatial error. Additionally, the mean mutual information allows to quantify the performance increases when complementary sensors are combined.

1 Introduction

We are interested in quantifying the localization performance of exteroceptive sensors solely by virtue of their sensor model. We define exteroceptive sensors as sensors that perceive the environment, such as GPS, camera, LIDAR, or Wi-Fi. Continuous localization, or tracking, is often aided by sensors that perceive the state of the object to be localized, like odometers, accelerometers, or gyroscopes, which we call proprioceptive sensors. A localization algorithm uses measurements from such exteroceptive or proprioceptive sensors, or a combination of those, to improve its guess of the location of the device being localized.

The localization performance of a specific sensor strongly depends on the underlying sensor model. This model is responsible for translating a sensor measurement into a likelihood over all possible positions. A localization algorithm will use this likelihood to obtain a position estimation. The performance of a localization algorithm is generally indicated as a mean distance with a certain standard deviation between the estimated position and the true position. While this is a valid metric to compare localization algorithms, the additional processing and assumptions required in the localization algorithm can hide the actual performance of the underlying sensor model. It also assumes a unimodal Gaussian distribution for the position estimate, which is seldom correct.

In this article, we present a system which is not dependent on the aforementioned assumptions and which enables the user to formulate conclusions without extensive domain specific knowledge of the sensors.

We propose to measure the performance of sensor models used in localization algorithms by calculating the mean mutual information between all positions in the environment and the measurements that can be performed at a certain position. This metric is specific for a certain position in an environment: at different positions in the environment, and in different environments, the results are different. When we know the performance of a sensor at a certain position, we can increase the performance by modifying the environment, we can choose a different sensor, or combine the sensor with a complementary one.

Our contribution is a general approach of calculating the mean mutual information between the positions and measurements. We apply our approach to four different sensors, namely a camera, a geomagnetic flux sensor, a radio frequency identification (RFID) reader, and a Wi-Fi scanner.

*CoSys-Lab, FTI, University of Antwerp, Paardenmarkt 92, B-2000 Antwerp, Belgium rafael.berkvens@uantwerpen.be
†ENM, FTEW, University of Antwerp, Prinsstraat 13, B-2000 Antwerp, Belgium

Additionally, we show that our approach allows to easily obtain a result for the increase of mean mutual information when combining two exteroceptive sensors.

This article continues as follows: In Section 2 we analyze literature to support our claim that current state-of-the-art uses accuracy to measure the performance of localization systems. In Section 3 we begin with explaining our proposed method using information theory in general. Next, we describe the implementation of the sensor models that were used to demonstrate our method. We also describe the testing environments and give some details on the practical implementation. In Section 4 we show the usefulness of our method by applying it to the sensor models in the testing environments. We discuss these results for every sensor. In Section 5 we summarize our conclusions.

2 Background

We find that in literature, the performance of a localization system is usually measured by its accuracy, *i.e.*, its spatial error. The spatial error of a localization system seems the most intuitive metric to evaluate a localization system as testified by its widespread use in a broad sample of research on localization. Renaudin and Combettes [14] propose Magnetic, Acceleration Fields, and Gyroscope Quaternion (MAGYQ), an algorithm to estimate the position of pedestrians in indoor environments, where they focus on improving the heading direction estimation. They find that their algorithm keeps the heading error below 10° after a 1.5 km walk. To compare their technique with the integration of the gyroscope signal and GPS positions, they present a metric map of the trajectory.

Ito *et al.* [10] describe a localization algorithm based on Wi-Fi and a depth camera. The Wi-Fi scanner is used to initialize their algorithm, reducing the probability of estimating the location of the robot in a local minima. They express the global localization performance increase over using only a depth camera in a cumulative histogram of the spatial error and present graphs of the Root Mean Square (RMS) spatial error over time.

Biswas and Veloso [3] describe a localization system based on a depth camera. A cumulative histogram displays the spatial error of their system. Eckert *et al.* [7] describe a localization system for a four rotor flying robot and give a histogram of the height error. Hornung *et al.* [9] describe a localization algorithm for a Nao humanoid robot. They give the mean and standard deviation of the spatial and rotational error over time. In another paper Biswas and Veloso [4] describe a Wi-Fi localization system for mobile robots. Here, they provide a spatial error based on the number of visible access points, and additionally the cumulative histogram of spatial error. Also Thrun *et al.* [17] use spatial error to estimate the accuracy of their robust Monte Carlo localization.

In most of the examples described above, sensor fusion happens by multiplication of the posterior distributions provided by the sensor model of the various sensors, as clearly demonstrated by [9]. According to the recent information fusion survey by Khaleghi *et al.* [11], this is referred to as probabilistic fusion, a part of the fusion of imperfect data techniques. Localization can be seen as dealing with imperfect data, every sensor having its typical shortcomings. In another survey, van Laere [18] indicates that it is difficult to evaluate sensor fusion performance, because the metrics usually have to be adapted for each environment. We see this in localization, where performance is dependent on the environment: Wi-Fi localization works better when there are many Wi-Fi access points available, algorithms using camera images have trouble when there is no light, or when locations are visually similar, and those using laser range finders have difficulties with plain, square rooms, to name but a few.

The approaches described above measure sensor performance indirectly by quantifying the performance of a specific localization scheme using the sensor's measurements. Furthermore, a spatial error, assumed to be normally distributed, is used as the standard performance measure. We propose to quantify the performance of the sensor in a manner that is both more direct, *i.e.*, not associated with a particular localization scheme, and more general, *i.e.*, not assuming the location estimates to be normally distributed.

3 Methods

To quantify the localization performance of a measurement model, we generalize the approach discussed by Steckel and Peremans [15], where the mean mutual information is calculated between a position and the measurements that can be performed at a certain position. In other words, what does a sensor measurement tell us about our position. We start with the mutual information between two random variables as defined by Cover [6], and use the notation from MacKay [12]:

$$I(X;Y) \equiv \sum_{xy \in \mathcal{A}_X \mathcal{A}_Y} P(x,y) \log \frac{P(x,y)}{P(x)P(y)}, \tag{1}$$

where X and Y are ensembles which are defined as triples consisting of an outcome x and y, respectively, an alphabet $\mathcal{A}_X$ and $\mathcal{A}_Y$, and the probabilities $\mathcal{P}_X$ and $\mathcal{P}_Y$. An outcome x is a random variable that has a value from $\mathcal{A}_X$, say a_o, with a probability from $\mathcal{P}_X$, p_o. The number of elements in both $\mathcal{A}_X$ and $\mathcal{P}_X$ is O, with a_o having a probability of p_o. The function $P(x)$ indicates the probability of the outcome x, which is p_o in this situation. In the following, we define the ensemble $P = (p, \mathcal{A}_P, \mathcal{P}_P)$ for all possible positions in the environment, assumed to be finite. A position is defined as both a location and a heading, hence our choice for P. To avoid confusion, the function $P(x)$ will always indicate the probability of x and we will not use the shorthand p for the probability of $x = a$, but use p to indicate a position. We also define the ensemble $M_{p_i} = (\vec{m}_{p_i}, \mathcal{A}_{M_{p_i}}, \mathcal{P}_{M_{p_i}})$ for all possible measurements at the position p_i. We emphasize with our notation that a measurement is a vector. Inserting our ensembles into Equation (1) gives:

$$I(P; M_{p_i}) \equiv \sum_{p\vec{m}_{p_i} \in \mathcal{A}_P \mathcal{A}_{M_{p_i}}} P(p, \vec{m}_{p_i}) \log \frac{P(p, \vec{m}_{p_i})}{P(p)P(\vec{m}_{p_i})}, \tag{2}$$

where we assume that in all this article, a logarithm is a base two logarithm, thus the resultant mutual information is expressed in bits. A measurement is sampled from the Gaussian distribution $\mathcal{N}$ of the actual measurements at position p_i:

$$\vec{m}_{k,p_i} = \mathcal{N}\left(\overline{M^+_{p_i}}, \Sigma_{M^+_{p_i}}\right), \tag{3}$$

where $M^+_{p_i}$ are the actual sensor measurements performed at the position p_i; $\overline{M^+_{p_i}}$ is the sample mean of $M^+_{p_i}$; and $\Sigma_{M^+_{p_i}}$ is the covariance matrix of $M^+_{p_i}$.

The sensor model, *i.e.*, the probability density function of the sensor measurement $\vec{m}$ given the position p_j is described by:

$$P(\vec{m}_{p_i} \mid p = p_j) = \frac{1}{\sqrt{(2\pi)^{|M^+_{p_j}|} \det \Sigma_m}} \exp\left(-\frac{1}{2}\left(\vec{m}_{p_i} - \overline{M^+_{p_j}}\right)^T \Sigma_m^{-1} \left(\vec{m}_{p_i} - \overline{M^+_{p_j}}\right)\right), \tag{4}$$

where p_j is an element of all possible positions P; $|M^+_{p_j}|$ is the dimension of the measurements in $M^+_{p_j}$; and Σ_m is the covariance matrix of the sensor model of the given sensor. This is just a theoretical sensor model to aid the current discussion; the actual sensor models will be discussed below. Any probabilistic sensor model can be used to calculate a likelihood for a measurement given a position.

The posterior probability of the position p_j given a particular sensor measurement $\vec{m}$ is calculated using Bayes:

$$P(p = p_j | \vec{m}_{p_i}) = \frac{P(\vec{m}_{p_i} \mid p = p_j)P(p = p_j)}{\sum_{p \in \mathcal{A}_P} P(\vec{m}_{p_i} \mid p)P(p)}, \tag{5}$$

where we assume a uniform distribution for $P(p = p_j)$ since we do not have any prior information about the position. The marginal probability of a measurement $\vec{m}$ performed at position p_i is not known, so we first calculate the mutual information [6, 12] between all possible position P and the measurement $\vec{m}_{k,p_i}$:

$$I(P; \vec{m}_{p_i} = \vec{m}_{k,p_i}) = \sum_{p \in \mathcal{A}_P} P(p | \vec{m}_{p_i} = \vec{m}_{k,p_i}) \log \frac{P(p \mid \vec{m}_{p_i} = \vec{m}_{k,p_i})}{P(p)}, \tag{6}$$

where $\vec{m}_{k,p_i}$ is a measurement sampled from the distribution of the actual measurements at position p_i, see Equation (3).

Finally, we establish the mean mutual information by Monte Carlo approximation [16] from the mutual information between all possible positions P and the measurement $\vec{m}_{k,p_i}$ as follows:

$$< I(P; M_{p_i}) > \cong \frac{1}{K} \sum^{K} I(P; \vec{m}_{p_i} = \vec{m}_{k,p_i}), \tag{7}$$

where K is the number of Monte Carlo samples.

In this article, we study four different sensors, each with a specific sensor model. We will use these sensor models to present our localization performance quantification approach. The sensors are a camera, a geomagnetic flux sensor, a RFID reader, and a Wi-Fi scanner. For the camera,

geomagnetic flux, and Wi-Fi sensors, we used a kernel matching sensor model. For the RFID reader we used a custom binary sensor model. For every sensor, various other sensor models exist; we choose our sensor models with the purpose of demonstrating mean mutual information as a performance metric, rather than demonstrating their particular fitness over sensor models not mentioned in this article. We test our approach in three different environments. The localization performance of the sensors varies by environment, which we can quantify with our approach. The environments are different locations in the university's buildings, so we use the university's labels to identify them. The first environment is CPM.E2, the CoSys-Lab offices, the second environment is CST.R0, an oval donut shaped hall of 60 m at the long diameter, and the third environment is CPM.D4, a corridor of 80 m.

The first sensor we study is camera, for which we implement the sensor model used in [1]. This sensor model stores images as a single row vector, after reducing the resolution to twelve times nine pixels. Milford [13] proves that such a reduced image still contains enough information to enable localization. When referring to the camera sensor model, we denote the measurement vector $\vec{m} = \vec{c}$. The vectors are compared pixel by pixel using a kernel matching approach to calculate the likelihood of position p_j, with the assumption that the pixels are independent; the general sensor model in Equation (4) becomes:

$$P(\vec{c} \mid p_j) = \prod_{n}^{N} \frac{1}{\sqrt{2\pi}\sigma_c} \exp - \frac{\left(c_n - \overline{C^+_{n,p_j}}\right)^2}{2\sigma_c^2}, \tag{8}$$

where $\vec{c}$ is a camera measurement, *i.e.*, the single row vector mentioned earlier; $C^+_{p_j}$ is the set of actual camera measurements at position p_j; c_n is the nth pixel of the camera measurement; and σ_c is the sensor model's kernel width. The kernel width is set to 2, as was the case in [1].

The second sensor we study is geomagnetic flux, for which we implement the sensor model used in [2, 19]. This model uses the x', y', and z' components of the earth's magnetic B field, measured with a magnetometer. The $'$ indicates that the components are not calibrated, resulting in measurements that are dependent on the pose of the sensor. When referring to the flux sensor model, we denote the measurement vector $\vec{m} = \vec{f}$, with $\vec{f} = \{x', y', z'\}$. The vectors are compared component-wise using a kernel matching approach to calculate the likelihood of position p_j, with the assumption that the components are independent; the theoretical sensor model in Equation (4) becomes:

$$P(\vec{f} \mid p_j) = \prod_{x',y',z'} \frac{1}{\sqrt{2\pi}\sigma_f} \exp - \frac{\left(f_n - \overline{F^+_{n,p_j}}\right)^2}{2\sigma_f^2}, \tag{9}$$

where $\vec{f}$ is a flux measurement; $F^+_{p_j}$ is the set of actual flux measurements at position p_j; f_n is the x', y', or z' value of the corresponding component of the magnetic flux measurement; and σ_f is the sensor model's kernel width. The kernel width is set to 0.67 µT, as was the case in [2].

The third sensor we study is a passive, 13.56 MHz RFID reader, for which we implement a custom sensor model. RFID tags can be either active or passive, with an active tag being powered to transmit its identification number, and a passive tag using the power of the reader to communicate its identification number. We distributed passive tags in the CPM.E2 environment as a ground truth for an earlier project. These tags have a limited range, in the order of 20 mm, so the RFID reader must be close to the tag to obtain its identification number. Consequently, when a measurement returns the identification number of a position p_j, it must have been performed at p_j. As the system is highly reliable, we exclude misidentification of RFID tags. In addition, for positions that include a tag, it might not always be measured. Our model incorporates this probability, but in implementation we set it to zero. If no identification number is measured at a position, we do not know where the measurement is performed, except that it cannot have been performed at locations where a tag is present. Since the measurement of an RFID reader is a single identification number, our measurement vector becomes a scalar, $\vec{m} = r$, where r can also be empty when no tag is measured. We apply the rules mentioned above so that the general sensor model in Equation (4) becomes:

$$P(r \mid p_j) = \begin{cases} P_d & \text{if } r^+_{p_j} = t_{p_j} \text{ and } r = t_{p_j}, & \text{(10a)} \\ 1 - P_d & \text{if } r^+_{p_j} = t_{p_j} \text{ and } r = \varnothing, & \text{(10b)} \\ 0 & \text{if } r^+_{p_j} = t_{p_j} \text{ and } r \neq t_{p_j}, & \text{(10c)} \\ 1 & \text{if } r^+_{p_j} = \varnothing \text{ and } r = \varnothing, & \text{(10d)} \\ 0 & \text{if } r^+_{p_j} = \varnothing \text{ and } r \neq \varnothing, & \text{(10e)} \end{cases}$$

where r is an RFID measurement; P_d is the probability of detection of the tag when it is present; $r^+_{p_j}$ is the actual measurement at position p_j; and t is the identification number of a tag. When the measurement r is empty, it means that the RFID reader did not find any tag. We model our reader in the assumption that if a tag is present at a position, it will always be correctly read, in other words, P_d is one.

The last sensor we study is a Wi-Fi scanner, for which we adopt the sensor model used in [20]. In this case, our measurement vector contains the received signal strength (RSS) values of every access point in the environment, $\vec{m} = \vec{w}$, where $\vec{w} = \{w_1, w_2, \ldots, w_A\}$ with A the set of access points in the environment, so w_a is the RSS value of access point a in measurement $\vec{w}$. If the signal strength of an access point is very low at a certain position, our Wi-Fi scanner might not pick up the signal, and we say that we did not see the access point at that position. The vectors are compared by access point to calculate the likelihood of position p_j, with the assumption that they are independent; the general sensor model in Equation (4) becomes:

$$P(\vec{w} \mid p_j) = \prod_a^A P\left(w_a \mid p_j\right), \tag{11}$$

where $\vec{w}$ is a Wi-Fi measurement. We define four mutually exclusive events when comparing the RSS values of an access point in the set of measurements collected at p_j denoted by $W^+_{p_j}$ with our new measurement denoted by w: either a *hit*, a *miss*, an *extra*, or a *none*. A *hit* occurs when the access point has an RSS value both in the collection of measurements and in the new measurement. A *miss* occurs when the access point has an RSS value in the collection, but was not seen in the new measurement. An *extra* occurs when the access point has an RSS value in the new measurement, but was not seen in the collection of measurements at p_j. Lastly, a *none* occurs when the access point was not seen both in the collection and in the new measurement. The probability of w_{a,k,p_i} in Equation (11) is defined as:

$$P(w_a \mid p_j) = \begin{cases} P(w_a, hit|p_j) & \textit{hit}\text{: } W^+_{a,p_j} \neq \varnothing \text{ and } w_a > w_t\ , \quad (12a) \\ P(w_a, miss|p_j) & \textit{miss}\text{: } W^+_{a,p_j} \neq \varnothing \text{ and } w_a < w_t\ , \quad (12b) \\ P(w_a, extra|p_j) & \textit{extra}\text{: } W^+_{a,p_j} = \varnothing \text{ and } w_a > w_t\ , \quad (12c) \\ P(w_a, none|p_j) & \textit{none}\text{: } W^+_{a,p_j} = \varnothing \text{ and } w_a < w_t\ , \quad (12d) \end{cases}$$

where W^+_{a,p_j} is the set of measurements collected at position p_j that included access point a; w_t is a threshold RSS value, under which our hardware will no longer detect an access point—if we do not detect an access point, we say that we are under this threshold; and the mutually exclusive events are calculated as:

$$P(w_a, hit|p_j) = \frac{1}{\sqrt{2\pi}\sigma_w} \exp - \frac{\left(w_a - \overline{W^+_{a,p_j}}\right)^2}{2\sigma_w^2} \times P(W^+_{a,p_j} \neq \varnothing|p_j), \tag{13a}$$

$$P(w_a, miss|p_j) = \frac{1}{\sqrt{2\pi}\sigma_w} \int_{-\infty}^{w_t} \exp - \frac{\left(x - \overline{W^+_{a,p_j}}\right)^2}{2\sigma_w^2} dx \times P(W^+_{a,p_j} \neq \varnothing|p_j), \tag{13b}$$

$$P(w_a, extra|p_j) = \int_{-\infty}^{\infty} \alpha \left[\frac{1}{\sqrt{2\pi}\sigma_w} \int_{-\infty}^{w_t} \exp - \frac{\left(x - \overline{W^+_{a,p_j}}\right)^2}{2\sigma_w^2} dx \right]^N \ldots \frac{1}{\sqrt{2\pi}\sigma_w} \exp - \frac{\left(w_a - \overline{W^+_{a,p_j}}\right)^2}{2\sigma_w^2} d\overline{W^+_{a,p_j}} \times P(W^+_{a,p_j} = \varnothing|p_j), \tag{13c}$$

Figure 1: The grid map and trajectory of the ground truth for the three testing environments.

$$P(w_a, none|p_j) = \int_{-\infty}^{\infty} \alpha \left[\frac{1}{\sqrt{2\pi}\sigma_w} \int_{-\infty}^{w_t} \exp - \frac{\left(x - \overline{W^+_{a,p_j}}\right)^2}{2\sigma_w^2} dx \right]^N \dots \left[\frac{1}{\sqrt{2\pi}\sigma_w} \int_{-\infty}^{w_t} \exp - \frac{\left(x - \overline{W^+_{a,p_j}}\right)^2}{2\sigma_w^2} dx \right] d\overline{W^+_{a,p_j}} \times P(W^+_{a,p_j} = \varnothing | p_j), \tag{13d}$$

where σ_w is the sensor model's kernel width; and $P(W^+_{a,p_j} \neq \varnothing|p_j)$ is the chance of measuring an access point at position p_j. The threshold w_t is set to -89 dBm, based on our hardware specifications. The kernel width is set to 3.46 dBm, based on [5].

Our three testing environments are CPM.E2, CST.R0, and CPM.D4. The occupancy grid maps constructed using a laser range finder are displayed in Figure 1. The length of the traveled path in CPM.E2 is 440.08 m, in CST.R0 this is 1104.75 m, and in CPM.D4 this is 490.45 m. The data was collected using a Pioneer 3DX mobile robot with on-board odometry. A SICK LMS100 laser range finder was mounted on the front of the robot. A PlayStation®Eye Camera was mounted on top of the robot to collect the camera images. An LG Nexus 5 smartphone was also mounted on top of the robot to collect the geomagnetic flux measurements, but it was elevated from the base of the robot by a cardboard box to prevent interference from the robot's metal base. A passive RFID reader was attached under the robot to scan for RFID tags on the ground, which were only available in the CPM.E2 environment. Lastly, a laptop was mounted on top of the robot to collect the Wi-Fi measurements.

The Simultaneous Localization And Mapping (SLAM) algorithm used to construct these maps, and the position ground truth, is the algorithm described in [8]. This algorithm is implemented in the Robot Operating System (ROS).

To divide the actual measurements into positions, we create a three-dimensional grid. The first two dimensions indicate the x and y location, while the third dimension indicates the heading direction, θ. The grid cell width is $(0.3\,\text{m}, 0.3\,\text{m}, 0.25\pi\,\text{rad})$. We place the origin of the grid on the first point of the traveled path. All positions in the traveled path that fall within the same grid cell are said to belong to the same position p. In our set of possible positions, we only include those grid cells that contain positions of the traveled path. Because we know when our robot was in a certain grid cell, we know which measurements belong to a certain position, and we can define the set of actual measurements M_p^+ for each position p.

The noise model in Equation (3) is implemented slightly differently for every sensor. For our camera images, we calculate the mean and variance value for every pixel of the image vector. We apply the noise model for every pixel independently. Because of the chose grid size, there is much variance in images belonging to the same grid cell, which will influence the results—we come back to this in the discussion. We do the same with the geomagnetic flux measurements, for the x', y', and z' components. These measurements do not suffer from the same problem as camera images for the chosen grid size. The RFID noise model is different, because we model a sensor without

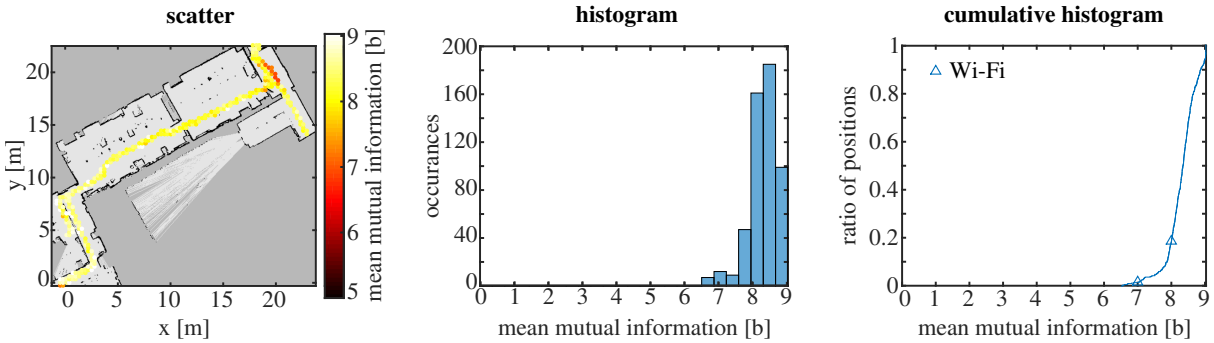


Figure 2: The mean mutual information in CPM.E2 for Wi-Fi, once by location, once as a histogram, and once as a cumulative histogram. Do note the color scale of the scatter plot.

Table 1: The mean mutual information necessary for a sensor to indicate a single position. This is the base two logarithm of the number of grid cells, which depends on the positions where sensor measurements are obtained.

Sensor	CPM.E2	CST.R0	CPM.D2
Camera	9.35 bit	11.39 bit	10.27 bit
Flux	9.35 bit	11.39 bit	10.23 bit
RFID	9.17 bit	N.A.	N.A.
Wi-Fi	9.03 bit	11.16 bit	9.99 bit

noise. However, not every RFID measurement in a certain grid cell will detect a tag, even if there is a tag inside the grid cell. This is because a grid cell is larger than the tag detection range, so a measurement may be performed in the grid cell, but without detecting the tag. Additionally, we did not try to align the grid with the tag locations, so a tag may exist on the border of two grid cells. The noise model provides the probability of measuring the RFID tag in the grid cell. If a tag is detected, the identification number of the tag will always be correctly read. The Wi-Fi noise model is a combination of these two types of noise models. To generate a Wi-Fi measurement, we choose an access point in the environment based on the probability of seeing that access point at the current position. This probability is based on the observations in the grid cell. If the access point is detected, the RSS value is modeled by the mean and variance values of the actual measurements at the current position.

4 Results and Discussion

We will now show the results of applying the sensor models described above. Note again that we do not intend to demonstrate the superiority of any of the sensors or sensor models used in this article. Rather, we wish to demonstrate the usefulness of expressing the performance of the sensor model by its mean mutual information. We calculate the mean mutual information for every position in the environment visited during the exploration phase, where a position is expressed by a x, y, and θ coordinate. We show in Figure 2 a two-dimensional plot of the mean mutual information of the Wi-Fi sensor model in the CPM.E2 environment, labeled 'scatter' and located on the left side of the figure. In the middle of the figure, we show a histogram of the same data, which makes it easier to interpret the distribution of mean mutual information, but discards the position data. The position data is interesting, but for the scope of this article we will focus on the distribution of mean mutual information. On the right side of the figure, we show a cumulative histogram. This type of plot enables us to compare the mean mutual information with various sensor models.

We will extensively use this cumulative histogram, where we will normalize the mean mutual information to compare sensors between environments. The maximum mean mutual information for the four sensors in the three environments can be found in Table 1. These values are based on the base two logarithm of the maximum number of grid cells with sensor data in the environment, which is the amount of information necessary to describe a single position.

To discuss the relative mean mutual information of the four sensors in the three environments, we will use the cumulative histograms in Figure 3. The simple sensor model we use for the camera

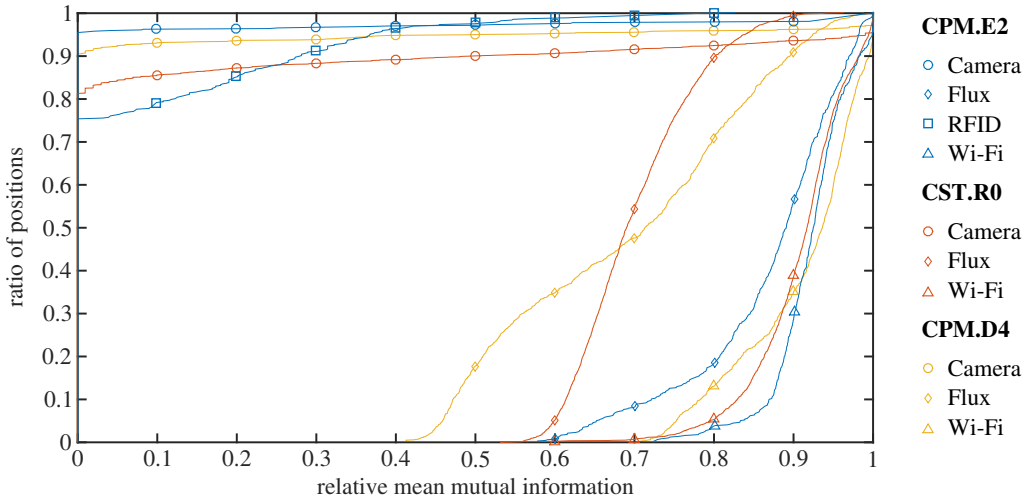


Figure 3: A cumulative histogram of the relative mean mutual information, for the four sensors in the three environments.

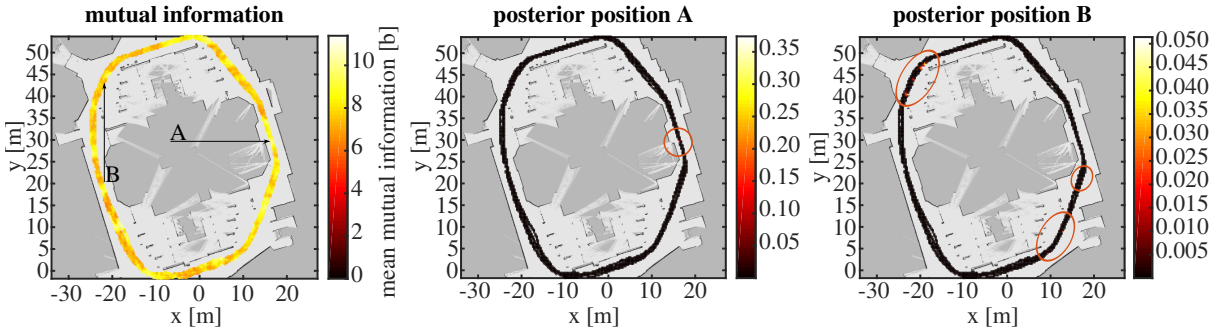


Figure 4: The mean mutual information by position and the posterior distribution of the position estimate for a measurement collected at position A and position B in the environment CST.R0 for the geomagnetic flux sensor.

has a mean mutual information of zero at 80 % of positions. At these positions, sample images are created from a large collection of images taken at a $(0.3\,\text{m}, 0.3\,\text{m}, 0.25\pi\,\text{rad})$ grid size. There is much visual variation in such a grid cell, which results in a high variance when sampling images, explaining the low information content of these measurements. The positions with a mean mutual information different from zero have a low number of actual images taken in the corresponding grid cell, causing an artificially low variance when sampling images. This is an example of a sensor model that would perform better with a smaller grid size.

The geomagnetic flux sensor model that we used for this study shows a large performance dependency on the environments in which we performed our tests. We hypothesize that geomagnetic flux localization with this sensor model is better for smaller environments, since we see the relative mean mutual information decrease in the larger environments CST.R0 and CPM.D4. In [2], we found that the x', y', and z' values are unique for different positions in small environments such as a room, but can be found repeatedly in larger environments.

The RFID sensor model knows its position only when the position contains a tag; if the position does not contain a tag, it might be on any position without a tag. This behavior is comparable to that of the camera sensor model, which was specifically designed to be very precise. The CPM.E2 environment is the only environment in which this sensor was tested.

The Wi-Fi sensor model performs better than the other sensor models in all environments. All these environments are equipped with many Wi-Fi access points, to provide wireless network access to students.

Figure 4 shows the relationship between the mean mutual information and the posterior distribution of the position estimate. As in Figure 2, we average over the θ dimension to obtain a

two-dimensional plot. We select two positions with different mean mutual information and display the posterior distribution of the position estimate for a measurement collected at these positions. Because we use the mean posterior because when we obtain the mean mutual information by Monte Carlo sampling, see Equation (7), we also had to sample the sensor model's posterior for each position. Position A has a mean mutual information of 9.57 bit, close to the maximum mean mutual information for geomagnetic flux in the CST.R0 environment, see Table 1. The corresponding posterior distribution is very selective, with the probability mass function concentrated on positions very close to position A. Position B has a mean mutual information of 6.83 bit. The corresponding posterior distribution is not as selective as with position A, but significant probabilities can again be found mainly in the vicinity of position B. They are not arranged around position B, but exist in small groups close to the position, with the highest posterior value not being at position B. There are also significant posterior values at the opposite side of the environment, which we assume to be caused by the symmetric architecture of the environment. Note that these posteriors clearly show the multi-modal nature of the position estimate distribution, making the use of spatial error, based on an implicit assumption of a unimodal distribution, as a performance measure less appropriate.

Figure 5 shows three plots for the combination of the camera, geomagnetic flux, and Wi-Fi sensor models. We produced these results by multiplying the localization posteriors. These results are interesting, because they show to what extent the relative mean mutual information increases when we increase the available information by adding a sensor. Combinations with the camera sensor model result in a mostly insignificant increase in mean mutual information. The combination of the geomagnetic flux and Wi-Fi sensor model has a significant increase in mean mutual information. This indicates a complementarity between the two sensors.

Upon inspection of the results, we assume that there may be a correlation between the number of measurements collected at a position and the mean mutual information at this position. In other words, positions that have a more accurate distribution of actual measurements due to an increased number of actual measurements, see Equation (3), might be more easily recognized by the sensor model. In the supplementary material, we provide figures to analyze this correlation. We find that the mean mutual information is not related to the number of actual samples. However, for the camera sensor model there is more mean mutual information when very few measurements are used to construct the distribution from which to sample. This seems evident, because there can be a lot of visual variation in a grid cell of 0.3 m, decreasing the uniqueness of the average image when more measurements are performed at a position.

5 Conclusions

We showed the use of information theory to quantify the localization quality of sensor models. Our approach gives a more accurate performance of the sensor model than conventional spatial error, since we do not need an additional algorithm to deal with multi-modal distributions or determining a specific location. In other words, we directly compare the sensor model localization performance, instead of comparing the localization algorithm's performance.

We demonstrated this approach for a camera, a geomagnetic flux sensor, a passive RFID reader, and a Wi-Fi scanner. We used an appropriate sensor model for each sensor and established the mean mutual information for each sensor in three different environments. The result depends both on the sensor model and on the environment. We also demonstrated the combination of the posterior localization of sets of two sensor models, resulting in an increase in mean mutual information.

Our technique allows comparing sensor models more accurately. It can be used for environment analysis, to determine which sensor or sensor model can best be used in a certain environment. It can be used to determine if two sensors are complementary under certain circumstances. In the future, we plan to study whether the mutual information can also be used as a penalty function to apply regression over certain sensor model parameters. In particular, whether regression over the grid cell width can determine the optimal localization scale of the camera sensor model. Additionally, we plan to extend the technique to tracking, incorporating proprioceptive sensors.

Conflict of Interests

The authors declare that there is no conflict of interests regarding the publication of this paper.

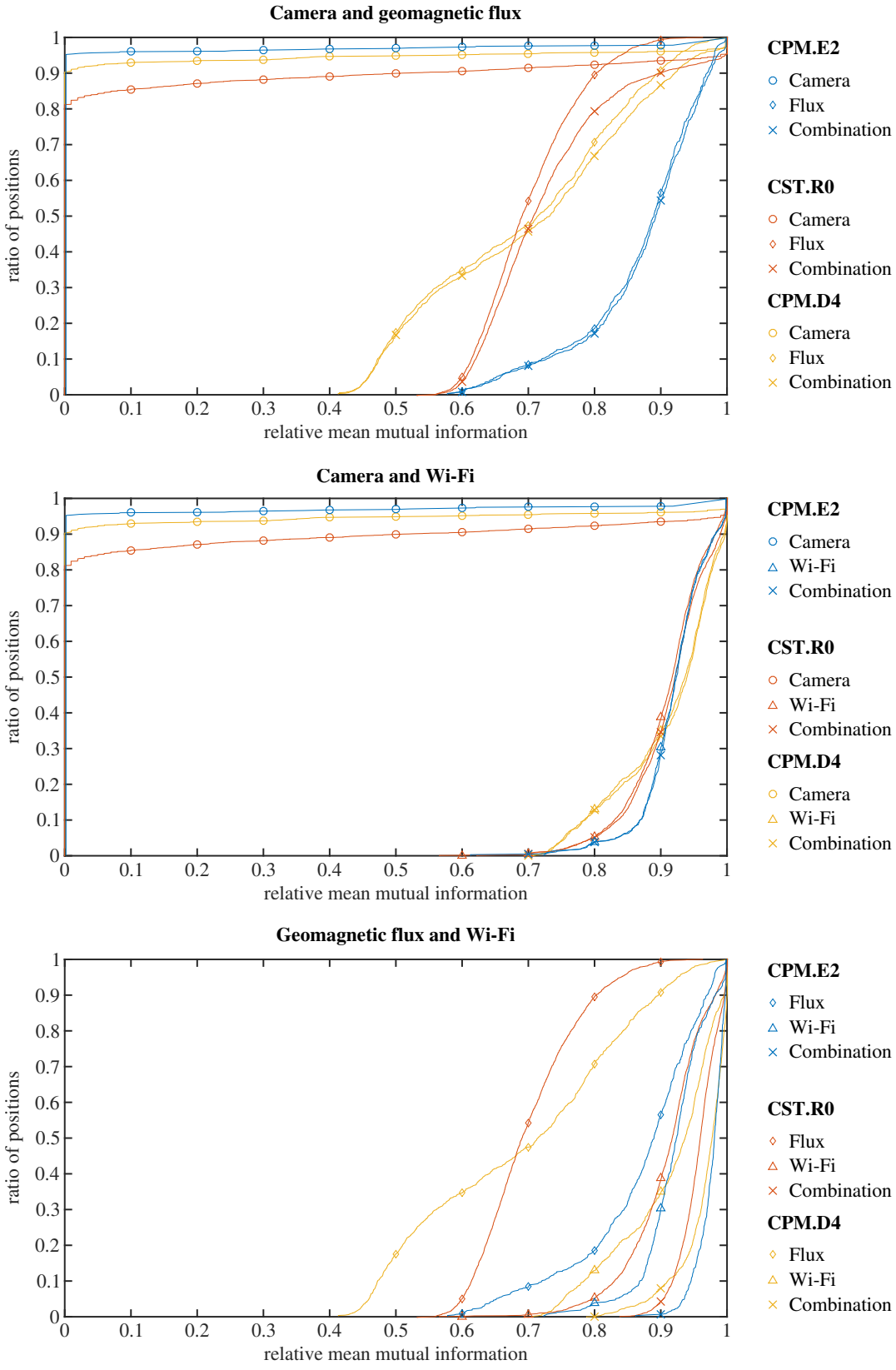


Figure 5: The relative mean mutual information of the probabilistic combination of the localization posteriors of the sensor models.

References

[1] Rafael Berkvens, Adam Jacobson, Michael Milford, Herbert Peremans, and Maarten Weyn. Biologically Inspired SLAM Using Wi-Fi. In *Intelligent Robots and Systems (IROS), 2014 IEEE/RSJ International Conference on*, pages 1804–1811, Chicago, IL, USA, 2014. IEEE.

[2] Rafael Berkvens, Dries Vandermeulen, Charles Vercauteren, Herbert Peremans, and Maarten Weyn. Feasibility of Geomagnetic Localization and Geomagnetic RatSLAM. *International Journal On Advances in Systems and Measurements*, 7(1&2):44–56, 2014.

[3] J Biswas and M Veloso. Depth camera based indoor mobile robot localization and navigation. In *Robotics and Automation (ICRA), 2012 IEEE International Conference on*, pages 1697–1702, Saint Paul, MN, USA, 2012. IEEE.

[4] Joydeep Biswas and Manuela Veloso. WiFi localization and navigation for autonomous indoor mobile robots. In *Robotics and Automation (ICRA), 2010 IEEE International Conference on*, pages 4379–4384, Anchorage, Alaska, 2010. IEEE.

[5] Yih-Shyh Chiou, Chin-Liang Wang, Sheng-Cheng Yeh, and Ming-Yang Su. Design of an adaptive positioning system based on WiFi radio signals. *Comput. Commun.*, 32(7-10):1245–1254, May 2009.

[6] Thomas M. Cover and Joy A. Thomas. *Elements of Information Theory*. John Wiley & Sons, New York, NY, 2 edition, 1991.

[7] Juergen Eckert, Reinhard German, and Falko Dressler. An Indoor Localization Framework for Four-Rotor Flying Robots Using Low-Power Sensor Nodes. *Instrumentation and Measurement, IEEE Transactions on*, 60(2):336–344, February 2011.

[8] Giorgio Grisetti, Cyrill Stachniss, and Wolfram Burgard. Improved Techniques for Grid Mapping With Rao-Blackwellized Particle Filters. *IEEE Transactions on Robotics*, 23(1):34–46, February 2007.

[9] Armin Hornung, Kai M. Wurm, and Maren Bennewitz. Humanoid robot localization in complex indoor environments. In *Intelligent Robots and Systems (IROS), 2010 IEEE/RSJ International Conference on*, pages 1690–1695, Taipei, Taiwan, 2010. IEEE.

[10] Seigo Ito, Felix Endres, Markus Kuderer, Gian Diego Tipaldi, Cyrill Stachniss, and Wolfram Burgard. W-RGB-D: Floor-plan-based indoor global localization using a depth camera and WiFi. In *Robotics and Automation (ICRA), 2014 IEEE International Conference on*, pages 417–422, Hong Kong, China, 2014. IEEE.

[11] Bahador Khaleghi, Alaa Khamis, Fakhreddine O. Karray, and Saiedeh N. Razavi. Multisensor data fusion: A review of the state-of-the-art. *Information Fusion*, 14(1):28–44, January 2013.

[12] David J. C. MacKay. *Information Theory, Inference, and Learning Algorithms*. Cambridge University Press, 2003.

[13] Michael Milford. Vision-based place recognition: how low can you go? *The International Journal of Robotics Research*, 32(7):766–789, 2013.

[14] Valérie Renaudin and Christophe Combettes. Magnetic, Acceleration Fields and Gyroscope Quaternion (MAGYQ)-Based Attitude Estimation with Smartphone Sensors for Indoor Pedestrian Navigation. *Sensors*, 14(12):22864–22890, 2014.

[15] Jan Steckel and Herbert Peremans. Biomimetic sonar for biomimetic SLAM. In *Sensors, 2012 IEEE*, IEEE Sensors, pages 1–4, Taipei, Taiwan, October 2012. IEEE.

[16] Sebastian Thrun, Wolfram Burgard, and Dieter Fox. *Probabilistic Robotics*. The MIT Press, Cambridge, 2006.

[17] Sebastian Thrun, Dieter Fox, Wolfram Burgard, and Frank Dellaert. Robust Monte Carlo localization for mobile robots. *Artificial Intelligence*, 128(1-2):99–141, May 2001.

[18] Joeri van Laere. Challenges for IF Performance Evaluation in Practice. In *Information Fusion (FUSION), 2009 12th International Conference on*, pages 866 – 873, Seattle, Washington, USA, 2009. IEEE.

[19] Dries Vandermeulen, Charles Vercauteren, and Maarten Weyn. Indoor localization Using a Magnetic Flux Density Map of a Building. In *The Third International Conference on Ambient Computing, Applications, Services and Technologies*, pages 42–49, Porto, 2013. IARIA.

[20] Maarten Weyn. *Opportunistic Seamless Localization*. PhD thesis, University of Antwerp, 2011.

Appendix A: case extra

Herbert proposed a subtle correction of the probabilistic Wi-Fi sensor model. This files tries to model the third case (extra) distribution.

$$P(C_3|p_j) = P(W^+_{a,p_j} = 0, w_{a,k,p_i} \neq 0|p_j)$$

$$P(C_3|p_j) = P(w_{a,k,p_i} \neq 0|p_j, W^+_{a,p_j} = 0)P(W^+_{a,p_j} = 0|p_j)$$

with 0 indicating empty set, rather than the value 0. MATLAB cannot produce the empty set symbol. We are interested in the first part of the formula.

$$P(w_{a,k,p_i}|p_j, W^+_{a,p_j} = 0) = \int_{-\infty}^{\infty} P(\overline{W}^+_{a,p_j}|p_j, W^+_{a,p_j} = 0) \left[P(w_{a,k,p_i}|p_j, \overline{W}^+_{a,p_j})\right] d\overline{W}^+_{a,p_j}$$

where:

$$P(\overline{W}^+_{a,p_j}|p_j, W^+_{a,p_j} = 0) = \alpha P(W^+_{a,p_j} = 0|p_j, \overline{W}^+_{a,p_j})$$

$$P(\overline{W}^+_{a,p_j}|p_j, W^+_{a,p_j} = 0) = \alpha \prod_i^N \frac{1}{\sqrt{2\pi}\sigma_w} \int_{-\infty}^{w_t} \exp - \frac{(w_i - \overline{W}^+_{a,p_j})^2}{2\sigma_w^2} dw_i$$

which is equal to

$$P(\overline{W}^+_{a,p_j}|p_j, W^+_{a,p_j} = 0) = \alpha \left[\int_{-\infty}^{w_t} \exp - \frac{(w_i - \overline{W}^+_{a,p_j})^2}{2\sigma_w^2} dw_i\right]^N$$

In both functions, N is the number of measurements used to calculate the template at p_j.

Contents

Initial values

Values are in dBm.

```
model_std = sqrt(12);
model_th = -89;
```

N is the number of measurements used to calculate the template at p_j.

```
N = 1:30;
```

We estimate the distribution over all possible RSS values. To make this feasible, we reduce the range of values to from twice the threshold until zero, in steps of a tenth.

```
template = 2*model_th:0.1:0;
measurement = template;
```

Step 1

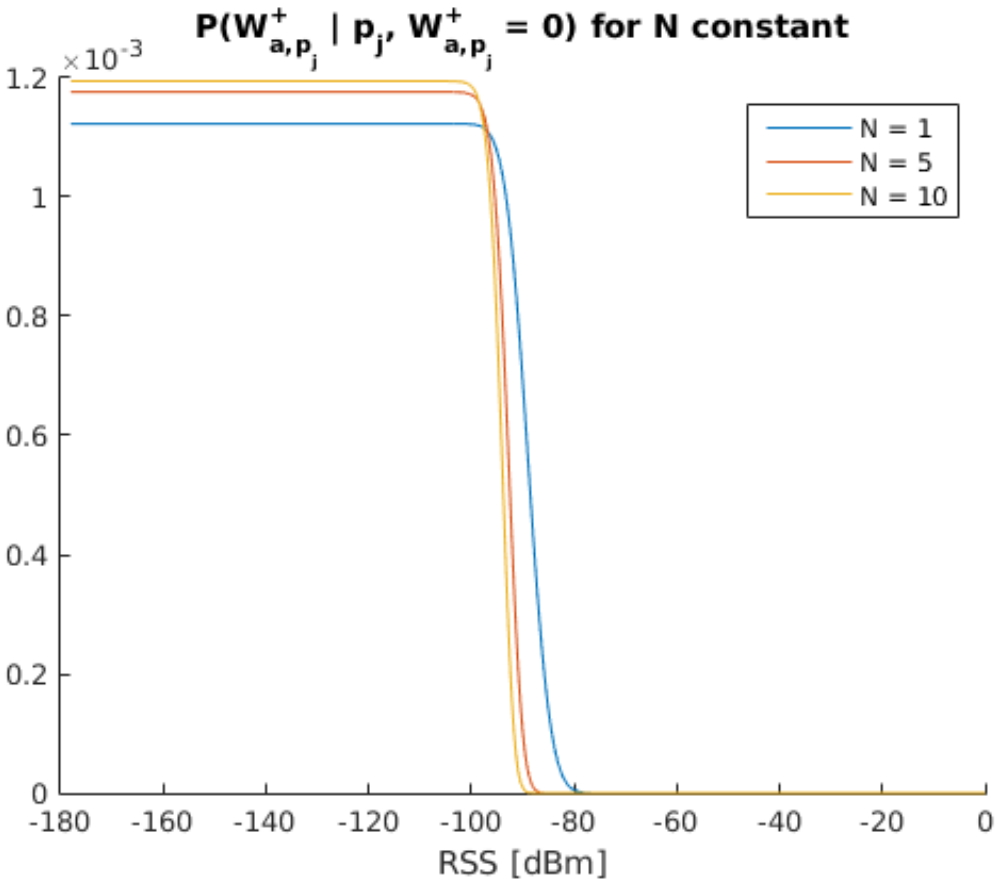
$$P(\overline{W}^{+}_{a,p_j}|p_j, W^{+}_{a,p_j} = 0) = \alpha \left[\int_{-\infty}^{w_t} \exp - \frac{(w_i - \overline{W}^{+}_{a,p_j})^2}{2\sigma_w^2} dw_i \right]^N$$

```
N = N(ones(size(template, 2), 1), :)';
likelihood = normcdf(model_th, template, model_std);
likelihood = likelihood(ones(size(N, 1), 1), :) .^ N;
alpha = sum(likelihood, 2);
p_template = likelihood ./ alpha(:, ones(size(likelihood, 2), 1));
```

Output

We see that $P(\overline{W}^{+}_{a,p_j}|p_j, W^{+}_{a,p_j} = 0)$ is uniformly distributed for values under the threshold. It is zero for values above the threshold. Around the threshold, the slope is influenced by the number of measurements. This reflects the behavior that if only one measurement was used to construct the template, we do not have much information about the RSS value at position p_j, so there is a chance that we simply missed the access point. The more measurements performed, the more we trust that the actual RSS value must be under the threshold.

```
figure
hold on
plot(template, p_template(1,:));
plot(template, p_template(5,:));
plot(template, p_template(10,:));
xlabel('RSS [dBm]');
title('P(W^+_{a,p_j} | p_j, W^+_{a,p_j} = 0) for N constant');
legend('N = 1','N = 5','N = 10');
```



Step 2

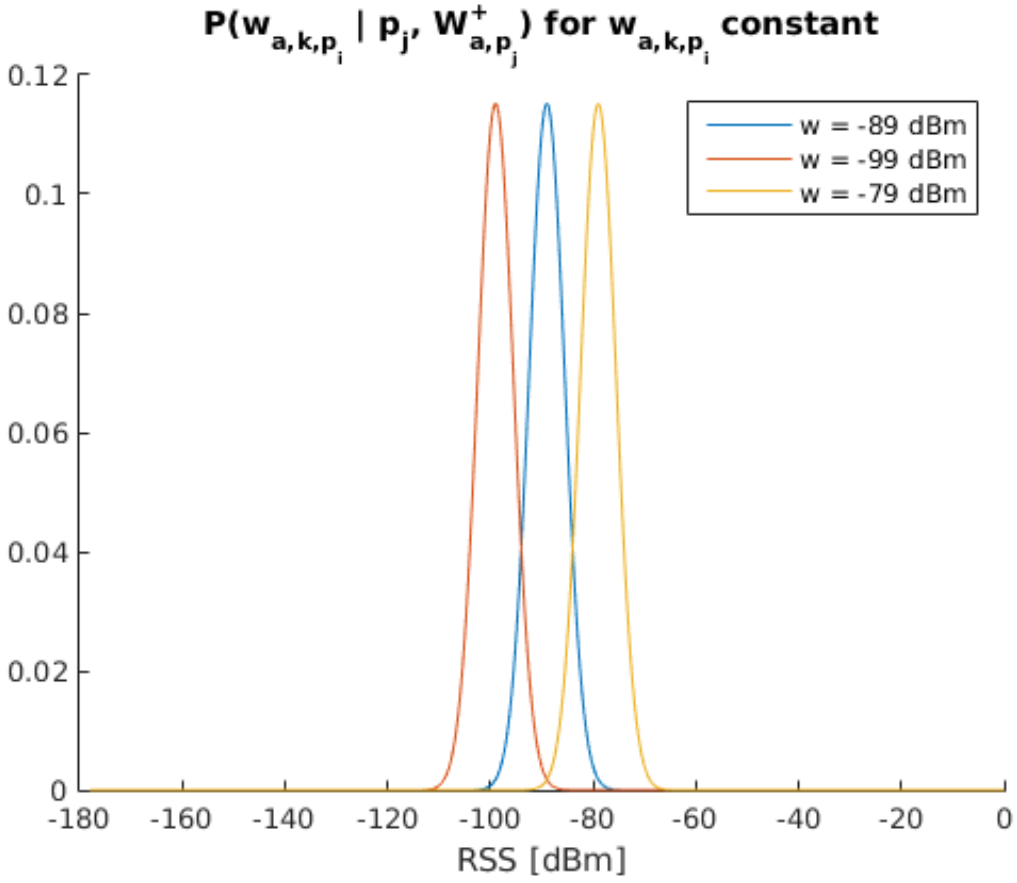
$P(w_{a,k,p_i}|p_j, \overline{W}^+_{a,p_j})$

```
p_measurement = normpdf(measurement(ones(size(measurement, 2), 1), :)', ...
    template(ones(size(template, 2), 1), :), model_std);
```

Output

This is the normal distribution used when we measure an access point that also has an RSS value in the template, i.e., a hit. It must also be calculated for RSS values under the threshold.

```
figure;
hold on
plot(measurement, p_measurement(:, measurement==-89));
plot(measurement, p_measurement(:, measurement==-99));
plot(measurement, p_measurement(:, measurement==-79));
xlabel('RSS [dBm]');
title('P(w_{a,k,p_i} | p_j, W^+_{a,p_j}) for w_{a,k,p_i} constant');
legend('w = -89 dBm','w = -99 dBm','w = -79 dBm');
```



$$P(w_{a,k,p_i}|p_j, W^+_{a,p_j} = 0) = \int_{-\infty}^{\infty} P(\overline{W}^+_{a,p_j}|p_j, W^+_{a,p_j} = 0) \left[P(w_{a,k,p_i}|p_j, \overline{W}^+_{a,p_j})\right] d\overline{W}^+_{a,p_j}$$

with the negative to positive infinity range reduced to from -178 to 0, with an interval of 0.1; using sum instead of integral.

```
p_wc3 = sum(...
    permute(p_measurement(:, :, ones(size(N, 1), 1)), [1 3 2]) .* ...
    permute(p_template(:, :, ones(size(template, 2), 1)), [2 1 3]));
p_wc3 = permute(p_wc3, [2 3 1]);
```

Normalize, because this is a distribution.

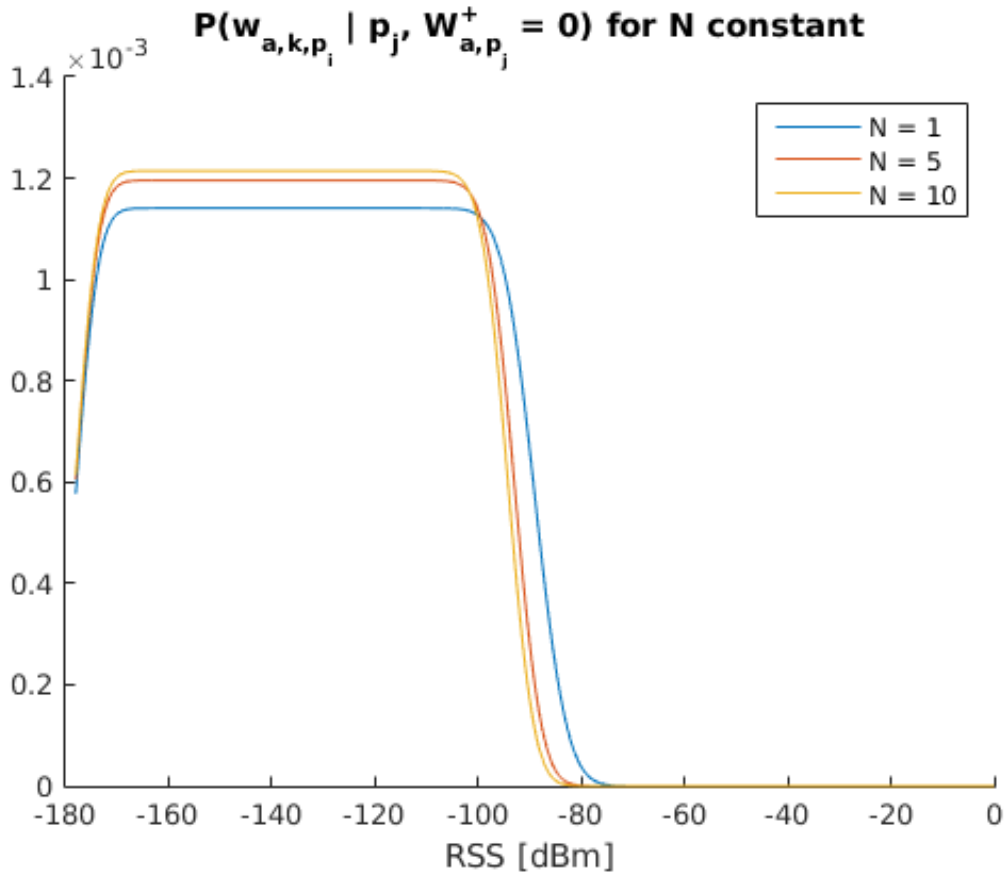
```
alpha = sum(p_wc3, 2);
p_wc3 = p_wc3 ./ alpha(:, ones(size(p_wc3, 2), 1));
```

Output

These outputs show $P(w_{a,k,p_i}|p_j, W^+_{a,p_j} = 0)$, the distribution that is important for the third case of the sensor model. The first plot looks a lot like the first plot in the document, but instead of being the chance for the true value of the template, it is the chance for a value of the measurement. Because it is also matched against the hypothetical value of the template, the slopes are less steep.

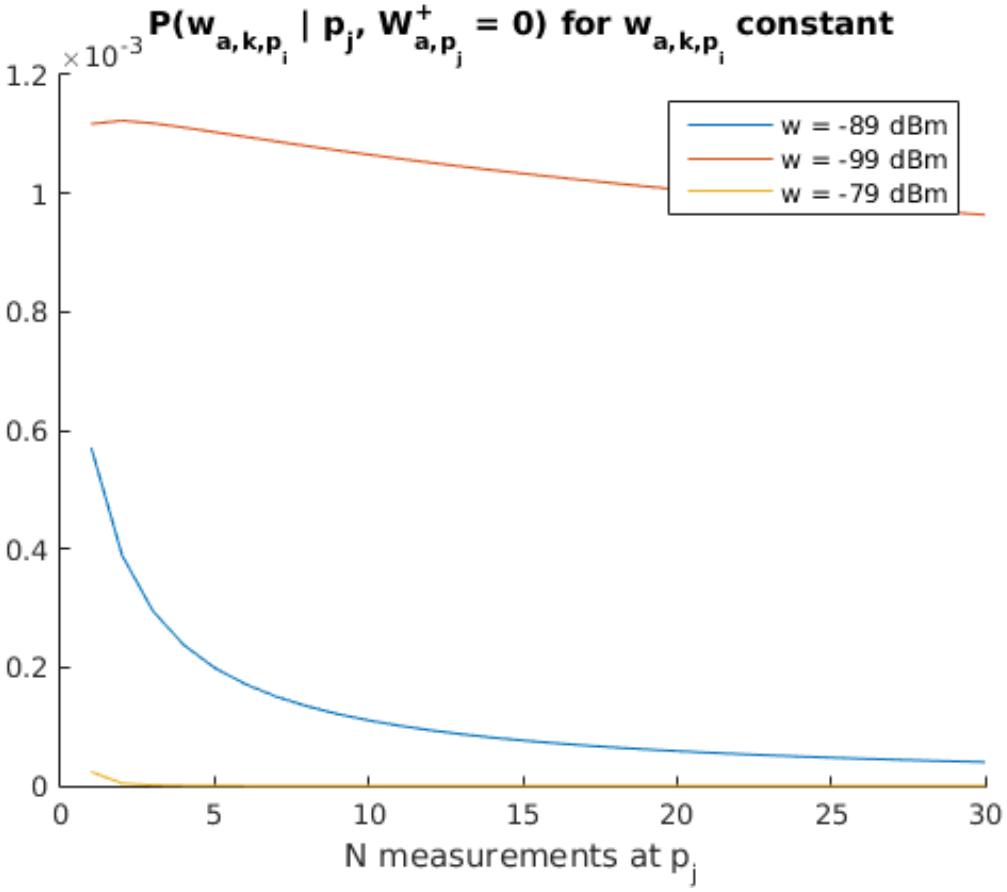
```
figure
hold on
plot(measurement, p_wc3(1,:));
plot(measurement, p_wc3(5,:));
plot(measurement, p_wc3(10,:));
```

```
xlabel('RSS [dBm]');
title('P(w_{a,k,p_i} | p_j, W^+_{a,p_j} = 0) for N constant');
legend('N = 1','N = 5','N = 10');
```



The second output shows the change in probability of a certain measured RSS value, for changing values of N. One can see that the probability is always higher for a measured value under the threshold, that it quickly descends for a measured value at the threshold and that it is always near zero for values above the threshold.

```
figure;
hold on
plot(1:30, p_wc3(:, measurement==-89));
plot(1:30, p_wc3(:, measurement==-99));
plot(1:30, p_wc3(:, measurement==-79));
xlabel('N measurements at p_j');
title('P(w_{a,k,p_i} | p_j, W^+_{a,p_j} = 0) for w_{a,k,p_i} constant');
legend('w = -89 dBm','w = -99 dBm','w = -79 dBm');
```



Published with MATLAB® R2015a

Appendix B: case none

Herbert proposed a subtle correction of the probabilistic Wi-Fi sensor model. This files tries to model the fourth case (none) distribution.

$$P(C_4|p_j) = P(W^+_{a,p_j} = 0, w_{a,k,p_i} = 0|p_j)$$

$$P(C_3|p_j) = P(w_{a,k,p_i} = 0|p_j, W^+_{a,p_j} = 0)P(W^+_{a,p_j} = 0|p_j)$$

with 0 indicating empty set, rather than the value 0. MATLAB cannot produce the empty set symbol. We are interested in the first part of the formula.

$$P(w_{a,k,p_i} = 0|p_j, W^+_{a,p_j} = 0) = \int_{-\infty}^{\infty} P(W^+_{a,p_j}|p_j, W^+_{a,p_j} = 0) \left[\int_{-\infty}^{w_t} P(w_{a,k,p_i} = 0|p_j, W^+_{a,p_j} = 0) dw \right] dW^+_{a,p_j}$$

where the second integral is only indicative, since it is implied by the chance it integrates.

Contents

Initial values

Values are in dBm.

```
model_std = sqrt(12);
model_th = -89;
```

N is the number of measurements used to calculate the template at p_j.

```
N = 1:30;
```

We estimate the distribution over all possible RSS values. To make this feasible, we reduce the range of values to from twice the threshold until zero, in steps of a tenth.

```
template = 2*model_th:0.1:0;
%measurement = template;
```

Step 1

$$P(\overline{W}^+_{a,p_j}|p_j, W^+_{a,p_j} = 0) = \alpha \left[\int_{-\infty}^{w_t} \exp - \frac{(w_i - \overline{W}^+_{a,p_j})^2}{2\sigma_w^2} dw_i \right]^N$$

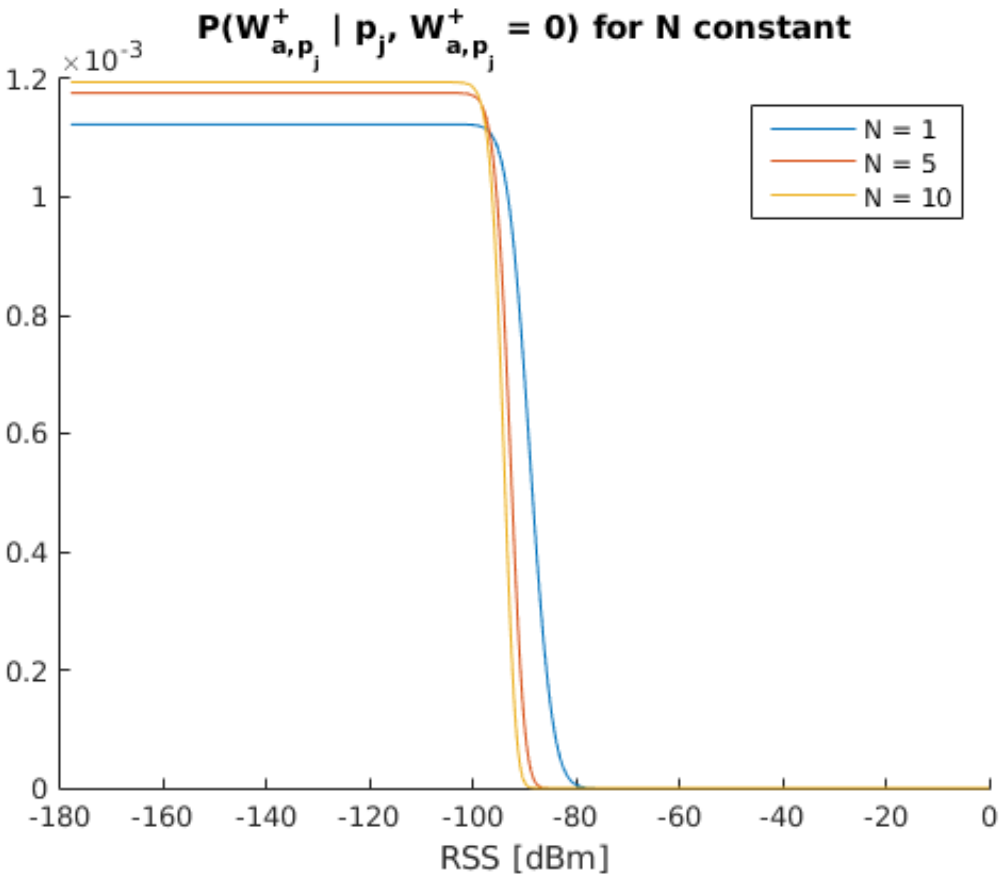
```
N = N(ones(size(template, 2), 1), :)';
likelihood = normcdf(model_th, template, model_std);
likelihood = likelihood(ones(size(N, 1), 1), :) .^ N;
alpha = sum(likelihood, 2);
```

```
p_template = likelihood ./ alpha(:, ones(size(likelihood, 2), 1));
```

Output

This is the same as with case 3.

```
figure
hold on
plot(template, p_template(1,:));
plot(template, p_template(5,:));
plot(template, p_template(10,:));
xlabel('RSS [dBm]');
title('P(W^+_{a,p_j} | p_j, W^+_{a,p_j} = 0) for N constant');
legend('N = 1','N = 5','N = 10');
```



Step 2

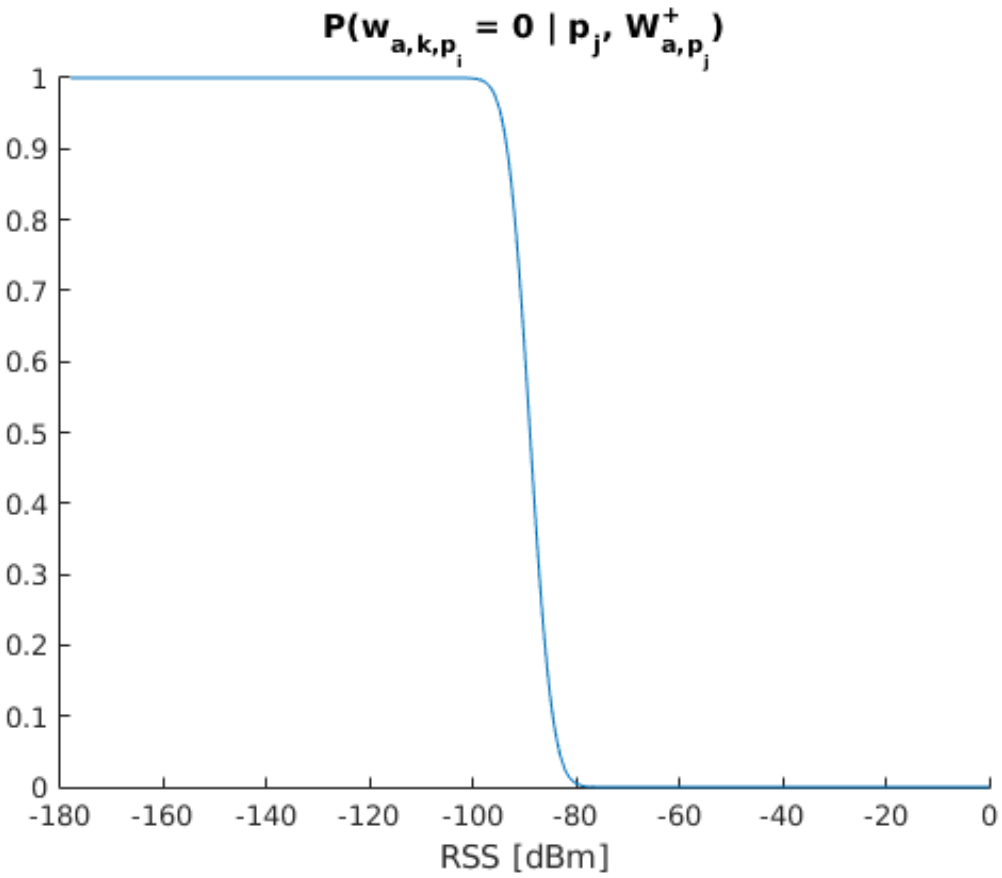
$$P(w_{a,k,p_i} = 0 | p_j, \overline{W}^+_{a,p_j}) = \int_{-\infty}^{w_t} \exp - \frac{(w_i - \overline{W}^+_{a,p_j})^2}{2\sigma_w^2} dw_i$$

```
p_measurement = normcdf(model_th(ones(size(template, 2), 1))', ...
    template, model_std);
```

Output

This is the value used when we do not measure an access point when we did find one in the template, i.e., a miss.

```
figure;
hold on
plot(template, p_measurement);
xlabel('RSS [dBm]');
title('P(w_{a,k,p_i} = 0 | p_j, W^+_{a,p_j})');
```



$$P(w_{a,k,p_i} = 0 | p_j, W^+_{a,p_j} = 0) = \int_{-\infty}^{\infty} P(\overline{W}^+_{a,p_j} | p_j, W^+_{a,p_j} = 0) \left[\int_{-\infty}^{w_t} \exp - \frac{(w_i - \overline{W}^+_{a,p_j})^2}{2\sigma_w^2} dw_i \right] d\overline{W}^+_{a,p_j}$$

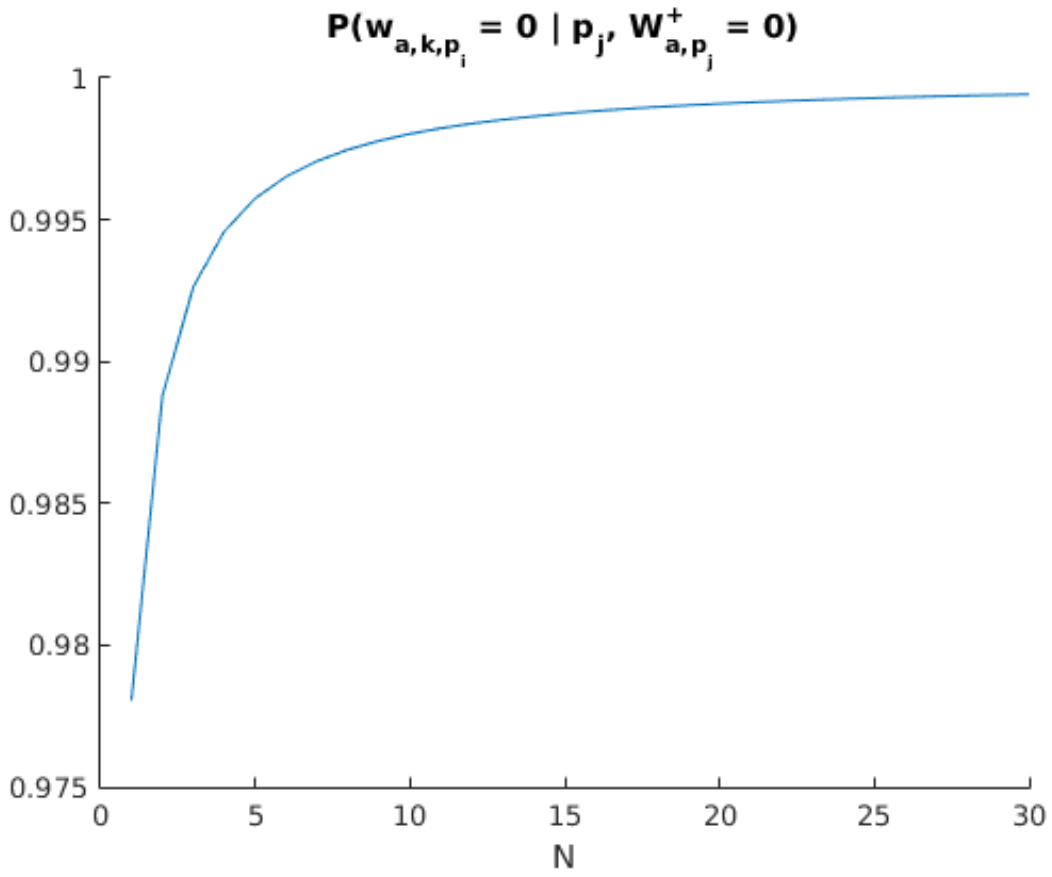
with the negative to positive infinity range reduced to from -178 to 0, with an interval of 0.1; using sum instead of integral.

```
p_wc4 = sum(p_measurement(ones(size(N, 1), 1), :) .*  p_template, 2);
```

Output

This shows the value of case 4 for changing numbers of measurements with which the template was built.

```
figure;
hold on
plot(p_wc4);
xlabel('N');
title('P(w_{a,k,p_i} = 0 | p_j, W^+_{a,p_j} = 0)');
```



Published with MATLAB® R2015a